UNIVERSITY OF WEST BOHEMIA

Faculty of Electrical Engineering Plzen

Department of Applied Electronics and Telecommunications

BACHELOR THESIS

Image Enhancement Methods and Implementation in Matlab

Alaa Kassab   2012

Abstract

This bachelor thesis provides a description of image enhancement algorithms for multimedia technology. It contains two parts, theoretical and practical, the theoretical part of the project focuses mainly on methods of edge and color enhancement. Selected algorithms will be subsequently implemented and their efficiency will be verified using supplied image tests. In the practical part, Matlab will be used for evaluation of results.

A compact disc is attached to this work and it contains the thesis in pdf form and the Matlab code.

PROHLÁŠENÍ

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr studia na Fakultě elektrotechnické Západočeské univerzity v Plzni.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této diplomové práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

V Plzni dne 29.8.2012   Jméno příjmení

                                                                          ....

AKNOWLEDGEMENT

I would like to acknowledge and extend my heartfelt gratitude to my supervisor Eng. Radek Holota Ph.D for his kind help, encouragement and support; also I would like to thank all the professors of Department of Applied Electronics and Telecommunication and all the people that supported me to complete my thesis. Most special thanks to my parents for the constant reminders and much needed motivation. And before all, to God, who makes every things possible.

TABLE OF CONTENTS

TABLE OF FIGURES

ABBREVIATIONS

AI = Artificial Intelligence

BHPF = Butterworth High-pass Filter

EM = Electromagnetic

FSHS = Fall Scale Histogram stretch

GHPS = Gaussian High-pass Filter

HPF = High-pass Filter

IE = Image Enhancement

IHPF = Ideal High-pass Filter

ILPF = Ideal Low-pass Filter

JPEG = Joint Photographic Expert Group

LUT = Look up Table

MRI = Magnetic resonance imaging

RGB = Red, Green, Blue

YCbCr = Luminance and chrominance

Matlab = Matrix Laboratory

INTRODUCTION

My project is about image enhancement, there are two parts, spatial domain and frequency domain. I will focus on edge and color enhancement, especially on the sharpening techniques and I will use its different types. I will use some examples of filters based on sharpening technique.

In the first chapter I will introduce image enhancement and its uses, advantages, principles, and types.

In the second chapter I will introduce enhancement in the spatial domain, its theory, techniques and types.

In the third chapter I will explain the enhancement in frequency domain. And in the fourth chapter will be the practical part; I will use Matlab for my calculations.

Furthermore, I shall thorough fully introduce the image processing in detail.

One picture is worth more than ten thousand words. The ability to see is one of the truly remarkable characteristics of living beings. It enables them to perceive and assimilate in a short span of time an incredible amount of knowledge about the world around them. The scope and variety of that which can pass through the eye and be interpreted by the brain is nothing short of astounding.

It is thus with some degree of trepidation that we introduce the concept of visual information, because in the broadest sense, the overall significance of the term is overwhelming. Instead of taking into account all of the ramifications of visual information; the first restriction we shall impose is that of finite image size, In other words, the viewer receives his or her visual information as if looking through a rectangular window of finite dimensions. This assumption is usually necessary in dealing with real world systems such as cameras, microscopes and

telescopes for example; they all have finite fields of view and can handle only finite amounts of information.

The second assumption we make is that the viewer is incapable of depth perception on his own. That is, in the scene being viewed he cannot tell how far away objects are by the normal use of binocular vision or by changing the focus of his eyes.

An image may be defined as a two-dimensional function, f(x, y), where x and y are spatial (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. The digital images are defined when x, y, and the amplitude values of f are all finite, discrete quantities. The field of digital image processing refers to processing digital images by means of a digital computer. Note that a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as picture elements, image elements, peels and pixels. Pixel is the term most widely used to denote the elements of a digital image. Vision is the most advanced of our senses, so it is not surprising that images play the single most important role in human perception.

The area of image analysis is in between image processing and computer vision. There are no clear-cut boundaries in the continuum from image processing at one end to computer vision at the other. However, one useful paradigm is to consider three types of computerized processes in this continuum: low-, mid-, and high-level processes. Low-level processes involve primitive operations such as image preprocessing to reduce noise, contrast enhancement, and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images. Mid-level processing on images involves tasks such as segmentation, description of those objects to reduce them to a form suitable for computer processing, and classification (recognition) of individual objects. A mid-level process is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images. Finally, higher-level processing involves "making sense" of an ensemble of recognized objects, as in image analysis, and, at the far end of the continuum, performing the cognitive functions normally associated with vision. Based on the preceding comments, we see that a logical place of overlap between image processing and image analysis is the area of recognition of individual regions or

objects in an image. As a simple illustration to clarify these concepts, consider the area of automated analysis of text. The processes of acquiring an image of the area containing the text, preprocessing that image, extracting the individual characters, describing the characters in a form suitable for computer processing, and recognizing those individual characters are in the scope of what we call digital image processing. As will become evident shortly, digital image processing, as we have defined it, is used successfully in a broad range of areas of exceptional social and economic value. **[1][3][4]**

1 Image Enhancement

The main definition of enhancing is to make something greater in value, desirability or attractiveness. The term of enhancement implies a process to improve the visual quality of the image. Image Enhancement transforms images to provide better representation of the subtle details. The principal objective of enhancement is to process an image so that the result is more suitable than the original image for a specific application. Image enhancement processes consist of a collection of techniques that seek to improve the visual appearance of an image or to convert the image to a form better suited for analysis by a human or a machine. In an image enhancement system, there is no conscious effort to improve the fidelity of a reproduced image with regard to some ideal form of the image, as is done in image restoration.

Actually, there is some evidence to indicate that often a distorted image, for example, an image with amplitude overshoot and undershoot about its object edges, is more subjectively pleasing than a perfectly reproduced original. Enhancement of an image is necessary to improve appearance or to highlight some aspect of the image is converted from one into another acquired, scanned, transmitted, copied or printed many types of noise can be present in the image. Image enhancement has come to specifically mean a process of smothering irregularities or noise that has somehow corrupted the image. The term "image enhancement" has been widely used in the past to describe any operation that improves image quality by some criteria. However, in the recent years the meaning of the term has evolved to denote image-preserving noise smoothing.

This primarily serves to distinguish it from similar-sounding terms, such as image restoration and image reconstruction, which also taking specific meaning. Image enhancement has played and will continue to play an important role into different fields such as medical, industrial, military and scientific applications. In addition to these applications, image enhancement is increasingly being used in consumer electronics. Internet Web users, for instance, not only rely on built-in image processing protocols such as JPEG (Joint Photographic Expert Group) and interpolation, but they also have become image processing users equipped with powerful yet

inexpensive software such as Photoshop. Users not only retrieve digital images from the Web but they are now able to acquire their own by use of digital cameras or through digitization services. Image enhancement is an indispensable tool for researchers in a wide variety of fields:

1. In forensics, image enhancement is used for identification, evidence gathering and surveillance. Images obtained from fingerprint detection, security videos analysis and crime scene investigations are enhanced to help in identification of culprits and protection of victims.

2. In atmospheric sciences IE is used to reduce the effects of haze, fog, mist and turbulent weather for meteorological observations. It helps in detecting shape and structure of remote objects in environment sensing. Satellite images undergo image restoration and enhancement to remove noise.

3. Astrophotography faces challenges due to light and noise pollution that can be minimized by IE. For real time sharpening and contrast enhancement several cameras have in-built IE functions. Moreover, numerous softwares allow editing such images to provide better and bright results.

4. In oceanography the study of images reveals interesting features of water flow, remains concentration, geomorphology and bathymetric patterns to name a few. These features are more clearly observable in images that are digitally enhanced to overcome the problem of moving targets, deficiency of light and obscure surroundings.

5. IE techniques when applied to pictures and videos help the visually impaired in reading small print, using computers and television and face recognition. Several studies have been conducted that highlight the need and value of using IE for the visually impaired.

6. The technique of image enhancement is often employed by virtual restoration of historic paintings and artifacts in order to reduce stains and crevices. Color contrast enhancement, sharpening and brightening are just some of the techniques used to make the images bright. IE is a powerful tool for restorers who can inform decisions by viewing the results of restoring a painting beforehand. It is evenly useful in discerning text from worn-out historic documents.

7. In the field of e-learning, IE is used to clarify the contents of chalkboard as viewed on streamed video; it improves the content readability and helps students to focus on the text.

Similarly, collaboration through the whiteboard is facilitated by enhancing the shared data and diminishing artifacts like shadows and blemishes.

8. Medical imaging uses IE techniques for reducing noise and sharpening details to improve the visual representation of the image. Since minute details play a critical role in diagnosis and treatment of disease, it is essential to highlight important features while displaying medical images. This makes IE a necessary aiding tool for viewing anatomic areas in MRI, ultrasound and x-rays to name a few.

9. Numerous other fields including law enforcement, microbiology, biomedicine, bacteriology, climatology, meteorology, etc., benefit from various IE techniques. These benefits are not limited to professional studies and businesses but extend to the common users who employ IE to cosmetically enhance and correct their images. **[1] [3][5]**

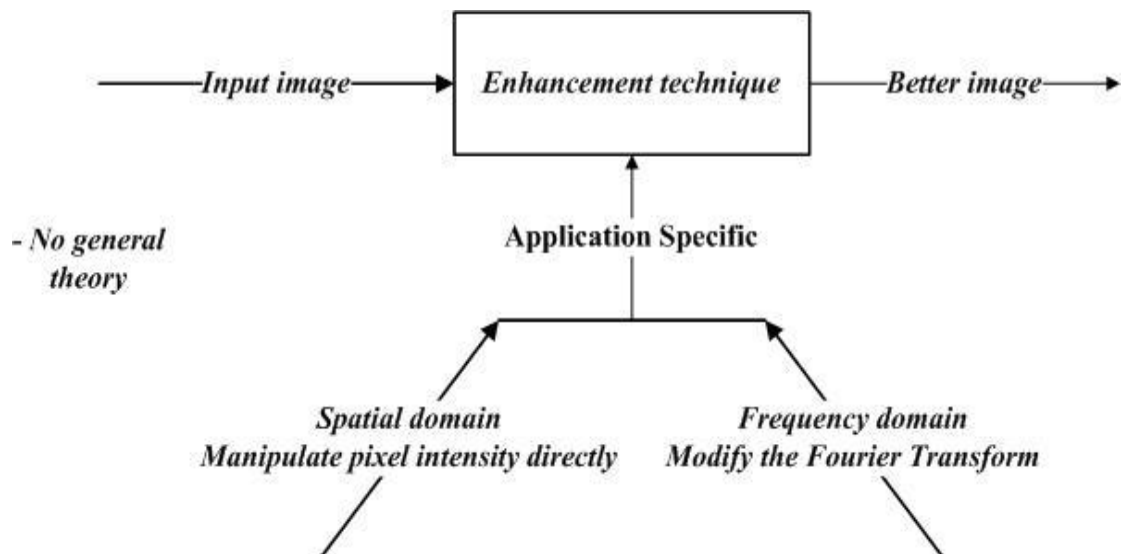The following image explains the different types of image enhancement techniques.

Figure 1: Image enhancement techniques [5]

2 Spatial domain techniques

The spatial domain techniques are based on gray level mappings, where used mapping depends on the criterion. For example, let's consider the problem of enhancing as the contrast of an image. Let r and s denote any gray level in the original and enhanced image respectively. Suppose that for every pixel with level r in original image we create a pixel in the enhanced image with level $S = T(r)$. If $T(r)$ has the form as shown in figure below:
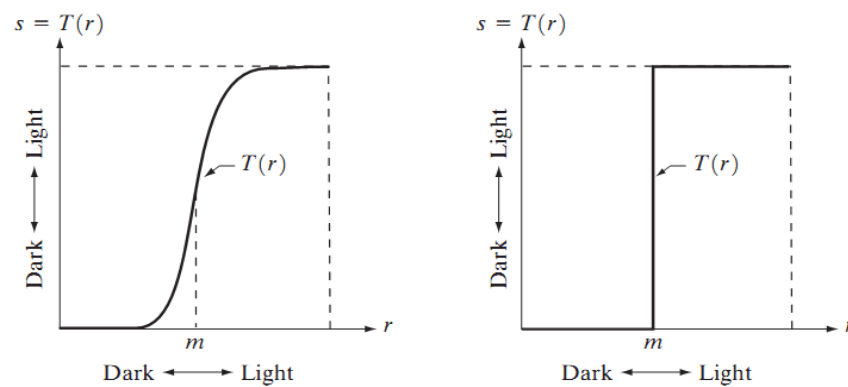
Figure 2: Example of contrast enhancement in spatial domain. [3]

The effect of this transformation will be to produce an image of higher contrast than the original by darkening the levels below a value m and brightening the levels above m in the original pixel spectrum. The technique is referred to as contrast stretching. The values of r below m are compressed by the transformation function into a narrow range of S towards the dark end of the spectrum; the opposite effect takes place for values of r above m. In the limiting case shown in figure, $T(r)$ produces 2-level (binary) image. This is also referred to as image thresholding. Many powerful enhancement processing techniques can be formulated in the spatial domain of an image. It does not exist a general definition of an image enhancement. When an image is processed for visual interpolation, the observer is the ultimate judge of how well a particular method works. Visual evaluation of image quality is a subjective process consequently making the definition of a "good image" an elusive standard by which to compare algorithm performance. When the problem is one of processing images for machine perception, the evaluation task is

easier. For example, if we take the problem of character recognition by a machine the best image processing method would be the one that yields the best machine recognition result. In general, if there is somewhere a clear cut criterion of performance imposed on a problem there is usually a certain amount of trial and error before one selects a particular image processing approach. **[3][5]**

2.1 Point processing

Point processing techniques are among the best simplest of all image enhancement techniques, considering processing methods that are based only the intensity of single pixels.

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

Simplest case: Neighborhood is (x, y)

[$g(.)$ Depends only on the value of f at (x, y)]
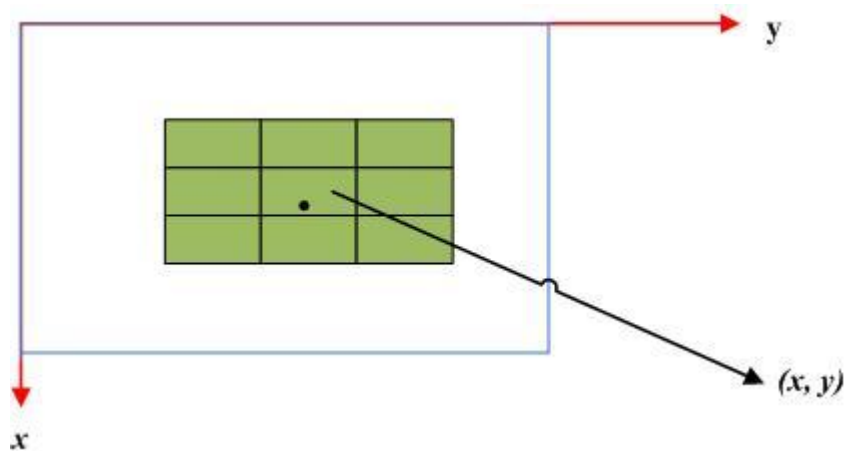
Figure 3: Simplest case of Neighborhood (x, y) **[7]**

2.1.1   Gray level transformation

Image Negative is a typical grey scale transformation that does not depend on the position of the pixel in the image. The output grey value s is related to the input grey value as follows:
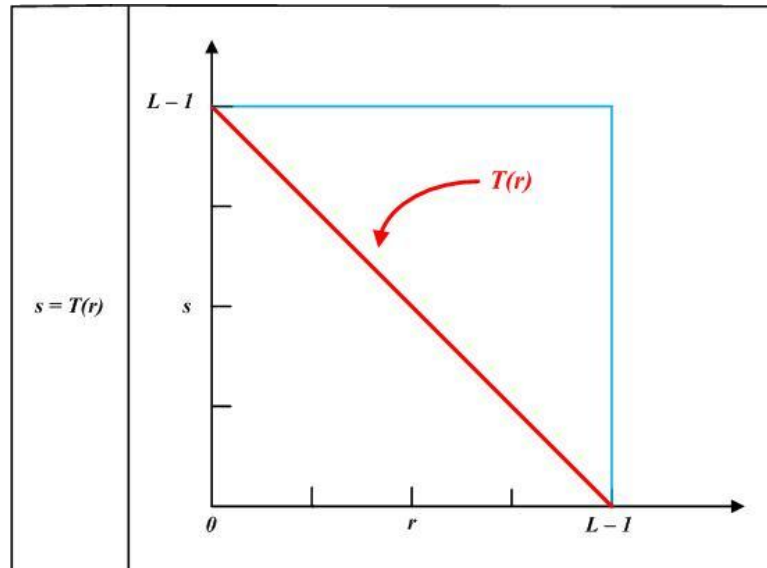
Figure 4: Gray level transformation function for obtaining the image negative of an image.[3]

Negatives of digital images are useful in numerous applications, such as displaying medical images and photographing a screen with monochrome positive film with the idea of using the resulting negatives as normal slides.

Figure 5: Example of a Negative image.

2.2 Histogram processing

The histogram in the context of image processing is the operation by the occurrences of each intensity value in the image. Normally, the histogram is a graph showing the number of pixels in an image at each different intensity value found in that image. For an 8-bit grayscale image there are 256 different possible intensities, and so the histogram will graphically display 256 numbers showing the distribution of pixels amongst those grayscale values. **[6]**

2.2.1 Normailized Histogram

Normalized histogram is the histogram in which the no. of pixels for each intensity level is divided by the total no. of pixels in the image. Hence, if the whole of the image is of the same color, e.g. a white paper picture, then its normalized histogram contains only one non-zero number, and that is 1 for the 255th intensity level, which is the intensity level for the white color.

Normalized Histogram (i) = (Total Number of Pixels of Intensity i)/(Total Number of Pixels)

The image is scanned in a single pass and a running count of the number of pixels found at each intensity value is kept. This is then used to construct a suitable histogram. This operation is performed in a single pass algorithm, surfing each pixel once, and increment the number of occurrences of the specific intensity level depending upon the intensity of that particular pixel.

$$p(r_k) = n_k / n$$

For k = 0,1,…,L-1, and $p(r_k)$ gives an estimate of the probability of occurrence of gray level n_k, and n is total number of pixels. **[6]**
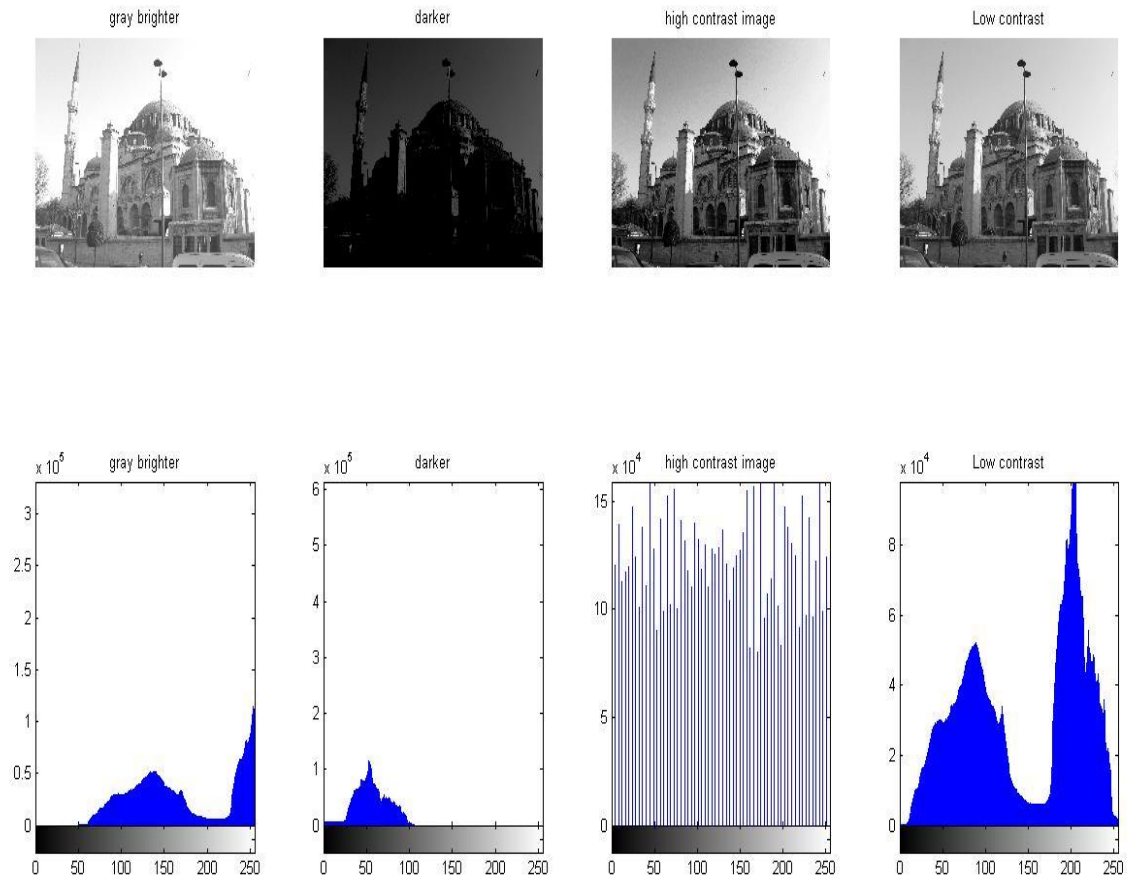
Figure 6: Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms.

2.2.1   Histogram equalization

Histogram equalization is the technique by which the dynamic range of the histogram of an image is increased. Histogram equalization assigns the intensity values of pixels in the input image such that the output image contains a uniform distribution of intensities. It improves contrast and the goal of histogram equalization is to obtain a uniform histogram. This technique can be used on a whole image or just on a part of an image. **[3]**

Histogram equalization redistributes intensity distributions. If the histogram of any image has many peaks and valleys, it will still have peaks and valley after equalization, but peaks and valley will be shifted. Because of this, "spreading" is a better term than "flattening" to describe histogram equalization. In histogram equalization, each pixel is assigned a new intensity value based on its previous intensity level. **[3]**

As the low-contrast image's histogram is narrow and centered toward the middle of the gray scale, if we distribute the histogram to a wider range the quality of the image will be improved.

It can be done, by adjusting the probability density function of the original histogram of the image, so that the probability spread equally. **[3][6]**
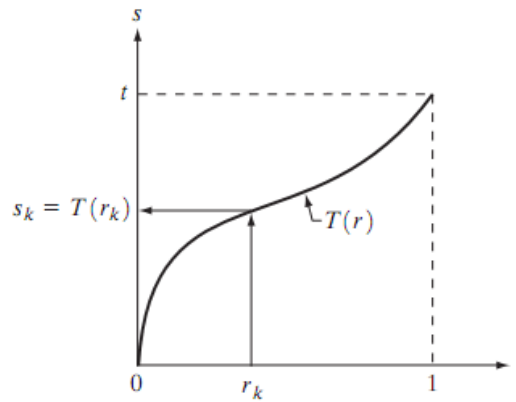
Figure 7: gray-level transformation function that is both single valued and monotonically increasing.[3]

Let the variable *r* represent the grey level of the pixels in the image to be enhanced. Assume that the pixel values are normalized to lie in the range $0 \leq r \leq 1$ with $r = 0$ represents black , $T(r)$ represents white in the gray scale. For r_k , we consider the transformations of the

$S_k = T(r_k)$ which produce a level S_k for every pixel value r_k in the original image. It is assumed that the transformation function satisfies the conditions:

(1) $T(r_k)$ is singled valued and monotonically increasing in the interval { $0 \leq r \leq 1$ };

$$(2) \ 0 \leq T(r_k) \leq 1 \ for \ 0 \prec r_k \leq 1$$

One of the most important nonlinear point operations is histogram equalization, also called histogram flattening. The idea behind it extends that of FSHS: not only should an image fill the available gray scale range, but it should be uniformly distributed over that range. **[3][6][5]**

2.3 Basic Spatial Filtering (Mask processing)

A filter is a special kind of tool designed to take an input layer or image, apply a mathematical algorithm to it, and return the input layer or image in a modified format. Enhance filters are used to compensate for image imperfections. Such imperfections include dust particles, noise, interlaced frame and insufficient sharpness.
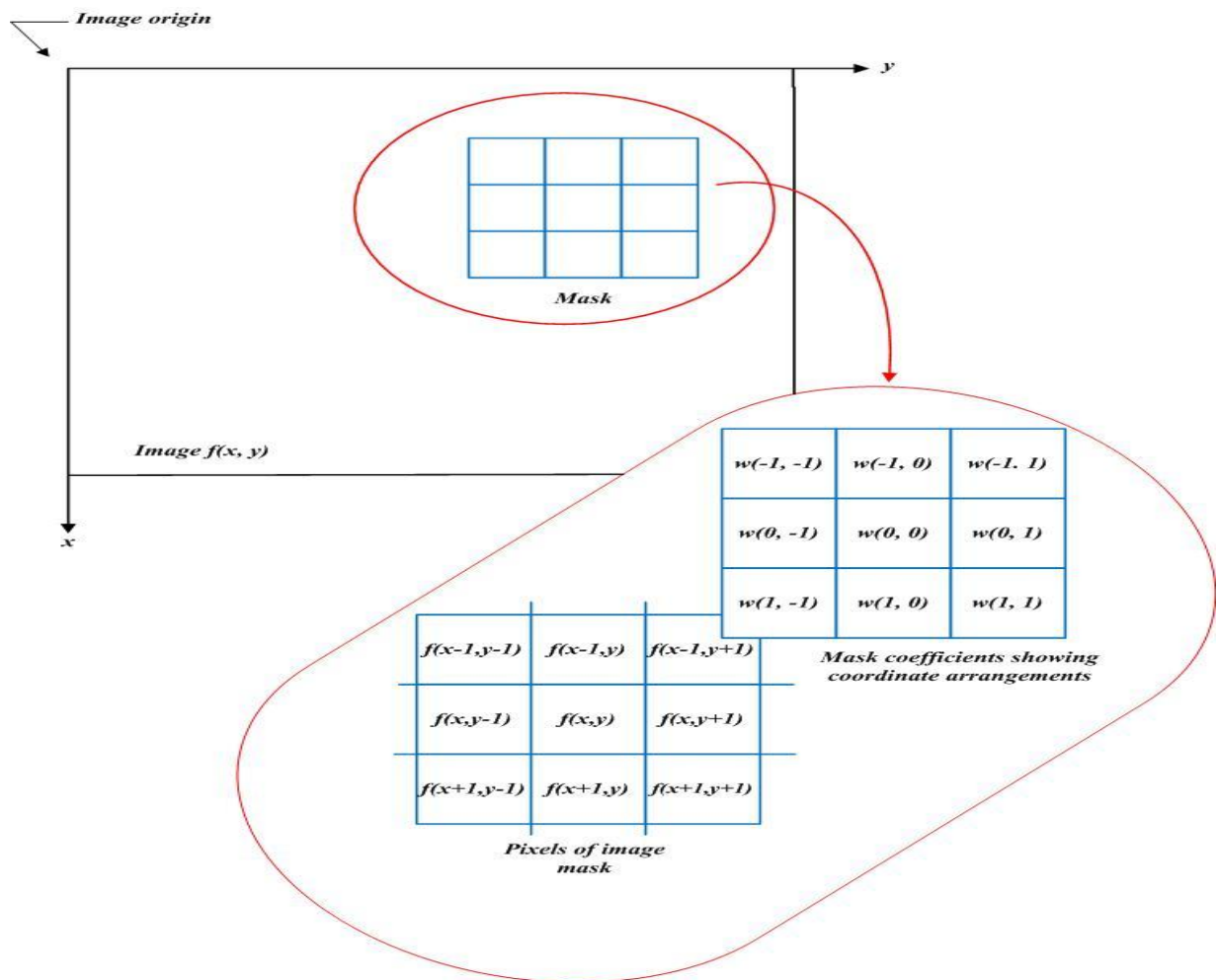

Figure 8:The mechanics of spatial filtering.[3]

2.3.1   Smoothing spatial filters

The Smoothing filters in the spatial domain can be used for blurring and for noise reduction. In processing; blurring has basic steps, as removing the small details from the input images, extraction, and bridging of small gaps in lines or curves. Blurring linear filter and the non-linear filters, can accomplish the noise reduction. **[3]**

2.3.2   Sharpening spatial filters

Sharpening is one of the most wonderful conversions; it brings an output image details were not clear before. When mention the term "sharpening ", we usually recall edges, essentially it is to emphasize edges and make them more prominent so they will be easier for the eye to pick out, and no new details are added or created, some people they consider that sharpening is a simple illusion.

The pixel averaging in the neighborhood in the spatial domain can accomplish the image blurring in the, since averaging is analogous to integration, which mean that sharpening could be accomplished in the spatial domain. This is in fact, the case in this section deals with different ways of defining and implementing operators for sharpening. Fundamentally, the strength of the response of a derivative operator is proportional to the degree of discontinuity of the image at the point at which the operator is applied. Thus, image differentiation enhances edges and other discontinuities and deemphasizes areas with slowly varying gray-level values. **[3]**

3 Frequency domain techniques

Frequency domain methods are based on modification of Fourier transform of an image. Let $g(x, y)$ be an image formed by the convolution of an image $f(x, y)$ and a position invariant operator $h(x, y)$,

$$g(x, y) = h(x, y) * f(x, y)$$

From convolution theorem, we have

$$G(u, v) = H(u, v) \cdot F(u, v)$$

Where *G, H F* are *F T* of *g, h* and *f* respectively. The transform $H(u, v)$ is referred to as the transfer function of the process.

In the frequency domain relation the discrete convolution is often more efficiently the using fast Fourier transform algorithm

In a typical image enhancement problem $f(x, y)$ is given and the goal after computation of $F(u, v)$ is to select $H(u, v)$ so that the desired image given by:

$$g(x, y) = F^{-1}\{H(u, v)F(u, v)\}$$

It also exhibits some highlighted features of $f(x, y)$ for example edges in $f(x, y)$ can be accentuated by using a function $H(u, v)$ which emphasizes the high frequency components of $F(u, v)$.**[3][6]**

3.1 Smooting Frequency-Domain Filters

Edges and other sharp transitions in the gray levels of an image contribute significantly to the high-frequency content of its Fourier transform. Hence smoothing (blurring) is achieved in the

frequency domain by attenuating a specified range of high-frequency components in the transform of a given image.

We consider three types of low-pass filters: ideal, Butterworth Gaussian filters. These three filters cover the range from very sharp (ideal) to very smooth (Gaussian) filter functions. **[1]**

3.1.1   Ideal low-pass filter

The simplest low-pass filter we can envision is a filter that "cuts off" all high-frequency components of the Fourier transform that are at a distance greater than a specified distance, from the origin of the (centered) transform. Such a filter is called a two-dimensional ideal low pass filter and has the transfer function:

$$H(u,v) = \begin{cases} 0 & if\ D(u,v) \le D_0 \\ 1 & if\ D(u,v) > D_0 \end{cases}$$

D_0 is specified as a nonnegative quantity, and $D(u,v)$ is the distance from the point to the center of the frequency rectangle. **[3]**
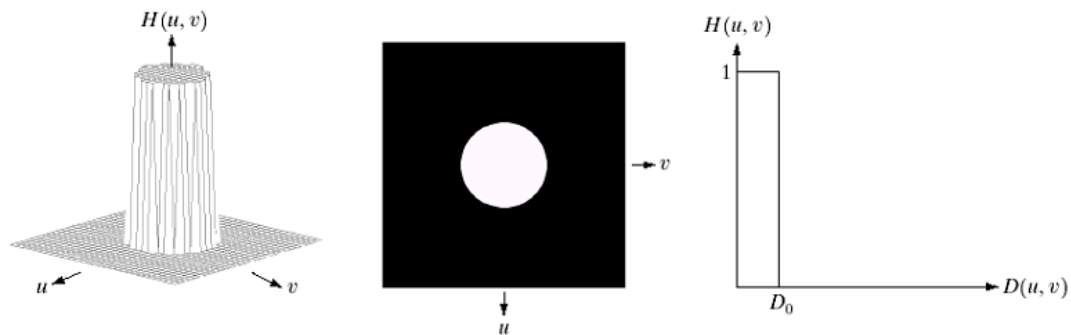
Figure 9: (a) Perspective plot of an ideal low-pass filter function. (b) Filter displayed as an image. (c) Filter radial cross section.[3]

3.2   Sharpening Frequency Domain Filters

To sharpen the image in the frequency domain, it can be accomplished by high pass filtering process which attenuates low frequency components without disturbing high frequency information in the Fourier transform. We intend in this filter is to perform precisely the reverse operation of the ideal low pass filter, the transfer function of the HPF can be expressed by this relation:

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

Where $H_{lp}(u,v)$ is the transfer function of the corresponding low pass filter, that's when: the low pass filter attenuates frequencies, the high pass filter pass them, and vice versa. So high-pass filter is a filter that passes high frequencies well, but attenuates (reduces the amplitude of) frequencies lower than the cutoff frequency. The actual amount of attenuation for each frequency varies from filter to filter; sometimes it's called a low-cut filter.

It is useful as a filter to block any unwanted low frequency components of a complex signal while passing the higher frequencies; high pass filter technique works so well at sharpening images is because any areas in the image which are not an edge are left untouched. The main areas that have sharpening applied to them are the edges, which is exactly what we need. In this section we concentrate on Ideal, Gaussian and Butterworth high pass filters. Butterworth filter represents a transition between the sharpness of the ideal filter and the total smoothness of the Gaussian filter, to illustrate of these filters. **[3]**
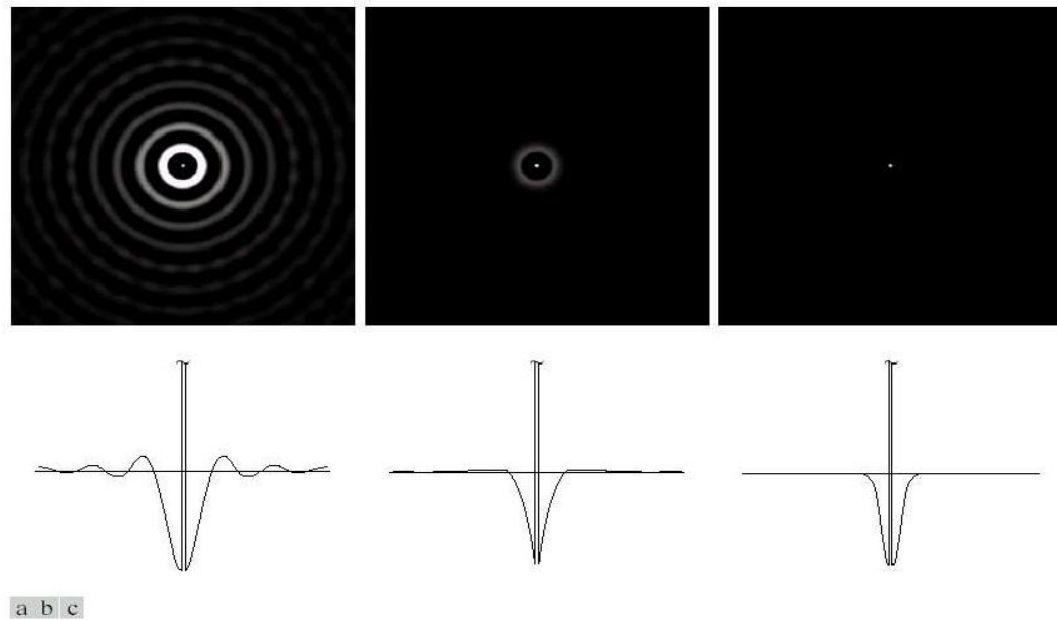
Figure 10: Spatial representation of typical (a) ideal. (b) Butterworth and (c) Gaussian frequency domain high-pass filters, and corresponding the grey-level profiles.[3]

3.2.1   Ideal high-pass filter

High pass filtering means that we filter away the low frequencies of something, and let the high frequency bands pass. In image terms, this means that the detail of an image is kept, while the larger scale gradients are removed.

Ideal high-pass filter IHPF is defined as:

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \leq D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

Where D_0 the cut off distance, is measured from the distance from the origin of the frequency rectangle, and $D(u,v)$ is the distance from the origin of the Fourier transform, u & v are frequency variables of the Fourier transform. This filter is the opposite of the Ideal low pass

filter that is zeroing all frequencies inside the circle of radius D_0 while passing without attenuation any frequency outside the circle. **[3]**

3.2.2   Butterworth high-pass fitler

The transfer function of the Butterworth high-pass filter (BHPF) of order n and cutoff frequency locus at distance D_0 from the origin is given by

$$H(\mu,v) = \frac{1}{1+[D_0 / \ D(\mu,v)]^{2n}}$$

As in the case of a low-pass filters, we can expect Butterworth high-pass filters to behave smoother than IHPFs. **[3]**

3.2.3   Gaussian high-pass filter

The transfer function of the Gaussian high-pass filter (GHPF) with cutoff frequency locus at a distance D_0 from the origin given by

$$H(u,v) = 1 - e^{-D^{2(u,v)}/2D_0{}^2}$$

The results obtained are smoother than with the previous to filters Even the filtering of the smaller objects and thin bars is cleaner with Gaussian filter. **[3]**

3.2.4   Unsharp Masking

The Unsharp Mask filter sharpens edges of the elements without increasing noise or blemish. It is the king of the sharpen filters. In this method, edges in image are being exaggerated, and many details are produced in the reproduction. In digital unsharp masking, adjoining pixel values

are evaluated to locate the edges. When an edge is detected, the software exaggerates the edges by altering the value in two adjoining pixels in opposite directions, thereby increasing the edge contrast. This technique commonly used in the printing industry for crispening of edges. A signal proportional to the unsharp or low pass filtered version of the image is subtracted from the image. This is equivalent to adding a high pass signal to the image:
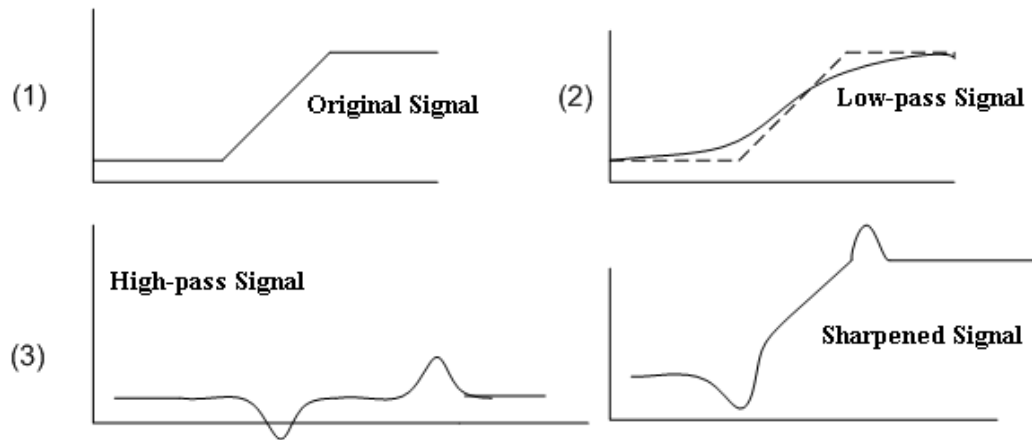

Figure 11: Unsharp Masking Operation. [1]

3.2.5   The Laplacian filtering

It is known for a single image to contain edges having big different sharpness. Edge of them is useful for understanding the image. For instances, edges at low resolution tend to indicate gross shapes, whereas texture tends to become important at higher resolutions." An edge detected over a wide range of scale is more likely to be physically significant in the scene than an edge found only within a narrow range of scale. Furthermore, the effects of noise are usually most deleterious at the finer scales". **[3][6]**

The Gaussian smoothing operation serves to band-limit the image to a small range of frequencies, reducing the noise sensitivity problem when detecting zero crossings. The image is filtered over a variety of scales and the Laplacian zero crossings are computed at each. This

produces a set of edge maps as a function of edge scale. Each edge point can be considered to reside in a region of scale space, for which edge point location is a function of x, y and s. Scale space has been successfully used to refine and analyze edge maps. **[6]**

The Gaussian has great properties that make this edge detection procedure unique. First, the Gaussian function is in the spatial and frequency domains, giving a good compromise between the need for avoiding false edges and for minimizing errors in edge position, and the Gaussian is the only function which can minimize the product of both spatial and frequency domain. The Laplacian of Gaussian essentially acts as a bandpass filter because of its smoothing properties. Second, the Gaussian is separable, and this helps us to make an efficient computation. **[6]**

Omitting the scaling factor, the Gaussian filter can be written

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

Its frequency response $G(\Omega_x \Omega_y)$ is also Gaussian:

$$G(\Omega_x \Omega_y) = 2\pi\sigma^2 \exp\left(-\frac{\sigma^2}{2}\left(\Omega_x^2 + \Omega_y^2\right)\right)$$

The σ parameter is inversely related to the cutoff frequency.

Both of the convolution and Laplacian operations are linear and shift invariant as well, so their computation order can be interchanged.

$$\nabla^2\left[f_c[x, y] * g_c(x, y)\right] = \left[\nabla^2 g_c(c, y)\right] * f_c(x, y)$$

So we know that the derivative is a linear operator, and Gaussian filtering following by differentiation is the same as filtering with the derivative of a Gaussian. This part of the equation $\left[\nabla^2 g_c(c, y)\right]$ is usually providing an efficient computation since preparing it in advance as a

result of its image independence. The Laplacian of Gaussian (LoG) filter, $h_c(x,y)$ therefore has the following impulse response. **[6]**

$$h_c(x,y) = \nabla^2 g_c(x,y) = \frac{x^2 + y^2 - 2\sigma^2}{\sigma 4} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

3.2.6   High-boost filtering

Often it is desirable to emphasize high frequency components representing the image details without eliminating low frequency components. In this case, the high-boost filter can be used to enhance high frequency component while still keeping the low frequency components
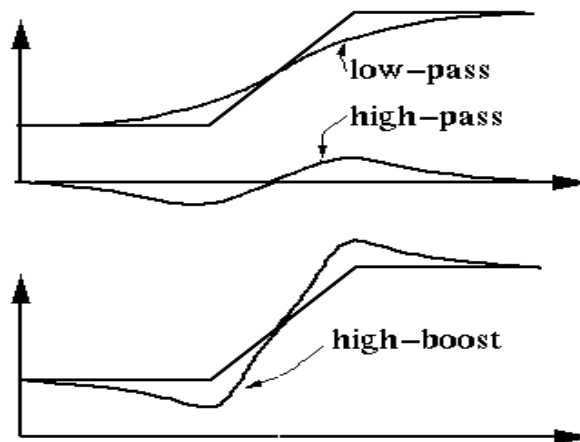

Figure 12: showing how High-Boost Filtering technique works.

Unsharp masking consists of generating a sharp image by subtracting from an image a blurred version of itself. Using frequency domain terminology, this means obtaining a high-pass filtered image by subtracting from the image a low-pass filtered version of itself, and that is:

$$f_{hp}(x,y) = f(x,y) - f_{lp}(x,y)$$

High-boost filtering generalizes this by multiplying $f(x,y)$ by a constant $A \geq 1$:

$$f_{hb}(x,y) = Af(x,y) - f_{hp}(x,y)$$

Thus, high boost filtering gives us the flexibility to increase the contribution made by the image to the overall enhanced result:

$$f_{hb}(x,y) = (A-1)f(x,y) + f(x,y) - f_{lp}(x,y)$$

Or,

$$f_{hb}(x,y) = (A-1)f(x,y) + f_{hp}(x,y)$$

This result is based on a high-pass rather than a low pass image. When A =1, high boost filtering reduces to regular high-pass filtering. As 'A' is the increase past 1, the contribution made by the image itself becomes more dominant. **[1][3]**

4 Color image enhacement

The human eye can distinguish thousands of different colors, visible colors usually occur in the range between 400nm (violet) and 700nm(red) as it's shown on the electromagnetic spectrum in figure below:
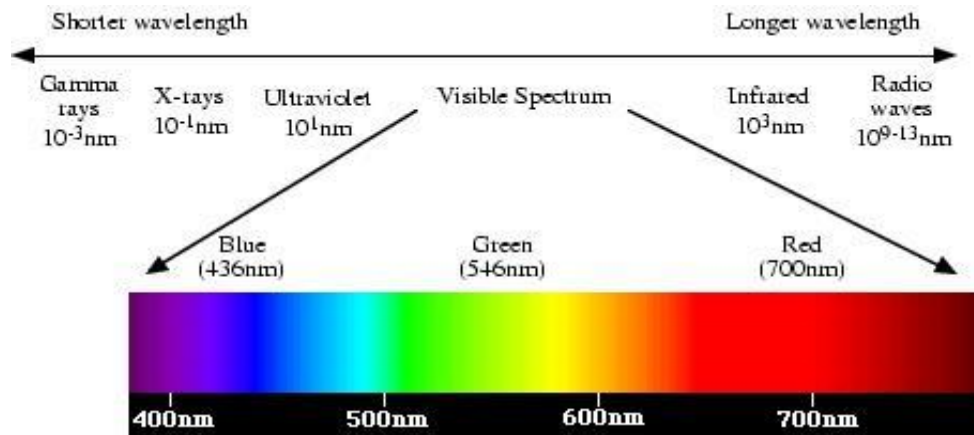

Figure 13: The visible spectrum. [11]

Colors are usually visualized to the human eye as the primary colors of light which are red, green and blue, and the characteristics of color to distinguish one from another are brightness which is the intensity of each color, hue which is determined by the dominant wavelength and saturation that can be determined by the amount of the color intensity and the hue. **[11]**

4.1 Color Models

The use of the color models is to qualify and specify colors in some standard accepted way; the color models use the system of *3-D coordinate*, where each single point in the subspace of this system represents a certain unique color. There are many of the color models are in use today for example: the RGB color model (Red, Green, Blue), HSV color model (Hue, Saturation, Value), HSI color model (Hue, Saturation, Intensity) and CMY color model (Cyan, Magenta, Yellow); each of these models has its use which helps user to analyze and modify his image. **[10] [11]**

4.1.1   The RGB Model

The name comes from primary colors that each color appears as a combination of red, green and blue, that's why this model is called additive. To analyze each color is by specifying each amount of primary components which color presents, as we see in figure 14 the geometry of the RGB color model to represent each color in the Cartesian coordinate system, the values of RGB are sometimes assumed in the range of [0,1] and in other cases in the range of [0-255], by this way white is represented as (1,1,1) or (255,255,255) and black is represented in the opposite corner of the cube as (0,0,0), and the grayscale as it's shown in the figure the line between black and white vertices, and Red is the X-axis, Blue is the Y-axis, and Green is the Z-axis.
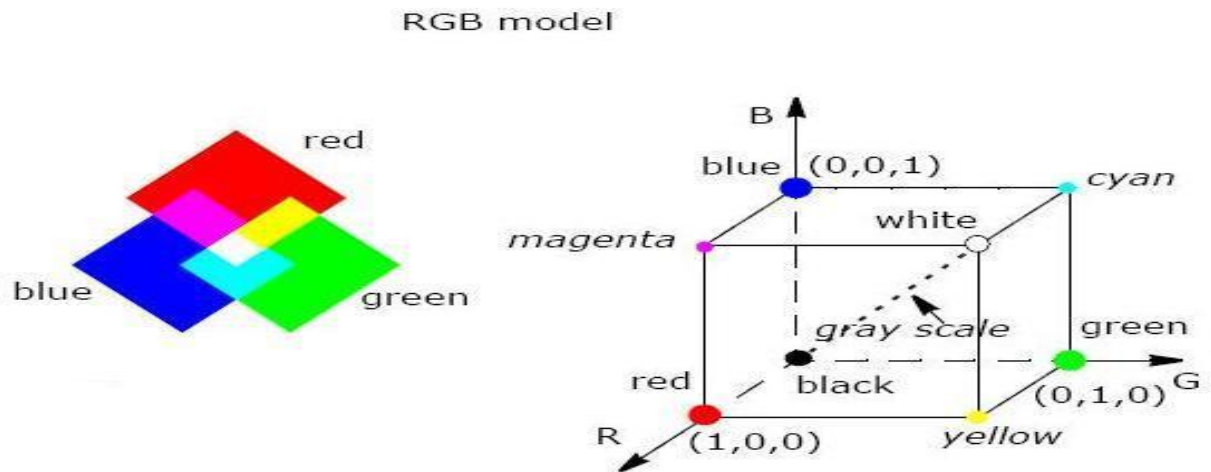
Figure 14: The figure on the left shows the additive mixing of primary RGB colors. The Figure on the right is RGB color cube. [15]

The Primary colors red green and blue can be mixed to each other to obtain the secondary colors, which are Magenta from read plus blue, Cyan which is blue plus green and yellow which is red plus green, as it's shown in figure 14, and by mixing the three primary colors we obtain white color. The RGB color space is the best known for color models, and the RGB model is the most widely used in digital imaging devices because it's the most important model in image processing. **[10][15]**

4.1.2   The CMY Model

This model is mostly used in printing color devices, the model's name comes from the secondary colors of the RGB model cyan, magenta and yellow; simply it's a subset of the RGB model, in the figure 15 we can see that by mixing the CMY colors we obtain red, green and blue colors which becomes the secondary colors of CMY model, and mixing same quantities of three color of cyan, magenta, and yellow we obtain black color, and to convert RGB to CMY is given by:

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

Which means that cyan is white minus red, magenta is white minus green and yellow is white minus blue, where 1 presents white color. **[10] [15]**
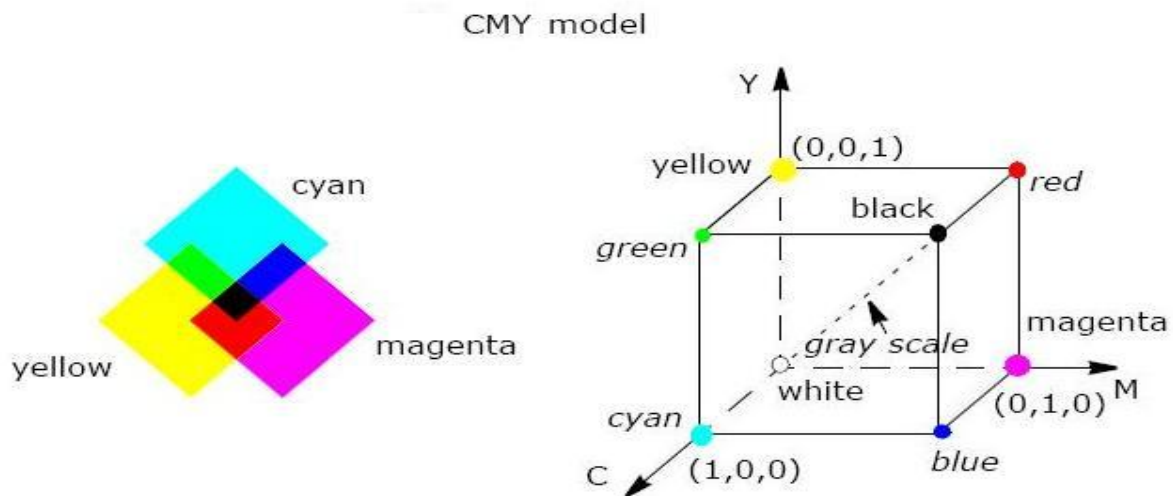
Figure 15: The figure on the left shows the mixing of primary CMY colors. The Figure on the right is CMY cube color cube. [15]

4.1.3   The HSI Model

The HSI model is special in image processing and it's useful for comparing and measring colors characteristics and to change color to another. The RGB and CMY models are not representing the colors in terms how it's perceived by the human eye, that's why the HSI model is an attractive model and very important in the processing applications because it represents the colors as the

35

human eyes sense them, in HSI model, colors are represented according to the characteristics of every color; hue H, saturation S and intensity I, we can see in figure 16 how the hue components can describe colors by the angel between [0,360], where 0 degree is red, 120 degrees is green, 240 degrees is blue, 60 degrees is yellow and magenta is at 300 degrees, the range of components for S and I is between [0,1] where 0 represents white color. **[14][15]**
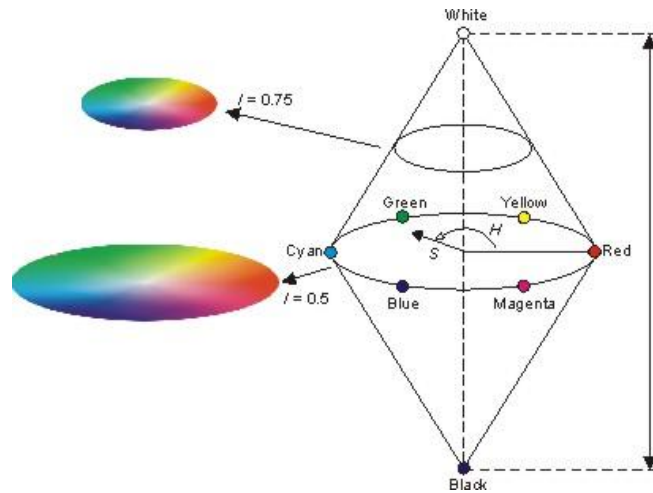

Figure 16: The figure illustrates how the HSI color space represents colors. [14]

In fact we can obtain the hue, saturation and intensity values from the color cube of RGB, by converting any RGB point to a corresponding HIS point, The intensity I is given by;

$$I = \frac{1}{3} \ (R + G + B)$$

And the saturation S is give by;

$$S = 1 - \frac{3}{R + G + B} [\min(R, G, B)]$$

4.2 Natural Color Image Enhancement

The monochrome image enhancement methods could be applied to natural color images by processing each color component individually. This comprises the class of intra component processing algorithms. There is also a class of inter component processing algorithms in which

color pixels are combined on a pixel-by-pixel basis. Finally, there is a class of vector processing algorithms. **[5]**

4.2.1   Intra component Processing

Typically, color images are processed in the *RGB* color space. This approach works quite well for noise cleaning algorithms in which the noise is independent between the *R, G* and *B* components. Edge crispening can also be performed on an intra component basis, but better, and more efficient, results, are often obtained by processing in other color spaces. Contrast manipulation and histogram modification intra component algorithms often result in severe shifts of the hue and saturation of color images. Hue preservation can be achieved by using a single point transformation for each of the three *RGB* components. For example, form a sum image, and then compute a histogram equalization function, which is used for each *RGB* component.

For some image enhancement algorithms, there are computational advantages to processing in a luma-chroma space, such as, or a lightness-chrominance space. **[5]**

4.2.2 Inter component Processing

"The intra component processing algorithms previously discussed provide no means of modifying the hue and saturation of a processed image in a controlled manner. One means of doing so is to transform a source *RGB* image into a three component image, in which the three components form separate measures of the brightness, hue and saturation (*BHS*) of a color image. Ideally, the three components should be perceptually independent of one another. Once the *BHS* components are determined, they can be modified by amplitude scaling methods"**. [5]**

5 Matlab implementation

As I mentioned before, the task of this project is divided into two parts; theoritical part and practical part which includes the image processing toolbox in Matlab, and my work on graphical user interface GUI in Matlab. Matlab has a lot of functions for image processing, so it depends on the user which technique is useful for the process needed for the image, Matlab is very creative program and have intersting reslut sepcially when the used data is very complex so it can make an intersting results. Toolbox is a great tool for creating mathematical operations, and toolbox for image processig is the best help for the user in language programming, in the program as well the GUI is used to creat develop to the program.

5.1 Image Processing Toolbox in Matlab

Image Processing Toolbox is a large set of algorithms that deal with images, and it's almost supporting all the types of images, which gives the users so many options for processing of images. A lot of functions are responsible for image enhancement for example: deblurring, filtering, contrast, spatial transformations, adjustment of color balance, creating histograms, hue and saturation, and as well for the detection of objects.

"All descriptions are based on the website www.mathworks.com, which provides wide compendium of knowledge about all Matlab functions, including those from Image Processing Toolbox. First group of operations is responsible for changes and information concerning color transformation of images. Couples of functions can't change anything in the image but they are crucial when it comes to gain information about it, without need of opening the actual object of interests. Isbw(I) returns value 1 if the image is black&white, and value 0 otherwise. Some operations have sense only when executed on binary graphic files. For example adjusting contrast, brightness or other changes, usually made on colorful pictures, would not work with black&white images. Function isgray(I), similarly to previous one, checks colormap of the image. As the name suggests, this time function returns value 1 if the picture is grayscale and

value 0 otherwise. It may also become useful while deciding if some operations can be performed on the file. Isrgb(I) informs if examined file is the RGB image. These three functions are essential when it comes to deciding about changing the colormap or color system. Knowing if the image is black&white, grayscale or RGB determines what transformations can be done to the file. There would be no point to try having some changes to image, if they are inoperative for some color models or maps. Command colormap 'map' is connected with the previously mentioned however it is not Image Processing Toolbox function. It exists in Matlab main library. Current image colormaps are set to one that stands in the brackets as a parameter. There is about twenty ready-built colormaps in Matlab." **[2]**

### 5.1.1   Open, Save, and Display functions

To handle image processing in Matlab, there are some things we need to keep in mind; such as opening, closing, displaying and saving an image. In addition we should know that library of Matlab has many useful commands that we will need while working on images on Matlab, 'Imread' for example deals with reading images from graphics file, it takes name of the file and it's extension as needed parameters in brackets, format as well which are supported by Matlab for example bmp, jpeg, png and tiff. The command '*Imread*' can return the two-dimensional array when image is a grayscale, and if the image is color it returns it to three-dimensional one.

'Imwrite' is a command which can write image to the graphics file. Another great useful function is the 'Imshow', which is responsible for displaying the color, grayscale and black&white images. 'Imfinifo' is a command that shows the different information of image, such as filename, size, format, width, number of bit by pixels and color type of the image. **[9]**

One of the most complex functions image processing is the function 'Fspecial', this parameter takes one value to determine the type of filter interested to use, for example 'disk', 'motion' and 'unsharp'. Filter 'disk' creates circular averaging and result will be a blurred image and user specifies the radius, the 'motion' filter is similar to the linear motion of cameras, and the

'unsharp' filter is useful for sharpen the blurred images. In figure 17 we can see the different effects of the filters of fspecial on image.
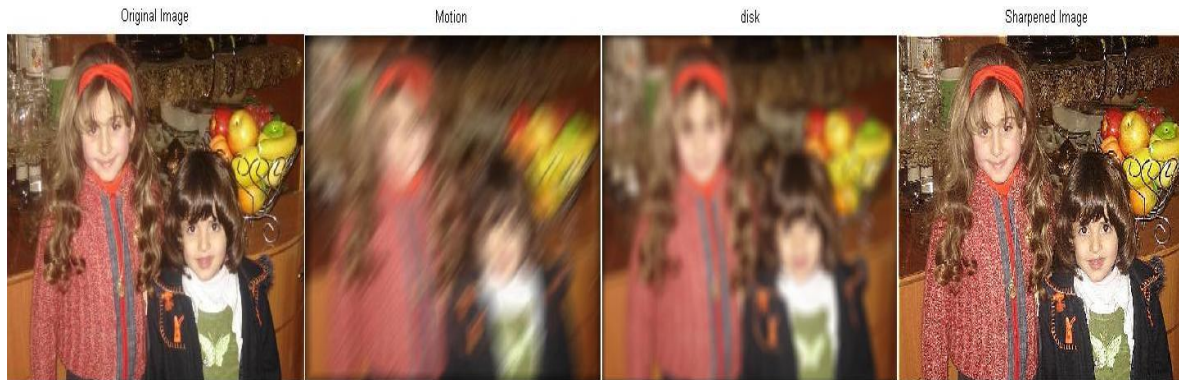

Figure 17: Effect of fspecial filters on image, 'motion' and 'disk' and 'unsharp'.

5.1.2   Graphical user interface (GUI)

Graphical user interface is a graphical display in one or more windows contaings controls, so it's a group of build techniques help to make a fast comminication between the user and program, which helps user after to creat the task easier and faster. In Matlab we have a very helpful tool called GUID, after clicking on the guide button in Matlab appears a start window, then it's recommended to choose 'Blank GUI', then appears for us new window where we can design the GUI we need, there it's possible to create the object we need by choosing the button we need from the left side list from the created figure, the list obtains pushbutton, axes, static, slider, radio button, check box, and edit texts. **[9]**

After saving any result of the created GUI the GUID save as two files .fig file which has description of the graphic part, and .m file, where we can find the code which controls the actions. Usually actions can be changed in the .m file, they are called 'callbacks', and each component has a 'Tag' property that refers the name of the callbacks.
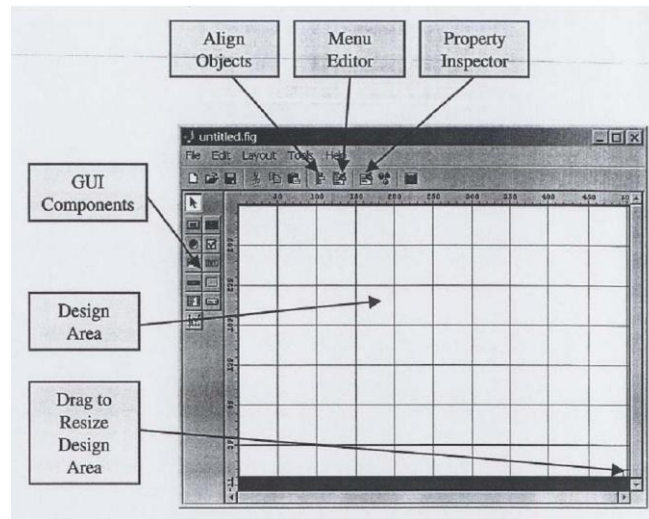
Figure 18: Guide tool window. [13]

These components are called 'uicontrols', which also contain different of properties for settings, a window named Property Inspector appears after creating object in Guide, which contains elements to set characteristics of every component such as color, the displayed text and so on.
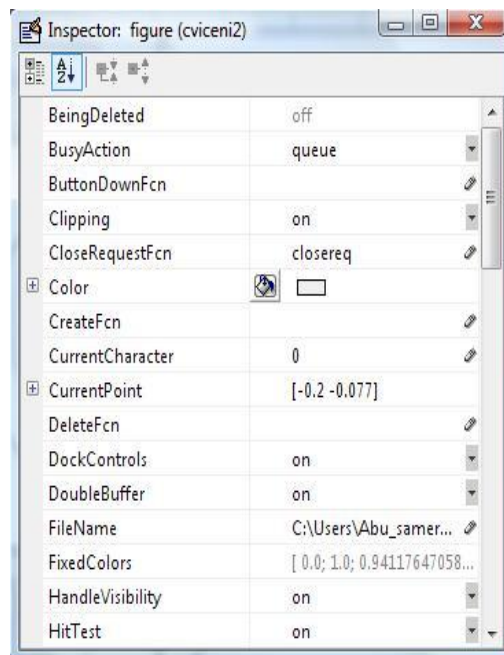
Figure 19: Property Inspector.

Pushbuttons allow users to deal the with program in GUI in easy and simple level , and sliders which are important as well, in GUI that helps user to change brightness ,contrast and value of gamma for example.

Axes is the area which will be used as display for input and output data and it can be created by the function axes, and user can changes it's settings by inspector for axes.

Every application on GUI should have a menu bar to help the user select the option among list of option of techniques, to create a menu bar is from the tool popup menus, and we can change the 'lable' and 'tag' from the menu editor, this tool help users to make the GUI more easier and intuitive, to change settings of menu and it's elements it can be from property inspector for the menu.

Radiobutton is a button that generates its function when it's checked, so it has two states on and off, and by clicking mouse on it it's activated.

Panel is to put some relative components in one group to make them easier for the user to understand the purpose of the components for example images properties. **[9][2]**

5.2 Working on GUI:

In my work of GUI in Matlab my aim was to design an application that can generate some of the techniques I studied about image enhancement for example histogram and imadjust, also to read the RGB and grayscale components of images. The GUI I designed is a simple one, that you load your image and then chose the function you need from the menubar and the possibility of saving the result afterwards.

So at first I create the menu from the Pop-up Menu, then to edit from the menu editor, and added two menus to it, first column is 'file', it contains three options 'load an Image', 'save' and 'exit', the second column is 'functions', it contains two techniques 'Histogram' and 'Im.adjust', as it's shown in figure 20.
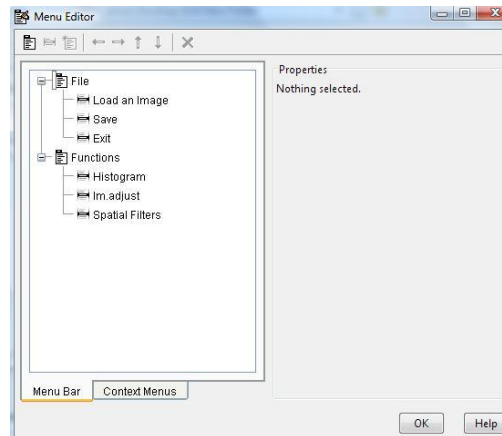
Figure 20: Menu Editor.

As it's shown in figure 20, we can see three axes, first one is named as 'axes_orig' and it's for the input image, and second is named as 'axes_disp' and it's for the output image, and the third axes which is named as 'axes-hist', it reads the histogram of the image. I also created two panels, first one is for 'images properties', Width, ColorType and Height, the second panel which is named by 'Display' includes four radiobuttons, they are Grayscale, RGB, R, G and B, and it's for the way we want the program to read and display the image. I created five horizontal sliders as well, and they are for input the parameters of the imadjust function, which are Gamma, Low In. Low Out, High In and High Out.
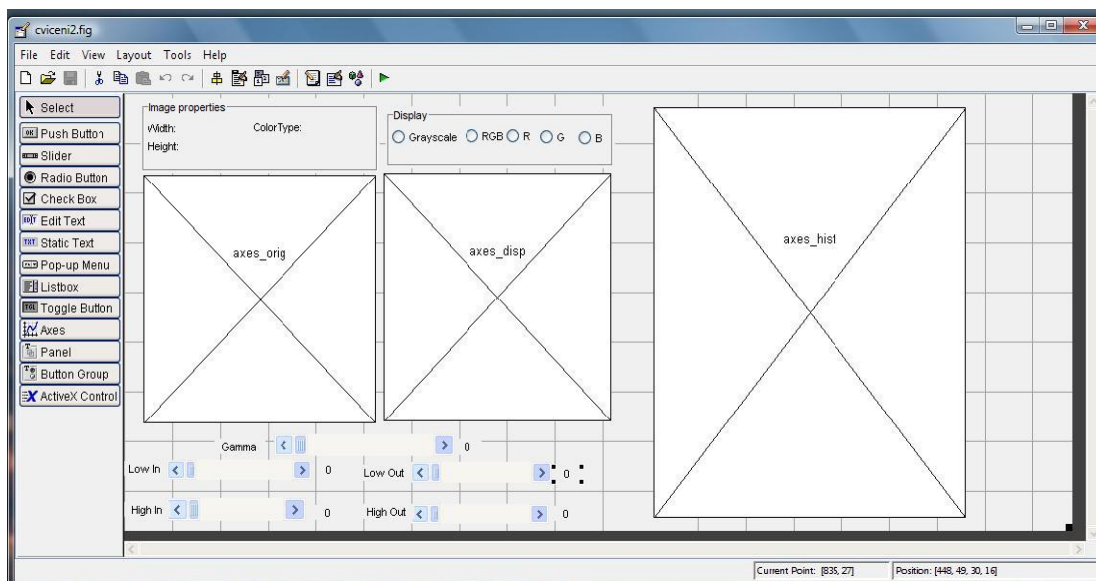

Figure 21: Building GUI in Matlab.

By running the designed GUI from the 'run', we can get our built GUI as it's show in figure 23, in the same figure it's show how to load the image by selecting 'load an image' from the file column of the menubar.
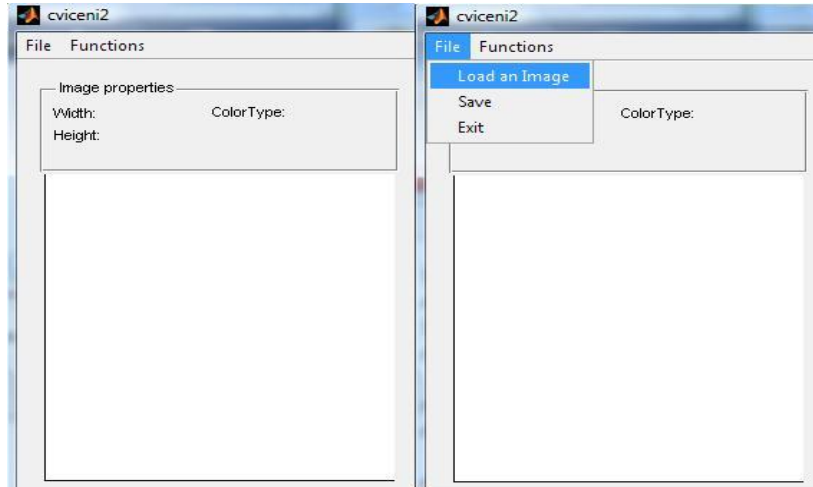

Figure 22: Simple GUI built in Matlab, the right figure shows how to load the image.

Loading the input image to the program, first it reads its properties, in the upper panel of image properties. So to apply the functions effects on the image, we select the function we need from the 'Functions' column of the menubar, for example if the user select the option of 'Histogram', it appears the second axes 'axes_disp' and the 'Display' panel,
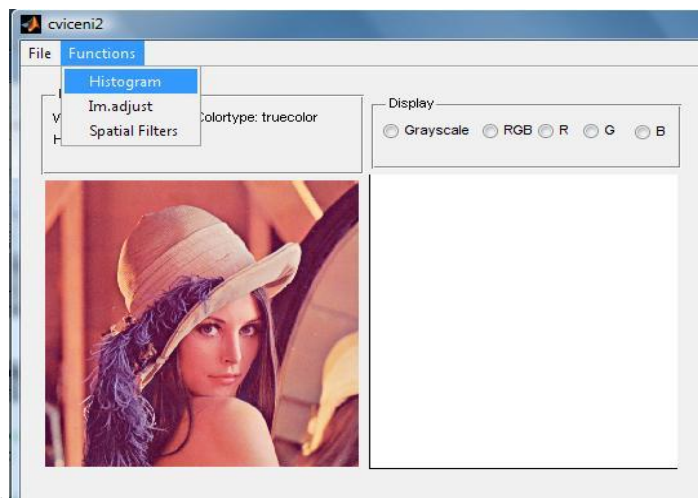
Figure 23: Histogram function.

by selecting any of the display options of the radiobutton, grayscale, RGB, R,G or B, the program will read components of image according to option will chosen, and will show the effect on the image in the display axes, then the third axes appears 'hist-axes' which reads the histogram of the output images in the second axes.
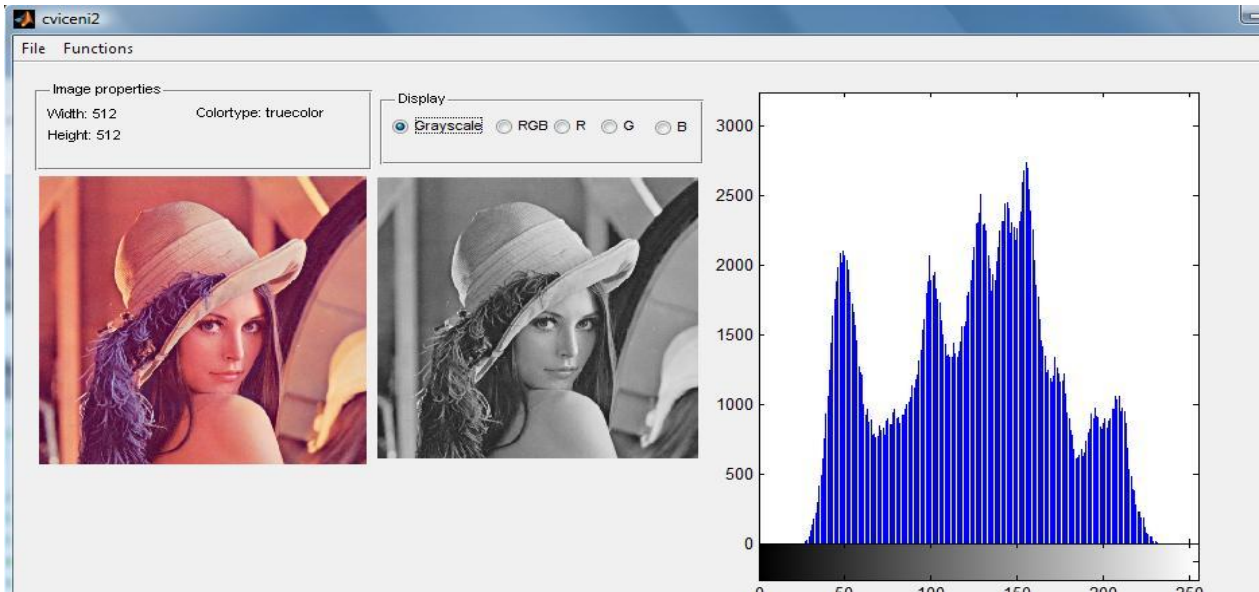
Figure 24: An example of the Grayscale and its Histogram.

If the user selected the function 'Im.adjust' from the Functions column, it will appear five changeable parameters of the function imadjust as horizontal sliders, which are gamma, low_in, low_out, high_in and high_out, so the user can change in parameters and can output image in the second axes on the right side of the input image, last thing is the command save from the 'file' menu, the user can save the output enhanced image after the using the techniques and apply it on the input image, as shown in figure 25.
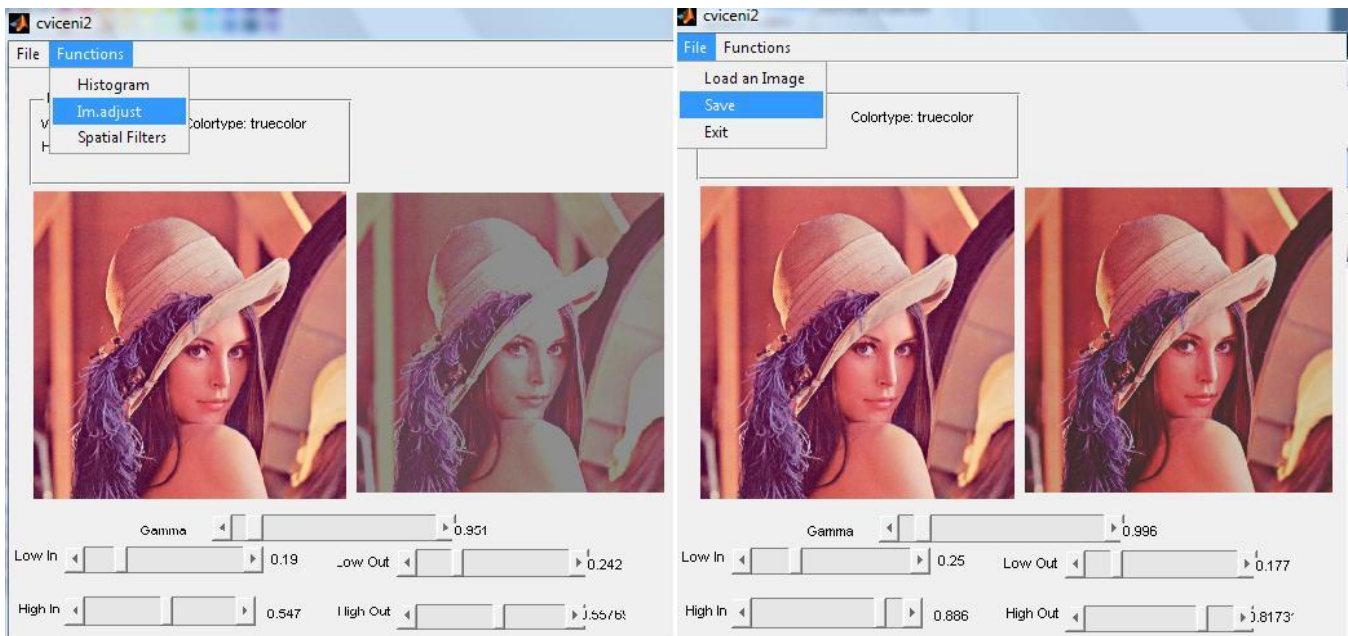
Figure 25: 'Im.adjust' function and its effect on image with different parameters.

5.2.1   Histogram method

I will use here the normal histogram to an input image; the figure shows a simple histogram technique in Matlab to the input image and then showing the output image and histogram for both of them.
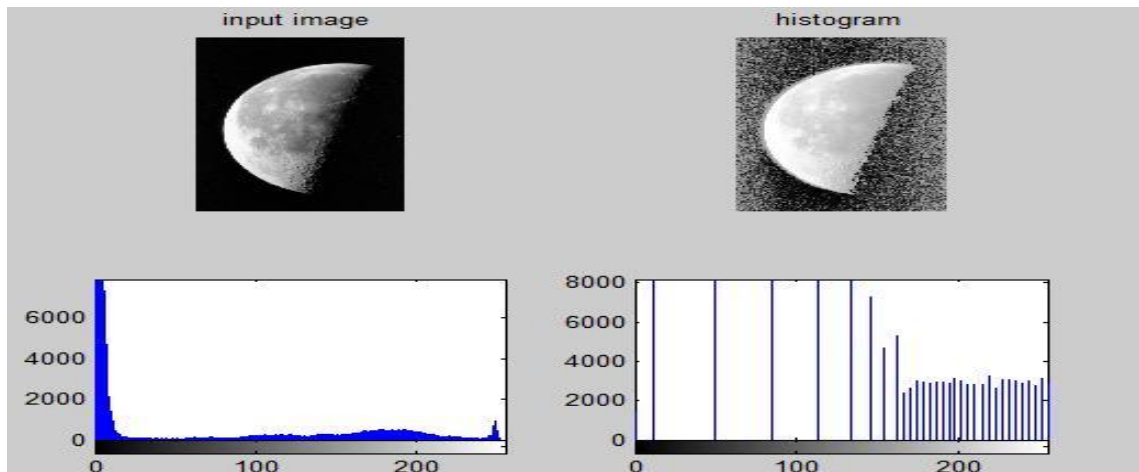

Figure 26: Histogram method on image.

Now I'm going to use the technique imadjust for the grayscale image, by changing only the gamma parameter, to see the different results of output images and their histograms by different values of gamma, so we don't add values for low_in, high_in, low_out, and hight_out, if we already know that the coode of imadjust in Matlab is:

J = imadjust(I,[low_in; high_in],[low_out; high_out],gamma) **[2]**

Where 'I' is the input image, we can see in Figures 27,28 and 29 the results for output imges and their histograms by changing values of gamma 2, 0.6 and 1. And we can notice that by setting gamma to 1, the output image is the same as input image, and different results were when gamma was set less than 1 so we obtain brighter output image as it's shown in Figure 28, and when gamma was set greater than 1, the result was darker output image as shown in Figure 27.
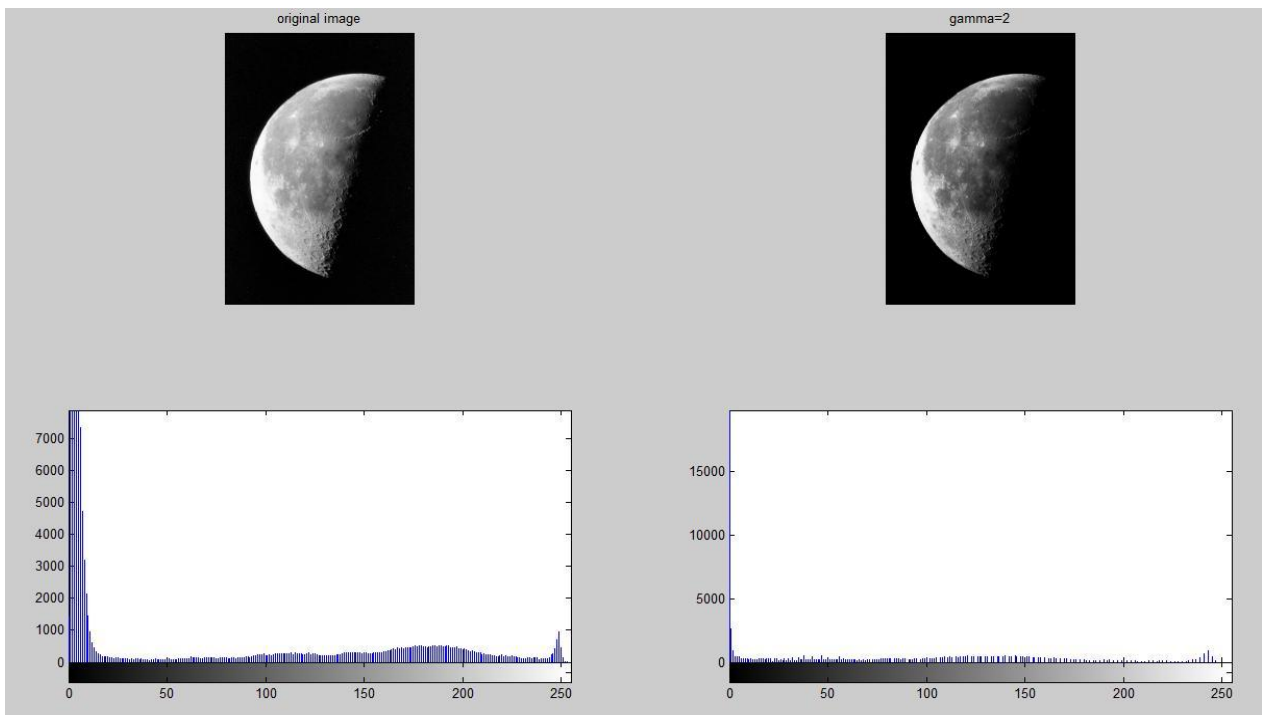
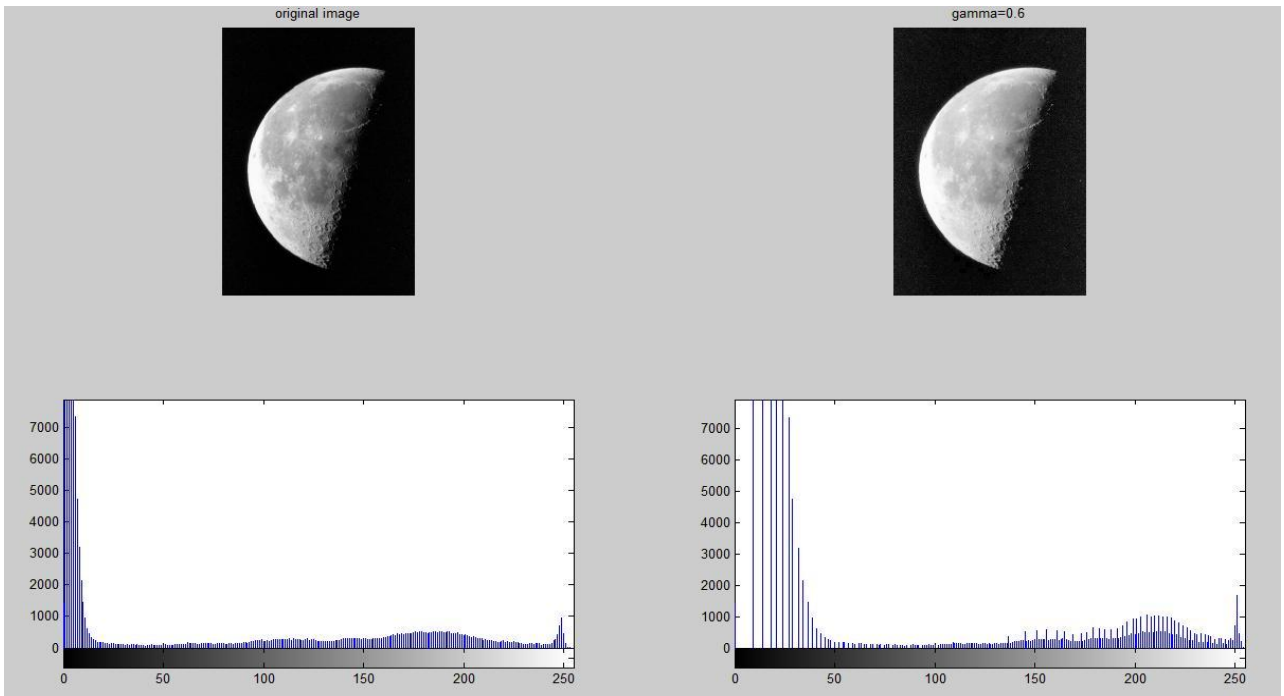**Figure 27:Imadjust with gamma=2 ,J = imadjust(I,[],[],2)**

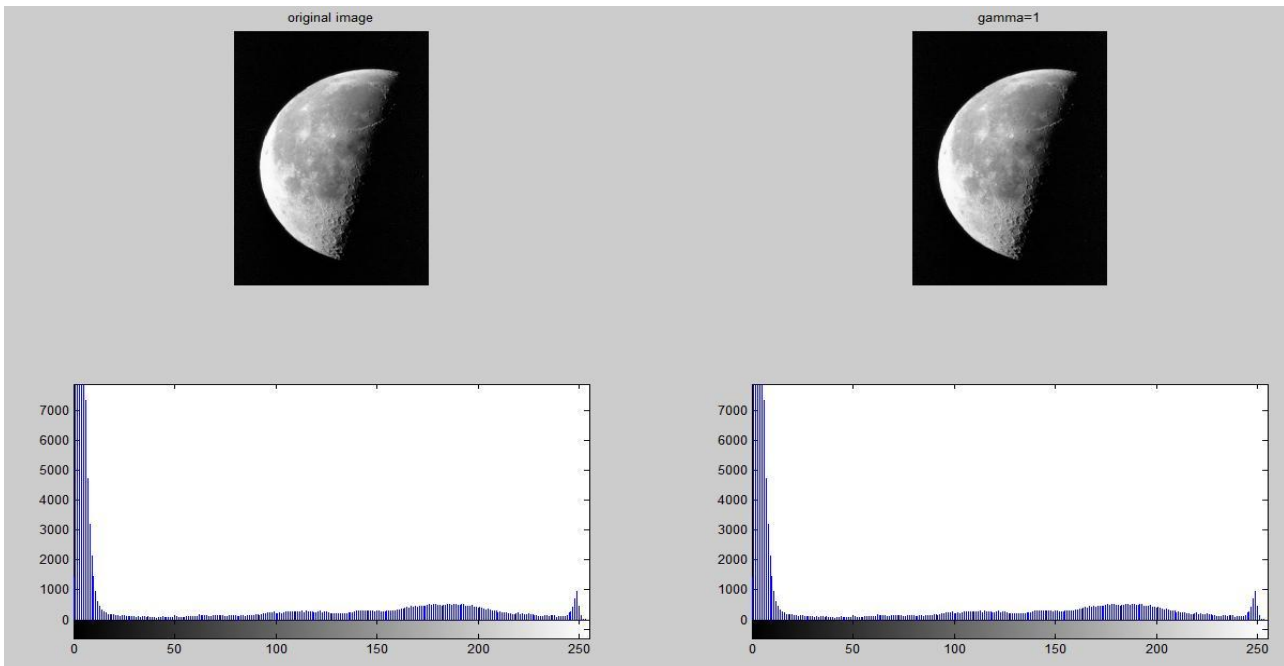**Figure 28: imadjust with gamma=0.6, J = imadjust(I,[],[],0.6)**

Figure 29: imadjust with gamma=1, J = imadjust(I,[],[],1)

5.2.2   Color Image Adjusting

Here I just applied the function of 'imadjust' for an input color image, and as we see the difference in the colors between the four output images with different values of the imadjust parameters, we notice how the imadjust technique perform four unique result by reform the RGB components of the input image in every time the user change four parameters low_in, high_in, low_out, and hight_out.
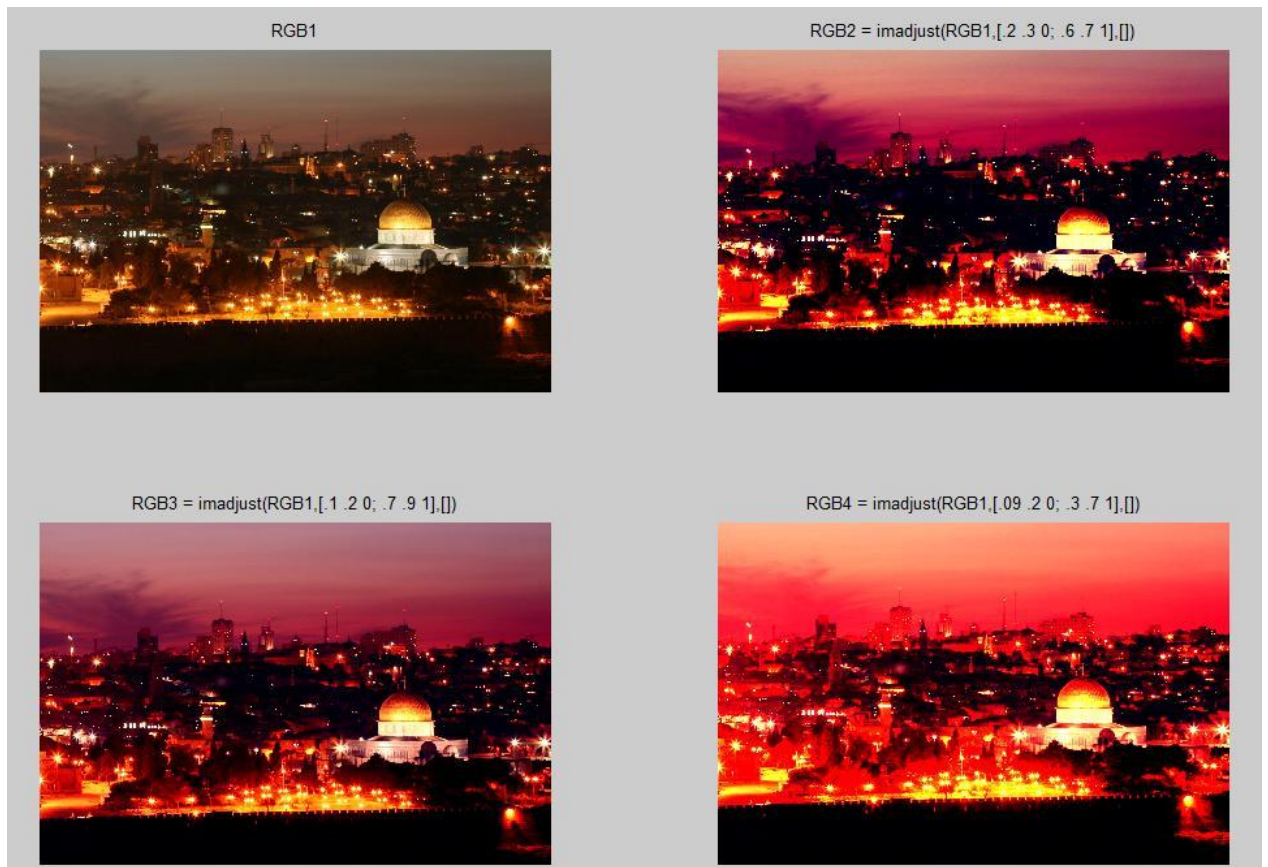

Figure 30: four different results of imadjust on input image.

5.2.3　F-special Filters

As mentioned in the beginning of the chapter that fspecial are from the complex functions and it create filters can be used in the image processing, fspecial creates 2D(two dimensional) special filters, with given relation h=fspecial (*type*), where fspecial returns h as a correlation kernel, and every filter can be used with the help of the command imfilter in Matlab, and some of these types of filters we will use and apply them are these values, the Laplacian of Gaussian 'log', horizontal edge-emphasizing and averaging filter, as shown in Figure 31. **[2]**
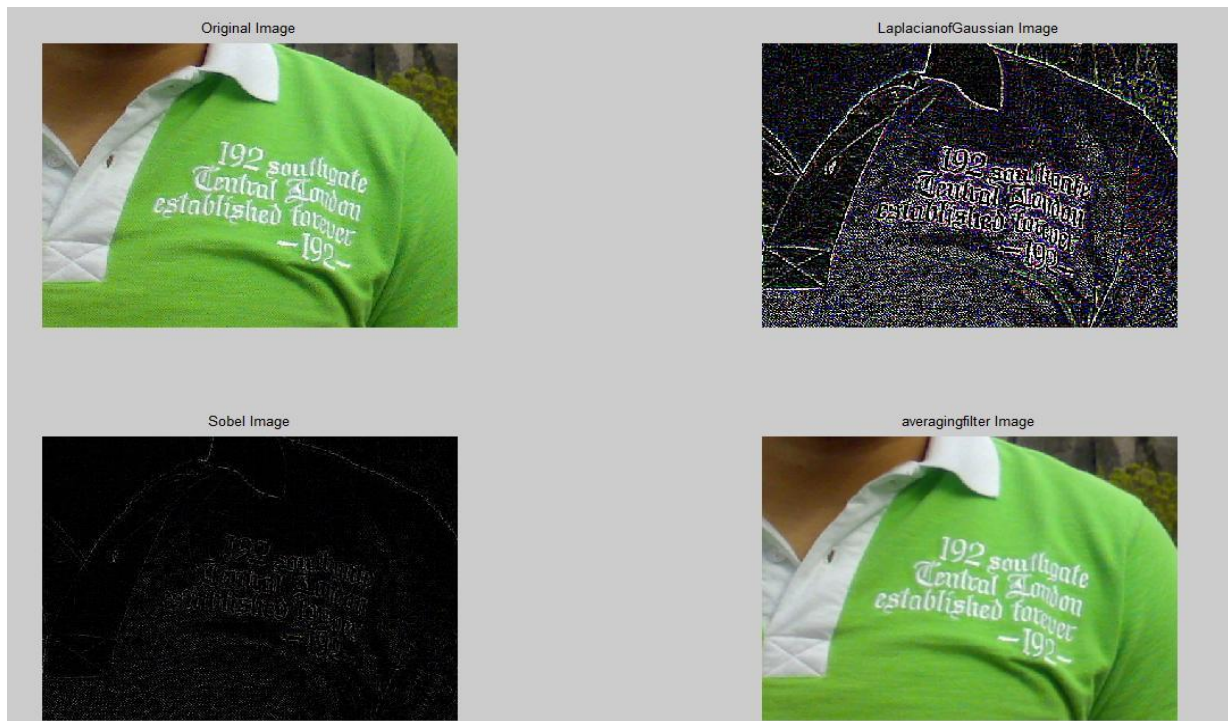

Figure 31: fspecial filters 'log','sobel'and 'average'.

Codes of Matlab of fspecial filters for the result above in the Figure 31 are illustrated below,

```
subplot(2,2,1);imshow(I);title('Original Image');
h = fspecial('log',[5 5],.3);
LaplacianofGaussian= imfilter (I,h);
subplot(2,2,2);imshow(LaplacianofGaussian,[]);title('LaplacianofGaussian
Image');
h = fspecial('sobel');
Sobelim = imfilter(I,h);
subplot(2,2,3);imshow(Gaussianlowpass,[]);title('Sobel Image');
h = fspecial('average',[3 3]);
averagingfilter = imfilter (I,h);
subplot(2,2,4);imshow(averagingfilter);title(h=fspecial('average',[3 3]');
```

5.2.4   Color Image Enhancement:

The next final application in my work, was to perform simple implementation, to apply it for an input color image to get ehnacened better result outpout image for the user, I used this pepper color image as an input.

Figure 32: input color image

The process of color image enhancement in Matlab works in three different components and never works as all components in once. So I separated the components into R component, G component and B component. We can see the result in Figure 33, with their histograms.
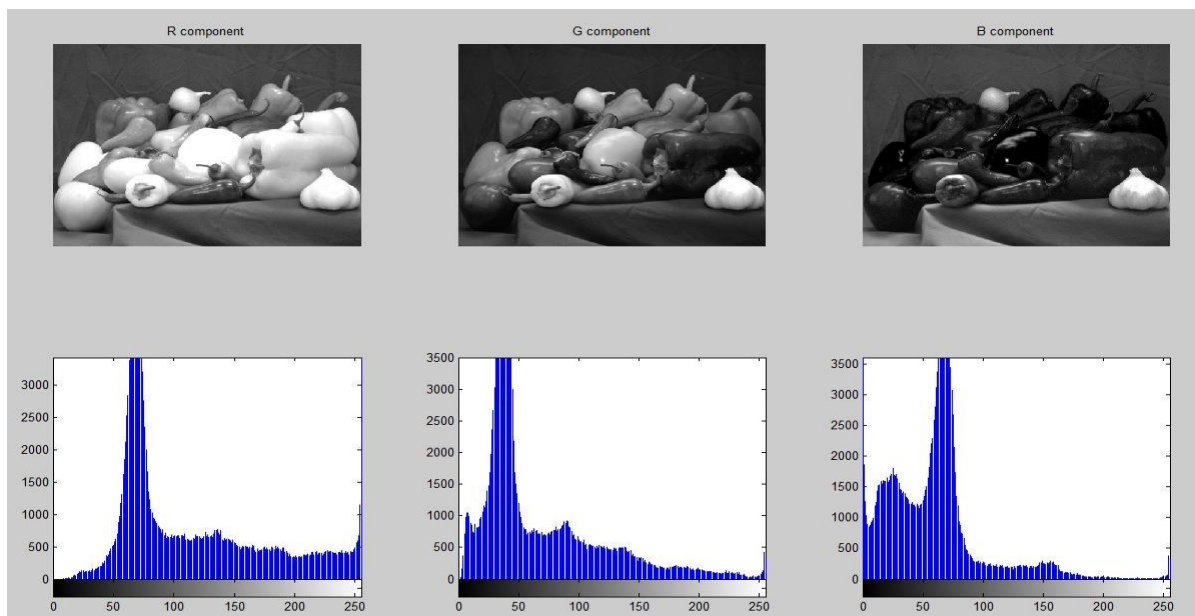

Figure 33: RGB Components and their histograms

Then I convert the RGB values in map of the YCbCr color space, which converts the truecolor of RGB image to its equivalent in YCbCr, YCbCr is the most commonly used color coordinate system for the compression of image and video signals. It is divided into two components: Y components and Cb, Cr components, where Y is the luminance and Cb, Cr are the chrominance. Saturation is one of the properties for color image, in the transformation of the input images I set saturation by 1.3, then I implement three enhancement techniques to the saturated images, imadjust, averaging filter and unsharp filter as shown in the Figure34, and we can see the output image, using unsharp masking filter effect on the saturated image, and comparing it with other techniques, it's the best result for the human eye. **[2]**

Figure 34: Enhancement techniques implemented to the saturated image.

The Codes of Matlab for this implementation are illustrated in Appendix A.

6 Conclusion

My project is about image enhancement methods and implementation in Matlab, in the theoretical part, I studied two parts of the image enhancement, the spatial domain and the frequency domain. And I focused on edge and color enhancement, especially on the sharpening techniques and I used its different types. And I studied different examples of filters based on sharpening technique, and compared them, and I got results of how the effect to enhance edge and color in the digital image. In the practical part I have created simple graphical user interface in Matlab for *histogram* and *imadjust* enhancement techniques, and I implemented another enhancement implementations for *imadjust*, histogram and *fspecial* filters in Matlab.

During my work I have learn and gain a lot of information about image enhancement. However, like each student I faced some troubles, especially in programming, so I was learning how to program in Matlab, and this took time, but it's still a simple work and simple knowledge I gained about Matlab.

I believe I filled the task of the project, and I would like to focus and work more on this issue in the future, I hope to improve more.

According to the results in Matlab, working on *GUI*, and designing a simple implementation with some image enhancement functions, I believe it can be useful, for teaching and presentation about image enhancement techniques; by using my *GUI* with different functions, so the user will notice the different enhancement effects on the output image.

7 <u>LITERATURE</u>

[1] Bovik, Alan. 2005. *Handbook of Image and Video Processing*. Elsevier Academic Press, New York. 2005, ISBN: 0-12-119792-1.

[2] The Math works 2007. *Image Processing Toolbox 7*. Massachusetts, USA. Accessible from <www.mathworks.com>.

[3] Gonzalez, Rafael C.; Woods, Richard E. 2002. *Digital Image Processing*. Second Edition. Prentice Hall, New Jeresy. 2002, ISBN: 0-130-94650-8.

[4] Bernd Jahne. 2002. *Digital image processing*. 5th Edition, Springer, Berlin. 2002. ISBN: 3-540-67754-2.

[5] William K. Pratt. 2007. *Digital Image Processing*, PIKS Scientific Inside. 4th edition, Wiley-Interscience, 2007. ISBN: 978-0-471-76777-0.

[6] Prof. Sumana Gupta. *Digital image processing*, Indian Institute of technology. [online] Published 2008. [citied 1.5.2012]. Accessible from <http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT-KANPUR/Digi_Img_Pro/ui/About-Faculty.html>.

[7] Prof. P.K.Bisawas. *Images Enhancement*. [online] Published 2008 [citied 22.5.2012]. Accessible from <http://nptel.iitm.ac.in/courses.php?branch=Ece

[8] John C. Russ. 2006. *The Image Processing Handbook*, 5th Edition, CRC Press. 2007. ISBN: 978-084372544.

[9] Justyna Inglot. 2012. *Advanced Image Processing with Matlab*, Mikkeli University of Applied Sciences. [online] Published 12.May.2012.[citied 14.June.2012].Accessible from <http://publications.theseus.fi/bitstream/handle/10024/43274/FinalThesis.pdf?sequence=1>.

[10] Black Ice Software. 2012. *Color space conversion Overview*, [online] Published 2012. [cited 15.08.2012]. Accessible from <http://www.blackice.com/ImageColorConversion.htm>.

[11]University of Westren Australia. 1999. *Colour image Processing*, Department of Computer Science [online] Published 14.May. 1999. [cited 15.08.2012] Accessible from <http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT14/lecture12.html#gonzwoods>.

[13] IEEE UAEU Student Branch. 2001. *Introduction to Graphical User Interface (GUI) Matlab 6.5*. The Mathworks. [online] Published 2001 [cited 20.08.2012]. Accessible from <http://ewh.ieee.org/r8/uae/GUI.pdf>.

[14] Black Ice Software. 2012. *HSI Color Space*. [online] Published 2012. [cited 15.08.2012]. Accessible from <http://www.blackice.com/colorspaceHSI.htm>.

[15] Software Intel® IPP. 2012. *Color Models*. Images color Conevrsion, Color Image Processing. [online] Published 10.June. 2012. [cited 20.August.2012]. Accessible from <http://software.intel.com/sites/products/documentation/hpc/ipp/ippi/ippi_ch6/ch6_color_models.html>.

<u>Appendix</u> A: Matlab code for ….

```matlab
im_in = imread('peppers.png');
figure
imshow(im_in);
im_R = im_in(:,:,1);
im_G = im_in(:,:,2);
im_B = im_in(:,:,3);

figure
subplot(2,3,1);
imshow(im_R);title 'R component'
subplot(2,3,2);
imshow(im_G);title 'G component'
subplot(2,3,3);
imshow(im_B);title 'B component'
subplot(2,3,4);
imhist(im_R);
subplot(2,3,5);
imhist(im_G);
subplot(2,3,6);
imhist(im_B);

im_R_out = im_R;
im_G_out = uint8(0.8*double(im_G));
im_B_out = uint8(1*im_B);
 im_out(:,:,1) = im_R_out;
im_out(:,:,2) = im_G_out;
im_out(:,:,3) = im_B_out;

figure
imshow(im_out)
im_ycbcr = rgb2ycbcr(im_in);
im_y = im_ycbcr(:,:,1);
im_cb = double(im_ycbcr(:,:,2)) - 128;
im_cr = double(im_ycbcr(:,:,3)) - 128;

 k_satur = 1.3;
im_cb_out = uint8(k_satur * im_cb + 128);
im_cr_out = uint8(k_satur * im_cr + 128);
im_ycbcr_out(:,:,1) = im_y;
im_ycbcr_out(:,:,2) = im_cb_out;
im_ycbcr_out(:,:,3) = im_cr_out;
im_rgb_out = ycbcr2rgb(im_ycbcr_out);

h = fspecial('average',[3 3]);
averagingfilter = imfilter (im_rgb_out,h)
```

56

```matlab
RGB2 = imadjust(im_rgb_out,[.2 .3 0; .6 .7 1],[]);
H = fspecial('unsharp');
sharpened = imfilter(im_rgb_out,H)
figure
subplot(2,2,1), imshow(im_rgb_out); title ('Saturated image')
subplot(2,2,2), imshow(averagingfilter); title ('Saturated image after
averaging filter')
subplot(2,2,3), imshow(RGB2); title ('Saturated image after imadjust')
subplot(2,2,4), imshow(sharpened); title ('Saturated image after unsharp
filter')
```