CHAPTER 1
IMAGE PROCESSING

1.1 Introduction

Digital image processing is an area characterized by the need for extensive experimental work to establish the viability of proposed solutions to a given problem. An important characteristic underlying the design of image processing systems is the significant level of testing & experimentation that normally is required before arriving at an acceptable solution. This characteristic implies that the ability to formulate approaches &quickly prototype candidate solutions generally plays a major role in reducing the cost & time required to arrive at a viable system implementation.

1.2 Image processing

An image may be defined as a two-dimensional function f(x, y), where x & y are spatial coordinates, & the amplitude of 'f ' at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. When x, y & the amplitude values of 'f' are all finite discrete quantities, we call the image a digital image. The field of DIP refers to processing digital image by means of digital computer. Digital image is composed of a finite number of elements, each of which has a particular location & value. The elements are called pixels.

Vision is the most advanced of our sensor, so it is not surprising that image play the single most important role in human perception. However, unlike humans, who are limited to the visual band of the EM spectrum imaging machines cover almost the entire EM spectrum, ranging from gamma to radio waves. They can operate also on images generated by sources that humans are not accustomed to associating with image.

There is no general agreement among authors regarding where image processing stops & other related areas such as image analysis& computer vision start. Sometimes a distinction is made by defining image processing as a discipline in which both the input &

output at a process are images. This is limiting & somewhat artificial boundary. The area of image analysis (image understanding) is in between image processing & computer vision.

There are no clear-cut boundaries in the continuum from image processing at one end to complete vision at the other. However, one useful paradigm is to consider three types of computerized processes in this continuum: low-, mid-, & high-level processes. Low-level process involves primitive operations such as image processing to reduce noise, contrast enhancement & image sharpening. A low- level process is characterized by the fact that both its inputs & outputs are images. Mid-level process on images involves tasks such as segmentation, description of that object to reduce them to a form suitable for computer processing & classification 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 & at the far end of the continuum performing the cognitive functions normally associated with human vision. Digital image processing, as already defined is used successfully in a broad range of areas of exceptional social & economic value.

1.3 Image

An image is represented as a two dimensional function f(x, y) where x and y are spatial co-ordinates and the amplitude of 'f' at any pair of coordinates (x, y) is called the intensity of the image at that point. Images may be two-dimensional, such as a photograph, screen display, and as well as a three dimensional, such as a statue or hologram. They may be captured by optical devices such as cameras, mirrors, lenses, telescopes, microscopes, etc. and natural objects and phenomena, such as the human eye or water.

The word image is also used in the broader sense of any two-dimensional figure such as a map, a graph, a pie chart, or a painting. In this wider sense, images can also be rendered manually, such as by drawing, the art of painting, carving, rendered automatical by printing or computer graphics technology, or developed by a combination of methods, especially in a pseudo-photograph.

A volatile image is one that exists only for a short period of time. This may be a reflection of an object by a mirror, a projection of a camera , or a scene displayed on a cathode ray tube. A fixed image, also called a hard copy, is one that has been recorded on a material object, such as paper or textile by photography or any other digital process.

A mental image exists in an individual's mind, as something one remembers or imagines. The subject of an image need not be real; it may be an abstract concept, such as a graph, function, or "imaginary" entity. For example, Sigmund Freud claimed to have dreamed purely in aural-images of dialogs. The development of synthetic acoustic technologies and the creation of sound art have led to a consideration of the possibilities of a sound-image made up of irreducible phonic substance beyond linguistic or musicological analysis.

A still image is a single static image, as distinguished from a kinetic image (see below). This phrase is used in photography, visual media and the computer industry to emphasize that one is not talking about movies, or in very precise or pedantic technical writing such as a standard. A film still is a photograph taken on the set of a movie or television program during production, used for promotional purposes.

1.3.1 Gray scale image

In photography and computing, a grayscale or grayscale digital image is An image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort, also known as black-and-white, are composed exclusively of shades of grey, varying from black at the weakest intensity to white at the strongest. Grayscale images are distinct from one-bit bi-tonal black-and-white images, which in the context of computer imaging are images with only the two colors, black, and white (also called bilevel or binary images). Grayscale images have many shades of gray in between.

Figure 1.1 gray scale image describes the digital black and white image which tells about intensity information. It offers shades of the pixels varying from black to white.

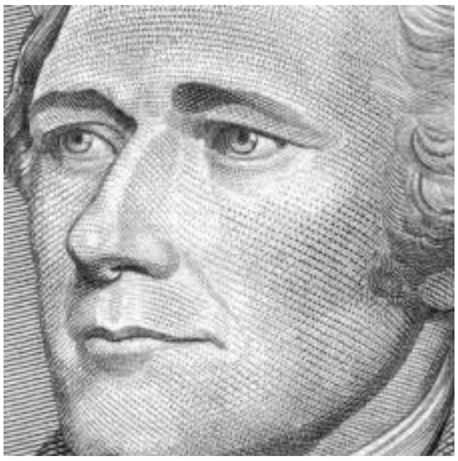


Fig1.1 gray scale image

Grayscale images are often the result of measuring the intensity of light at each pixel in a single band of the electromagnetic spectrum (e.g. infrared, visible light, ultraviolet, etc.), and in such cases they are monochromatic proper when only a given frequency is captured. But also they can be synthesized from a full color image; see the section about converting to grayscale.

1.3.2 Color image

It can be represented by three functions, R (xylem)for red, G (xylem)for green and B (xylem)for blue. An image may be continuous with respect to the x and y coordinates and also in amplitude. Converting such an image to digital form requires that the coordinates as well as the amplitude to be digitized. Digitizing the coordinate's values is called sampling. Digitizing the amplitude values is called quantization.

The intensity of a pixel is expressed within a given range between a minimum and a maximum, inclusive. This range is represented in an abstract way as a range from 0 (total absence, black) and 1 (total presence, white), with any fractional values in between. This notation is used in academic papers, but this does not define what "black" or "white" is in terms of colorimetric.

Figure 1.2 describes the color image with each pixel is termed as some color. This is to the graphics oriented platform which can recognize and modulate easily.

Fig 1.2: color image

Another convention is to employ percentages, so the scale is then from 0% to 100%. This is used for a more intuitive approach, but if only integer values are used, the range encompasses a total of only 101 intensities, which are insufficient to represent a broad gradient of grays. Also, the percentile notation is used in printing to denote how much ink is employed in half toning, but then the scale is reversed, being 0% the paper white (no ink) and 100% a solid black (full ink).

In computing, although the grayscale can be computed through rational numbers, image pixels are stored in binary, quantized form. Some early grayscale monitors can only show up to sixteen (4-bit) different shades, but today grayscale images (as photographs) intended for visual display (both on screen and printed) are commonly stored with 8 bits per sampled pixel, which allows 256 different intensities (i.e., shades of gray) to be recorded, typically on a non-linear scale. The precision provided by this format is barely sufficient to avoid visible banding artifacts, but very convenient for programming due to the fact that a single pixel then occupies a single byte.

1.4 Coordinate convention

The result of sampling and quantization is a matrix of real numbers. We use two principal ways to represent digital images. Assume that an image f(x, y) is sampled so that the resulting image has M rows and N columns. We say that the image is of size M X N. The values of the coordinates (xylem) are discrete quantities. For notational clarity and convenience, we use integer values for these discrete coordinates. In many image processing books, the image origin is defined to be at (xylem)=(0,0).The next coordinate values along the first row of the image are (xylem)=(0,1).It is important to keep in mind that the notation (0,1) is used to signify the second sample along the first row. It does not mean that these are the actual values of physical coordinates when the image was sampled. Following figure shows the coordinate convention. Note that x ranges from 0 to M-1 and y from 0 to N-1 in integer increments.

The coordinate convention used in the toolbox to denote arrays is different from the preceding paragraph in two minor ways. First, instead of using (xylem) the toolbox uses the notation (race) to indicate rows and columns. Note, however, that the order of coordinates is the same as the order discussed in the previous paragraph, in the sense that the first element of a coordinate topples, (alb), refers to a row and the second to a column. The other difference is that the origin of the coordinate system is at (r, c) = (1, 1); thus, r ranges from 1 to M and c from 1 to N in integer increments. IPT documentation refers to the coordinates. Less frequently the toolbox also employs another coordinate convention called spatial coordinates which uses x to refer to columns and y to refers to rows. This is the opposite of our use of variables x and y.

1.5 Image as Matrices

The preceding discussion leads to the following representation for a digitized image function:

$$f(xylem)= \begin{pmatrix} f(0,0) & f(0,1) & \dots\dots\dots.. & f(0,N-1) \\ f(1,0) & f(1,1) & \dots\dots\dots\dots & f(1,N-1) \\ . & . & & . \\ . & . & & . \\ f(M-1,0) & f(M-1,1) & \dots\dots\dots\dots & f(M-1,N-1) \end{pmatrix} \qquad \_\_\_\_\_\_\_\_\_\_(1.1)$$

Equation “1.1, 1.2” denotes the complete image as matrix form. Here f(M,N) are called the pixel value corresponds to 0^{th} row and 0^{th} column of the image.

The right side of this equation is a digital image by definition. Each element of this array is called an image element, picture element, pixel or pel. The terms image and pixel are used throughout the rest of our discussions to denote a digital image and its elements.

A digital image can be represented naturally as a MATLAB matrix:

$$f = \begin{pmatrix} f(1,1) & f(1,2) \ldots\ldots & f(1,N) \\ f(2,1) & f(2,2) \ldots\ldots & f(2,N) \\ f(M,1) & f(M,2) \ldots\ldots & f(M,N) \end{pmatrix} \_\_\_\_\_\_\_\_\_ (1.2)$$

Where f(1,1) = f(0,0) (note the use of a monoscope font to denote MATLAB quantities). Clearly the two representations are identical, except for the shift in origin. The notation f(p ,q) denotes the element located in row p and the column q. For example f(6,2) is the element in the sixth row and second column of the matrix f. Typically we use the letters M and N respectively to denote the number of rows and columns in a matrix. A 1xN matrix is called a row vector whereas an Mx1 matrix is called a column vector. A 1x1 matrix is a scalar.

Matrices in MATLAB are stored in variables with names such as A, a, RGB, real array and so on. Variables must begin with a letter and contain only letters, numerals and underscores. As noted in the previous paragraph, all MATLAB quantities are written using mono-scope characters. We use conventional Roman, italic notation such as f(x ,y), for mathematical expressions

1.6 Image types

The toolbox supports four types of images:

- Intensity images
- Binary images
- Indexed images
- Red green Blue images

Most monochrome image processing operations are carried out using binary or intensity images, so our initial focus is on these two image types. Indexed and RGB colour images.

1.6.1 Intensity Images

An intensity image is a data matrix whose values have been scaled to represent intentions. When the elements of an intensity image are of class unit8, or class unit 16, they have integer values in the range [0,255] and [0, 65535], respectively. If the image is of class double, the values are floating _point numbers. Values of scaled, double intensity images are in the range [0, 1] by convention.

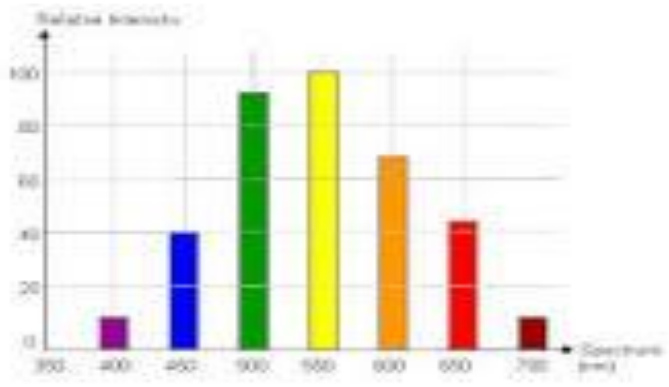


Fig.1.3 Intensity image

Figure 1.3 describes the intensity image where the information is in the amplitudes of the image. The digitizing of amplitudes is called quantization.

1.6.2 Binary Images

Binary images have a very specific meaning in MATLAB.A binary image is a logical array 0s and1s.Thus, an array of 0s and 1s whose values are of data class, say unit8, is not considered as a binary image in MATLAB .A numeric array is converted to binary using function logical. Thus, if A is a numeric array consisting of 0s and 1s, we create an array B using the statement.

B=logical (A)

If A contains elements other than 0s and 1s.Use of the logical function converts all nonzero quantities to logical 1s and all entries with value 0 to logical 0s.Using relational and logical operators also creates logical arrays.To test if an array is logical we use the Is logical function

D = islogical(c)

If c is a logical array, this function returns a 1.Otherwise returns a 0. Logical array can be converted to numeric arrays using the data class conversion functions.

Figure 1.4 binary image describes the image is turned in to 0's and 1's where it can easily understand by the computer.

Fig.1.4 Binary image

1.6.3 Indexed Images

An indexed image has two components:

- A data matrix integer, x.
- A color map matrix, map.

Matrix map is an m*3 arrays of class double containing floating_ point values in the range [0, 1].The length m of the map are equal to the number of colors it defines. Each row of map specifies the red, green and blue components of a single color. An indexed images uses “direct mapping” of pixel intensity values color map values. The color of each pixel is determined by using the corresponding value the integer matrix x as a pointer in to map. If x is of class double ,then all of its components with values less than or equal to 1 point to the first row in map, all components with value 2 point to the second row and so on. If x is of class units or unit 16, then all components value 0 point to the first row in map, all components with value 1 point to the second and so on.

Figure 1.5 indexed image describes that each and every value of image pixel is represented by some colors which may use as a language for the readers view. The below figure shows the pixel values corresponding colors assigned to it.

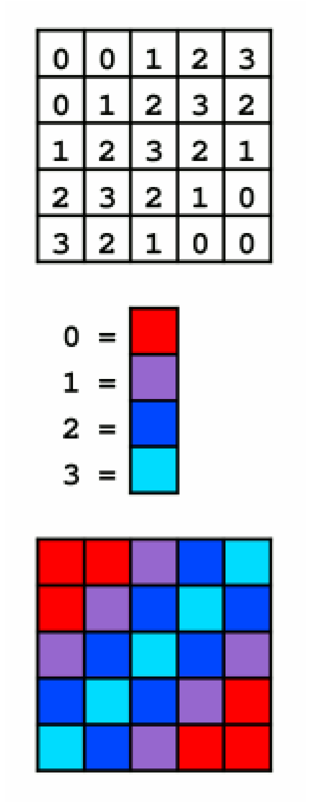


Fig.1.5 Indexed image

1.6.4 Red Green Blue Image

An RGB color image is an M*N*3 array of color pixels where each color pixel is triplet corresponding to the red, green and blue components of an RGB image, at a specific spatial location. An RGB image may be viewed as “stack” of three gray scale images that when fed in to the red, green and blue inputs of a color monitor

Produce a color image on the screen. Convention the three images forming an RGB color image are referred to as the red, green and blue components images. The data class of the components images determines their range of values. If an RGB image is of class double the range of values is [0, 1].

Similarly the range of values is [0,255] or [0, 65535].For RGB images of class units or unit 16 respectively. The number of bits use to represents the pixel values of the component images determines the bit depth of an RGB image. For example, if each component image is an 8bit image, the corresponding RGB image is said to be 24 bits deep.

Generally, the number of bits in all component images is the same. In this case the number of possible color in an RGB image is (2^b) ^3, where b is a number of bits in each component image. For the 8bit case the number is 16,777,216 colors.

Figure 1.6 RGB image describes the color image where the colors red, green and blue are primary colors and cyan, yellow and magenta are the secondary colors. The RGB image is the most suitable image to read an image by the human eye.

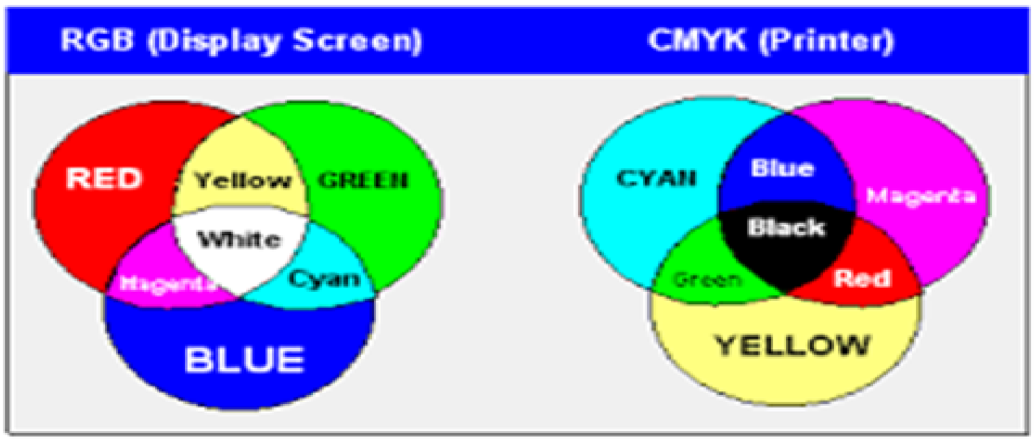


Fig.1.6 Red Green Blue image

1.7 Existing system

One approach to provide privacy is to obfuscate sensitive regions within an image/video which prevents the identification of the persons being captured. The authors in have proposed an irreversible obfuscation method which however, prevents the use of the captured videos from aiding criminal investigation or to be used as evidence in court. De-identification is the process used to prevent a person's identity from being connected with information. Common uses of de-identification include human subject research for the sake of privacy for research participants. Common strategies for de-identifying datasets are deleting or masking personal identifiers, such as name and social security number, and suppressing or generalizing quasi-identifiers.

1.8 Proposed system

This work presents a novel Reversible De-Identification method that can be used in conjunction with any obfuscation process. The residual information needed to reverse the obfuscation process is compressed, authenticated, encrypted and embedded within the obfuscated image using a two-level Reversible Watermarking scheme. The proposed method ensures an overall single-pass embedding capacity of 1.25 bpp(bits per pixel), where 99.8% of the images considered required less than 0.8 bpp(bits per pixel) while none of them required more than 1.1 bpp(bits per pixel). Experimental results further demonstrate that the proposed method managed to recover and authenticate all images considered.

1.9 Problems in security

Network security deals with the requirements needed for a company, organization or a network administrator to help in protecting the network, computer systems and the resources that are network accessible. They are protected from any unauthorized entry, malicious components as well as monitoring continuously, consistently and measuring the effectiveness or lack of effectiveness of the network.

Network security is a major concern of every company that has a computer and is connected to a network. A network security that has been compromised means that a competitor or any hacker can gain entry to the sensitive or critical data and they may delete or make off with the information resulting in data loss or complete system destruction. The terms information security and network security are most of the time used to represent the same meaning. Network security, though, is more specifically taken as the provision protection from outside intruders.

The process of network security begins from the authentication of any user who logs in with the appropriate password and user name which is ‘one factor authentication’. There is another method of authentication known as ‘two factor’ where when one is using an item like an ATM card or mobile phone and another three factor authentication can also be used where a body part is used like a retinal scan or fingerprint. When authentication has been verified, there is a firewall that decides, which programs or services are allowed for network users to access. This component may be effective in the ability to prevent any unauthorized access but it fails to check harmful contents like computer worms that are transmitted across the network. An IPS or intrusion prevention system is able to detect and stop the activities of this sort of malware. The firewall and IPS settings are created by the network’s System Administrator who also installs a viable antivirus system, which is up-to-date.

1.10 Transmission time

In telecommunication networks, the transmission time, is the amount of time from the beginning until the end of a message transmission. In the case of a digital message, it is the time from the first bit until the last bit of a message has left the transmitting node. The packet transmission time in seconds can be obtained from the packet size in bit and the bit rate in bit/s. propagation delay is also the important factor in transmitting the data.

CHAPTER 2
WATERMARKING

2.1 Introduction

In recent years, the world has been witnessing a rapid growth of Internet technologies, and proliferation in multimedia distribution and e-commerce. Many techniques for copying digital information have been created, which enable illegal copies to be reused, manipulated or distributed. Therefore, the need for content and copyright protection techniques has emerged.

Cryptography was suggested as an effective tool to prevent illegal copies distribution. However, in order to avoid digital data access, a special hardware should be merged with the cryptography tool, which made it costly. Several watermarking technologies were introduced since they are more useful and not costly.

A digital watermark is a distinguishing piece of information that is stuck to the data intended to be protected. Watermarking is a method for encoding visible or invisible watermarks into multimedia data applications. Visible watermarks are clearly detectable. They are intentionally perceptible to a human observer. Visible watermarking is used to prevent unauthorized access to an image.

On the other hand, invisible watermarking is used to identify the owner or the origin of the host image. It is also used to detect any unauthorized image copies to identify a customer or to prove ownership. There are numerous techniques for digital watermarking. The first publications that related to watermarking were in 1979.

In order to have efficient watermarking, one should consider three main criteria: robustness against standard manipulations and attacks such as noise and compression, security (resistance against forgery of the watermark), and imperceptibility which shows how much the watermarked image gets distorted.

2.2 Watermarking techniques

Watermarking techniques can be classified in several ways. According to the need of the original image for watermark extraction or detection, watermarking is classified to non-blind and blind.

Non-blind watermarking techniques, such as, require that the original image exists when detecting the watermark, whereas, blind techniques do not. Another way to classify watermarking is by how the watermark is inserted; in the spatial domain, or in the transform domain.

Figure 2.1 shows the watermarked image with original image. It is the visible watermarked image where we can see the text highlighted in the figure.

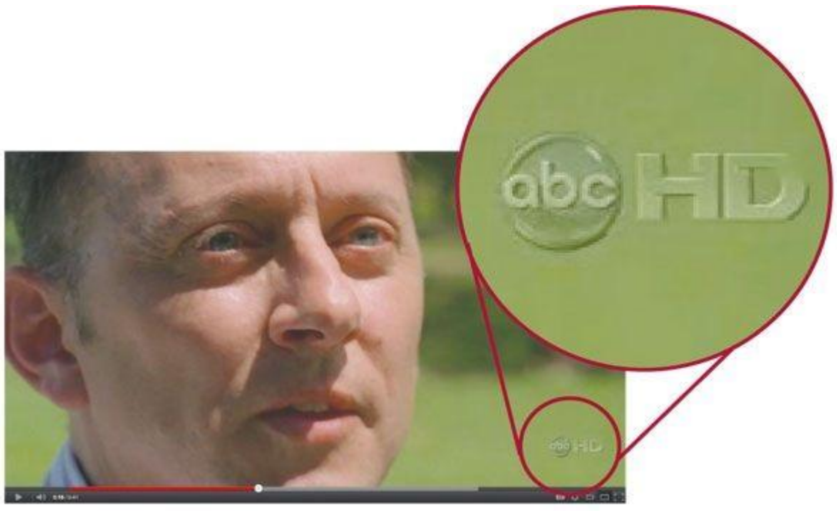


Fig.2.1 Watermarked image

Early watermarking schemes that were introduced, worked in the spatial domain, where the watermark is added by modifying pixel values of the host image. Some of the spatial domain watermarking approaches were based on the modification of the least significant bit (LSB) of an image based on the assumption that the LSB data are insignificant.

Generally, spatial domain watermarking is easy to implement from a computational point of view, but too fragile to resist numerous attacks. In order to have more promising techniques, researches were directed towards watermarking in the transform domain, where the watermark is not added to the image intensities, but to the values of its transform coefficients. Then to get the watermarked image, one should perform the transform inversely.

Some of the transform based watermarking techniques used the Discrete Cosine Transform (DCT). The wavelet transform is another type of the transform domain. Wavelet based transform gained popularity recently since the property of multi resolution analysis that it provides. Wavelets can be orthogonal (orthonormal) or biorthogonal. Most of the wavelets used in watermarking were orthogonal wavelets.

A few techniques used bi-orthogonal wavelets. The biorthogonal wavelet transform is an invertible transform. It has some favorable properties over the orthogonal wavelet transform, mainly, the property of perfect reconstruction and smoothness (vanishing points). In this project, we will use the biorthogonal wavelets in the watermarking method we propose.

In 1998, Kundur and Hatzinakos suggested a watermarking model using biorthogonal wavelets based on embedding a watermark in detail wavelet coefficients of the host image. The results showed that the model was robust against numerous signal distortions, but required the presence of the watermark at the detection and extraction phases, i.e. the method was nonblind.

Results reported that the technique is robust even against few geometric attacks such as cropping and rescaling. However, both techniques used a spread spectrum (SS) method for watermarking. Such methods serve limited applications of watermarking since the watermark used is a pseudo-random noise (PN) signature; therefore, it does not represent any meaningful information about the owner of the image.

The biorthogonal methods used even-length wavelets and the latter used odd-length wavelets. Those methods were robust against several attacks, but were presented for the sake of detecting the presence of a watermark not for extracting it.

2.3 Steganography

The advantage of steganography over cryptography alone is that the intended secret message does not attract attention to itself as an object of scrutiny. Plainly visible encrypted messages no matter how unbreakable will arouse interest, and may in themselves be incriminating in countries where encryption is illegal. Thus, whereas cryptography is the practice of protecting the contents of a message alone, steganography is concerned with concealing the fact that a secret message is being sent, as well as concealing the contents of the message.

Steganography includes the concealment of information within computer files. In digital steganography, electronic communications may include stegnographic coding inside of a transport layer, such as a document file, image file, program or protocol. Media files are ideal for steganographic transmission because of their large size. For example, a sender might start with an innocuous image file and adjust the color of every 100th pixel to correspond to a letter in the alphabet, a change so subtle that someone not specifically looking for it is unlikely to notice it.

2.4 Cryptography

Cryptography prior to the modern age was effectively synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The originator of an encrypted message shared the decoding technique needed to recover the original information only with intended recipients, thereby precluding unwanted persons to do the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.

Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary. It is theoretically possible to break such a system, but it is infeasible to do so by any known practical means. These schemes are therefore termed computationally secure; theoretical advances, e.g., improvements in integer factorizationalgorithms, and faster computing technology require these solutions to be continually adapted. There exist information-theoretically secure schemes that provably cannot be broken even with unlimited computing power an example is the one-time pad but these schemes are more difficult to implement than the best theoretically breakable but computationally secure mechanisms.

Cryptology-related technology has raised a number of legal issues. In the United Kingdom, additions to the Regulation of Investigatory Powers Act 2000 require a suspected criminal to hand over his or her decryption key if asked by law enforcement. Otherwise the user will face a criminal charge.

2.5 Integer Wavelet Transform based watermarking

Generally wavelet domain allows us to hide data in regions that the human visual system (HVS) is less sensitive to, such as the high resolution detail bands (HL, LH and HH), Hiding data in these regions allow us to increase the robustness while maintaining good visual quality. Integer wavelet transform maps an integer data set into another integer data set. In discrete wavelet transform, the used wavelet filters have floating point coefficients so that when we hide data in their coefficients any truncations of the floating point values of the pixels that should be integers may cause the loss of the hidden information which may lead to the failure of the data hiding system .

To avoid problems of floating point precision of the wavelet filters when the input data is integer as in digital images, the output data will no longer be integer which doesn't allow perfect reconstruction of the input image and in this case there will be no loss of information through forward and inverse transform. Due to the mentioned difference between integer wavelet transform (IWT) and discrete wavelet transform (DWT) the LL sub band in the case of IWT appears to be a close copy with smaller scale of the original image while in the case of DWT the resulting LL sub band is distorted. Lifting schemes is one of many techniques that can be used to perform integer wavelet transform it is also the scheme used in this paper. Figure 2.2 describes the embedding process of an image in to another image by several steps. The original image is taken as reference and another image is viewed in 2D view and then transforming in to frequency domain and a key is to set for the security of the image.

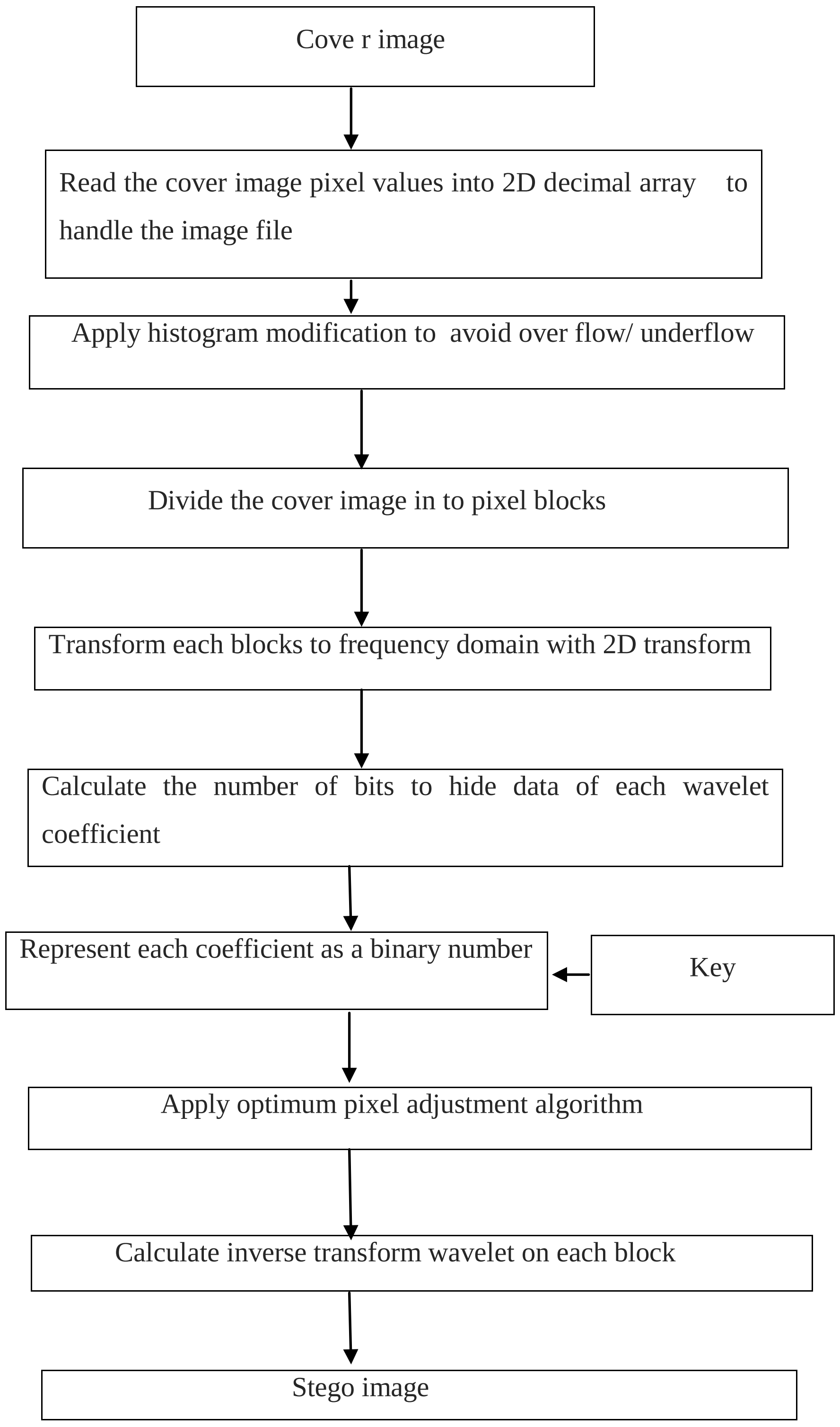

Cove r image
Read the cover image pixel values into 2D decimal array to handle the image file
Apply histogram modification to avoid over flow/ underflow
Divide the cover image in to pixel blocks
Transform each blocks to frequency domain with 2D transform
Calculate the number of bits to hide data of each wavelet coefficient
Represent each coefficient as a binary number
Key
Apply optimum pixel adjustment algorithm
Calculate inverse transform wavelet on each block
Stego image

Fig 2.2 Embedding process

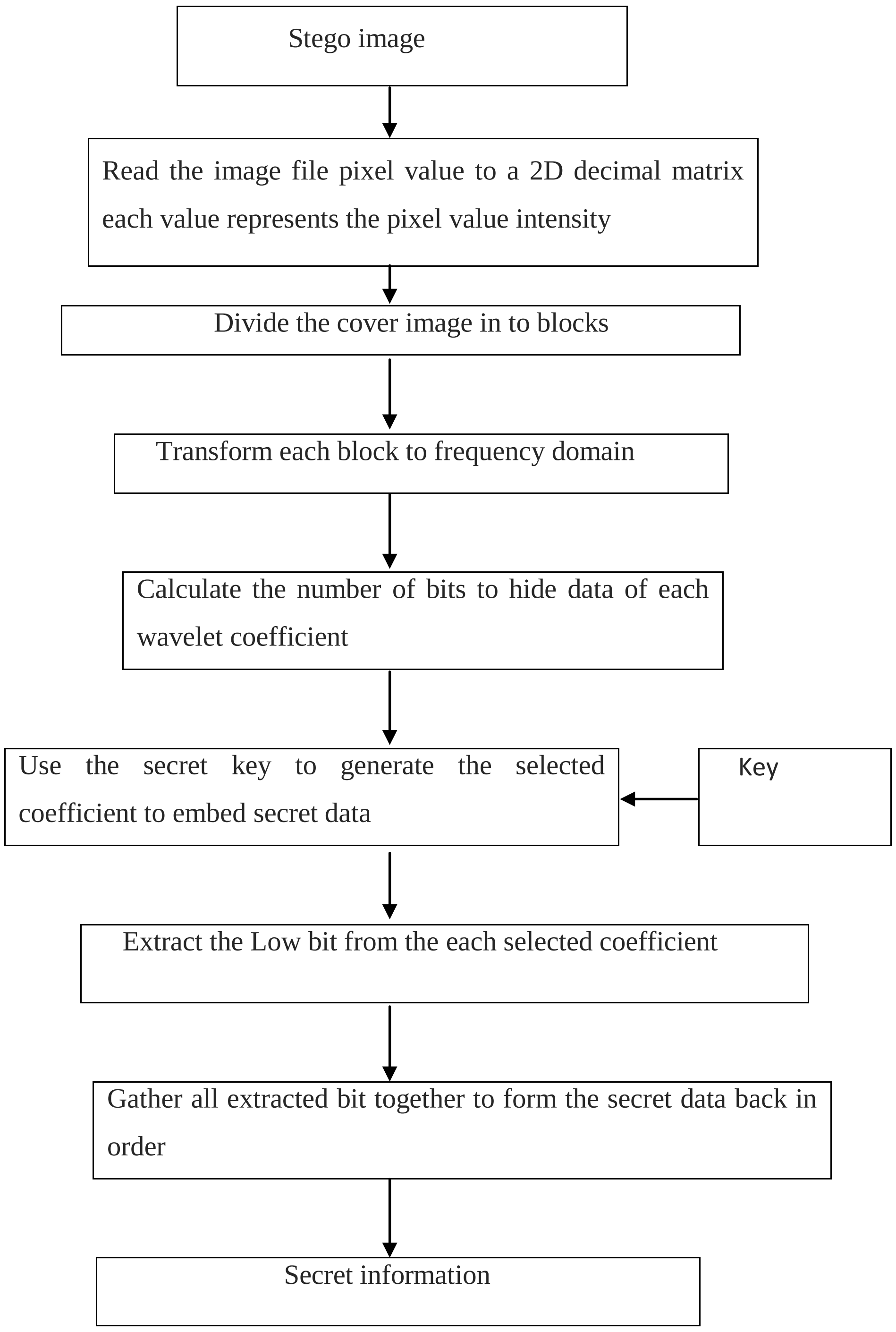

Stego image
Read the image file pixel value to a 2D decimal matrix each value represents the pixel value intensity
Divide the cover image in to blocks
Transform each block to frequency domain
Calculate the number of bits to hide data of each wavelet coefficient
Use the secret key to generate the selected coefficient to embed secret data
Key
Extract the Low bit from the each selected coefficient
Gather all extracted bit together to form the secret data back in order
Secret information

Fig 2.3: Extraction process

Figure 2.3 explains the extraction process of the embedded image with the particular key as shown in the above process.

CHAPTER 3

EMBEDDING AND EXTRACTING

3.1 Introduction

One of the imports aspects of image storage is its efficient Compression. To make this fact clear let's see an example.

An image, 1024 pixel x 1024 pixel x 24 bit without compression would require 3 MB of storage and 7 minutes for transmission, utilizing a high speed 64 Kbits/s ISDN line. If the image is compressed at a 10:1 compression ratio, the storage requirement is reduced to 300 KB and the transmission time drops to under 6 seconds. Seven 1 MB images can be compressed and transferred to a floppy disk in less time than it takes to send one of the original files, uncompressed, over an AppleTalk network.

In a distributed environment large image files remain a major bottleneck within systems. Compression is an important component of the solutions available for creating file sizes of manageable and transmittable dimensions. Increasing the bandwidth is another method, but the cost sometimes makes this a less attractive solution.

Platform portability and performance are important in the selection of the compression/decompression technique to be employed. Compression solutions today are more portable due to the change from proprietary high end solutions to accepted and implemented international standards. JPEG is evolving as the industry standard technique for the compression of continuous tone images.

3.2 Reading Images

Images are read into the MATLAB environment using function imread whose syntax is

```
Imread('filename')
```

Table 3.1 shows the overview of different image formats and their recognized extensions used in different fields.

Format name	Description	recognized extension
JPEG	Joint Photograph Experts Group	.jpg, .jpeg
GIF	Graphics interchange format	.gif
BMP	Windows Bitmap	.bmp
PNG	Portable Network Graphics	.png
XWD	X Window Dump	.xwd
TIFF	Tagged Image File Format	.tif,.ti

Table 3.1 Formats

Here filename is a spring containing the complete of the image file(including anY applicable extension).For example the command line

>> f = imread (‘8. jpg’); reads the JPEG (above table) image chest xray into image array f. Note the use of single quotes (‘) to delimit the string filename. The semicolon at the end of a command line is used by MATLAB for suppressing output. If a semicolon is not included. MATLAB displays the results of the operation(s) specified in that line. The prompt symbol(>>) designates the beginning of a command line, as it appears in the MATLAB command window.

When as in the preceding command line no path is included in filename, imread reads the file from the current directory and if that fails it tries to find the file in the MATLAB search path. The simplest way to read an image from a specified directory is to include a full or relative path to that directory in filename.

For example,

```
>> f = imread ( ‘D:\myimages\chestxray.jpg’);
```

reads the image from a folder called my images on the D: drive, whereas

```
>> f = imread(' . \ myimages\chestxray .jpg');
```

reads the image from the my images subdirectory of the current of the current working directory. The current directory window on the MATLAB desktop tool bar displays MATLAB's current working directory and provides a simple, manual way to change it. Above table lists some of the most of the popular image/graphics formats supported by imread and imwrite.

3.2.1 Joint photographic experts group

JPEG (Joint Photographic Experts Group) is a compression method. JPEG compressed images are usually stored in the JFIF (JPEG File Interchange Format) file format. JPEG compression is lossy compression. Nearly every digital camera can save images in the JPEG/JFIF format, which supports 8 bits per color (red, green, blue) for a 24-bit total, producing relatively small files. Photographic images may be better stored in a lossless non-JPEG format if they will be re-edited, or if small "artifacts" are unacceptable. The JPEG/JFIF format also is used as the image compression algorithm in many Adobe PDF files.

3.2.2 Exchangeable image file format

The EXIF (Exchangeable image file format) format is a file standard similar to the JFIF format with TIFF extensions. It is incorporated in the JPEG writing software used in most cameras. Its purpose is to record and to standardize the exchange of images with image metadata between digital cameras and editing and viewing software. The metadata are recorded for individual images and include such things as camera settings, time and date, shutter speed, exposure, image size, compression, name of camera, color information, etc. When images are viewed or edited by image editing software, all of this image information can be displayed.

3.2.3 Tagged Image File Format

The TIFF (Tagged Image File Format) format is a flexible format that normally saves 8 bits or 16 bits per color (red, green, blue) for 24-bit and 48-bit totals, respectively, usually using either the TIFF or TIF filename extension. TIFFs are lossy and lossless.

Some offer relatively good lossless compression for bi-level (black & white) images. Some digital cameras can save in TIFF format, using the LZW compression algorithm for lossless storage. TIFF image format is not widely supported by web browsers. TIFF remains widely accepted as a photograph file standard in the printing business. TIFF can handle device-specific color spaces, such as the CMYK defined by a particular set of printing press inks.

3.2.4 Portable Network Graphics

The PNG (Portable Network Graphics) file format was created as the free, open-source successor to the GIF. The PNG file format supports true color (16 million colors) while the GIF supports only 256 colors. The PNG file excels when the image has large, uniformly colored areas. The lossless PNG format is best suited for editing pictures, and the lossy formats, like JPG, are best for the final distribution of photographic images, because JPG files are smaller than PNG files. PNG, an extensible file format for the lossless, portable, well-compressed storage of raster images. PNG provides a patent-free replacement for GIF and can also replace many common uses of TIFF. Indexed-color, grayscale, and true color images are supported, plus an optional alpha channel. PNG is designed to work well in online viewing applications, such as the World Wide Web. PNG is robust, providing both full file integrity checking and simple detection of common transmission errors.

3.2.5 Graphics Interchange Format

GIF (Graphics Interchange Format) is limited to an 8-bit palette, or 256 colors. This makes the GIF format suitable for storing graphics with relatively few colors such as simple diagrams, shapes, logos and cartoon style images. The GIF format supports animation and is still widely used to provide image animation effects. It also uses a lossless compression that is more effective when large areas have a single color, and ineffective for detailed images or dithered images.

3.2.6 Windows bitmap

The BMP file format (Windows bitmap) handles graphics files within the Microsoft Windows OS. Typically, BMP files are uncompressed, hence they are large. The advantage is their simplicity and wide acceptance in Windows programs.

3. 3 Data compression algorithms

Two categories of data compression algorithm can be distinguished: lossless and ‘lossy’. Lossy techniques cause image quality degradation in each compression/decompression step. Careful consideration of the human visual perception ensures that the degradation is often unrecognizable, though this depends on the selected compression ratio. In general, lossy techniques provide far greater compression ratios than lossless techniques.

Here we'll discuss the roles of the following data compression techniques:

3.4 Lossless coding techniques

Lossless coding guaranties that the decompressed image is absolutely identical to the image before compression. This is an important requirement for some application domains, e.g. medial imaging, where not only high quality is in demand, but unaltered archiving is a legal requirement. Lossless techniques can also used for the compression of other data types where loss of information is not acceptable, e.g. text documents and program executables.

Some compression methods can be made more effective by adding a 1D or 2D delta coding to the process of compression. These deltas make more effective use of run length encoding, have (statistically) higher maxima in code tables (leading to better results in Huffman and general entropy codings), and build greater equal value areas usable for area coding.

Some of these methods can easily be modified to be lossy. Lossy element fits perfectly into 1D/2D run length search. Also, logarithmic quantization may be inserted to provide better or more effective results.

3.4.1 Run length encoding

Run length encoding is a very simple method for compression of sequential data. It takes advantage of the fact that, in many data streams, consecutive single tokens are often identical. Run length encoding checks the stream for this fact and inserts a special token each time a chain of more than two equal input tokens are found. This special input advises the decoder to insert the following token n times into his output stream. The effectivity of run length encoding is a function of the number of equal tokens in a row in relation to the total number of input tokens. This relation is very high in undeterred two tone images of the type used for facsimile. Obviously, effectively degrades when the input does not contain too many equal tokens. With a rising density of information, the likelihood of two follow tokens being the same does sink significantly, as there is always some noise distortion in the input.

Figure 3.1 run length encoding explains that repeated bits in a message is coded as coefficient and its parameter. Here the coefficient is figured as how many time its parameter is repeated.

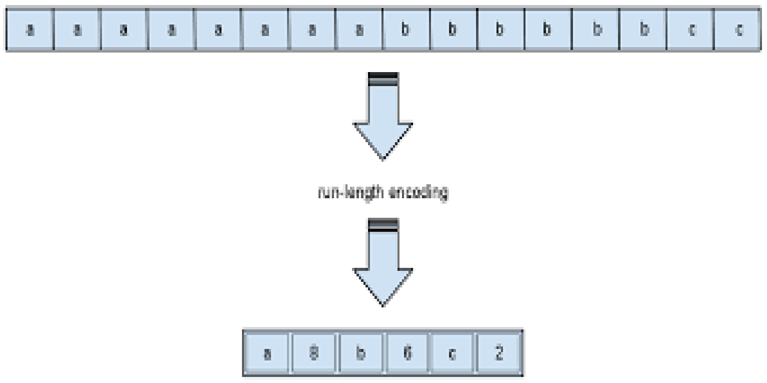


Fig 3.1: Run length encoding

Run length coding is easily implemented, either in software or in hardware. It is fast and very well verifiable, but its compression ability is very limited.

3.4.2 Huffman encoding

This algorithm, developed by D.A. Huffman, is based on the fact that in an input stream certain tokens occur more often than others. Based on this knowledge, the algorithm builds up a weighted binary tree according to their rate of occurrence. Each element of this tree is assigned a new code word, whereat the length of the code word is determined by its position in the tree. Therefore, the token which is most frequent and becomes the root of the tree is assigned the shortest code. Each less common element is assigned a longer code word. The least frequent element is assigned a code word which may have become twice as long as the input token.

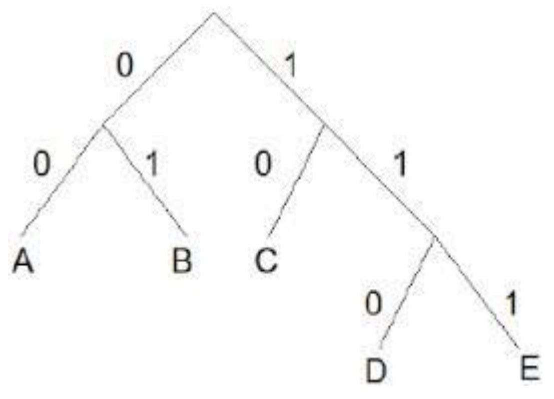


Fig 3.2 : Huffman encoding

In the figure 3.2 huffman coding is done by grouping the message bits with codes like ASCII numbers.

The compression ratio achieved by Huffman encoding uncorrelated data becomes something like 1:2. On slightly correlated data, as on images, the compression rate may become much higher, the absolute maximum being defined by the size of a single input token and the size of the shortest possible output token (max. compression = token size[bits]/2[bits]). While standard palletized images with a limit of 256 colors may be compressed by 1:4 if they use only one color, more typical images give results in the range of 1:1.2 to 1:2.5.

3.4.3 Entropy coding (Lempel/Ziv)

The typical implementation of an entropy coder follows J. Ziv/A. Lempel's approach. Nowadays, there is a wide range of so called modified Lempel/Ziv coding. These algorithms all have a common way of working. The coder and the decoder both build up an equivalent dictionary of met symbols, each of which represents a whole sequence of input tokens.

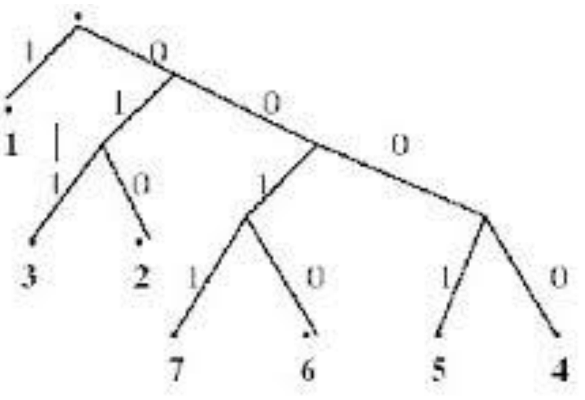


Fig 3.3: Entropy coding

In the figure 3.3 entropy coding is given and is explained averages and probabilities of their occurrences.

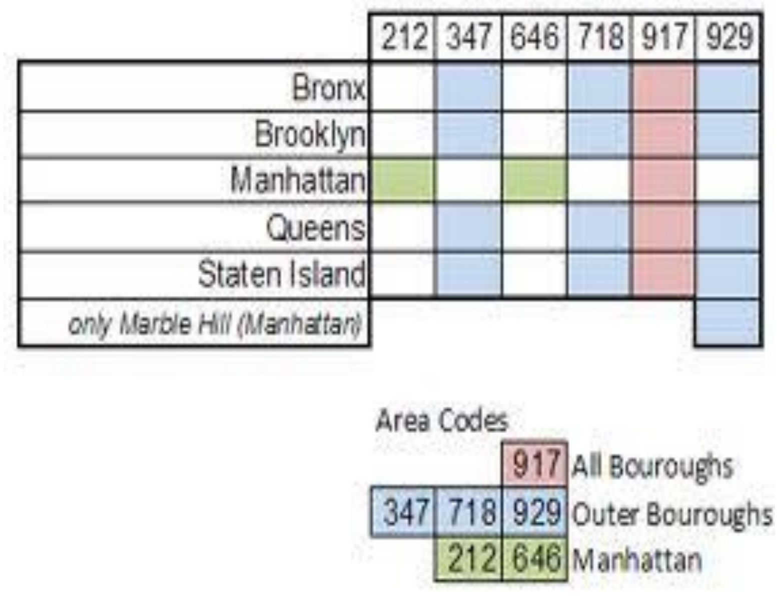
If a sequence is repeated after a symbol was found for it, then only the symbol becomes part of the coded data and the sequence of tokens referenced by the symbol becomes part of the decoded data later. As the dictionary is build up based on the data, it is not necessary to put it into the coded data, as it is with the tables in a Huffman coder.

This method becomes very efficient even on virtually random data. The average compression on text and program data is about 1:2; the ratio on image data comes up to 1:8 on the average GIF image. Here again, a high level of input noise degrades the efficiency significantly. Entropy coders are a little tricky to implement, as there are usually a few tables, all growing while the algorithm runs.

3.4.4 Area coding

Area coding is an enhanced form of run length coding, reflecting the two dimensional character of images. This is a significant advance over the other lossless methods. For coding an image it does not make too much sense to interpret it as a sequential stream, as it is in fact an array of sequences, building up a two dimensional object. Therefore, as the two dimensions are independent and of same importance, it is obvious that a coding scheme aware of this has some advantages. These regions are coded in a descriptive form as an Element with two points and a certain structure. The whole input image has to be described in this form to allow lossless decoding afterwards.

The possible performance of this coding method is limited mostly by the very high complexity of the task of finding largest areas with the same characteristics. Practical implementations use recursive algorithms for reducing the whole area to equal sized sub rectangles until a rectangle does fulfill the criteria defined as having the same characteristic for every pixel.



	212	347	646	718	917	929
Bronx		🟦		🟦	🟥	🟦
Brooklyn		🟦		🟦	🟥	🟦
Manhattan	🟩		🟩		🟥	
Queens		🟦		🟦	🟥	🟦
Staten Island		🟦		🟦	🟥	🟦
only Marble Hill (Manhattan)						🟦

Area Codes

		917	All Bouroughs
347	718	929	Outer Bouroughs
	212	646	Manhattan

Fig 3.4: Area coding

In the figure 3.4 area coding is given with particular area codes to recognize the particular area. It is an advanced version of run length encoding. This type of coding can be highly effective but it bears the problem of a nonlinear method, which cannot be implemented in hardware. Therefore, the performance in terms of compression time is not competitive.

3.5 Lossy coding techniques

In most of applications we have no need in the exact restoration of stored image. This fact can help to make the storage more effective, and this way we get to lossy compression methods. Lossy image coding techniques normally have three components:

Image modeling which defines such things as the transformation to be applied to the image. Parameter quantization whereby the data generated by the transformation is quantized to reduce the amount of information

Encoding, where a code is generated by associating appropriate codeword to the raw data produced by the quantization. Each of these operations is in some part responsible of the compression. Image modeling is aimed at the exploitation of statistical characteristics of the image (i.e. high correlation, redundancy). Typical examples are transform coding methods, in which the data is represented in a different domain (for example, frequency in the case of the Fourier Transform [FT], the Discrete Cosine Transform [DCT], the Kahrunen-Loewe Transform [KLT], and so on), where a reduced number of coefficients contains most of the original information. In many cases this first phase does not result in any loss of information.

The aim of quantization is to reduce the amount of data used to represent the information within the new domain. Quantization is in most cases not a reversible operation: therefore, it belongs to the so called 'lossy' methods.

Encoding is usually error free. It optimizes the representation of the information (helping sometimes to further reduce the bit rate), and may introduce some error detection codes.

In the following sections, a review of the most important coding schemes for lossy compression is provided. Some methods are described in their canonical form (transform coding, region based approximations, fractal coding, wavelets, hybrid methods) and some variations and improvements presented in the scientific literature are reported and discussed.

3.5.1 Transform coding (DCT/Wavelets/Gabor)

A general transform coding scheme involves subdividing an NxN image into smaller nxn blocks and performing a unitary transform on each sub image. A unitary transform is a reversible linear transform whose kernel describes a set of complete, ortho normal discrete basic functions. The goal of the transform is to decorate the original signal, and this declaration generally results in the signal energy being redistributed among only a small set of transform coefficients. In this way, many coefficients may be discarded after quantization and prior to encoding. Also, visually lossless compression can often be achieved by incorporating the HVS contrast sensitivity function in the quantization of the coefficients.

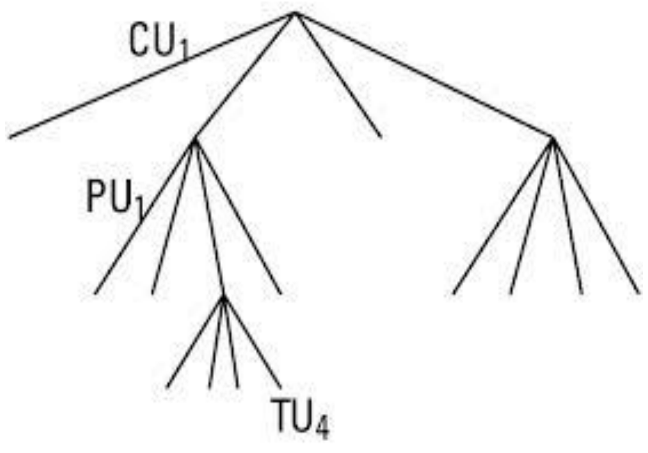


Fig 3.5: Transform coding

In the figure 3.5 transform coding the particular message bits is divided in to sub groups and some raw bits is added and transformations are done for effective security.

Transform coding can be generalized into four stages:

1. Image subdivision

2 Image transformations

3. Coefficient quantization

4. Huffman encoding.

For a transform coding scheme, logical modeling is done in two steps: a segmentation one, in which the image is subdivided in bi dimensional vectors (possibly of different sizes) and a transformation step, in which the chosen transform (e.g. KLT, DCT, and Hadamard) is applied.

Quantization can be performed in several ways. Most classical approaches use 'zonal coding', consisting in the scalar quantization of the coefficients belonging to a predefined area (with a fixed bit allocation), and 'threshold coding', consisting in the choice of the coefficients of each block characterized by an absolute value exceeding a predefined threshold. Another possibility, that leads to higher compression factors, is to apply a vector quantization scheme to the transformed coefficients.

The same type of encoding is used for each coding method. In most cases a classical Huffman code can be used successfully. The JPEG and MPEG standards are examples of standards based on transform coding.

3.5.2 Vector quantization

A vector quantize can be defined mathematically as a transform operator T from a K-dimensional Euclidean space R^K to a finite subset X in R^K made up of N vectors. This subset X becomes the vector codebook or more generally, the codebook.

Clearly, the choice of the set of vectors is of major importance. The level of distortion due to the transformation T is generally computed as the most significant error (MSE) between the "real" vector x in R^K and the corresponding vector x' = T(x) in X. This error should be such as to minimize the Euclidean distance d. An optimum scalar quantizer was proposed by Lloyd and Max. Later on Linde, Buzo and Gray resumed and generalized this method, extending it to the case of a vector quantizer.

In the figure 3.6 vector quantization the compression is done based on compression factor and some distortion is produced on reconstructing the code.

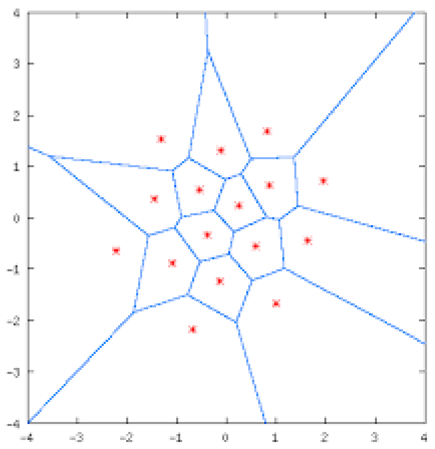


Fig 3.6: Vector quantization

The algorithm that they proposed is derived from the KNN cauterization method, and is performed by iterating the following basic operations:

1) Subdivide the training set into N groups (called 'partitions' or 'Voronoi regions'), which are associated with the N codebook letters, according to a minimum distance criterion.

2) The centroids of the Voronoi regions become the updated codebook vectors.

3) compute the average distortion: if the percent reduction in the distortion (as compared with the previous step) is below a certain threshold, then STOP.

The LBG algorithm for the design of a vector codebook always reaches a local minimum for the distortion function, but often this solution is not the optimal one. A careful analysis of the LBG algorithm's behavior allows one to detect two critical points: the choice of the starting codebook and the uniformity of the Voronoi regions' dimensions. For this reason some algorithms have been designed that give better performances. With respect to the initialization of the LBG algorithm, for instance, one can observe that a random choice of the starting codebook requires a large number of iterations before reaching an acceptable amount of distortion. Moreover, if the starting point leads to a local minimum solution, the relative stopping criterion prevents further optimization steps.

3.5.3 Segmentation and approximation methods

With segmentation and approximation coding methods, the image is modeled as a mosaic of regions, each one characterized by a sufficient degree of uniformity of its pixels with respect to a certain feature (e.g. grey level, texture); each region then has some parameters related to the characterizing feature associated with it.

> The operations of finding a suitable segmentation and an optimum set of approximating parameters are highly correlated, since the segmentation algorithm must take into account the error produced by the region reconstruction (in order to limit this value within determined bounds). These two operations constitute the logical modeling for this class of coding schemes; quantization and encoding are strongly dependent on the statistical characteristics of the parameters of this approximation (and therefore, on the approximation itself).

Classical examples are polynomial approximation and texture approximation. For polynomial approximation regions are reconstructed by means of polynomial functions in (x, y); the task of the encoder is to find the optimum coefficients. In texture approximation, regions are filled by synthesizing a parameterized texture based on some model (e.g. fractals, statistical methods, Markov Random Fields [MRF]). It must be pointed out that, while in polynomial approximations the problem of finding optimum coefficients is quite simple (it is possible to use least squares approximation or similar exact formulations), for texture based techniques this problem can be very complex.

3.5.4 Efficiency and quality of different lossy compression techniques

The performances of lossy picture coding algorithms are usually evaluated on the basis of two parameters:

a. the compression factor (or analogously the bit rate) and

b. the distortion produced on the reconstruction.

The first is an objective parameter, while the second strongly depends on the usage of the coded image. Nevertheless, a rough evaluation of the performances of a method can be made by considering an objective measure of the error, like MSE or SNR. For the methods described in the previous pages, average compression ratios and SNR values obtainable are presented in the following table.

Method	VQ	DCT-SQ	DCT-VQ	AP	Spline	TSDFractals
Bit rate	0.8-0.4	0.8-0.3	0.3-0.08	0.3-0.1	0.4-0.1	0.8-0.0
SNR(dB)	36-30	36-31	30-25	Image dependent	36-32	Image dependent

Table 3.2: Compression method

In the table 3.2 we compared bit rate and signal to noise ratio of different compression methods with relevant values to it.

3.6 Comparison of Different Compression Methods

During the last years, some standardization processes based on transform coding, such as JPEG, have been started. Performances of such a standard are quite good if compression factors are maintained under a given threshold (about 20 times). Over this threshold, artifacts become visible in the reconstruction and tile effect affects seriously the images decoded, due to quantization effects of the DCT coefficients.

On the other hand, there are two advantages: first, it is a standard, and second, dedicated hardware implementations exist. For applications which require higher compression factors with some minor loss of accuracy when compared with JPEG, different techniques should be selected such as wavelets coding or spline interpolation, followed by an efficient entropy encoder such as Huffman, arithmetic coding or vector quantization.

Some of these coding schemes are suitable for progressive reconstruction (Pyramidal Wavelet Coding, Two Source Decomposition, etc). This property can be exploited by applications such as coding of images in a database, for previewing purposes or for transmission on a limited bandwidth channel.

3.7 Lossy compression Vs Lossless compression

The advantage of lossy methods over lossless methods is that in some cases a lossy method can produce a much smaller compressed file than any known lossless method, while still meeting the requirements of the application.

Lossy methods are most often used for compressing sound, images or videos. The compression ratio of lossy video codec's are nearly always far superior to those of the audio and still-image equivalents. Audio can be compress at 1:10 with no noticeable loss of the quality; video can be compressed immensely with little visible quality loss, e300:1.

Lossy compressed images are still compressed to 1/10t their original size, as with audio, but the quality loss is more noticeable, especially on closer inspection. When a user acquires a lossy-compressed file, the retrieved file can be quite different to the original at the bit level while being indistinguishable to the human ear or eye for most practical purposes.

3.8 General Image Compression system

As shown in figure 3.7 below image compression systems are composed of two distinct structural blocks: an encoder & a decoder. Image f(x, y) is fed into the encoder, which creates a set of symbols from the input data &uses them to represent the image.

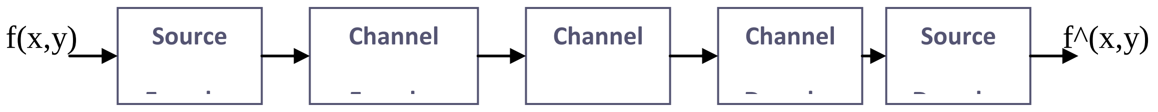


Figure 3.7: General image compression system block diagram.

If we let n1 & n2 denote the number of information carrying units in the original & encoded images respectively, the compression that is achieved can be quantified numerically via the compression ratio,

CR=n1/n2.

A compression ratio like 10 or 10:1 indicates that the original image has 10 information carrying units for every 1 unit in the compressed data set.

3.9 YUV component

Y'UV was invented when engineers wanted color television in a black-and-white infrastructure. They needed a signal transmission method that was compatible with black-and-white (B&W) TV while being able to add color. The luma component already existed as the black and white signal; they added the UV signal to this as a solution.

The UV representation of chrominance was chosen over straight R and B signals because U and V are color difference signals. This meant that in a black and white scene the U and V signals would be zero and only the Y' signal would need to be transmitted. If R and B were to have been used, these would have non-zero values even in a B&W scene, requiring all three data-carrying signals. This was important in the early days of color television, because holding the U and V signals to zero while connecting the black and white signal to Y' allowed color TV sets to display B&W TV without the additional expense and complexity of special B&W circuitry. In addition, black and white receivers could take the Y' signal and ignore the color signals, making Y'UV backward-compatible with all existing black-and-white equipment, input and output. It was necessary to assign a narrower bandwidth to the chrominance channel because there was no additional bandwidth available. If some of the luminance information arrived via the chrominance channel (as it would have if RB signals were used instead of differential UV signals), B&W resolution would have been compromised.

YUV(‘y’ is luminance, ‘u’ is red luminance, ‘v’ is green luminance) is a color space typically used as part of a color image pipeline. It encodes a color image or video taking human perception into account, allowing reduced bandwidth for chrominance components, thereby typically enabling transmission errors or compression artifacts to be more efficiently masked by the human perception than using a "direct" RGB-representation. Other color spaces have similar properties, and the main reason to implement or investigate properties of Y'UV would be for interfacing with analog or digital television or photographic equipment that conforms to certain Y'UV standards.

The scope of the terms Y'UV, YUV, YCbCr, YPbPr, etc., is sometimes ambiguous and overlapping. Historically, the terms YUV and Y'UV were used for a specific analog encoding of color information in television systems, while YCbCr was used for digital encoding of color information suited for video andstill-image compression and transmission such as MPEG and JPEG. Today, the term YUV is commonly used in the computer industry to describe file-formats that are encoded using YCbCr.

The Y'UV model defines a color space in terms of one luma (Y') and two chrominance (UV) components. The Y'UV color model is used in the PAL(phase alternating lock device) and SECAM(sequential color with memory) composite color video standards. Previous black-and-white systems used only luma (Y') information. Color information (U and V) was added separately via a sub-carrier so that a black-and-white receiver would still be able to receive and display a color picture transmission in the receiver's native black-and-white format.

Y' stands for the luma component (the brightness) and U and V are the chrominance (color) components; luminance is denoted by Y and luma by Y' the prime symbols (') denote gamma compression, with "luminance" meaning perceptual (color science) brightness, while "luma" is electronic (voltage of display) brightness.

The YPbPr color model used in analog component video and its digital version YCbCr used in digital video are more or less derived from it, and are sometimes called Y'UV. (C_B/P_B and C_R/P_R are deviations from grey on blue–yellow and red–cyan axes, whereas U and V are blue–luminance and red–luminance differences.) The Y'IQ color space used in the analog NTSC television broadcasting system is related to it, although in a more complex way.

3.9.1 Conversion to RGB

Y'UV signals are typically created from RGB (red, green and blue) source. Weighted values of R, G, and B are summed to produce Y', a measure of overall brightness or luminance. U and V are computed as scaled differences between Y' and the B and R values.

Defining the following constants:

$$\left.\begin{aligned} &WR=0.299\\ &WB=0.144\\ &WG=1-WR-WB=0.587 \end{aligned}\right\} \longrightarrow 3.1$$

$$U\ Max=0.436 \longrightarrow 3.2$$

$$V\ Max=0.615 \longrightarrow 3.3$$

Equations 3.1 describes the wieghted red,green and blue values. Equation 3.2 and 3.3 shows the corresponding luminance values of red and green.

Y'UV is computed from RGB as follows:

$$Y'=WRR+WGG+WBB \longrightarrow 3.4$$

$$U=UMax'\frac{B-Y'}{1-WB} \approx 0.492(B-Y') \longrightarrow 3.5$$

$$V=Vmax\frac{R-Y'}{1-WR} \approx 0.877(R-Y') \longrightarrow 3.6$$

Equations 3.4, 3.5 and 3.6 gives the Y'UV component from RGB.

The resulting ranges of Y', U, and V respectively are [0, 1], [-U_{Max}, U_{Max}], and [-V_{Max}, V_{Max}].

inverting the above transformation converts Y'UV to RGB. Equivalently, substituting values for the constants and expressing them as matrices gives:

$$\begin{pmatrix} Y' \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \longrightarrow 3.7$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{pmatrix} \begin{pmatrix} Y' \\ U \\ V \end{pmatrix} \longrightarrow 3.8$$

Equation 3.7 describes the Y’UV matrix values from that of RGB and the equation 3.8 describes the RGB values from by inverting Y’UV values.

CHAPTER 4
SOURCE CODE

4.1 History

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. He designed it to give his students access to LINPACK and EISPACK without them having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematics community. Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded Math Works in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK.

MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching oflinear algebra, numerical analysis, and is popular amongst scientists involved in image processing.

4.2 MATLAB

MATLAB® is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include

- Math and computation
- Algorithm development
- Data acquisition
- Modeling, simulation, and prototyping

- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including graphical user interface building.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non interactive language such as C or FORTRAN.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learnand apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

4.3 The MATLAB System

The MATLAB system consists of five main parts:

- Developing environment
- MATLAB mathematical function
- MATLAB language
- Graphics

- MATLAB application program interface

4.3.1 Development Environment

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.

4.3.2 The MATLAB Mathematical Function

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigen values, Bessel functions, and fast Fourier transforms.

4.3.3 The MATLAB Language

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs. MATLAB can call functions and subroutines written in the C programming language or Fortran. A wrapper function is created allowing MATLAB data types to be passed and returned. The dynamically loadable object files created by compiling such functions are termed "MEX-files" (for **M**ATLAB **ex**ecutable).

Libraries written in Perl, Java, ActiveX or .NET can be directly called from MATLAB, and many MATLAB libraries (for example XML or SQL support) are implemented as wrappers around Java or ActiveX libraries. Calling MATLAB from Java is more complicated, but can be done with a MATLAB toolbox which is sold separately by MathWorks, or using an undocumented mechanism called JMI (Java-to-MATLAB

Interface), (which should not be confused with the unrelated Java Metadata Interface that is also called JMI).

As alternatives to the MuPAD based Symbolic Math Toolbox available from MathWorks, MATLAB can be connected to Maple or Mathematica.Libraries also exist to import and export MathML

4.3.4 Graphics

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications. MATLAB supports developing applications with graphical user interface features. MATLAB includes GUIDE (GUI development environment) for graphically designing GUIs. It also has tightly integrated graph-plotting features. For example the function plot can be used to produce a graph from two vectors x and y.

5.3.5 The MATLAB Application Program Interface (API)

This is a library that allows you to write C and Fortran programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

4.4 MATLAB working environment

The following shows the entire MATLAB working environment

4.4.1 MATLAB desktop

MATLAB Desktop is the main MATLAB application window. The desktop contains five sub windows, the command window, the workspace browser, the current directory window, the command history window, and one or more figure windows, which are shown only when the user displays a graphic.

The command window is where the user types MATLAB commands and expressions at the prompt (>>) and where the output of those commands is displayed. MATLAB defines the workspace as the set of variables that the user creates in a work session. The workspace browser shows these variables and some information about them. Double clicking on a variable in the workspace browser launches the Array Editor, which can be used to obtain information and income instances edit certain properties of the variable.

The current directory tab above the workspace tab shows the contents of the current directory, whose path is shown in the current directory window. For example, in the windows operating system the path might be as follows: C:\MATLAB\Work, indicating that directory “work” is a subdirectory of the main directory “MATLAB”; WHICH IS INSTALLED IN DRIVE C. clicking on the arrow in the current directory window shows a list of recently used paths. Clicking on the button to the right of the window allows the user to change the current directory.

MATLAB uses a search path to find M-files and other MATLAB related files, which are organize in directories in the computer file system. Any file run in MATLAB must reside in the current directory or in a directory that is on search path. By default, the files supplied with MATLAB and math works toolboxes are included in the search path. The easiest way to see which directories are on the search path. The easiest way to see which directories are soon the search path, or to add or modify a search path, is to select set path from the File menu the desktop, and then use the set path dialog box. It is good practice to add any commonly used directories to the search path to avoid repeatedly having the change the current directory.

The Command History Window contains a record of the commands a user has entered in the command window, including both current and previous MATLAB sessions. Previously entered MATLAB commands can be selected and re-executed from the command history window by right clicking on a command or sequence of commands.

This action launches a menu from which to select various options in addition to executing the commands. This is useful to select various options in addition to executing the commands. This is a useful feature when experimenting with various commands in a work session.

4.4.2 Using the MATLAB Editor to create M-Files

The MATLAB editor is both a text editor specialized for creating M-files and a graphical MATLAB debugger. The editor can appear in a window by itself, or it can be a sub window in the desktop. M-files are denoted by the extension .m, as in pixelup.m. The MATLAB editor window has numerous pull-down menus for tasks such as saving, viewing, and debugging files. Because it performs some simple checks and also uses color to differentiate between various elements of code, this text editor is recommended as the tool of choice for writing and editing M-functions. To open the editor , type edit at the prompt opens the M-file filename.m in an editor window, ready for editing. As noted earlier, the file must be in the current directory, or in a directory in the search path.

Figure 4.1 shows the internal MATLAB operation. In the MATLAB there are program interface language which is a intermediate language which can easily understand by the user and some graphic cards, tool boxes and external interface is in builted to it.

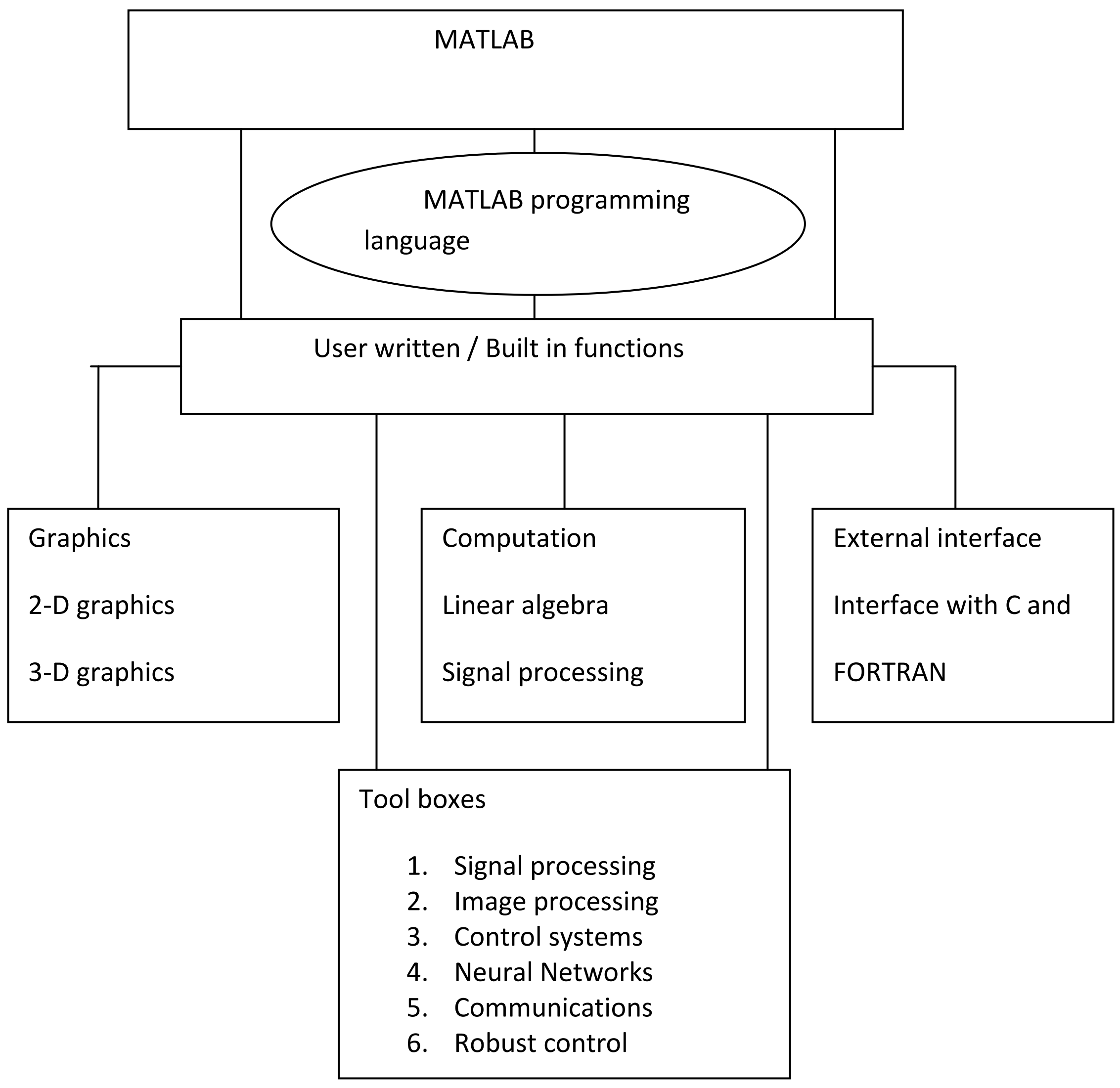
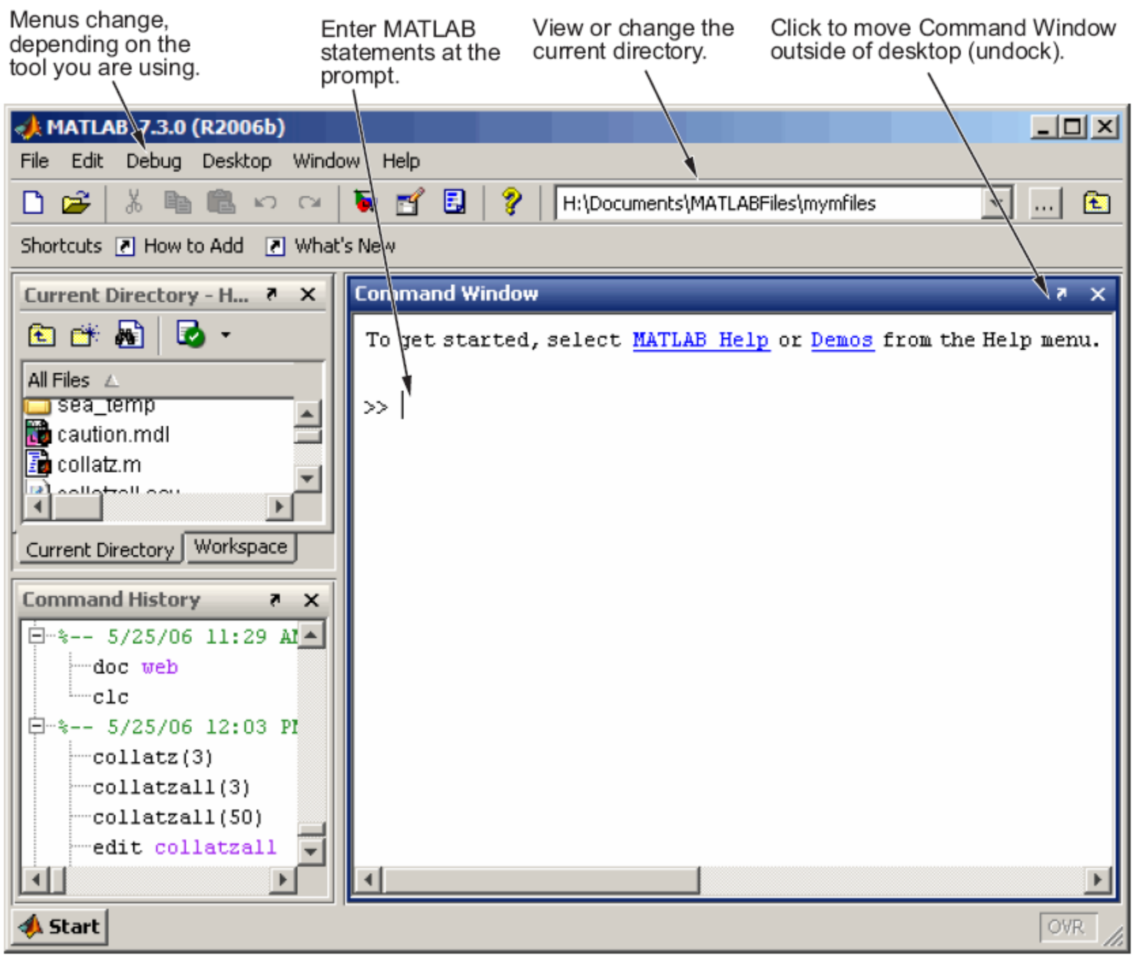


Fig 4.1 Internal MATLAB operation

4.5 Getting Help

The principal way to get help online is to use the MATLAB help browser, opened as a separate window either by clicking on the question mark symbol (?) on the desktop toolbar, or by typing help browser at the prompt in the command window. The help

Browser is a web browser integrated into the MATLAB desktop that displays a Hypertext Markup Language(HTML) documents. The Help Browser consists of two panes, the help navigator pane, used to find information, and the display pane, used to view the information. Self-explanatory tabs other than navigator pane are used to perform a search.


Menus change, depending on the tool you are using.
Enter MATLAB statements at the prompt.
View or change the current directory.
Click to move Command Window outside of desktop (undock).
MATLAB 7.3.0 (R2006b)
File Edit Debug Desktop Window Help
H:\Documents\MATLABFiles\mymfiles
Shortcuts How to Add What's New
Current Directory - H...
All Files
sea_temp
caution.mdl
collatz.m
Current Directory
Workspace
Command History
Command Window
Start
OVR

```
To get started, select MATLAB Help or Demos from the Help menu.

>> |
```

```
%-- 5/25/06 11:29 AM
   doc web
   clc
%-- 5/25/06 12:03 PM
   collatz(3)
   collatzall(3)
   collatzall(50)
   edit collatzall
```

Fig 4.2 MATLAB window

In figure 4.2 MATLAB window we can see the command window and to create new M-files and having command history and command directory.

CHAPTER 5

RESULTS ANALYSIS

5.1 Input image

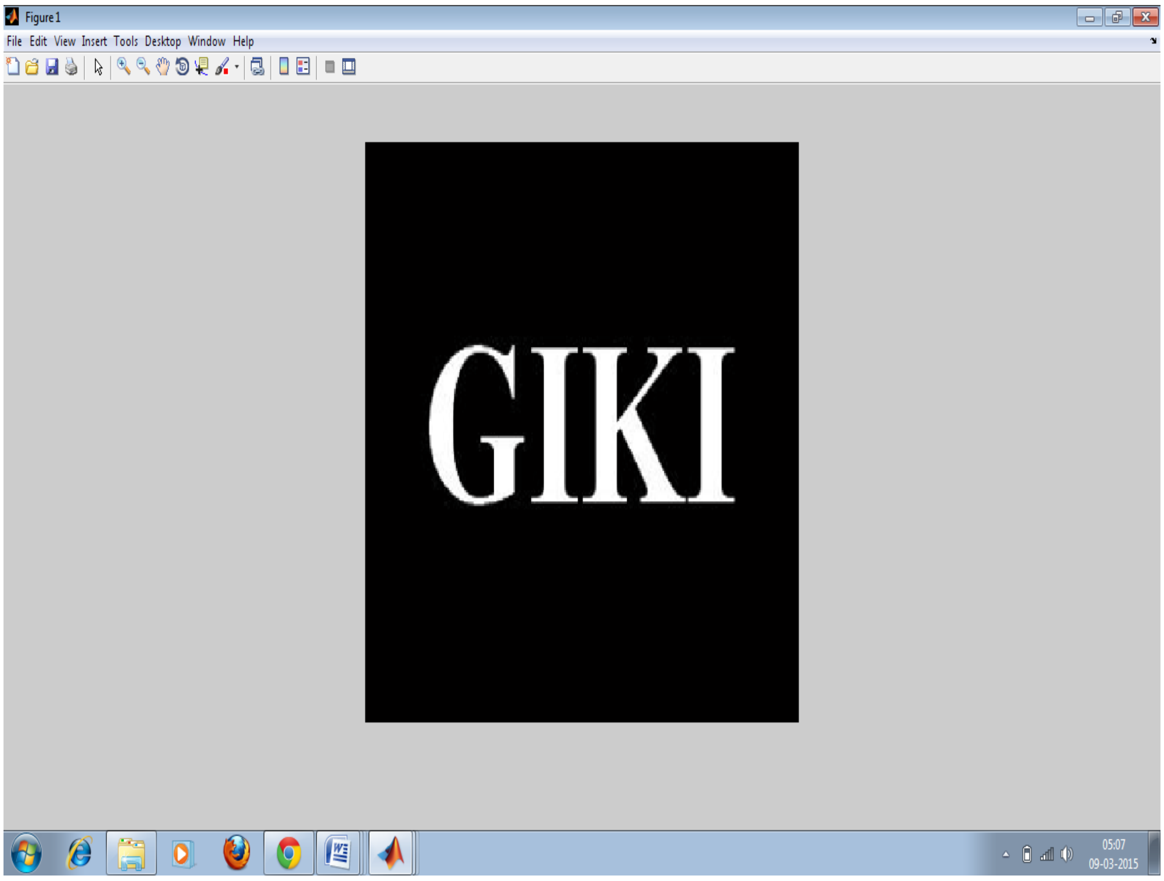


Fig 5.1 Logo image

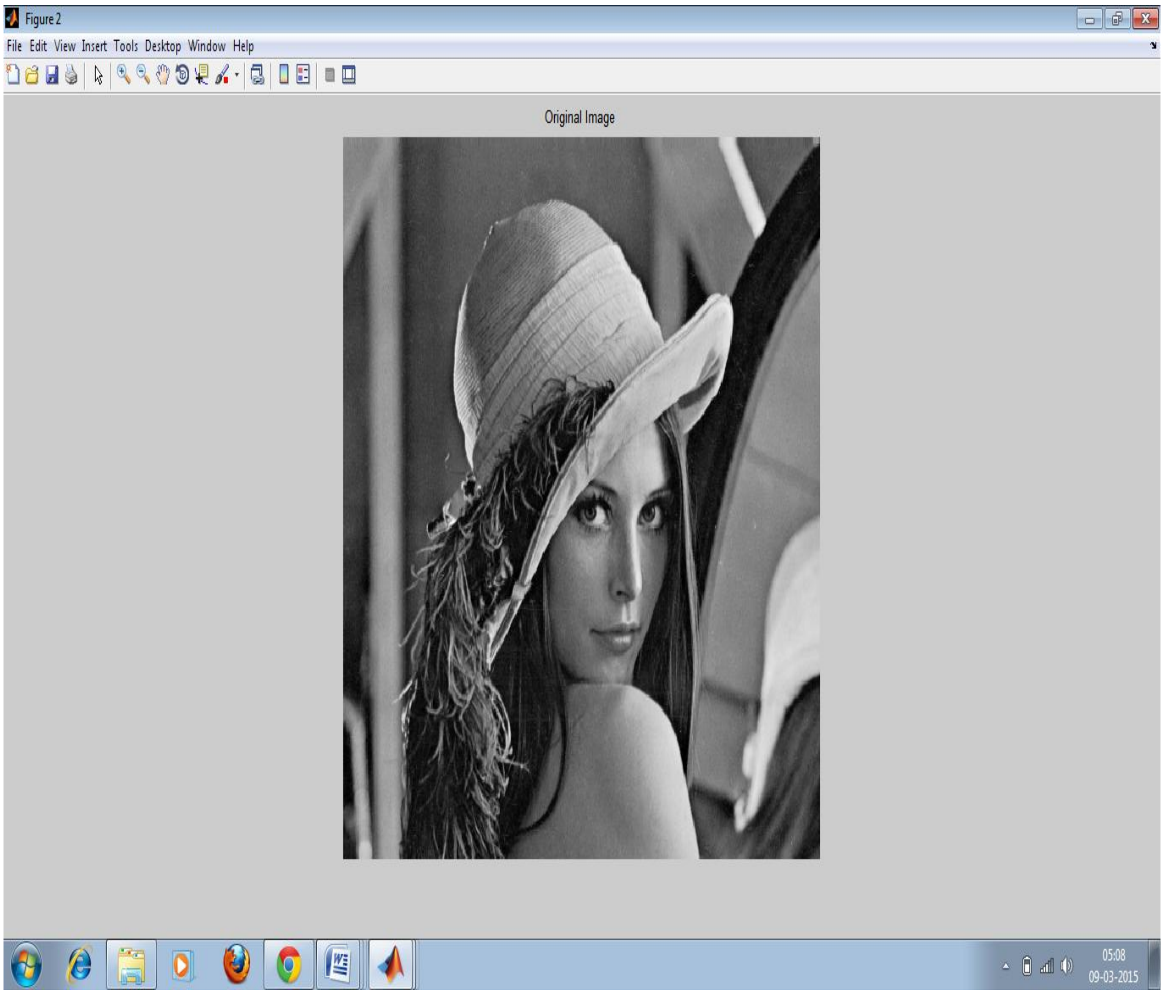


Fig 5.2 Original image

5.2 Out watermarked image

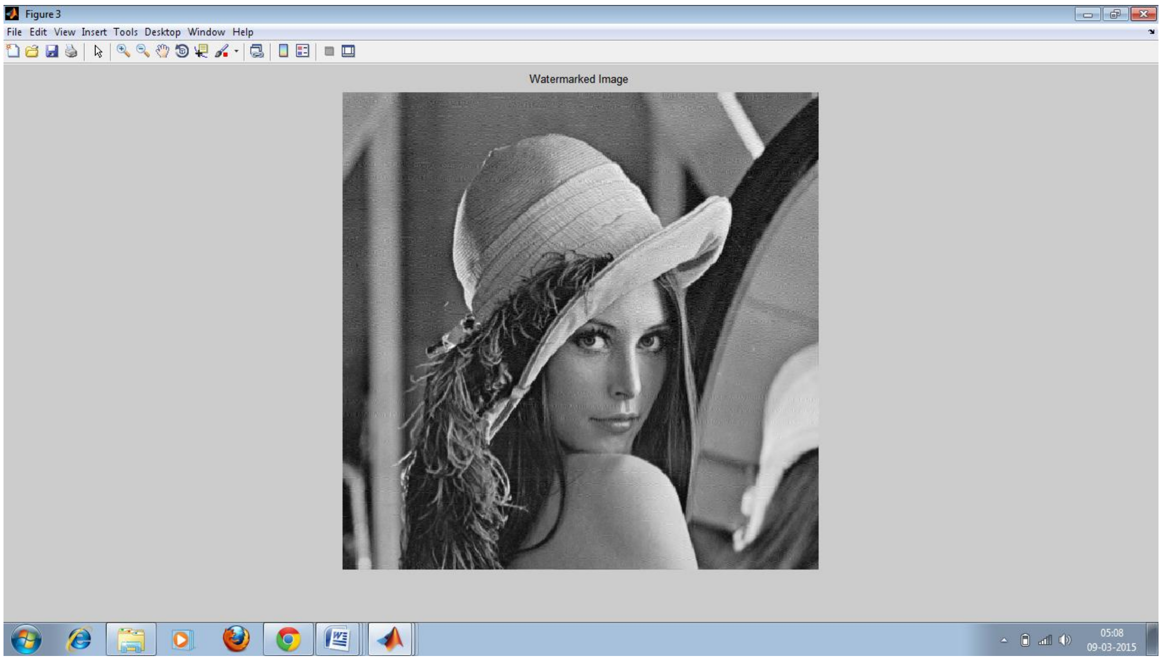


Fig 5.3 Watermarked image

5.3 Retrieved watermarked image

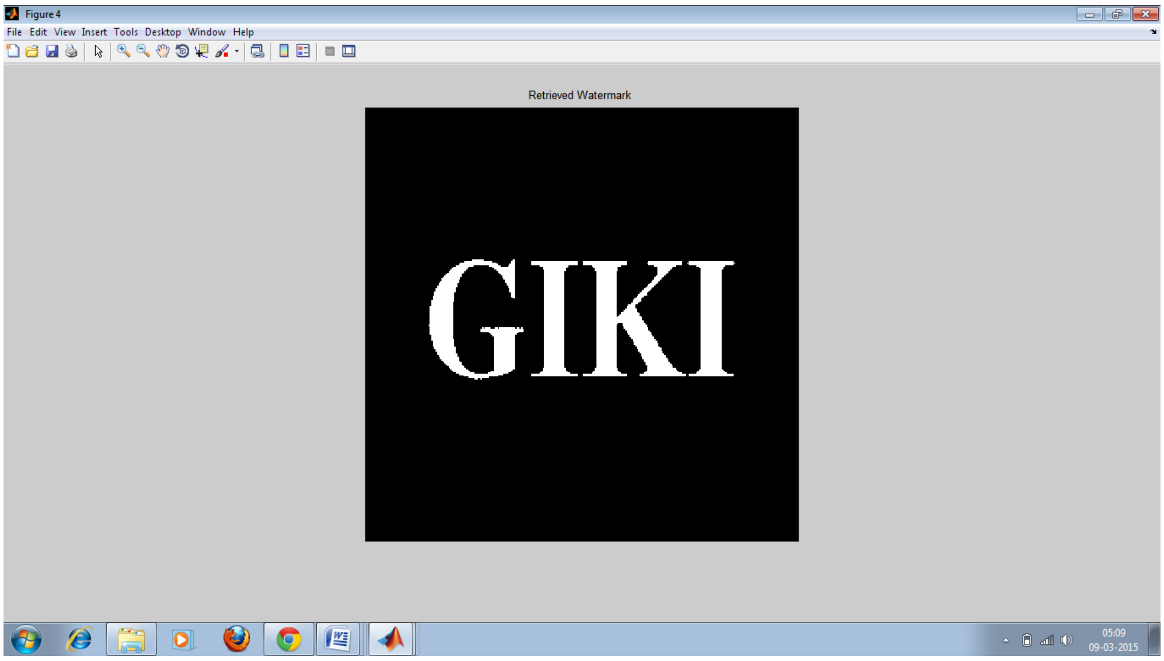


Fig 5.4 Retrieved watermark

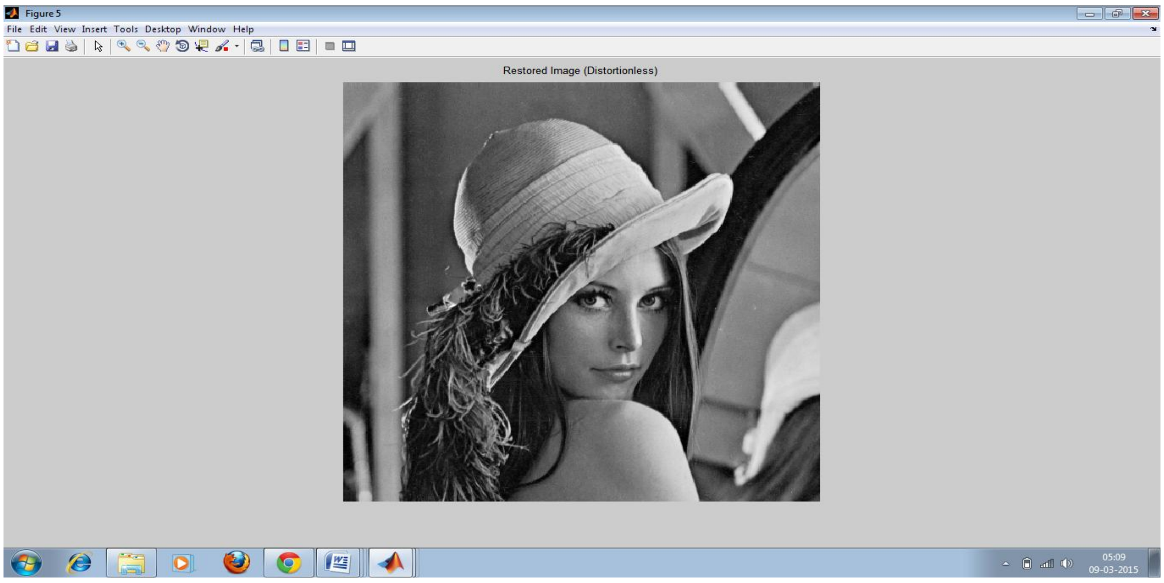


Figure 5
File Edit View Insert Tools Desktop Window Help
Restored Image (Distortionless)
05:09
09-03-2015

Fig 5.5 Original image

5.4 Difference image

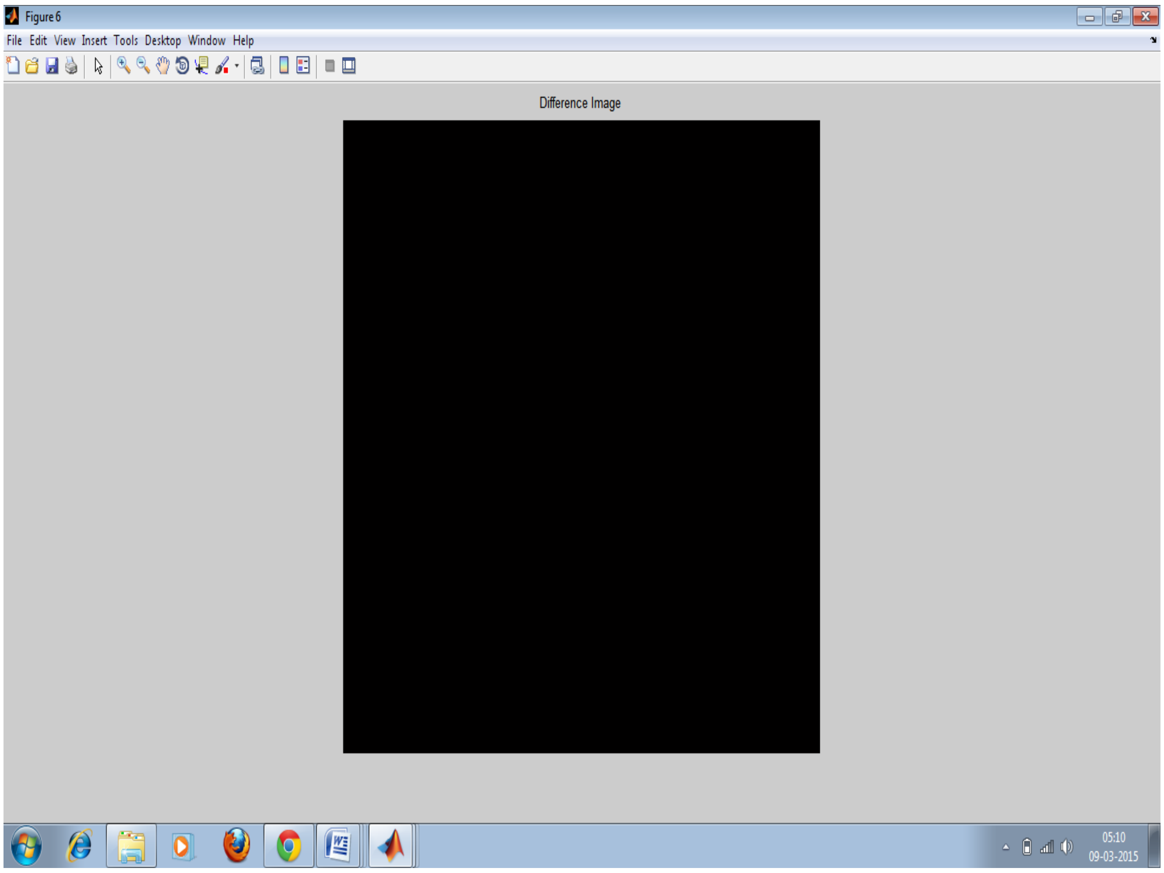


Fig 5.6 Difference image

5.5: Analysis of result on images

We performed our operation on two images, that is figure 5.1 original image, figure 5.2 compressed image. Here the two images are compressed in to single image by means of water marking. The figure 5.3 watermarked image contains two images in it. We can't find the difference between original image because it looks like same. But the person who knows the actual operation can know what is the image hidden contains in that image. Here the watermarked image is having less intensity than the original image. When successful extraction is made we have figure 5.4 retrieved watermark and the figure 5.5 original image from the watermarked image. We can also have figure 5.6 the difference of original image and watermarked image**.**

While on the other side in the command window we have our compressed bits which will make us to understand more easily about the image. In the embedding part we observe that compression of the logo image bits in to lesser one. In the extraction part we are going to retrieve the same image.

CHAPTER 6
CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

This work presents a novel Reversible De- Identification method for lossless compressed images. The proposed scheme is generic and can be employed with other obfuscation strategies other than *k*-Same. A two level Reversible-Watermarking scheme was adopted which uses Differential Evolution to find the optimal set of thresholds and provides a single-pass embedding capacity close to 1.25 bpp. Simulation results have shown that this method is able to recover the original image if the correct encryption key is employed. It further shows that 0.8 bpp were sufficient to cater for 99.8% of the frontal images considered and none of the image needed more than 1.1 bpp. Future work will focus on the extension of this algorithm for lossy image and video compression standards.

6.2 Future scope

Till now we discussed about single level watermarking i.e; hiding only one image in to another image. But in the future we do many like this in to a single image. This involves high security and less transmission time to send the entire data. But the only problem in this is the watermarked image having distortions.

References

[1] A. Senior, S. Pankanti, A. Hampapur, L. Brown, Y. Li Tian and A. Ekin, "Blinkering Surveillance: Enabling video privacy through Computer Vision," IBM Research Report, vol. 22886, 2003.

[2] E.M. Newton, L. Sweeney and B. Malin, "Preserving privacy by de-identifying face images," IEEE Trans. on Knowl and DataEng., vol. 17, no. 2, pp. 232-243, Feb. 2005.

[3] W. Zhang, S.S. Cheung and M. Chen, "Hiding privacy information in video surveillance systems," in IEEE Int. Conf. on Image Processing, Genoa, Italy, Sep. 2005.

[4] I. Martinez-Ponte, X. Desumont, J. Meessen and J.F. Delaigle, "Robust Human Face Hiding ensuring Privacy," in Proc. of Int. Workshop on Image Analysis for Multimedia Services, Montreux, Switzerland, Apr. 2005.

[5] T.E. Boult, "Pico: Privacy through invertible cryptographic obscuration," in IEEE Proc. of the Computer Vision for Interactive Intelligent Environment, Washington DC, USA, Nov. 2005.

[6] K. Martin and K.N. Plataniotis, "Privacy protected surveillance using secure visual object coding," IEEE Trans. Circuits and System for Video Technol., vol. 18, no. 8, pp. 1152-1162, Aug. 2008.

[7] F. Dufaux, M. Ouaret, Y. Abdeljaoued, A. Navarro, F.Bergnenegre and T. Ebrahimi, "Privacy Enabling Technology for Video Surveillance," in SPIE Mobile Multimedia/Image Processing for Military and Security Applications, Orlando, Florida, May 2006.

[8] F. Dufaux and T. Ebrahimi Scrambling for privacy protection in video surveillance systems, "Scrambling for privacy protection in video surveillance systems," in IEEE Trans on Circuits and Systems for Video Technol., vol. 18, no. 8, pp. 1168-1178, Aug. 2008.

[9] H. Sohn, W. De Neve and Y-M. Ro, "Privacy protection in video surveillance systems: Analysis of subband-adaptive scrambling in JPEG XR," in IEEE Trans. Circuits and Systems for Video Technol., vol. 21, no. 2, pp. 170-177, Feb. 2011.

[10] J. Meuel, M. Chaumont and W. Puech, "Data hiding in H.264 video for lossless reconstruction of region of interest," in European Signal Processing Conf., Poznan, Poland, Sep. 2007.