Mohamed Khider University of Biskra

Faculty of Science and Technology

Department of Electrical Engineering

MASTER THESIS

Science and Technology
Electrical Engineering
Telecommunication

Submitted and Defended by:

Mr. Helela Nacer Eddine

On: 25/9/2020

Extracting The Information From A Satellite Image With Application OpenCV, NumPy, and SciPy.

Board of Examiners:

Dr. Abd Malek Ouamane	MCA	University of Biskra	President
Mr. Nacer Eddine Rahmani	MAA	University of Biskra	Examiner
Mr. Dhiabi Fethi	MCB	University of Biskra	Supervisor

2019-2020

Dedication

I dedicate this work to

The memory of my Father, Abdelhafid, who appreciated knowledge and urged me to gain it.

My lovely Mother who has been my source of inspiration and strength, and who continually provides me with moral, spiritual and emotional support

My Dear Sisters for their endless love, encouragement and support throughout my life and during the time of research.

My dear friends and colleagues.

Acknowledgment

I would like first to thank Allah the Almighty for having given me strength and enlightenment to accomplish this work.

I would like to express a sincere gratitude to my supervisor Mr. Dhiabi Fethi for his priceless support, constant guidance, and patience.

I would like to thank the board of examiners members Dr. Abd Malek Ouamane and MR.Nacer Eddine Rahmani for their precious effort in evaluating this work.

Finally, I would like to thank all my friends and colleagues wherever they are for the help they offered.

Abstract

Our work that was presented in this thesis aims to extract important information from satellite type imagery. Our purpose is to use this information to analyze it and process it into a useful data that is used multiple and multi-purpose applications and algorithms. In order to extract this information, we use a programming language called Python, because of its various function in the computer vision field thanks to its many libraries, which we talk about how we applied them in our work. In addition to its easiness to practice as we explained thoroughly how to use this amazing vast programming language. Afterwards we wrote a code to extract the information we selected, which they are: image gradient, histogram, color detection and contour extraction. We applied some of these information to detect the edges of the image and determine approximately the type of terrains shown in the earth surface of the satellite image.

يهدف عملنا الذي تم تقديمه في هذه الأطروحة إلى استخراج معلومات مهمة من الصور من نوع القمر الصناعي. هدفنا هو استخدام هذه المعلومات لتحليلها ومعالجتها في بيانات مفيدة تستخدم تطبيقات وخوارزميات متعددة ومتعددة الأغراض. من أجل استخراج هذه المعلومات، نستخدم لغة برمجة تسمى Python، بسبب وظيفتها المختلفة في مجال رؤية الكمبيوتر بفضل مكتباتها العديدة التي نتحدث عنها عن كيفية تطبيقها في عملنا. بالإضافة إلى سهولة ممارستها كما أوضحنا بدقة كيفية استخدام هذه اللغة البرمجية الهائلة والمذهلة. بعد ذلك كتبنا كود لاستخراج المعلومات التي اخترناها، وهي: تدرج الصورة، والمدرج التكراري، واكتشاف اللون، واستخراج الكفاف. قمنا بتطبيق بعض هذه المعلومات لاكتشاف حواف الصورة وتحديد نوع التضاريس تقريبًا الموضحة في سطح الأرض لصورة القمر الصناعي.

List of Abbreviations

RGB: Red, Green, and Blue

HSV: Hue, Saturation, and Value

List of Tables

List of Graphs

Table of Content

Chapter One: An Overview of Digital Image

Chapter Two: Python Programming Language.

Chapter Three: Introduction to Numpy, OpenCV and Scipy Libraries.

Chapter Four: Practical Part.

General Introduction

General Introduction

Pursuing information has been the humanity purpose since creation, finding answers give us reassurance and motivating to open new doors and find new invention every day, we even live in the information age. Information it is news or knowledge given or received concerning a particular fact or circumstances. It is gained through study, talking, or researching the subject matter.

Computer data is information processed or stored by a computer. This information may be in the form of text documents, images, audio clips, software programs, or other.

Image processing is a method to perform operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image.

Many applications are based on extracting information from images, like text information extraction from images is based on edge detection, and some segmentation algorithms are based on histogram extraction from images.

Applications on satellite imagery are numerous; a lot of them are based on extraction features or information from the image before starting the manipulation process, and some of them the information extraction is the role of application. Like region of interest detection application, which requires histogram based techniques such as Histogram classification, Histogram segmentation. Or edge detecting, satellite image's object is hardly detected because many objects are covered with cloud shadows in the sky. Edge detection has an important role in image analysis. Edge detection aims to extract the boundary of an object contained in the image. those application prove that extraction information from satellite imagery like gradient that are used in edge detection and histograms or contours are very important.

This memoire will be divided into 4 chapters:

- **Chapter 1:** first we are going to basics of images and imagery, and explain what is image processing and analysis. Then go into details on what is pixels and color spaces.
- **Chapter 2:** in this chapter we are going to talk about the programing language we are going to use in our project; which Python. We are going to explain why we chose it and the difference between it and other languages. Then we learn the basic of using it.

- **Chapter 3:** it speaks about the main python libraries that we used in this project; NumPy, SciPy, and OpenCV.
- **Chapter 4:** we conclude our memoire by representing our algorithm structure and explaining thoroughly how to extract some useful information from satellite imagery, and support it with some experimental results to prove its functionality.

Lastly, we finish our memoire with a general conclusion.

Chapter One:

An Overview of Digital Image

I.1. Introduction

Imaging has come a real long way in this new age of technology because of the big advancements that are happening in the field of computer vision.

For those of you who don’t know what computer vision is; it’s the field of study that allows computers to see and understand images and videos. In addition, it took great leaps over the years thanks to the advances of deep learning and artificial neural networks.

This chapter discusses what are images and imaging and its history and the many types of it

Also the extraction of information and manipulation images.

I.2. Image Basics

I.2.1. Digital Image

A digital image is a computer file generated by a scanner or digital camera. it is an electronic graphic representation of the object scanned or photographed with the digital camera.[1] The digital image itself is really a data structure within the computer, containing a number or code for each pixel in the image. This code determines the color of that pixel. Each pixel can be thought of as a discrete sample of a continuous real image. It is helpful to think about the common ways that a digital image is created. Some of the main ways are via a digital camera, a page or slide scanner, a 3D rendering program, or a paint or drawing package. The simplest process to understand is the one used by the digital camera [2].

I.2.2. History of The Digital Image

One of the earliest applications of digital image, in the early 1920s, can be seen in the newspaper industry. It was about the pictures that were sent by submarine cable between London and New York. The Bart lane cable picture transmission system reduced the amount of time tremendously weeks to hours across the Atlantic. As the field of digital image processing developed along with the development of the modern digital computers in 1950s, various techniques, methods, and technologies of digital image processing were developed in the 1960s at various places. Some of those places can be named as Bell Laboratories, the Jet Propulsion Laboratory, Massachusetts Institute of Technology, and University of Maryland. Together with them, there were also some other research facilities for satellite imagery, medical imaging, wirephoto standards conversion, photograph enhancement, videophone, and character recognition . In the early days, image processing was mainly meant for improving the image quality in general. Very basic and commonly used techniques in image processing included enhancement, restoration, encoding, and compression of images.

American Jet Propulsion Laboratory (JPL) happened to be the first successful application in 1960s. Using this, in 1964, Space Detector Ranger 7 sent thousands of lunar photos. They mainly used image processing techniques like geometric correction, gradation transformation, and noise removal on the sent lunar photos. It was a big success story to have the successful computerized mapping of the moon's surface. The success kept progressing so much so that spacecraft sent nearly 100,000 photos that were processed with more complex imaging functionalities. It helped to obtain the topographic map, color map, and panoramic mosaic of the moon. This resulted in extraordinary achievements and happened to be landmark basis of history for human landing on the moon.

This is true that, due to computing machines of 1960s and earlier, the processing cost was fairly high. With the passage of time, in the 1970s, however, things changed relatively with faster digital image processing and cheaper computing equipment. Slowly and gradually afterwards, processing power kept increasing together with lower cost machines which resulted in images to be processed faster and faster. So much so, various complex problems like television standards conversion were managed in real time. In the years 2000s and after, the general-purpose computing equipment became much faster. Various developments in the technological world led to dedicated and special purpose hardware and equipment. Today, digital image processing has turned to a vital computing discipline which is playing a significant role to solve various real life problems in real time. [3]

I.2.3. What Is a Pixel?

A pixel, or picture element is a physical point in a raster image, or the smallest addressable element in an all points addressable display device; so it is the smallest controllable element of a picture represented on the screen.

Each pixel is a sample of an original image; more samples typically provide more accurate representations of the original. The intensity of each pixel is variable. In color imaging systems, a color is typically represented by three or four component intensities such as red, green, and blue, or cyan, magenta, yellow, and black. [4]

In a digital image, all the coordinates on 2-d function and the corresponding values are finite. Each value available in every location is considered as a pixel. In other words, a pixel is the smallest part of an image. So a digital image can be thought as 2-d array of pixels.[5]

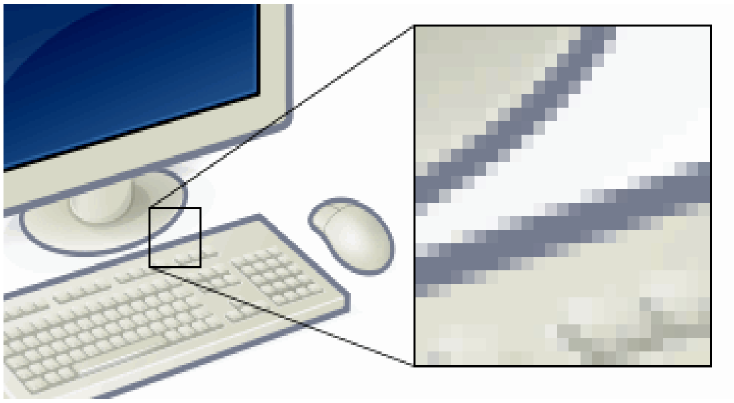


Figure I.1. Group of Pixel Square in Image Example [4].

I.2.4. Pixel Resolution or Image Resolution

Image resolution is the detail an image holds. The term applies to raster digital images, film images, and other types of images. Higher resolution means more image detail. [6]

Resolution refers to the number of pixels in an image. Resolution is sometimes identified by the width and height of the image as well as the total number of pixels in the image. [7]

I.2.5. Types of Images

I.2.5.1. Binary Image

Binary images are the simplest type of images and can take on two values, typically black and white, or 0 and 1. A binary image is referred to as a 1-bit image because it takes only 1 binary digit to represent each pixel. These types of images are frequently used in applications where the only information required is general shape or outline, for example optical character recognition. [8]

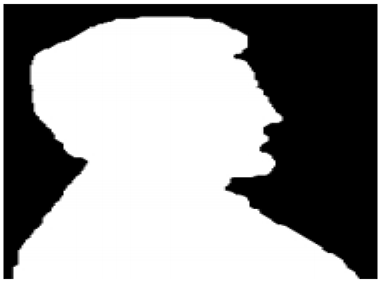


Figure I.2. Binary image [8]

I.2.5.2. Grayscale Image

Grayscale image has range of values from 0 to 255 i.e, each pixel location can have any value between 0 and 255. If you watch old films around the 1950s, you are watching grayscale images (films are nothing but videos which are collection of individual images in proper sequence).[5]

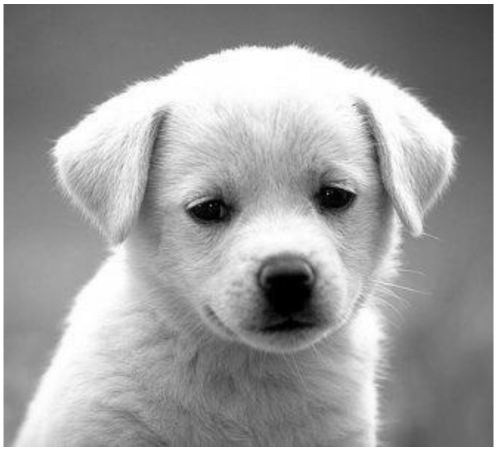


Figure I.3. Grayscale image (0–255 range) [5]

I.2.5.3. Color Image

Both binary image and grayscale image are 2-dimensional arrays, where at every location, you have one value to represent the pixel. Remember to represent a color image, we need more than one value for each pixel. But how many values we need to represent a color? Typically you need 3 values for each pixel to represent any color. This came from the idea that any color can be formed by combining 3 basic colors Red, Blue and Green. Ex: you get yellow by mixing red and green, violet can be formed by combining red and blue etc., This is actually called RGB color space. [5]

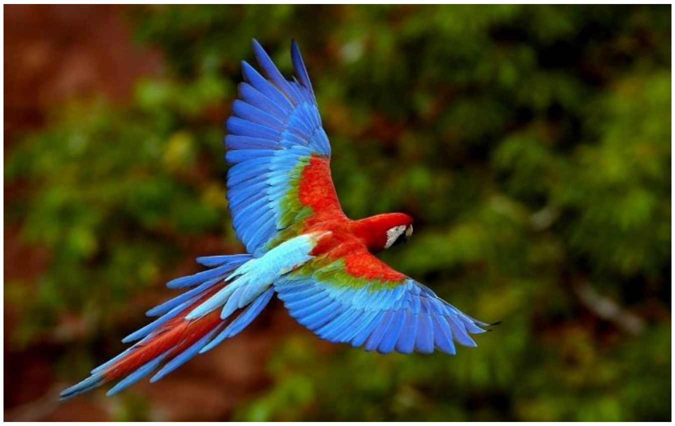


Figure I.4. Color Image (3-dim array with 3 values at every pixel location) [5]

I.3. How Do See Colors Digitally and Biologically

I.3.1. What Is Color?

Is the characteristic of visual perception described through color categories, with names such as red, orange, yellow, green, blue, or purple. This perception of color derives from the stimulation of photoreceptor cells by electromagnetic radiation. Color categories and physical specifications of color are associated with objects through the wavelengths of the light that is reflected from them and their intensities. This reflection is governed by the object's physical properties such as light absorption, emission spectra, etc. [9]

I.3.2. Color Vision

Light can vary in both wavelength and intensity. Color vision is the ability to make discriminations based on the wavelength composition of the light independent of its intensity. Color vision is distributed widely throughout the animal kingdom, and it appears to have evolved independently multiple times. This suggests that it serves important purposes. Color vision is used to determine the location and shapes of objects (e.g., fruit among foliage) and their identity and characteristics (e.g., what kind of fruit and whether it is ripe). It is particularly useful in cluttered natural scenes, where intensity variations may arise from either shadows or object borders.[10]

I.3.3. How Do We See Color?

When light hits an object, the object absorbs some of the light and reflects the rest of it. Which wavelengths are reflected or absorbed depends on the properties of the object. For a ripe banana, wavelengths of about 570 to 580 nanometers bounce back. These are the wavelengths of yellow light. When you look at a banana, the wavelengths of reflected light determine what color you see. The light waves reflect off the banana's peel and hit the light-sensitive retina at the back of your eye. That's where cones come in.

Cones are one type of photoreceptor, the tiny cells in the retina that respond to light. Most of us have 6 to 7 million cones, and almost all of them are concentrated on a 0.3 millimeter spot on the retina called the fovea centralis [11].

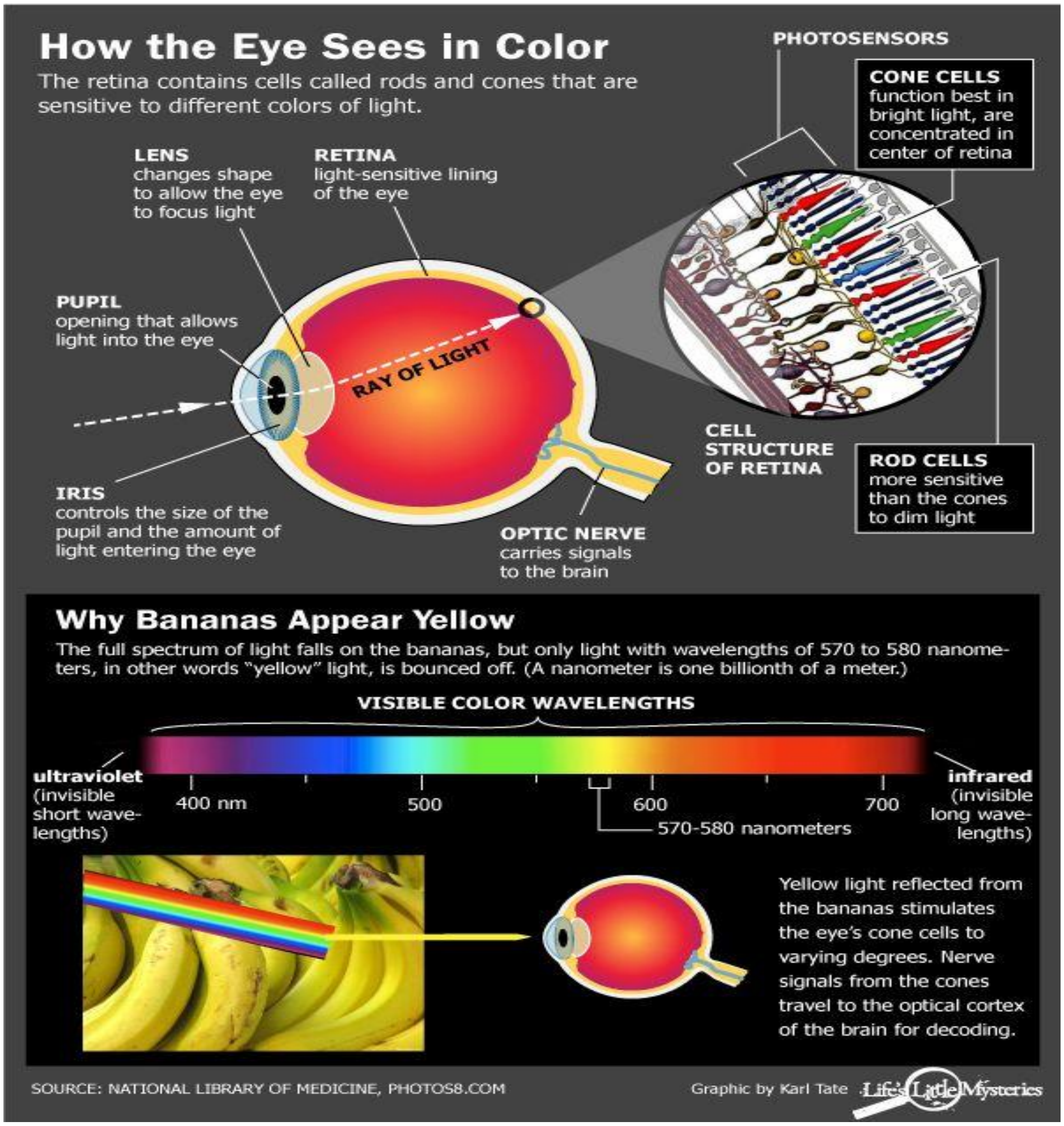

How the Eye Sees in Color
The retina contains cells called rods and cones that are sensitive to different colors of light.
PHOTOSENSORS
CONE CELLS function best in bright light, are concentrated in center of retina
LENS changes shape to allow the eye to focus light
RETINA light-sensitive lining of the eye
PUPIL opening that allows light into the eye
RAY OF LIGHT
IRIS controls the size of the pupil and the amount of light entering the eye
OPTIC NERVE carries signals to the brain
CELL STRUCTURE OF RETINA
ROD CELLS more sensitive than the cones to dim light
Why Bananas Appear Yellow
The full spectrum of light falls on the bananas, but only light with wavelengths of 570 to 580 nanometers, in other words "yellow" light, is bounced off. (A nanometer is one billionth of a meter.)
VISIBLE COLOR WAVELENGTHS
ultraviolet (invisible short wavelengths)
400 nm
500
600
700
570-580 nanometers
infrared (invisible long wavelengths)
Yellow light reflected from the bananas stimulates the eye's cone cells to varying degrees. Nerve signals from the cones travel to the optical cortex of the brain for decoding.
SOURCE: NATIONAL LIBRARY OF MEDICINE, PHOTOS8.COM
Graphic by Karl Tate
Life's Little Mysteries

Figure I.5. How Human Visualize Color. [11]

Not all of these cones are alike. About 64 percent of them respond most strongly to red light, while about a third are set off the most by green light. Another 2 percent respond strongest to blue light. When light from the banana hits the cones, it stimulates them to varying degrees. The resulting signal is zapped along the optic nerve to the visual cortex of the brain, which processes the information and returns with a color: yellow.

Humans, with our three cone types, are better at discerning color than most mammals, but plenty of animals beat us out in the color vision department. Many birds and fish have four types of cones, enabling them to see ultraviolet light, or light with wavelengths shorter than what the human eye can perceive.

Some insects can also see in ultraviolet, which may help them see patterns on flowers that are completely invisible to us. To a bumblebee, those roses may not be so red after all [11].

I.3.4. What Is The Visible Light Spectrum?

The visible spectrum is the portion of the electromagnetic spectrum that is visible to the human eye. Electromagnetic radiation in this range of wavelengths is called visible light or simply light. A typical human eye will respond to wavelengths from about 380 to 740 nanometers.[11] In terms of frequency, this corresponds to a band in the vicinity of 405–790 THz.

The spectrum does not contain all the colors that the human visual system can distinguish. Unsaturated colors such as pink, or purple variations like magenta, for example, are absent because they can only be made from a mix of multiple wavelengths. Colors containing only one wavelength are also called pure colors or spectral colors. [12]

I.3.5. The Wavelengths of Visible Light

Violet: 380–450 nm (688–789 THz frequency)

Blue: 450–495 nm

Green: 495–570 nm

Yellow: 570–590 nm

Orange: 590–620 nm

Red: 620–750 nm (400–484 THz frequency). [13]

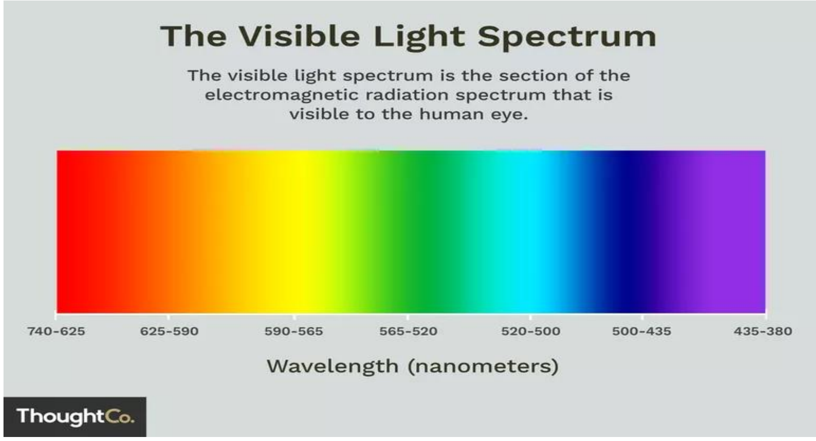

The Visible Light Spectrum
The visible light spectrum is the section of the electromagnetic radiation spectrum that is visible to the human eye.
740-625
625-590
590-565
565-520
520-500
500-435
435-380
Wavelength (nanometers)
ThoughtCo.

Figure I.6. The Visible Light Spectrum [14]

I.3.6. Pixel Values to Color Representation

Each of the pixels that represents an image stored inside a computer has a pixel value which describes how bright that pixel is, and/or what color it should be. In the simplest case of binary images, the pixel value is a 1-bit number indicating either foreground or background. For a grayscale images, the pixel value is a single number that represents the brightness of the pixel. The most common pixel format is the byte image, where this number is stored as an 8-bit integer giving a range of possible values from 0 to 255. Typically zero is taken to be black, and 255 is taken to be white. Values in between make up the different shades of gray.

To represent color images, separate red, green and blue components must be specified for each pixel (assuming an RGB colorspace), and so the pixel `value' is actually a vector of three numbers. Often the three different components are stored as three separate `grayscale' images known as color planes (one for each of red, green and blue), which have to be recombined when displaying or processing.

The actual grayscale or color component intensities for each pixel may not actually be stored explicitly. Often, all that is stored for each pixel is an index into a colormap in which the actual intensity or colors can be looked up. [15]

I.3.7. What Is Color Models?

A color model is a system that uses three primary colors to create a larger range of colors. There are different kinds of color models used for different purposes, and each has a slightly different range of colors they can produce. The whole range of colors that a specific type of color model produces is called a color space. All color results from how our eye processes light waves, but depending on the type of media, creating that color comes from different methods. [16]

I.3.8. Additive and Subtractive Color Models

I.3.8.1. RGB Color Model

Media that transmit light (such as television) use additive color mixing with primary colors of red, green, and blue, each of which stimulates one of the three types of the eye's color receptors with as little stimulation as possible of the other two. This is called "RGB" color space. Mixtures of light of these primary colors cover a large part of the human color space and thus produce a large part of human color experiences. This is why color television sets or color computer monitors need only produce mixtures of red, green and blue light. See Additive color.

Other primary colors could in principle be used, but with red, green and blue the largest portion of the human color space can be captured. Unfortunately there is no exact consensus as to what loci in the chromaticity diagram the red, green, and blue colors should have, so the same RGB values can give rise to slightly different colors on different screens. [17]

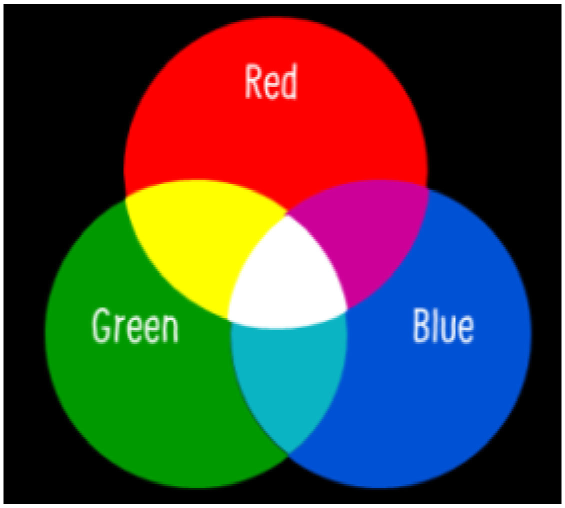


Figure I. 7.RGB Color Model [18]

I.3.8.2. CMYK Color Mode

It is possible to achieve a large range of colors seen by humans by combining cyan, magenta, and yellow transparent dyes/inks on a white substrate. These are the subtractive primary colors. Often a fourth ink, black, is added to improve reproduction of some dark colors. This is called the "CMY" or "CMYK" color space.

The cyan ink absorbs red light but transmits green and blue, the magenta ink absorbs green light but transmits red and blue, and the yellow ink absorbs blue light but transmits red and green. The white substrate reflects the transmitted light back to the viewer. Because in practice the CMY inks suitable for printing also reflect a little bit of color, making a deep and neutral black impossible, the K (black ink) component, usually printed last, is needed to compensate for their deficiencies. Use of a separate black ink is also economically driven when a lot of black content is expected, e.g. in text media, to reduce simultaneous use of the three colored inks. The dyes used in traditional color photographic prints and slides are much more perfectly transparent, so a K component is normally not needed or used in those media. [17]

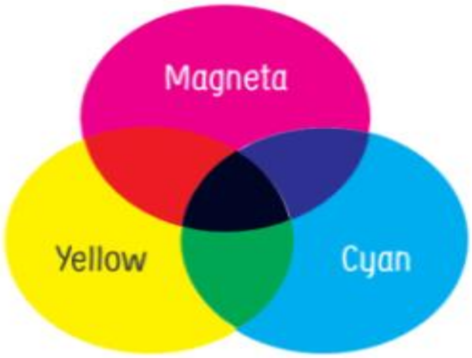


Figure I.8. CMYK color model [18]

I.4. Image Analysis and Image Processing

I.4.1. What Is Image Analysis?

Image analysis is the extraction of meaningful information from images; mainly from digital images by means of digital image processing techniques. Image analysis tasks can be as simple as reading bar coded tags or as sophisticated as identifying a person from their face.

Computers are indispensable for the analysis of large amounts of data, for tasks that require complex computation, or for the extraction of quantitative information. On the other hand, the human visual cortex is an excellent image analysis apparatus, especially for extracting higher-level information, and for many applications — including medicine, security, and remote sensing — human analysts still cannot be replaced by computers. For this reason, many important image analysis tools such as edge detectors and neural networks are inspired by human visual perception models. [19]

I.4.2. What Is Image Processing?

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too. [20]

I.4.3. Difference between Image Processing and Image Analysis

Image analysis methods extract information from an image by using automatic or semiautomatic techniques termed: scene analysis, image description, image understanding, pattern recognition, computer/machine vision etc.). Image analysis differs from other types of image processing methods, such as enhancement or restoration in that the final result of image analysis procedures is a numerical output rather than a picture.[21]

I.5. Contour Extraction

I.5.1. What Is a Contour?

A contour is a closed curve joining all the continuous points having some color or intensity; they represent the shapes of objects found in an image. Contour detection is a useful technique for shape analysis and object detection and recognition.[22]

I.5.2. Contour Extraction Objective

Why would we want to extract the contour of a given pattern? Contour tracing is one of many preprocessing techniques performed on digital images in order to extract information about their general shape. Once the contour of a given pattern is extracted, it's different characteristics will be examined and used as features which will later on be used in pattern classification. Therefore, correct extraction of the contour will produce more accurate features which will increase the chances of correctly classifying a given pattern.

Nevertheless, you might be wondering: Why waste precious computational time on first extracting the contour of a pattern and then collecting its features? Why not collect features directly from the pattern?

Well, the contour pixels are generally a small subset of the total number of pixels representing a pattern. Therefore, the amount of computation is greatly reduced when we run feature extracting algorithms on the contour instead of on the whole pattern. Since the contour shares a lot of features with the original pattern, the feature extraction process becomes much more efficient when performed on the contour rather on the original pattern.

In conclusion, contour tracing is often a major contributor to the efficiency of the feature extraction process -an essential process in the field of pattern recognition.[23]

I.6. Filters in Image Processing

In order to explain filtering correctly we need to understand what noise is.

I.6.1. What Is Noise?

Noise is a random variation of image Intensity and visible as a part of grains in the image. It may cause to arise in the image as effects of basic physics-like photon nature of light or thermal energy of heat inside the image sensors. It may produce at the time of capturing or image transmission. Noise means, the pixels in the image show different intensity values instead of true pixel values that are obtained from image. [24]

I.6.2. What Is a Filter?

In image processing filters are mainly used to suppress either the high frequencies in the image, i.e. smoothing the image, or the low frequencies, i.e. enhancing or detecting edges in the image. An image can be filtered either in the frequency or in the spatial domain.

The first involves transforming the image into the frequency domain, multiplying it with the frequency filter function and re-transforming the result into the spatial domain. The filter function is shaped so as to attenuate some frequencies and enhance others.[25]

Applying filters to the image is another way to modify image. In addition, the difference compare to point operation is the filter use more than one pixel to generate a new pixel value. For example, smoothing filter which replace a pixel value by average of its neighboring pixel value. Filters can divided in 2 types, linear filter and non-linear filter.[26]

I.6.3. Types of Filters

I.6.3.1. Linear Filter

Linear filter is a filter which operate the pixel value in the support region in linear manner (i.e.,as weighted summation). The support region is specified by the 'filter matrix' and be represent as H(i,j). The size of H is call 'filter region' and filter matrix has its own coordinate system, i is column index and j is row index. The center of it is the origin location and it is called the 'hot spot'.[26]

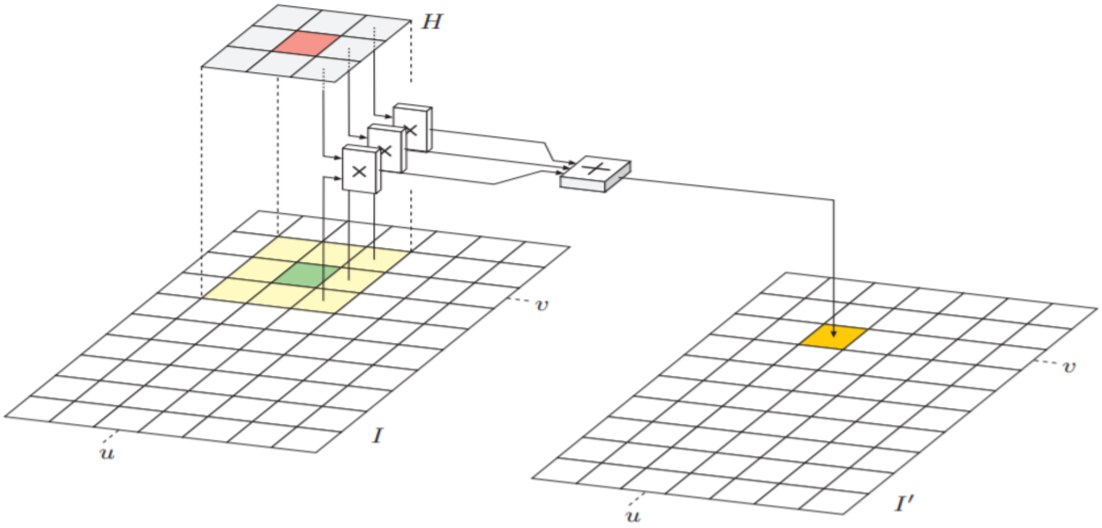


Figure I.9. Example of How Linear Filter Works [26]

I.6.3.2. Non-linear filter

A nonlinear (or non-linear) filter is a filter whose output is not a linear function of its input. In short, the input values produce output values that are not linear translations. It may be as simple as an exponential increase, as in a squared, or cubed output. Nonlinear filters are useful in image processing and convolutional neural nets.[27]

Noise removing with smoothing filter (a linear filter) provide the result in burred of the image structure, line and edge. Non-Linear Filters were used to solve this problem and it works in non-linear manner.[26]

I.6.4. Image Gradient

An image gradient is a directional change in the intensity or color in an image. The gradient of the image is one of the fundamental building blocks in image processing. For example, the Canny edge detector uses image gradient for edge detection. In graphics software for digital image editing, the term gradient or color gradient is also used for a gradual blend of color which can be considered as an even gradation from low to high values, as used from white to black in the images to the right. Another name for this is color progression.[28]

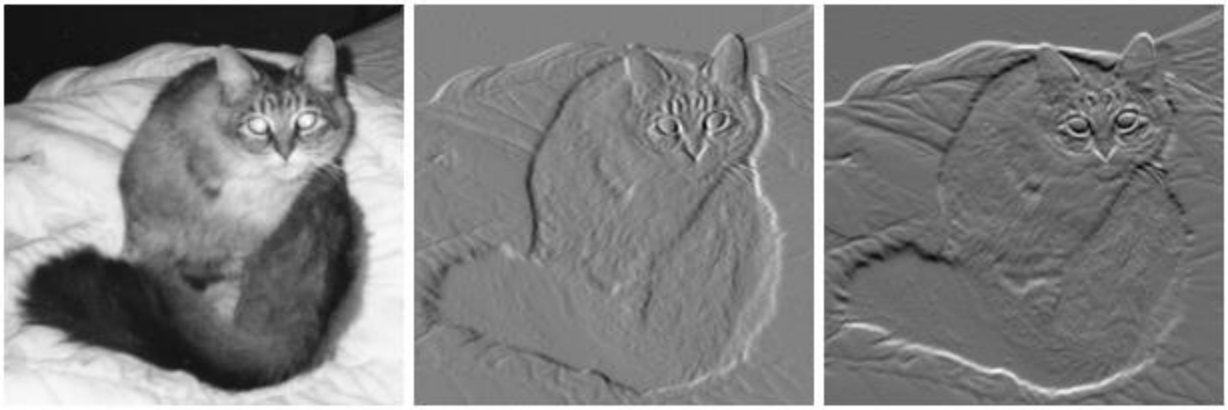


Figure I.10. On The Left, An Intensity Image Of A Cat. In The Center, A Gradient Image In The X Direction Measuring Horizontal Change In Intensity. On The Right, A Gradient Image In The Y Direction Measuring Vertical Change In Intensity. Gray Pixels Have A Small Gradient; Black Or White Pixels Have A Large Gradient. [28]

Image gradients can be used to extract information from images. Gradient images are created from the original image (generally by convolving with a filter, one of the simplest being the Sobel filter) for this purpose. Each pixel of a gradient image measures the change in intensity of that same point in the original image, in a given direction. To get the full range of direction, gradient images in the x and y directions are computed.[28]

I.6.5. Laplacian Filters

A Laplacian filter is an edge detector used to compute the second derivatives of an image, measuring the rate at which the first derivatives change. This determines if a change in adjacent pixel values is from an edge or continuous progression.

Laplacian filter kernels usually contain negative values in a cross pattern, centered within the array. The corners are either zero or positive values. The center value can be either negative or positive. The following array is an example of a 3x3 kernel for a Laplacian filter.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$ [29]

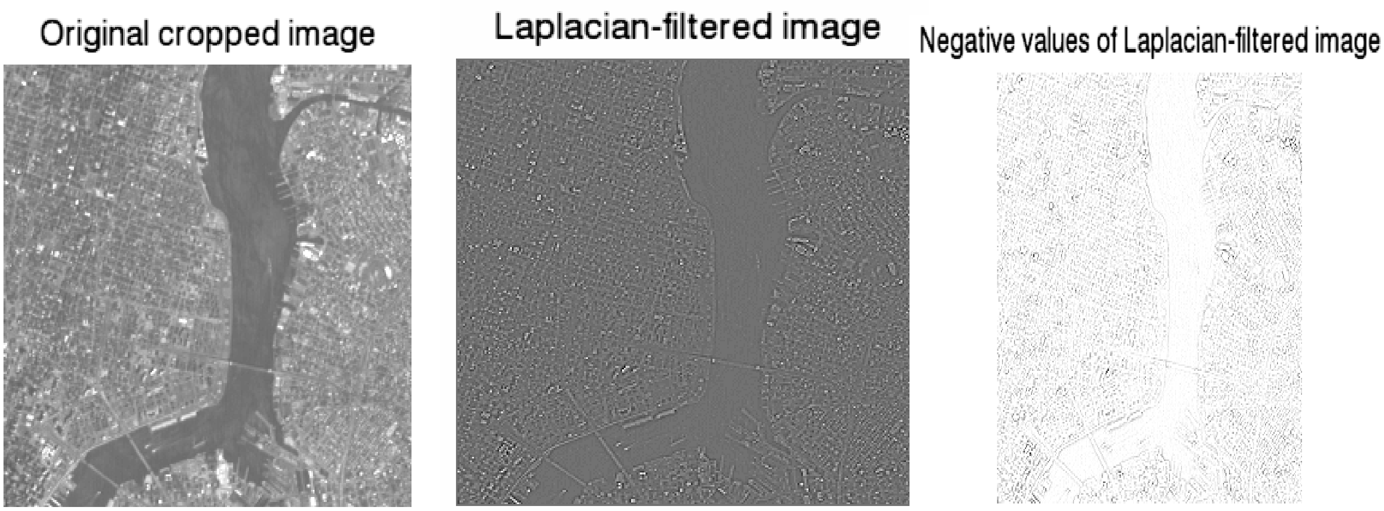


Figure I.11. Applied Laplacian Filter On A Cropped Image.[29]

I.7. Image Segmentation

I.7.1. What Is Image Segmentation?

Image segmentation is a critical process in computer vision. It involves dividing a visual input into segments to simplify image analysis. Segments represent objects or parts of objects, and comprise sets of pixels, or “super-pixels”. Image segmentation sorts pixels into larger components, eliminating the need to consider individual pixels as units of observation.[30]

The goal of segmentation is to simplify and/or to change the representation of an image into something that is more meaningful and easier to analyze. In areas like the medical imaging, satellite imaging, and security purposes, image segmentation plays a vital role in better understanding about any particular section. [31]

There are three levels of image analysis:

- **Classification** – categorizing the entire image into a class such as “people”, “animals”, “outdoors”
- **Object detection** – detecting objects within an image and drawing a rectangle around them, for example, a person or a sheep.
- **Segmentation** – identifying parts of the image and understanding what object they belong to. Segmentation lays the basis for performing object detection and classification.[30]

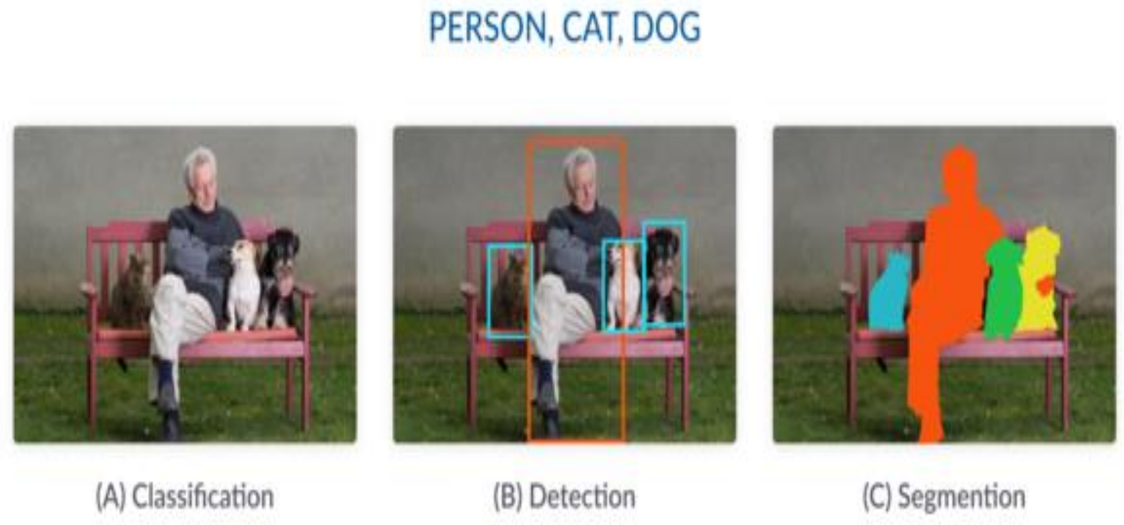


Figure I.12. Levels of Image Analysis [30]

I.7.2. Semantic Segmentation vs. Instance Segmentation

Within the segmentation process itself, there are two levels of granularity:

I.7.2.1. Semantic Segmentation

Semantic Segmentation classifies all the pixels of an image into meaningful classes of objects. These classes are “semantically interpretable” and correspond to real-world categories. For instance, you could isolate all the pixels associated with a cat and color them green. This is also known as dense prediction because it predicts the meaning of each pixel.[30]

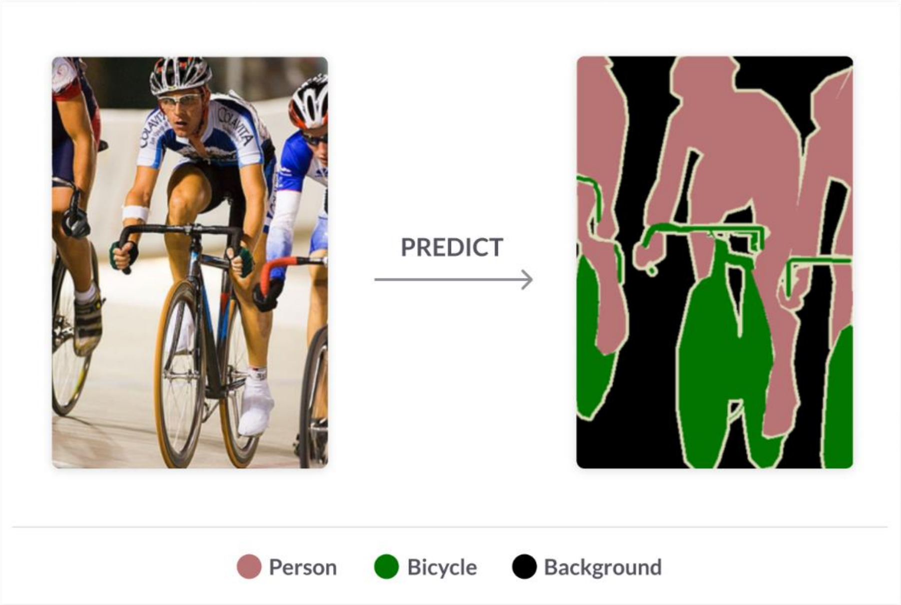


Figure I.13. Semantic Segmentation [30]

I.7.2.2. Instance Segmentation:

Instance Segmentation identifies each instance of each object in an image. It differs from semantic segmentation in that it does not categorize every pixel. If there are three cars in an image, semantic segmentation classifies all the cars as one instance, while instance segmentation identifies each individual car. [30]

Conclusion

Image processing and analyzing have a big important role computer vision and our everyday lives. Classification and detecting objects and visual recognition, these methods are the basic tools in many fields that uses image processing or computer vision. for example : medical sciences and self-driving cars and satellite imaging analysis

In our next chapter we would be talking about the python coding program and taking closer look at its tools

Chapter Two:

Python Programming Language.

II.1. Introduction

Python is an object-oriented, high-level programming language with integrated dynamic semantics primarily for web and app development. It is extremely attractive in the field of Rapid Application Development because it offers dynamic typing and dynamic binding options.

Python is relatively simple, so it's easy to learn since it requires a unique syntax that focuses on readability. Developers can read and translate Python code much easier than other languages. In turn, this reduces the cost of program maintenance and development because it allows teams to work collaboratively without significant language and experience barriers.

Additionally, Python supports the use of modules and packages, which means that programs can be designed in a modular style and code can be reused across a variety of projects. Once you've developed a module or package you need, it can be scaled for use in other projects, and it's easy to import or export these modules.[32]

II.2. History

The history of the Python programming language dates back to the late 1980s. Python was conceived in the late 1980s and its implementation was started in December 1989 by Guido.

Van Rossum [33] as a member of the National Research Institute of Mathematics and Computer Science. Initially, it was designed as a response to the ABC programming language that was also foregrounded in the Netherlands. Among the main features of Python compared to the ABC language was that Python had exception handling and was targeted for the Amoeba operating system.[34]

Python, like other languages, has gone through a number of versions. Python 0.9.0 was first released in 1991. In addition to exception handling, Python included classes, lists, and strings. More importantly, it included lambda, map, filter and reduce (JavaScript anyone?), which aligned it heavily in relation to functional programming.

In 2000, Python 2.0 was released. This version of was more of an open-source project from members of the National Research Institute of Mathematics and Computer Science. This version of Python included list comprehensions, a full garbage collector, and it supported Unicode.

Python 3.0 was the next version and was released in December of 2008 (the latest version of Python is 3.6.4). Although Python 2 and 3 are similar there are subtle differences. Perhaps most noticeably is the way the print statement works, as in Python 3.0 the print statement has been replaced with a print () function.[34]

II.3. Advantages/Benefits of Python

The diverse application of the Python language is a result of the combination of features, which give this language an edge over others. Some of the benefits of programming in Python include:

- **Presence of Third Party Modules**

The Python Package Index (PyPI) contains numerous third-party modules that make Python capable of interacting with most of the other languages and platforms.

- **Extensive Support Libraries**

Python provides a large standard library which includes areas like internet protocols, string operations, web services tools and operating system interfaces. Many high use programming tasks have already been scripted into the standard library which reduces length of code to be written significantly.

- **Open Source and Community Development**

Python language is developed under an OSI-approved open source license, which makes it free to use and distribute, including for commercial purposes.

Further, its development is driven by the community, which collaborates for its code through hosting conferences and mailing lists, and provides for its numerous modules.

- **Learning Ease and Support Available**

Python offers excellent readability and uncluttered simple-to-learn syntax which helps beginners to utilize this programming language. The code style guidelines, PEP 8, provide a set of rules to facilitate the formatting of code. Additionally, the wide base of users and active developers has resulted in a rich internet resource bank to encourage development and the continued adoption of the language.

- **User-friendly Data Structures**

Python has built-in list and dictionary data structures which can be used to construct fast runtime data structures. Further, Python also provides the option of dynamic high-level data typing which reduces the length of support code that is needed.

- **Productivity and Speed**

Python has clean object-oriented design, provides enhanced process control capabilities, and possesses strong integration and text processing capabilities and its own unit testing framework, all of which contribute to the increase in its speed and productivity. Python is considered a viable option for building complex multi-protocol network applications.[35]

II.4. C vs Python Comparison Table

Below is the top comparison between C vs Python: [36]

C	Python
C is mainly used for hardware related applications.	Python is general purpose programming language.
Follows an imperative programming model.	Follows object-oriented programming language
Pointers available in C.	No pointers functionality available.
C is compiled.	Python is interpreted.
A limited number of built-in functions.	Large library of built-in functions.
Code execution is faster than python.	Slower compared to C as python has garbage collection.
Implementing data structures required its functions to be explicitly implemented.	Gives ease of implementing data structures with built-in insert, append functions.
It is compulsory to declare the variable type in C.	No need to declare a type of variable.
C program syntax is harder than python.	Python programs are easier to learn, write and read.

Table II.1. Comparison between Programing Languages ‘C’ and Python. [36]

II.5.The Differences between Python 2 and Python 3

There is some differences to using different versions of python, some of them we collected inside this table: [37]

Basis of comparison	Python 2	Python 3
Release Date	2000	2008
Function print	print "hello"	print ("hello")
Division of Integers	When two integers are divided, you always provide integer value.	Whenever two integers are divided, you get a float value
Unicode	The syntax of Python 2 was comparatively difficult to understand.	The syntax is simpler and easily understandable.
Rules of ordering Comparisons	Rules of ordering comparison are very complex.	In this version, Rules of ordering comparisons have been simplified.
Iteration	In Python 2, the xrange () is used for iterations.	The new Range () function introduced to perform iterations.
Exceptions	It should be enclosed in notations.	It should be enclosed in parenthesis.
Leak of variables	The value of the global variable will change while using it inside for-loop.	The value of variables never changes.
Backward compatibility	Python version 3 is not backwardly compatible with Python 2.	Not difficult to port python 2 to python 3 but it is never reliable.
Library	Many older libraries created for Python 2 is not forward-compatible.	Many recent developers are creating libraries which you can only use with Python 3.

Table II.2. The Differences between Python 2 and Python 3. [37]

II.6. Python Data Structures

Data structures are a way of organizing and storing data so that they can be accessed and worked with efficiently. They define the relationship between the data, and the operations that can be performed on the data.

Data structures are actually an implementation of Abstract Data Types or ADT. This implementation requires a physical view of data using some collection of programming constructs and basic data types.[38]

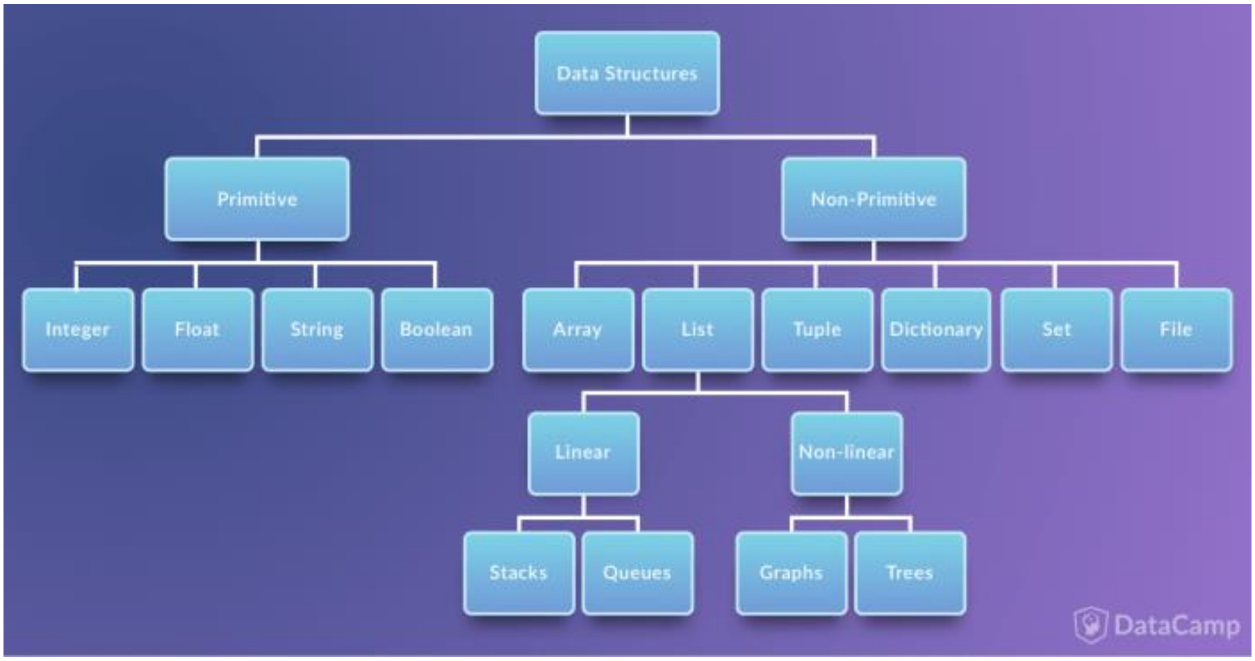


Figure II.14. Data Structures. [38]

Data structures help you to focus on the bigger picture rather than getting lost in the details. This is known as data abstraction.

Generally, data structures can be divided into two categories in computer science: primitive and non-primitive data structures. The former are the simplest forms of representing data, whereas the latter are more advanced: they contain the primitive data structures within more complex data structures for special purposes. [38]

II.6.1. Primitive Data Structures

These are primitive or the basic data structures. They are the foundation of data manipulation and contain pure, simple values of a data. Python has four primitive variable types:

II.6.1.1. Integers

You can use an integer represent numeric data, and more specifically, whole numbers from negative infinity to infinity, like 4, 5, or -1.

II.6.1.2. Float

"Float" stands for 'floating point number'. You can use it for rational numbers, usually ending with a decimal figure, such as 1.11 or 3.14.

II.6.1.3. String

Strings are collections of alphabets, words or other characters. In Python, you can create strings by enclosing a sequence of characters within a pair of single or double quotes. For example: 'cake', "cookie", etc.

II.6.1.4. Boolean

The built-in data type that can take up the values: True and False, which often makes them interchangeable with the integers 1 and 0. Booleans are useful in conditional and comparison. [39]

II.6.2. Non-Primitive Data Structures

Non-primitive types are the sophisticated members of the data structure family. They do not just store a value, but rather a collection of values in various formats. The non-primitive data structures are divided into:

II.6.2.1. Array

Arrays can be seen as a more efficient way of storing a certain kind of list. This type of list has elements of the same data type, though.

II.6.2.2. List

Lists in Python are used to store collection of heterogeneous items. These are mutable, which means that you can change their content without changing their identity. You can recognize lists by their square brackets [and] that hold elements, separated by a comma ,. Lists are built into Python: you do not need to invoke them separately.

II.6.2.3. Tuples

Tuples are another standard sequence data type. The difference between tuples and list is that tuples are immutable, which means once defined you cannot delete, add or edit any values inside it. This might be useful in situations where you might to pass the control to someone else but you do not want them to manipulate data in your collection, but rather maybe just see them or perform operations separately in a copy of the data.

II.6.2.4. Dictionary

Dictionaries are exactly what you need if you want to implement something similar to a telephone book. None of the data structures that you have seen before are suitable for a telephone book.

II.6.2.5. Sets

Sets are a collection of distinct (unique) objects. These are useful to create lists that only hold unique values in the dataset. It is an unordered collection but a mutable one, this is very helpful when going through a huge dataset.

II.6.2.6. Files

Files are traditionally a part of data structures. And although big data is commonplace in the data science industry, a programming language without the capability to store and retrieve previously stored information would hardly be useful. You still have to make use of the all the data sitting in files across databases and you will learn how to do this.[38]

II.7. Methods of Declaring Variables in Python

As we seen in the last title, all values in python has a data type. In order to store those values, we use “variables”.

II.7.1. What is a Variable in Python?

A Python variable is a reserved memory location to store values. In other words, a variable in a python program gives data to the computer for processing. [40]

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables. [41]

II.7.2. How To Declare a Variable

Python is completely object oriented, and not "statically typed". You do not need to declare variables before using them, or declare their type. Every variable in Python is an object.[42]

II.7.3. Basic Input, Output Syntax in Python

II.7.3.1 Reading Input from the Keyboard

Programs often need to obtain data from the user, usually by way of input from the keyboard. The simplest way to accomplish this in Python is with : “input().”

Input () pauses program execution to allow the user to type in a line of input from the keyboard.[43]

II.7.3.2. Writing Output to the Console

In addition to obtaining data from the user, a program will also usually need to present data back to the user. You can display program data to the console in Python with : “ print().”

Print () separates each object by a single space and appends a newline to the end of the output. [43]

II.7.4. Syntax of Declaring Variables

II.7.4.1. Numbers

Python supports two types of numbers - integers and floating point numbers. (It also supports complex numbers, which will not be explained in this tutorial).

To define an integer, use the following syntax:

-Script:

```
Variable = 7
Print (variable)
```

-Result:

```
7
```

II.7.4.2. Strings

Strings are defined either with a single quote or with a double quote.

-Script:

```
mystring = 'hello'
Print (mystring)
```

-Result:

```
Hello
```

-Script:

```
1-mystring = "hello"
2-print(mystring)
```

-Result:

```
Hello
```

II.8. Python Basic Operators

II.8.1. Python Arithmetic Operators [44]

Operator	Description
+ Addition	Adds values on either side of the operator.
- Subtraction	Subtracts right hand operand from left hand operand.
* Multiplication	Multiplies values on either side of the operator
/ Division	Divides left hand operand by right hand operand
% Modulus	Divides left hand operand by right hand operand and returns remainder
** Exponent	Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed. But if one of the operands is negative, the result is floored, i.e., rounded away from zero (towards negative infinity) −

Table II.3.Python Arithmetic Operators. [44]

II.8.1. Python Comparison Operators

These operators compare the values on either sides of them and decide the relation among them. They are also called Relational operators. [44]

Operator	Description
==	If the values of two operands are equal, then the condition becomes true.
!=	If values of two operands are not equal, then condition becomes true.
<>	If values of two operands are not equal, then condition becomes true.
>	If the value of left operand is greater than the value of right operand, then condition becomes true.
<	If the value of left operand is less than the value of right operand, then condition becomes true.
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.

Table II.4.Python Comparison Operators. [44]

II .9. Python Control Structures

A control structure (or flow of control) is a block of programming that analyses variables and chooses a direction in which to go based on given parameters. In simple sentence, a control structure is just a decision that the computer makes. Therefore, it is the basic decision-making process in programming and flow of control determines how a computer program will respond when given certain conditions and parameters.[45]

Flow of control through any given program is implemented with three basic types of control structures: Sequential, Selection and Repetition. [45]

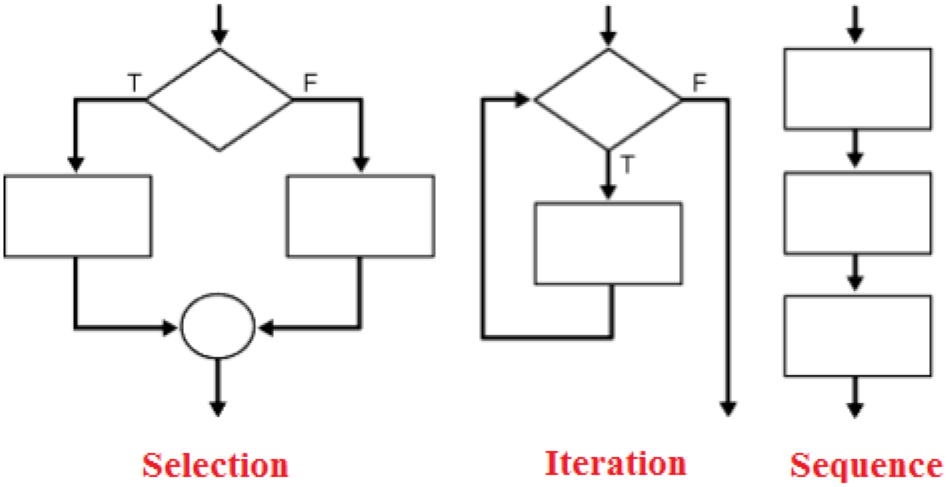


Figure II.15. Diagram of Types of Data Control Structures [45]

II .9.1. Sequential

Sequential execution is when statements are executed one after another in order. You don't need to do anything more for this to happen.[45]

II .9.2. Selection

Selection used for decisions, branching - choosing between 2 or more alternative paths.

If or if...else. [45]

II .9.3. Repetition

Repetition used for looping, i.e. repeating a piece of code multiple times in a row.

While loop or for loop. [45]

II.10. Python Decision Making and Conditional Statements

Decision-making is anticipation of conditions occurring while execution of the program and specifying actions taken according to the conditions.

Decision structures evaluate multiple expressions which produce TRUE or FALSE as outcome. You need to determine which action to take and which statements to execute if outcome is TRUE or FALSE otherwise. [46]

II.10. 1.Python if Statements

If statement evaluates the test expression inside parenthesis. If test expression is evaluated to true (nonzero) , statements inside the body of if is executed. If test expression is evaluated to false (0) , statements inside the body of if is skipped.[47]

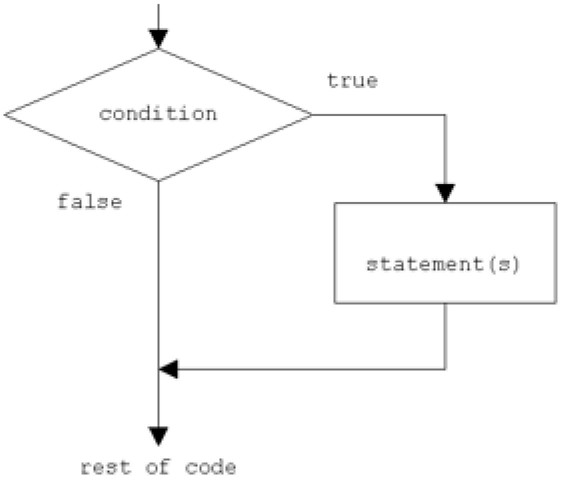


Figure II.16. Diagram of Python If Statement. [|47]

<u>Syntax</u> :

Input

```
x=20

y=10

if x > y :

  print(" X is bigger ")
```

Output

```
X is bigger
```

II.10.2. Python if..else Statements:

The else statement is to specify a block of code to be executed, if the condition in the if statement is false. Thus, the else clause ensures that a sequence of statements is executed.[47]

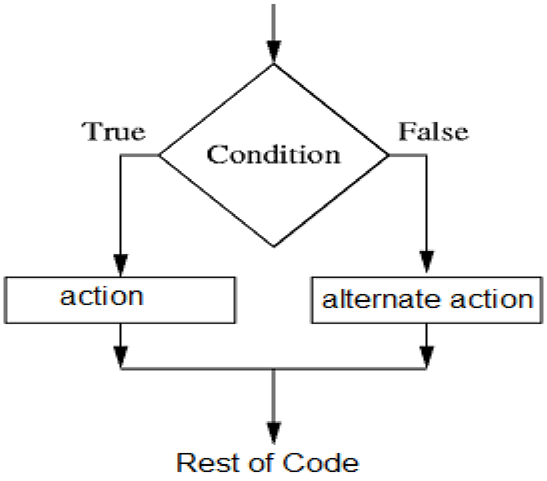


Figure II.17. Diagram of Python if..else Statements [47]

Syntax:

Input

```
x=10
y=20
if x > y :
  print(" X is bigger ")
else :
  print(" Y is bigger ")
```

Output

```
Y is bigger
```

II.11. Python Loop Statements

Loops are one of the most important features in computer programming languages . As the name suggests is the process that get repeated again and again . It offer a quick and easy way to do something repeated until a certain condition is reached.

Every loop has 3 parts:

- Initialization
- Condition
- Updation [48]

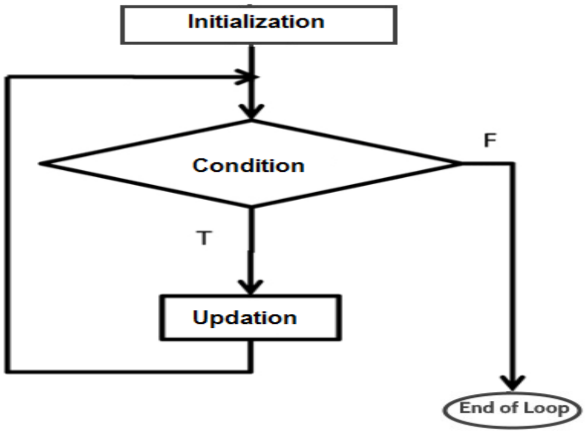


Figure II.18. Diagram of Python Loop Statement [48]

II.11.1.Python While Loop

While loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. That means, while loop tells the computer to do something as long as the condition is met. It consists of condition/expression and a block of code. The condition/expression is evaluated, and if the condition/expression is true, the code within the block is executed. This repeats until the condition/expression becomes false. [48]

Syntax:

Input

```
x=0
while(x < =4):
  print(x)
  x+=1
```

Output

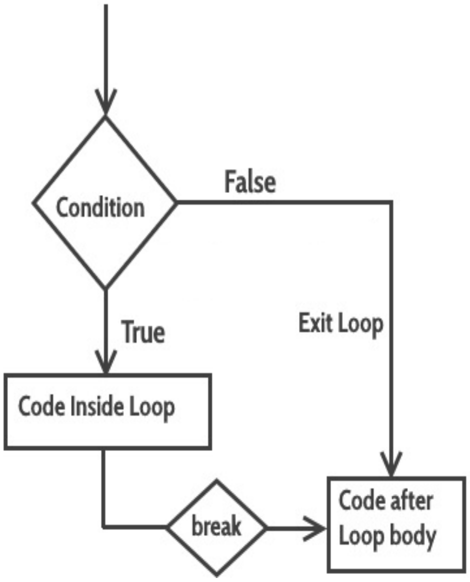
```
0
1
2
3
4
```

II.11.2. Python while Loop: Break and Continue

Python provides two keywords that terminate a loop iteration prematurely: break and continue.

- Break Leaves A Loop.
- Continue Jumps To The Next Iteration.

The continue statement in Python while loop is used when we want to skip one or more statements in loop's body and to transfer the control to the next iteration. But Sometimes it's necessary to exit from a Python while loop before the loop has finished fully iterating over all the step values. This is typically achieve them by a "break" or "continue” statement. [48]



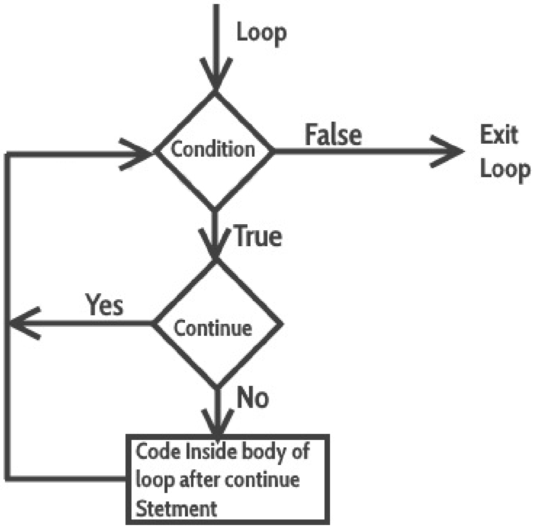


Figure II.20. Diagram of Python While Loop Break [48]

Figure II.19. Diagram of Python While Loop Continue[48]

II.11.3.Python for Loop

For loop has the ability to iterate over the items of any sequence, such as a list or a string.

It has two sections: a header specifying the iterating conditions, and a body which is executed once per iteration.

The header often declares an explicit loop counter or loop variable, which allows the body to know which iteration is being executed. [49]

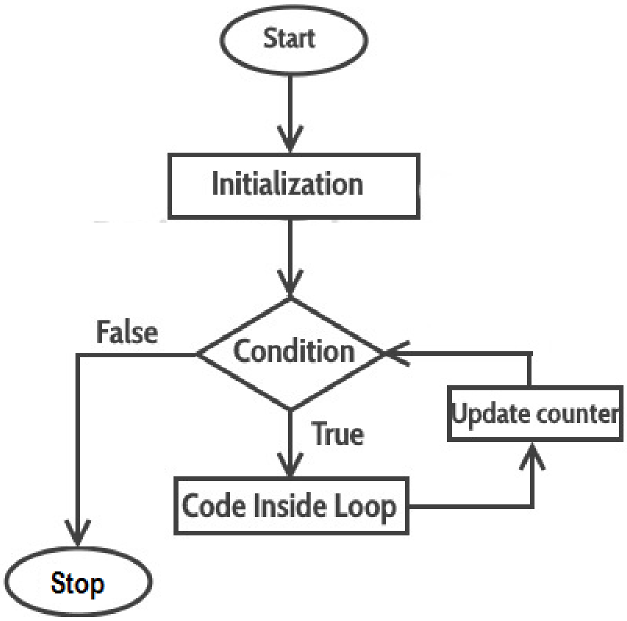


Figure II.21. Diagram of Python for Loop [49]

II.12. Python Functions

Functions are one of the fundamental building blocks in Python programming language. Functions are used when you have a block of statements that needs to be executed multiple times within the program. Rather than writing the block of statements repeatedly to perform the action, you can use a function to perform that action. This block of statements are grouped together and is given a name which can be used to invoke it from other parts of the program. [33]

II.12.1. The Importance Of Functions

Functions are block codes that were created for specific reasons, one of them is to avoid redundancy inside your code, and shorten it because there is no more repeated lines from of the same use, also the code becomes easily readable and more understandable for other readers, but the most important thing is the re-usability of the code without having to rewrite it again and again. [50]

II.12.2. Types of Functions

There are two types of functions in Python:

1. Built-in functions: These functions are pre-defined in Python and we need not to declare these functions before calling them. We can freely invoke them as and when needed.

2. User defined functions: The functions, which we create in our code, are user-defined functions. The add() function that we have created in above examples is a user-defined function.[51]

II.12.3. Syntax of Functions in Python

-Function Declaration:

```
def function_name(function_parameters):
                                        function_body # Set of Python statements
                                        return # optional return statement
```

-Calling the Function:

```
# when function doesn't return anything
function_name(parameters) [51]
```

Conclusion

In this chapter we took a closer look at python programming language: Structure, difference with C language, difference betwine V.2 and V.3 , type of variables, condition and loop statements and function. We understood it uses and learn how to practice it. And we find out its advantages and why would pick it over some other programming languages

In the next chapter we are going to talk about python packages. Specially 'Scipy and Numpy' and their uses in our thesis.

.

Chapter Three:

Introduction to Numpy, OpenCV and Scipy Libraries

III.1. Introduction

Python programming language is a real powerful language because it’s such an easy and versatile language to learn for beginners or experts. Python’s rose to greatness among other languages thanks to its readability that allows you to focus on your work other than wasting time worrying about confusing syntax.

In this chap we will get deeper in this fascinating language and discover its modules and packaging and Libraries, specially ‘NumPy and SciPy’ and a lot more that we will be needing to complete our work.

III.2. Getting Familiar with Modules, Packaging, and Libraries

III.2.1. What Are the Modules?

In programming, a module is a piece of software that has a specific functionality. For example, when building a ping pong game, one module would be responsible for the game logic, and another module would be responsible for drawing the game on the screen. Each module is a different file, which can be edited separately.[52]

In other words, a module is a file consisting of Python code. It can define functions, classes and variables. A module can also include runnable code.[53]

III.2.1.1 Python Modules Types

There are two modules types:

A. Built-in Modules

The Python interpreter has a number of built-in functions. They are loaded automatically as the interpreter starts and are always available.[54]

Built-in modules are written in C and integrated with the Python interpreter. Each built-in module contains resources for certain system-specific functionalities such as OS management, disk IO, etc. [54]

B. User-Defined Modules

Python files that are imported in to the top-level file, or among each other, and provide separate functionalities. These files are usually not launched directly from your command prompt, and are custom-made for the purpose of the project.[55]

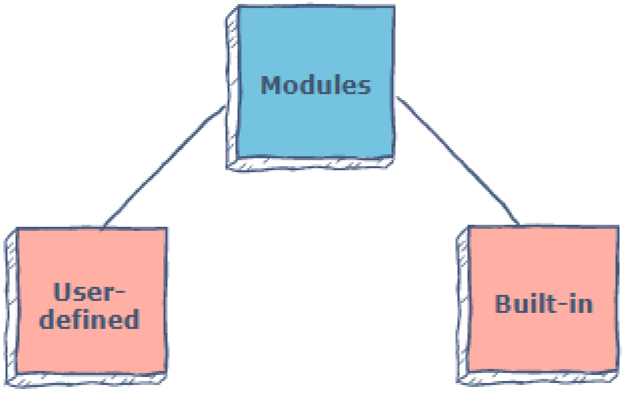


Figure III. 22. Types of Modules [56]

III.2.1.2. Benefits of Modules in Python

The big idea behind creating and integrating modules inside the program was to give the codes more structure which make it logical organized into one python file and makes development easier and less error-prone also clearer and simpler to understand and use.

The other facture is reusability, Functionality defined in a single module can be easily reused by other parts of the application. This eliminates the need to recreate duplicate code.[56]

III.2.1.3. How to Create and Import A Module?

A. Create

Creating a module in python is similar to writing a simple python script using the .py extension. let’s try to make a module for various calculation operations.

<u>**Syntax:**</u>

```
def add(x,y):
    return x + y

def sub(x, y):
   return x – y

def prod(x, y):
  return x * y
```

```

def div(x, y):
    return x / y
```

Save the above code in a file Calc.py. This is how we create a module in python. We have created different functions in this module. [57]

B. Importing

We will use the **import** keyword to incorporate the module into our program.

For instance, say we have our file with a name **main.py.**

Syntax:

```
import calc as a
a = 10
b = 20

addition = a.add(a,b)
print(addition)
```

As shown in the code above, we have created an alias using the as keyword. The output of the above code will be the addition of the two numbers a and b using the logic specified in the add function in the calc.py module.[57]

III.2.2. What Are Packages?

A python package is a collection of modules. Modules that are related to each other are mainly put in the same package. When a module from an external package is required in a program, that package can be imported and its modules can be put to use.

A package is a directory of Python modules that contains an additional **__init__.py** file, which distinguishes a package from a directory that is supposed to contain multiple Python scripts. Packages can be nested to multiple depths if each corresponding directory contains its own **__init__.py** file. [58]

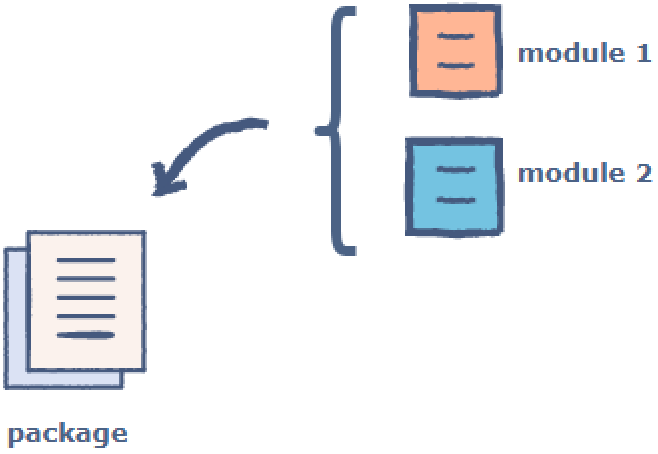


Figure III.23. Illustration of Package [58]

III.2.2.1. Structure of A Package

As said before, a package may hold other Python packages and modules. But what distinguishes a package from a regular directory? Well, a Python package must have an **__init__.py** file in the directory. You may leave it empty, or you may store initialization code in it. But if your directory does not have an **__init__.py** file, it isn't a package; it is just a directory with a bunch of Python scripts.[59]

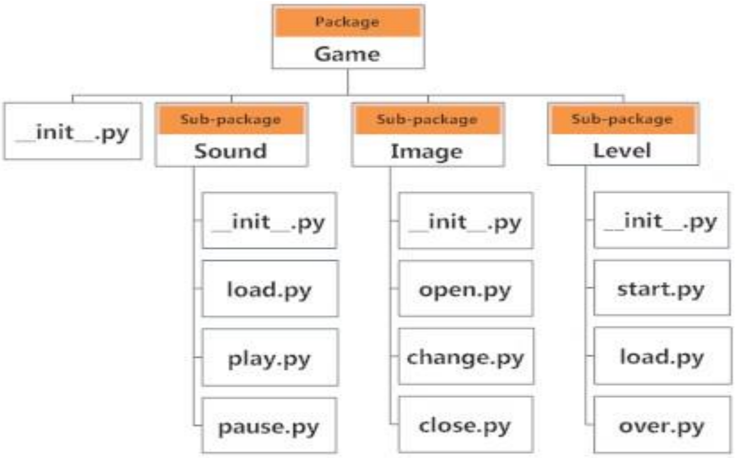


Figure III.24. Structure for a Game Package [60]

The main package has subpackages, sound image and level. Each with its own modules for example sound, it has the load play and pause modules apart of the __init__.py file.

III.2.2.2. Create a Python Package

You are working on a project and reach a certain point when you have a large number of python modules, you do not just want but you need to start organizing your work or it will fall apart. Therefore, you put them in packages,

Working with python packages is really simple, these few steps is all you need:

A. Create The Package Directory

Create a directory and give it a name for example animals

B. Add Classes or Modules

Now, we create the two classes for our package.

First, create a file named Mammals.py inside the Animals directory and put the following code in it:

```
class Mammals:
    def __init__(self):
        ''' Constructor for this class. '''
        # Create some member animals
        self.members = ['Tiger', 'Elephant', 'Wild Cat']


    def printMembers(self):
        print('Printing members of the Mammals class')
        for member in self.members:
            print('\t%s ' % member)
```

The code is pretty much self-explanatory! The class has a property named **members** – which is a list of some mammals we might be interested in. It also has a method named **printMembers**, which simply prints the list of mammals of this class! Remember, when you create a Python package, all classes must be capable of being imported, and won't be executed directly.[61]

Now we create another class but change the name and the members of class as we in the last example, we name them birds and the members are Sparrow, Robin, and Duck.

C. Create and add the __init__.py file in the directory

Finally, we create a file named __init__.py inside the Animals directory and put the following code in it: [61]

```
from Mammals import Mammals
from Birds import Birds
```

III.2.2.3.Differences between Python Modules and Packages

So, now that we have revised both modules and packages, let’s see how they differ:

A module is a file containing Python code. A package, however, is like a directory that holds sub-packages and modules. In addition, a package must hold the file __init__.py. This does not apply to modules. And to To import everything from a module, we use the wildcard . But this does not work with packages.[62]

III.2.3. What is A Python library?

A Python library is a reusable chunk of code that you may want to include in your programs/ projects. Compared to languages like C++ or C, a Python libraries do not pertain to any specific context in Python. Here, a ‘library’ loosely describes a collection of core modules. Essentially, then, a library is a collection of modules. A package is a library that can be installed using a package manager.[63]

III.2.3.1. The Python Standard Library

This library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python’s standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming.[64]

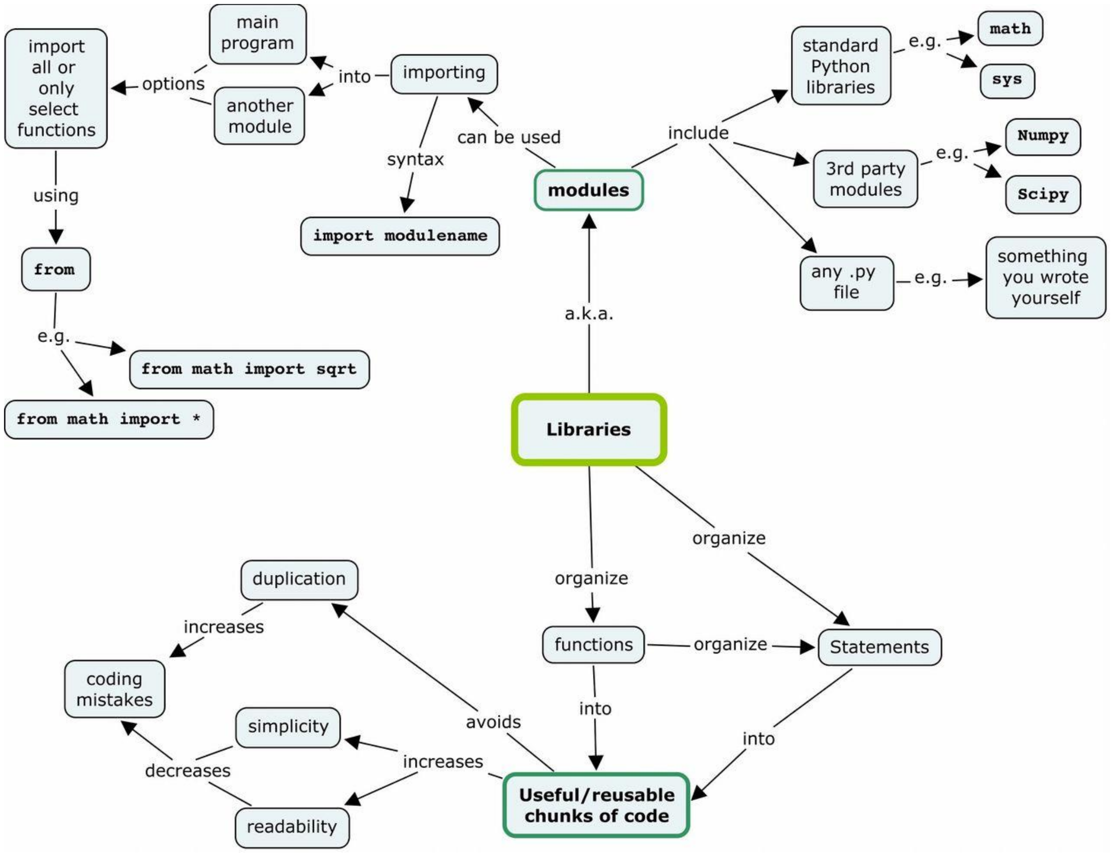


Figure III.25. Diagram Structure of Libraries [65]

III.3. A Closer Look At Some Python Libraries

III.3.1. What is NumPy Library?

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance.[66]

III.3.1.1. Difference between Numpy Arrays and the Standard Python Sequences

A. What Is a List in Python?

A list is a data structure that is built into Python and holds a collection of items.

B. What Is an Array in Python?

An array is also a data structure that stores a collection of items. Like lists, arrays are ordered, mutable, enclosed in square brackets, and able to store non-unique items.

Now that we know their definitions and features, we can talk about the differences between lists and arrays in Python:

Arrays need to be declared. Lists don't, since they are built into Python. lists are created by simply enclosing a sequence of elements into square brackets. Creating an array, on the other hand, requires a specific function from either the array module (i.e., array.array()) or NumPy package (i.e., numpy.array()). Because of this, lists are used more often than arrays. Also, arrays can store data very compactly and are more efficient for storing large amounts of data. And they are great for numerical operations; lists cannot directly handle math operations. For example, you can divide each element of an array by the same number with just one line of code. If you try the same with a list, you'll get an error.[67]

III.3.1.2. Image Processing with NumPy

Converting images to Numpy arrays or 'ndarray', means we can perform various processing tasks using the Numpy functions, such as acquisition and rewriting of pixel values, generation of single colors, Binarization, Color reduction and more. These tasks generate important data to be collected and used to analyze images for many purposes the programmer needs to manipulate the image in the input of the code.[68]

III.3.1.3. How to Import Numpy

Step 1: Install pip; pip is the tool to installing python libraries, by typing the next syntax in the terminal:

```
sudo apt install python-pip
```

Step 2: Install Numpy using the next syntax:

```
pip install numpy
```

Step 3: Import the Numpy library to your python by typing:

```
import numpy as np
```

[69]

III.3.2. What Is SciPy library?

SciPy is a library in Python, or more precisely a collection of mathematical algorithms and other functions particularly used in science. SciPy is built on NumPy, a library that extends the Python language to better manage mathematical calculations.

Thanks to SciPy, a Python session has nothing to envy to other environments like Matlab, IDL, Octave and SciLab. SciPy will prove to be an irreplaceable tool for anyone dealing with the world of scientific calculations.[70]

III.3.2.1. How the SciPy Library Is Organized

SciPy is organized in different sections, each of which covers different topics of scientific calculation. The structure of the library is therefore composed of different modules (sub-packages), each inherent to a specific application topic

- **cluster** – Clustering algorithms
- **constants** – Mathematical and physical constants
- **fftpack** – Fourier Transform functions
- **integrate** – Integration and ordinary differential equations
- **interpolate** – Interpolation and smoothing spline
- **io** – Input and Output
- **linalg** – Linear algebra
- **ndimage** – N-dimensional image processing
- **odr** – Regression of orthogonal distances

- **optimize** – Optimization and root-finding functions
- **signal** – Signal Processing
- **sparse** – Sparse matrices and associated functions
- **spatial** – Spatial data structures and algorithms
- **special** – Other special features
- **stats** – Statistical distributions and related functions[70][71]

III.3.2.2. Why We Use SciPy

SciPy contains varieties of sub packages which help to solve the most common issue related to Scientific Computation. It is the most used Scientific library only second to GNU Scientific Library for C/C++ or Matlab's. and that’s because it is Easy to use and understand as well as fast computational power and Easy to use and understand as well as fast computational power.[71]

III.3.2.3. Installing and Importing SciPy

We will use **pip** command to install the SciPy library.

```
pip install Scipy
```

In order to use the functions of this library, we will need to import this library using the following statement:

```
Import Scipy
```

[72]

III.3.3. What is Open CV?

OpenCV (Open Source Computer Vision) : Computer vision is a multidisciplinary scientific field that operates on digital images or videos to automate tasks that the human visual system can do. Computer vision tasks include gathering, processing and analyzing the information from digital images

OpenCV is a library for computer vision that includes numerous highly optimized algorithms that are used in Computer vision tasks. The library has more than 2500 algorithms and is capable of processing images and videos to detect faces, identify objects, classify human actions, track moving objects, color detection, pattern recognition and many more.[73][74]

III.3.3.1. Features of OpenCV Library

The use of OpenCV library will allow you to:

- Read and write images
- Capture and save videos
- Image processing such as filtering and transformation
- Feature detection
- image object detection such as human body parts, cars, signage
- color detection [75]

III.3.3.2. Installing and Importing SciPy

We will use **pip** command to install the OpenCV library.

```
pip install OpenCV
```

In order to use the functions of this library, we will need to import this library using the following statement:

```
Import OpenCV
```

[76]

Conclusion

In this chapter we talked about python libraries, and we explained the difference between models' packages and libraries. Also, displayed how to create and import them inside a code. In addition, we discussed the most important libraries that we are going to use in our practical work, which they are NumPy OpenCV and SciPy. And, we seen how to download them and use them in our work.

In the next chapter will be displaying our experimental results of our work by explaining how we are going to extract the information. and applying our code to some satellite images to show its functionality.

Chapter Four:

Practical Part

IV.1. Introduction

In our previous chapters we explained thoroughly image analysis and processing, and the language of python that we are going to be using in this final chapter. What we intend to do in this chapter is to create an algorithm which perform analysis of a satellite type image. We are going to use several operations to extract information from this image.

First of all, we are going to present he overall architecture of the algorithm, using a diagram that illustrate its various stages. Then we apply this algorithm on several images, to validate it. Finally, we present the results obtained with their discussions.

IV.2. Algorithm Structure

To extract information, we decided to separate our algorithm into three different blocks. All starting with the input image set by a path so all the blocks get the same input image.

Each block has a specific role to extract specific information from the image

- **Block 1 Edge detection:** applying SciPy and NumPy, we compute the gradient magnitude and direction and then implementing the Sobel Kernels using convolution to get the edges of an image.
- **Block 2 Histogram extraction:** using the libraries NumPy and OpenCV. we calculate and draw histograms of the RGB color channels of the image.
- **Block 3 color detection and contour extraction:** with the help of NumPy and OpenCV, we are going to detect “Water, Ice, forests or green land, and deserted or mountain area” using color detection on satellite imagery and segmentation to highlight the needed zones then draw out the contours to extract the zones or the areas detected.

The following figure summarizes the structure of our information extraction algorithm:

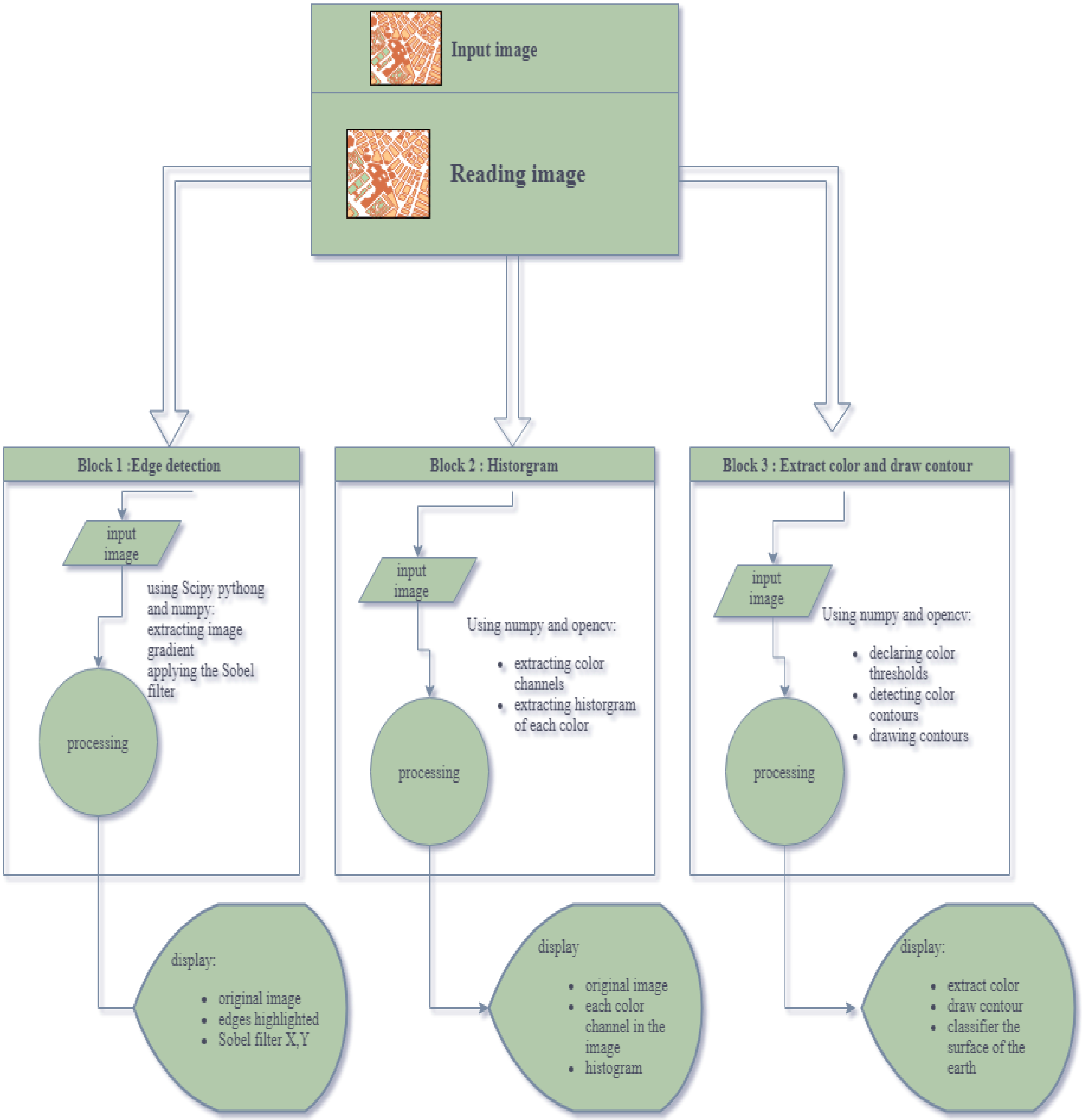


Figure IV.26. Diagram of The structure of our information extraction algorithm

IV.2.1. Block 1 Edge Detection

IV.2.1.1. Transformation from RGB to GRAYSCALE

In this part of our work, we need to convert our image from RGB to a GRAYSCALE. Because the Sobel algorithm works by measuring the varying pixel intensity in an image. So it’s easier to accomplish when the image is a standardized grayscale image.

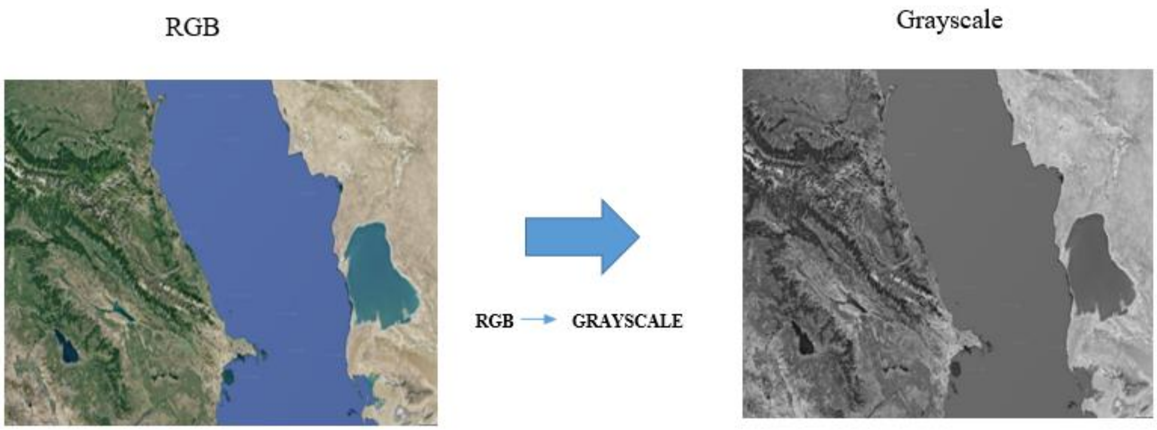


Figure IV.27. Image Transformation from RGB and Grayscale

IV.2.1.2. Gradient Computation

Computing the gradient requires first to compute gradient magnitude and direction. To calculate the gradient magnitude, we need to compute gradient in both X and Y direction. Lets call them Gx and Gy. In order to calculate Gx and Gy, consider a grayscale image of rectangle. To get Gx, we move from left to right showing the points where the intensity of the image is changing. And also do the same concept to Gy, we move from top to bottom while showing the points where the intensity of the image are changing again.

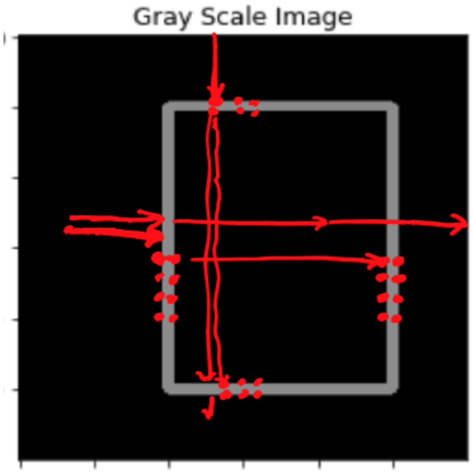


Figure IV.28. Gray ScaleImage highlights the points Gx, Gy Where the Direction of Intensity Change

In addition, the following images will explain thoroughly how to extract the gradient magnitude from combining Gx and Gy:

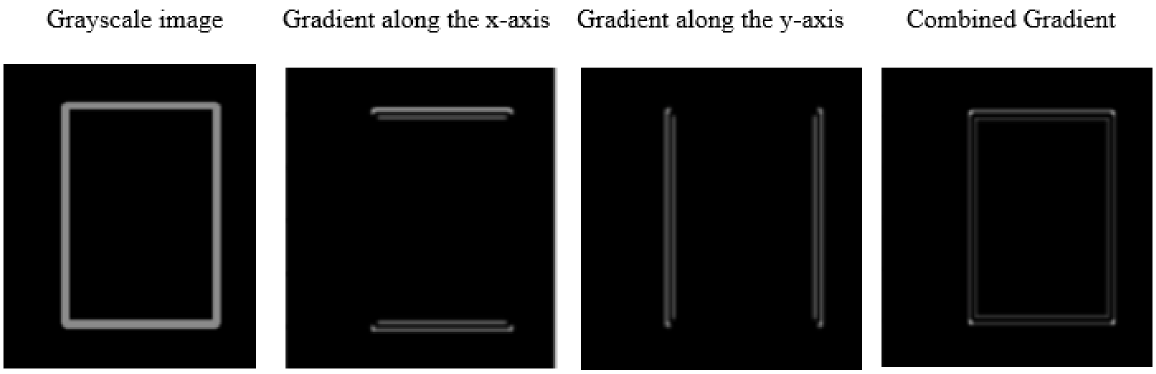


Figure IV. 29. Gx, Gy Directions Highlighted in Grayscale Image

Gx is simply the difference between pixel intensity of points left to right, and Gy is the difference between the pixel intensity of points top to bottom.

Mathematically Gradient magnitude is given as:

$$G = \sqrt{Gx^2 + Gy^2}$$

And, Gradient direction is given as:

$$\vartheta = \tan^{-1} 2\,(Gy.Gx) \times \frac{180}{\pi}$$

This gives only Gradient magnitude and gradient direction for a small 3x3 area of whole image.

IV.2.1.3. Sobel Kernel

For us to implement our previous work on the entire image, it is easily achievable by using predefined kernels.

The sobel is most common in the field of edge detection, It is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical direction. There are 2 predefined 3x3 sobel kernels Kx and Ky that are used to compute gradients. These kernels are applied to 3x3 area of image (I) at a time using convolution and then is repeated across the whole image.

Kx=

-1	**0**	**+1**
-2	**0**	**+2**
-1	**0**	**+1**

Ky=

+1	**+2**	**+1**
0	**0**	**0**
-1	**-2**	**-1**

Gx = I convolve Kx , Gy = I convolve Ky

Convolution provides a way of `multiplying together' two arrays of numbers, convolve operation is element wise multiplication of matrix elements and then summing it, We then repeat the process by sliding kernels from left to right and top to bottom.

IV.2.2. Block 2 Histogram Extraction

IV.2.2.1. Visualizing the RGB Image Channels

Before getting to the histograms, first we retrieve the image RGB color channels so we can present them properly. After reading our image, we create three channels 'red green and blue'. To visualize the image in green color we make the blue and components equal to zeroes. And we do the same to thing to get the other blue and red channel.

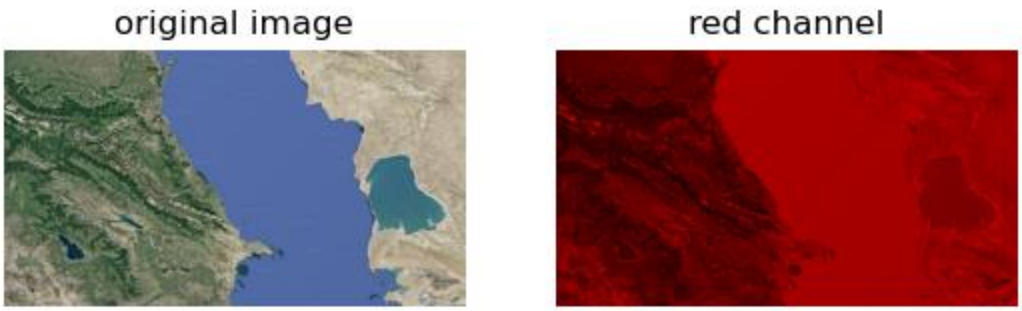


Figure IV.30. Image Demonstrates Red Channel Extraction

IV.2.2.2. Separating the Channels of the Image and Drawing the Histogram

Now to extract the RGB color histograms, first we separate the image into 3 channels, Now, to split the image into its three channels, we simply need to call the split function "cv2.split" from the cv2 module, passing as input our original image. This function will return a list with the three channels. Each channel is represented as a ndarray with two dimensions. Then we use the "ravel()" from NumPy; which is used to change a 2-dimensional array or a multi-dimensional array into a contiguous flattened array. And apply it on each channel we extracted from the image split function and combining it with the function “matplotlib.pyplot.hist()” to draw it. the histogram of channel that is needed or wanted will be drawn.

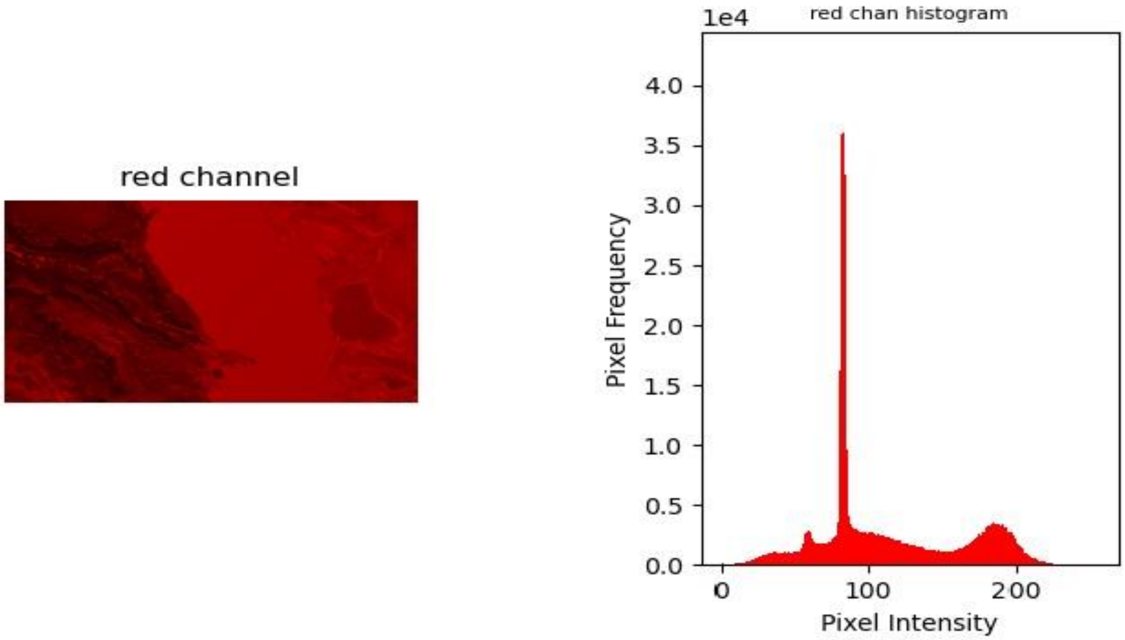


Figure IV.31. Extraction Histogram of the Red Color

IV.2.3. Block 3 Color Detection and Contour Extraction

IV.2.3.1. Transformation from RGB to HSV

In this part of our project, we will be needing the HSV color model. Because it’s easier to isolate colors with HSV values than RGB. HSV stands for:

- **Hue:** is the color portion of the model, it determines the color you need or want.
- **Saturation:** controls the amount of color used, it determines the intensity of the color.
- **Value:** determines the lightness or brightness of the color image used.

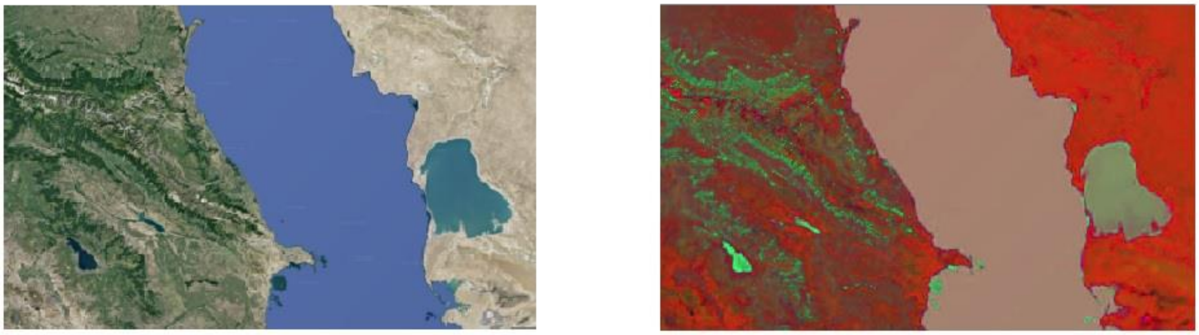


Figure IV.32. Transformation of RGB to HSV

IV.2.3.2. Defining the Range of the Colors Needed and Creating the Mask (Thresholding)

We start by determining the upper and lower HSV Color needed, they represent the boundaries that will specific the what color is detected, they come in the shape of NumPy array, so we can use the “Cv2.inrange()” function, it will isolate the boundaries of color with the image to extract a mask, this mask will select the of the areas of the color that we wanted like for example a blue color and it will detect.

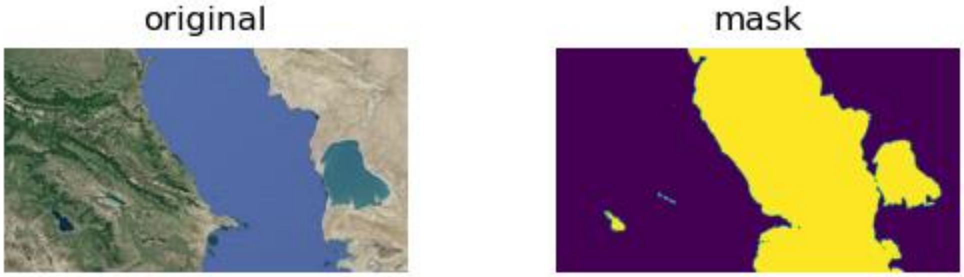


Figure IV.33. Extraction Of The Mask

IV.2.3.3. Segmenting Out the Detected Color

In the previous step, we generated a mask to determine the region of the color that was wished to detect. Now we are going to polish this mask and generate another refined mask to segment out the any region with the color selected. To accomplish that we use the “cv2.cvtcolor()” it is used to

No to use this new mask to segment our color, we are going to use the “cv2.bitwise_and()” function to bitwise AND the mask0 and the HSV image. it will give us our “Res” which is the color detected area but its on HSV so we have to change to get color space using “cv2.cvtcolor” function. The segmenting and of the color is done.

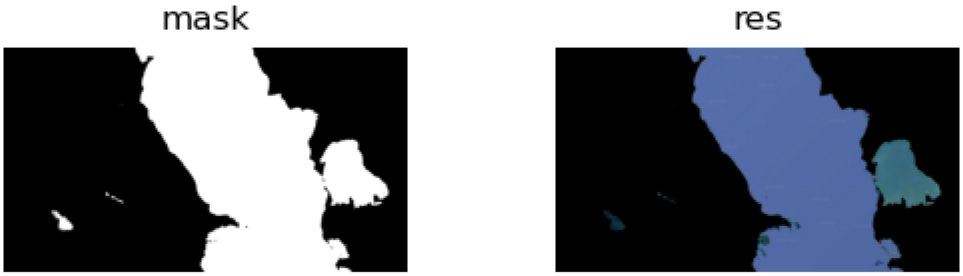


Figure IV.34. Segmentation of the Detected Color

IV.2.3.4. Contour Extraction

The last step of block 3 is to extract and draw the contour, it is a relatively easy process because we just need a threshold to draw it; Since our contour is highlighting the colored area. So we will be using the mask that was retrieved from color detection. We just use the "cv2.findcotours()" function and the contour is selected. Now to draw it we use another OpenCV function "cv2.drawcontour()" as our image and the contour selected as it inputs and it will draw the contour using our image a canvas.

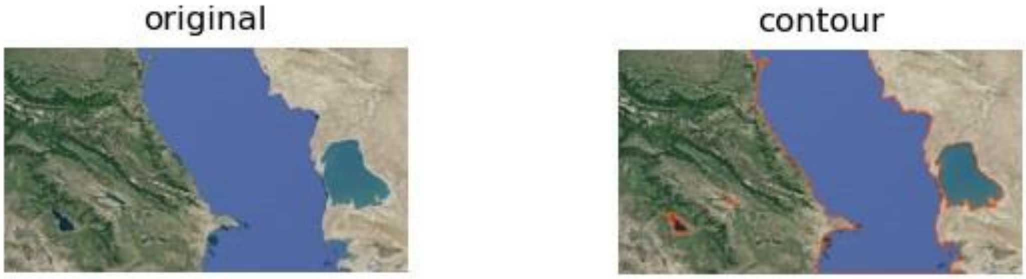


Figure IV.35. Contour Extraction

Table of Images Used

Image	Image type	Image size
A)	PNG	974x562

B)	JPEG	1345x619
C)	JPEG	978x545

Table IV.5. Information about Images Used In Our Research Work

IV.3. Experimental Results

To show the functionality of our work, we choose 3 different satellite images, those images highlight the surface of the earth from snowy lands and oceans, desert or mountain areas or greens and forests, our algorithm will extract important information from theses images that are represented in our blocks. Those information have two types qualitative meaning visual like edge detect and type of surface detection (color detection) or quantitative represented by the histograms of RGB channels.

Our results will be divided to three, image (A) result, image (B) result, and image (C) result. Every image will be the input of our algorithm and each block of the algorithm will display the information that it extracts sequentially first second and third block.

IV.3.1. Information Extracted From Image (A)

IV.3.1.1. Block 1 Edge Detection

Block1's job is to display for us 3 resulting images, that show the changes of the intensity of the pixels in two different direction, on the X axis and the Y axis; meaning the derivative x and y. and the magnitude of the gradient which shows the edges of various surfaces of the earth in the image. As we can see, the main areas where the pixel intensity doesn't change too much, don't have many edges in them, but where there is big changes of pixel intensity, it draws a lines that represent the edges in the image, as for the derivative x and y, the pixels that are representing the same areas in the original image, they show a slight change in the way they are both representing the original image because they are in different direction. And when those intensity changes are convoluted with the sobel, they give the final product which is our magnitude gradient image that represent the edge detection result.

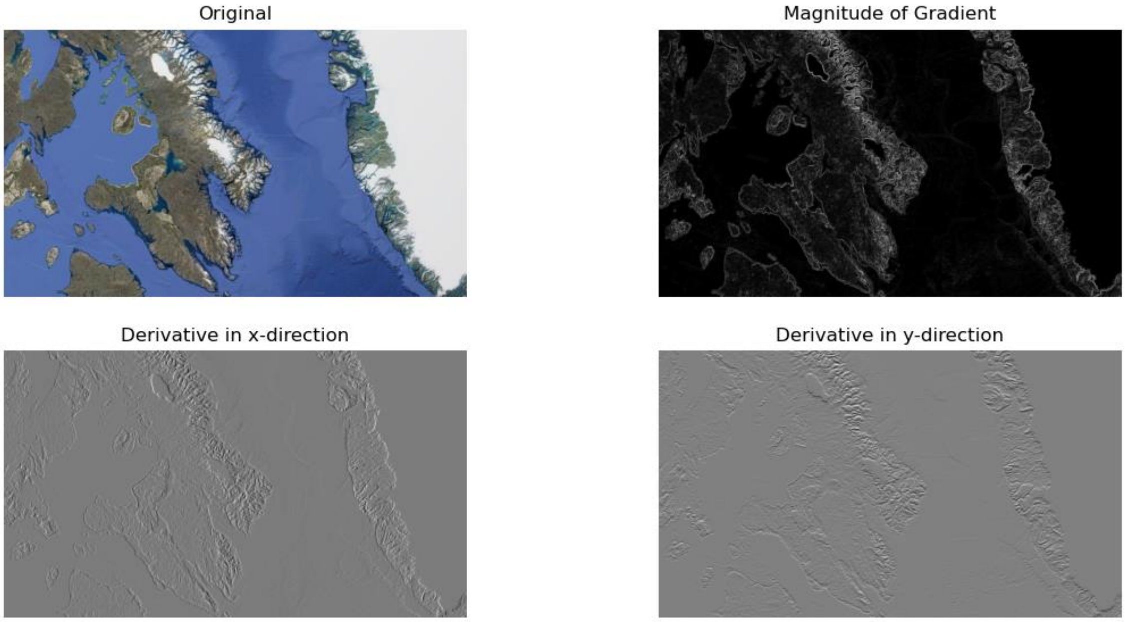


Figure IV.36. Block 1 Results of Image (A) Gradient Magnitude; i.e.,Edge Detection

IV.3.1.2. Block 2 Histogram Extraction

Block2 has a chore of extracting the RGB color channels, red channel, blue channel and green channel. Then it will calculate the histogram of each color channel. The result of the RGB channels are three images showing only the color of the channel. Each histogram has its own result, so lets talk about them separately:

- Red channel histogram: pixels with intensity from 0 with almost 70 have really low frequency with some spikes along the way, from 70 to 105 pixel intensity its shows really good frequency and hitting its peak 31000 frequency on 83, from 105 to 210 the frequency of those pixel intensities is really low, and from it to 255 the pixels have decent frequency of almost have its peak result.
- Green channel histogram: from 0 to 50 pixel intensity they are almost none existing but from 50 to 115 it keeps rising with some spikes until hits its peak of 23000 on 110. And then it will drop to low levels until it gets to 230 and will spike to high and stay at it.

- Blue channel histogram: from 0 to 50 it is very very low none existing, but then it will take a shape of a hump peaking at 4800 and keeps going low until hits 113 and spikes to highest levels of frequency at of 18500 then get very low again and peak at 232.
- global histogram: it combines all the RGB histograms

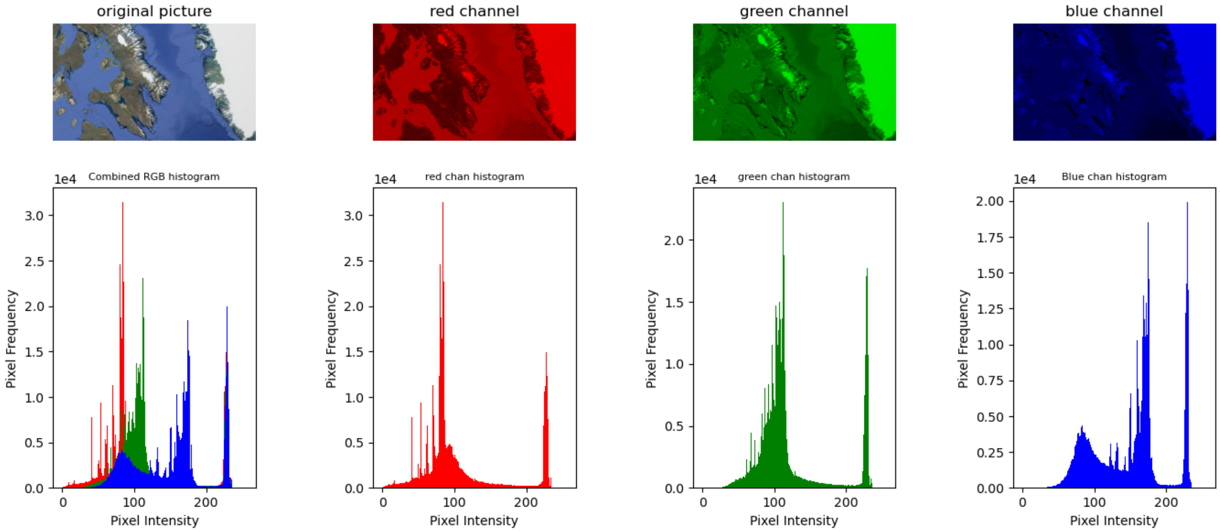


Figure IV.37. Block 2 Results of Image (A) Color Channels and RGB Colors Histograms

IV.3.1.3. Block 3 Color Detection (Surface Type) and Contour Extraction

We reach the main event of our algorithm when we will use our color detection block to detect surface type of our satellite image, then extract that part of the image using the mask and the res to show off that region and highlight its Perimeter using contour extraction. To see in the first raw of our result represents the blue color detection which will represent the oceans or water area of the satellite image, the mask will highlight color pixels , the res will separate it and show only oceans or blue color and the contour will highlight its perimeter on the original image. Same for the other colors, the mask will highlight color pixels , the res will separate it and show only the color and the contour will highlight its perimeter.

- First blue color representing oceans and water regions: the color blue is on the majority of the image, its contour is drawn by orange color

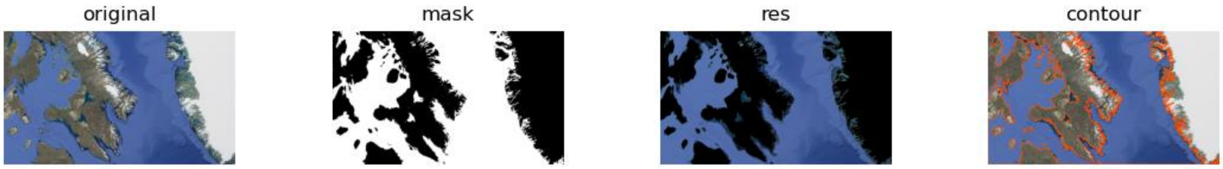


Figure IV.38. Blue Color Detection-Water Surface Classification- and Its Contour

- Second the green color representing forests: the green has a big area also in this image and its contour is highlighted by the color pink

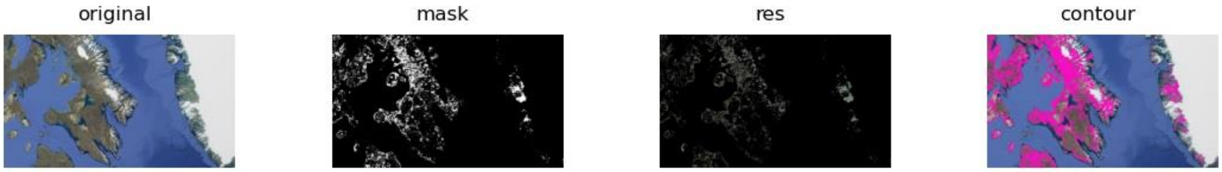


Figure IV.39. Green Color Detection-Forest Surface Classification- and Its Contour

- Third the yellow color representing desert or mountain areas: yellow is really missing this image, it doesn't show much and the parts that are showing are highlighted with blue on its contour.

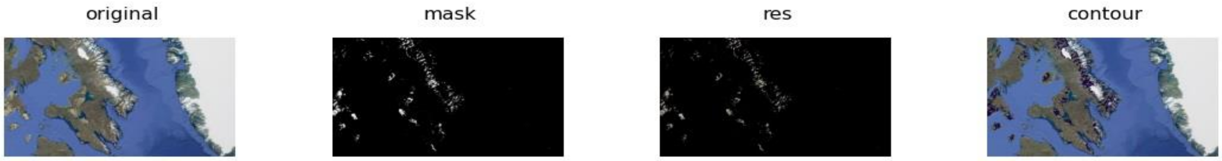


Figure IV.40. Yellow Color Detection-Desert Surface Classification- and Its Contour

- Forth the white color representing snowy areas: its really highlighted on the right of the image with black color on the contours but we also see in the left drawing lines where it shouldn't, that is because the changes of the color intensities which spikes to give us the white color and then, the contour will draw those edges.

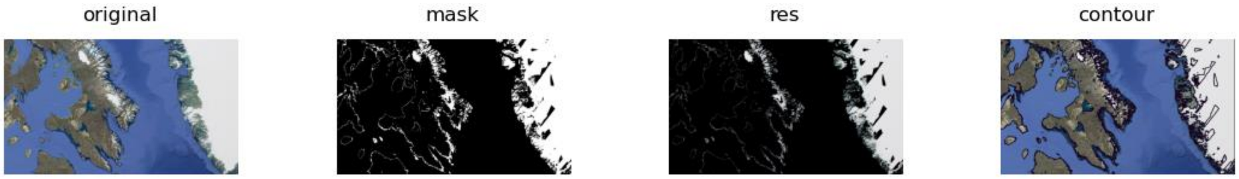


Figure IV.41. White Color Detection-Snow Surface Classification- and Its Contour

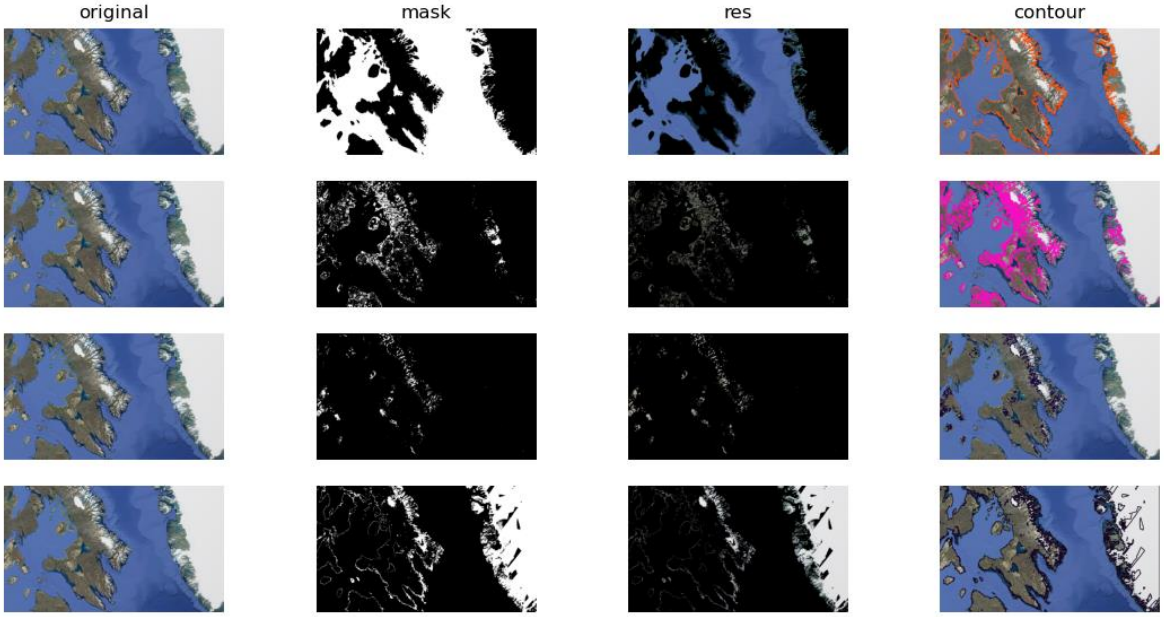


Figure IV.42. Block 3 Results of Image (A) Color Detection and Contour Extraction: i.e., Earth Surface Classification off Satellite Image

IV.3.2. Information Extracted From Image (B)

IV.3.2.1. Block 1 Edge Detection

Since block 1 has the same function on all images, the result on this image (B) it will be the same on image (A), it will extract the derivative x and derivative y and then show the gradient magnitude of the original image show the edges of the pixels intensity changes.

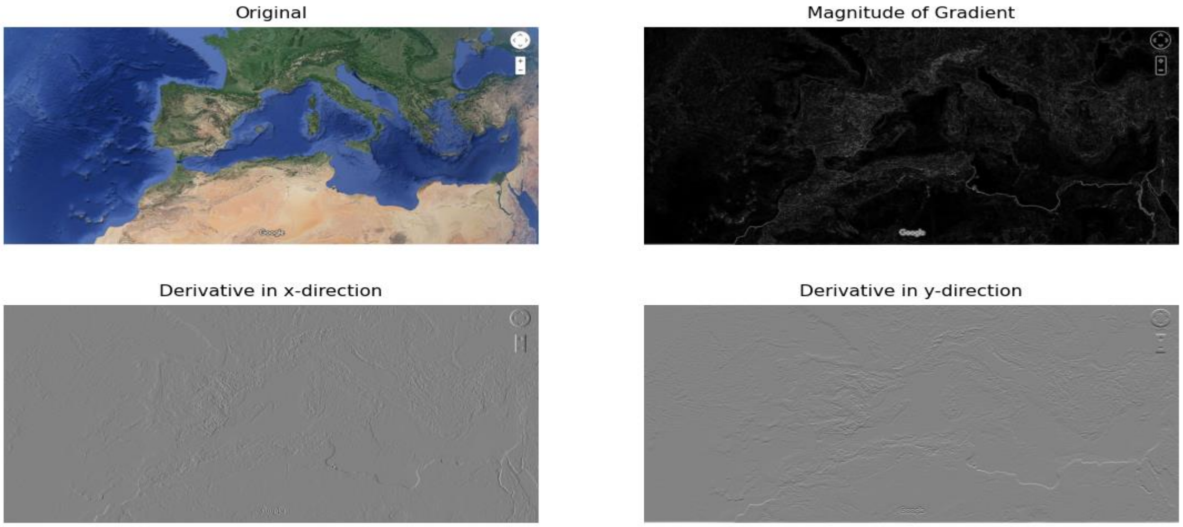


Figure IV.43. Block 1 Results of Image (B) Gradient Magnitude; i.e., Edge Detection

IV.3.2.2. Block 2 Histogram Extraction

Block 2 will continue its chores on this image by separating and showing its colors channels RGB, and then it will extract their histograms that will be explaining its results:

- Red channel histogram: from 0 to 90, the reading of the pixel intensities show that they have very high frequencies hitting its peak of 11000 on 43, then it will drop and stay on approximately 1500 from 100 to 165 then it will rise and keep rising to hit 4600 on 220 and stay very low after.
- Green channel histogram: it starts from 20 with very high frequencies and hitting the peak of 10000 and starts to drop at 114 then recover and rise then drop at 196, then its not existing until it spikes again on some intensities.
- Blue channel histogram: it starts at 20 pixel intensities and the frequencies keep rising until hits peak of 13800 then starts dropping to hit almost none existing at 200 pixel intensity.

- Global histogram: combines all 3 RGB channels histograms

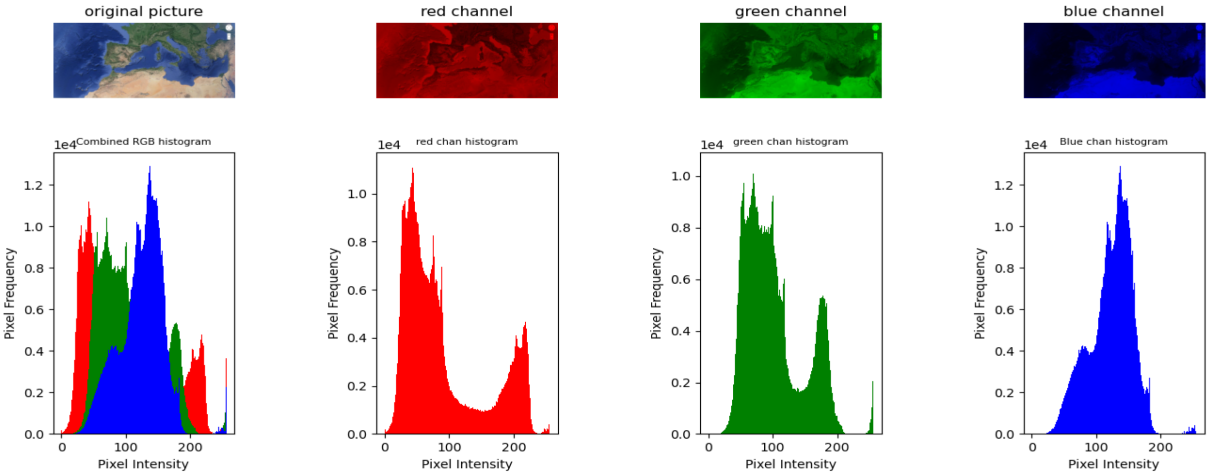


Figure IV.44. Block 2 Results of Image (B) Color Channels and RGB Colors Histograms

IV.3.2.3.Block 3 Color Detection (Surface Type) and Contour Extraction

Block 3 will do the same job done in image (A) to image (B), the results are basically the same with just one deference, the white color is missing from this image. and that’s why we see edges drawn because of color intensity changing as explained why in image (A) results.

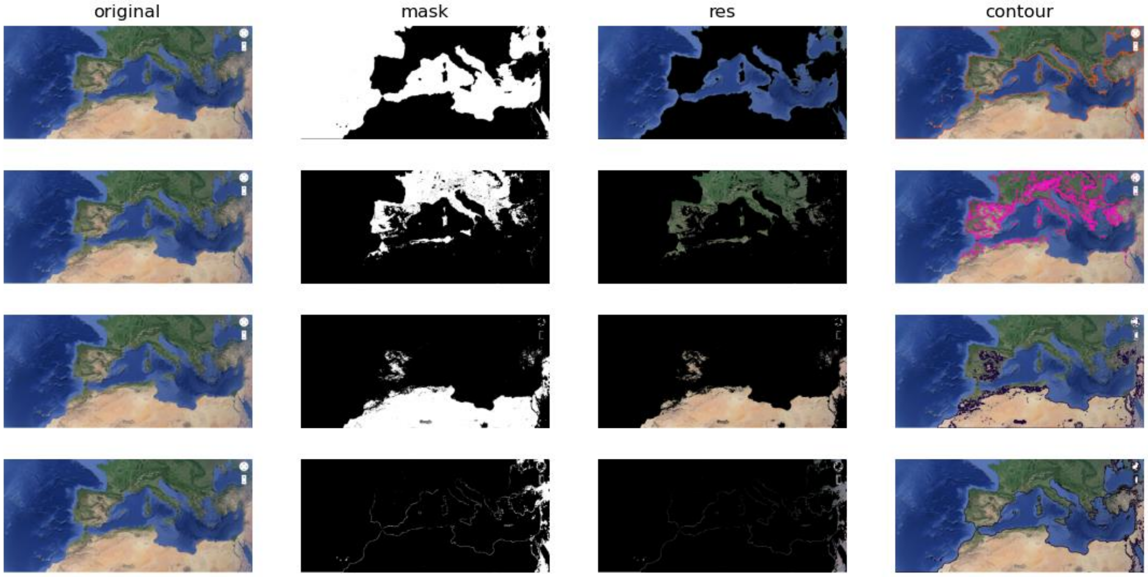


Figure IV.45. Block 3 Results of Image (B) Color Detection and Contour Extraction: i.e., Earth Surface Classification off Satellite Image

IV.3.3. Information Extracted From Image (C)

IV.3.3.1. Block 1 Edge Detection

Results of block 1 performed on image (C), it has the same explication of image (A) and image(B).

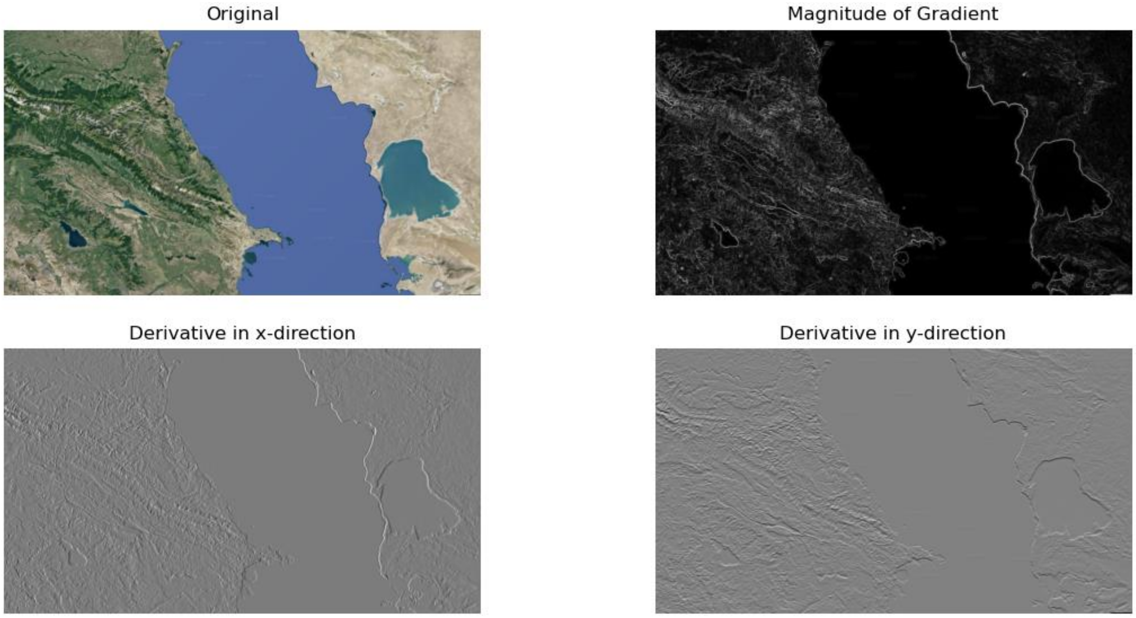


Figure IV.46. Block 1 Results of Image (C) Gradient Magnitude; i.e., Edge Detection

IV.3.3.2. Block 2 Histogram Extraction

After getting the color channels, lets explain the histograms:

- Red channel histogram: the reading of the intensity of pixels are very low all over the histogram and doesn’t go above 3600, it just spikes and peaks on the 80s intensity to hit 35000 intensity.
- Green channel histogram: green channel histogram on this image have very very low intensity from 0 to 214 and doesn’t go above 3700, it just spikes and peaks on 100 to 105 to hit 43000 frequency.
- Blue channel histogram: same as other channels, this channel histogram has very low pixel intensity, it spikes and peaks between 162 and 168 pixel intensity with a peak of 31000 pixel frequency.
- Global histogram combines all 3 RGB channels histograms

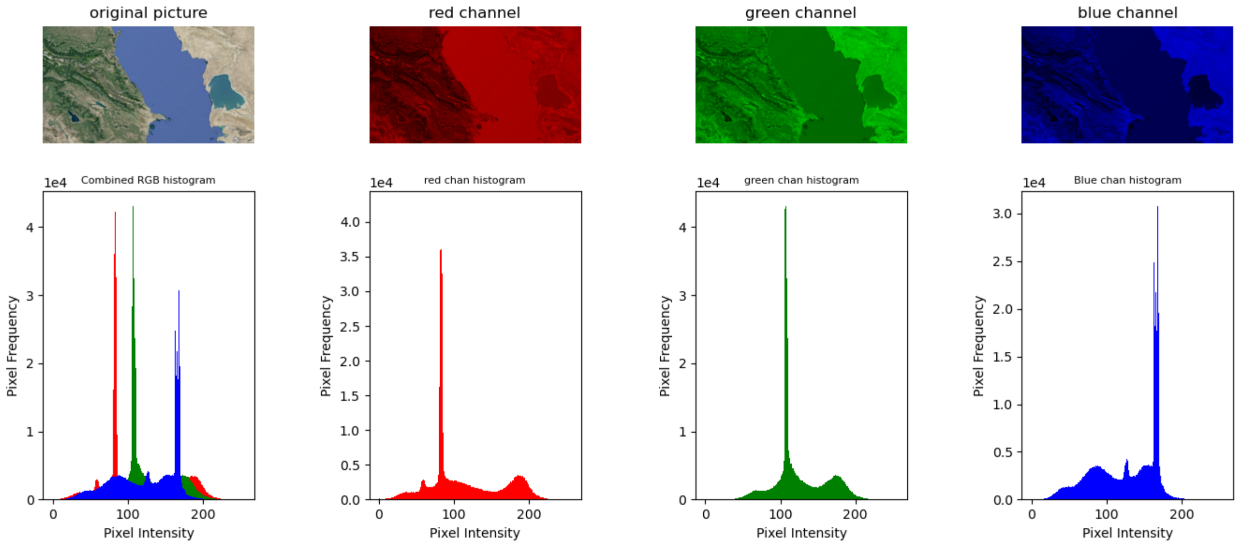


Figure IV.47. Block 2 Results of Image (C) Color Channels and RGB Colors Histograms

IV.3.3.3. Block 3 Color Detection (Surface Type) and Contour Extraction

Block 3 will do the same job done in image (B) to image (C), for the sake of not repeating, I will just show the result of the block performed on image (C).

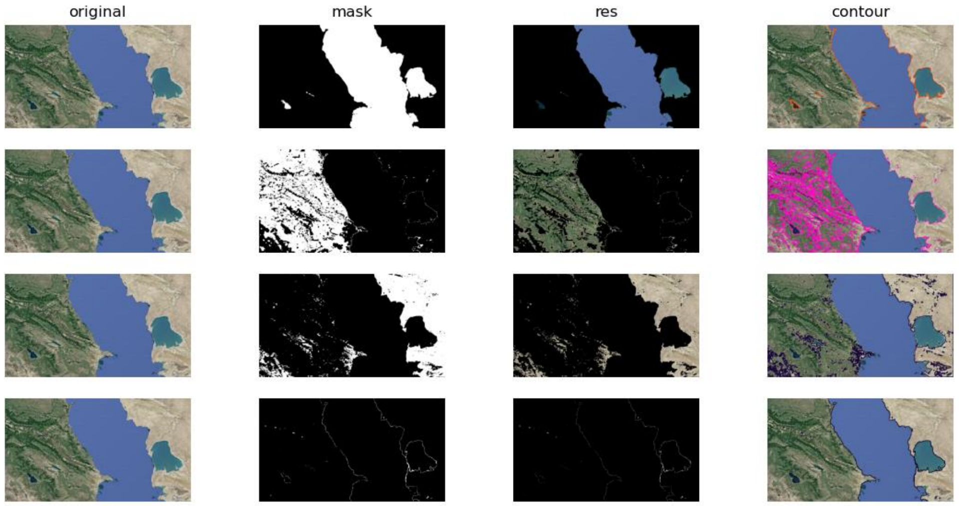


Figure IV.48. Block 3 Results of Image (C) Color Detection and Contour Extraction: i.e., Earth Surface Classification off Satellite Image

Conclusion

Extraction information is the basic step for every image analysis algorithm or image processing algorithm, those information will determine how to write your code, like the gradient which is very important and its used for edge detection algorithm that combed with contour extraction on some satellite image algorithms will result to highlight borders of forests, or the histogram can be used to draw the roads on the map using histogram values.

Our color detection block does not have much precision but with more tweaking and maybe introducing some artificial intelligence to the algorithm, it possesses a great informational starting point.

General Conclusion

General Conclusion

Our objective in this thesis was to extract information from a satellite type imagery, these information were the image gradient, its histogram, color detection and contour drawing. For the purpose of using it to extract the edges of the image and detect the type of the surface of the earth shown the in the satellite image. we used the python programming language to create and apply our code. Our code used the python libraries NumPy, OpenCV and SciPy. Our results of extracting the information were excellent, but the result of our implementation of the data was successful at detecting the edges of the image but mediocre at detecting the type of the terrain surface of the earth shown in the satellite image.

Extraction information is the foundation of any image analysis or image processing algorithms. It is the house base structure of code writing algorithms related to image manipulation. As the computer vision field expands, those applications that manipulate imagery will always use information extraction as a start to its functions.

The advancement of the information extraction is dependent on the advancement of feature recognition and advanced image analysis techniques. It seems confusing but in order to analyze image you need to extract the information from it. But to do so you need to have advanced techniques to analyze it. Those techniques like removing noises help extract more clear data from images.

Many of satellite image applications intend to extract information from the image before doing their tasks. Such as region of interest detection application that requires histogram based techniques such as Histogram classification, Histogram segmentation. Or edge detecting, satellite image's object is hardly detected because many objects are covered with cloud shadows in the sky. Edge detection has an important role in image analysis. Edge detection aims to extract the boundary of an object contained in the image. which shows the importance of information extraction of an image to manipulate it.

To conclude our work computer vision field is has an important role in the development of computer science, advanced data extraction helped grow this field for the sake of elevating how we see images and videos in the digital world

List of References

[1] - Jill Marie Koelling. Digital Imaging: A Practical Approach. [Online]. USA. 2001 .Available at: https://books.google.dz/books?id=eQH8EQKy1T0C&pg=PA27&dq=history+of+the+digital+image> (Site consulted 10/6/2020)

[2]- Donald H.House. The Digital Image Course Notes. [Online]. Texas. College of Architecture Texas A&M Universtiy,2002. 211p. available at: http://people.tamu.edu/~ergun/courses/viza654/book.pdf (Site consulted 10/6/2020).

[3] - Sarfraz Muhammad. Introductory Chapter: On Digital Image Processing. [Online]. 2020. Available at: https://www.intechopen.com/books/digital-imaging/introductory-chapter-on-digital-image-processing > (Site consulted 10/6/2020).

[4]- https://en.wikipedia.org/wiki/Pixel > (Site consulted 10/6/2020).

[5] - Sambasivarao. K . Digital Image Processing: Introduction to Digital Images. [Online]. 2019. Available at: https://towardsdatascience.com/introduction-to-images-c9c7abe6bfd2 > (Site consulted 12/6/2020).

[6] - https://en.wikipedia.org/wiki/Image_resolution#Pixel_resolution > (Site consulted 12/6/2020).

[7]- Image Resolution, Size and Compression. [Online]. Available at: https://microscope-microscope.org/microscope-info/image-resolution >(Site consulted 12/6/2020).

[8]- Wasseem Nahy Ibrahem. ELECTRICAL 101: Image processing lecture 2. [Online] Lahore. University of South Asia. Available at: https://www.uotechnology.edu.iq/ce/lecture%202013n/4th%20Image%20Processing%20_Lectures/DIP_Lecture2.pdf >(Site consulted 13/6/2020).

[9] – https://en.wikipedia.org/wiki/Color >(Site consulted 13/6/2020).

[10]- Dr. Karen K. DeValois, Michael A. Webster . Color vision. 2011. Available at : http://www.scholarpedia.org/article/Color_vision >(Site consulted 16/6/2020).

[11] - Pappas Stephanie. How Do We See Color?. [Online]. (Modified on April 29, 2010). Available at: https://www.livescience.com/32559-why-do-we-see-in-color.html > (Site consulted 16/6/2020).

[12] - https://en.wikipedia.org/wiki/Visible_spectrum >(Site consulted 16/6/2020).

[13] - Helmenstine Anne Marie. The Visible Spectrum: Wavelengths and Colors. Wavelengths of Visible Light. [Online]. 2020. Available at: https://www.thoughtco.com/understand-the-visible-spectrum-608329 >(Site consulted 16/6/2020).

[14] – Jones Andrew Zimmerman . What Is the Visible Light Spectrum?. [Online]. 2020. Available at: https://www.thoughtco.com/the-visible-light-spectrum-2699036 >(Site consulted 16/6/2020)

[15]- R. Fisher, S. Perkins, A. Walker and E. Wolfart. HIPR Hypermedia Image Processing Reference. [Online]. JOHN WILEY & SONS LTD Chichester , New York, Brisbane, Toronto, Singapore. 1997. Available at: https://www.dsi.unive.it/~atorsell/Hipr.pdf (Site consulted 25/6/2020).

[16]- What is a Color Model? - Uses & Definition. (2019, September 8). Available at https://study.com/academy/lesson/what-is-a-color-model-uses-definition.html. (Site consulted 25/6/2020).

[17]-https://en.wikipedia.org/wiki/Color_model#Additive_and_subtractive_color_models >(consulted25/6/202)

[18]- The 4 important color models for presentation design (Part III). [Online]. (Modified March 30, 2016). Available at: https://presentitude.com/color-theory-part-iii/ > (Site consulted 25/6/2020)

[19]- https://en.wikipedia.org/wiki/Image_analysis>(Site consulted 25/6/2020).

[20]- Introduction to digital image processing. [Online]. University of Tartu. Available at: https://sisu.ut.ee/imageprocessing/book/1 . (Site consulted 25/6/2020).

[21]-P.Strumillo. Image segmentaion. [Online]. Available at: http://www.eletel.p.lodz.pl/mstrzel/imageproc/segmentation.PDF (Site consulted 25/6/2020)

[22] - Rockikz Abdou. How to Detect Contours in Images using OpenCV in Python. [Online]. 2020. Available at: https://www.thepythoncode.com/article/contour-detection-opencv-python (Site consulted 25/6/2020).

[23]- Ghuneim George Abeer. Contour Tracing. [Online]. (Modified on 2000). Available at: http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/intro.html (Site consulted 26/6/2020).

[24] - Abdalla Mohamed Hambal , Dr. Zhijun Pei , Faustini Libent Ishabailu. Image Noise Reduction and Filtering Techniques. [Online]. 2017. Volume 6. Issue 3. Available at: https://www.ijsr.net/archive/v6i3/25031706.pdf (Site consulted 26/6/2020).

[25] –image processing learning resources HIPR2. [Online]. (Modified on 13/10/ 2013) available at: https://homepages.inf.ed.ac.uk/rbf/HIPR2/filtops.htm (Site consulted 26/6/2020).

[26]- Thipkham Pitchaya. Image Processing Class #4 Filters. [Online]. 2018. available at: https://towardsdatascience.com/image-processing-class-egbe443-4-filters-aa1037676130 (Site consulted 26/6/2020).

[27] - Nonlinear Filter website. [Online]. Available at: https://deepai.org/machine-learning-glossary-and-terms/nonlinear-filter (Site consulted 26/6/2020).

[28] - https://en.wikipedia.org/wiki/Image_gradient (Site consulted 26/6/2020).

[29] - L3Harris Geospatial. Apply Laplacian Filters. [Online]. Available at: https://www.harrisgeospatial.com/docs/LaplacianFilters.html (Site consulted 26/6/2020).

[30]- MissingLink.ai. Image Segmentation in Deep Learning: Methods and Applications. [Online]. (Site created 2016). Available at: https://missinglink.ai/guides/computer-vision/image-segmentation-deep-learning-methods-applications/ (Site consulted 26/6/2020).

[31] - Meera Ramadas, Ajith Abraham. Metaheuristics for Data Clustering and Image Segmentation. [Online]. Switzerland. 2019. Available at: https://books.google.dz/books?id=i8l-DwAAQBAJ&printsec=frontcover&dq=Meera+Ramadas (Site consulted 26/6/2020).

[32] - PythonForBeginners.com . What is Python?. [Online] (Site created 2012) available at: https://www.pythonforbeginners.com/learn-python/what-is-python (site consulted 5/4/2020).

[33] - Introduction to Python Programming. Boca Raton, USA.   By Gowrishankar ,S. Veena A. 2017.

[34] - John Wolfe. A Brief History of Python. [Online]. 2018. Available at: https://medium.com/@johnwolfe820/a-brief-history-of-python-ca2fa1f2e99e (Site consulted 5/4/2020).

[35] - Moore William. Benefits of Python over Other Programming Languages. [online]. (modified on March 12, 2015). Available at: https://www.invensis.net/blog/benefits-of-python-over-other-programming-languages/ (Site consulted 5/4/2020).

[36]- EDUCBA. C vs Python [online]. Available at: https://www.educba.com/c-vs-python/ (Site consulted 5/4/2020).

[37]- Guru99. Python 2 vs Python 3: Key Differences. [Online] available at: https://www.guru99.com/python-2-vs-python-3.html (site consulted 6/4/2020).

[38]- Sejal Jaiswal. Python Data Structures Tutorial. [Online]. (Modified on 8/12/2017). Available at: https://www.datacamp.com/community/tutorials/data-structures-python. (Site consulted 6/4/2020).

[39]- Edpresso. Data types in Python. [Online]. (Modified on 27/8/ 2020). Available at :

https://www.educative.io/edpresso/data-types-in-python. (Site consulted 23/9/2020).

[40]- Guru99. Python Variables: Declare, Concatenate, Global & Local. [Online]. Available at: https://www.guru99.com/variables-in-python.html . (Site consulted 10/4/2020).

[41]- Tutorialspoint. Python - Variable Types. [Online]. Available at: https://www.tutorialspoint.com/python/python_variable_types.htm (Site consulted 10/4/2020).

[42]- learnpython.org. Variables and Types. [Online]. Available at: https://www.learnpython.org/en/Variables_and_Types (Site consulted 10/4/2020).

[43]- Sturtz John. Basic Input, Output, and String Formatting in Python. [Online]. Available at: https://realpython.com/python-input-output/#formatted-string-output (Site consulted 10/4/2020).

[44] – Tutorialspoint. Python - Basic Operators. [Online]. Available at: https://www.tutorialspoint.com/python/python_basic_operators.htm (site consulted 10/4/2020).

[45] Net-informations. Python control structures. [Online]. Available at: http://net-informations.com/python/flow/default.htm (site consulted 10/4/2020).

[46] – Tutorialspoint. Python - Decision Making. [Online]. Available at: https://www.tutorialspoint.com/python/python_decision_making.htm (Site consulted 10/4/2020).

[47]- Net-informations. Python Conditional Statements. [Online]. Available at: http://net-informations.com/python/flow/if.htm (site consulted 10/4/2020).

[48] - Net-informations. Python while loop Statements. [Online]. http://net-informations.com/python/flow/loop.htm (Site consulted 13/4/2020).

[49]- Net-informations. Python for Loop [Online]. Available at: http://net-informations.com/python/flow/for.htm (Site consulted 13/4/2020).

[50]- Programiz. Python User-defined Functions. [Online]. Available at: https://www.programiz.com/python-programming/user-defined-function (Site consulted 13/4/2020).

[51]- Singh Chaitanya. Python Functions. [Online]. (Modified on 01/2018). Available at: https://beginnersbook.com/2018/01/python-functions/ (Site consulted 13/4/2020).

[52]- Learnpython.org. Modules and Packages. [Online]. Available at: https://www.learnpython.org/en/Modules_and_Packages (Site consulted 25/7/2020).

[53]- Tutorialspoint. Python – Modules. [Online]. Available at: https://www.tutorialspoint.com/python/python_modules.htm (Site consulted 30/7/2020).

[54]- Tutorialsteacher. Built-in Modules in Python. [Online]. Available at: https://www.tutorialsteacher.com/python/python-builtin-modules (Site consulted 30/7/2020).

[55]- Mihajlo Pavloski. Python Modules: Creating, Importing, and Sharing. [Online]. Available at: https://stackabuse.com/python-modules-creating-importing-and-sharing/ (Site consulted 30/7/2020).

[56]- Edpresso. What are Python modules? [Online]. (Modified on 8/2020). Available at: https://www.educative.io/edpresso/what-are-python-modules. (Site consulted 5/8/2020).

[57]- Waseem Mohammad. Python Modules- All You Need To know. [Online]. (Modified on 27/11/2019). Available at: https://www.edureka.co/blog/python-modules/ (Site consulted 5/8/2020).

[58]- Edpresso. What are Python packages?. [Online]. (Modified on 8/2020). Available at: https://www.educative.io/edpresso/what-are-python-packages (Site consulted 5/8/2020).

[59]- Data flair. Python Packages Tutorial – How to Create Your Own Package. [Online]. Available at: https://data-flair.training/blogs/python-packages/ (Site consulted 10/8/2020).

[60]- Programiz. Python Package. [Online]. Available at : https://www.programiz.com/python-programming/package (Site consulted 10/8/2020).

[61]- Pythoncentral. How to Create a Python Package. [Online]. (site created 10/4/2012, modified on 9/5/2013). Available at: https://www.pythoncentral.io/how-to-create-a-python-package/ (Site consulted 11/8/2020).

[62]- Data flair. Python Modules vs Packages | Differences Between Python Modules and Packages. [Online]. Available at: https://data-flair.training/blogs/python-modules-vs-packages/ (Site consulted 11/8/2020).

[63]- Data flair. Python Libraries – Python Standard Library & List of Important Libraries. [Online]. Available at: https://data-flair.training/blogs/python-libraries/ (Site consulted 11/8/2020).

[64]- Python. The Python Standard Library. [Online]. Available at: https://docs.python.org/3/library (Site consulted 17/8/2020).

[65]- Technoinfinity engineering. PYTHON LIBRARIES. [Online]. (Modified on 27/10/2019). Available at: https://technoinfinityengg.com/civil/f/python-libraries (Site consulted 18/8/2020)

[66]- NumPy. What is NumPy. [Online]. Available at: https://numpy.org/doc/stable/user/whatisnumpy.html (Site consulted 18/8/2020).

[67]- Koidan Kateryna. Array vs. List in Python – What's the Difference? [Online]. (Modified on 17/12/2019). Available at: https://learnpython.com/blog/python-array-vs-list/ (Site consulted 19/8/2020).

[68]- Note.nkmk.me . Image processing with Python, NumPy (read, process, save). [Online]. (Modified on 14/05/2019). Available at: https://note.nkmk.me/en/python-numpy-image-processing/ (Site consulted 19/8/2020).

[69]- PHOENIXNAP . How To Install NumPy. [Online]. (Modified on 8/5/2020). Available at: https://phoenixnap.com/kb/install-numpy (Site consulted 19/8/2020).

[70]- Meccanismo complesso. SciPy, a Python library for mathematics, science and engineering. [Online]. (Modified on 24/11/2019). Available at: https://www.meccanismocomplesso.org/en/scipy-a-python-library-for-mathematics-science-and-engineering/ (Site consulted 19/8/2020).

[71]- Jin Hans Kim. Getting started with Python SciPy. [Online]. (Modified on 2/2020). Available at: https://morioh.com/p/7c0f837f99d1 (Site consulted 20/8/2020).

[72]- Askpython. Python SciPy Tutorial. [Online]. Available at: https://www.askpython.com/python-modules/python-scipy (Site consulted 22/8/2020).

[73]- Data flair. OpenCV Python Tutorial – Implementation of Computer Vision with a Case Study on Amazon Go. [Online]. Available at: https://data-flair.training/blogs/opencv-python-tutorial/ (Site consulted 22/8/2020).

[74]- Pal Saurabh. 16 OpenCV Functions to Start your Computer Vision journey (with Python code). [Online]. (Modified on 25/03/2019). Available at: https://www.analyticsvidhya.com/blog/2019/03/opencv-functions-computer-vision-python/ (Site consulted 22/8/2020).

[75]- Full Scale. An Overview of OpenCV. [Online]. (Modified on 10/4/2019). Available at: https://fullscale.io/blog/opencv-overview/ (Site consulted 22/8/2020).

[76]- The Python Package Index. OpenCV-python. [Online]. Available at: https://pypi.org/project/opencv-python/ (Site consulted 22/8/2020).