Sudan University of Science and Technology

College of Graduated Studies

Suitable Steganography Scheme for Securing Data through internet Applications

طريقة إخفاء مناسبة لتأمين البيانات عبر تطبيقات الإنترنت

A Thesis Submitted in partial fulfillment for M.Sc communication Engineering

Prepared by:

Noha Omer Abuelgasim Ahmed

Supervisor:

Dr: Jacqueline George

الآية

قال تعالى:

{ يَرْفَعِ اللهُ الَّذِينَ آمَنُوا مِنْكُمْ

وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ}

صدق الله العظيم

سورة المجادلة, الآية 11

Abstract

Steganography is the technique that hides the presence of a secret data in a cover-medium without drawing the attention of unauthorized viewers. So, with the fact that the internet acts as common channel, the problem of information security arise, Steganography is one of the techniques used to resolve this problem. In fact there are many image-based steganography techniques in use, and there are several problems arise due to distortion of stego-image produced after embedding the secret data, and error swill be occurred during restoration the sent image. Therefore, the primary objective of this thesis is to compare between some of them, two in the image spatial domain (LSB and PVD) and the other are in its frequency domain (IWT and DCT). to chose the best one suitable for securing data through internet applications. In order to achieve the goal, the mentioned techniques will be studied, analyzed and evaluated in terms of restored message accuracy and quality of the produced stego-image. Then the graphical user interface (GUI) for the best one will be designed to give easy usage for the program users.

مستخلص

إخفاء المعلومات هو الأسلوب الذي يخفي وجود بيانات سرية في وسط مغطي دون أن يلفت انتباه المشاهدين غير المصرح بهم .لذلك، ومع حقيقة أن الإنترنت بمثابة قناة مشتركة، وان مشكلة أمن المعلومات تنشأ، فان إخفاء المعلومات هي واحدة من التقنيات المستخدمة لحل هذه المشكلة . في الواقع هناك العديد من تقنيات إخفاء المعلومات القائم على صورة قيد الاستخدام، وهناك العديد من المشاكل تنشأ بسبب تشويه الصورة الناتجة بعد تضمين البيانات السرية والعديد من الأخطاء تقع خلال استعادة الصورة المرسلة .ولذلك، فإن الهدف الرئيسي من هذه الدراسة هو المقارنة بين بعضهم، اثنان في نطاق الصورة المكاني (LSB و PVD)واثنان في النطاق الترددي للصورة (DCT وIWT)لاختيار أفضل واحدة مناسبة لتأمين البيانات من خلال تطبيقات الإنترنت .من أجل تحقيق هذا الهدف،ستتم دراسة التقنيات المذكورة وتحليلها وتقييمها من حيث دقة استعادتها للرسالة وجودة إنتاجها للصورةالتي تحمل البيانات. ثم واجهة المستخدم الرسومية (GUI) لأفضل واحدة سيتم تصميمها لإعطاء الاستخدام السهل لمستخدمي البرنامج.

الإهداء

إلىمن أحبتني كما أنا,دون قيد ولا شروط,أمي أطال الله في عمرها.

الى روح ابي التي سافرت قبل أن يهنئني ويفرح بي؛ وسع الله في قبره و اسكنه فسيح جناته.

إلى شركائي في السراء و الضراء أخوتي و أبناءهم الَّذِين أظلوني طوال مسيري في مجير البحث.

إلى نصفي الآخر و رفيق دربي زوجي الغالي.

إلى بناتي اللواتي تحملن إهمالي و إنشغالي و إنتظرن هذا اليوم بشغف.

إلى كل من علمني حرفاً.

إلى جميع أصدقائي المخلصين.

إلي رفقائي في الدراسة.

Acknowledgments

I am very fortunate to have performed my post graduate work at Sudan University for science and technology. I am lucky enough to have been given the supportive gift of amazing people in my life, without all of whom, this work would not have been completed. Thank you to...

My Parents, for instilling in me from a young age the belief that I Can;

My Sisters and brothers, you have all provided support, encouragement and interest in my thesis work.

My close friends, for seeing things in me that I didn't;

A special thanks to Dr. Jacqueline George, my advisor, for her countless hours of reflecting, reading, encouraging, and most of all patience throughout the entire process.

Table of Contents

List of Figures

List of Tables

.

Chapter One

Introduction

1.1 Background

Since the rise of the Internet one of the most important factors in networking is the security of information. Many securing techniques are being used for security purposes (e.g. Steganography and cryptography) which are the popular techniques. Cryptography scrambles a message in such a way that it cannot be understood, while Steganography is the art and science of invisible communication. It is accomplished by hiding information in other information, thus hiding the existence of the information. Steganography techniques may be implemented using text, image, audio or video files as a data to be hidden or a medium to hide a secret data. In the image Steganography the information is hidden exclusively in images. Today Steganography is mostly used on computers with digital data being the carriers and networks being the high speed delivery channels.

Steganography is used for a wide range of applications; such as in smart identity cards where personal details are embedded in the photograph itself for copyright control of materials, medical imaging where patient's details are embedded within image providing protection of information and reducing transmission time, and finally on performing secure banking Operations.

.

1.2 Problem Statement

The main problem in internet applications is transferring the data through the internet environment which is an open, non-secure media.

1.3 Research Objectives

The objectives of this research is to

- Use steganography data hiding technique to secure data sending through

internet media.

- Compare the performance of the steganography techniques both in the spatial and frequency domain and choose the one suitable for security purposes.

1.4 Research Methodology

In this thesis some image steganography techniques from different domains were compared to choose the best steganography technique suitable for securing through internet information's. The techniques that were considered are:simple LSB substitution technique, pixel-value differencing (PVD)-based technique from the spatial domain techniques; Discrete Cosine Transform Technique (DCT), Discrete Wavelet Transform Technique (DWT) from the frequency domain techniques.A graphical user interface (GUI) for the best technique was designed using matlab program

1.5 Thesis Outline

This thesis consists of five chapters detailed as follow;

Chapter 2: presents an in-depth literature review on the thesis topic including the related works, history of steganography, steganographic categories, and classification of steganographic techniques, steganography in security domain, and the difference between cryptography, watermarking and steganography.

Chapter 3: discusses about the process of defining, implementing the detailed algorithms for the chosen techniques.

Chapter 4: provides the simulation and discusses the results.

Chapter 5: provides the conclusion and suggest some future work.

Chapter two

Literature survey

2.1 Preview

Steganography is the art and science of invisible communication. It is accomplished by hiding information in other so no one can sense the presence of data or secret message. So, andwith the problem of the internet nature which is a common communication channel, the art of hiding through internet information have received much attention in the recent years, as security of information was become a big concern in this internet era.The basic model for image Steganography is shown in fig. (2.1).

Figure (2.1): Basic Steganography Model

It shows the basic process involved in image Steganography technique as an example which consists of Carrier, Message and Key[1]. Carrier is also known as cover-image, in which the message is embedded after converting the image to a specific redundant data and serves to hide the presence of the message. The secret message can be any type of data (plain text, cipher text or other image) that the sender wishes to remain confidential. Key is known as stego-key, which ensures that only recipient who knows the key, corresponding decoding key will be able to recover the message from a cover-image. The cover-image with the image secretly embedded message is then called the stego-image[1].

Recovering message from a stego-image requires the cover-image itself and a corresponding decoding key if an image stego-key was used during the information encoding process

2.2 Related Work

Many researches have been done in this area, some of which are listed as follow;

In [2] the authors used LSB technique to secure data in e-banking environment. The authors used bmp image format as acover media, sent a message and restored it back at the receiver. The results showed there was a noticeable difference between the cover and the stego image.

In [3] the author apply a Steganography algorithm known as Dynamic Pattern based Image Steganography (DPIS), 'this algorithm is depend on LSB technique' To give more securityfor E-Banking purposes.The author said that he was get good results.

In [4] The Discrete Cosine Transform algorithm is used to embed secret data into the jpeg image,for internet banking security.The technique produced unnoticeablechange to the cover image, and gave good results at the receiving side. The author preferred the jpeg image because of it is ability to be compressed and saved in a less space, resulting in a less transferring cost.

In [5] the author used (DCT) with multi factor for authentication purposes. The author aims was tomaximize the amount of hidden message and minimize the difference between the cover image and stego-image. The proposed scheme conceal large amount of secret information in the cover images and also

generate repaired image, which has a small distortion compared with the original cover image.

In [6] The authors preferred pixel domain steganography technique because Images in JPEG format are very poor choice for cover-images. The authors show thatsecurity of e-commerce has been improved using random LSB steganography and cryptography methods together instead of using either steganography or cryptography.

In [7] the author hide data for security purposes using discrete wavelet transform for embedding data within skin region of an image. The author'sstudy showed that by adopting an object oriented steganography mechanism, in the sense that, skin tone objects in image is tracked, a higher security is obtained as well as a satisfactory PSNR (Peak- S-to-Noise Ratio) and MSE. But the mechanism suffered from less extraction quality.

In [8] the author compared between some spatial and frequency domain techniques and thenidentified the requirements of a good Steganography algorithm and briefly reflected on which Steganography techniques are more suitable for which applications. The author showed that, each of them having different strong and weak points respectively. Whereas LSB technique has high payload capacity, while lacks in robustness, The DCT technique have less payload capacity,while higher robustness against manipulation

In [9] the authors compare between (LSB), (DCT), and (DWT), and found that LSB technique have the highest payload capacity, DCT technique have the highest PSNR, DWT technique have the highest robustness against manipulation.

2.3 History of Steganography

It is believed that steganography was first practiced during the Golden Age in Greece. An ancient Greek record describes the practice of melting wax off wax tablets used for writing messages and then inscribing a message in the underlying wood. The wax was then reapplied to the wood, giving the appearance of a new, unused tablet. The resulting tablets could be innocently transported without anyone suspecting the presence of a message beneath the wax. Another ancient story is about Histiæus sent his most trusted slave to the Ionian city of Miletus with a secret message tattooed on his scalp. After tattooing, the slave grew his hair back in order to conceal the message. Later, Aeneas the Tactician, proposed numerous steganographic techniques such as hiding messages in women's earrings or messages carried by pigeons. Based on the literature in [11] another idea that was proposed by Brewster in 1857 to shrink the hidden messages for not larger than a full stop or small dot of ink. The shrinking technique was successfully realized by a French photographer Dragon during the Franco-Prussian War (1870 - 1871). Apparently, microscopic images were hidden in ears, nostrils, and under fingernails. In World War I, Brewster's idea was used when secret messages sent by spies were reduced to microdots and hidden in the corners of postcards slit open with a knife and resealed with starch. In the twentieth-century, the modern microdots could hold up to one page of text and even contain photographs [11]. Another common form of invisible writing is through the use of Invisible inks. Such inks were used with much success as recently as WW-II. An innocent letter may contain a very different message written between the lines. Early in WW-II steganographic technology consisted almost exclusively of invisible inks. Common sources for invisible inks are milk, vinegar, fruit juices and urine. All of these darken when heated.

Today, unlike the past, the boom of steganography coincides with the appearance of the Internet especially when the shift to digitization of media (e.g. digital documents, digital images, digital audio and so forth) in the computer networks has created a very favorable environment for steganographic usage.

2.4 Steganographic Categories

Steganography is classified into 3 categories;

- Pure steganography where there is no stego key. It is based on the assumption that no other party is aware of the communication.
- Secret key steganography where the stego key is exchanged prior to communication. This is most suspectible to interception.
- Public key steganography where a public key and a private key is used for secure communication.

2.5 Classification of Steganographic Techniques

Steganography techniques may be implemented as in figure (2.2) using; text, image, audio or video files as a data to be hidden or a medium to hide a secret data.

Steganographic techniques can be grouped in different ways. Figure (2.3) show that the steganographic techniques grouped into six categories by how the algorithm encodes information in the cover object.

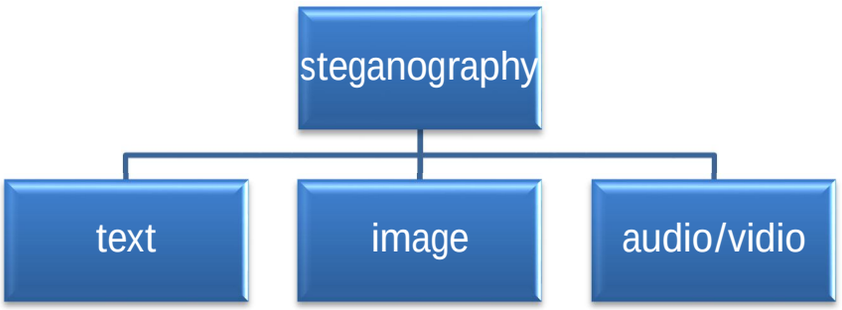


Figure (2.2): Classification of Steganographic Techniques

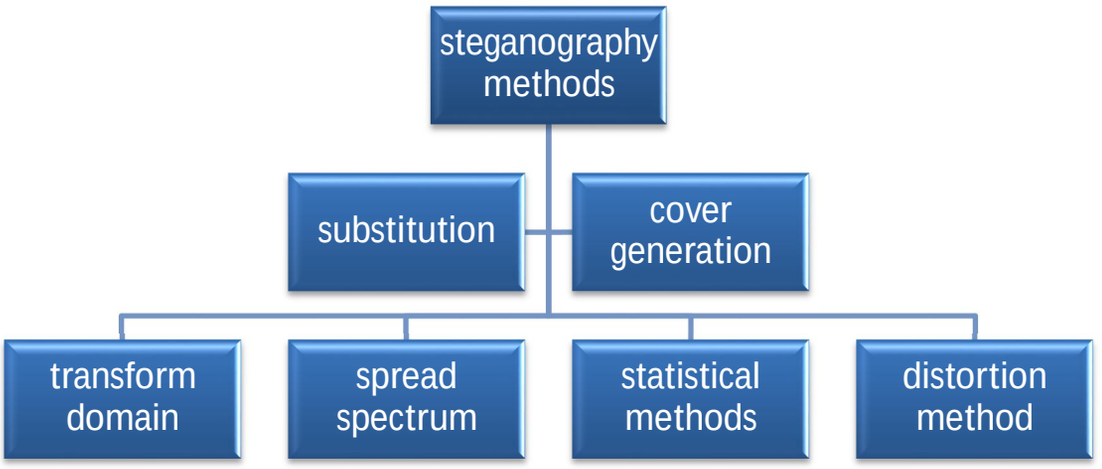


Figure (2.3): Classification of Steganographic methods

The Substitution methods substitute redundant parts of a cover with a secret message (spatial domain). The Transform domain techniques embed secret information in a transform space of the signal (frequency domain). The Spread spectrum techniques adopt ideas from the spread spectrum communication.Statistical methods encode information by changing several statistical properties of a cover and use hypothesis testing in the extraction process. The Distortion techniques store information by signal distortion and measure the deviation from the original cover in the decoding step.Cover

generation methods encode information in the way a cover for secret communication is created.

2.6 Steganography in Security Domain

Electronic communication and online finance transaction are increasingly susceptible to obtrusive eavesdropping, malicious frauds and unauthorized interception. The issues of security and privacy have conventionally being safeguarded by using one of two ways of concealing information:cryptography and steganography. So that only the intended recipient is able to verify the authentication and retrieve the secret message.

Figure (2.4) illustrate steganography in securitydomain. The security system uses one of two ways to secure information; Cryptography, the technique that send unreadable data, except with the intended recipient, and Information Hiding which isthe way a data is sent after being hidden in another object. So no one senses the presence of a hidden data except the person that knows or expects it.

There are two data hiding techniques; Watermarking and Steganography. Watermarking is to embed a mark in a cover-object to produce watermarked object which is indistinguishable from the cover, thus an eavesdropper is not able to replace or remove it. It is used to protect the ownership and copyrights of digital data. Steganography uses two techniques to hide data; Linguistic steganography and Technical steganography.

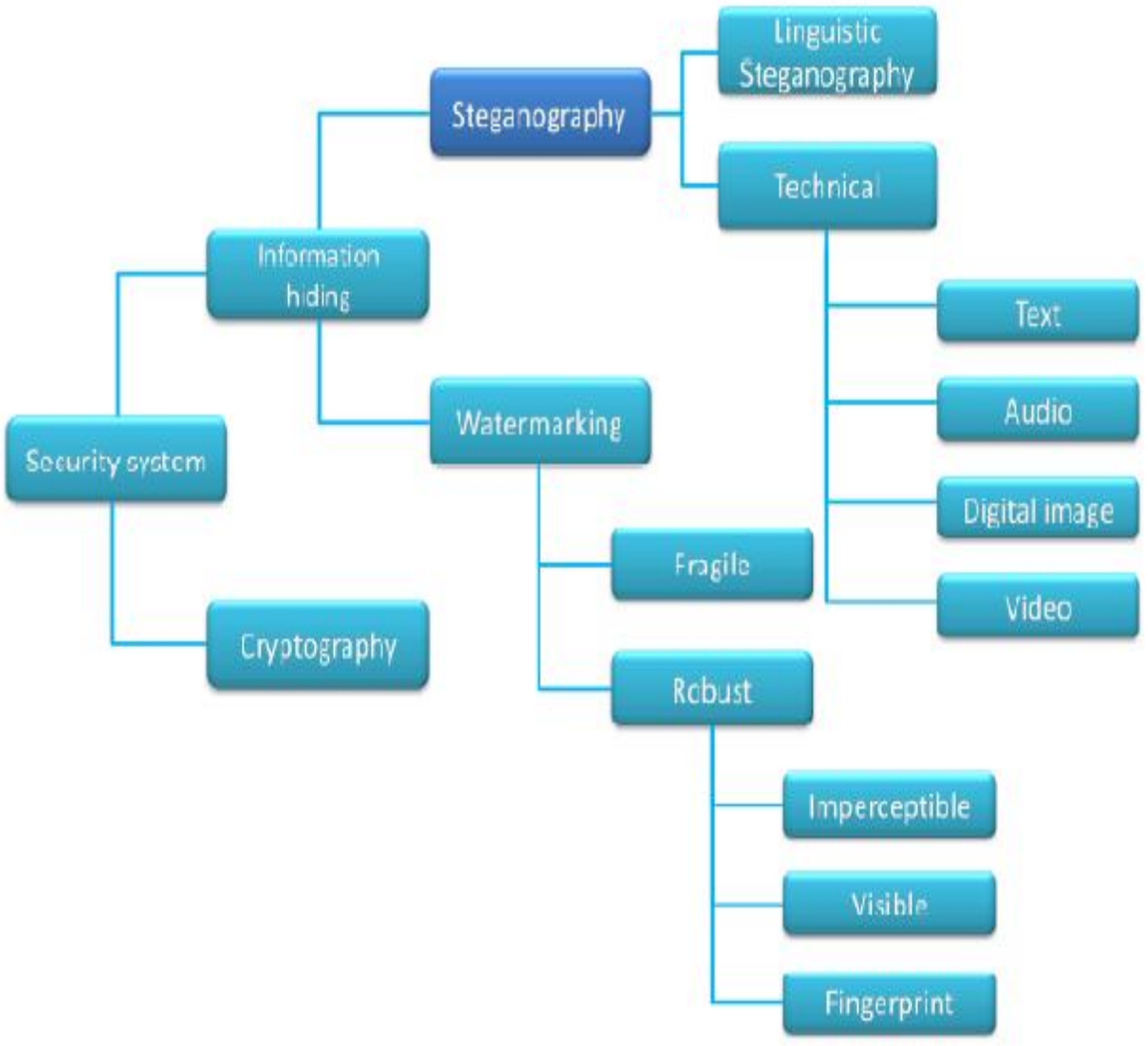


Figure (2.4): Steganography in Security Domain

Linguistic steganography, i.e. using the language for sending a secret message – by symbols, ambiguous meanings, re-arrangement of letters and other forms of linguistic manipulation. The linguistic steganography for computer systems is still purely theoretical. In technical steganography, the secret data or a medium is used to hide a secret data(cover medium) may be implemented using; text, image, audio or video files.

2.6.1 Image Steganography:

In image steganography the image is used as the medium in which the information is hidden. Figure (2.5) shows the process of image steganography. Figure (2.5a) shows the image which is used as a cover medium to send the secret text message, it is 12 Kbyte. After imbedding the secret data the image is then known as the stego-image and it will 253Kbyte as shown in Figure (2.5b).

(a) (b)

Figure (2.5): Image Steganography a) Cover-image b) Stego-image

The stego-object (embedded with secret data) looks similar to the original cover-object, so it will not raise suspicion about the presence of secret data.

2.6.2 Audio Steganography:

In Audio Steganography the message is added to the unused frequency in them, and – once again – the human ear is unable to detect the difference in the sound quality. And whereas the person may be able to detect the difference by looking at the diagram, it is much more difficult to hear.

Figure (2.6.a): Frequency Diagram of an Audio Transcript

Figure (2.6.b): The same audio transcript within the secret message

2.6.3 Video Steganography:

Although BMP files are perfect for steganographic use, they are able to carry only small files. So there is a problem, how to get much enough files to hide the message, and what to do to read them in a correct order? Good way out is to hide information in a video file, because the AVI files are created out of bitmaps, combined into one piece, which are played in correct order and with appropriate time gap. Keeping that in mind what needs to be done is to get out the file single frames and save them as BMP files. If an algorithm is used for hiding data in digital pictures, the message can be hidden in a bitmap obtained in this way, and then saved into a new AVI file.

2.6.4 Text Steganography:

In Text steganography, the message is hidden in a normal text file. Sometimes, this is done by hiding the message in the blank spaces between words. The message is separated between the LSBs of the binary code for the empty space throughout the text. This method requires the text that is sent to be considerably longer than the message hidden within it. A message can also be hidden in PDF documents and in a variety of other standards, depending on the program used.

2.7 Cryptography vs Steganography:

Steganography is an alternative practice in protecting confidential information in the realm of information security. Even though steganography is like a cousin of cryptography, both of them differs in terms of certain aspects as shown in Table 2.1.

Table 2.1: Difference between Cryptography and Steganography

Cryptography	Steganography
The goal of cryptography is aim for a strong encryption algorithm so that an attacker will face difficult to decipher the cipher text.	The goal of steganography is to conceal a secret data so that any unauthorized party will not Detect the stego-object.
The encrypted data which appear in a form of scrambled message will raise suspicion.	The stego-object (embedded with secret data) looks similar to the original cover object which will not raise suspicion.

2.8 Steganography vs Digital Watermarking

Steganography and digital watermarking are closely related fields that have a great deal of overlapping and similarities of many technical approaches.

However, there are fundamental criteria differences between steganography and digital watermarking as shown in Table 2.2.

Table (2.2): Difference between Steganography and Digital Watermarking

Steganography	**Digital Watermarking**
The main purpose is to embed a secret data in a cover-object to produce a stego-object which is in distinguishable from the cover, thus an eavesdropper is not able to detect the presence of the secret data.	The main purpose is to embed a mark in a cover-object to produce a watermarked object which is indistinguishable from the cover, thus an eavesdropper is not able to replace or remove it.
To hide a secret message in one-to-one communication	To hide a message in one-to-many communications
To conceal the presence of a secret message	To protect the ownership and Copyrights of digital data.
Optional to provide a strong security against any modification and manipulation of the Embedded message.	Must provide a very robust embedding method to prevent modification and manipulation that results in removing the mark.

Chapter Three

Research Methodology

3.1 Overview

This chapter describes the techniques that used in this research to choose the best one of them for Securing through internet Applications. Initially, a brief introduction is covered then the Algorithm to embed text and the Algorithm to retrieve it back, for each technique. Before concluding the chapter, each of the steps taken to carry out the research is explained.

There are two types of domains in which steganography is implemented i.e. spatial domain & frequency domain. Each domain has different techniques that used to embed data in the cover media.

3.2 Steganography Techniques

There are two types of domains in which steganography is implemented; the spatial domain and the frequency.

3.2.1 Spatial Domain Based Steganography Techniques

In Spatial domain, processing is applied directly on the pixel values of the image. There are different techniques used in the spatial domain such as; The Least Significant Bits (LSB) technique, the Pixel Value Differencing technique (PVD), pseudorandom permutation, cover regions and parity bit, statistical techniques, information hiding in binary images, Pixel Intensity based steganography, bit-plane complexity segmentation (BPCS) steganography. In this thesis the LSB and PVD techniques will be considered.

3.2.1.1 Least Significant Bit (LSB)

In Least significant bit substitution Technique (LSB) steganography, the least significant bits of the cover media’s digital data are used to conceal the message.

The simplest of the LSB steganography techniques is LSB replacement. LSB replacement steganography flips the last bit of each of the data values to reflect the message that needs to be hidden. An example of LSB technique is provided;
Consider a grid for 3 pixels of a 24-bit image.

Original image pixels:
(01010101 01011100 11011000)
(10110110 11111100 00110100)
(11011110 10110010 10110101)
The character C is to be embedded using LSB technique.
C: 10000011

The resulting new image is as follows:
(01010101 01011100 11011000)
(1011011011111100 00110100)
(1101111**1** 1011001**1** 10110101)

The bits of the number C was embedded into the first 8 bytes of the grid, only 2 bits needed to be changed according to the embedded message (marked in red) .On average, only half of the bits in an image will need to be modified to hide a secret message using the maximum cover size.

3.2.1.2 Pixel-Value Differencing (PVD)-Based Scheme

The pixel-value differencing (PVD)-based scheme is a scheme, in which the number of embedded bits is determined by the difference between a pixel and its neighbour. The larger is the difference, the larger the number of secret bits that can

be embedded which is the case where the pixel is in an edge area. In PVD method a range table has been designed with n contiguous ranges R_k (where k=1,2,...,n) where the range is 0 to 255. The lower and the upper bound are denoted as l_k and u_k respectively, then $R_k \in [l_k, u_k]$. The width of R_k is calculated as $w_k = u_k - l_k + 1$. w_k decides how many bits can be hidden in a pixel block. For security purpose R_k is kept as a variable, as a result, original range table is required to extract the embedded data. The embedding algorithm follows the following steps:

1.Calculate the difference value d of two consecutive pixels p_i and p_{i+1} for each block in the cover image. This is given by $d_i = |p_{i+1} - p_i|$.

2.Compute the optimal range where the difference lies in the range table by using d_i. This is calculated as $R_i = \min(u_k - d_i)$, where $u_k \geq d_i$ for all $1 \leq k \leq n$

3. Compute the number of bits (t) to be hidden in a pixel block this can be defined as $t = \log_2 w_i$. Where w_i is the width of the range in which the pixel difference di is belonging

4. Read t bits from binary secret data and convert it into its corresponding decimal value b. For instance if t=010, then b=2

5. Calculate the new difference value di' which is given by di'=li +b

6. Modify the values of pi and pi+1by the following formula:

$$(p_i', p_{i+1}') = \begin{cases} (p_i + (m/2), p_{i+1} - (m/2)), \text{ if } p_i \geq p_{i+1} \text{ and } d_i' > d_i \blacktriangleright \\ (p_i - (m/2), p_{i+1} + (m/2)), \text{ if } p_i < p_{i+1} \text{ and } d_i' > d_i \blacktriangleright\blacktriangleright\blacktriangleright\blacktriangleright \\ (p_i - (m/2), p_{i+1} + (m/2)), \text{ if } p_i \geq p_{i+1} \text{ and } d_i' \leq d_i \blacktriangleright\blacktriangleright\blacktriangleright\blacktriangleright \ldots\ldots (3.1) \\ (p_i + (m/2), p_{i+1} - (m/2)), \text{ if } p_i < p_{i+1} \text{ and } d_i' \leq d_i \blacktriangleright \end{cases}$$

Where $m = |d_i' - d_i|$. Now the pixel pair (p_i', p_{i+1}') is obtained.After embedding the secret data into pixel pair (p_i, p_{i+1}). Repeat step 1-6 until all secret data are embedded into the cover image. Hence the stego-image is obtained. When

extracting the hidden information from the stego-image, original range table is required. At first, partition the stego-image into pixel blocks, containing two consecutive non-overlapping pixels each. Calculate the difference value for each block as $di'=|p_i'-p_{i+1'}|$.Then find the optimum range R_i of d'_i. Then b' is obtained by subtracting l_i from d_i'. Convert b' into its corresponding binary of (t) bits, where $t=\log_2 w_i$. These t bits are the hidden secret data obtained from the pixel block $(p_i',p_{i+1'})$.

An illustration of the data embedding process is shown in Fig. 3.1 In the figure, the gray values of a sample two-pixel block are assumed to be (50;65). The difference value is 15, which is in the range of 8 through 23. The width of the range is $16=2^4$, which means that a difference value in the range can be used to embed four bits of secret data.

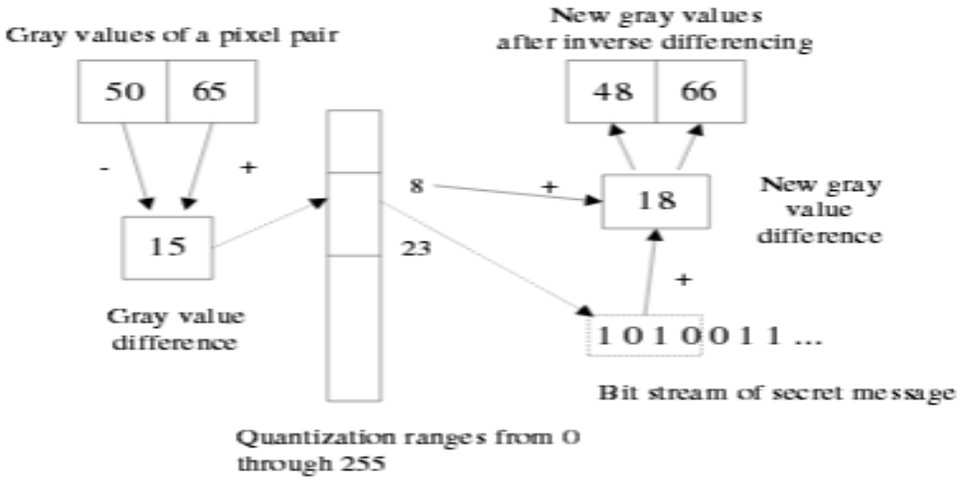


Figure 3.1: An illustration of the data embedding process.

Assume that the four leading bits of the secret data are 1010. The value of this bit stream is 10. It is added to the lower bound value 8 of the range to yield the new difference value 18. Finally, by Eq. (3.1) the values (48;66) are computed for use as the gray values in the stego-image. Note that 66 – 48 = 18.

3.2.2 Data Hiding Techniques in Frequency Domain

In frequency domain, pixel values are transformed to the frequency domain and then processing is applied on the transformed coefficients. There are different technique used in the frequency domain such as; discrete Fourier transforms (DFT). Discrete cosine transformation (DCT) , F3, F4 and F5 techniques. Two methods are used in this thesis to embed the information in images; the discrete cosine transformation (DCT) and the integer wavelet transforms.

3.2.2.1 Discrete Cosine Transform Technique (DCT)

DCT coefficients are used for JPEG compression. It separates the image into parts of differing importance. It transforms a signal or image from the spatial domain to the

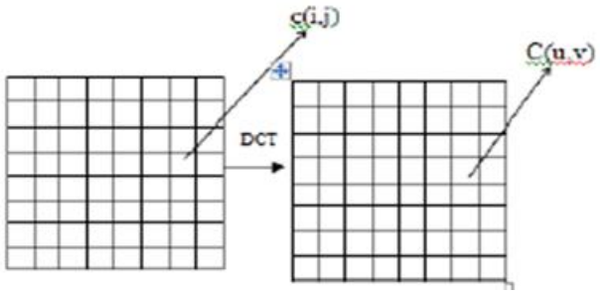


Figure (3.2): Discrete Cosine Transform of an image

frequency domain. It can separate the image into high, middle and low frequency components. In low frequency sub-band, much of the signal energy lies at low frequency which contains most important visual parts of the image while in high frequency sub-band, high frequency components of the image are usually removed through compression and noise attacks. So the secret message is embedded by modifying the coefficients of the middle frequency sub-band, so that the visibility

of the image will not be affected. The general equation for a 1D (N data items) DCT is defined by the following equation:

$$C(u) = a(u) \sum_{i=0}^{N-1} xi \cos\left(\frac{(2i+1)u\pi}{2N}\right) \dots\dots\dots (3.2)$$

Where:

N: the number of data items that will be turns in to frequency domain.

u= 0, 1, 2….. N-1

a (u): is a factor which equal to $1/\sqrt{2}$ if u is 0, and a(u) = 1 for all other cases.

The general equation for a 2D (*N* by *M* image) DCT is defined by the following equation:

$$C(u,v) = a(v) \sum_{i=0}^{N-1} [a(u) \sum_{i=0}^{N-1} xi \cos\left(\frac{(2i+1)u\pi}{2N}\right)] \times \cos\left(\frac{(2i+1)v\pi}{2N}\right) \dots\dots (3.3)$$

Where u, v = 0, 1, 2….N-1

a(x) = $1/\sqrt{2}$ if x is 0, and a(x) = 1 for all other cases.

Here, the input image is of size N X M. c (i, j) is the intensity of the pixel in row i and column j; C (u, v) is the DCT coefficient in row u and column v of the DCT matrix. DCT is used in steganography; Image is broken into 8×8 blocks of pixels. Working from left to right, top to bottom, DCT is applied to each block. Then the message is embedded in DCT coefficients, using the simple LSB substitution algorithm .For example, Fig.(3.3)(a) shows a block of 8 _ 8 pixels in an original cover image. JPEG uses DCT to transform the block into DCT coefficients. The

result of the DCT coefficients of the block is listed in Fig. (3.3)(b). After DC transformation, JPEG uses the standard quantization table to quantize the DCT coefficients. The result of the quantized DCT coefficients is listed in Fig. (3.3)(c). Jpeg–Jsteg embeds the secret messages in LSB of the quantized DCT coefficients whose values are not 0, 1, or -1. In this block, only two coefficients 79 and -2 can embed the secret message. Assume the secret message is 012. Then the result of this block after embedding will be listed

in Fig. (3.3)(d).The message capacity of Jpeg–Jsteg is limited. If there are many quantized coefficients equal to 0, 1, or -1, then the message capacity of Jpeg–Jsteg will be decreased. Besides, in DCT transformation, most important coefficients are located around the low-frequency part. Jpeg–Jsteg modifies the quantized DCT coefficients right in the low-frequency part. Therefore, the image quality of Jpeg–Jsteg is degraded, especially when the cover-image undergoes a high compression ratio.

(a)
$$\begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158 \end{bmatrix}$$

(b)
$$\begin{bmatrix} 1260 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & -1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix}$$

(c)
$$\begin{bmatrix} 79 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(d)
$$\begin{bmatrix} 78 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig (3.3). An example of embedding data in (DCT) coefficient. (a) A block of 8 _ 8 pixel values. (b) The DCT coefficients. (c) The quantized DCT coefficients. (d) The result of the coefficients after the embedding step.

3.2.2.2 Integer Wavelet Transform Technique (IWT)

Generally wavelet domain allows us to hide data in regions that the human visual system (HVS) is less sensitive to, such as the high resolution detail bands (HL, LH and HH), Hiding data in these regions increases the robustness while maintaining good visual quality. Integer wavelet transform maps an integer data set into another integer data set. In discrete wavelet transform, the used wavelet filters have floating point coefficients so that when data is hidden in their coefficients any truncations of the floating point values of the pixels that should be integers may cause the loss of the hidden information which may lead to the failure of the data hiding system . To avoid problems of floating point precision of the wavelet filters when the input data is integer as in digital images, the output data will no longer be

integer which doesn't allow perfect reconstruction of the input image and in this case there will be no loss of information through forward and inverse transform . Due to the mentioned difference between integer wavelet transform (IWT) and discrete wavelet transform (DWT) the LL sub band in the case of IWT appears to be a close copy with smaller scale of the original image while in the case of DWT the resulting LL sub band is distorted. Lifting schemes is one of many techniques that can be used to perform integer wavelet transform it is also the scheme used in this research. The following procedure shows the lifting schemes to obtain integer wavelet transform by using simple truncation and without losing inevitability.

At first, the simplest Haar2D-Discrete wavelet transform consists of two operations: a horizontal operation and a Vertical operation.

Detailed procedures of a 2-D Haar-DWT are described in Figure 3.4:

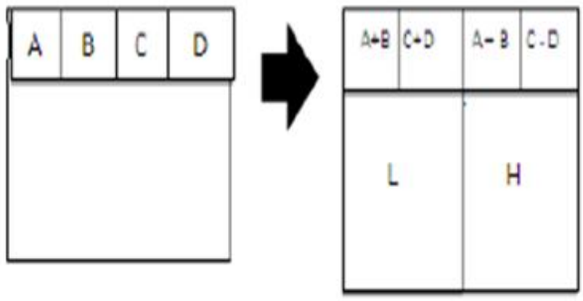


Figure 3.4 Horizontal Operations on the First Row

Step 1: At first, the pixels are scanned from left to right in horizontal direction. Then, the addition and subtraction operations are performed on the neighboring pixels. The sum on the left and the difference on the right are stored as illustrated in Figure3.4. This operation is repeated until all the rows are processed. The pixel sums represent the low frequency part (denoted as symbol L) while the pixel differences represent the high frequency part of the original image (denoted as symbol H), and this obtained using the following equation:

$$x_{n-1,i} = \frac{x_{n,2i} + x_{n,2i+1}}{2} \quad d_{n-1,i} = \frac{x_{n,2i} - x_{n,2i+1}}{2} \text{.....(3.4)}$$

Step 2:The pixels are scanned from top to bottom in vertical direction. The addition and subtraction operations are performed on neighbouring pixels as mentioned in step1. Then the sum on the top and the difference on the bottom are stored as illustrated in Figure 3.5. This operation is repeated until all the columns are processed. Finally,a 4 sub-bands will be obtained denoted as LL, HL, LH, and HH respectively.

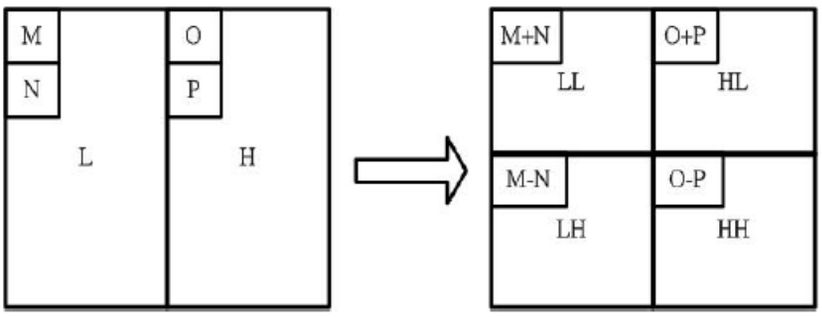


Figure 3.5. The vertical operation

The LL sub-band is the low frequency portion and hence looks very similar to the original image. The whole procedure described is called the first-order 2-DHaar-DWT. Secondly,If the original image (I) is X pixels high and Y pixels wide, the level of each of the pixel at (i,j) is denoted by $I_{i,j}$.

The IWT coefficients are given by

$LL_{i,j}=\text{floor}(I_{2i,2j} + I_{2i+1,2j})/2$ ... ... (3.5)

$HL_{i,j}=I_{2i+1,2j}-I_{2i,2j}$.....(3.6)

$LH_{i,j}=I_{2i,2j+1}-I_{2i,2j}$.....(3.7)

$HH_{i,j}=I_{2i+1,2j+1}-I_{2i,2j}$.....(3.8)

Where,$1\leq i \leq X/2$, $1\leq j\leq Y/2$

Chapter Four

Simulation and Results

4.1 Preview

Steganography is the art and science of invisible communication. It is accomplished by hiding information in otherso no one can sense the presence of data or secret message. In this thesis the data is hidden in an image, which is known as image steganography. The image where the data will be hidden in is called the cover image. The image with the hidden data will be called a stego image. Image steganography have many techniques. In this thesis four methods where used; two in the spatial domain and two in the frequency domain. A comparison between them in terms of PSNR and the restored message was performed. A Graphical user Interface was designed with the algorithm that gave the best result to be used in internet applications.

4.2 Parameters Assumptions

In this thesis two types of images where used; BMP and JPEG as shown in Table 4.1.Matlab program was used to test the different techniques.

Table 4.1: Parameters Assumptions

Type of Image	**Color of Image**	**Parameters of Image**
BMP	Colored Image	(392x392) dimensions bit depth of 24 bit and size 450KB.
	Gray Level Image	(392x392) dimensions bit depth of 8 bit and size 151KB.
JPEG	Colored Image	(392x392) dimensions, bit depth of 8 bit, and size 13.2KB.
		(1200x1600) dimensions, bit depth of 24 bit, and size 110KB.
	Gray Level Image	(392x392) dimension, bit depth of 8 bit, and size 11,4KB.
		(1200x1600) dimensions, bit depth of 8 bit, and size 108KB.

4.3 System Flowcharts

In this thesis different steganography techniques were used to embed the transmitted message and then to extract it at the end user.

4.3.1 Least Significant Bit Flowchart

1. Embedding part:

Least significant bit steganography technique, work with the spatial domain (pixel domain) as shown in Figure 4.1. It embeds a bit of data in the least significant bit of each cover image pixel. If the LSB pixel is equal to the message data bit, then the new image (stego image) pixel will be as same as the cover one. But if they are not equal, then there are two cases to be considered:

- If the message bit is (1) and the image pixel is even, then the stego image pixel will be the cover pixel mines one.
- If the message bit is (0) and the image pixel is odd, then the stego image pixel will be the cover pixel plus one.

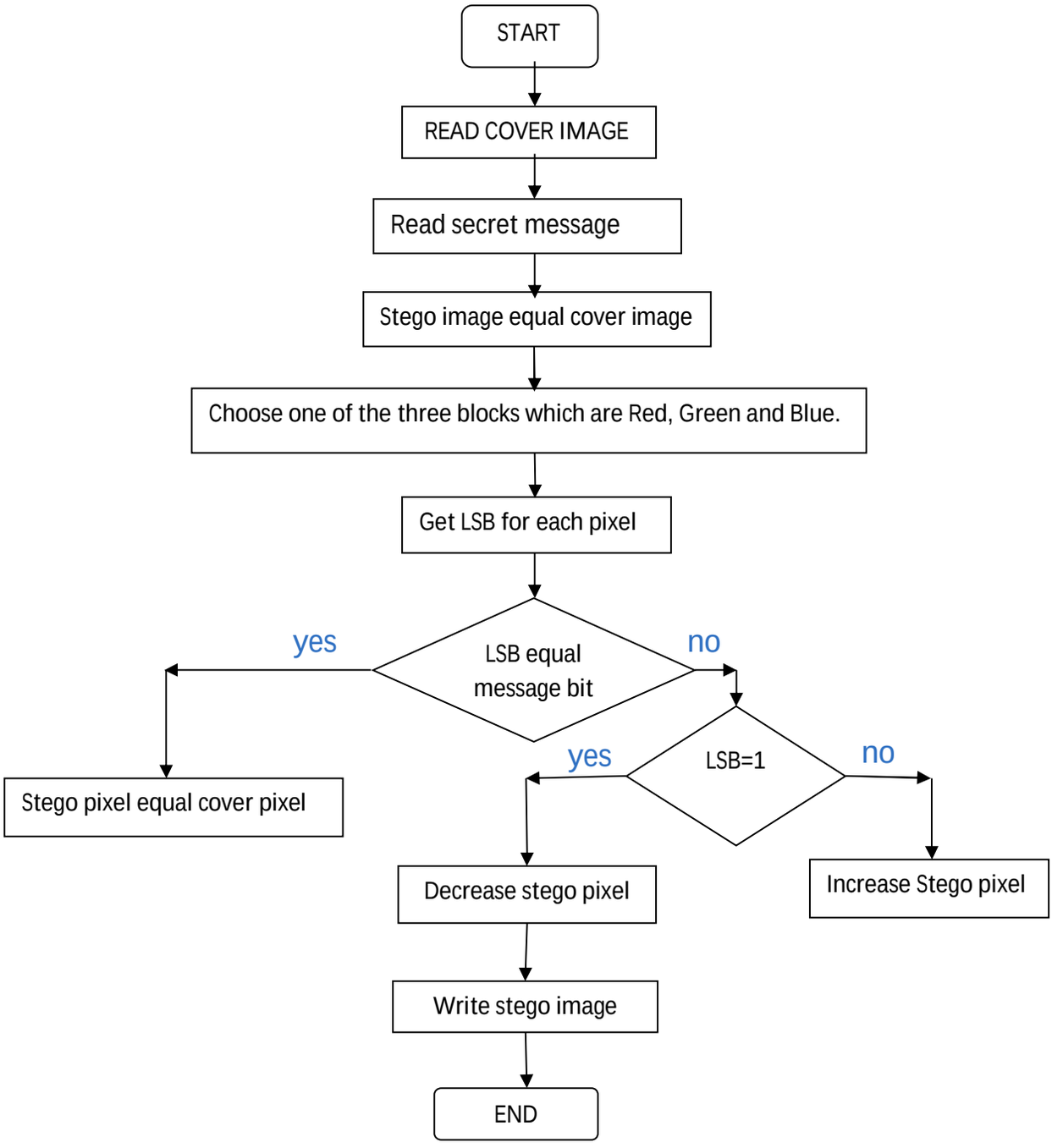


Figure (4.1): LSB Embedding Flowchart

2. Extracting part:

To recover the original message, the extractor reads the stego image and extracts the LSB from each pixel and converts it into characters as shown in Figure 4.2.

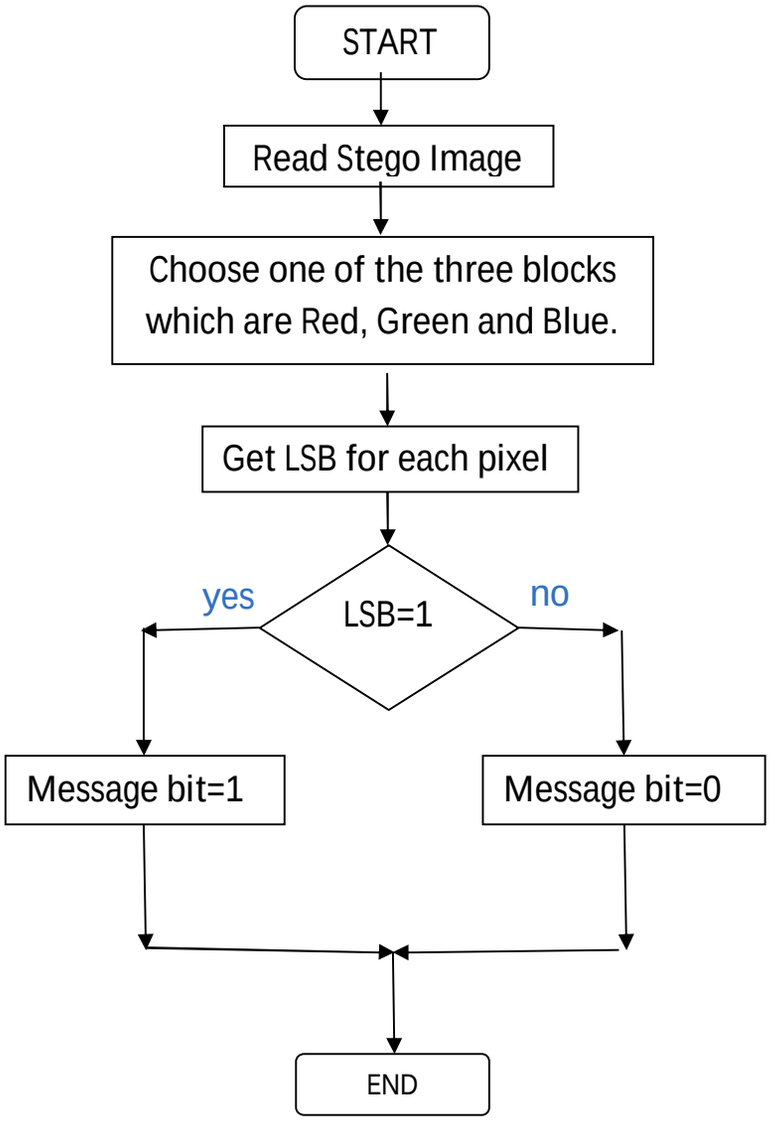


Figure (4.2): LSB Extracting Flowchart

4.3.2 Pixel Value Differencing (PVD) Flowchart

1. Embedding part:

To embed data using PVD steganography technique, firstly, the image is broken into two pixels non overlapping blocks. The difference (d_i) between each block pixels should be calculated, and then compute the optimal range where the difference lies in the range table, then Compute the number of bits (t) to be hidden in a pixel block, then Read t bits from binary secret data and convert it into its corresponding decimal value b, and finally, modify difference value d_i' and then get the modification factor w which is equal to the difference between the old and the new difference over tow values then for the stego image block:

- If the block first pixel is less than the second pixel and d_i' is less than d_i
  - The first pixel will be the old one plus w
  - The second pixel will be the old one minus w

- If the block first pixel is less than the second pixel and d_i' is greater than d_i
  - The first pixel will be the old one minus w
  - The second pixel will be the old one plus w

- If the block first pixel is greater than the second pixel and d_i' is greater than d_i
  - The first pixel will be the old one plus w
  - The second pixel will be the old one minus w

- If the block first pixel is greater than the second pixel and d_i' is less than d_i
  - The first pixel will be the old one minus w
  - The second pixel will be the old one plus w

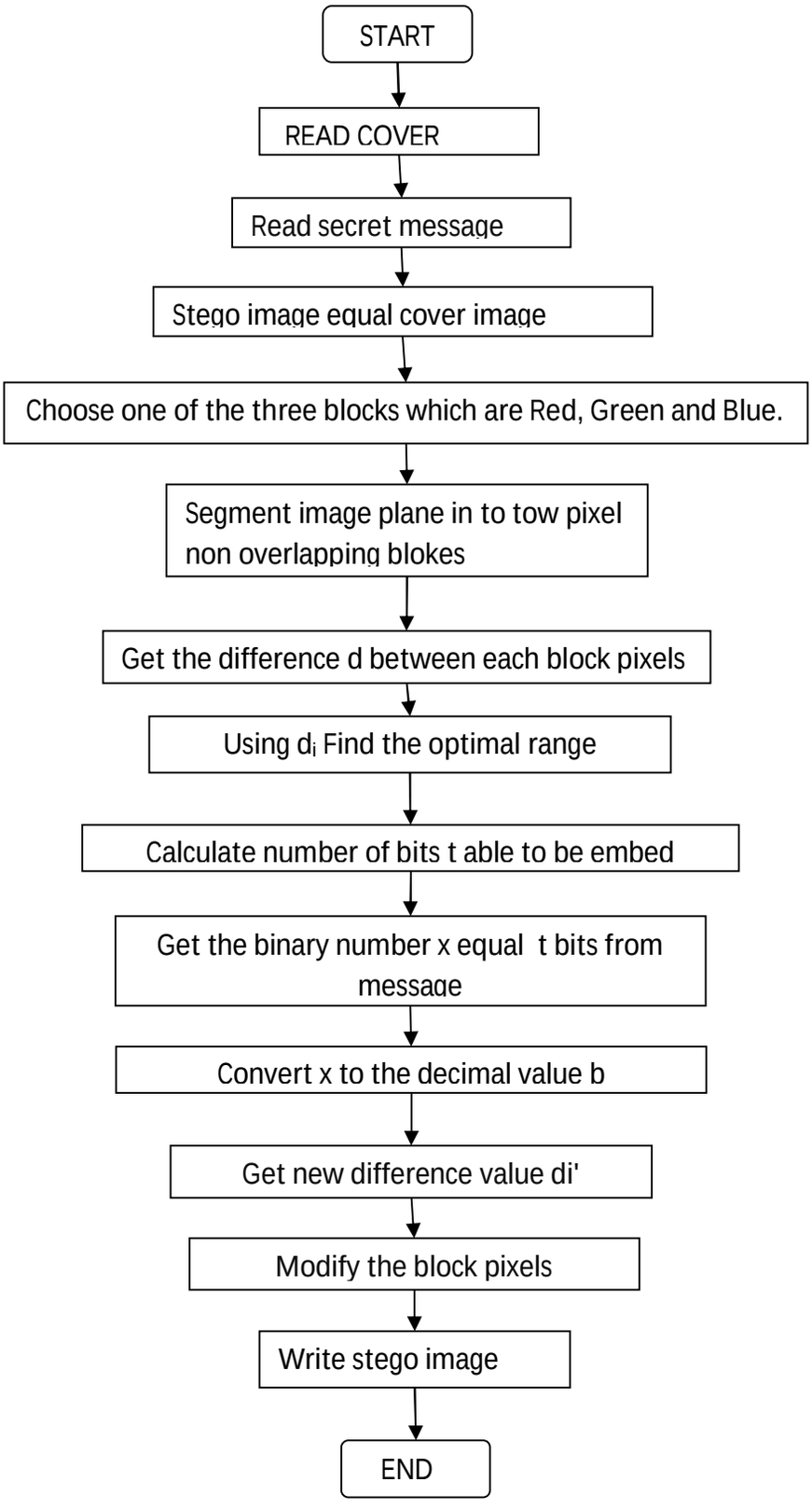


Figure (4.3): PVD embedding flow chart

2. Extracting Part:

To extract embedded data using PVD steganography technique, firstly, the image is broken into two pixels non overlapping blocks. The difference (d_i) between each block pixels should be calculated, and then compute the optimal range where the difference lies in the range table, then obtain the decimal value b' by subtracting the range lower value from d_i', and then convert it into its corresponding binary of (t) bits, these t bits are the hidden secret data obtained from an adjacent pixels.

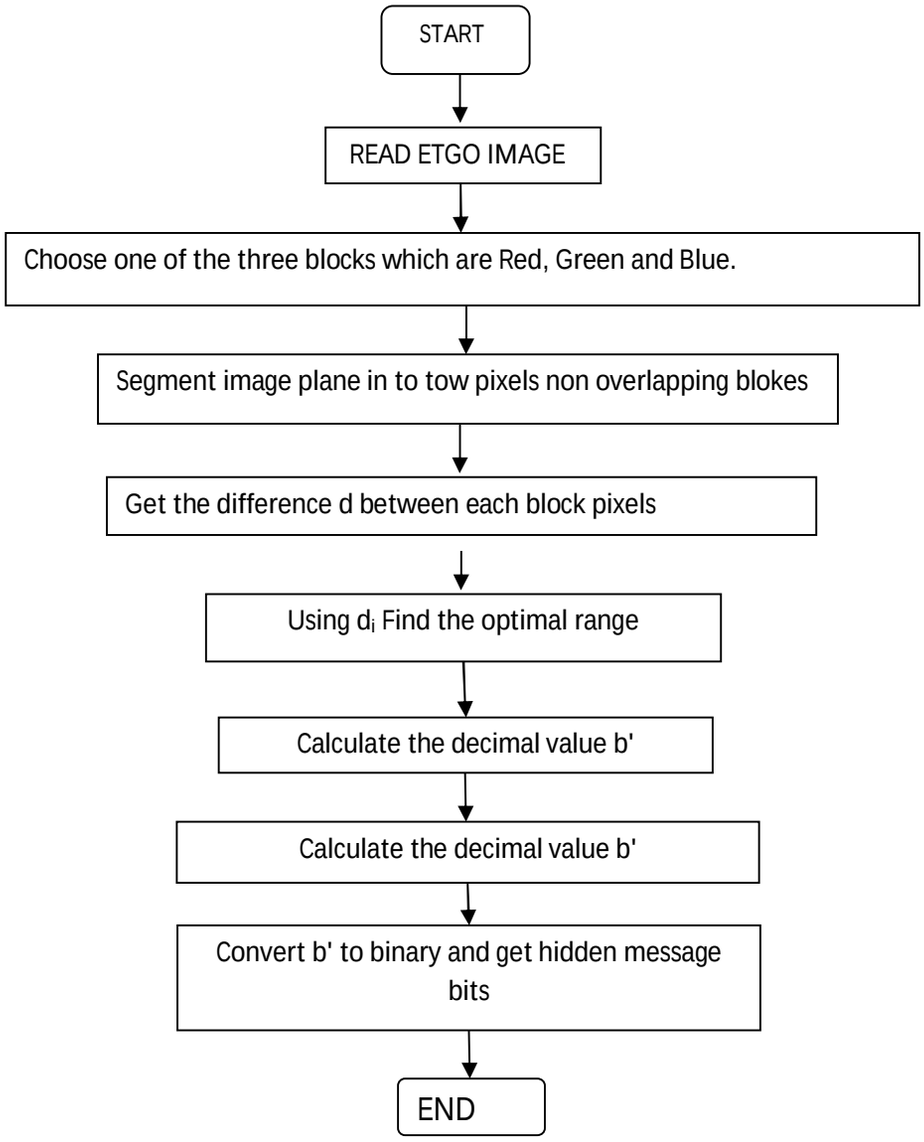


Figure (4.4): PVD extracting flow chart

4.3.3 Discreet Cosine Transforms

1. Embedding Part:

In discrete cosine transform to Embed data, the image is first broken into 8x8 pixel blocks, each block must modify before having the dct transform, then obtain the dct transform, then the obtained dct coefficients compressed using the standard quantization table, then for all block DCT compressed coefficient:

- If the compressed DCT coefficient is non 0, 1 or -1
  - Embed bit of data using the simple LSB embedding technique

Then decompress the image block, de modify the image block and take the inverse DCT. Then save the new block as a stego image's block.

2. **DCT Extracting Part**:

In the receiving side , the image is first broken into 8x8 pixel blocks, each block must modify before having the dct transform, then obtain the dct transform, then the obtained dct coefficients compressed using the standard quantization table, then for all block DCT compressed coefficient:

If the compressed DCT coefficient is non 0, 1 or -1

- Extract bit of data using the simple LSB embedding technique

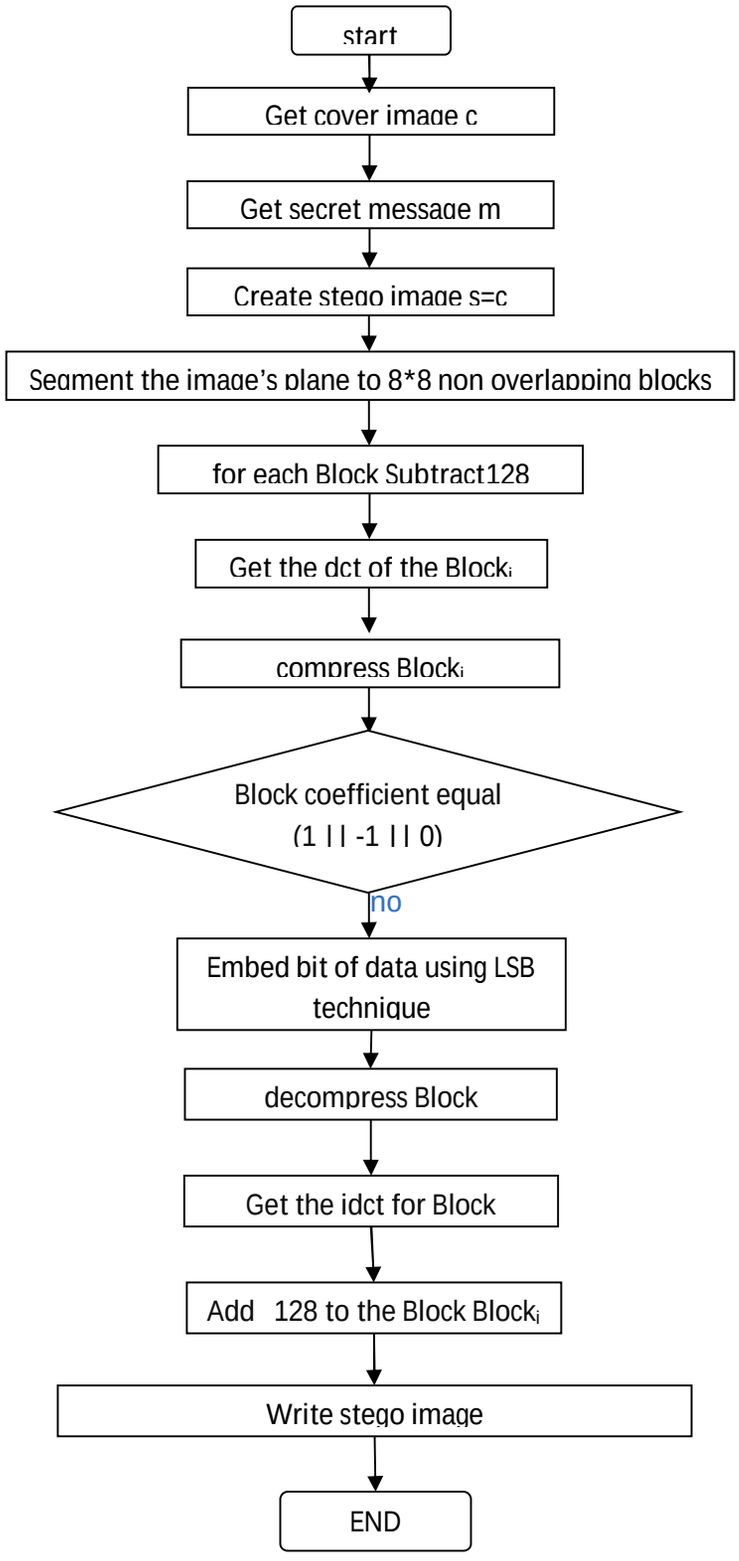


Figure (4.5): DCT Embedding Flow Chart

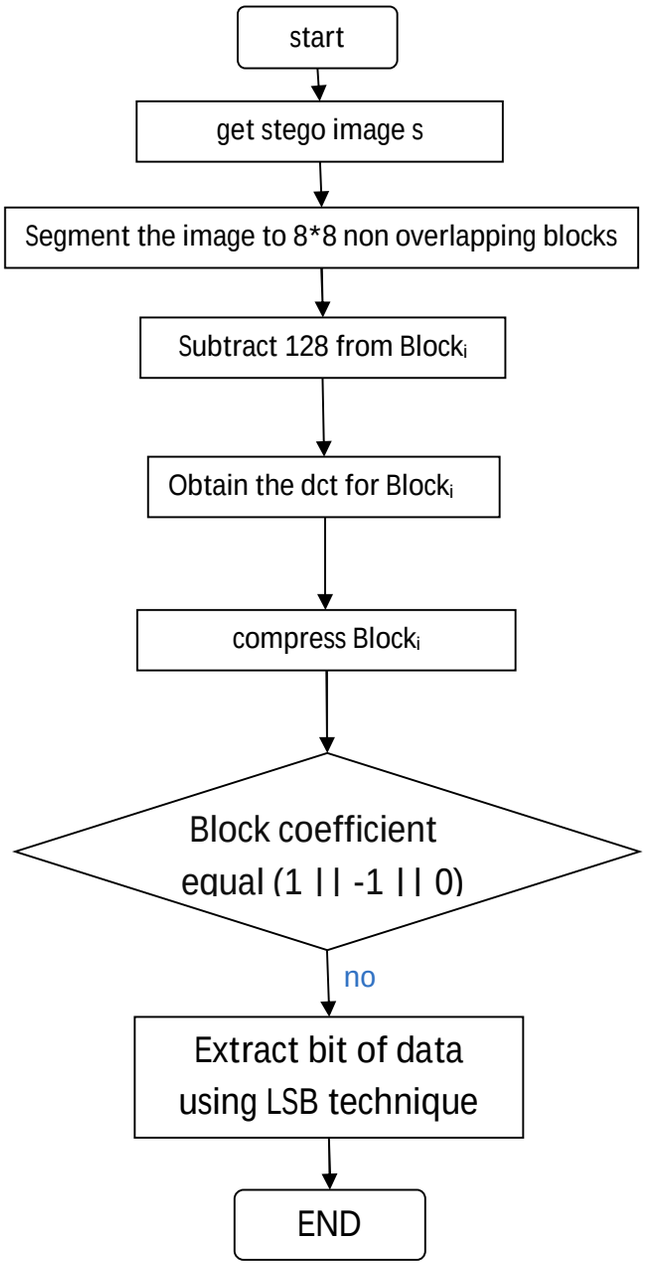


Figure (4.6): DCT extracting flow chart

4.3.4 Integer Wavelet Transforms (IWT) Flowcharts:

1. Embedding Part:

In integer wavelet transform steganography technique also to embed data in image, the image is modified then broken into 8x8 non overlapping blocks. Then we must get the third level (IWT) to each block, and then for each block:

- Embed bit of data to the LSB of each3_level (IWT) coefficient.

Then the inverse (IWT) transform is obtained and the image was saved as the stego image.

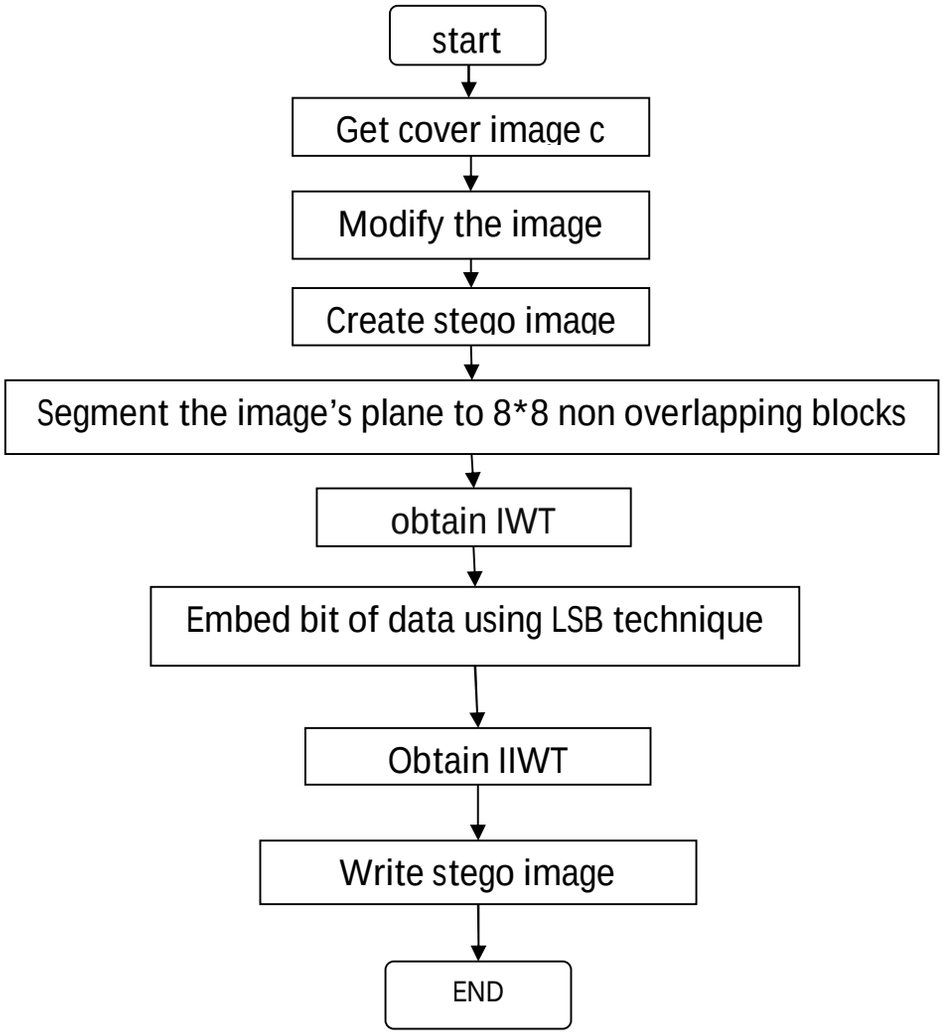


Figure (4.7): IWT embedding flow chart

2. Extracting Part:

In the receive side also to extract data from image, the image is modified then broken into 8x8 non overlapping blocks. Then we must get the third level (IWT) to each block, and then for each block:

- Extract bit of data to the LSB of each3_level (IWT) coefficient

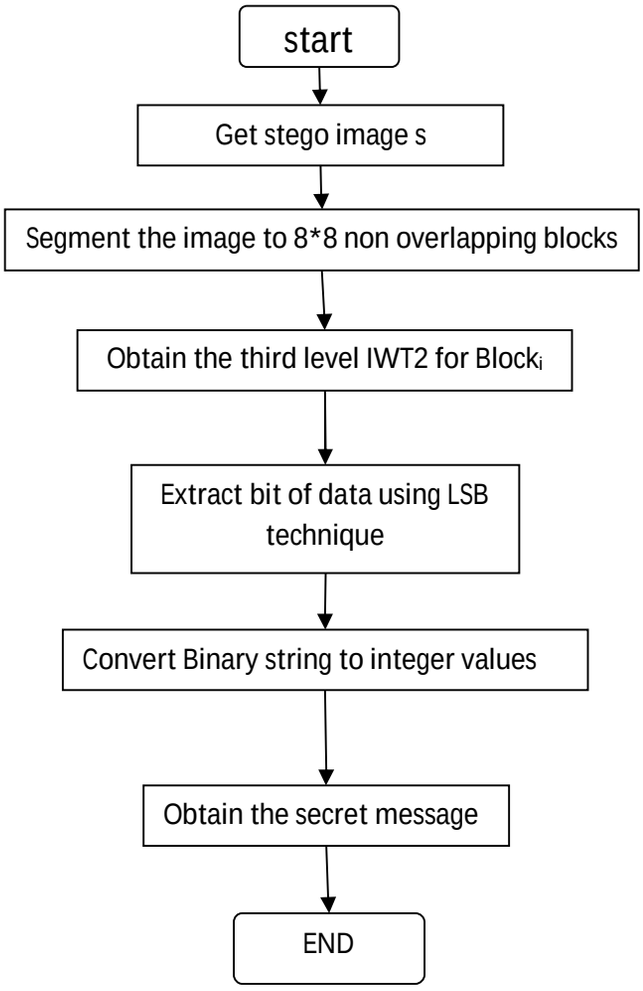


Figure (4.8): IWT extracting flow chart

4.6 Graphical User Interface Flowchart:

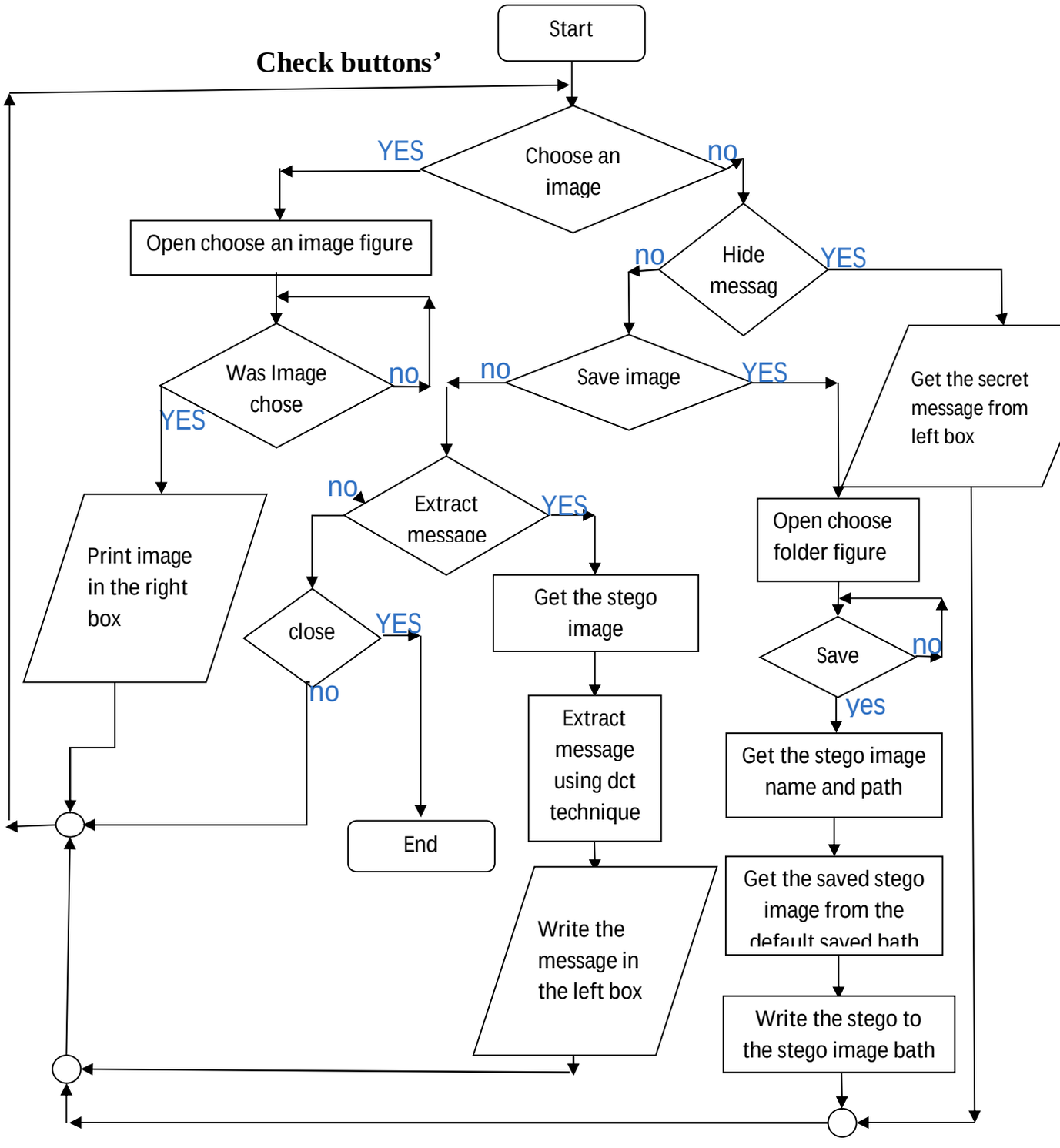


Figure (4.9): Graphical User Interface (GUI) flowchart

4.4 Results and Discussion

Here were the resultant images,peak signal to noise ratios (PSNR), and extracted messages, after embedding: (aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa)
as a data in both used image types, using the four different techniques.

4.5.1resultant images

1. BMP Input Image:

1. Color Image

(a)




(b) (c) (d) (e)

Figure (4.10) embedding data in JPEG color image, a) cover image , the other are stego images using the techniques b) LSB c) PVD d) DCT e) IWT

2. Gray Image

(a)

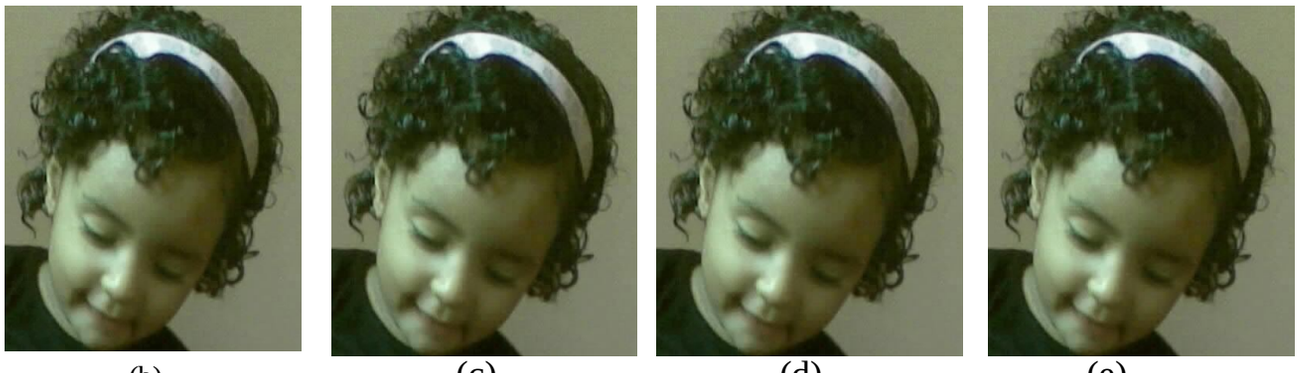
(b) (b) (c) (d)

Figure (4.11) embedding data in the grey level image a) cover image, the other are stego images using the techniques b) LSB c) PVD d) DCT e) IWT

2. JPEG Input Image:

1. Color Image

(a)



(b) (c) (d) (e)

Figure (4.12) embedding data in JPEG color image, a) cover image, the other are stego images using the techniques b) LSB c) PVD d) DCT e) IWT

2. Gray Image

(a)




(a) (b) (c) (d)

Figure (4.13) embedding data in JPEG gray level image, a) cover image, the other are stego images using the techniques b) LSB c) PVD d) DCT e) IWT

4.4.2 PSNR and recovered messages:

Here the calculated PSNR's between the cover and the stego images after embedding (aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa) As a secret message in the cover image will viewed and also the restored message in the receiving part. In the color image the message was embedded in all images color planes to see whether some planes have best results than others. The tables bellow shows these Parameters analysis after applying steganography techniques.

Table (4.2): Parameter analysis of steganography methods when using bmp image as a cover media:

ALGORITHM	RESTORED MESSAGE	PSNR IN DB
LSB IN R	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	54.8931
LSB IN G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	54.5957
LSB IN B	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	54.8590
LSB WITH GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	54.1414
PVD IN R	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	Inf
PVD IN G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`	51.7981
PVD IN B	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaPppppppp	52.7297
PVD WITH GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa`	52.1110
IWT IN R	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	36.8656
IWT IN G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	37.0415
IWT IN B	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	36.7972
IWT WITH GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	31.7322
DCT IN R	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	29.8989
DCT IN G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	29.2718
DCT IN B	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	30.6636
DCT WITH GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	29.3869

- In LSB error will not occur even the message is so long
- In PVD the error may occur with long message
- In integer wavelet with (color or gray image) the error will not occur even the message is so long
- In DCT with (color or gray image) the error will not occur even the message is so long

Table (4.3): Parameter analysis of steganography methods when using JEPG image as a cover media:

ALGORITHM	RESTORED MESSAGE	PSNR IN DB
LSB IN R	·Ùfbóþp‡·" @ƒ/«çpCÈ´ á ¬+j“üœ-‡p |Î,~ ÿ Ï…ÃìÇÀÌü½	55.2123
LSB IN G	H&™ óÿsr¹Yýó7WØðþWAKQËÛSˆ•µ’Rxðÿ ÎSiçùÿœ…<ç2Ç À3————————————Bÿ	54.6601
LSB IN B	H&fb ÿµ·"ÿ¿ƒ´«ép·ÿ´¯-NÐ%)_-‡ÿƒ1S è ÿ¸ÇÃÇÿ3————————————½	54.8590
LSB WITH GRAY	·“™Móÿ 9Y ý|ÆTLØAKPD±ºÚÖu ———————————— Òà ÿƒ1,–çþÿ ˜Ç<çìÇÌ	54.1127
PVD IN R	Empty	50.5029
PVD IN G	Empty	50.5029
PVD IN B	Empty	52.5466
PVD WITH GRAY	Empty	55.3621
IWT IN R	0saéÑà%b…ÿÃuce! é-i`k%dåe`aãÇaàSqB"Õã@AïpèmmÀ g	36.9703
IWT IN G	p¡caéñ!ãà!ba`a#áaká 1a#å`¨<!is÷!á£g`íbEixmijmímcáÀi©	37.1139
IWT IN B	Ññq!òÃÌ9 aÒI4Su«áãçà£ådï{Mxwu|2ãŸAÙv@ëMfèég1£S›	36.8656
IWT WITH GRAY	aa!aaaaaaaaaa!iaaaacaaa!aaaaqaaaaaaa`aaaaaaaaaaÁaaáaa	31.7326
DCT IN R	aaAaaaaaaaaaaaaaaaaáa`a!aaaaaaaaa!aaaaaaaaaaaa!aaaaai	29.4836
DCT IN G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	29.3205
DCT IN B	aAe0°XXXXxYHXXXX,,,,,,†r————————————	30.5864
DCT WITH GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	29.1895

Now, in the next table there is the parameter analysis after embedding the same data in another jpeg image, to show that some images have better results than other.

Table (4.4): Parameter analysis of steganography methods when using another JEPG image as a cover media:

ALGORITHM	RESTORED MESSAGE	PSNR IN DB
LSB R	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿø?ÿà 0Œÿ/ÿéÿ	63.3573
LSB G	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ Àþÿÿÿÿ ÿÿãìÿýúÿ	62.9017
LSB B	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ ?ÿà 0Œÿ/ÿíÿ	62.9190

LSB GRAY	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÌŒ ÿÏ 3¬,)ÿ	63.2623
PVD R	EMPTY	55.1346
PVD G	EMPTY	55.1346
PVD B	EMPTY	55.1346
PVD GRAY	SO LONG STRING OF LETERS	35.7776
IWT R	aaaaeÕA)´Caaaaaaaaaaaa AaçP¼0aaaaaaaaiaaaa ————————————ñp)X‰Aaaaa	52.8160
IWT G	aaaaaaE`¡@ãaaaaaaaaaaaaÉGAS„Aaaaaaaaaaaap U+-AAaaaa	45.1631
IWT B	aaaaay}b¡aaaaaaaaaaaaHëa!aaaaaaaeaaaa9±f)`Aaaaaa	45.1817
IWT GRAY	aaaaaaaaaaaaaaaaaaaaaaaaAaaámaaaaaaaaaaaaaaaaaaaaaaaa	20.9571
DCT R	ÿÿÿÿñaaaaaab,ÂÂÿÿÿÿÿÿÿÿÿÿÂÃÂÂÂÂÅ …… ……å•„	36.5196
DCT G	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaÂÂÂÃ	35.4134
DCT B	ÂÂÂÂÂÁ=======>====BÂÂÂÂÂÍ==== ÂÊÂ ÂÂÀ	35.4143
DCT GRAY	aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	36.4484

- The message which is send through jpeg image using LSB or PVD will be extremely lost.

- The message which is sent using IWT algorithm with color or gray level image will be restored with some error, depend upon the image and compression quality.

- The message which is sent using DCT embedding algorithm with color or gray level image will be restored with some error, depend upon the image and compression quality .

- Even though IWT give better PSNR over DCT, the message restored from DCT stego image always better than that restored from IWT stego.

4.5 Graphical User Interface (GUI)

A graphical user interface (GUI) for the best steganography technique was designed using Matlab. It was designed for easy use purposes. It allows the user to choose the cover image from a specific folder, write the message to be sent, hide message in the cover image, and save the stego image as a new image to be send. or read the sent image from a specific folder and then extract the sent message. Figure (4.14) shows the GUI of the system, using DCT technique with gray image.

1. When the user click choose an image pushbutton, a new window with list of folders will appear to choose an image from a specific folder. The following figure (4.15) and figure (4.16) show the new window.

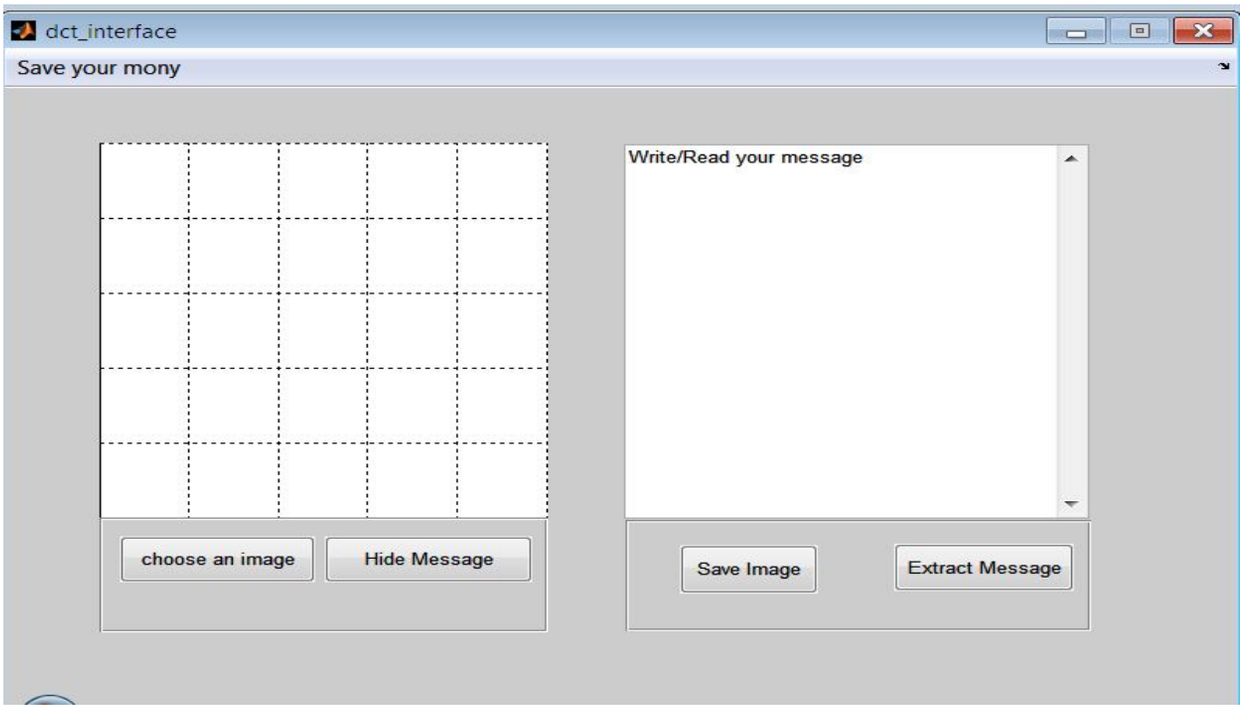


Figure (4.14): System GUI

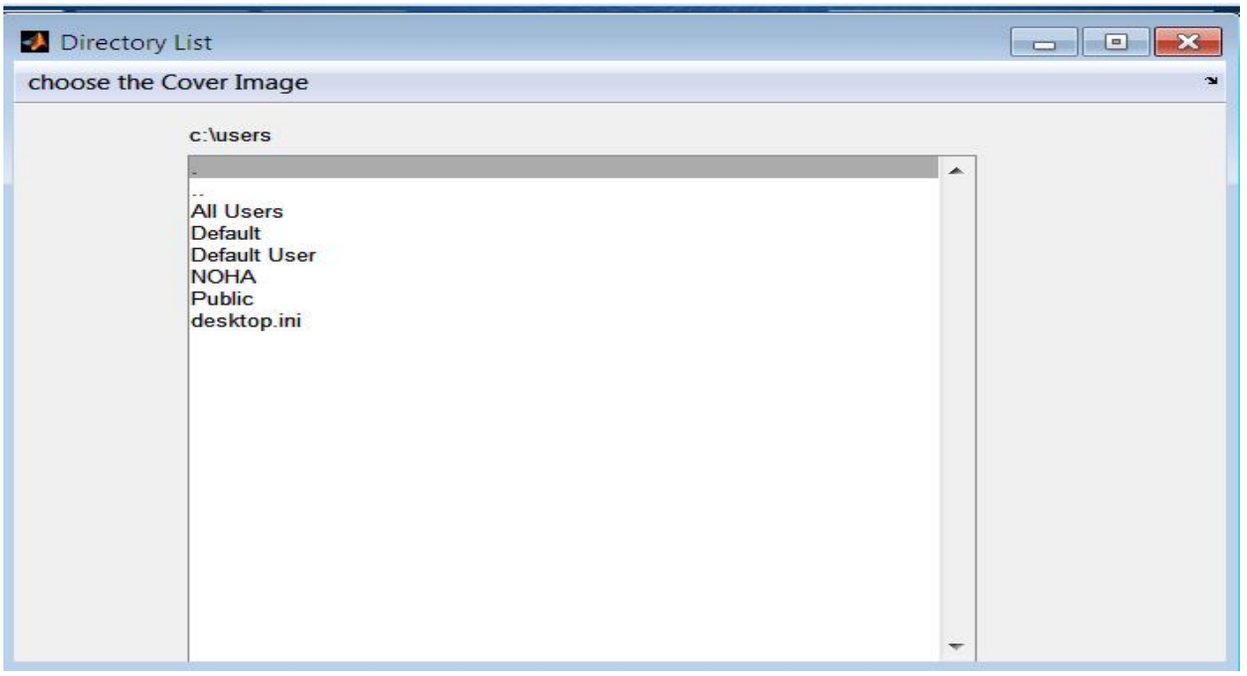


Figure (4.15): the new window with list of folders

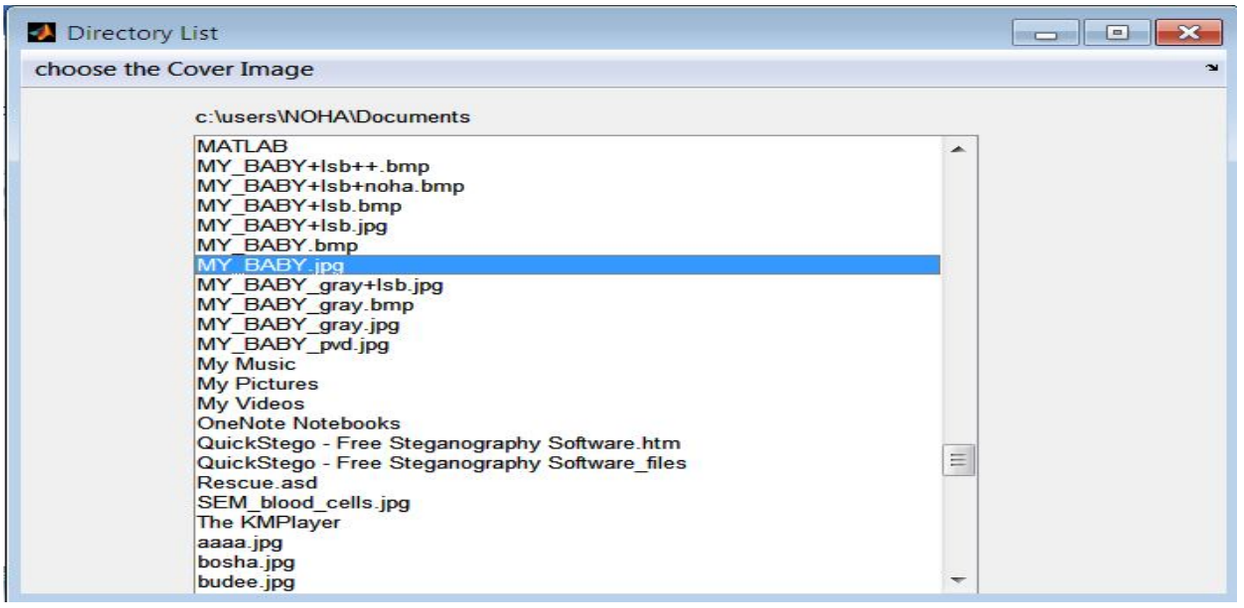


Figure (4.16): new window with list of image files

2. After an image chosen, it will be displayed in the first figure window as in figure (4.17),

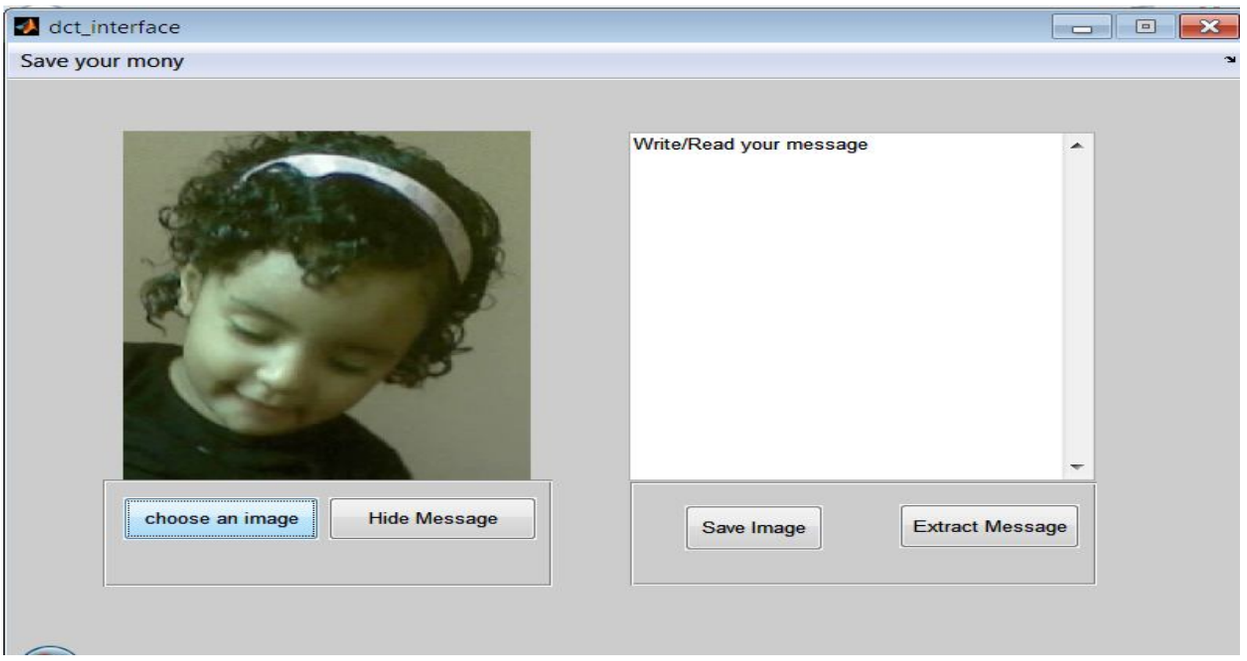


Figure (4.17): displaying the chosen image in the first figure window

3. Now if the user write him message in the right box then he can hide message in the cover image using hide message pushbutton as in the figure (4.18).

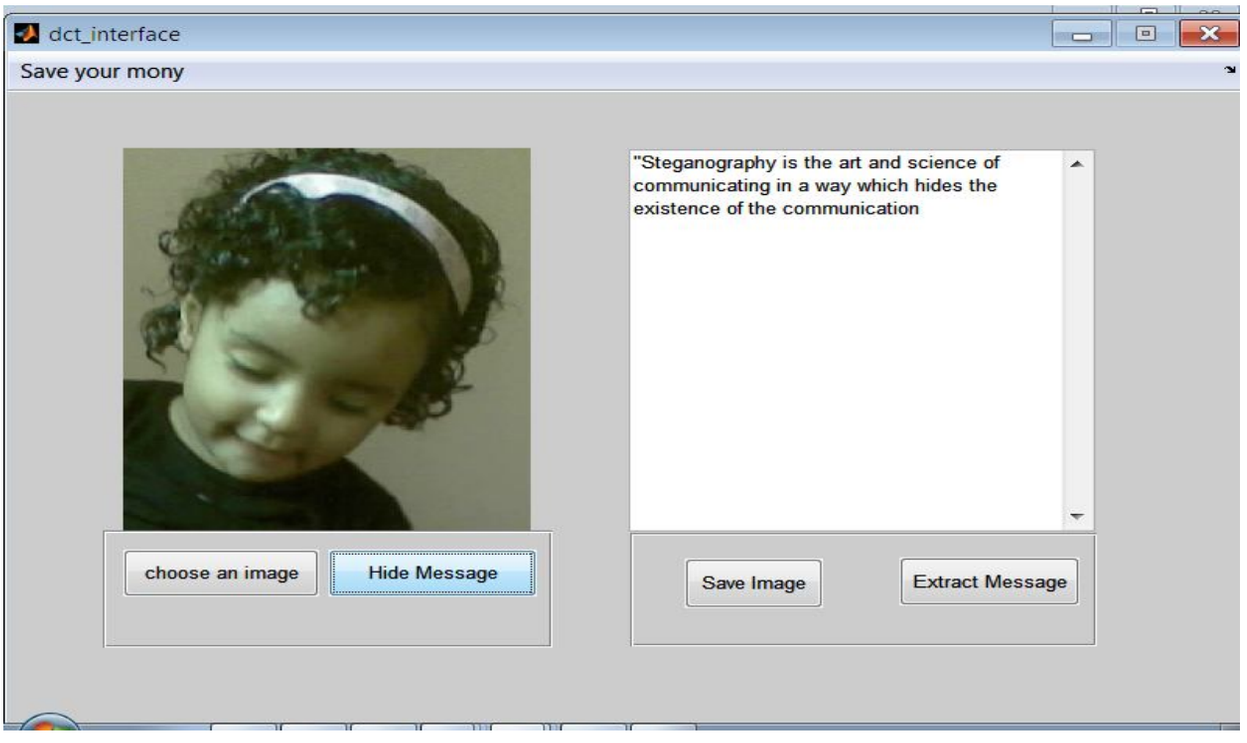


Figure (4.18): writing and hiding the secret message

4.then when the user click on Save Image pushbutton, a new figure with folder list will appear to choose the destination file folder, the window also contain an editing box to write the stego image name, and pushbutton to save the image, as in figure (4.19).

5. After the user clicking the save pushbutton the stego image will be saved in the in the highlights file folder and ready to be sent.

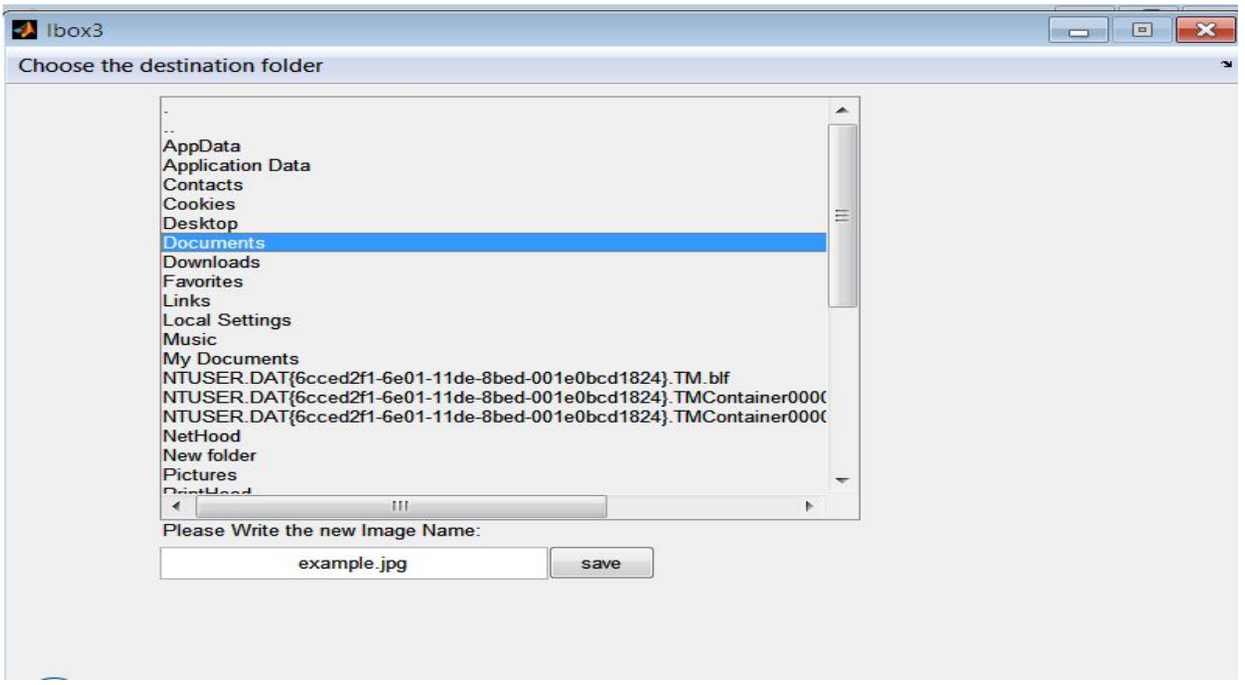


Figure (4.19): a new figure with folder list to save the stego image

6. now if the user click on choose an image pushbutton and chooses the saved stego image as in figure (4.20), and then click on extract message pushbutton, he will obtain the hidden message as in figure (4.21),

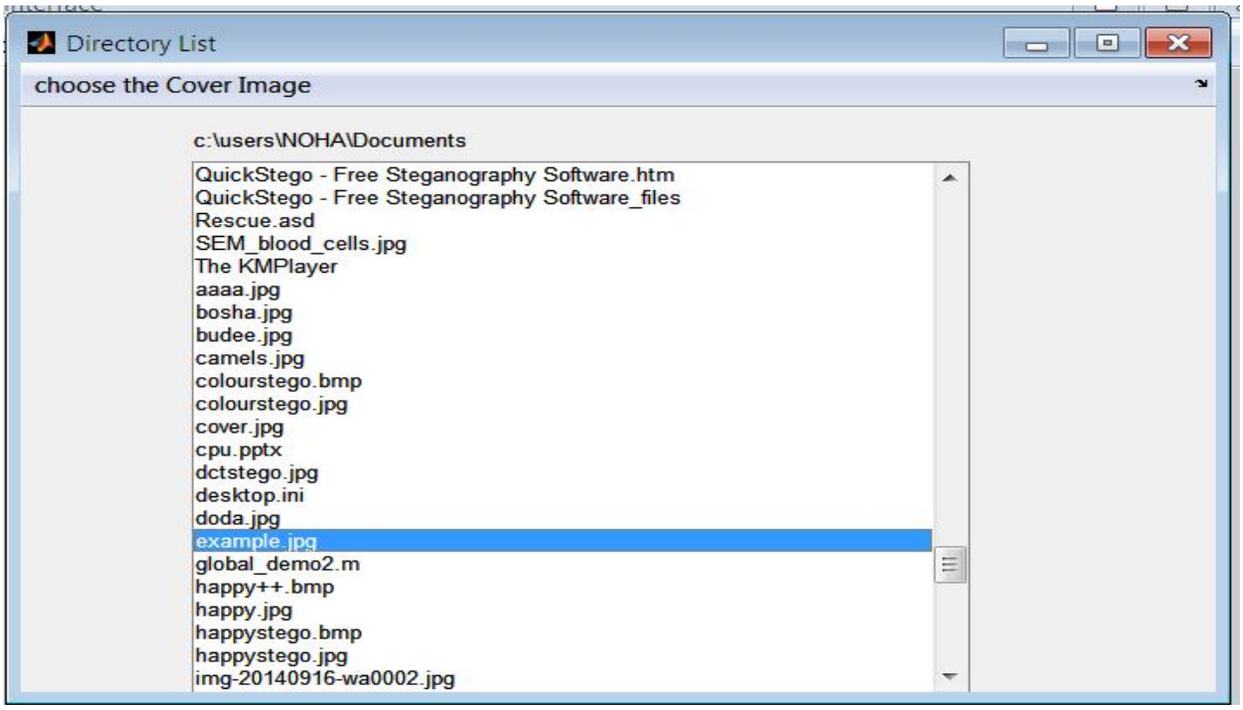


Figure (4.20): choosing the stego image

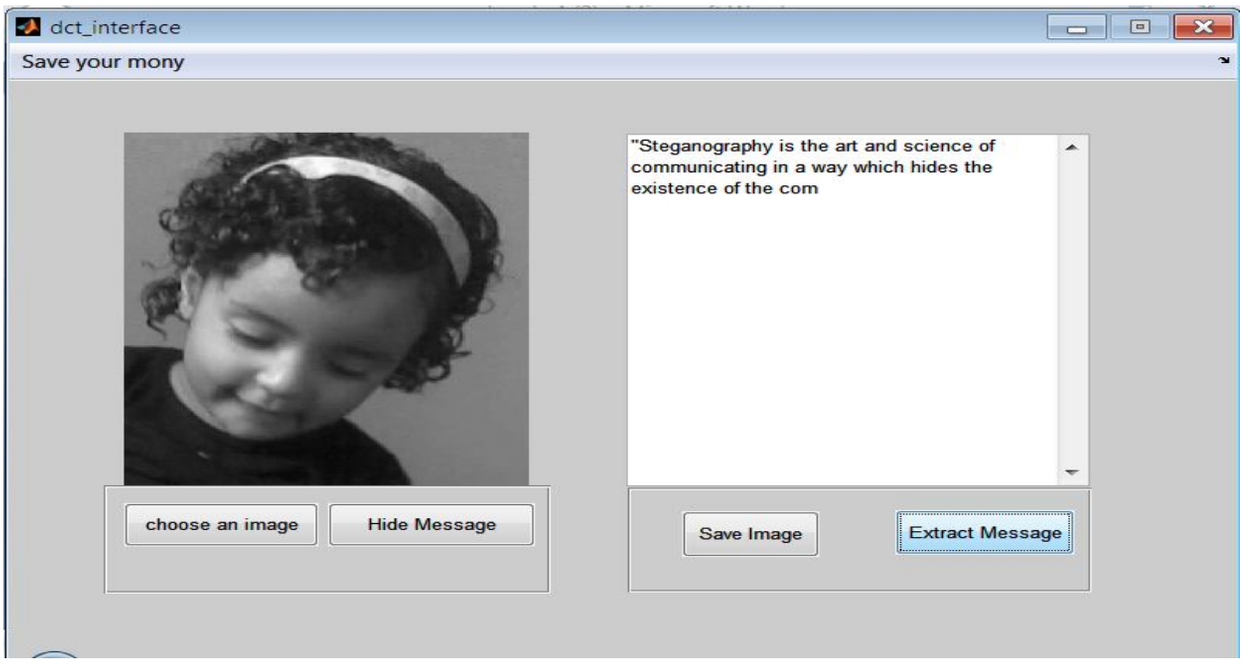


Figure (4.21): extracting secret data from the stego imag**e**

Chapter Five

Conclusion and Recommendations

5.1 Conclusion

Image steganography hide data inside image so no one can sense the presence of data. The secret data is embedded within an image which is called cover image, and the image with data inside it is called a stego image.

Since the internet is an open media, the information sent through it faces security issues. This problem could be solved by image steganography.

Image steganography have many techniques. In this thesis, four methods where used; two in the image spatial domain (LSB and PVD) and two in the frequency domain (IWT and DCT). A comparison between them in terms of PSNR and the restored image was performed. The results showed that the Integer Wavelet transform had the best results when used with the bmp images because of its robustness against image manipulations. While, the Discrete Cosine Transform Technique (DCT) had the best results when the JPEG images were used as a cover media, this resulted in having the best restored message. A graphical user interface was designed using Matlab using the DCT technique to be used in internet application

5.2 Recommendations

A lot of work has been done in this thesis but still there is room for improvement;

- In this thesis steganography was used. It is recommended to use cryptograph with steganography for better security.
- In this thesis the message was sent without authentication key. It is recommended to use an authentication key.

References

[1] T. Sharp, “An implementation of key-based digital signal Steganography”, in Proc. Information Hiding Workshop,Springer LNCS 2137, pp. 13–26, 2001.

[2]., Mukesh Kumar Soneand ,Gaurav Agarwal3.''Securing Data in Fiber Optics through Steganography''International Journal of Advanced Research in Computer Science and Software Engineering. Volume 4, Issue 6, June 2014.

[3]. P.Thiyagarajan,G.Aghila. V. ParasannaVenkatesan. From nagar.''Qualitative analysis of Dynamic Pattern based steganography Algorithm in providing E-Banking Security''.

[4].Yair Wiseman, ''Steganography Based Seaport Security Communication System''. in Advanced Science and Technology Letters Vol.46 (Signal Processing 2014).

[5]. TusharBhivgade ,MithileshBhusari , Ajay Kuthe , BhavnaJiddewar ,Prof. PoojaDubey. From india.''Multi-factor Authentication in Banking Sector'', in International Journal of Advanced Research in Computer Science and Information Technologies, Vol. 5 (2) , 2014.

[6].Pratiksha Y. Pawar and S. H. Gawande, Member, IACSIT''M-Commerce Security Using Random LSBSteganography and Cryptography''.International Journal of Machine Learning and Computing, Vol. 2, No. 4, August 2012

Pratiksha Y. Pawar and S. H. Gawande, Member, IACSIT

[7]. SwapnaliZagade, SmitaBhosale. ''Secret Data Hiding in Images by using DWT Technique's'' n International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-3, Issue-5, June 2014.

[8]. Mr. Falesh M. Shelke1, Miss. Ashwini A. Dongre2, Mr. Pravin D, Soni3, from India.'' Comparison of different techniques forSteganography in images''International Journal of Application or Innovation in Engineering & Management (IJAIEM)Volume 3, Issue 2, February 2014.

[9]. 1Stuti Goel,2Arun Rana,3Manpreet Kaur''A Review of Comparison Techniques of Image Steganography''. in IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE) e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 6, Issue 1 (May. - Jun. 2013).

[10]. J. Fridrich and M. Goljan, "Steganalysis based on JPEG compatibility",

SPIE Multimedia Systems and Applications IV, vol. 4518, no. 275,

2001.

[11]. Cox et al., 2008

Appendices

ppendix A. LSB Technique:

```
clc

c = imread('c:\users\NOHA\pictures\MY_BABY.bmp');

figure(1),imshow(c);title('1.Cover image')

message = '   aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa '

message = strtrim(message);%delete the spaces pefor the %first and after the last elements

m = length(message) * 8;

AsciiCode = uint8(message);%convert each char to its %ascicode

binaryString = transpose(dec2bin(AsciiCode,8));%convert %from dec to binary of 8bits(ascicode
length row and %8coloumn,then transpose convert to matrix of 8row and %asciicode %length
colomn

binaryString = binaryString(:);

N = length(binaryString);

b = zeros(N,1); %b is a vector of bits

for k = 1:N

if(binaryString(k) == '1')

b(k) = 1;

else

b(k) = 0;

end

end


%%%%%EMBEDDING%%%%%%


s = c;

height = size(c,1);

width = size(c,2);

k = 1;

for i = 1 :  height

for j = 1 : width
```

```
LSB = mod(double(c(i,j)), 2);
if (k>m || LSB == b(k))
s(i,j) = c(i,j);
k = k + 1;
else
if(LSB == 1)
s(i,j) = c(i,j) - 1;
k = k + 1;
else
s(i,j) = c(i,j) + 1;
k = k + 1;
end
end
end
end
imwrite(s,'c:\users\NOHA\MyDocuments\msg+MY_BABY.jpg');
%Retriever coding
s =imread('c:\users\NOHA\MyDocuments\msg+ MY_BABY.jpg' );
height = size(s,1);
width = size(s,2);
%For this example asume that the max size is 100 bytes, %or 800 bits, (byte = 8 bits
m = 800;
k = 1;
for i = 1 : height
for j = 1 : width
if (k <= m)
b(k) = mod(double(s(i,j)),2);
k = k + 1;
end
```

```
end
end
binaryVector = b;
binValues = [ 128 64 32 16 8 4 2 1 ];
if mod(length(binaryVector),8) ~= 0
error('Length of binary vector must be a multiple of 8.')
end
binMatrix = reshape(binaryVector,8,100);
% display(binMatrix);
textString = char(binValues*binMatrix);
disp(textString);
```

APPENDIX B. PVD

%%%%%PIXLE VALUE DIFFERENT%%%%%%%%%%%%%%%

```
clc;
clear all;
close all;
message = 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
cover = imread('c:\users\NOHA\pictures\ MY_BABY.bmp');
bits = reshape(dec2bin(message, 8)', 1, [])-'0';

stego = cover;
stegoemg=PVDEMBEDING(message,stego);
m=length(message);
stego1=imread('c:\users\NOHA\My Documents\pvd+MY_BABY. bmp');
extracted = zeros(1, m);
extractedSteg=PVDEXTRACTING(stego1,extracted);
textString = char(extractedSteg)
```

%%%%%PVD EMBEDDING FUNCTION%%%%%%%%%%

```
function[final]= PVDEMBEDING(inputText,stegoemg)
m=inputText;
len = length(m);
in = [];
in=[in dec2bin(len,20)];
for i=1:len        %character convert to binary
in=[in dec2bin(m(i),7)];
end
%get cover image
a = imread('c:\users\NOHA\pictures\ MY_BABY.bmp');
```

```
[r,c]=size(a);
final=double(a);
next=0;
capacity=0; %bit space that can be embedded
for x=0:1:r-1      %traverse through all the pixel value on
for y=0:2:c-1   %the image by 2 consecutive pair non %overlaping pixel
if (1+x>r || 2+y>c)
break;
end
g=a(1+x,1+y:2+y);
g=double(g);
d=g(1,2)-g(1,1);    %d=difference between 2 pixel
lb=[0 8 16 32 64 128]; %lowerbound
ub=[7 15 31 63 127 255]; %upperbound
dap=abs(d);
for i=1:1:6        %test the R boundary
if(dap >= lb(i)&& dap <= ub(i))
w=ub(i)-lb(i)+1; %quantization width of range
t=log2(w);      %maximum bit can be embedded between 2 pixel
capacity=capacity+t;
nt=2^t;
FREM =mod((g(1,1)+g(1,2)),nt);
if(next>length(in))
m=0;
m1=0;
elseif(next+t>length(in))
if(1+next>=length(in))
k=zeros(1,t);
else
```

```
k=in(1+next:length(in));
end
diff =next+t-length(in);
k1=zeros(1,t);
if(diff>0)
for j=1:next+t-length(in)
k1(j)=k(j);
end
end
k=k1;
next=next+t;
k=bin2dec(char(k));
if(1+next>length(in))
m=0;
m1=0;
else
m=abs(FREM-k);
m1=2^t-abs(FREM-k);
end
else
k=in(1+next:t+next);
next=next+t;
k=bin2dec(char(k));
m=abs(FREM-k);
m1=2^t-abs(FREM-k);
end
end
end
```

```
if(FREM>k && m<=((2^t)/2) && g(1,1)>=g(1,2))
P0=[g(1,1)-ceil(m/2) g(1,2)-floor(m/2)];
elseif (FREM>k && m<=((2^t)/2) && g(1,1)<g(1,2))
P0=[g(1,1)-ceil(m/2) g(1,2)-floor(m/2)];
elseif (FREM>k && m>((2^t)/2) && g(1,1)>=g(1,2))
P0=[g(1,1)+ceil(m1/2) g(1,2)+floor(m1/2)];
elseif (FREM>k && m>((2^t)/2) && g(1,1)<g(1,2))
P0=[g(1,1)+ceil(m1/2) g(1,2)+floor(m1/2)];
elseif (FREM<=k && m<=((2^t)/2) && g(1,1)>=g(1,2))
P0=[g(1,1)+ceil(m/2) g(1,2)+floor(m/2)];
elseif (FREM<=k && m<=((2^t)/2) && g(1,1)<g(1,2))
P0=[g(1,1)+ceil(m/2) g(1,2)+floor(m/2)];
elseif (FREM<=k && m>((2^t)/2) && g(1,1)>=g(1,2))
P0=[g(1,1)-ceil(m1/2) g(1,2)-floor(m1/2)];
elseif (FREM<=k && m>((2^t)/2) && g(1,1)<g(1,2))
P0=[g(1,1)-ceil(m1/2) g(1,2)-floor(m1/2)];
end
if((g(1,1)==0 || g(1,2)==0)&&(P0(1)<0 || P0(2)<0))
P1=[P0(1)+((2^t)/2) P0(2)+((2^t)/2)];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
final(1+x,2+y)=P1(1,2);
elseif((g(1,1)==255 || g(1,2)==255)&&(P0(1)>255 || P0(2)>255))
P1=[P0(1)-((2^t)/2) P0(2)-((2^t)/2)];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
final(1+x,2+y)=P1(1,2);
elseif(dap>128)
if(P0(1)<0 && P0(2)>=0)
P1=[0 P0(1)+P0(2)];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
```

```
final(1+x,2+y)=P1(1,2);
elseif(P0(1)>=0 && P0(2)<0)
P1=[P0(1)+P0(2) 0];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
final(1+x,2+y)=P1(1,2);
elseif(P0(1)>255 && P0(2)>=0)
P1=[255 P0(2)+(P0(1)-255)];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
final(1+x,2+y)=P1(1,2);
elseif(P0(1)>=0 && P0(2)>255)
P1=[P0(1)+(P0(2)-255) 255];
final(1+x,1+y)=P1(1,1); %replace new pixel value into final
final(1+x,2+y)=P1(1,2);
end
else
final(1+x,1+y)=P0(1,1); %replace new pixel value into final
final(1+x,2+y)=P0(1,2);
end
end
end
if(next>length(in))
disp('Embedded Successfully...Writing to output image');
try
imwrite(uint8(final), 'c:\users\NOHA\My Documents\pvd+happy.bmp');
%construct new image using final pixel values
catch
disp('Unable to write into output image file');
disp('Execution Unsuccessful...Exiting');
end
```

```
end
```

%%%%%PVD EXTRACTING FUNCTION%%%%%%%%%%

```
function[finaltxt]= PVDEXTRACTING(embedded,outputtxt)
finaltxt=outputtxt;
msg = [];
flag = 0;
a=embedded;
[r,c]=size(a);
j=0;
length=0;
for x=0:1:r-1
for y=0:2:c-1
if (1+x>r || 2+y>c)
break;
end
gp=a(1+x,1+y:2+y);
gp=double(gp);
nd=abs(gp(1,2)-gp(1,1));
lb=[0 8 16 32 64 128];
ub=[7 15 31 63 127 255];
for i=1:1:6
if(nd>=lb(i)&&nd<=ub(i))
w=ub(i)-lb(i)+1;
t=log2(w);
FREM=mod(gp(1,1)+gp(1,2),2^t);
k=dec2bin(FREM,t);
msg = [msg k];
```

```
j=j+t;
if(flag==0 && j>=20)
length=bin2dec(msg(1:20))+3; %possible 3 char error
length=length*7;
flag=1;
end
if(flag==1 && j>=length)
j=1;
for i=20:7:length-7
finaltxt(j)=bin2dec(msg(1+i:7+i));
j=j+1;
end
end
end
end
end
end
```

APPENDIX C. IWT

%%%%%INTEGER WAVELET GRAY IMAGE%%%%%

```
clc;
clear all;
close all;
message = 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaAAAAAaaab'
m=length(message);
cover1 =imread('c:\users\NOHA\My Documents\ MY_BABY. jpg');
cover2=rgb2gray(cover1);
cover=double(cover2);
stego = cover2;
height = size(cover, 1);
width = size(cover, 2);
for i = 1 : height
for j = 1 : width
if (cover(i,j)>=(255-15))
cover(i,j) =255-15;
elseif (cover(i,j)<= 15)
cover(i,j)=15;
end
end
end
blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
bits = reshape(dec2bin(message, 8)', 1, [])-'0';
msgbitslength=length(bits);
counter = 0;
bf = false;
for i = 1 : blocks_h
```

```
for j = 1 : blocks_w
counter=counter+1;
subMatrix = cover((i*8)-7:(i*8), (j*8)-7:(j*8));
if (counter>msgbitslength)
bf = true;
break;
end
bit = bits(counter);
[ modSubMatrix ] = IWTEMBEDDING( bit  , subMatrix);
stego((i*8)-7:(i*8), (j*8)-7:(j*8)) = modSubMatrix;
if bf
break;
end
end
if bf
break;
end
end
imwrite(uint8(stego), 'c:\users\NOHA\MyDocuments\stego.jpg', 'jpg', 'Quality' ,90);
steg = double(imread('c:\users\NOHA\My Documents\stego.jpg'));
height = size(steg, 1);
width = size(steg, 2);
blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
extractedSteg = zeros(1, msgbitslength);
bf = false;
count=0;
for i = 1 : blocks_h
for j = 1 : blocks_w
```

```
count=count+1;
if(count>msgbitslength)
bf = true;
break;
end
subMatrix = steg((i*8)-7:(i*8), (j*8)-7:(j*8));
bit = IWTDATAEXTRACTING(subMatrix);
extractedSteg(count)=bit;
end
if bf
break;
end
end
exstract=extractedSteg;
binValues = [ 128 64 32 16 8 4 2 1 ];
binMatrix = reshape(exstract,8,m);
textString = char(binValues* binMatrix)
```

%%%%%Integer wavelet (color image)%%%%%

```
clc;
clear all;
close all;
message = 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
m=length(message);
cover1 = imread('c:\users\NOHA\My Documents\ MY_BABY. jpg');
height = size(cover1, 1);
width = size(cover1, 2);
for i = 1 : height
```

```
for j = 1 : width
if (cover1(i,j)>=(255-15))
cover1(i,j) =255-15;
elseif (cover1(i,j)<= 15)
cover1(i,j)=15;
end
end
end
stego=cover1;
cover=double(cover1);
blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
bits = reshape(dec2bin(message, 8)', 1, [])-'0';
msgbitslength=length(bits);
counter = 0;
bf = false;
for i = 1 : blocks_h
for j = 1 : blocks_w
counter=counter+1;
subMatrix = cover((i*8)-7:(i*8), (j*8)-7:(j*8),2);
if (counter>msgbitslength)
bf = true;
break;
end
bit = bits(counter);
[ modSubMatrix ] = IWTEMBEDDING( bit  , subMatrix);
stego((i*8)-7:(i*8), (j*8)-7:(j*8),2) = modSubMatrix;
if bf
break;
```

```
end
end
if bf
break;
end
end
imwrite(uint8(stego), 'c:\users\NOHA\My Documents\colourstego.jpg', 'jpg', 'Quality', 90);
```

%%%%%Exttracting Embedded message%%%%%

```
stego1 = double(imread('c:\users\NOHA\MyDocuments\colourstego. jpg'));
height = size(stego1, 1);
width = size(stego1, 2);
blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
extractedSteg = zeros(1, msgbitslength);
bf = false;
count=0;
for i = 1 : blocks_h
for j = 1 : blocks_w
count=count+1;
if(count>msgbitslength)
bf = true;
break;
end
subMatrix = stego1((i*8)-7:(i*8), (j*8)-7:(j*8),2);
bit = IWTDATAEXTRACTING(subMatrix);
extractedSteg(count)=bit;
if bf
```

```
break;
end
end
if bf
break;
end
end
exstract=extractedSteg;
binValues = [ 128 64 32 16 8 4 2 1 ];
binMatrix = reshape(exstract,8,m);
textString = char(binValues* binMatrix)
```

% EMBED DATA IN INTEGER WAVELET COEFFICENTS%

```
function [ modMatrix ] = IWTEMBEDDING( bit, origMatrix )
height = size(origMatrix, 1);
width = size(origMatrix, 2);
if (height ~= 8 || width ~= 8)
error('dimensions of matrix is not 8x8');
end
lshaar = liftwave('haar','int2int');
[cA1, cH1, cV1, cD1] = lwt2(origMatrix,lshaar);
[cA2, cH2, cV2, cD2] = lwt2(cA1,lshaar);
[cA3, cH3, cV3, cD3] = lwt2(cA2,lshaar);
LSB = mod(double(cA3), 2);
if (LSB ==1&& bit ==1 )
cA3=cA3;
elseif (LSB == 0&&bit==0)
cA3=cA3;
```

```
elseif(LSB==1)
cA3 = cA3 -1;
else
cA3 = cA3 +1;
end
modcA2 = ilwt2(cA3, cH3, cV3, cD3,lshaar);
modcA1 = ilwt2(modcA2, cH2, cV2, cD2,lshaar);
modMatrix = ilwt2(modcA1, cH1, cV1, cD1,lshaar);
end
```

DATA EXTRACTING FROM INTEGER WAVELETCOEFFICENTS

```
function [ bit ] = IWTDATAEXTRACTING( matrix )
height = size(matrix, 1);
width = size(matrix, 2);
if (height ~= 8 || width ~= 8)
error('dimensions of matrix is not 8x8');
end
lshaar = liftwave('haar','int2int');
[cA1, cH1, cV1, cD1] = lwt2(matrix,lshaar);
[cA2, cH2, cV2, cD2] = lwt2(cA1,lshaar);
[cA3, cH3, cV3, cD3] = lwt2(cA2,lshaar);
lsb = mod(double (cA3), 2);
if (lsb == 0)
bit = 0;
elseif(lsb==1)
bit = 1;
end
end
```

APPENDIX D. DCT

```
%%%%%DISCRETE COSINE TRANSFORM%%%%%
clc;
clear all;
close all;
message = 'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
m=length(message);
cover1 = (imread('c:\users\NOHA\My Documents\MY_BABY. jpg'));
cover=rgb2gray(cover1);
stego=cover;
height = size(cover, 1);
width = size(cover, 2);
cover=double(cover);

blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
bits = reshape(dec2bin(message, 8)', 1, [])-'0';
msgbitslength=length(bits);
counter = 1;
bf = false;
for i = 1 : blocks_h
for j = 1 : blocks_w
subMatrix = cover((i*8)-7:(i*8), (j*8)-7:(j*8));
if (counter>msgbitslength)
bf = true;
break;
end
[ modSubMatrix,counter ] = DCT_EMBEDDING(counter ,bits  , subMatrix);
stego((i*8)-7:(i*8), (j*8)-7:(j*8)) = modSubMatrix;
```

```
if bf
break;
end
end
if bf
break;
end
end
imwrite(uint8(stego),'c:\users\NOHA\MyDocuments\MY_BABY+DCT.jpg');
stego1 = double(imread('c:\users\NOHA\My Documents\ MY _BABY+DCT.jpg'));

height = size(stego1, 1);
width = size(stego1, 2);
blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
extractedSteg = zeros(1, msgbitslength);
bf = false;
count=1;
for i = 1 : blocks_h
for j = 1: blocks_w
if(count>msgbitslength)
bf = true;
break;
end
subMatrix = stego1((i*8)-7:(i*8), (j*8)-7:(j*8));
[bits,counter] = DCT_EXTRACTING(msgbitslength,subMatrix);
c=1;
for c=1:counter
if count>msgbitslength
```

```
break;
end
extractedSteg(count)=bits(c);
count=count+1;
end
if bf
break;
end
end

if bf
break;
end
end
if ( isequal(extractedSteg, bits) )
x=('elhamd li alla')
else
x=('alla kareim')
end
binValues = [ 128 64 32 16 8 4 2 1 ];
binMatrix = reshape(extractedSteg,8,m);
textString = char(binValues* binMatrix)
```

THIS FUNCTION TO EMBED DATA IN DCT COEFFICENTS

```
function [ modMatrix,count ] = DCT_EMBEDDING( cou,bits, origMatrix )
modMatrix=origMatrix;
height = size(origMatrix, 1);
width = size(origMatrix, 2);
```

```
m=length(bits);
if (height ~= 8 || width ~= 8)
error('dimensions of matrix is not 8x8');
end
modMatrix=modMatrix-128;
dctco= dct2(modMatrix);
q=[16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];
quantizeddctco=q;
quantizeddctco = round(dctco./q);
count=cou;
for i=1:8
for j=1:8
if(count>m)
break;
end
if (quantizeddctco(i,j)~=0 && quantizeddctco(i,j)~=-1 && quantizeddctco(i,j)~=1)
LSB = mod(double(quantizeddctco(i,j)), 2);
if ( LSB == bits(count))
quantizeddctco(i,j) = quantizeddctco(i,j);
count = count + 1;
else
if(quantizeddctco(i,j)>0 )
```

```
if(LSB == 1)
quantizeddctco(i,j) = quantizeddctco(i,j) - 1;
count= count + 1;
else
quantizeddctco(i,j) = quantizeddctco(i,j) + 1;
count = count + 1;
end
else
if(LSB == 1)
quantizeddctco(i,j) = quantizeddctco(i,j) + 1;
count= count + 1;
else
quantizeddctco(i,j) = quantizeddctco(i,j) - 1;
count = count + 1;
end
end
end
end
end
end
end
modMatrix=round(quantizeddctco.*q);
modMatrix= idct2(modMatrix);
modMatrix = modMatrix+128;
x=1;
end
function[data,count]= DCT_EXTRACTING(messageLength,stego)
data=zeros(1, messageLength);
height = size(stego, 1);
width = size(stego, 2);
```

```
if (height ~= 8 || width ~= 8)
error('dimensions of matrix is not 8x8');
end
stego = stego-128;
dctco= dct2(stego);
q=[16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];
quantizeddctco=q;
quantizeddctco = round(dctco./q);
count=1;
for i=1:8
for j=1:8
if count>messageLength
break;
end
if (quantizeddctco(i,j)~=0 && quantizeddctco(i,j)~=-1 && quantizeddctco(i,j)~=1)
LSB = mod(double(quantizeddctco(i,j)), 2);
data(count)=LSB;
count=count+1;
end
end
end
count=count-1;
```

APPENDIX E. DCT_GUI

```
function varargout = dct_interface(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @dct_interface_OpeningFcn, ...
                   'gui_OutputFcn',  @dct_interface_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
 if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
 % --- Executes just before dct_interface is made visible.
function dct_interface_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
 % Update handles structure
guidata(hObject, handles);
function varargout = dct_interface_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
 nction message_Callback(hObject, eventdata, handles)
function message_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end
 % --- Executes on button press in extracting.
function extracting_Callback(hObject, eventdata, handles)
clc
[path,name]=global_demo1
filename=fullfile(path,name);
stego1 =double( imread(filename));
height = size(stego1, 1);
width = size(stego1, 2);
 blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
m=100;
msgbitslength=100*8;
extractedSteg = zeros(1, msgbitslength);
 bf = false;
count=1;
for i = 1 : blocks_h
  for j = 1: blocks_w
      if(count>msgbitslength)
         bf = true;
        break;
      end
      subMatrix = stego1((i*8)-7:(i*8), (j*8)-7:(j*8));
      [bits,counter] = DCT_EXTRACTING(msgbitslength,subMatrix);
     c=1;
     for c=1:counter
        if count>msgbitslength
           break;
        end
```

```
      extractedSteg(count)=bits(c);
      count=count+1;
      end
   if bf
     break;
  end
  end
   if bf
     break;
  end
end
 binValues = [ 128 64 32 16 8 4 2 1 ];
 binMatrix = reshape(extractedSteg,8,m);
 textString = char(binValues* binMatrix);
 set(handles.message,'string',textString);
 % --- Executes on button press in saving.
function saving_Callback(hObject, eventdata, handles)
 lbox3('dir','c:\users')
 % --- Executes on button press in embedding.
function embedding_Callback(hObject, eventdata, handles)
clc
[path,name]=global_demo1
filename=fullfile(path,name);
cover1 = imread(filename);
cover2=rgb2gray(cover1);
message=get(handles.message,'string');
 message = strtrim(message);%delete the spaces pefor the first and after the last elements
   m=length(message);
      stego=cover2;
```

```
height = size(cover2, 1);
width = size(cover2, 2);
cover=double(cover2);
 blocks_h = floor (height / 8);
blocks_w = floor (width / 8);
bits = reshape(dec2bin(message, 8)', 1, [])-'0';
msgbitslength=length(bits);
counter = 1;
bf = false;
 for i = 1 : blocks_h
  for j = 1 : blocks_w
       subMatrix = cover((i*8)-7:(i*8), (j*8)-7:(j*8));
       if (counter>msgbitslength)
          bf = true;
         break;
       end
       [ modSubMatrix,counter ] = DCT_EMBEDDING(counter ,bits  , subMatrix);
       stego((i*8)-7:(i*8), (j*8)-7:(j*8)) = modSubMatrix;
   if bf
     break;
  end
 end
  if bf
     break;
  end
end
imwrite(uint8(stego), 'c:\users\NOHA\My Documents\happystego.jpg','Quality',90);
 % --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
```

```
imshow(get(handles.CoverOrRecieve_Path ,'String'));
%lbox2('dir','c:\users\NOHA\My Documents')
 % ------------------------------------------------------------------
% --- Executes on button press in LoadImageList.
function LoadImageList_Callback(hObject, eventdata, handles)
%====================================================================
% Load up the listbox, lstImageList, with image files
% in the folder handles.CalibrationFolder.
 list_box('create','c:\users\NOHA\My Documents')
handles.output = hObject;
guidata(hObject, handles);
 if nargin == 3,
   initial_dir = pwd;
elseif nargin > 4
   if strcmpi(varargin{1},'dir')
      if exist(varargin{2},'dir')
         initial_dir = varargin{2};
      else
         errordlg({'Input argument must be a valid',...
               'directory'},'Input Argument Error!')
         return
      end
   else
      errordlg('Unrecognized input argument',...
            'Input Argument Error!');
      return;
   end
end
% Populate the listbox
```

```
load_listbox(initial_dir,handles)
 %--------------------------------
function WarnUser(warningMessage)
   uiwait(warndlg(warningMessage));
   return; % from WarnUser()
 function lbox2_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
 % Update handles structure
guidata(hObject, handles);

if nargin == 3,
   initial_dir = pwd;
elseif nargin > 4
   if strcmpi(varargin{1},'dir')
      if exist(varargin{2},'dir')
         initial_dir = varargin{2};
      else
         errordlg({'Input argument must be a valid',...
               'directory'},'Input Argument Error!')
         return
      end
   else
      errordlg('Unrecognized input argument',...
            'Input Argument Error!');
      return;
   end
end
% Populate the listbox
load_listbox(initial_dir,handles)
```

```
 % ------------------------------------------------------------
% Read the current directory and sort the names
% ------------------------------------------------------------
 function load_listbox(dir_path, handles)
cd (dir_path)
dir_struct = dir(dir_path);
[sorted_names,sorted_index] = sortrows({dir_struct.name}');
handles.file_names = sorted_names;
handles.is_dir = [dir_struct.isdir];
handles.sorted_index = sorted_index;
guidata(handles.figure1,handles)
set(handles.listbox1,'String',handles.file_names,...
   'Value',1)
set(handles.text1,'String',pwd)
% --- Executes on button press in listbox1.
function listbox1_Callback(hObject, eventdata, handles)
lbox2('dir','c:\users')
% --------------------------------------------------------------------
function Untitled_1_Callback(hObject, eventdata, handles)
```

Acronyms

LSB	least significant bit
MSB	Most significant bit
PVD	Pixel-Value Differencing
DCT	Discrete Cosine Transform Technique
DWT	Discrete Wavelet Transform Technique
IWT	Integer Wavelet Transform Technique
GUI	Graphical User Interface
JPEG	Joint Photographic Experts Group
MSE	Mean Square Error
PSNR	Peak- Signal-to-Noise Ratio
BMP	Bitmaps
AVI	Audio Video Interleave
PDF	Portable Documents Files