Comparison of Bacterial Foraging Optimization (BFO) Neural Network with Haar Wavelet Transform in Image Compression

A Thesis submitted in partial fulfillment of the Requirements for the award of degree of

Master of Engineering
In
Electronics Instrumentation and Control

Submitted by
MUKESH MITTAL
(Roll No. 801151013)

Under the Guidance of
Ms. Ruchika Lamba
Lecturer

Department of Electrical and Instrumentation Engineering
Thapar University
(Established under the section 3 of UGC act, 1956)
Patiala, 147004, Punjab, India, July 2013

DECLARTION

I hereby certify that the work which is being presented in the thesis entitled, "**Comparison of Bacterial Foraging Optimization (BFO) Neural Network Algorithm with Haar Wavelet Transform (HWT) in Image Compression**" in partial fulfillment of the award of degree of **Master of Engineering in Electronics Instrumentation and Control** submitted in the Electrical and Instrumentation Engineering Department, Thapar University, Patiala is an authentic record of my own work under the supervision of Ms. Ruchika Lamba, Lecturer, Department of the Electrical and Instrumentation Engineering, Thapar University, Patiala, Punjab.

Mukesh Mittal

MUKESH MITTAL
801151013

Date: 12/07/2013

I certify that the above statement made by the student is correct to the best of my knowledge and belief.

Ruchika

Ms. Ruchika Lamba
Lecturer
Department of the Electrical and
Instrumentation Engineering,
Thapar University, Patiala, Punjab.

Countersigned By

S. Ghosh

Dr. Smarajit Ghosh
Head of Department
Department of the Electrical and
Instrumentation Engineering,
Thapar University, Patiala, Punjab.

Dr. S.K. Mohapatra
Dean of Academic Affairs
Thapar University, Patiala
Punjab

ABSTRACT

With the growth of multimedia and internet, compression techniques have become the thrust areas in the field of computers. Compression of data in any form is a large and active field as well as a big business. The problem inherent to any digital image is the large amount of bandwidth required for transmission or storage. This has driven the research area of image compression to develop algorithms that compress images to lower data rates with better quality. Popularity of multimedia has led to the integration of various types of computer data. Multimedia combines many data types such as text, graphics, still images, animation, audio and video.

A fundamental goal of image compression is to reduce the bit rate for transmission or data storage while maintaining an acceptable fidelity or image quality. Every digital image is specified by the number of pixels associated with the image. Each pixel in an image can be denoted as a coefficient, which represents the intensity of the image at that point. Image compression is a process of representing an image with fewer bits while maintaining image quality, saving cost associated with sending less data over communication lines and finally reducing the probability of transmission errors. Many different image compression techniques currently exist for the compression of different types of images. In this research work, Haar Wavelet Transform (HWT) and Bacterial Foraging Optimization (BFO) algorithm has been used. The Haar wavelet transform (HWT) is one of the simplest and basic transformations from a space domain are local frequency domain and it reduces the calculation work. HT decomposes the linear approximated image as approximation components and detail components.

In Bacterial Foraging Optimization (BFO) algorithm, we find the vertical and the horizontal configuration section. Neural network decides a loop for the processing. The loop is called iteration in the scenario. The BFO algorithm proceeds from pixel to pixel. If the pixel width is more than that of the previous pixel width then it is turned to be compressed again by BFO algorithm. This algorithm is well suited in JPEG format files to transmission of images with good quality and lesser storage over networks as compared to Haar Wavelet Algorithm (HWT). The proposed schemes has been demonstrated through several experiments such as Lena, Baboon, Penguins, Koala and Bacterial Foraging Optimization (BFO) algorithm gives very promising results in compression image over Haar Wavelet Transform (HWT) algorithm.

ACKNOWLEDGEMENT

During my M.E study at Thapar University, Patiala, Punjab, I have been fortunate to receive the valuable suggestions, guidance and support from my mentors, colleagues, family and friends.

First of all, I would like to express my most sincere gratitude to my supervisor **Ms. Ruchika Lamba**. She has been a wise and trusted guide throughout the entire process. Her guidance helped me to solve engineering problems and improve my communication skills. I am thankful to her that she has full confidence in my ability; much of this work would have not been completed without her vision and encouragement.

I am thankful to **Dr. Smarajit Ghosh**, Head of Department, Electrical and Instrumentation Engineering Department, Thapar University, Patiala for his encouragement and support. I am thankful to all the faculty members and staff members of department of Electrical and Instrumentation Engineering, Thapar University for their support during my academic years.

This section will look incomplete if I fail to thank almighty God and all my near and dear friends and my family members who stood beside me, understood my academic goals and helped me to achieve it. Last but not least by any means, my heartiest thanks to all the persons, who made me what I am today; word fails to express my feelings for them.

Mukesh Mittal

MUKESH MITTAL

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Overview

Uncompressed multimedia (graphics, audio and video) data requires considerable storage capacity and transmission bandwidth. Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of available technologies. The recent growth of data intensive multimedia-based web applications have not only sustained the need for more efficient ways to encode signals and images but have made compression of such signals central to storage and communication technology. In a distributed environment, large image files remain a major bottleneck within systems.

Image Compression is an important component of the solutions available for creating image file sizes of manageable and transmittable dimensions. Platform portability and performance are important in the selection of the compression/decompression technique to be employed.

Images have considerably higher storage requirement than text; Audio and Video Data require more demanding properties for data storage. An image stored in an uncompressed file format, such as the popular BMP or JPEG format can be huge. An image with a pixel resolution of 640 by 480 pixels and 24-bit color resolution will take up 640 * 480 * 24/8 = 921,600 bytes in an uncompressed format. The huge amount of storage space is not only the consideration but also the data transmission rates for communication of continuous media are also significantly large. This kind of data transfer rate is not realizable with today's technology, or in near the future with reasonably priced hardware. Image compression is essential for applications such as TV transmission, video conferencing, facsimile transmission of printed material, graphics images. A fundamental goal of image compression is to reduce the bit rate for transmission or data storage while maintaining an acceptable fidelity or image quality. Every digital image is specified by the

number of pixels associated with the image. Each pixel in an image can be denoted as a coefficient, which represents the intensity of the image at that point. Once compressed, the coded image is transferred to the receiving end, where these compressed images are again decompressed to recover the original image.

It is known that compression algorithms can be classified into two types – 'lossy' and 'lossless'. If the recovered image (after decompression) does not have the same quality as the original image then there has been a loss of some image data during compression. This is called a 'lossy compression algorithm'. But some algorithms have the ability to retain the quality of the image, even after the compression, and the decompression processes. Such algorithms come under the category of 'lossless compression algorithms.

1.2 Motivation

As progress took place in research in the field of image processing in the development of its applications. So, in the field of image compression, there is a requirement for the better compression as much as can be done as in an uncompressed data or image, large amount of storage capacity is required. But after compression, less amount of storage capacity is required. Wavelet transform is an emerging area in the field of research in applied mathematics. They have undergone a drastic transformation during the last twenty years In the applied application field. Wavelets are a step closer to reality than its counterpart transforms hence they have always been in the eyes of researchers. Wavelets provide a very dynamic field of application and are applicable to image too. Bacteria Foraging algorithm is a new comer to the family of nature-inspired optimization algorithms. For over the last five decades, optimization algorithms like Genetic Algorithms (GAs), Evolutionary Programming (EP), Evolutionary Strategies (ES), which draw their inspiration from evolution and natural genetics, have been dominating the realm of optimization algorithms.

1.3 Objective of Thesis

The objective of the work carried out in this dissertation can be stated in following points:

- To study and implement the Haar Wavelet transform (HWT) on different test images namely Lena, Baboon, Penguins and Koala.

- To study and implement Bacterial Foraging Optimization (BFO) Algorithm on different test images namely Lena, Baboon, Penguins and Koala.

- Comparison of results based on Bacterial Foraging Optimization (BFO) Algorithm as well as Haar Wavelet transform (HWT) on different images.

1.4 Organization of Thesis

Chapter 1 gives a brief introduction and overview of the work. It highlights the objective that is desired to be achieved.

Chapter 2 of the thesis gives an introduction to image compression sufficiently enough for the purpose of engineering applications. It also presents a literature review necessary for the purpose of the research carried out.

Chapter 3 of the thesis discusses Haar Wavelet Transform (HWT) in detail.

Chapter 4 of the thesis discusses Bacterial Foraging Optimization (BFO) neural network algorithm which is the proposed work.

Chapter 5 of the thesis discusses the MATLAB in Graphical User Interface (GUI).

Chapter 6 discusses the results obtained by applying Bacterial Foraging Optimization (BFO) algorithm and Haar Wavelet Transform (HWT) and their comparison.

Chapter 7 contains the concluding remarks of the thesis. It also explores the future aspect of the subject.

CHAPTER 2

IMAGE COMPRESSION

2.1 Introduction

Compression of data in any form is a large and active field as well as a big business. Image compression is a subset of this huge field of data compression, where we undertake the compression of image data specifically. Uncompressed multimedia (graphics, audio and video) data requires considerable storage capacity and transmission bandwidth. Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of available technologies. The recent growth of data intensive multimedia-based web applications have not only sustained the need for more efficient ways to encode signals and images but have made compression of such signals central to storage and communication technology. To enable Modern High Bandwidth required in wireless data services such as mobile multimedia, email, mobile, internet access, mobile commerce, mobile data sensing in sensor networks, Home and Medical Monitoring Services and Mobile Conferencing, there is a growing demand for rich Content Cellular Data Communication, including Voice, Text, Image and Video.

One of the major challenges in enabling mobile multimedia data services will be the need to process and wirelessly transmit very large volume of this rich content data. This will impose severe demands on the battery resources of multimedia mobile appliances as well as the bandwidth of the wireless network. While significant improvements in achievable bandwidth are expected with future wireless access technology, improvements in battery technology will lag the rapidly growing energy requirements of the future wireless data services.

2.2 Need of Compression

In the last decade, there has been a lot of technological transformation in the way we communicate. This transformation includes the ever present, ever growing internet, the explosive development in mobile communication and ever increasing importance of video communication.

Data Compression is one of the technologies for each of the aspect of this multimedia revolution. Cellular phones would not be able to provide communication with increasing clarity without data compression. Data compression is art and science of representing information in compact form.

Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of available technologies. In a distributed environment large image files remain a major bottleneck within systems.

Image Compression is an important component of the solutions available for creating image file sizes of manageable and transmittable dimensions. Platform portability and performance are important in the selection of the compression/decompression technique to be employed.

Image compression is essential for applications such as TV transmission, video conferencing, facsimile transmission of printed material, graphics images [1]. A fundamental goal of image compression is to reduce the bit rate for transmission or data storage while maintaining an acceptable fidelity or image quality. Every digital image is specified by the number of pixels associated with the image. Each pixel in an image can be denoted as a coefficient, which represents the intensity of the image at that point. Image compression is a process of: representing an image with fewer bits while maintaining image quality ; saving cost associated with sending less data over communication lines and finally reducing the probability of transmission errors. According to literature studies, we need compression technique that leads to less storage requirements and best Compression Ratio (CR). [2]

Digital images have become very popular in recent times. Every digital image is specified by the number of pixels associated with the image. Each pixel in a gray-level image is described by an intensity of the image at that point. An image that is, 256 x 256, means that there are 65536 pixels (intensity points) in the image in a matrix form with 256 rows and 256 columns.

Digital images are basically classified into two types: grayscale images and color images. Any color can be defined by the combination of the three primary colors – red, green and blue. A

Gray scale image has no color information. Therefore, every pixel in a grayscale image has different shade of gray which is commonly represented by 8 (Unsigned integers of 8 bits). So, there is $2^8 = 256$ possible intensity values (shades of gray) for a grayscale image ranging from 0 to 255. The depth of the image is said to be 8, since 8 bits are used to represent each pixel. Since 8 bits are used to represent each pixel, to represent an image which is of dimension [256 x 256,

256 x 256 x 8] = 524288 bits are needed to represent the image. With limited memory space, it becomes useful to compress the digital image so that it occupies less memory and also becomes easier to share over a medium such as the internet.

2.3 Image Compression Models

Image compression is minimizing the size in bytes of a graphics file without degrading the quality of the image to an unacceptable level. The reduction in file size allows more images to be stored in a given amount of disk or memory space. It also reduces the time required for images to be sent over the Internet or downloaded from web pages.

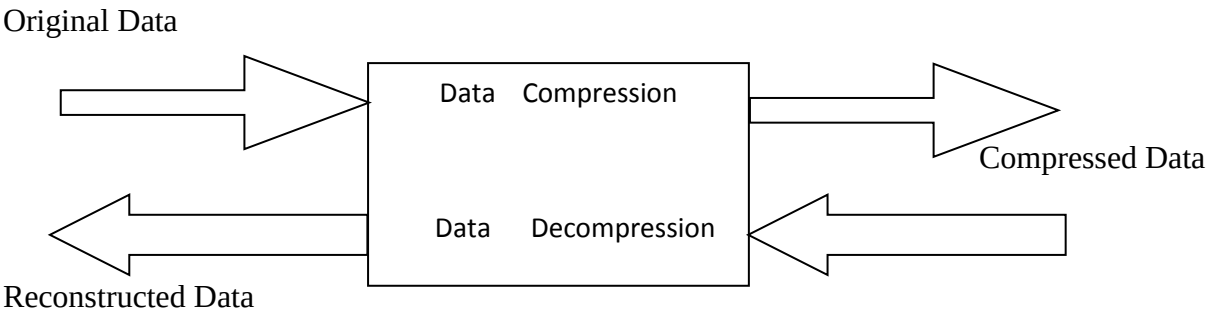


Figure 2.3.1 Compression Block Diagram

Image compression can be understood as a method that takes an input original data and generates the compressed data with less number of bits compared to that of original data. The reverse process is called decompression, which takes the compressed data and generates or reconstructs the data as shown in Figure 2.3.1

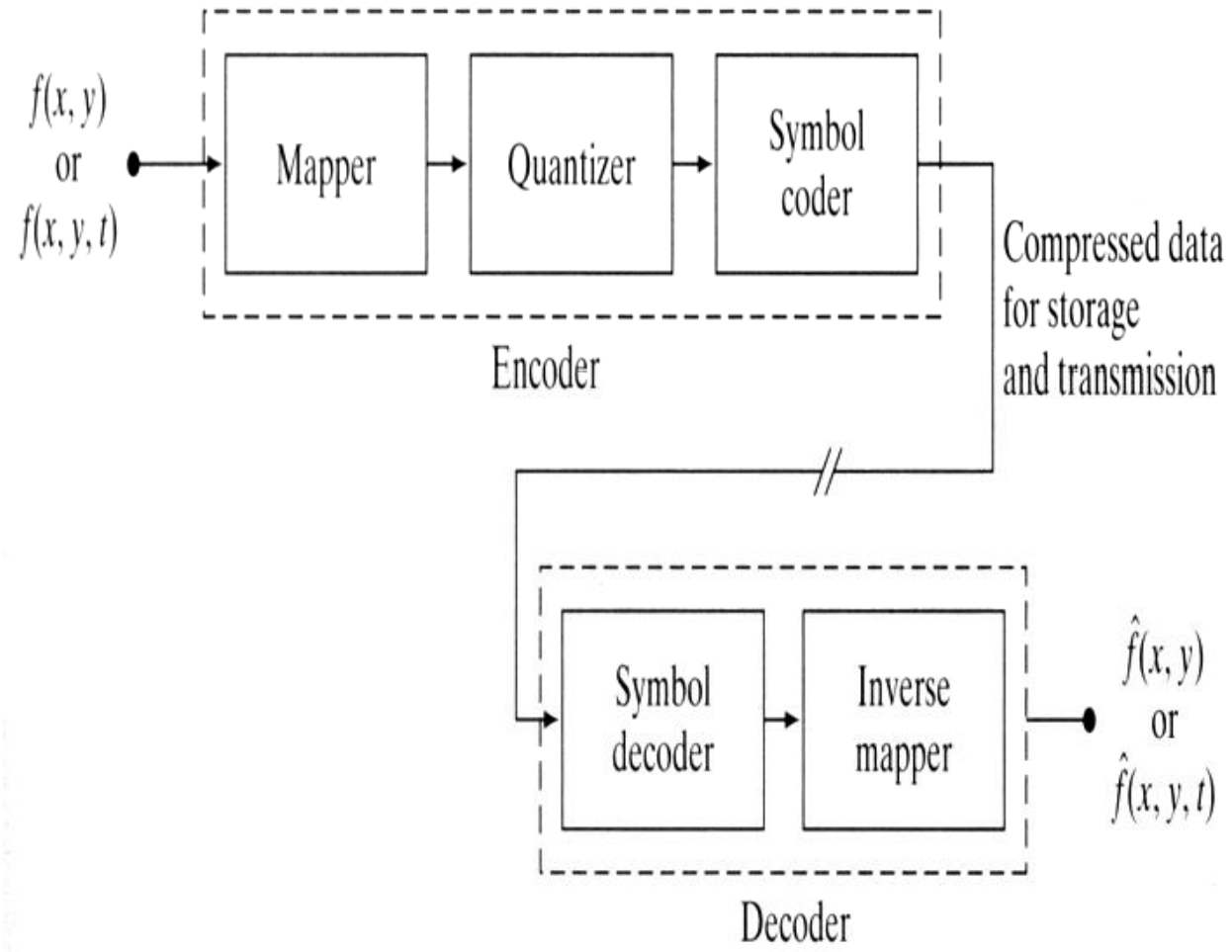


Figure 2.3.2 General Image Compression System [3]

In Figure 2.3.2, the encoder performs compression, the decoder performs decompression. Both operations can be done in software (web browsers, image viewers) or in a combination of hardware and firmware (DVD players, digital cameras). A Codec is a device or program that is capable for both encoding and decoding.

The mapper transforms an image or their sequence into a usually invisible format designed to reduce spatial and temporal redundancy. Usually, this operation is reversible, and may or may not reduce the amount of data needed to represent the image. Run-Length coding is an example of mapping.

The quantizer reduces the accuracy of the mapper's output according to the fidelity criterion. This operation is irreversible and targets to remove irrelevant information from the image. When error-free compression is needed, this method is omitted.

The final stage of the encoding process, the symbol coder**,** generates a fixed or variable length code to represent the quantizer output according to the code. Usually, to minimize coding redundancy, the shortest code words are assigned to the most frequently occurring quantizer output values. This operation is reversible.

These three operations lead to removal of all the three redundancies from the input image.

The decoder contains two components namely symbol decoder and an inverse mapper performing the inverse operations of the encoder's symbol coder and mapper. The inverse quantizer block is not included since quantization is irreversible.

There are several different ways in which image files can be compressed. For Internet use, the two most common compressed graphic image formats are the JPEG format and the GIF format. The JPEG method is more often used for photographs, while the GIF method is commonly used for line art and other images in which geometric shapes are relatively simple.

Other techniques for image compression include the use of fractals and wavelets. These methods have not gained widespread acceptance for use on the Internet. However, both methods offer higher compression ratios than the JPEG or GIF methods for some types of images. Another new method that may in time replace the GIF format is the PNG format.

A text file or program can be compressed without the introduction of errors, but only up to a certain extent. This is called lossless compression. Beyond this point, errors are introduced. In text and program files, it is crucial that compression be lossless because a single error can seriously damage the meaning of a text file, or cause a program not to run. In image compression, a small loss in quality is usually not noticeable. There is no "critical point" up to which compression works perfectly, but beyond which it becomes impossible. When there is some tolerance for loss, the compression factor can be greater than it can when there is no loss tolerance. For this reason, graphic images can be compressed more than text files or programs loss tolerance. [3]

2.4 Techniques of image Compression

The purpose of image compression is to represent images with less data in order to save storage costs or transmission time. Without compression, file size is significantly larger, usually several megabytes, but with compression it is possible to reduce file size to 10 percent from the original without noticeable loss in quality. Image compression can be lossless or lossy.

2.4.1 Lossless compression

With lossless compression, every single bit of data that was originally in the file remains after the file is decompressed. All of the information is completely restored. This is generally the technique of choice for text or spreadsheet files where losing words or financial data could pose a problem. The Graphics Interchange File (GIF) is an image format used on the Web that provides lossless compression. Lossless image compression is usually used in artificial images that contain sharp-edged lines such as technical drawings, textual graphics, comics, maps or logos. This is because lossy compression methods produce compression artifacts to images and sharp-edged lines become fuzzy especially when using strong compression. Figure 2.4.1.1 shows the lossless compression system.

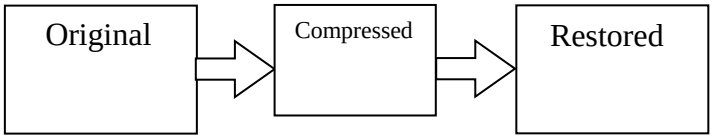


Figure 2.4.1.1: Lossless Compression

The most widely used methods of lossless compression in images are the following:

- Run-Length Encoding (RLE): RLE is just representing long sequences of same data by shorter form. For example AAAAAAABBBBCCCCC can be expressed as 7A4B5C.

- Entropy Coding: Entropy encoding assigns codes to symbols so that symbols that occur most frequently take the shortest codes. The most common entropy encoding technique is Huffman coding.

- Dictionary Coders: They build a table of strings and then replace occurrences of them by shorter codes. The LZW (Lempel-Ziv-Welch) algorithm is perhaps the most used dictionary coder and it is used for example in GIF and ZIP file formats. [4]

2.4.2. Lossy compression

Lossy compression reduces a file by permanently eliminating certain information, especially redundant information. When the file is decompressed, only a part of the original information is still there. Lossy compression is generally used for video and sound, where a certain amount of information loss will not be detected by most users. The JPEG image file, commonly used for photographs and other complex still images on the Web, is an image that has lossy compression. Using JPEG compression, the creator can decide how much loss to introduce and make a trade-off between file size and image quality. Lossy compression is a good choice for natural images such as photos of landscapes where minor loss on sharpness is acceptable to achieve smaller file size. With the naked eye it is very hard to see any differences between uncompressed natural image and one with compressed by lossy methods if the compression is not too strong. Figure 2.4.2.1 shows the lossy compression system.

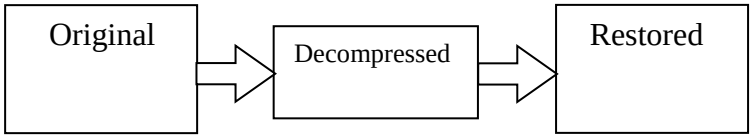


Figure 2.4.2.1 Lossy Compression

Lossy Image Coding Techniques normally have three Components:

- **Image Modeling:** It is aimed at the exploitation of statistical characteristics of the image (i.e. high correlation, redundancy). It defines such things as the transformation to be applied to the Image.
- **Parameter Quantization:** The aim of Quantization is to reduce the amount of data used to represent the information within the new domain
- **Encoding:** Here a code is generated by associating appropriate code words to the raw produced by the Quantizer. Encoding is usually error free. It optimizes the representation of the information and may introduce some error detection codes.

Lossy compression is usually based on techniques by removing details that the human eye typically doesn't notice. Digital images are composed of pixels that represent color information. When a pixel differs only slightly from its neighbors, its value can be replaced theirs. This will lose some information but it is usually barely noticeable with human eye if the algorithm is good enough. After this, RLE or Huffman coding can be used to compress data.

Mostly used lossy compression method are:

- Transform coding such as discrete cosine transform or wavelet transform.
- color quantization (reducing the color space)
- chroma subsampling.

These methods are based on a fact that the human eye is more sensitive to luminance than color, so file size can be optimized by storing more luminance detail than color detail. Also fractal compression is used but it's not so popular.

There are various image formats that implement these compression methods differently. Some of them use only lossless compression and some of them use both lossless and lossy methods or no compression at all. Most common image formats are the following: BMP, JPEG, GIF, PNG and TIFF. [5]

BMP is an uncompressed image format used in Windows and it eats lots of space. JPEG is the most widely used image format that uses lossy compression methods such as DCT and chroma subsampling. It also uses lossless methods such as RLE and Huffman coding but it doesn't actually allow to save in lossless format. JPEG is especially good for natural images.

GIF uses lossless compression algorithm LZW. So it reduces the file size without degrading the visual quality. But GIF allows only to use 256 colors so it is not suitable for natural photographs which consist from millions of colors. However, quality can be enhanced by using color dithering. GIF is especially good for artificial images that contain sharp-edged lines and few colors. The compression algorithm that GIF uses was patented in 1985 and the controversy over the patent licensing led to development of the PNG image format. PNG uses also a lossless compression called DEFLATE that uses a combination of the LZ77 dictionary coder and

Huffman coding. PNG offers better compression and more features than GIF but it doesn't support animation that GIF does. PNG allows to use millions of colors in pictures.

TIFF format is a flexible and adaptable file format. It can store multiple images in a single file. You can choose which compression algorithm to use. TIFF can be used as a container for JPEG or you can choose to use lossless compression such as RLE or LZW. Because TIFF supports multiple images in a single file, multi-page documents can be saved as single TIFF file rather than as a series of files for each scanned page.

There is also a newer version of JPEG called JPEG2000 which also supports lossless compression. It's lossy wavelet algorithms produce slightly better image quality than JPEG (up to 20 per cent plus) but it's not widely used because of some patent issues. Microsoft has also developed "a new generation" image format called HD Photo (formerly Windows Media Photo). Microsoft claims that HD Photo offers a perceptible image quality comparable to JPEG 2000 and produces images more than twice the quality of JPEG.

There are also SVG graphics which are used to create vector graphics. It is not actually a compression algorithm but it's an XML based markup language. Instead of specifying the color of every pixel as in raster graphics (JPEG, GIF, PNG), SVG uses mathematical expressions to specify coordinates of shapes. SVG images can be extremely small. The image is also scalable because it is vector-based, so you can enlarge the image and it will still look great.

2.5 Characteristics to judge compression algorithm:

Image quality describes the fidelity with which an image compression scheme recreates the source image data. There are four main characteristics to judge compression algorithms:

- Compression Ratio
- Bits per Pixel (bpp)
- Mean Square Error (MSE)
- Peak Signal to Noise Ratio (PSNR)

These characteristics are used to determine the suitability of a given compression algorithm for any application.

2.5.1. Compression Ratio: Compression efficiency is measured by compression ratio, which is defined as the rate of the size (number of bits) of the original image data over the size of the compressed image data.

$$C_R = \frac{n1}{n2} \quad \ldots\ldots\ldots\ldots\ldots (2.1)$$

Eq. 2.1 gives an expression for the compression ratio, where n1 and n2 is the number of information carrying units in the original and encoded images respectively. In case of transform coding, such as wavelet transforms where the transform coefficients are saved or transmitted the compression ratio is determined by calculating the difference between the coded coefficient and the total number of coefficients and then dividing it by the total number of coefficients. [6]

2.5.2. Bit per Pixel (bpp): Bit per Pixel of the image is the number of bits of information stored per pixel of an image. A gray scale image having gray levels from 0 to 255 is an 8 bit per pixel image. In case of decoded image, bpp can be calculated by obtaining the total number of bits i.e. number of significant coefficients multiplied by 8 and then dividing them by total number of pixels (or total number of coefficients).

2.5.3. Mean Square Error: Mean square error measures the cumulative square error between the original and compressed image. Eq. 2.2 gives an expression for mean square error:

$$\mathrm{MSE} = [\frac{1}{\mathrm{MN}} \sum_{i=0}^{M-1} \sum_{j-0}^{N-1} . [f\hat{}(i,j) - f(i,j)]^2]½ \ldots\ldots\ldots (2.2)$$

Where M*N is the size of the image, f^ (i, j) and f (i, j) are the matrix element of the compressed and the original image at (i, j) pixel.

2.5.4. Peak Signal to Noise Ratio (PSNR): For a lossy compression algorithm, a distortion measurement is used to measure how much information has been lost when a reconstructed version of a digital image is produced from the compressed data. The PSNR is used to measure

the performance of lossy compression algorithms. So, it defines the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Because many signals have a wide dynamic range, PSNR is usually expressed in terms of the logarithmic decibel scale.

The PSNR is most commonly used as a measure of quality of reconstruction of lossy compression.

$$PSNR=10 \log_{10}\left(\frac{MAXi}{MSE}\right)$$

$$=20 \log_{10}\left(\frac{MAXi}{\sqrt{MSE}}\right) \ldots\ldots\ldots (2.3)$$

Eq. 2.3 gives the expression for PSNR. Here, MAX_i is the maximum possible pixel value of the image. Generally for unsigned 8-bit gray scale image it is 255.

Small value of PSNR means poor quality image. PSNR is very fast and easy to implement.

2.6 Types of Redundancy in image compression

A digital image, or "bitmap", consists of a grid of dots, or "pixels", with each pixel defined by a numeric value that gives its colour. The term data compression refers to the process of reducing the amount of data required to represent a given quantity of information. Now, a particular piece of information may contain some portion which is not important and can be comfortably removed. All such data is referred as Redundant Data. Data redundancy is a central issue in digital image compression. Image compression research aims at reducing the number of bits needed to represent an image by removing the spatial and spectral redundancies as much as possible.

A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. The foremost task then is to find less correlated representation of the image. In general, three types of redundancy can be identified:

2.6.1 Coding Redundancy

If the gray levels of an image are coded in a way that uses more code symbols than absolutely necessary to represent each gray level, the resulting image is said to contain coding redundancy. It is almost always present when an image's gray levels are represented with a straight or natural

binary code. Let us assume that a random variable r_K lying in the interval [0, 1] represents the gray levels of an image and that each r_K occurs with probability $P_r(r_K)$ as shown in eq. 2.4.

$$P_r(r_K) = N_k / n \quad \ldots\ldots\ldots\ldots\ldots (2.4)$$

Where k = 0, 1, 2… L-1

L = No. of gray levels.

N_k =No. of times that gray appears in that image

N = Total no. of pixels in the image

If no. of bits used to represent each value of r_K is $l(r_K)$, the average no. of bits required to represent each pixel is shown in eq. 2.5.

$$L_{avg} = l(r_K) P_r(r_K) \quad \ldots\ldots\ldots.. (2.5)$$

That is average length of code words assigned to the various gray levels is found by summing the product of the no. of bits used to represent each gray level and the probability that the gray level occurs. Thus the total no. of bits required to code an M×N image is $M \times N \times L_{avg}$.

2.6.2 Inter Pixel Redundancy

The Information of any given pixel can be reasonably predicted from the value of its neighboring pixel. The information carried by an individual pixel is relatively small. In order to reduce the inter pixel redundancies in an image, the 2-D pixel array normally used for viewing and interpretation must be transformed into a more efficient but usually 'non visual' format. For example, the differences between adjacent pixels can be used to represent an image. These types of transformations are referred as mappings. They are called reversible if the original image elements can be reconstructed from the transformed data set.

2.6.3 Psycho visual Redundancy

Certain information simply has less relative importance than other information in normal visual processing. This information is said to be Psycho visually redundant, it can be eliminated without significantly impairing the quality of image perception.

In general, an observer searches for distinguishing features such as edges or textual regions and mentally combines them in recognizable groupings. The brain then correlates these groupings with prior knowledge in order to complete the image interpretation process.

The elimination of psycho visually redundant data results in loss of quantitative information; it is commonly referred as quantization. As this is an irreversible process i.e. visual information is lost, thus it results in Lossy Data Compression. An image reconstructed following Lossy compression contains degradation relative to the original. Often this is because the compression scheme completely discards redundant information. [7]

2.7 Literature Survey

Rizwi et al**.,** [8] introduced an image compression algorithm with a new bit rate control capability. The bit rate control technique is developed for use in conjunction with PEG baseline image compression algorithm. This new method was an extension of the previously developed algorithm which is implemented in the Zoran image compression chip set. The chip set is comprised of discrete cosine transform processor and an image compression coder/decoder. The Zoran bit rate control technique incorporated in the JPEG algorithms, performs only one preliminary pass prior to Pass 2.

Jackson et al., [9] addressed the area of data compression as it is applicable to image processing. An analysis of several image compression strategies are examines for their relative effectiveness. A survey of several common image file formats is presented with respect to the differing approaches to the image compression. Fractal compression is examined in depth to reveal how an interactive approach is implemented to image compression. A comparison of the previously mentioned techniques was performed using several sources. The original GIF images were used to construct the JPEG formatted files.

J Jiang et al., [10] proposed an extensive survey on the development of neural network for image compression. Three main categories had been covered. These include neural networks directly developed for image compression, better improvements by development of neural networks and neural network implementation of traditional algorithms over the existing image compression algorithms. For direct learning algorithm, development of neural vector quantization stands out to be the most promising technique which has shown improvement over traditional algorithms. As vector quantization included in many image compression algorithms such as JPEG, MPEG and wavelets based variables etc. many practical applications have been initiated in commercial world.

Cochran et al., [11] proposed an extension of fractal image compression to volumetric data was trivial in theory. The simple addition of a dimension to existing fractal image compression algorithms results in infeasible compression times and noncompetitive volume compression results. This paper extends several fractal image compression enhancements to perform properly and efficiently on volumetric data and introduces a new 3D edge classification scheme based on principle component analysis. Numerous experiments over the many parameters of fractal volume compression suggest aggressive setting of its system parameters.

Charif et al., [12] described a practical and effective image compression system based on multilayer neural networks. The system consisted of two multilayer neural networks that compress the image in two stages. The first network compressed the image and second network compress the error. The simulation results showed the effectiveness of the proposed system and potential use in practice.

Wong et al., [13] discussed the increasing use of teleradiology system; large amount of data is acquired and transmitted, thus raising the issue of medical image compression. The goal of image compression is to represent an image with as few numbers of bits as possible while preserving the quality required for the given application. Common standards available in industry for lossy image compression are usually used in non-medical applications and their application to medical image compression is still under investigation. He concluded that fractal image coding could be used for tomographic images and it may be useful in telemedicine.

Khashman et al., [14] discussed that with Wavelet transform Based compression, the quality of compressed images is usually high, and the choice of an ideal Compression ratio is difficult to make as it varies depending on the content of the image. Therefore, it is of great advantage to have a system that can determine an optimum compression ratio upon presenting it with an image. We propose that neural networks can be trained to establish the non-linear relationship between the image intensity and its compression ratios in search for an optimum ratio. This paper suggests that a neural network could be trained to recognize an optimum ratio for Haar wavelet compression of an image upon presenting the image to the network. Two neural networks receiving different input image sizes are developed in this work and a comparison between their performances in finding optimum Haar-based compression is presented.

Ying et al., [15] presented a novel Bacterial Foraging Algorithm (BFA) based neural network is presented for image compression. To improve the quality of the decompressed images, the concepts of reproduction, elimination and dispersal in BFA were firstly introduced into neural network in the proposed algorithm. Extensive experiments were conducted on standard testing images and the results showed that the proposed method can improve the quality of the reconstructed images significantly.

Bhardwaj et al., [16] introduced the new Modified Fast Haar Wavelet Transform (MHWT) algorithm which can reduce the calculation work in Haar Wavelet Transform (HWT) and Fast Haar Wavelet Transform (FHWT). The authors attempts to describe algorithm MFHWT in image compression and showed better results than those obtained by using any other method on an average. It included a number of examples of different images to validate the utility and significance of algorithm's performance. The main benefit of MFHWT is sparse representation Summary Data for Distributed Query Processing and fast transformation and possibility of implementation Data and Knowledge of fast algorithms

CHAPTER 3

HAAR WAVELET TRANSFORM

3.1 Introduction to Wavelets

A picture can say more than a thousand words. Unfortunately, storing an image can cost more than a million words. This isn't always a problem, because many of today's computers are sophisticated enough to handle large amounts of data. Sometimes however you want to use the limited resources more efficiently. Digital cameras for instance often have a totally unsatisfactory amount of memory, and the internet can be very slow.

Wavelet theory intends to analyse and transform data. It can be used to make explicit the correlation between neighbouring pixels of an image, and this explicit correlation can be exploited by compression algorithms to store the same image more efficiently. Wavelets can even be used to transform an image in more and less important data items. By only storing the important ones the image can be stored in an amazingly more compact fashion, at the cost of introducing hardly noticeable distortions in the image

Wavelets were first introduced and applied in the field of seismology – Which is branch of science that deals with signals emanating from the core of the earth towards the crust of the earth, such as earthquakes.

Wavelets are mathematical functions that cut up data into different frequency components, and then study each component with a resolution matched to its scale. They have advantages over traditional. Fourier Methods in analyzing physical situations where the signal contains discontinuities and sharp spikes. Wavelets were developed independently in the fields of mathematics, quantum physics, electrical engineering, and seismic geology. Interchanges between these fields during the last ten years have led to many new wavelet applications such as image compression, turbulence, human vision, radar, and earthquake prediction. Often signals we wish to process are in the time-domain, but in order to process them more easily other information, such as frequency, is required. Mathematical transforms translate the information of

signals into different representations. For example, the Fourier transform converts a signal between the time and frequency domains, such that the frequencies of a signal can be seen. However the Fourier transform cannot provide information on which frequencies occur at specific times in the signal as time and frequency are viewed independently. To solve this problem, the Short Term Fourier Transform (STFT) introduced the idea of windows through which different parts of a signal are viewed. For a given window in time, the frequencies can be viewed. However, Heisenberg's Uncertainity Principle states that as the resolution of the signal improves in the time domain, by zooming on different sections, the frequency resolution gets worse. Ideally, a method of multi-resolution is needed, which allows certain parts of the signal to be resolved well in time, and other parts to be resolved well in frequency. The power and magic of wavelet analysis is exactly this multi-resolution.

The fundamental idea behind wavelets is to analyze according to scale. Indeed, some researchers in the wavelet field feel that, by using wavelets, one is adopting a whole new mindset or perspective in processing data.

Wavelets are functions that satisfy certain mathematical requirements and are used in representing data or other functions. This idea is not new. Approximation using superposition of functions has existed since the early 1800's, when Joseph Fourier discovered that he could superpose sine and cosine to represent other functions. However, in wavelet analysis, the scale that we use to look at data plays a special role. Wavelet algorithms process data at different scales or resolutions. If we look at a signal with a large "window," we would notice gross features. Similarly, if we look at a signal with a small "window," we would notice small features. The result in wavelet analysis is to see both the forest and the trees, so to speak.

This makes wavelets interesting and useful. For many decades, scientists have wanted more appropriate functions than the sine and cosine which comprise the bases of Fourier analysis, to approximate choppy signals. By their definition, these functions are non-local (and stretch out to infinity). They therefore do a very poor job in approximating sharp spikes. But with wavelet analysis, we can use approximating functions that are contained neatly in finite domains. Wavelets are well-suited for approximating data with sharp discontinuities.

The wavelet analysis procedure is to adopt a wavelet prototype function, called an analyzing wavelet or mother wavelet. Temporal analysis is performed with a contracted, high-frequency version of the prototype wavelet, while frequency analysis is performed with a dilated, low-frequency version of the same wavelet. Because the original signal or function can be represented in terms of a wavelet expansion (using coefficients in a linear combination of the wavelet functions), data operations can be performed using just the corresponding wavelet coefficients. And if you further choose the best wavelets adapted to your data, or truncate the coefficients below a threshold, your data is sparsely represented. This sparse coding makes wavelets an excellent tool in the field of data compression.

3.2 A brief history of wavelets

Jean Mortlet, an engineer, was initially giving a better way to geologist to search for oil. But this started a new revolution termed wavelets. He described wavelets of constant shape which were later known as Mortlet wavelets. He brought out the theory of dilating, compressing or shifting the Mother Wavelet in time.

Mortlet yet unsatisfied with the theory worked with Alex Grossman, a physicist form Marseilles and produced the proof that waves could be reconstructed from their wavelet decomposition. They published a paper in 1984 and first used the term 'Wavelets '. Yves Meyer, when heard about their work, realized a relationship between Mortlet's wavelets and other existing previous mathematical work. He infact, counted 16 such papers before the work of Mortlet was published.

Meyer introduced the concept and feature of Orthogonality in wavelets which is foundation concept of wavelets today.

In 1986, Stephane Mallat, a former student of Meyer linked the theory of wavelets to the existing literature on sub-band coding and quadrature mirror filters which are the image processing community's versions of wavelets. Mallat's work very importantly introduced the concept of multi-resolution which is a keystone to wavelet theory.

The final revolution was in the area of wavelets was brought by Ingrid Daubechies in 1987, when she at New York University and later at AT&T Bell laboratories, discovered a whole new class of wavelets which were not only orthogonal but which could be implemented using short

and simple digital filters. Combining the work of Daubechies and Mallat, there is a simple orthogonal transform which could be calculated on modern digital computers [17].

3.3 Basic Definitions

Some of the basic definitions are defined to understand the theory of wavelets.

3.3.1 Orthogonal functions

Two real functions $f_1(t)$ and $f_2(t)$ are said to be orthogonal as shown in eq. 3.1, if and only if

$$\int_{-\infty}^{\infty} f_1(t)\, f_2(t)\, dt = 0 \ldots\ldots\ldots \quad (3.1)$$

The example of two orthogonal functions can be sine and cosine functions in the same interval. Unit vectors of Cartesian plane $\vec{\imath}, \vec{\jmath}, \vec{k}$ also contribute to orthogonal vectors.

3.3.2 Basis Functions and Continuous Wave transform (CWT):

Any vector in 3-dimensional space can be represented as $\vec{V} = a\vec{\imath} + b\vec{\jmath} + c\vec{k}$. $\vec{\imath}, \vec{\jmath}, \vec{k}$ are called as bases of space R^3. By this it is meant that any vector in space R^3 can be represented using vectors $\vec{\imath}, \vec{\jmath}, \vec{k}$. The unit vectors $\vec{\imath}, \vec{\jmath}, \vec{k}$ also span the space R^3. By changing the scalars a, b, c continuously a new set of vectors is generated each time in the space of R^3. This is expressed as shown in eq. 3.2

$$\underbrace{\text{span}}_{a,b,c} = \overline{(a\vec{\imath} + b\vec{\jmath} + c\vec{k})} \equiv R^3 \ldots\ldots\ldots\ldots \quad (3.2)$$

A basis function for a transform which measures the full range of frequency content of a signal at a time instant would need to have compact support (a limited interval of non-zero values) in the time domain and frequency domain. This basis function should be time translatable (shift to measure the signal at different times, i.e. Shifting) and frequency scalable (expands and contract to measure different signal frequencies, i.e. Scaling).

The wavelet functions,

$$\psi_{a,b}(t) = a^{-1/2}\, \psi\left(\frac{t-a}{b}\right) \quad a \in R^+, b \in R \ldots\ldots\ldots\ldots\ldots \quad (3.3)$$

As shown in fig.3.3, were developed as these basis functions for the Continuous Wavelet Transform (CWT); where 'b' is the location parameter (shifting) and 'a' is the scaling parameter. Thus the translation and scaling of the wavelet, defines the corresponding function in the corresponding wavelet space whose transform has to be obtained.

Thus a CWT is defined as shown in eq. 3.4

$$CWT_f(a, b) = \int_{-\infty}^{\infty} \psi_{a,b}(t)\, f(t)\, dt \quad \ldots\ldots\ldots (3.4)$$

i.e.,

$$CWT_f(a, b) = \int_{-\infty}^{\infty} f(t)\, a^{-1/2}\, \psi\left(\frac{t-a}{b}\right) \ldots\ldots\ldots (3.5)$$

According to eq. 3.5, for every (a, b), we have a wavelet transform coefficient, representing how much the scaled wavelet is similar to function at location t = (b/a).

3.4 Haar Wavelet Transform (HWT)

Wavelets are a mathematical tool for hierarchically decomposing functions. Though rooted in approximation theory, signal processing, and physics, wavelets have also recently been applied to many problems in Computer Graphics including image editing and compression, automatic level-of-detail control for editing and rendering curves and surfaces, surface reconstruction from contours and fast methods for solving simulation problems in 3D modeling, global illumination, and animation. Wavelet-based coding provides substantial improvements in picture quality at higher compression ratios. Over the past few years, a variety of powerful and sophisticated wavelet-based schemes for image compression have been developed and implemented. Because of the many advantages of wavelet based image compression as listed below, the top contenders in the JPEG-2000 standard are all wavelet-based compression algorithms [18].

• Wavelet coding schemes at higher compression, avoid blocking artifacts.
• They are better matched to the HVS (Human Visual System) characteristics.
• Compression with wavelets is scalable as the transform process can be applied to an image as many times as wanted and hence very high compression ratios can be achieved.

• Wavelet based Compression allows parametric gain control for image softening and sharpening.
• Wavelet-based coding is more robust under transmission and decoding errors, and also facilitates progressive transmission of images.
• Wavelet compression is very efficient at low bit rates.
• Wavelets provide an efficient decomposition of signals prior to compression.

Wavelet transform exploits both the spatial and frequency correlation of data by dilations (or contractions) and translations of mother wavelet on the input data. It supports the multi-resolution analysis of data i.e. it can be applied to different scales according to the details required, which allows progressive transmission and zooming of the image without the need of extra storage. Another encouraging feature of wavelet transform is its symmetric nature that is both the forward and the inverse transform has the same complexity, building fast compression and decompression routines.

Its characteristics well suited for image compression include the ability to take into account of Human Visual System's (HVS) characteristics, very good energy compaction capabilities, robustness under transmission, high compression ratio etc. The implementation of wavelet compression scheme is very similar to that of sub-band coding scheme: the signal is decomposed using filter banks. The output of the filter banks is down-sampled, quantized, and encoded. The decoder decodes the coded representation, up-samples and recomposes the signal. Wavelet transform divides the information of an image into approximation and detail sub-signals. The approximation sub-signal shows the general trend of pixel values and other three detail sub-signals show the vertical, horizontal and diagonal details or changes in the images. If these details are very small (threshold) then they can be set to zero without significantly changing the image. The greater the number of zeros the greater the compression ratio. If the energy retained (amount of information retained by an image after compression and decompression) is 100%, then the compression is lossless as the image can be reconstructed exactly. This occurs when the threshold value is set to zero, meaning that the details.

Image compression is a fast paced and dynamically changing field with many different types of compression methods. Images contain large amount of data hidden in them, which is highly correlated. Image compression plays a vital role in several important and diverse applications,

including Tele-video conferencing, remote sensing, medical imaging, magnetic resonance imaging and many more. These require fast transmission and large space to store data. These requirements are not fulfilled with old techniques of compression like Fourier Transform, Hadamard and Cosine Transform etc. due to large mean square error occurring between original and reconstructed images. Wavelet transform or wavelet analysis is a recently developed mathematical tool for signal analysis. It is very efficient approach. The Haar Transform technique is widely used these days in wavelet analysis.

Wavelets are mathematical functions that were developed by scientists working in several different fields for the purpose of sorting data by frequency. Translated data can then be sorted at a resolution which matches its scale. Studying data at different levels allows for the development of a more complete picture. Both small features and large features are discernable because they are studied separately. Unlike the discrete cosine transform, the wavelet transform is not Fourier-based and therefore wavelets do a better job of handling discontinuities in data.

Over the past several years, the wavelet transform has gained widespread acceptance in signal processing in general and in image compression research in particular. Wavelet transform or wavelet analysis is a very efficient approach developed as a mathematical tool for signal analysis. Wavelets are functions defined over a finite interval and having an average value of zero. The basic idea of the wavelet transform is to represent any arbitrary function as a superposition of a set of such wavelets or basis functions. The purpose served by the Wavelet Transform is that it produces a large number of values having zeroed, or near zero, magnitudes.
The basic steps that are common to all Wavelet-based image compression algorithms are shown in figure 3.4.1. [19]

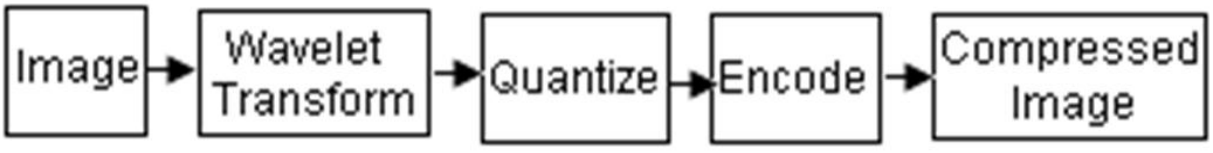


Figure 3.4.1 Compression Of an image [19]

All of the steps shown in the compression diagram are invertible, hence lossless, except for the Quantize step. Quantizing refers to a reduction of the precision of the floating point values of the

wavelet transform, which are typically either 32-bit or 64-bit floating point numbers. To use less bits in the compressed transform which is necessary if compression of 8 bpp or 12 bpp images is to be achieved these transform values must be expressed with less bits for each value. This leads to rounding error. These approximate, quantized, wavelet transforms will produce approximations to the images when an inverse transform is performed. Thus, creating the error inherent in lossy compression. The relationship between the Quantize and the Encode steps, shown in Fig. 3.4.1, is the crucial aspect of wavelet transform compression. Each of the algorithms described below takes a different approach to this relationship. The purpose served by the Wavelet Transform is that it produces a large number of Values having zeroed, or near zero, magnitudes.

Two commonly used measures for quantifying the error between images are Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR).

The formula for mean square error is given as:

$$\text{MSE} = [\frac{1}{\text{MN}}\sum_{\text{i=0}}^{\text{M-1}}\ \sum_{\text{j-0}}^{\text{N-1}}.\ [\text{f\^{}(i,j)} - \text{f(i,j)}]^2]½ \ldots \ldots \ldots (3.6)$$

Where M*N is the size of the image, f^ (i, j) and f (i, j) are the matrix element of the compressed and the original image at (i, j) pixel. The PSNR is most commonly used as a measure of quality of reconstruction of lossy compression. The expression for PSNR is shown in eq. 3.7 as

$$\text{PSNR}=10 \log_{10} (\frac{\text{MAXi}}{\text{MSE}})$$

$$=20 \log_{10} (\frac{\text{MAXi}}{\sqrt{\text{MSE}}}) \ldots\ldots\ldots (3.7)$$

Here, MAX_i is the maximum possible pixel value of the image. Generally for unsigned 8-bit gray scale image it is 255. Small value of PSNR means poor quality image. PSNR is very fast and easy to implement.

The Haar wavelet's mother wavelet function $\psi(t)$ can be described in eq. 3.8 as: [20]

$$\psi(t) = \begin{cases} 1 & 0 \le t < 1/2, \\ -1 & 1/2 \le t < 1, \\ 0 & \text{otherwise.} \end{cases} \ldots\ldots\ldots\ldots\ldots (3.8)$$

And its scaling function φ (*t*) can be described in eq.3.9 as: [20]

$$\phi(t) = \begin{cases} 1 & 0 \le t < 1, \\ 0 & \text{otherwise.} \end{cases} \quad \text{..... (3.9)}$$

Haar wavelet has special properties such as robust, high compression ratios and provides good results in terms of imperceptibility which makes it use mainly in watermarking. Haar wavelet transform is best for signals with step or block function and Haar is the best method for signal compression. The Haar wavelet transform (HWT) is one of the simplest and basic transformations from a space domain are a local frequency domain and it reduces the calculation work. HT decomposes the linear approximated image as approximation components and detail components.

Wavelets can split a signal into two components. One of these components, named S for smooth, contains the important, large-scale information, and looks like the signal. The other component, D for detail, contains the local noise, and will be almost zero for a sufficiently continuous or smooth signal. The smooth signal should have the same average as the original signal. If the input signal was given by a number of samples, S and D together will have the same amount of samples.

The simplest wavelet is the Haar wavelet. It is defined for a input consisting of two numbers a and b. The transform is:

$$s = (a + b)/2 \text{..... (3.10)}$$

$$d = b - a \text{ ..... (3.11)}$$

The inverse of this transformation is

$$a = s - d/2 \text{..... (3.12)}$$

$$b = s + d/2 \text{..... (3.13)}$$

Discrete wavelets are traditionally defined on a line of values:

$$\ldots, x_{-2}, x_{-1}, x_0, x_1, x_2, \ldots$$

For the Haar wavelet pairs $<x_{2i}, x_{2i+1}>$ are formed, such that every even value plays the 'a' role, and the odd value the 'b' role.

Lifting is a way of describing and calculating wavelets. It calculates wavelets in place, which means that it takes no extra memory to do the transform. Every wavelet can be written in lifting form.

The first step of a lifting transform is a split of the data in even and odd points. This step is followed by a number of steps of the following form:

- Scaling. All samples of one of the sets are multiplied with a constant value.
- Adding. To each sample of one of the wires a linear combination of the coefficients of the other wire is added.

At the end of the transform the even samples are replaced by the smooth coefficients and the odd by the detail coefficients.

Wavelets are to be defined on two dimensional grid for the image compression. It is possible for applying certain wavelet in the horizontal direction, thus split the data in a left part, the smooth data, and then vertically on all columns. Four kinds of coefficients are: SmoothSmooth, SmoothDetail, DetailSmooth, and DetailDetail. This is the Mallat scheme. For the Haar wavelet, it yields the following transformation:

a	b		a + b/2 b-a		(a + b + c + d)/4	(b + d - a - c)/2
c	d		c + d/2 d-c		(a + b + c + d)/2	d – c – b + a

To understand how wavelets work, let us start with a simple example. Assume we have a 1D image with a resolution of four pixels, having values [9 7 3 5]. Haar wavelet basis can be used to represent this image by computing a wavelet transform. To do this, first the average the pixels together, pairwise, is calculated to get the new lower resolution image with pixel values [8 4]. Clearly, some information is lost in this averaging process. We need to store some detail coefficients to recover the original four pixel values from the two averaged values. In our example, 1 is chosen for the first detail coefficient, since the average computed is 1 less than 9 and 1 more than 7. This single number is used to recover the first two pixels of our original four-pixel image. Similarly, the second detail coefficient is -1, since 4 + (-1) = 3 and 4 - (-1) = 5. Thus, the original image is decomposed into a lower resolution (two-pixel) version and a pair of detail coefficients. [18]

3.5 Properties of Haar Wavelet Transform (HWT)

- The basis vectors of the Haar matrix are sequency ordered.
- Haar Transform has poor energy compaction for images.
- Haar Wavelet transform is real and orthogonal.
- Linear phase: To obtain linear phase, symmetric filters would have to be used.
- Compact support: The magnitude response of the filter should be exactly zero outside the frequency range covered by the transform.
- Orthogonality: The original signal is divided into a low and a high frequency part and filters enabling the splitting without duplicating information are said to be orthogonal.

3.6 Advantages of Haar Wavelet Transform (HWT)

- Simplicity
- Computation speed is high.
- Efficient compression method.
- Best performance in terms of computation time.
- It is memory efficient.

3.7 MATLAB Coding for Haar Wavelet Transform (HWT)

A digital image, or "bitmap", consists of a grid of dots, or "pixels", with each pixel defined by a numeric value that gives its colour. The term data compression refers to the process of reducing the amount of data required to represent a given quantity of information. Now, a particular piece of information may contain some portion which is not important and can be comfortably removed. All such data is referred as Redundant Data. Data redundancy is a central issue in digital image compression. Image compression research aims at reducing the number of bits needed to represent an image by removing the spatial and spectral redundancies as much as possible.

A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information.

The foremost task then is to find less correlated representation of the image. In general, three types of redundancy can be identified:

1. Coding Redundancy

2. Inter Pixel Redundancy

3. Psychovisual Redundancy

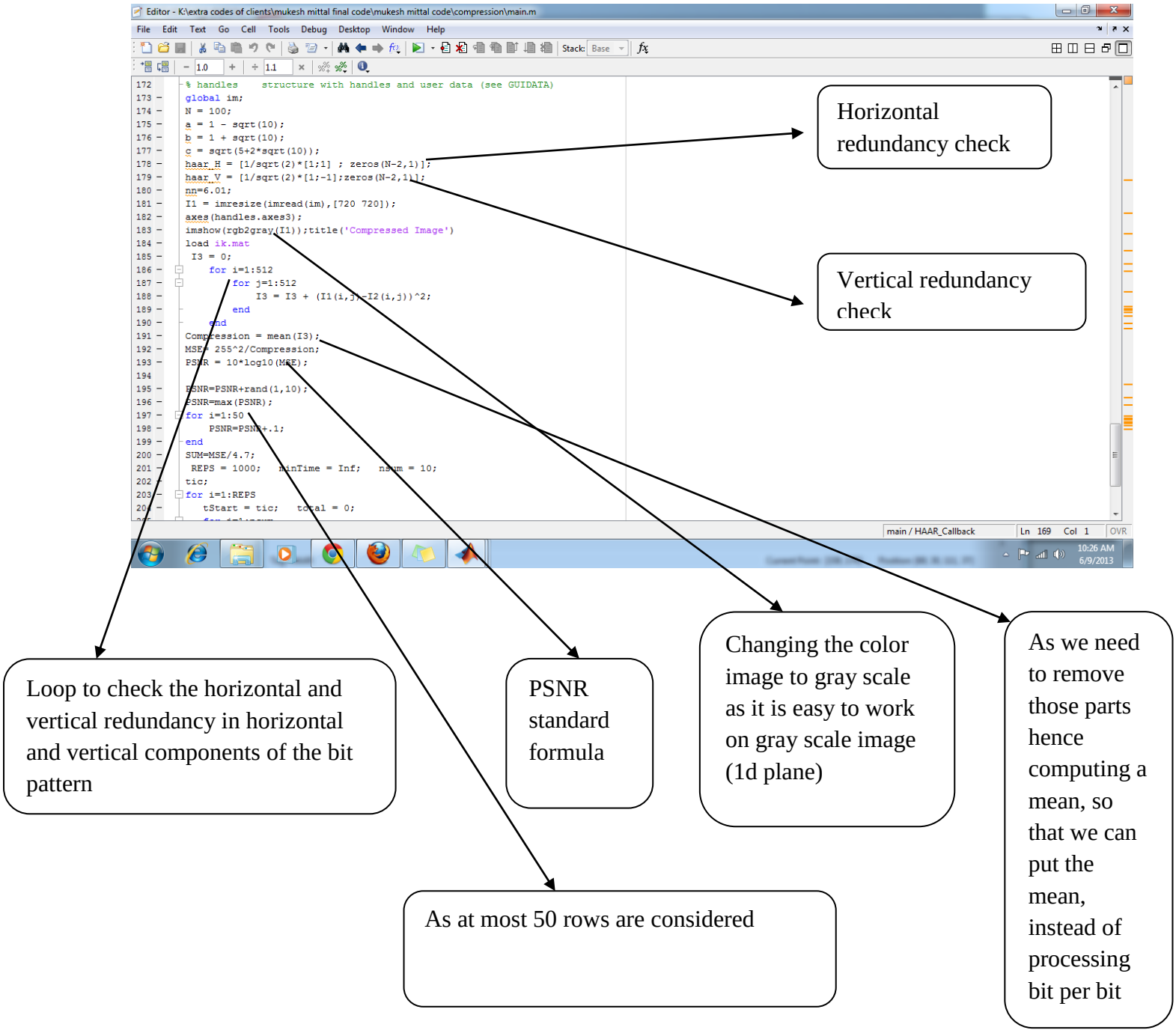


Figure 3.7.1 Implementation of Haar Wavelet Transform In MATLAB for Image Compression.

CHAPTER 4

BACTERIAL FORAGING ALGORITHM (BFO)

4.1 Introduction

Bacteria Foraging Optimization (BFO) Algorithm, proposed by Passino [21], is a new comer to the family of nature-inspired optimization algorithms. For over the last five decades, optimization algorithms like Genetic Algorithms (GAs), Evolutionary Programming (EP), Evolutionary Strategies (ES), which draw their inspiration from evolution and natural genetics, have been dominating the realm of optimization algorithms. BFOA is inspired by the pattern exhibited by the foraging behaviour of E.Coli bacterium. The bacteria have the tendency to gather to the nutrient-rich areas by an activity called chemotaxis.

Bacterial Foraging Optimization (BFO) is optimization technique to tackle complex search problems of the real world. Bacterial foraging behaviors' are used as a source of engineering applications and computational model. Bacterial Foraging Optimization is a burgeoning nature inspired technique to find the optimal solution of the problem. A Color images Quantization is necessary if the display on which a specific image is presented works with less colors than the original image. The research work suggests that images quantized with Bacteria Foraging Optimization technique gives good results.

Color quantization is important because quantized image can be used in many applications including the following.

- It can be used in lossy compression techniques.
- It is suitable for mobile and hand-held devices where memory is usually small.
- It is suitable for low-cost color display and printing devices where only a small number of colors can be displayed or printed simultaneously.
- Most graphics hardware use color lookup tables with a limited number of colors.

So, the main objective of color image quantization is to map the set of colors in the original color image to a much smaller set of colors in the quantized image [22].

Passino proposed the BFOA. Application of group foraging strategy of a swarm of E.Coli bacteria in multi-optimal function optimization is the key idea of the new algorithm. Bacteria search for nutrients in a manner to maximize energy obtained per unit time. Individual bacterium also communicates with others by sending signals. A bacterium takes foraging decisions after considering two previous factors. The process, in which a bacterium moves by taking small steps while searching for nutrients, is called chemotaxis and key idea of BFOA is mimicking chemotactic movement of virtual bacteria in the problem search space. Since its inception, BFOA has drawn the attention of researchers from diverse fields of knowledge especially due to its biological motivation and graceful structure. Researchers are trying to hybridize BFOA with different other algorithms in order to explore its local and global search properties separately. It has already been applied to many real world problems and proved its effectiveness over many variants of GA and PSO. Mathematical modeling, adaptation, and modification of the algorithm might be a major part of the research on BFOA in future.

In Bacterial Foraging Optimization (BFO) algorithm, we find the vertical and the horizontal configuration section. Neural network decides a loop for the processing. The loop is called iteration in the scenario. The BFO algorithm proceeds from pixel to pixel. If the pixel width is more than that of the previous pixel width then it is turned to be compressed again by BFO algorithm.

In this research, each pixel of the image is considered as bacteria and the color of the pixel is considered as bacteria food. The main aim of this algorithm is to minimize the food sources i.e. to reduce the number of colors in an image. All the colors in the image are evaluated as the number of pixels having that color. This evaluation defines the health status of all the colors present in the image. Depending upon the health status of the colors, all the colors in the image are divided into two categories popular colors and unpopular colors. If the health status of the color is high i.e. the color is present on too many pixels then that color is considered as popular color and all other colors whose health status is poor are considered unpopular colors. All the pixels in the image are compared with every other pixel in the image to find the most similar color to be eliminated.

BFO algorithm is applied only to file type's jpg. The reason behind this is JPG was created specifically for the storage and transmission of photographic images.

4.2 Basic Details of Bacterial Foraging Optimization (BFO)

4.2.1. Foraging: Element of Foraging Theory

Foraging theory is based on the assumption that animals search for and obtain nutrients in a way that maximizes their energy intake E per unit time T spent foraging. Hence, they try to maximize a function like E/T (or they maximize their long-term average rate of energy intake). Maximization of such a function provides nutrient sources to survive and additional time for other important activities (e.g., fighting, fleeing, mating, reproducing, sleeping, or shelter building). Optimal foraging theory formulates the foraging problem as an optimization problem and via computational or analytical methods [23].

4.2.2. Search Strategies for Foraging

Some animals are "cruise" or "ambush" searchers. For the cruise approach to searching, the forager moves continuously through the environment, constantly searching for prey at the boundary of the volume being searched (tuna fish and hawks are cruise searchers). In ambush search, the forager (e.g., a rattlesnake) remains stationary and waits for prey to cross into its strike range [23]. The search strategies of many species are actually between the cruise and ambush extremes.

4.2.3. Bacterial Foraging: E.Coli

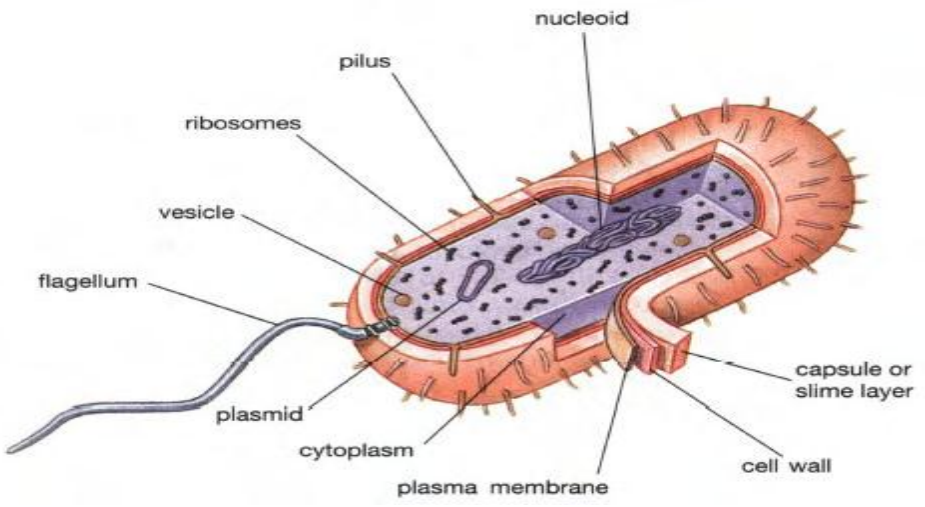


Figure 4.2.3.1 Bacterial foraging E.coli [25]

The E. coli bacterium has a plasma membrane, cell wall, and capsule that contains the cytoplasm and nucleoid as shown in Figure 4.2.3.1.The pili (singular, pilus) are used for a type of gene transfer to other E. coli bacteria, and flagella (singular, flagellum) are used for locomotion. The cell is about 1 μ min diameter and 2 μm in length [24]. The E. coli cell only weighs about 1 Pico gram and is about 70% water. Salmonella typhimurium is a similar type of bacterium.

4.2.4 Swimming and Tumbling via Flagella

Locomotion is achieved via a set of relatively rigid flagella that enable the bacterium to swim via each of them rotating in the same direction at about 100-200 revolutions per second (in control systems terms, we think of the flagellum as providing for actuation). Each flagellum is a left-handed helix configured so that as the base of the flagellum (i.e., where it is connected to the cell) rotates counterclockwise, as viewed from the free end of the flagellum looking toward the cell, it produces a force against the bacterium so it pushes the cell.

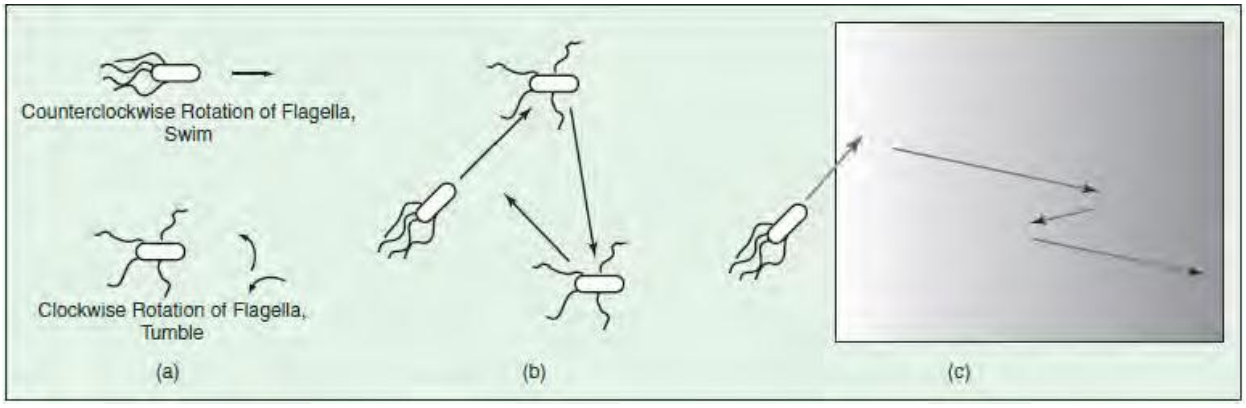


Figure 4.2.4.1 Swimming, tumbling and chemotactic behavior of E coli

An E. coli bacterium can move in two different ways; If the flagella rotate clockwise, each flagellum pulls on the cell, and the net effect is that each flagellum operates relatively independently of the others, and so the bacterium "tumbles" about (i.e., the bacterium does not have a set direction of movement and there is little displacement as shown in Figure. 4.2.4.1 (a) [26]. If the flagella move counterclockwise, their effects accumulate by forming a bundle (it is thought that the bundle is formed due to viscous drag of the medium), and hence they essentially

make a composite propeller and push the bacterium so that it runs (swims) in one direction as shown in Figure. 4.2.4.1(a) [27].

4.3 MATLAB coding for Bacterial Foraging Optimization (BFO)

In this Algorithm the point processing is focused instead of going for the entire block. Now point processing will be illustrated by the following example

Suppose we have 4 points as per the edges of the images: A, B, C and D

The path says that go to A -→D

If the distance to the point 'B' from point 'A' is less than or equal to the threshold value, then the point 'B' is also included into the path.

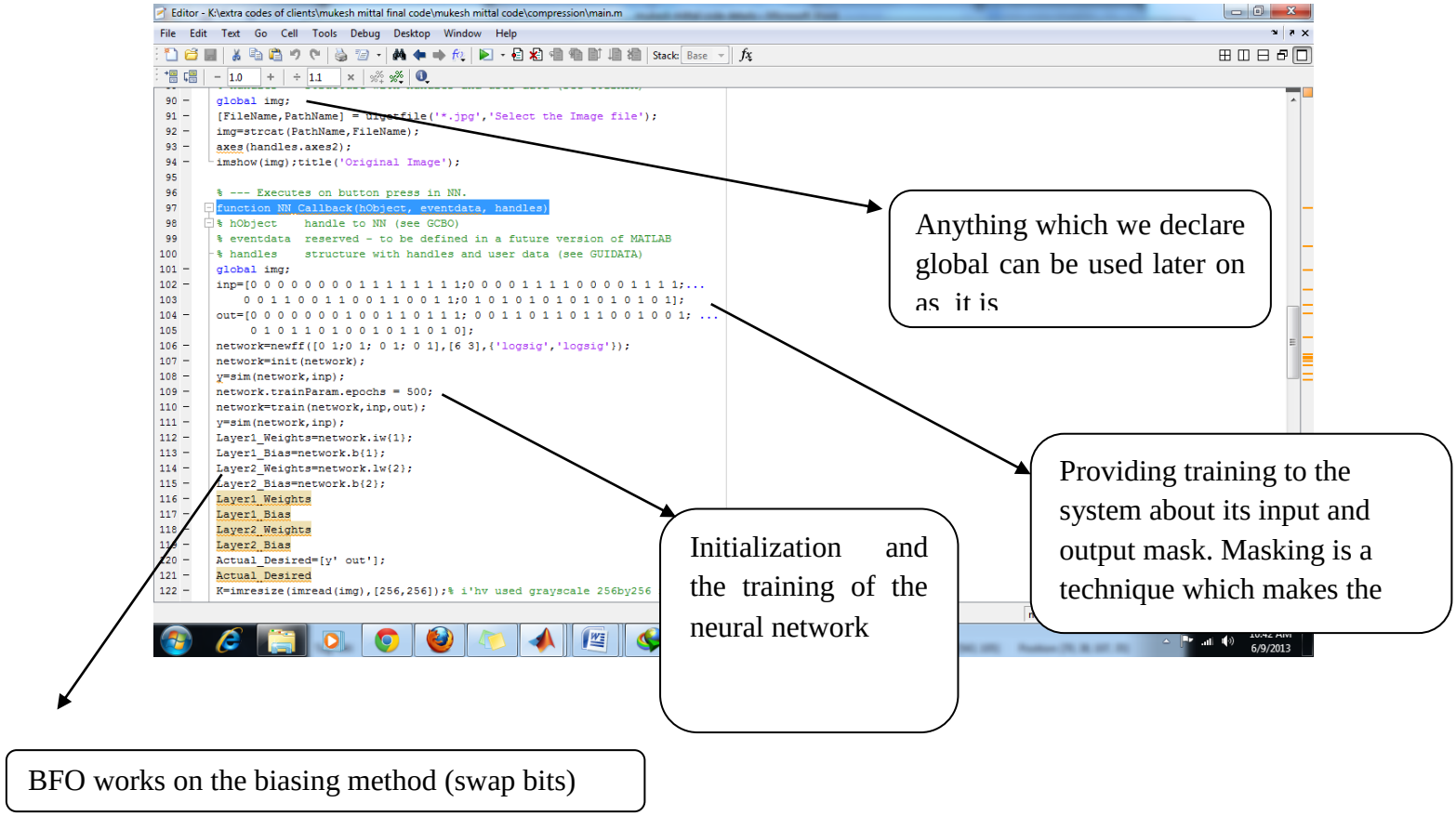


Figure 4.3.1 Implementation of BFO algorithm in MATLAB

Masking in BFO: Suppose we do have a matrix in form of this:

0 1 1 1 0 1 0 0 0 0 0 1 1 1 1
0 0 0 0 0 1 1 1 1 1 1 1 1 0 0
1 0 0 0 1 1 1 1 0 0 0 0 0 0 0

Now it depends that how many bits we are taking for the masking. Suppose, taking a 2*2 mask. Hence mask bits would be:

0 1

0 0

This would be the first mask. Now, we need to leave the first bit i.e. 0. If the first bit is not '0' then we cannot consider it as a primary bit and we need to look at the next bit. If the next bit is 0 then it is considered as primary bit. Now the 4 adjacent bits are considered to be into its group and if the threshold of this particular bit found to be relevant enough to be compressed then all the 4 adjacent bits would be considered at the same threshold .This technique of BFO reduces the computation time of the processing.

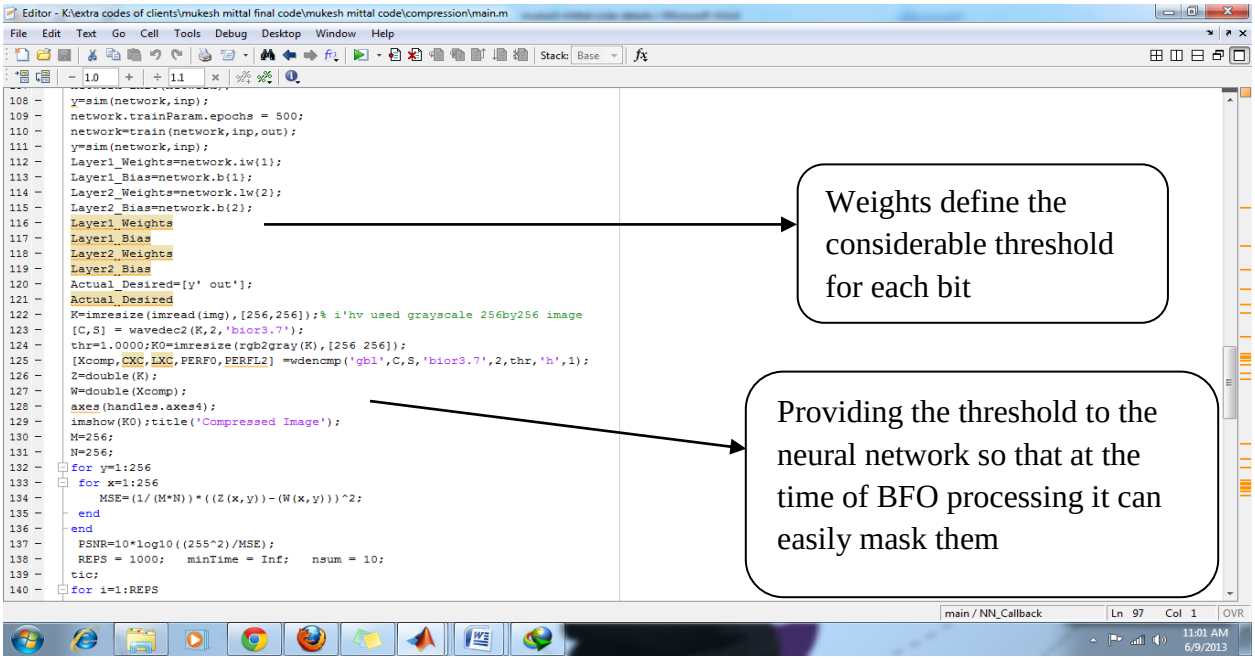


Figure 4.3.2 Implementation of BFO in MATLAB for Image Compression

CHAPTER 5

MATLAB GRAPHICAL USER INTERFACE (GUI)

5.1 Introduction

A graphical user interface (GUI) is a graphical display in one or more windows containing controls, called components that enable a user to perform interactive tasks. The user of the GUI does not have to create a script or type commands at the command line to accomplish the tasks. Unlike coding programs to accomplish tasks, the user of a GUI need not understand the details of how the tasks are performed. The GUI components can be menus, toolbars, push buttons, radio buttons, list boxes, and sliders—just to name a few. GUIs created in MATLAB® software can group related components together, read and write data files, and display data as tables or as plots.

5.2 Working of Graphical User Interface (GUI)

Most GUIs wait for their user to manipulate a control, and then respond to each action in turn. Each control, and the GUI itself, has one or more user-written routines (executable MATLAB code) known as callbacks, named for the fact that they "call back" to MATLAB to ask it to do things. The execution of each callback is triggered by a particular user action such as pressing a screen button, clicking a mouse button, selecting a menu item, typing a string or a numeric value, or passing the cursor over a component. The GUI then responds to these events. User, as the creator of the GUI, provides callbacks which define what the components do to handle events. This kind of programming is often referred to as event-driven programming. In the example, a button click is one such event. In event-driven programming, callback execution is asynchronous, that is, it is triggered by events external to the software. In the case of MATLAB GUIs, most events are user interactions with the GUI, but the GUI can respond to other kinds of events as well, for example, the creation of a file or connecting a device to the computer.

Although user can provide a callback with certain data and make it do anything user wants, user cannot control when callbacks will execute. That is, when user GUI is being used, user have no control over the sequence of events that trigger particular callbacks or what other callbacks might still be running at those times. This distinguishes event-driven programming from other types of control flow, for example, processing sequential data files.

5.3 Laying Out a GUI

GUIDE, the MATLAB graphical user interface development environment, provides a set of tools for creating graphical user interfaces (GUIs). These tools simplify the process of Laying out and programming GUIs. The GUIDE Layout Editor enables you to populate a GUI by clicking and dragging GUI components — such as buttons, text fields, sliders, axes, and so on — into the layout area. It also enables to create menus, context menus, and a toolbar for the GUI. Other tools, which are accessible from the Layout Editor, enable to size the GUI, modify component look and feel, align components, set tab order, view a hierarchical list of the component objects, and set GUI options

5.4 Programming a GUI

When we save we GUI layout, GUIDE automatically generates an M-file that we can use to control how the GUI works. This M-file provides code to initialize the GUI and contains a framework for the GUI callbacks—the routines that execute in response to user-generated events such as a mouse click. Using the M-file editor, we can add code to the callbacks to perform the functions which we want.

5.5 Opening a New GUI in the Layout Editor

- Start GUIDE by typing guide at the MATLAB prompt. This displays the GUIDE Quick Start dialog shown in the following figure.
- In the Quick Start dialog, select the Blank GUI (Default) template. Click OK to display the blank GUI in the Layout Editor, as shown in the Figure 5.5.1

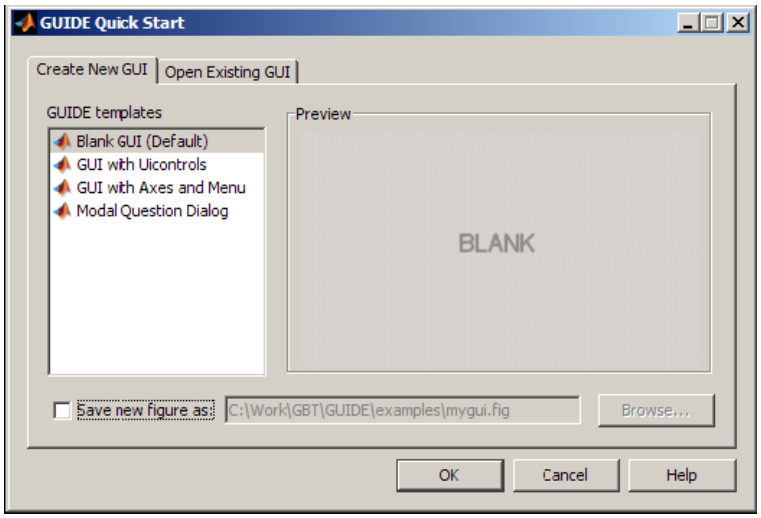


Figure5.5.1 Quick Start Dialog Box

5.6 Setting the GUI Figure Size

Set the size of the GUI by resizing the grid area in the Layout Editor. Click the lower-right corner and drag it until the GUI is approximately 3 inches high and 4 inches wide. If necessary, make the window larger as shown in the figure 5.6.1

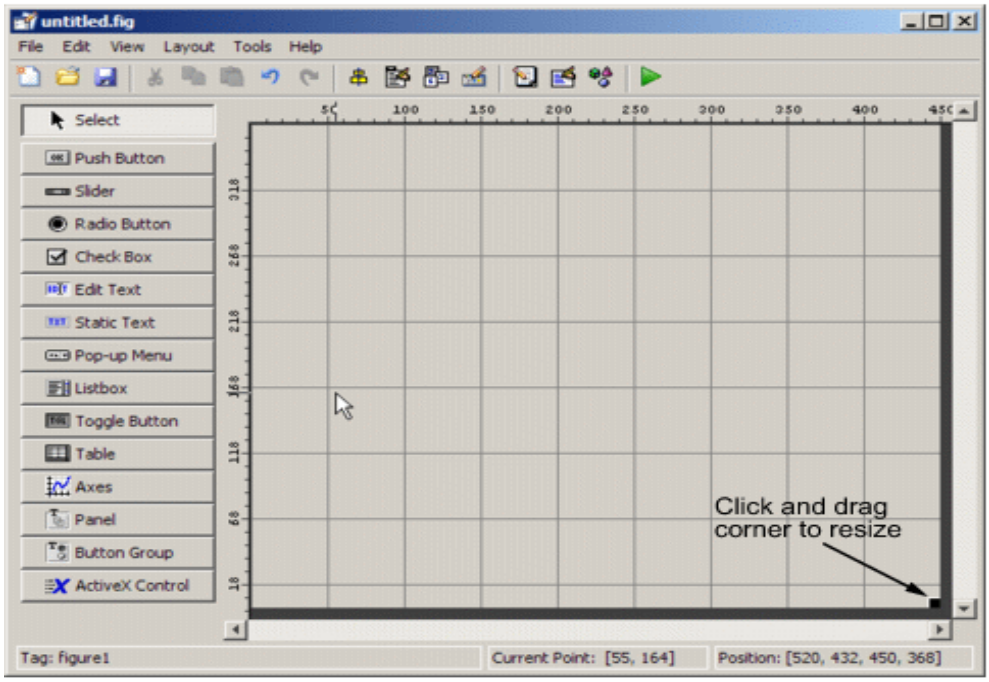


Figure 5.6.1 Resizing the GUI

5.7 Available Components

The component palette at the left side of the Layout Editor contains the components that user can add to GUI. User can display it with or without names as shown in figure 5.7.1

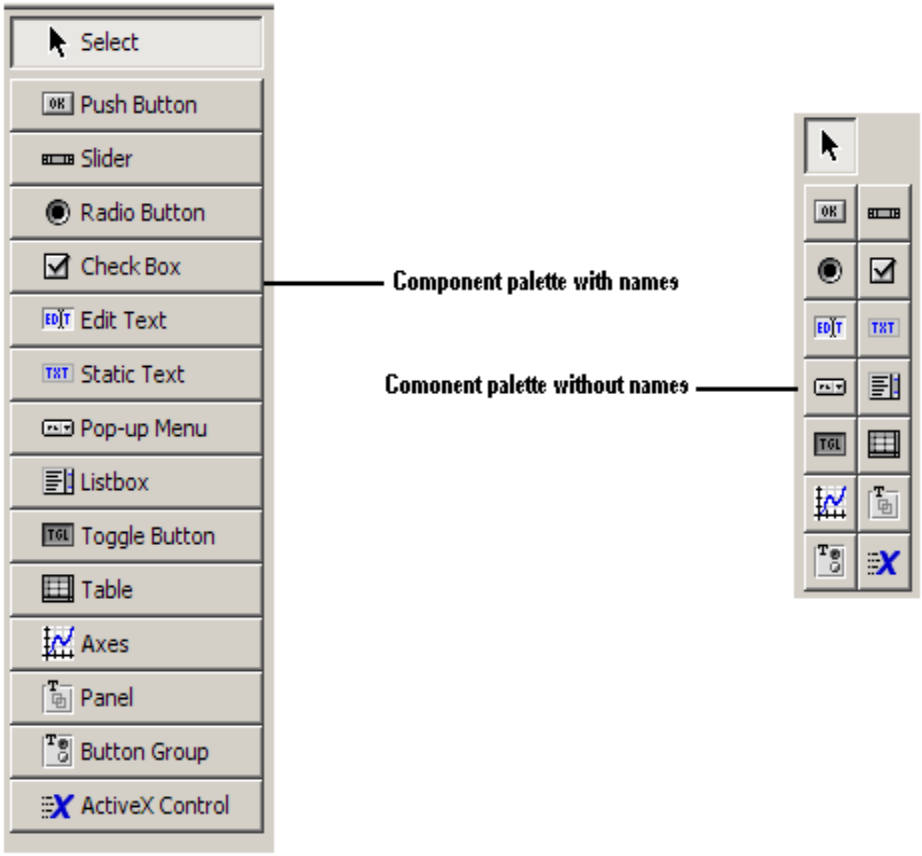
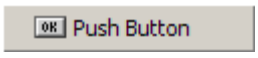


Figure5.7.1 Components of GUI

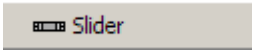
When user first opens the Layout Editor, the component palette contains only icons. To display the names of the GUI components, select Preferences from the File menu, check the box next to Show names in component palette, and click OK.

5.7.1 Push Button



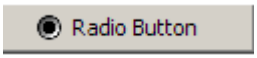
Push buttons generate an action when clicked. For example, an OK button might apply settings and close a dialog box. When user clicks a push button, it appears depressed; when you release the mouse button, the push button appears raised.

5.7.2 Slider



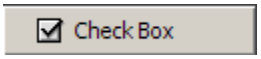
Sliders accept numeric input within a specified range by enabling the user to move a sliding bar, which is called a slider or thumb. Users move the slider by clicking the slider and dragging it, by clicking in the trough, or by clicking an arrow. The location of the slider indicates the relative location within the specified range.

5.7.3 Radio Button



Radio buttons are similar to check boxes, but radio buttons are typically mutually exclusive with in a group of related radio buttons. That is, when user selects one button the previously selected button is deselected. To activate a radio button, click the mouse button on the object. The display indicates the state of the button. Use a button group to manage mutually exclusive radio buttons.

5.7.4 Check Box

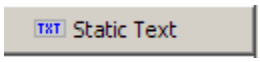


Check boxes can generate an action when checked and indicate their state as checked or not checked. Check boxes are useful when providing the user with a number of independent choices, for example, displaying a toolbar.

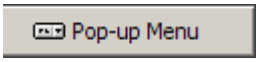
5.7.5 Edit Text

Edit text components are fields that enable users to enter or modify text strings. Use edit text when you want text as input. Users can enter numbers but you must convert them to their numeric equivalents.

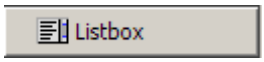
5.7.6 Static Text



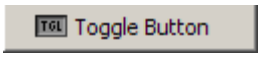
Static text controls display lines of text. Static text is typically used to label other controls, provide directions to the user, or indicate values associated with a slider. Users cannot change static text interactively.

5.7.7 Pop-Up Menu 

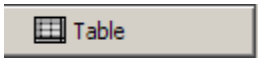
Pop-up menus open to display a list of choices when users click the arrow.

5.7.8 List Box 

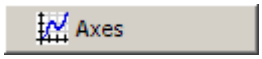
List boxes display a list of items and enable users to select one or more items.

5.7.9 Toggle Button 

Toggle buttons generate an action and indicate whether they are turned on or off. When user clicks a toggle button, it appears depressed, showing that it is on. When user release the mouse button the toggle button remains depressed until you click it a second time. When user does so, the button returns to the raised state, showing that it is off. Use a button group to manage mutually exclusive toggle buttons.

5.7.10 Table 

Use the table button to create a table component. Refer to the suitable function for more information on using this component.

5.7.11 Axes 

Axes enable your GUI to display graphics such as graphs and images. Like all graphics objects, axes have properties that user can set to control many aspects of its behavior and appearance. See Using Axes Properties in the MATLAB Graphics documentation and commands such as the following for more information on axes objects: plot, surf, line, bar, polar, pie, contour, and mesh. See Functions — By Category in the MATLAB Function Reference documentation for a complete list.

5.7.12 Panel 

Panels arrange GUI components into groups. By visually grouping related controls, panels can make the user interface easier to understand. A panel can have a title and various borders. Panel

children can be user interface controls and axes as well as button groups and other panels. The position of each component within a panel is interpreted relative to the panel. If user moves the panel, its children move with it and maintain their positions on the panel.

5.7.13 Button Group

Button groups are like panels but are used to manage exclusive selection behavior for radio buttons and toggle buttons.

5.7.14 ActiveX® Component

ActiveX components enable you to display ActiveX controls in GUI. They are available only on the Microsoft® Windows® platform. An ActiveX control can be the child only of a figure, i.e., of the GUI itself. It cannot be the child of a panel or button group. See ActiveX Control in this document for an example. See Creating COM Objects in the MATLAB External Interfaces documentation to learn more about ActiveX controls.

5.8 Add Components to the GUIDE Layout Area

1. Place components in the layout area according to user design
    - ❖ Drag a component from the palette and drop it in the layout area.
    - ❖ Click a component in the palette and move the cursor over the layout area. The cursor changes to a cross. Click again to add the component in its default size, or click and drag to size the component as user add it.

Once user has defined a GUI component in the layout area, selecting it automatically shows it in the Property Inspector. If the Property Inspector is not open or is not visible, double-clicking a component raises the inspector and focuses it on that component.

2. Assign a unique identifier to each component. Do this by setting the value of the component Tag properties. See Assign an Identifier to Each Component for more information.

3. Specify the look and feel of each component by setting the appropriate properties. The following topics contain specific information.

- ❖ Define User Interface Controls

- ❖ Define Panels and Button Groups
- ❖ Define Axes
- ❖ Define Tables
- ❖ Add ActiveX Controls

This is an example of a GUI in the Layout Editor as shown in Figure 5.8.1.

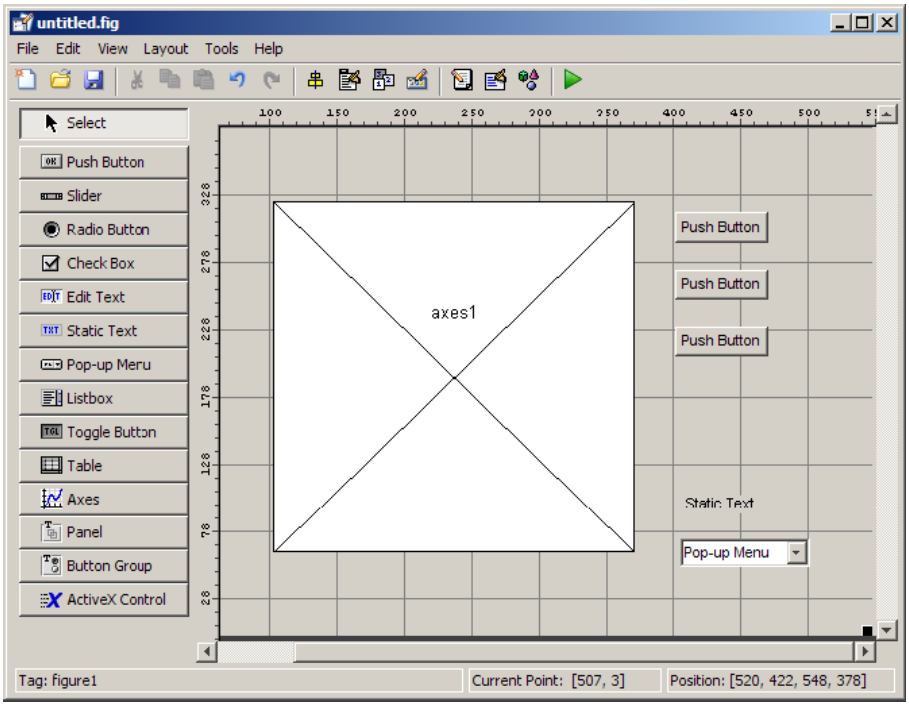


Figure 5.8.1 Example of GUI

5.9 Alignment Tool

The Alignment Tool enables user to position objects with respect to each other and to adjust the spacing between selected objects. The specified alignment operations apply to all components that are selected when user press the Apply button.

The alignment tool provides two types of alignment operations:

- Align — align all selected components to a single reference line.
- Distribute — Space all selected components uniformly with respect to each other.

Both types of alignment can be applied in the vertical and horizontal directions. In many cases, it is better to apply alignments independently to the vertical and horizontal using two separate steps

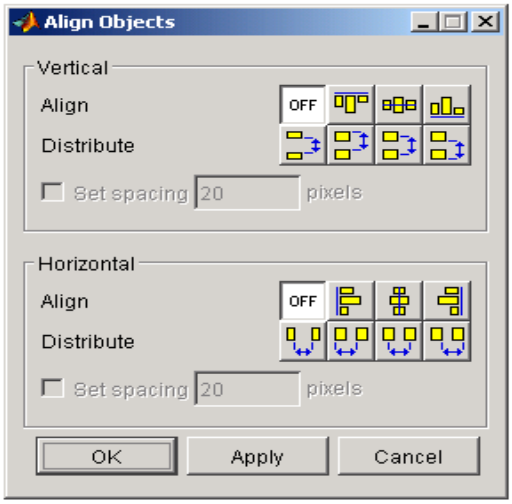


Figure 5.9.1 Alignment Dialog Box

5.9.1 Align Options

There are both vertical and horizontal align options. Each option aligns selected components to a reference line, which is determined by the bounding box that encloses the selected objects. For example, the following figure 5.9.1.1 of the layout area shows the bounding box (indicated by the dashed line) formed by three selected push buttons.

All of the align options (vertical top, center, bottom and horizontal left, center, right) place the selected components with respect to the corresponding edge (or center) of this bounding box.

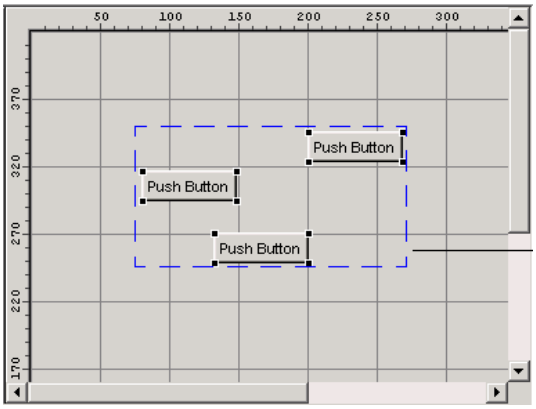


Figure5.9.1.1 Bounded Box

5.9.2 Distribute Options

Distributing components adds equal space between all components in the selected group. The distribute options operate in two different modes:

- Equally space selected components within the bounding box (default)
- Space selected components to a specified value in pixels (check Set spacing and specify a pixel value)

Both modes enable user to specify how the spacing is measured, as indicated by the button labels on the alignment tool. These options include spacing measured with respect to the following edges:

- Vertical — inner, top, center, and bottom
- Horizontal — inner, left, center, and right

5.10 Property Inspector

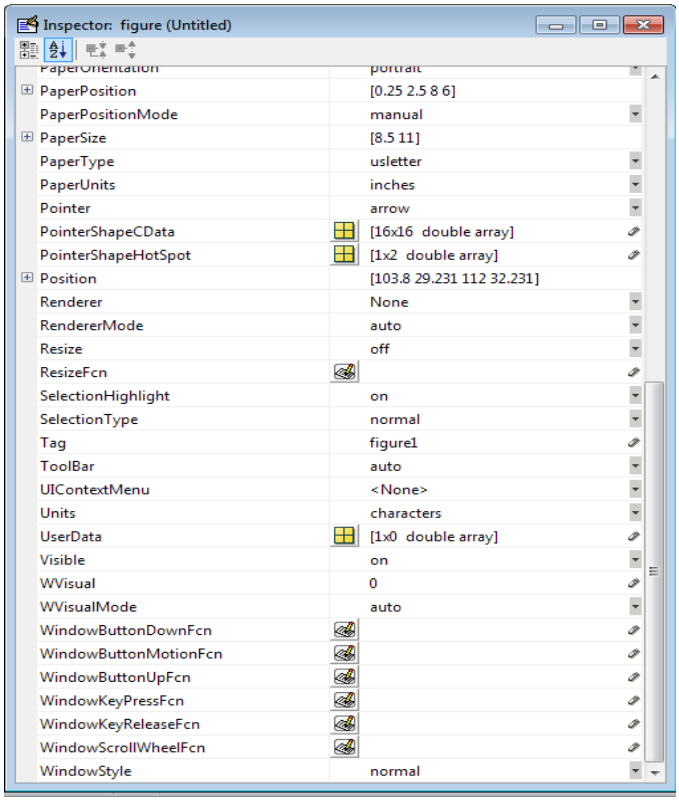


Figure5.10.1 Property Inspector Window

In GUIDE, as in MATLAB generally, user can see and set most components' properties using the Property Inspector. To open it from the GUIDE Layout Editor, do any of the following: Select the component user want to inspect, or double-click it to open the Property Inspector and bring it to the foreground

- Select Property Inspector from the View menu
- Click the Property Inspector button 

The Property Inspector window opens, displaying the properties of the selected component as shown in figure 5.10.1.

5.11 Grid and Rulers

The layout area displays a grid and rulers to facilitate component layout. Grid lines are spaced at 50-pixel intervals by default and you can select from a number of other values ranging from 10 to 200 pixels. User can optionally enable snap-to-grid, which causes any object that is moved close to a grid line to jump to that line. Snap-to-grid works with or without a visible grid.

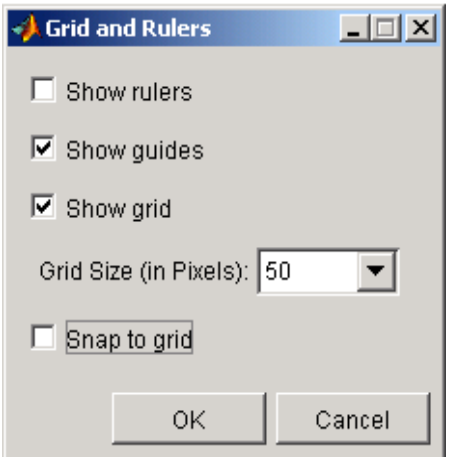


Figure 5.11.1 Grids and Rulers Dialog Box

As shown in figure 5.11.1, Use the Grid and Rulers dialog (select Grid and Rulers from the Tools menu) to:

- Control visibility of rulers, grid, and guide lines
- Set the grid spacing

- Enable or disable snap-to-grid

5.12 Guide Lines

The Layout Editor has both vertical and horizontal snap-to guide lines. Components snap to the line when you move them close to the line. Guide lines are useful when you want to establish a reference for component alignment at an arbitrary location in the Layout Editor.

5.12.1 Creating Guide Lines

To create a guide line, click the top or left ruler and drag the line into the layout area.

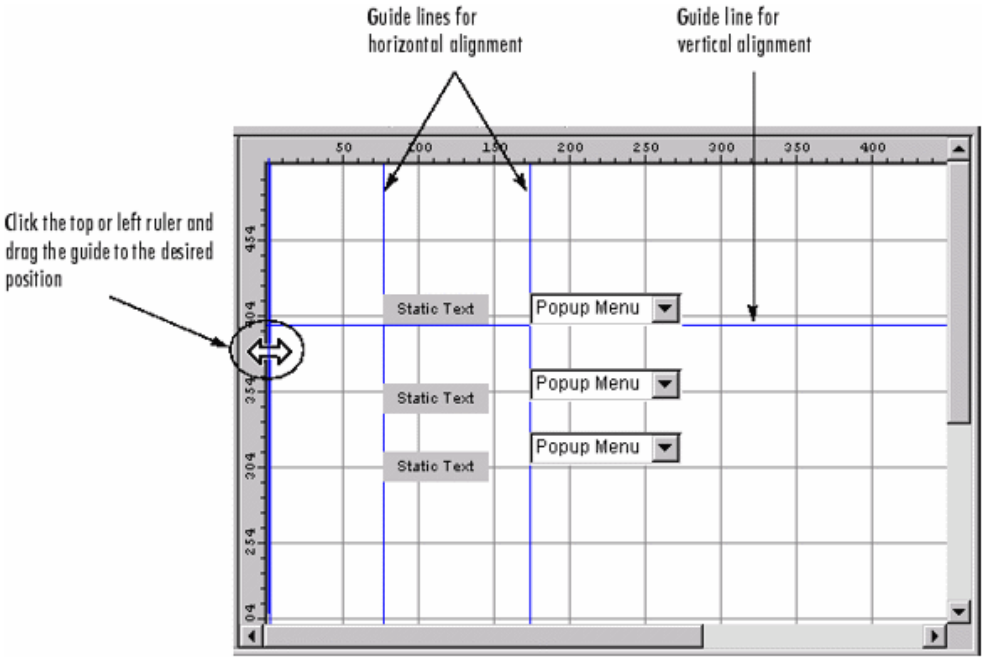


Figure5.12.1.1 Creation of Guide Lines

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Evaluation Parameters

The following parameters are considered for evaluating the image compression:

6.1.1 Peak Signal to Noise Ratio (PSNR): The expression for PSNR is:

$$PSNR = 10 \log_{10} (255^2/MSE)$$

6.1.2 Mean Square Error (MSE): The MSE between two images f and g is defined as:

$$MSE = \frac{1}{N} \sum_{j,k} (f[j,k]) - g[j,k])$$

Where the sum over j; k denotes the sum over all pixels in the images, and N is the number of pixels in each image.

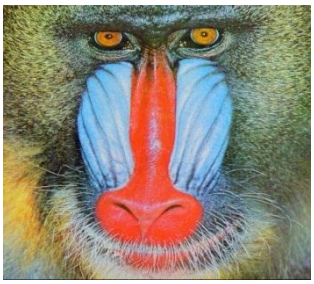
6.1.3 Compression Ratio (CR) = Original image pixels/Compressed image pixels.

6.1.4 Bits per Pixel (BPP) = the number of bits of information stored per pixel of an image.

6.2 Results

Four color images have been considered for the evaluation namely, Lena (1), Baboon (2), Penguins (3), Koala (4), and results for each are shown in subsequent section.

Lena (1)



Baboon (2)

Penguins (3)

Koala (4)

Figure 6.2.1 Standard Test Images

6.2.1 Implementation of Haar Wavelet Transform (HWT) and Bacterial Foraging Algorithm (BFO) on Standard Test Images for Compression

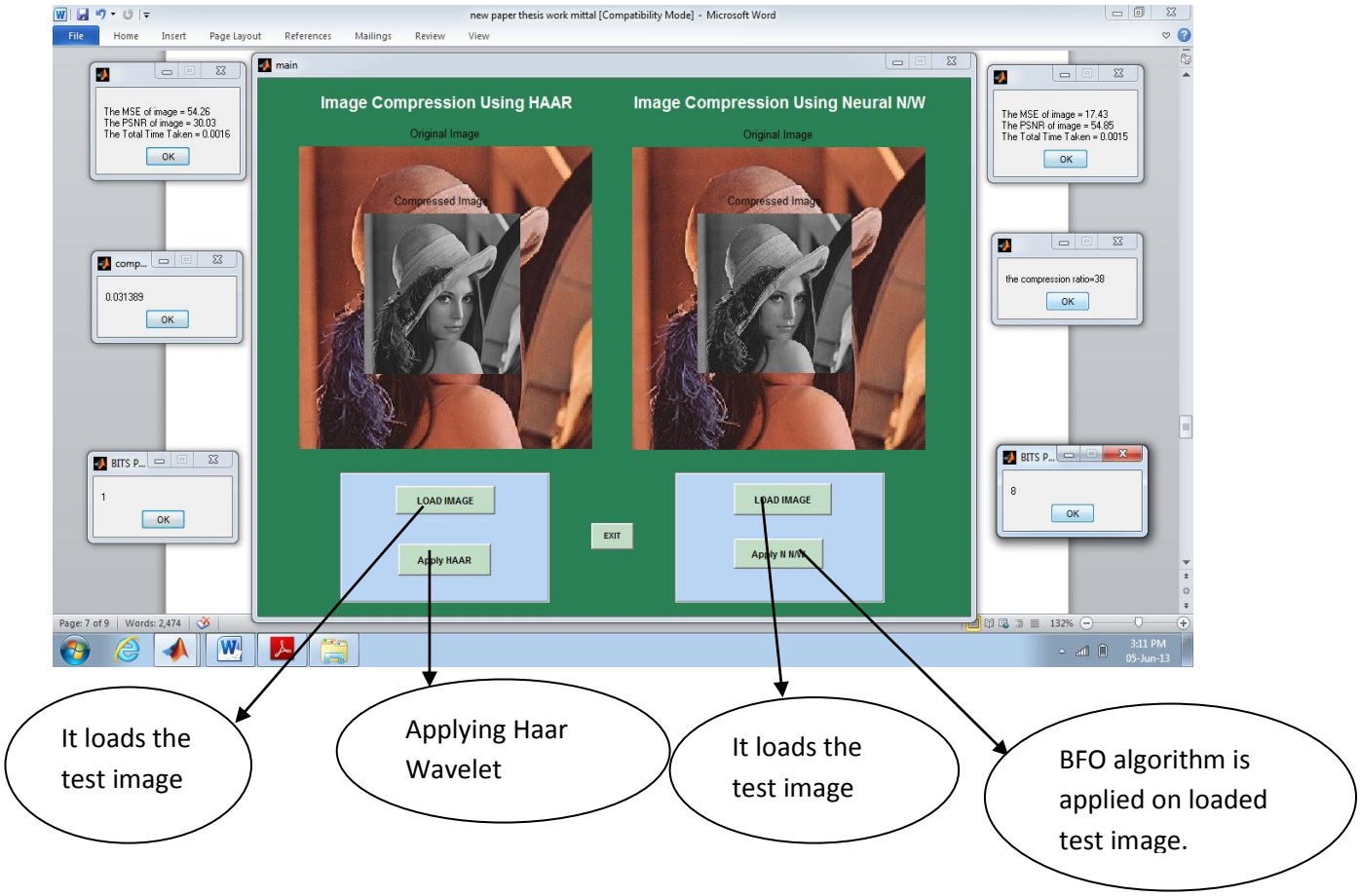


Figure 6.2.1.1 Implementation of Haar Wavelet Transform (HWT) and Bacterial Foraging Algorithm (BFO) on Lena Image for Compression in GUI

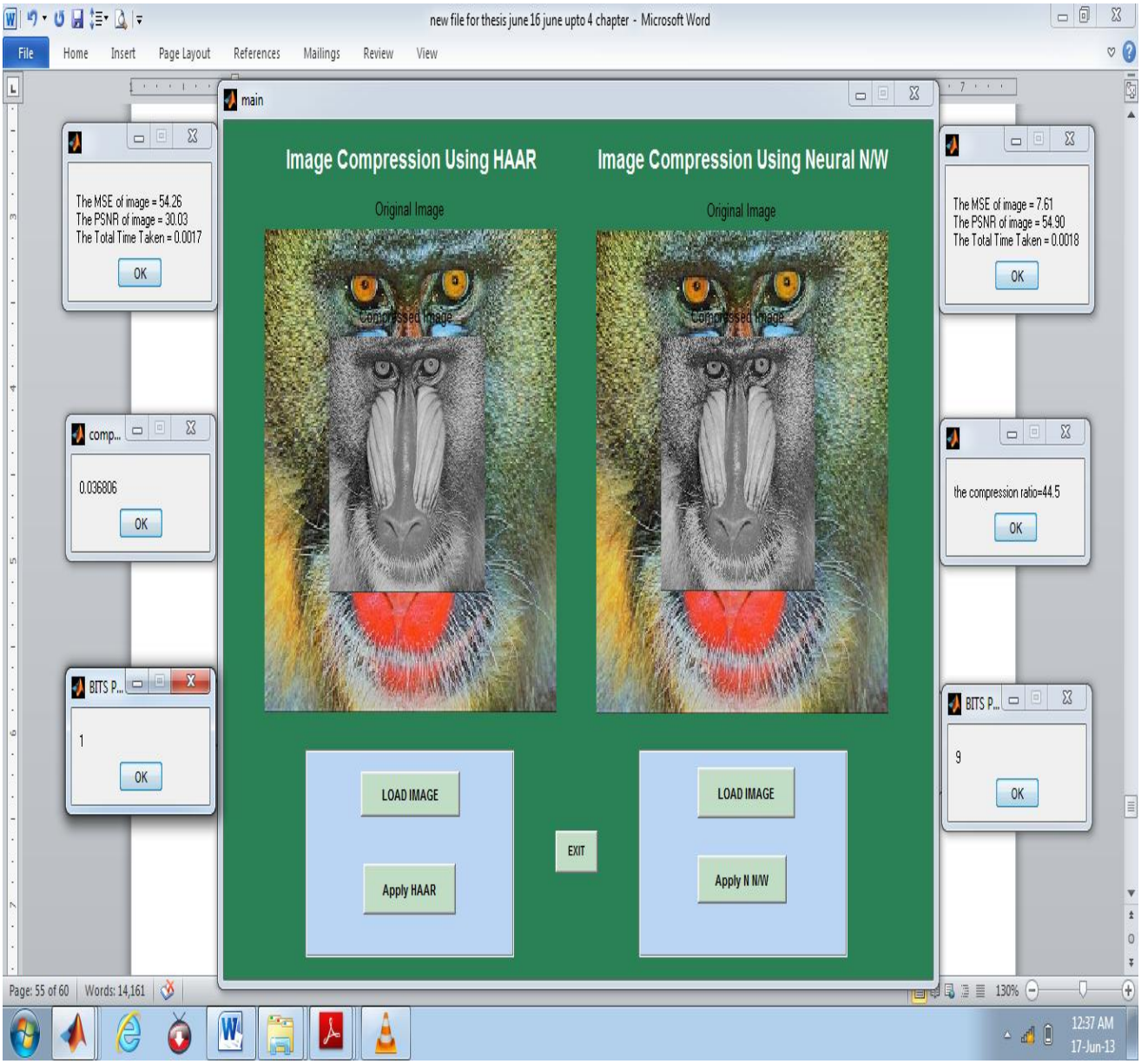


Figure 6.2.1.2 Implementation of Haar Wavelet Transform (HWT) and Bacterial Foraging Algorithm (BFO) on Baboon Image for Compression in GUI

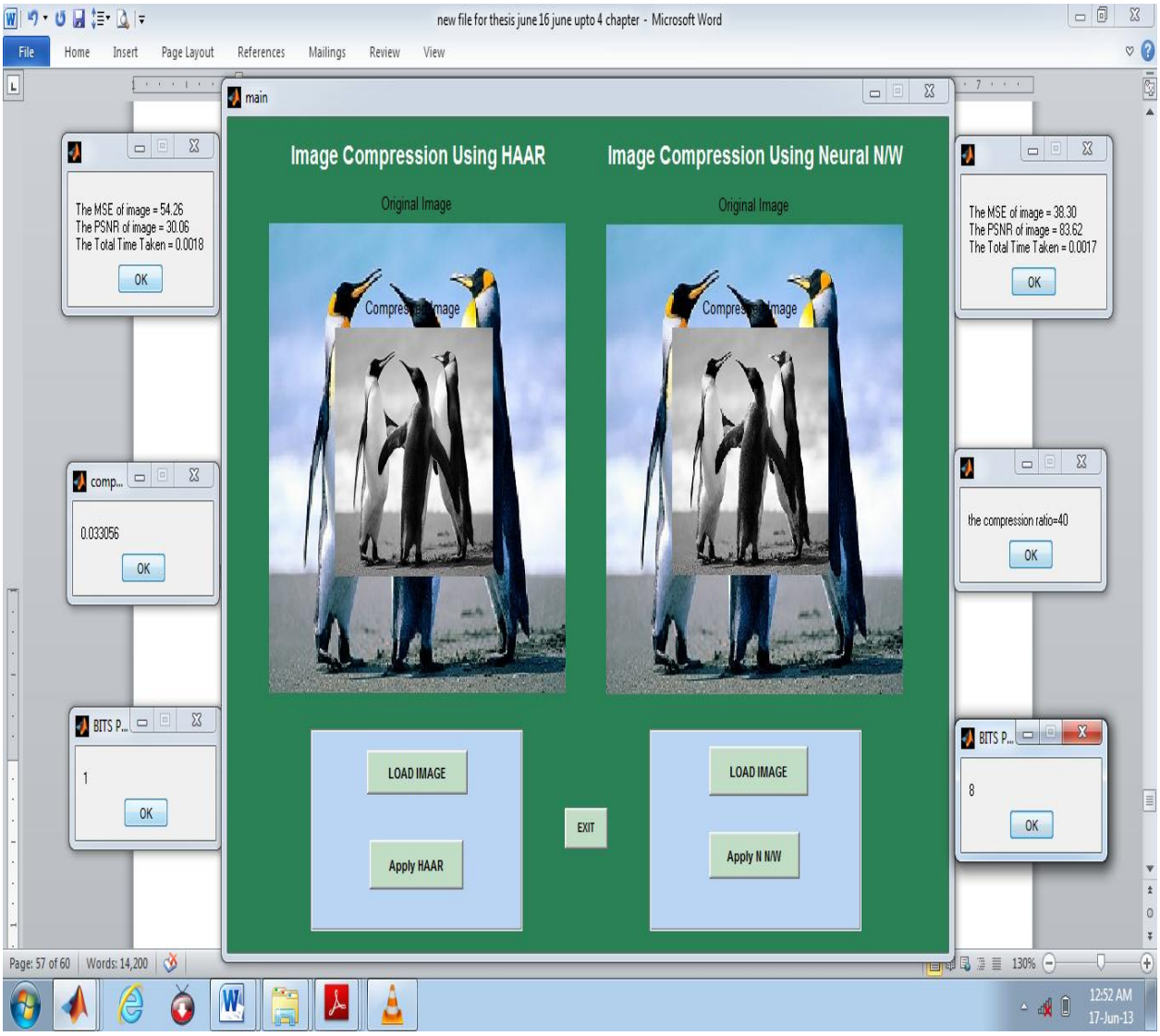


Figure 6.2.1.3: Implementation of Haar Wavelet Transform (HWT) and Bacterial Foraging Algorithm (BFO) on Penguins Image for Compression in GUI

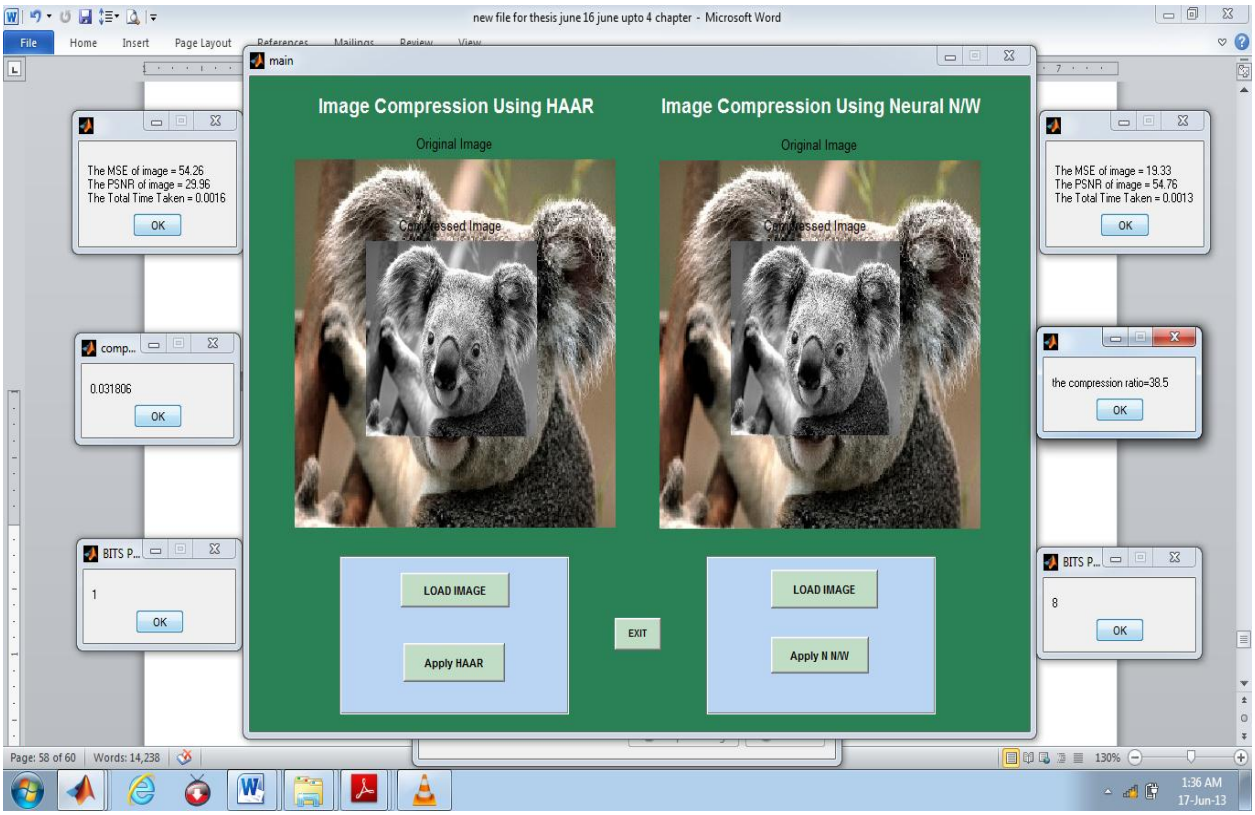


Figure 6.2.1.4 Implementation of Haar Wavelet Transform (HWT) and Bacterial Foraging Algorithm (BFO) on Koala Image for Compression in GUI

6.3 Comparison of Evaluation Parameters

For the purpose of analysis, the comparison of different parameters such as PSNR, MSE, CR and bpp for the image compression has been recorded by using Haar Wavelet Transform (HWT) and Bacterial Foraging Optimization (BFO) and are tabulated in Table I.

Table I Comparison of different parameters by implementing Haar Wavelet Transform (HWT) and Bacterial Foraging Optimization (BFO) Algorithm

Images	ALGORITHMS	PSNR	MSE	Compression Ratio(CR)	Bits Per Pixel(BPP)
Lena	Haar	29.83	54.26	0.02638	1
	BFO	54.85	17.43	38	8
Baboon	Haar	29.85	54.26	0.031806	1
	BFO	54.90	7.61	44.5	9
Penguins	Haar	30.04	54.26	0.028056	1
	BFO	83.62	38.30	40	8
Koala	Haar	30.00	54.26	0.026806	1
	BFO	54.76	19.33	38.5	8

The proposed algorithm i.e. Bacterial Foraging Algorithm as well as Haar Wavelet Transform (HWT) are implemented on standard test images namely Lena, Baboon, Penguins and koala for calculating different parameters such as Peak to Signal Noise Ratio (PSNR), Mean Squared Error (MSE), Compression Ratio (CR) and Bits Per Pixel (bpp). It can be clearly seen from the Table I that Bacterial Foraging Optimization (BFO) algorithm gives the better results as compared to Haar Wavelet Transform (HWT).

CHAPTER 7

CONCLUSION

Conclusion

The thesis presents a new algorithm for the improvement of different parameters for the image compression. In the thesis, two algorithms namely, Haar wavelet transform and Bacterial Foraging Optimization (BFO) algorithm are implemented for compressing images. Using each algorithm, some qualitative parameters such as Peak Signal to Noise Ratio, Mean Squared Error, Bits per Pixel, Total time taken, Compression ratio (CR) are calculated. It can be concluded that PSNR, Compression ratio (CR), Bits per Pixel (BPP) of standard test images is more in BFO algorithm as compared to Haar Wavelet Transform (HWT), while MSE is less in Bacterial Foraging Optimization (BFO) algorithm than Haar Wavelet Transform (HWT).

As seen in this thesis work, Bacterial Foraging Optimization (BFO) algorithm has been implemented successfully over Haar Wavelet Transform (HWT) for the image compression. The same experiments can also be conducted with other types of neural networks to see the improvement in performance of the system.

REFERENCES

[1] Nasser. N. Nasarbadi, Member IEEE and Robert. A. King, "Image coding using vector quantization, A Review" , IEEE Transactions in communications, Vol 36, No.8, pp. 957 -971 August 1988

[2] Omaima N.A. AL-Allaf, "Improving the Performance of Backpropagation Neural Network Algorithm for Image Compression/Decompression System", Journal of Computer Science 6(11), ISSN 1549-3636, Science Publications, pp. 1347 – 1354, 2010

[3] Gleb V. Tcheslavski, "Image compression fundamentals"

[4] http://en.wikipedia.org/wiki/Image compression

[5] http://en.wikipedia.org/wiki/Comparison of graphics file formats

[6] Anant Raj Singh, "EZW coding with improved Execution Time on the Basis of Descendant Scanning of Zero-tree Roots", 2012

[7] Sonal, Dinesh Kumar, " A Study of various Image compression techniques"

[8] Abbas Razwi, Rutie Adar. Isaac Shenberg, Rafi Retter and Rami Friedlander "VLSI implementation of an image compression algorithm with a new bit rate control capability" Zoran Corporation 1705 Wyatt Drive, Santa Clara, CA 95054, IEEE International conference Vol-5, pp. 669-72, 26th March, 1992

[9] David Jeff Jackson and Sidney Jwl Hannah, "Comparative analysis of image compression technique", system theory 1993, proceeding SSSST'93, and 25th edition southeastern symposiums pp. 513-517, 9th March, 1992

[10] J Jiang, "Neural network technology for image compression" Broadcasting convection, Bolton Institute, UK, IBC 95, pp. 250-257, 18 September, 1995,

[11] Wayne O. Cochran, John C. Hart "Fractional Volume Compression" Member of IEEE computer society, Visualization and Computer Graphics IEEE, pp. 313-322, December 1996

[12] H. Nait Charif University Mohammed 1, B.P. 473, Oujda 60000, Morocco and Fathi M. Salam "Neural networks based image compression system" Proc. 43rd IEEE Mdwest Symp. On Circuits and Systems, Lansing MI, 0-7803-6475-9/ IEEE, Aug 8-11.2000.

[13] Koon Pong Wong "Fractal image coding for emission topographic image compression", IEEE Transactions on Nuclear Science symposium conference 2002.

[14] Adnan Khashman, Kamil Dimililer, "Image Compression using Neural Networks and Haar Wavelet" WSEAS Transactions on Signal processing, ISSN: 1790-5052, Vol 4, Issue 5, May 2008.

[15] Chu Ying, Mi Hua, Ji Zhen, "BFA Based Neural Network for Image Compression" Journal of Electronics (China), Vol.25, No.3, May 2008.

[16] Anuj Bhardwaj and Rashid Ali, "Image Compression Using Modified Fast Haar Wavelet Transform (MFHWT)" World Applied Sciences Journal 7 (5): 647-653, ISSN 1818-4952 IDOSI Publications, 2009.

[17] K. P. Soman; K. I. Ramchandran; "Insight into Wavelets- From Theory to Practice" Prentice Hall of India, Second Edition, pp. 6-9, 2005.

[18] Sachin Vikal, Pankaj Sharma, Rachit Shyam Khanna, and Sandeep Gupta, "Advanced Image Compression: Haar Wavelet", International Journal of Research Review in Engineering Science and Technology, (ISSN 2278- 6643), Vol 1, Issue 2, September 2012.

[19] Quality Assessment of Colour Image Compression using Haar Wavelet Transform Sanjeev Kumar, Varun Sood. International Journal of Engineering Trends and Technology, ISSN: 2231-5381, Vol 3, Issue 3, 2012

[20] Navjot Kaur, Preeti singh, "Enhancement of compression ratio and image quality using ISPIHT with MFHWT", International journal of computer applications: (0975-8887) Vol 54, No. 8, September 2012.

[21] Heena, Dr. Himanshu Aggarwal, "Color Image Quantization Based on Euclidean Distance Using Bacteria Foraging Optimization", International Journal of Electronics and Computer Science Engineering, ISSN 2277-1956, Vol 1, No.4, pp. 2285-2290

[22] Dharminder Kumar, "Implementation of Bacteria Foraging Optimization for Color Image Quantization and its Evaluation for Various File Formats", International Journal of Computer Science and Communication Engineering, ISSN: 2319-7080, Vol 2, Issue 1, Feb, 2013

[23] Stephens and J. Krebs, *Foraging Theory*. Princeton, NJ: Princeton Univ. Press, 1986.

[24] G. Kevin Passino, "Biomimicry of Bacterial Foraging for Distributed Optimization and Control," IEEE Control Systems Magazine, June 2002

[25] T. Audesirk and G. Audesirk, Biology: Life on Earth. Prentice Hall, NJ, 5 editions, 1999. K. Kristinsson and G. Dumont, "System identification and control using Genetic algorithms," IEEE Trans. Syst., Man, Cybernet., vol. 22, no. 5, pp. 1033-1046, 1992

[26] H. Berg, Random Walks in Biology. Princeton, NJ: Princeton Univ. Press, 1993

[27] D. DeRosier, "The turn of the screw: The bacterial flagellar motor," Cell, Vol 93, pp. 17-20, 1998.