Sudan University of Science and Technology

College of Graduated Studies

The Effect of Compression Algorithms on Image Steganography

تأثير خوارزميات الضغط على إخفاء الصوره

Thesis Submitted in Partial Fulfillment of the Requirement for Degree of Master in Computer Science

Prepared By:

Neama Hassan Mahmoud Mustafa

Supervised By:

Dr. Talaat Mohielddin Wahby

February 2017

الآيـــه

قال تعالى :

"يَوْمَئِذٍ يَتَّبِعُونَ الدَّاعِيَ لا عِوَجَ لَهُ وَخَشَعَتِ الأَصْوَاتُ لِلرَّحْمَنِ فَلا تَسْمَعُ إِلا هَمْسًا"

صدق الله العظيم

سورة طه (108)

ACKNOWLEDGEMENT

*AT THE BEGINNING AND IN THE END ALL THANKS BELONGS TO **ALLAH**, I THANK THE ALMIGHTY FOR GIVING ME THE WILL POWER AND PATIENCE TO COMPLETE THIS WORK; TRULY WITHOUT HIS GRACE NOTHING IS ACHIEVABLE.*

A LOT OF APPRECIATION AND GRATITUDE TO THE ONES WHO HAVE PUT THEIR TRUST ON US TO COMPLETE THIS RESEARCH

*SUPERVISOR: **Dr. Talaat Mohielddin Wahby***

*THANKS TO EVERY **TEACHER** WHO HAVE TAUGHT ME AND GOT ME TO WHERE I AM TODAY.*

THANKS TO ALL OF MY COLLEAGUES IN THE SEVENTH BATCH, I AM HONORED TO HAVE STUDIED WITH SUCH A BATCH.

FOR EVERYONE KNOWS ME, THANK YOU.

DEDICATION

*First and last Praise is to **Allah**,*
*To the soul of all my life, my lovely **Mother,***
*To my **Father**, my first teacher,*
Tomy brothers, sisters,colleagues and all my friends.

Abstract

Steganography is the hiding of important and different structure of information in other medium without discovered method by unauthorized. In this research, we proposed a data hiding scheme using multilevel image steganography and losslesscompression. Apply multi-level steganography, in first level using Least Significant Bits (LSB) techniques and in second level using Pixel Value Difference technique (PVD). Lossless data compression technique Huffman and Run Length Encoding between first and second level of steganography are applied. Comparative analysis of proposed method has been done on basis of parameters like PSNR, MSE. The results showed that the compression algorithms have no significant impact on the process of steganography. Also,results demonstrated that the Huffman compression algorithm highly efficient than RLE comparison algorithm. Comparison was based on the PSNR values that were obtained, as an example, the value of PSNR is equal to 43.36dB when using Huffman to compress first image and the value is 40.69 dB when using RLE algorithm for image compression.

الخلاصه

الإخفاء(Steganography)هوإخفاءالمعلومات المهمه والمختلفة الهيئات داخل وسائط أخرى بطريقه لاتسمح للمتطفل بإكتشافها.في هذا البحث، اقترحنا طريقه إخفاء البيانات في صوره باستخدام الإخفاء متعدد المستويات. في المستوى الأول تم إستخدام خوارزميه (LSB)لإخفاء النص داخل الصوره، وتم إستخدام خوارزميه (PVD) في المستوى الثاني لإخفاء الصوره داخل الصوره الثانيه.تم ضغط الصوره المخرجه من المستوى الأول بإستخدام خوارزميه RLE مره و خوارزميه Huffman في المره ثانيه.عميله تحليل البيانات تمت بناءأ على حساب عوامل أساسيه هي PSNR وMSE ، أظهرت النتائج ان خوارزميتي الضغط ليست ذات اثر كبير على عمليه الاخفاء.كما اثبتت النتائج ان خوارزميه الضغط (Huffman)ذات كفاءه عاليه مقارنه بخوارزميه(RLE).تمت المقارنه بناءاً على قيم PSNR التي تم الحصول عليها من النتائج،على سبيل المثالكانت قيمهPSNR تساوي 43.36ديسبل عند استخدامخوارزميه Huffman في ضغط الصوره الاولى وكانت القيمه 40.69ديسبل عند استخدام خوارزميه RLE لضغط الصوره.

Table of Contents

List ofFigures

List of Tables

CHAPTERI

INTRODUCTION

1.1 Introduction

Most important factor of information technology and communication is information Security. Generally Information security means protecting information from unauthorized access, disruption, modification or simply illegal use. Most widely used data hiding techniques are cryptography and steganography.

Steganography is technique to hide the information in some media (cover medium). It is a technique of invisible communication which hides the existence of the message. The word come from steganos meaning "covered, concealed, or protected", and graphein meaning "writing". Most steganography jobs have been carried out on images, video clips, texts, music and sound.

Hiding information inside images is a popular technique nowadays. An image with a secret message inside can easily be spread over the World Wide Web or in newsgroups. The use of steganography in newsgroups has been researched by German steganographic expert Niels Provos, who created a scanning cluster which detects the presence of hidden messages inside images that were posted on the net. However, after checking one million images, no hidden messages were found, so the practical use of steganography still seems to be limited. To hide a message inside an image without changing its visible properties, the cover source can be altered in ”noisy” areas with many color variations, so less attention will be drawn to the modifications. The most common methods to make these alterations involve the usage of the least-significant bit or LSB, masking, filtering and transformations on the cover image. These techniques can be used with varying degrees of success on different types of image files.

Steganography can be single level or multilevel. In single level used one steganography method to hide the data.
Multi-Level Steganography, is a new concept for hidden communication in computer networks. It uses at least two steganographic methods are utilized either these methods are same or different type, in such a way that one method (the upper-level) serves as a carrier for the second one (the lower-level).

Image compression addresses the problem of reducing the amount of information required to represent a digital image. It is a process intended to yield a compact representation of an image, thereby reducing the image storage transmission requirements. Every image will have redundant data. Redundancy means the duplication of data in the image. Either it may be repeating pixel across the image or pattern, which is

repeated more frequently in the image. The image compression occurs by taking benefit of redundant information of in the image. Reduction of redundancy provides helps to achieve a saving of storage space of an image. Image compression is achieved when one or more of these redundancies are reduced or eliminated.

Image compression may be lossy or lossless. Lossless compression compresses the image by encoding all the information from the original file, so when the image is decompressed, it will be exactly identical to the original image. Examples of lossless image compression are PNG and GIF.Lossy compression as the name implies leads to loss of some information. The compressed image is similar to the original uncompressed image but not just like the previous as in the process of compression some information concerning the image has been lost. They are typically suited to images. The most common example of lossy compression is JPEG. [1]

1.2 Problem Statement

Systems that use only one level of Steganography are usually more vulnerable, to increase the security of data hiding two levels of Steganography will be used in this system.The proposed system uses a modified version of LSB. The second Steganography level use another algorithm called Pixel Value Differences (PVD).

Lossless compression algorithms (RLE and Huffman) will be used in this system and study the effect of these algorithms on image steganography.

1.3 Objectives

The main objective is applying the lossless compression algorithms on image steganography to reduce the number of bits required to represent an image and therefore reducing the image storage and transmission requirements. Additionally the proposed method has some sub-objectives:

-Enhance the information security by using Multi-level Steganography (MLS), level one LSB image steganography and level two pixel value difference techniques (PVD).

- Add more complexity to the Steganography process through applying it in two levels.

- Develop a System that provides a high degree of data confidentiality.

- Measure the performance of the proposed algorithm.

1.4 Research Methodology

Apply multi-level steganography, in first level using Least Significant Bits (LSB) techniques, LSB substitution is the process of adjusting the least significant bit pixels of the carrier image.A few least significant bits (LSB) are substituted with in data to be hidden then the pixels are arranged in a manner of placing the hidden bits before the pixel of each cover image to minimize the errors. In second level using pixel value difference technique (PVD). PVD scheme provides high imperceptibility to the stego image by selecting two consecutive pixels and designs a quantization range table to determine the payload by the difference value between the consecutive pixels. Besides, it offers the advantage of conveying a large number of payloads, while still maintaining the consistency of an image characteristic after data embedding.

Then applying compression process between first and second level of steganography. For image compression apply two types of lossless compression, Huffman coding**,** the technique is to use a lower number of bits to encode the data in to binary codes that occurs more frequently.And Run-Length Encoding (RLE) that encodes the case of two consecutive pixels of the same intensity into a single code word, hence gaining on compression.

Comparative analysis of compression has been done on basis of parameters like PSNR, MSE.The Peak Signal to Noise Ratio (PSNR) is the ratio between maximum possible power and corrupting noise that affect representation of image, and MSE is signal fidelity measure is to compare two signals by providing a quantitative score that describes the degree of similarity/fidelity or, conversely, the level of error/distortion between them. Usually, it is assumed that one of the signals is a pristine original, while the other is distorted or contaminated by errors.

1.5 THESIS LAYOUT

This research is comprised of five chapters divided as follows:

Chapter one contains brief introduction to Steganography and compression.

Chapter two literatures review and related works.

Chapter three contains the implementation of the proposed method.

Chapter four is the results and discussion.

Chapter five contains the conclusion and recommendations.

Chapter II

Literature Review and Related Work

2.1Introduction

Most important factor of information technology and communication is information Security. Generally Information security means protecting information from unauthorized access, disruption, modification or simply illegal use. Most widely used data hiding techniques are cryptography and steganography.Steganography is the art and science of invisible communication. This is accomplished through hiding information in other information, thus hiding the existence of the communicated information. The word steganography is derived from the Greek words "*stegos*" meaning "cover" and "*grafia*" meaning "writing" defining it as "covered writing". In image steganography the information is hidden exclusively in images. Steganography is mostly used on computers with digital data being the carriers and networks being the high speed delivery channels. [1]

The main goal of steganography is to hide information in the other cover media so that other person will not notice the presence of the information. This is a major distinction between this method and the other methods of covert exchange of information because, for example, in cryptography, the individuals notice the information by seeing the coded information but they will not be able to comprehend the information. However, in steganography, the existence of the information in the sources will not be noticed at all. [2]

Most steganography jobs have been carried out on images, video clips, texts, music and sounds .Nowadays, using a combination of steganography and the other methods, information security has improved considerably. In addition to being used in the covert exchange of information, steganography is used in other grounds such as copyright, preventing e-document forging. Almost all digital file formats can be used for steganography, but the formats that are more suitable are those with a high degree of redundancy. Redundancy can be defined as the bits of an object that provide accuracy far greater than necessary for the object's use and display. The redundant bits of an object are those bits that can be altered without the alteration being detected easily. Image and audio files especially comply with this requirement [1]. Figure 2.1 shows the four main categories of file formats that can be used for steganography.

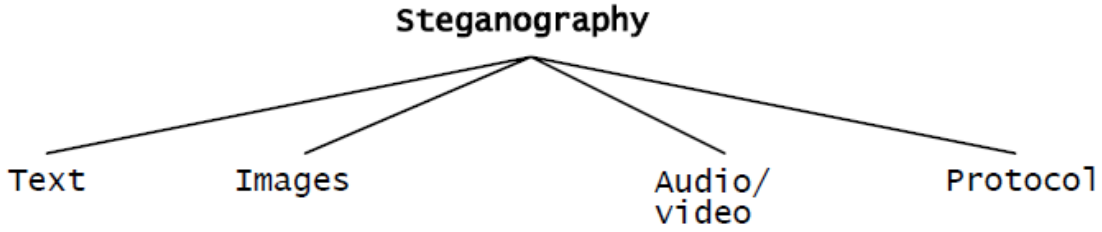


Figure 2.1: categories of steganography

There are other two techniques that seem to be same as Steganography. They are Watermarking and Fingerprinting. Both these techniques sounds to be same and provide same end goals but both are very different in the way they working. Watermarking allows a person to provide hidden copyright notices or other verification licenses. Whereas, Fingerprinting uses each copy of the content and make it as unique to the receiver [3]. In watermarking and fingerprinting the fact that information is hidden inside the files may be public knowledge sometimes it may even be visible , while in steganography the imperceptibility of the information is crucial.[1]

Steganography can be split into two types, these are fragile and robust:

Fragile: In Fragile steganography, if the file is modified, then the secret information is destroyed. For example the information is hidden the .bmp file format. If the file format is changed into .jpeg or some other format the hidden information is destroyed. [3]

Robust: Robust marking aims to embed information into a file which cannot easily be destroyed. [2]

2.2 Image Steganography

Hiding the data by taking the cover object as image is referred as image steganography. In image steganography pixel intensities are used to hide the data. In digital steganography, images are widely used cover source because there are number of bits presents in digital representation of an image.

Image steganography terminologies are as follows:

• Cover-Image: Original image which is used as a carrier for hidden information.

• Message: Actual information which is used to hide into images. Message could be a plaintext or some other image.

• Stego-Image: After embedding message into cover image is known as stego-image.
• Stego-Key: A key is used for embedding or extracting the messages from cover-images and stego-images.
Generally image steganography is method of information hiding into cover-image and generates a stego-image. This stego-image then sent to the other party by known medium, where the third party does not know that this stego-image has hidden message. After receiving stego-image hidden message can simply be extracted with or without stego-key (depending on embedding algorithm) by the receiving end. Basic diagram of image steganography is shown in Figure 2 without stego-key, where embedding algorithm required a cover image with message for embedding procedure. Output of embedding algorithm is a stego-image which simply sent to extracting algorithm, where extracted algorithm unhide the message from stego-image. [4]

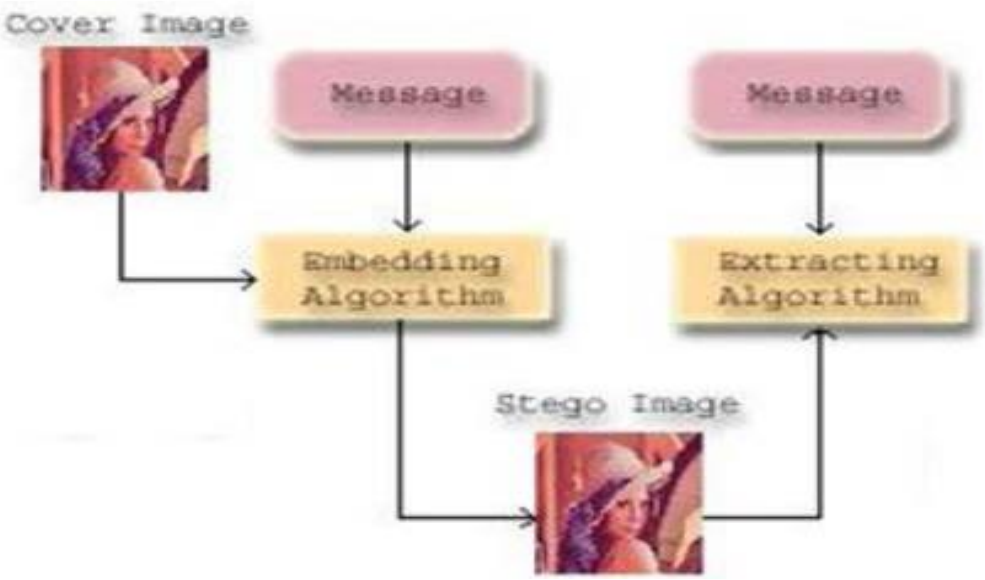


Figure 2.2: Image Steganography

Image steganography techniques can be divided into two groups the spatial Domain or Image Domain and Transform Domain or frequency domain.

2.2.1 Spatial Domain

There are many versions of spatial steganography, all directly change some bits in the image pixel values in hiding data.

2.2.1.1 Least Significant Bit (LSB)

LSB technique is implemented in spatial domain. The technique converts image into shaded Gray Scale image. This image will be act as reference image to hide the text. Using this grey scale reference image any text can be hidden. Single character of a text can be represented by 8-bit. If the reference image and the data file are transmitted through network separately, we can achieve the effect of Steganography. Here the image is not at all distorted because said image is only used for referencing. Any huge amount of text material can be hidden using a very small image. Decipher the text is not possible intercepting the image or data file separately. So, it is more secure.

In a gray scale image each pixel is represented in 8 bits. The last bit in a pixel is called as Least Significant bit as its value will affect the pixel value only by “1”. So, this property is used to hide the data in the image. Here we have considered last two bits as LSB bits as they will affect the pixel value only by “3”. This helps in storing extra data. The Least Significant Bit (LSB) steganography is one such technique in which least significant bit of the image is replaced with data bit.

In this method the least significant bits of some or all of the bytes inside an image is replaced with a bits of the secret message. The LSB embedding approach has become the basis of many techniques that hide messages within multimedia carrier data. LSB embedding may even be applied in particular data domains – for example, embedding a hidden message into the color values of RGB bitmap data, or into the frequency coefficients of a JPEG image. LSB embedding can also be applied to a variety of data formats and types. Therefore, LSB embedding is one of the most important steganography techniques in use today. The useful feature of the LSB steganography techniques is LSB replacement that makes LSB steganography as simple. To reflect the message it needs to be hidden, LSB replacement steganography flips the last bit of each of the data values. Consider an 8-bit gray scale bitmap image where each pixel is stored as a byte. And it also representing in a gray scale value. Suppose the first eight pixels of the original image have the following gray scale values: 11010010
01001010
10010111
10001100
00010101
01010111

00100110
01000011

The letter C whose binary value is 1000001. To hide this binary value it can replace the LSBs of these pixels to have the following new gray scale values:
11010011
01001010
10010110
10001100
00010100
01010110
00100111
01000011

On an average, only half the LSBs need to be changed. The difference between the cover (i.e. original) image and the stego image is difficult to observe by human eye. The LSB technique which is implemented to 24 bit format is very difficult to detect contrary to the 8 bit format. Another example of LSB technique is: Considering a grid for 3 pixels which is having 24-bit image and the number 300 is to be embedded using LSB technique.
PIXELS: (01010101 01011100 11011000)
(10110110 11111100 00110100)
(11011110 10110010 10110101)
C: 10000011
(01010101 01011100 11011000)
(10110110 11111100 00110100)
(1101111110110011 10110101)

In the above example the number C was embedded into the first 8 bytes of the grid and only the 2 bits need to be changed according to the embedded message .On an average, to hide a secret message using the maximum cover size, only half of the bits in an image will need to be modified.[5]

2.2.1.2 Pixel Value Differencing (PVD)

The pixel-value differencing (PVD) method was originally proposed to hide secret messages into 256 gray-valued images.[6]It can embed larger amount of data without much degradation in the image quality and thus are hardly noticeable by human eyes. It is based on the fact that human eyes can easily observe small changes in the gray values of smooth areas in the image but they cannot observe relatively larger changes at the edges areas. PVD uses the difference of each pair of pixels to determine the number of message bits that can be embedded into that pixel pair. It starts at the upper-left corner of the cover image and scans

the image in a zigzag manner. Then, it partitions the resulting sequence into blocks where each block consists of two consecutive non overlapping pixels. The differences of the two-pixel blocks are used to categorize the smoothness properties of the cover image. Pixels around an edge area will have larger differences whereas pixels at a smooth area will have smaller differences. The larger the difference, the more the bits that can be embedded into that pixel pair.[6]

2.2.2 Transform Domain

In frequency domain, images are first transformed and then the message is embedded in the image. When the data is embedded in frequency domain, the hidden data resides in more robust areas, spread across the entire image, and provides better resistance against statistical attacks. The most common frequency domain method usually used in image processing is the discrete cosine transform. In this technique the image is divided into 8×8 blocks and DCT transformation on each block is performed. DCT arranged the pixel of image according to their frequency value. The data bits are embedded in the low frequency coefficients of DCT. [7]

2.3 Multi-Level Steganography

In MLS, at least two steganography methods are utilized simultaneously, in such a way that one method (called the upper-level) serves as a carrier for the second one (called the lower-level). Such a relationship between two (or more) information hiding solutions has several potential benefits. The most important is that the lower-level method steganographic bandwidth can be utilized to make the steganogram unreadable even after the detection of the upper-level method: e.g., it can carry a cryptographic key that deciphers the steganogram carried by the upper-level one. It can also be used to provide the steganogram with integrity. Another important benefit is that the lower-layer method may be used as a signaling channel in which to exchange information that affects the way that the upper-level method functions, thus possibly making the steganographic communication harder to detect. [8]

2.4 Image Compression

Image compression addresses the problem of reducing the amount of data required to represent a digital image. It is a process intended to

yield a compact representation of an image, thereby reducing the image storage/transmission requirements.

The image compression techniques are broadly classified into two categories depending whether or not an exact replica of original image could be reconstructed using the compressed image. These are lossless and lossy technique.

2.4.1 Lossless Compression Technique

In Lossless Compression Technique, the image or video that is compressed that can be regenerated by using the decompression technique, no part of data is lost in the process. These are called noiseless since they do not add noise to the signal. It is also known as entropy coding since it use statistics/decompression techniques to eliminate/minimize redundancy. Lossless compression is used for only few applications with stringent requirements such as medical imaging. [8]

2.4.1.1 Run Length Encoding (RLE)

Run Length Encoding (RLE) is a simplest lossless compression technique which is most commonly used. This algorithm searches for runs of bits, bytes, or pixels of the same value, and encodes the length and value of the run. RLE achieves best results with images containing large areas of contiguous color, and especially monochrome images, For example The string is aaaaaaaabbbbbcc would have representation as (a; 8)(b; 5)(c; 3) Then compress each (char; length) as a unit .[9]

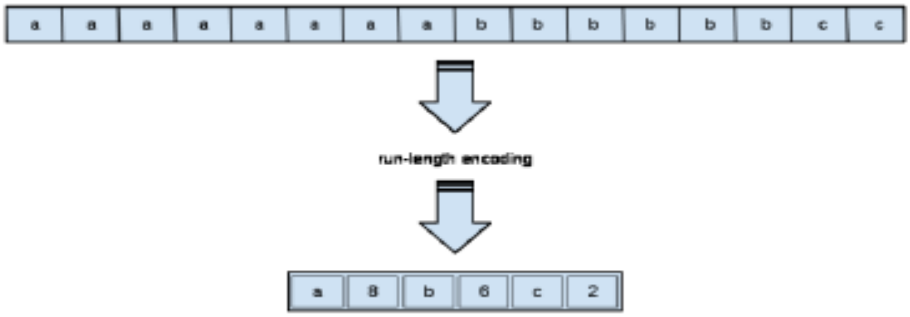


Figure 2.3: RLE Example

Steps of algorithm for RLE are as follows.

Step 1: Input the string.

Step 2: From first symbol or character give a unique value.

Step 3: Read the next character or symbol, if character is last in string then exit otherwise.

a: If: next symbol is same as the previous symbol then give same unique value as pervious.

b: Else if: next symbol is not same, than give its new value that is unmatched from previous value.
Step 4: Read and count additional symbols
Step 5: Go to step 3 until a non-matching value to the not same symbol from previous.
Step 6: Display the result that is count of occurrence of single symbol with that particular symbol.

2.4.1.2 Huffman Coding

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol, where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence. Huffman coding is based on frequency of occurrence of a data item. The pixels in the image are treated as symbols. The symbols that occur more frequently are assigned a smaller number of bits, while the symbols that occur less frequently are assigned a relatively larger number of bits. Huffman code is a prefix code. The binary code of any symbol is not the prefix of the code of any other symbol. Huffman algorithms have two ranges static as well as adaptive. Static Huffman algorithm is a technique that encodes the data in two passes. In first pass, it is required to calculate the frequency of each symbol and in the second pass it constructs the Huffman tree. On average, using Huffman coding on standard files can shrink them anywhere from 10% to 30% depending on the character distribution. Huffman encoding creates the tree like structure when it encodes the given string. The example of Huffman Encoding with algorithm is as follows: [9]

Step 1: Input the string

7	2	32	24
K	Z	C	M

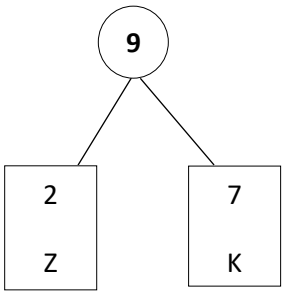
Step 2: Sorting the data by frequencies

2	7	24	32
Z	K	M	C

Step 3: Choose two smallest frequencies count.

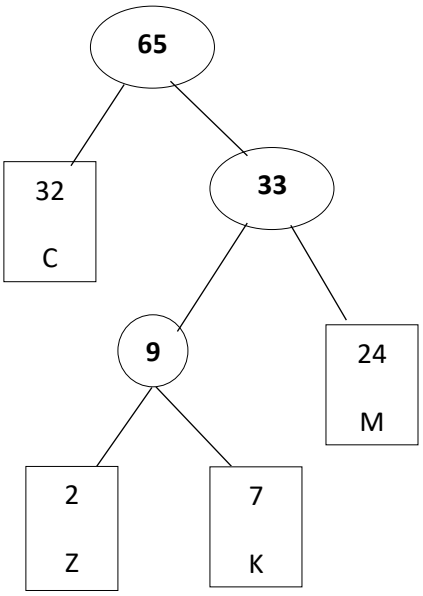
2	7
Z	K

Step 4: Merge them together with sum of them and update the data



Step 5: Repeat step 2, 3, 4.
The final Huffman tree is as follows:

.



2.4.2 Lossy Compression Technique

In Lossy compression technique, the image that is regenerated by the decompression process is not the original copy of the image that is compressed. Some data might be lost during the technique. These compression techniques are cheaper that is they take less time and space when it comes to sending millions of bits per second for images and video. Several methods have been developed using lossy compression techniques. JPEG(Joint Photographic Experts Group) encoding is used to compress pictures and graphics, MPEG(Moving Picture Experts Group) encoding is used to compress video, and MP3(MPEG audio layer 3) for audio compression.[8]

2.5Cryptography

Cryptography is the science of using mathematics to encrypt and decrypt data. Cryptography enables you to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone except the intended recipient.
A cryptographic algorithm works in combination with a key—a word, number, or phrase—to encrypt the plaintext.The same plaintext encrypts to different cipher text with different keys. The security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key.

2.5.1 RC4 Algorithm

RC4 is a stream cipher designed in 1987 by Ron Rivest for RSA Security. It is a variable key-size stream cipher with byte-oriented operations. The algorithm is based on the use of a randompermutation.Eight to sixteen machine operations are required per output byte, and thecipher can be expected to run very quickly in software. RC4 was kept as a trade secret by RSASecurity.

The RC4 algorithm is remarkably simply and quite easy to explain. A variable-length key of from 1 to 256 bytes (8 to 2048 bits) is used to initialize a 256-byte state vector S, withelements S[0], S[1], ..., S[255]. At all times, S contains a permutation of all 8-bit numbers from 0 through 255. For encryption and decryption, a byte K is generated from S byselecting one of the 255 entries in a systematic fashion. As each value of kis generated, theentries in S are once again permuted.
Initialization of S, the entries of S are set equal to the values from 0 through 255 in ascending order; thatis; S[0] = 0, S[1] = 1, ..., S[255] = 255. A temporary vector, T, is also created. If the length ofthe key K is 256 bytes, then K is transferred to T. Otherwise, for a key of length keylenbytes, thefirst keylenelements of T are copied from K and then K is repeated as many times as necessary to fill out T. These preliminary operations can be summarized as follows:

```
/*Initialization*/
for i = 0 to 255
Do
S[i] = i
T[i] = K [i modkeylen]
```

Next we use T to produce the initial permutation of S. This involves starting with S[0] andgoing through to S[255], and, for each S[i],

swapping S[i] with another byte in S according to ascheme dictated by T[i].

```
/* Initial Permutation of S*/
j = 0;
for i = 0 to 255
Do
j = (j + S[i] + T[i]) mod256
Swap (S[i], S[j]).
```

Because the only operation on S is a swap, the only effect is a permutation. S still containsall the numbers from 0 through 255.

In Stream GenerationOnce the S vector is initialized, the input key is no longer used. Stream generation involvesstarting with S[0] and going through to S[255], and, for each S[i], swapping S[i] with anotherbyte in S according to a scheme dictated by the current configuration of S. After S[255] isreached, the process continues, starting over again at S[0].

```
/*Stream Generation*/
i, j = 0
While (true)
i = (i + 1) mod256
j = (j + S[i]) mod256
Swap(S[i], S[j])
t = (S[i] + S[j]) mod256
k = S[t];
```

To encrypt, XOR the value kwith the next byte of plaintext. To decrypt, XOR the value kwith the next byte of cipher text.

2.6 Related Works

Section two in this chapter presents the related work in image steganography and lossless compression techniques and present compression table between them in the latter part of this section.

In [10] S. KHAN and others, presented “Run-Length Encoding Based Lossless Compressed Image Steganography”. This paper presents an RLE based compression image steganography technique. The RLE BASED hiding technique compresses the stego image without the loss of secret information. The proposed method insures the retrieval of secret information in it full health.

In implementation section, first the secret message is hidden in the least significant bits of the cover image pixels. The resulted stego image after the data hiding process is compressed by using a lossless compression technique. The stego image is encoded by using RLE scheme. In run-length coding the stego image is transformed into identical symbols’ segments. Each segment is represented in the form of a pair of symbol and its number of occurrences i.e. its probability of occurrence.

The proposed method is implemented by using Lena, Man, Space craft, Coral, Shuttle, Sphere, MRI and Fingerprint images as cover for hiding the same secret message. The effect of Steganography on the quality of the image is measured quantitatively by calculating MSE and PSNR for each stego image by keeping hiding capacity fixed at 50% level. Figure 2.4 show the resulted MSE and PSNR

Image	MSE	MSE (dB)	PNSR (dB)
Lena	19.6739	12.9389	35.1919
Man	19.1416	12.8208	35.3100
Space Craft	19.1950	12.8319	35.2983
Coral	17.0644	12.3209	35.8099

Figure 2.4: MSE and PSNR of lossless compressed image Steganography

After data hiding each of the stego images is processed by using RLE. The effect of RLE is measured in the term of compression ratio. Figure 2.5 show the compression ratio of images before and after data hiding.

Image	CR of Cover Image	CR of Stego Image
Lena	1.03	1.06
Man	1.07	1.10
Space Craft	1.15	1.21
Coral	1.13	1.15

Figure 2.5:RLE Compression Ratio of images

In [11] Rahul and Naresh they proposed a data hiding scheme using image steganography and compression. In this technique, secret data is preprocessed first and then the preprocessed secret data is embedded into the LSBs of the cover image depending upon the intensity of the pixel values of the cover image. For pre-processing a lossless data compression technique, LZW (Lempel–Ziv–Welch) technique is used for pre-processing the data. In this technique sequence of 8-bit secret data is encoded as fixed-length 12-bit codes. The code from value 0 to 255 represents one character sequences consisting of the corresponding 8-bit character. As the data is encoded, the codes with values 256 through 4095 are created in a dictionary depending upon the sequences encountered in the data. A dictionary is initialized to contain the single-character strings corresponding to all the possible input characters. At every step in the compression process, input characters are gathered into a sequence until the next character comes that will make a sequence for which there is no code in the dictionary. The code for the sequence without the character encountered is emitted, and a new code for the sequence with the character encountered, is added in the dictionary. The algorithm works by scanning the input secret data for successively longer substrings until a string is found that is not in the dictionary. When such a string is found, the index for the string without the last character is fetched from the dictionary and sent to output, and the new string including the last encountered character is added to the dictionary with the next available code. The last input character is then used as the next starting point to scan for substrings. In this way, successively longer strings are added in the dictionary and made available for subsequent encoding as single output values.

They compare their technique with other techniques that they are more recently developed and have good performance. The experimental results show that the proposed technique is better than the existing technique and produces better results. For every image the value of PSNR, MSE and CAP i.e. maximum embedding capacity of proposed technique is more than the MKA technique. The Capacity of all the cover images to embed the secret data increases by applying the proposed technique. The security of the secret data also increases due to its preprocessing.

In paper [12] they deal with the approach of embedding the bits at higher random layer which leads towards difficult discovery of data. Main aim of this paper is to improve capacity and robustness of this approach.

Firstly they Extract host audio file, evaluate sample rate, sample size etc. according to sample size (16bit or 8bit) read sample, Read message string, According to message length choose compressive

algorithm and embed output bit stream over audio, Generate new 16 bit samples by inserting message bits into random higher bits using algorithm, Embedding message bit at random higher layers such that the distortion can be minimized, Embed layer number (bit position) value with next sample and lastly convert all sample (16 bit) into regular audio stego file.

After that they calculate the SNR of different compression technique and compare between them. Experiment results show that, for smaller/medium size messages- Huffman compression algorithm gives better SNR value as compare to others. LZW is very powerful and popular too but this approach is very complex and time consuming and Shannon-Fano compression algorithm is most suitable in compression of very large messages. This approach is more robust and lossless compression techniques sufficiently improve storage capacity of host audio.

In this work [13] large colored image is hiding in smaller colored image (at least three times) by compressing embedded image with Huffman method and then using Oring bits to obtained a sequence of 0 and 1 values, Then with method that changed from Image Downgrading is used to hide this compressed image, the following diagram explain the steps of proposed method:

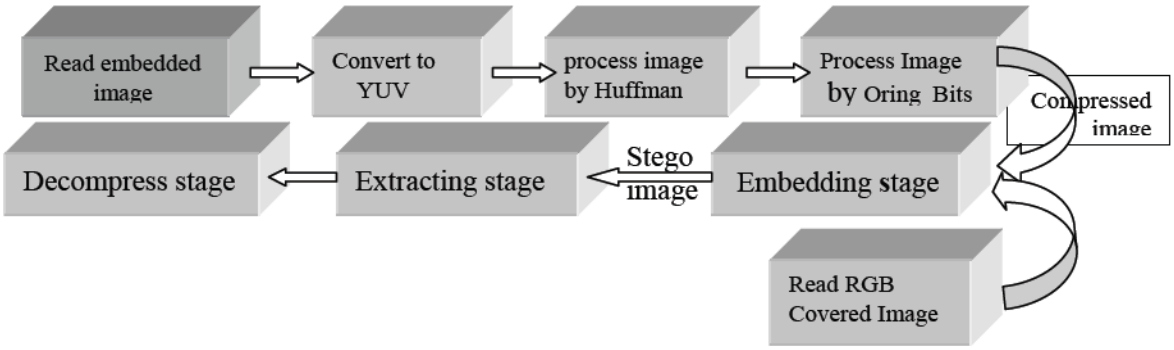


Figure 2.6: Diagram of proposed method

The proposed method tested with (150 × 150) embedded image and (50 × 50) covered image and then they compute compression ratio (CR)for compressed embedded image , Peak signal to noise ratio(PSNR) and Root Mean Square Error (RMSE) for the embedded image ,covered image, extracted image and stego-image.It appears from the observation of results of the mentioned experiments that the proposed method success in the embedding and extracting process and so the method they used to compress the embedded image was successes by the results they get it for the compression ratio to the embedded image and so PSNR, and they see the increasing with compression ratio will increase the PSNR for stego and extracted image.

In this paper [14] the proposed method CEET uses compression, encryption and LSB embedding in order to hide and recover data. Firstly, the secret data (image) is compressed by applying JPEG lossless compression algorithm.

The compressed image so obtained is then encrypted by applying the substitution encryption algorithm in the second layer, the encrypted image is converted into array of 0's and 1's. Finally these bits are then embedded into the cover image in the inner-most by inserting them into the last 2 LSBs of each pixel of the cover file. The image so obtained is the stego-image.

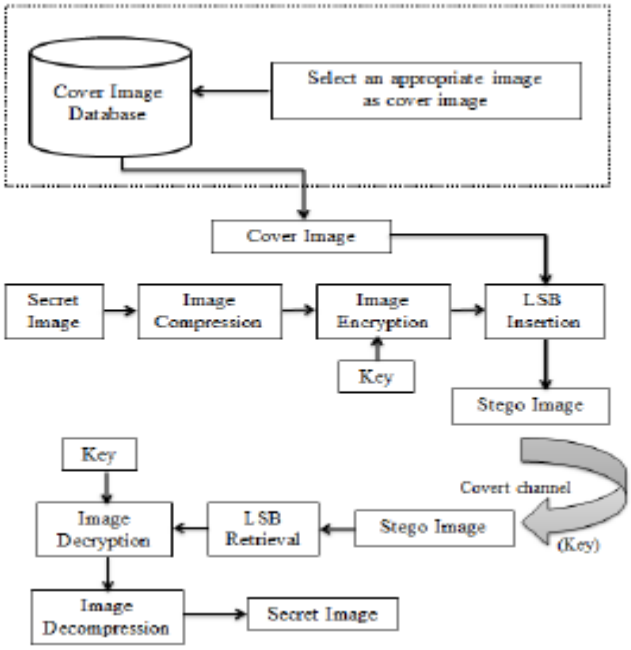


Figure 2.7: Block Diagram of proposed Steganography System

They calculate the PSNR of proposed method and compare the result with PSNR of S-tools, the resultant PSNR value of CEET is better than that of S-Tools. The proposed application CEET provides high security since encryption is applied as well as embedding capacity has also been increased by using compression algorithm keeping in mind the visual perspective of steganography.

In [15] Kamlesh Lakhwani and Kiran Kumari tried to provide concept by which lossless communication can take place. Using the Transform domain they changed the secret image to its RGB color components, converted their values in binary code and then using the RLE (RUN LENGTH ENCDING) compression algorithm compressed the binary code up to 35 percent. They were using the standard Triple DES and Hash code (MD5) algorithms to perform the security task. Then using the image domain steganography they put the encrypted and compressed binary code to the cover image and produced the stego image.

The results are then compared with various steganography methods and the PSNR of stego-image is calculated and compared with previous work as shown in figure 2.8. Comparative result shown that the PSNR will increase in proposed work so there is no difference in visible quality of cover (original) image and stego image.

Lena Image	LSB3	Jae Gilyu	First component alteration technique	Improved LSB	KVL Method
PSNR	37.92	38.98	46.11	46.65	51.98

Figure 2.8: Comparative study of various techniques with proposed Method

No.	Paper Name	Author	Year	Objective
1	Run-Length Encoding Based Lossless Compressed Image Steganography	S. KHAN, T. KHAN, M. NAEEM, N. AHMAD	2015	- The secret message is hidden in the LSB of the cover image pixels. - The stego image is encoded by using RLE scheme. - The PSNR of the resultant stego images remains above the 30dB threshold for data hiding capacity of 50%. - A reasonable compression ratio of greater than 1bpp has been achieved.[10]
2	Audio Steganography with Various Compression Algorithms to Improve Robustness and Capacity	Chintan R.Nagrecha, Prof. Prashant B. Swadas	2014	-Hide the secret data in audio file. - According to message length choose compressive algorithm. - For smaller/medium size messages- Huffman compression algorithm

				gives better SNR value - LZW is very powerful but this approach is very complex and time consuming. - Shannon-Fano compression algorithm is most suitable in compression of very large messages.[12]
3	CEET: A Compressed Encrypted & Embedded Technique for Digital Image Steganography	Palak Mahajan, Dr. Ajay Koul	2014	- The secret data (image) is compressed by applying JPEG lossless compression algorithm. - The compressed image is then encrypted by applying the substitution encryption algorithm and converted into array of 0's and 1's. - Finally these bits are inserting into the last 2 LSBs of each pixel of the cover file. - PSNR value of CEET is better than that of S-Tools.[14]
4	An Effective Compressed Image With Steganography	Anwar J.Moosa	2013	- Large colored image is hiding in smaller colored image by compressing embedded image with Huffman method. - using Oring bits to obtained a sequence of 0

				and 1 values. - The value of 0, 1 will be hiding in Least Bit Significant (LSB) for covered image. - The method that used to compress the embedded image was successes by the results they get it for the compression ratio to the embedded image and so PSNR.[13]
5	KVL Algorithm: Improved Security & PSNR for Hiding Image In Image Using Steganography	Kamlesh Lakhwani, Kiran Kumari	2013	- Changed the secret image to its RGB color components, converted their values in binary code. -Using the RLE compression algorithm compressed the binary code. - Using the standard Triple DES and Hash code (MD5) algorithms to perform the security task - Comparative result shown that the PSNR will increase in proposed work so there is no difference in visible quality of cover (original) image and stego image.[15]
6	Efficient data hiding scheme using lossless data compression and	Rahul Jain, Naresh	2012	- In this technique data is firstly preprocess by applying the LZW

	image steganography	Kumar		compression technique. - This preprocessed data is embedded into the LSBs of the cover image. - It has high PSNR value and low MSE value as compared to MKA.[11]

Table 2.1: show the summarization of related work

CHAPTER III

WORK ENVIROMENT AND PROPOSED SYSTEM ANALYSIS

3.1 Overview

This chapter describes the proposed method (Effect of Lossless Compression on Image Steganography) and explains the diagrams that clarify the proposed method. In level one modified Least Significant Bit (LSB) Image steganography, RGB image is used as a cover image with a secure data (text) converted to long bit-stream before concealing, Run Length Encoding(RLE) and Huffman compression are used to compress the image output from level one steganography, while level two pixel value differencing based image steganography (PVD) another RGB image is used as a cover image with a secure data (the RGB image output from level one) also converted to long bit-stream before concealing.

The programming language will used in implementation of the two levels (level one and level two) and compression algorithms(RLE and Huffman) is java programming language, since it contains appropriate and more suitable methods to read from file, write in file, manipulate and modify the pixels that belong to an image then save the modified image.

3.2 Proposed Method

The proposed method is using multilevel steganography and lossless compression, level one steganography will be done by embedding the secret message (text) into cover image (cover one) which is a colored image (RGB image) using Least Significant Bit (LSB) image steganography.

The output from level one is stego image referred to as (intermediate image), the intermediate image will be converted into binary and will work as input in compression.

In compression, run length encoding in which runs of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count,rather than as the original run. The Huffman encoding algorithm starts by constructing a list of all the alphabet symbols in descending order of their probabilities. It then constructs, from the bottom up, a binary tree with a symbol at every leaf. This is done in steps, where at each step two symbols with the smallest probabilities are selected, added to the top of the partial tree, deleted from the list, and replaced with an auxiliary symbol representing the two original symbols. When the list is reduced to just one auxiliary symbol (representing the entire alphabet), the tree is complete. The tree is then traversed to determine the codewords of the symbols.

In level two the proposed techniques is pixel value differencing (PVD), image is used as a cover image with a long bit-stream as the secret data. At first the cover image is partitioned into non-overlapping blocks of two consecutive pixels, pi and pi+1. From each block the difference value di is calculated by subtracting pi from pi+1.

The set of all difference values may range from -255 to 255. Therefore, |di| ranges from 0 to 255. The blocks with small difference value locates in smooth area where block with large difference values are the sharp edged area. Figure 3.1 below explain the general overview of the proposed method (embedding process).

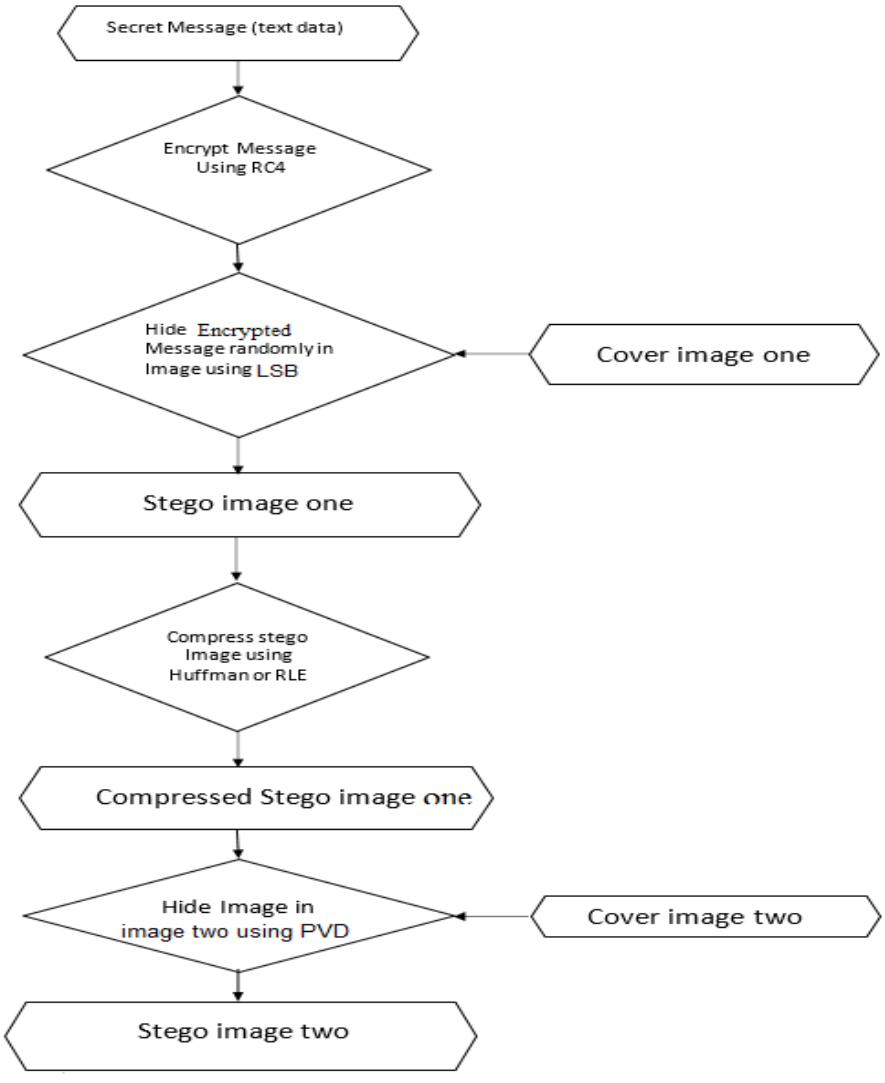


Figure 3.1: general overview of the proposed method

3.2.1 The Embedding Process in Level One Using Modified LSB

1- Read the secret message from file and Convert it to binary.
2-Encrypte message using RC4 algorithm with given password
3- Read the pixels of cover image (cover image one).
4-Put message length and password in first byte of image using LSB.
5-Generate random offset to put message randomly in image.

a. Using text size and bytes of image.
b. Subtractbyte of image from seed to get next value.
c. Insure that, the generated random value is unique.

6-Put message bits in random offset of LSB
7- After complete the bits in the binary secret text, save the resulting stego image (intermediate image).

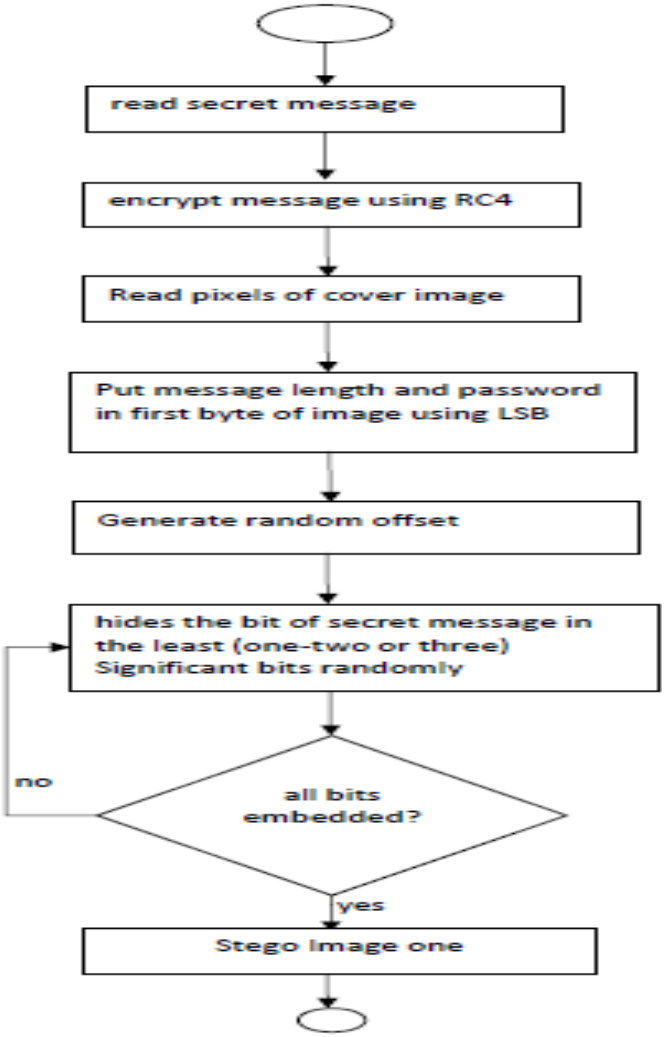


Figure 3.2: level one embedding process (enhanced LSB)

3.2.2 The Retrieving Process in Level One Using Modified LSB in Image

1- Read the stego image from level one (intermediate image).
2- Read the pixels of cover image.
3- Read message length and password from first byte of image using LSB.
4- Generate random offset to get message.
5- After reading all bits convert it to string.
6- Decrypt message using RC4 algorithm with given password.
7- Write the secret message in file.

3.2.3 The Embedding Process in Level two Using PVD

1. Calculate the difference value di of two consecutive pixels pi and p_{i+1} for each block in the cover image. This is given by $di=|p_{i+1}-p_i|$.
2. Compute the optimal range where the difference lies in the range table by using di. This is calculated as Ri=min(uk- di), where uk ≥di for all $1 \leq k \leq n$
3. Compute the number of bits‘t’ to be hidden in a pixel block can be defined as $t= \lfloor \log_2 wi \rfloor$. where wi is the width of the range in which the pixel difference di is belonging
4. Read t bits from binary secret data.
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:

(pi, pi+1)= (pi+⌈m/2⌉ , pi+1- ⌊m/2⌋), if pi ≥pi+1 and di'>di
pi- ⌊m/2⌋ , pi+1+ ⌈m/2⌉), if pi<pi+1and di>di
pi- ⌈m/2⌉), pi+1 + ⌊m/2⌋), if pi≥ pi+1 and d'i ≤ di
pi + ⌈m/2⌉ ,pi+1- ⌊m/2⌋), if pi< pi+1 and d'i ≤di .

Where m=|di'-di|. Now we obtain the pixel pair (pi', pi+1') after embedding the secret data into pixel pair (pi, pi+1). Repeat step 1-6 until all secret data are embedded into the cover image. Hence we get the stego-image.

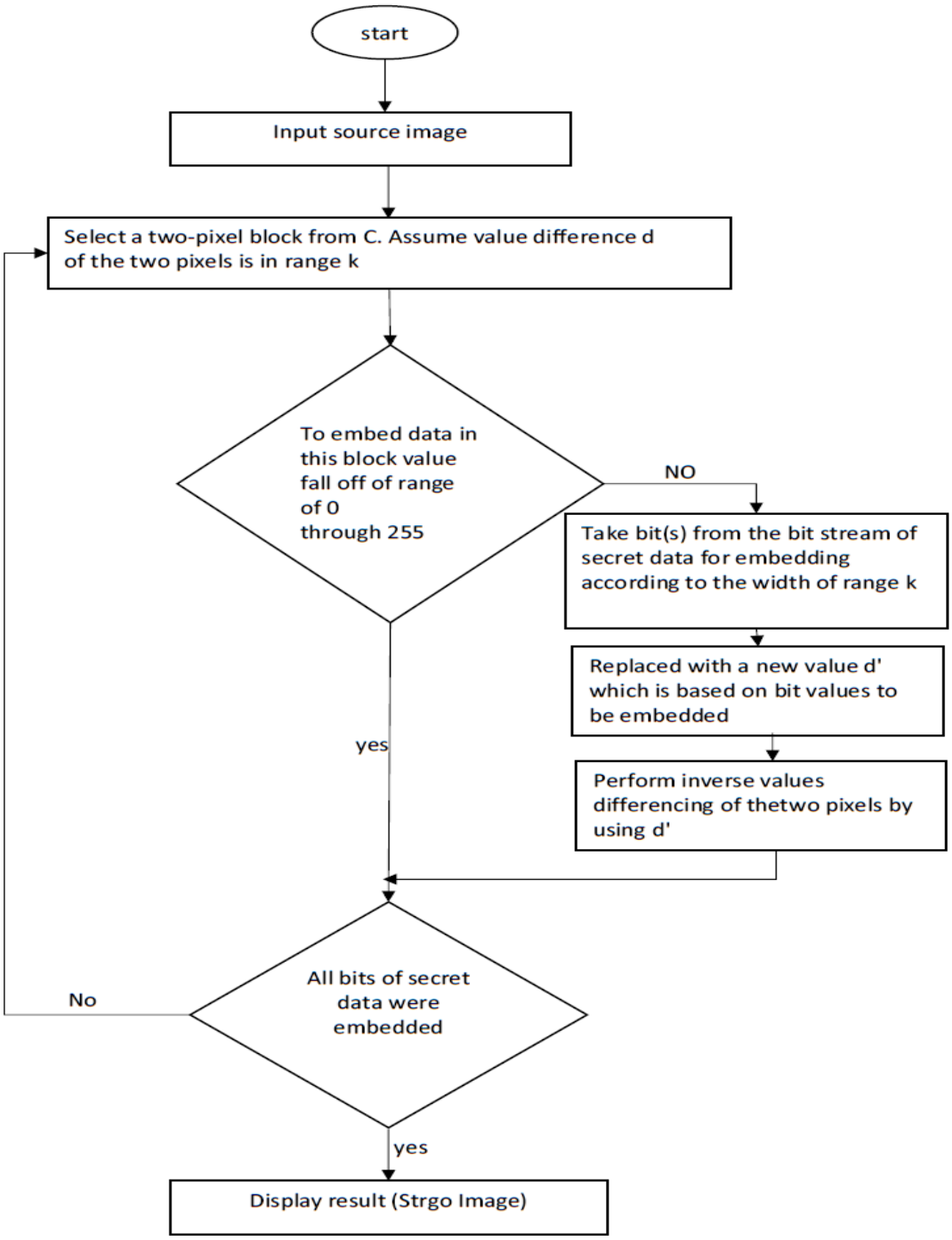


Figure 3.3: level two embedding process (PVD)

3.2.4 The Retrieving Process in Level Two Using PVD in Image

When extracting the hidden information from the stego-image, original range table is required.

1. Partition the stego-image into pixel blocks, containing two consecutive non-overlapping pixels each.
2. Calculate the difference value for each block as di'=|pi'-pi+1'|.
3. Then find the optimum range Ri of di'.
4. Then b' is obtained by subtracting li from di'.
5. Convert b' into its corresponding binary of ‘t’ bits, where t= ⌊log2wi⌋
6. These t bits are the hidden secret data obtained from the pixel block (pi',pi+1').

3.2.5 Compression Process Using RLE

1**.** Input the RGB source image file.
2. Find out the size of source image.
3. Read pixel values from first pixel of source image.
4. Read next pixel value, if current pixel is last thenexit otherwise
a. **If** next pixel value that is j is same asthe previous pixel value then Count = count+1;
b. **Else** any mismatch in RGB value of nextpixel as the previous then save as the new value of pixel in array.
5. Read and count all the value of pixel.
6. Go to step 4 until all pixel read.
7. Display the result.

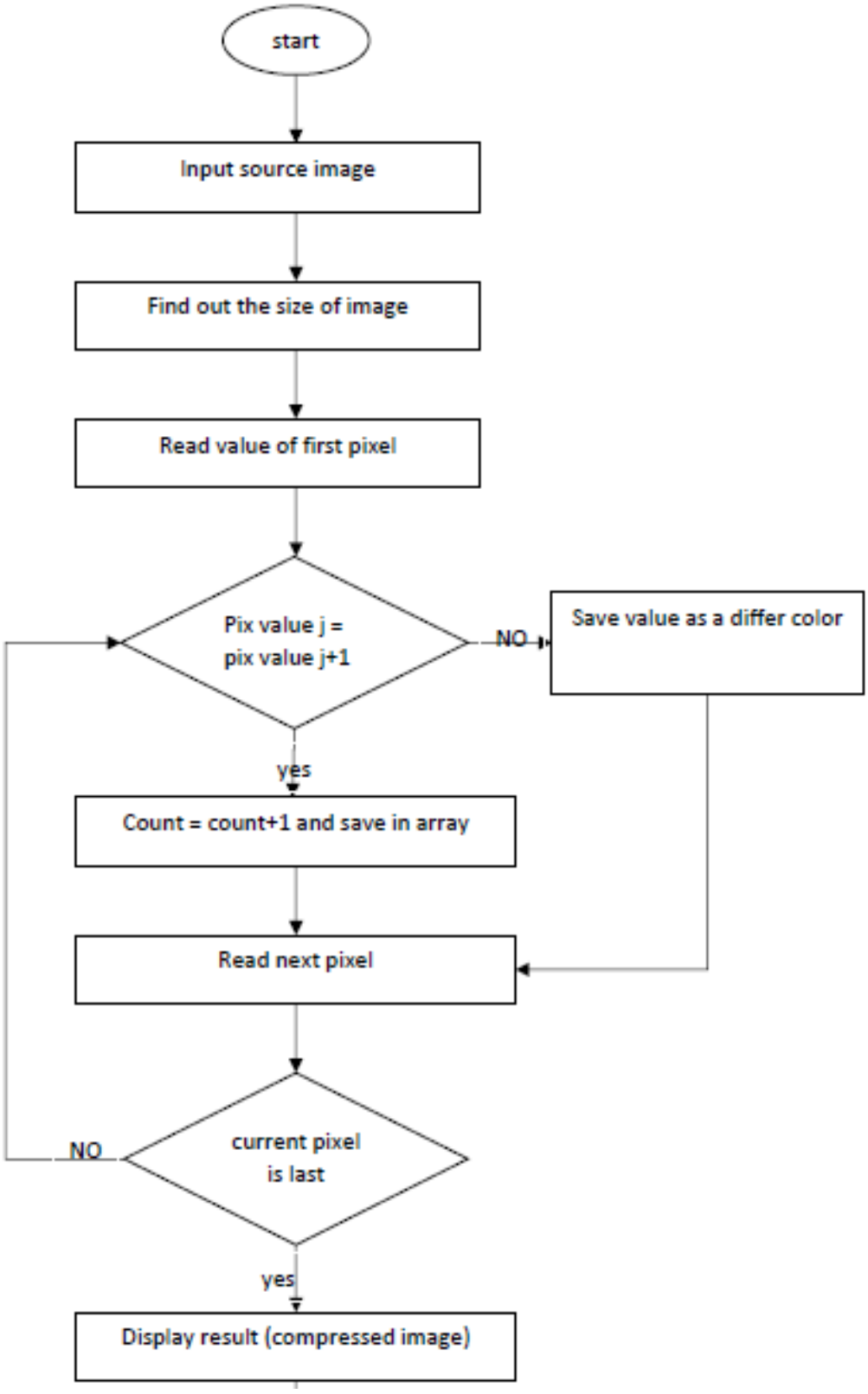


Figure 3.4: Run Length CompressionProcess

3.2.6 Decompression Process using RLE

1. Read the compressed array and obtain the imagesize.
2. Create the blank array for reconstruction ofcompressed image.
3. For reconstructing compressed image**,**
    a. Construct the i^{th} row of compressed image with putting run length value in reconstruct array from compressed array.
    b.Then construct $i+1^{th}$ row then next row and so on.
4. Step 3 is repeated until reconstruct array fill by value of compressed array.
5. Reconstruct array, store as a decompressed imagefile.
6. Display the decompressed image file.

3.2.7 Compression Process Using Huffman

1. **Count the frequency of each character in the file to be encoded.** Include a dummy character to mark the end of the file -- this will be important later. For now, call it the EOF (end of file) and mark it as having a frequency of 1.
2. **Store characters as tree nodes and put them into a priority queue.** Will be building a big binary tree with each character as a leaf, should store the characters in a format such that they can become nodes of the tree. Place these nodes into a priority queue with each character's frequency as its node's priority.
3. **Begin to build the tree.** Remove (or *dequeue*) the two most urgent things from the priority queue. Create a new tree node to be the parent of these two nodes, storing the first node as its left child and the second as its right child. The priority of the new node should be the sum of the priorities of its child. Then enqueue this new node in the priority queue.
4. **Finish building tree:** repeat the above step until there is only one node in the queue. Note that in addition to the nodes created for the characters and their frequencies, will also be dequeuing, turning into trees, and re-enqueueing parent nodes, nodes that are already themselves trees.
    - When finished, the last node in the queue will be the *root* of the encoding tree, with all the other nodes branching off from it.
    - The most frequently used characters will be the leaves closest to the top of the tree, while the rarely used characters will be positioned at the bottom of the tree, farther away from the root.
5. **Create an *encoding map.*** Walk through the tree to reach each character. Every time visit a node's left child, that's a '0'. Every time visit a node's right child, that's a '1'. When get to a character,

store the character with the sequence of 0s and 1s that it took to get there. This sequence is what the character will be encoded as in the compressed file. Store the characters and their sequences in a map.

6. **In the output file, include the encoding map as a header.** This will allow the file to be decoded.
7. **Encode the file.** For each character in the file to be encoded, write the binary sequence that have stored in the map. Once finished encoding the file, make sure to add the EOF to the end.

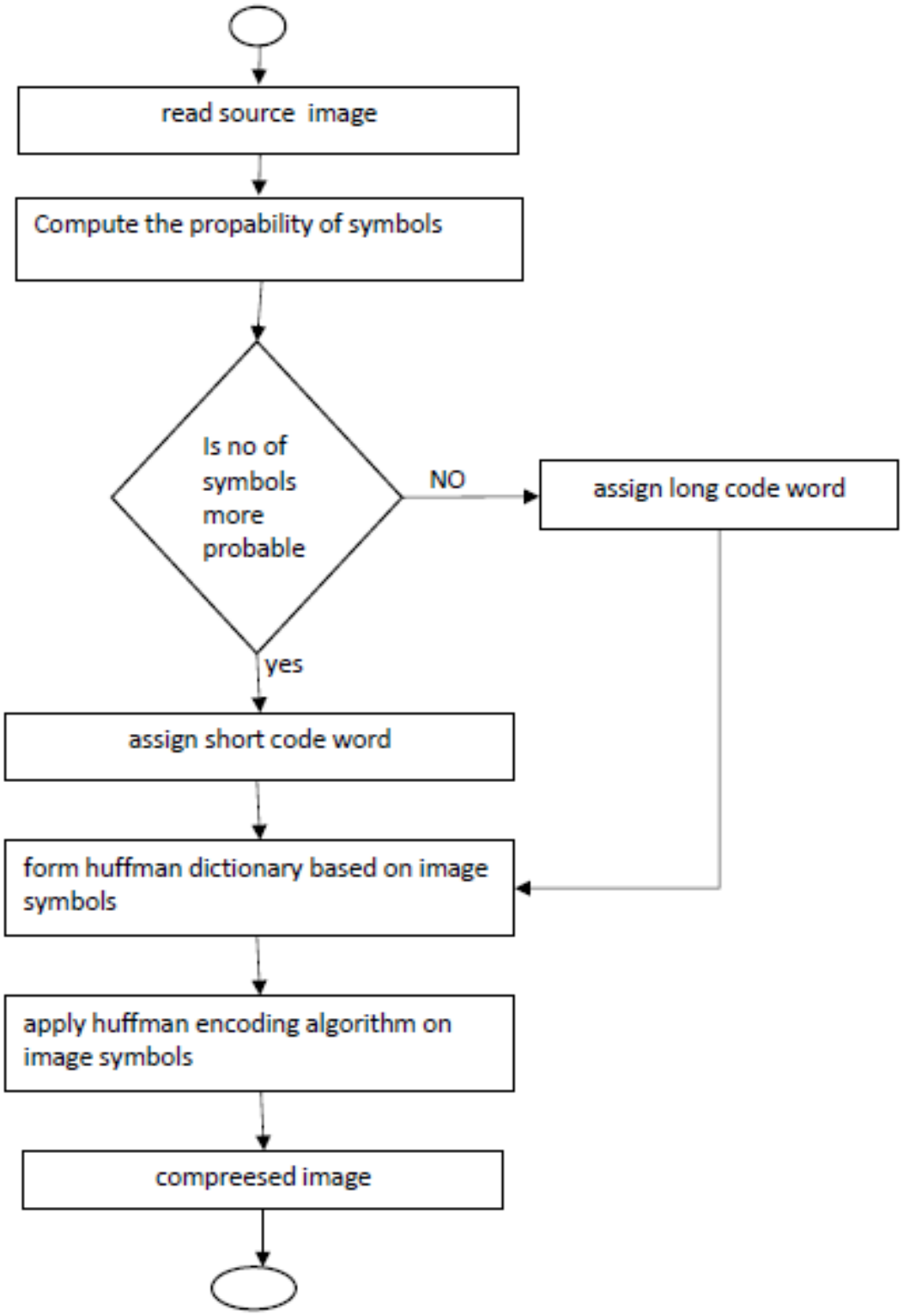


Figure 3.5: Huffman compression process

3.2.8 Decompression Process Using Huffman

1. **Read in a Huffman-encoded file.** First, read the header, which should be the encoding map. Use this to build a decoding tree in the same way you built the tree you used to encode the file. The two trees should be identical.
2. **Read in the binary one digit at a time.** Traverse the tree as read: if read in a '0', go to the left child of the node you're at, and if read in a '1', go to the right child. When reach a leaf (a node without any children), have arrived at a character. Write the character into the decoded file.
3. **Repeat until reach the EOF.** Finally have decoded the file.

CHAPTER IV

RESULTS AND DISCUSSION

4.1Introductoin

The proposed method is using multilevel steganography and lossless compression, level one steganography will be done by using Least Significant Bit (LSB) image steganography. The output from level one (intermediate image) was compressed using two types of lossless compression, Huffman codingand Run-Length Encoding (RLE). In second level using pixel value difference technique (PVD).

Comparative analysis of multilevel image steganography (Enhanced LSB and pixel differences value image steganography) has been done on basis of parameters like PSNR, MSE and embeds data size.

4.2 Evaluation Parameters

4.2.1 PSNR

The Peak Signal to Noise Ratio (PSNR) is the ratio between maximum possible power and corrupting noise that affect representation of image. PSNR is usually expressed as decibel scale. The PSNR is commonly used as measure of quality reconstruction of image. The signal in this case is original data and the noise is the error introduced. High value of PSNR indicates the high quality of image.

The mathematical representation of the **PSNR** is as follows:

$$\text{PSNR}=20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right)$$

4.2.2MSE

Mean Squared Error (MSE) is signal fidelity measure is to compare two signals by providing a quantitative score that describes the degree of similarity/fidelity or, conversely, the level of error/distortion between them.

The mathematical representation of the **MSE** is as follows:

$$\text{MSE}=\frac{1}{\text{mn}}\sum_0^{\text{m}-1}\sum_0^{\text{n}-1}\|f(\text{i},\text{j})-\text{g}(\text{i},\text{j})\|^{\ 2}$$

Legend:

f represents the matrix data of our original image

g represents the matrix data of our degraded image in question

m represents the numbers of rows of pixels of the images and i represents the index of that row

n represents the number of columns of pixels of the image and j represents the index of that column

$\mathbf{MAX_f}$ is the maximum signal value that exists in our original "known to be good" image

4.3USED SECRET MESSAGES

There are four different messages size have been used to embed them in different image size in the upper level of image steganography, examples of secret messages will be used shown in Figure 4.1, figure 4.2 and figure 4.3.

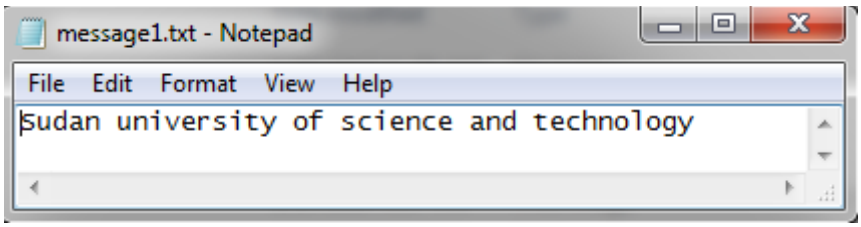


Figure 4.1: First secret message

The size of first secret message is 336 bits and the size will be increase in the next secret message, the second secret message is shown in Figure 4.2.

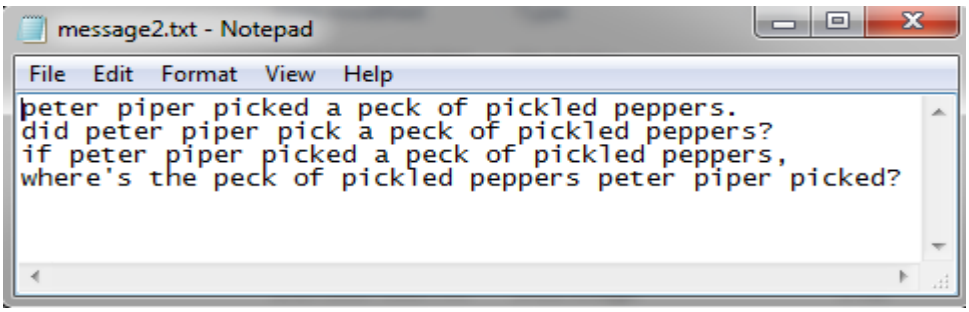


Figure 4.2: Second secret message

The size of second secret message is 1056 bits and the size will be increase in the next secret message, the second secret message is shown in Figure 4.3.

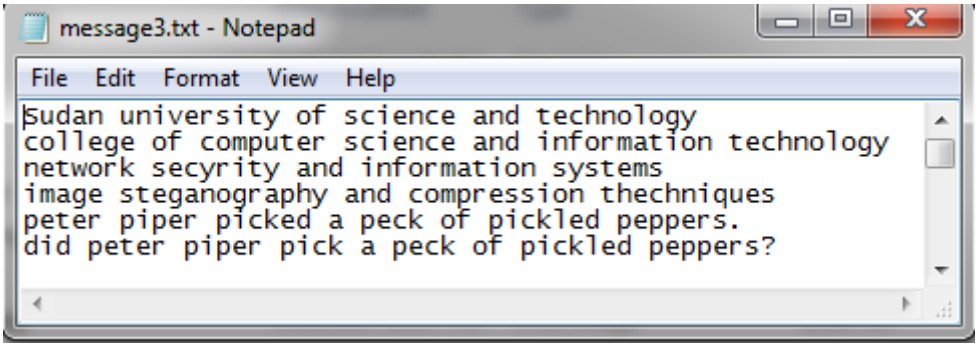


Figure 4.3: Third secret message

The size of third secret message is 11040 bits and the size of fourth secret message is 160000 bits (almost the maximum capacity).

4.4EXPERMINTAL RESULTS

After the upper level (level one – enhanced LSB) is applied to the above secret messages the output is three stego images. Each image concealing one of the secret messages. The first cover image is the stuffed-peppers image and is concealing (message1) as secret data; the size of the stego image is 5136000bits. Figure 4.4 shows the stuffed-peppers stego image.

Figure 4.4: stuffed-peppers stego image

The second cover image is baboon face image and is concealing (message2) as secret data; Figure 4.5 shows the baboon face stego image, the size of which is 1104000 bits.

Figure 4.5: baboon face stego image

The third cover image is monaliza image and is concealing (message3) as secret data; Figure 4.6 shows the monaliza stego image, the size of which is 1976000bits.

Figure 4.6: monaliza stego image

In the lower level (level two pixel value differences image steganography) three images with different sizes and dimensions have been used. The first image is the lena image Figure 4.7 with dimension 512 × 512 used as a cover image, the first secret data to be embedded in this cover image is the stuffed-peppers stego image, secondsecret data is the compressed stuffed-peppers stego image which were the output of the run length encoding and third secret data to be embedded is the

compressed stuffed-peppers stego image which were the output of the Huffman compression algorithm. The new stego images shows in Figure 4.8, Figure 4.9 and Figure 4.10 as examples.

Figure 4.7: Lena original image

Figure 4.8: Lena stego1

Figure 4.9: Lena stego2
Embedded data compressed stego image (RLE))

Figure 4.10: Lena stego3
(embedded data compressed stego image (Huffman))

Table 4.1 shows the experiment results of the lena stego images and contains the PSNR and MSE values of stego images. Figure 4.11 is a Diagram showing its PSNR values.

Secret message	Size of Secret message in bits	Level One Embedded image	Size Embedded image	Compression algorithm	Level Two Embedded image	PSNR	MSE
Messag1	336 bits	Stuffed peppers (750×349)	5136000 bits	Without compression	Lena (512×512)	43.3	3.043
				Huffman		43.36	3.001
				RLE		40.69	5.544
Messag2	1056 bits	Stuffed peppers (750×349)	5136000bits	Without compression	Lena (512×512)	43.4	2.968
				Huffman		43.4	2.969
				RLE		40.32	6.034
Messag3	11040 bits	Stuffed peppers (750×349)	5136000bits	Without compression	Lena (512×512)	43.41	2.963
				Huffman		43.32	3.025
				RLE		40.4	5.936
Messag4	160000 bits	Stuffed peppers (750×349)	5136000bits	Without compression	Lena (512×512)	43.19	3.116
				Huffman		43.04	3.232
				RLE		40.24	6.155

Table 4.1: Experimental results of Lena stego image

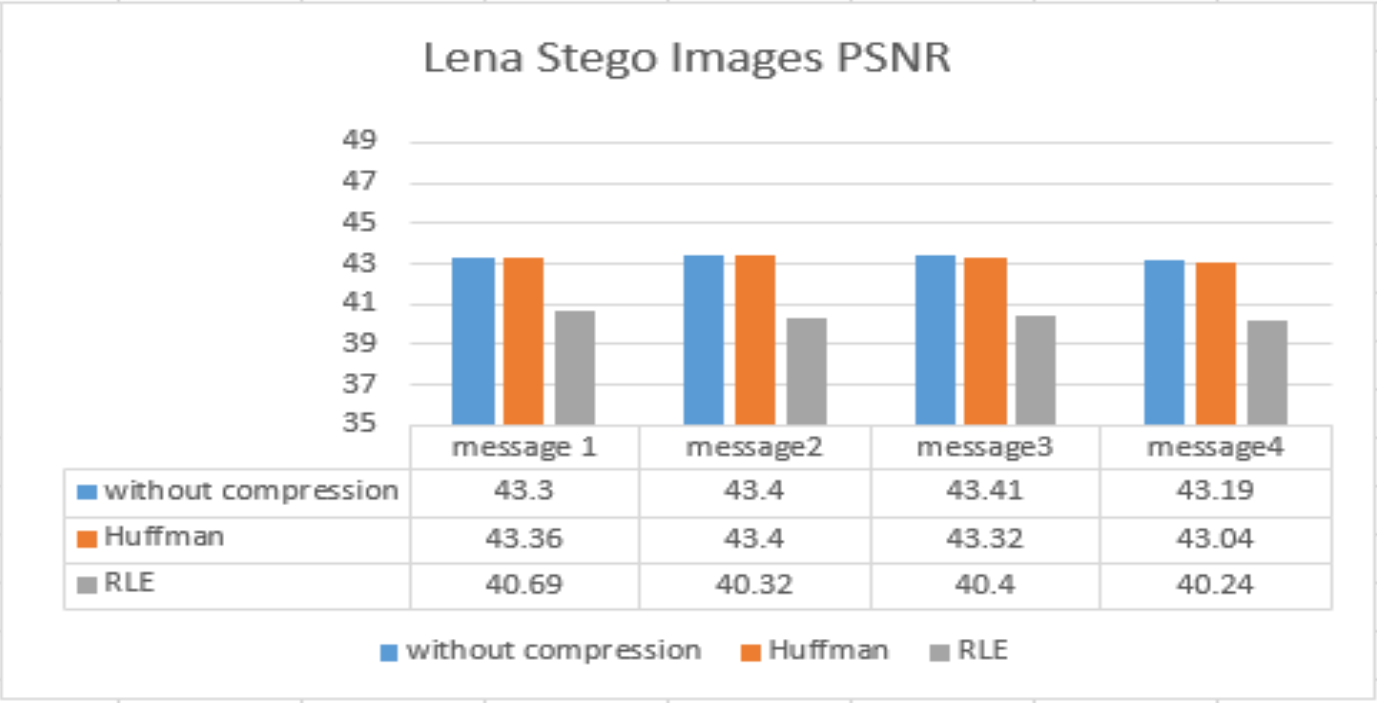


Figure 4.11: PSNR value for Lena stego images

The second image is the monaliza image (Figure 4.12) with dimension 800 × 800 used as a cover image, the first secret data to be embedded in this cover image is the baboon face stego image, the second secret data is compressed baboon face stego image which were the output of the run length encoding and third secret data to be embedded is the compressed baboon face stego image which were the output of the Huffman compression algorithm. The new stego images shows in Figure 4.13, Figure 4.14 and Figure 4.15.

Figure 4.12: monaliza original image

Figure 4.13: monaliza stego1 image

Figure 4.14: monaliza stego2 (embedded: data compressed stego image (RLE))

Figure 4.15: monaliza stego3 (embedded data: compressed stego image(Huffman))

Table 4.2 shows the experiment result of monaliza stego images and contains the PSNR and MSE values of stego images. Figure 4.16 shows the Diagram for its PSNR values.

Secret message	**Size of Secret message in bits**	**Level One Embedded image**	**Size Embedded image**	**Compression algorithm**	**Level Two Embedded image**	**PSNR**	**MSE**
Message1	768bits	Baboon face (250×250)	1104000 bits	Without compression	Monaliza (800×800)	48.06	1.015
				Huffman		47.97	1.037
				RLE		47.59	1.131

Message2	1608 bits	Baboon face (250×250)	1104000bits	Without compression	Monaliza (800×800)	48.13	0.999
				Huffman		48.03	1.023
				RLE		47.61	1.127
Message3	11040bits	Baboon face (250×250)	1104000bits	Without compression	Monaliza (800×800)	48.08	1.011
				Huffman		47.97	1.038
				RLE		47.64	1.12
Message4	160000 bits	Baboon face (250×250)	1104000bits	Without compression	Monaliza (800×800)	48.05	1.019
				Huffman		48.01	1.029
				RLE		47.57	1.137

Table 4.2: Experimental resultsof monaliza stego images

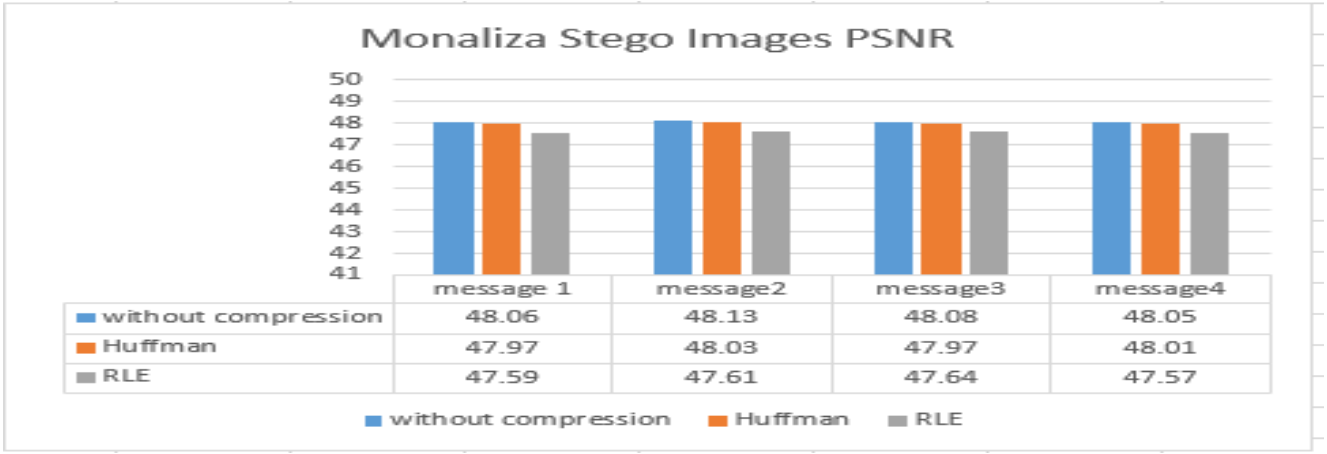


Figure 4.16: PSNR value for monaliza stego images

The third image is the baboon face image (Figure 4.17) with dimension 900 × 900 used as a cover image, the first secret data to be embedded in this cover image is the lena stego image, second secret data is the compressed lena stego image which were the output of the run length encoding and third secret data to be embedded is thecompressed lena stego image which were the output of the Huffman compression algorithm.

The examples of new stego images shows in Figure 4.18, Figure 4.19 and Figure 4.20.

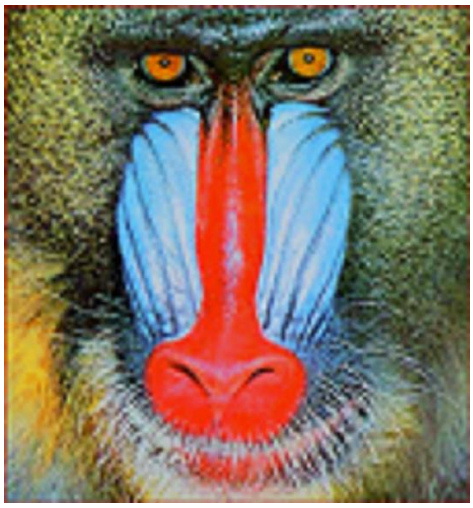


Figure 4.17: baboon face original image

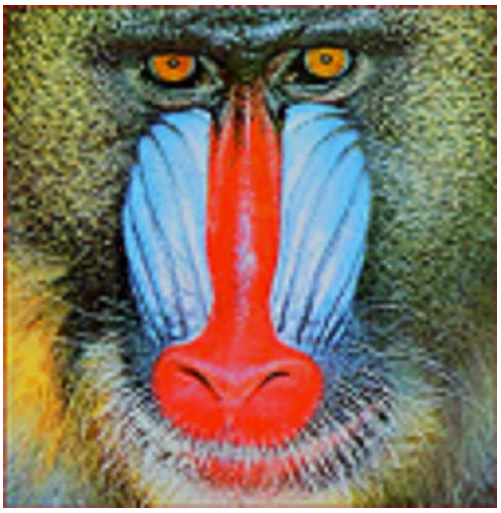


Figure 4.18: baboon face stego1 image

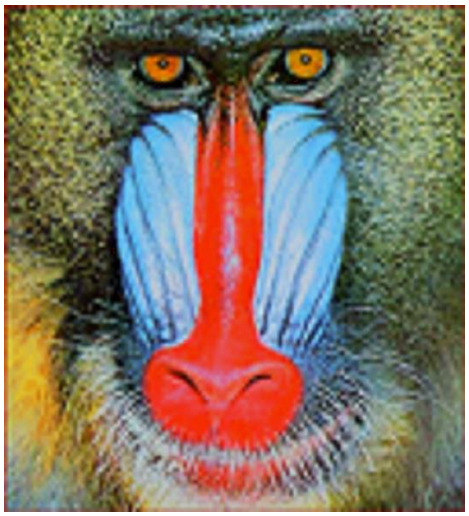


Figure 4.19: baboon face stego2 (data: compressed lena stego image (RLE)

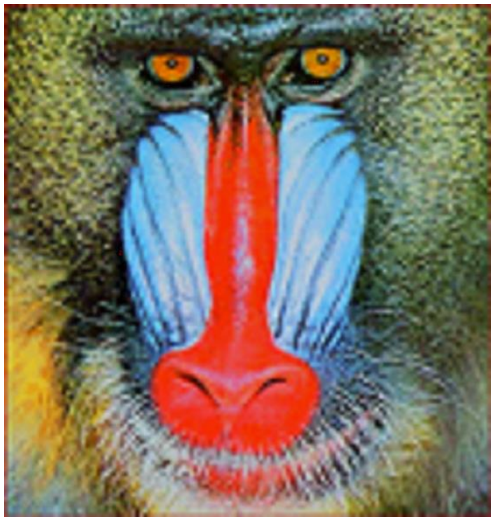


Figure 4.20: baboon face stego3 (embedded compressed lena stego image (Huffman))

Table 4.3 shows the experiment result of baboon face stego images and contains the PSNR and MSE values of stego images. Figure 4.21 shows the Diagram for its PSNR values.

Secret message	Size of Secret message in bits	Level One Embedded image	Size Embedded image	Compression algorithm	Level Two Embedded image	PSNR	MSE
Message1	368 bits	Monaliza (400×400)	1976000bits	Without compression	Baboon face (900×900)	47.51	1.153
				Huffman		47.45	1.169
				RLE		46.29	1.528
Message2	8184 bits	Monaliza (400×400)	1976000bits	Without compression	Baboon face (900×900)	47.51	1.153
				Huffman		47.48	1.160
				RLE		46.3	1.523
Message3	11280 bits	Monaliza (400×400)	1976000bits	Without compression	Baboon face (900×900)	47.49	1.160
				Huffman		47.56	1.141
				RLE		46.33	1.513
Message4	160000 bits	Monaliza (400×400)	1976000bits	Without compression	Baboon face (900×900)	47.41	1.180
				Huffman		47.29	1.212
				RLE		46.24	1.545

Table 4.3: Experiment Result of baboon face stego images

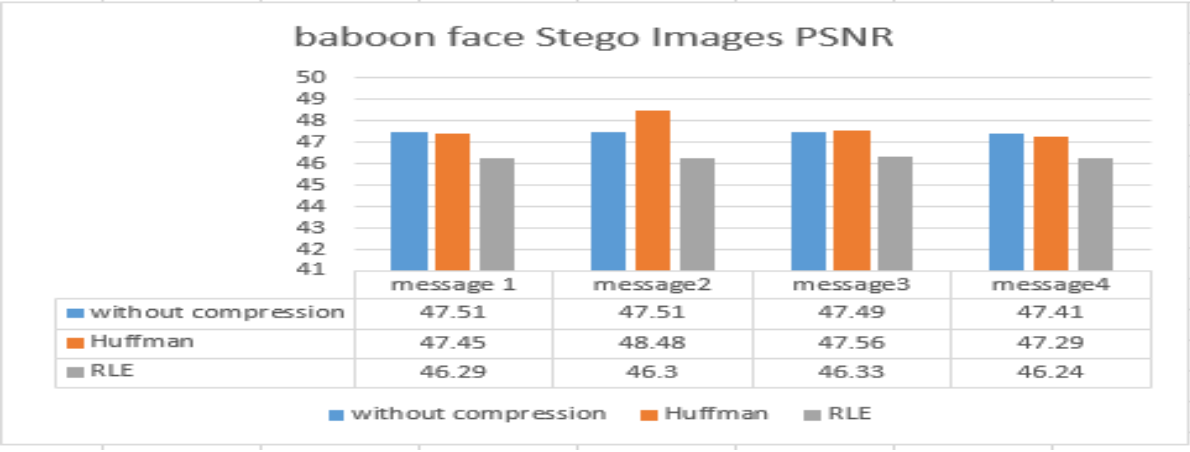
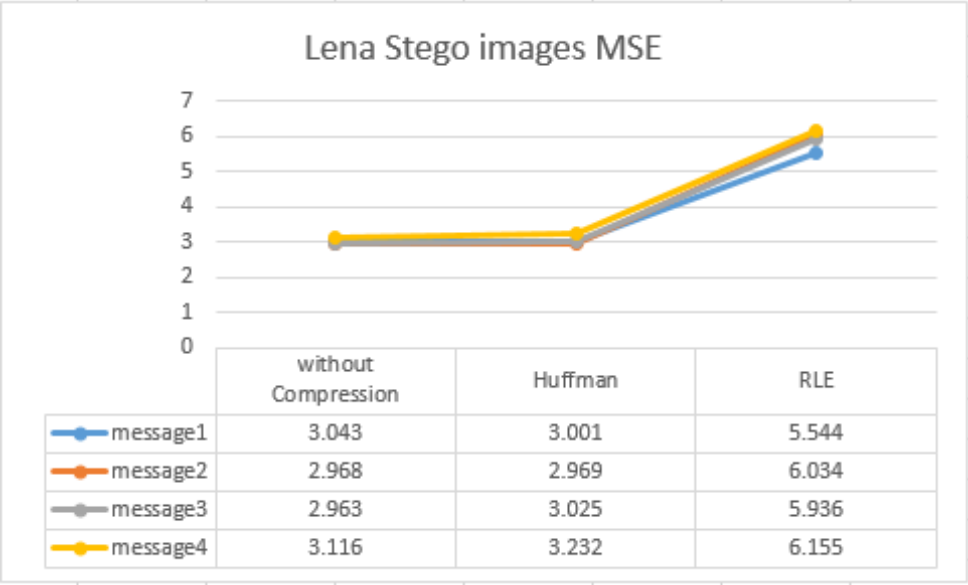


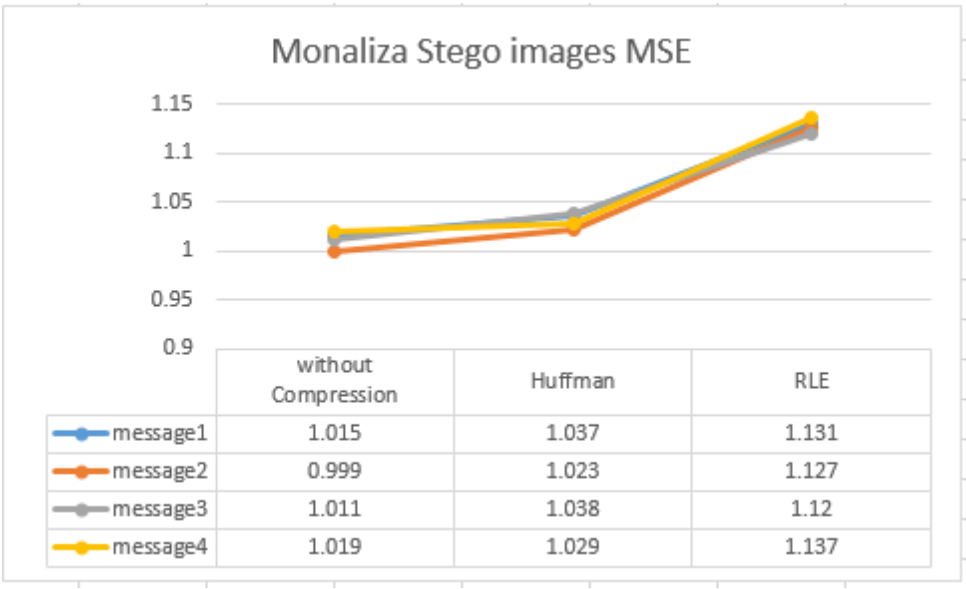
Figure 4.21: PSNR value for baboon face stego images

Figure 4.22, figure 4.23 and figure 4.24 shows the MSE values of the monaliza, Lena and Baboon face experiments.



	without Compression	Huffman	RLE
message1	3.043	3.001	5.544
message2	2.968	2.969	6.034
message3	2.963	3.025	5.936
message4	3.116	3.232	6.155

Figure 4.22: MSE values for Lena experiments.



	without Compression	Huffman	RLE
message1	1.015	1.037	1.131
message2	0.999	1.023	1.127
message3	1.011	1.038	1.12
message4	1.019	1.029	1.137

Figure 4.23: MSE values for Monaliza experiments.

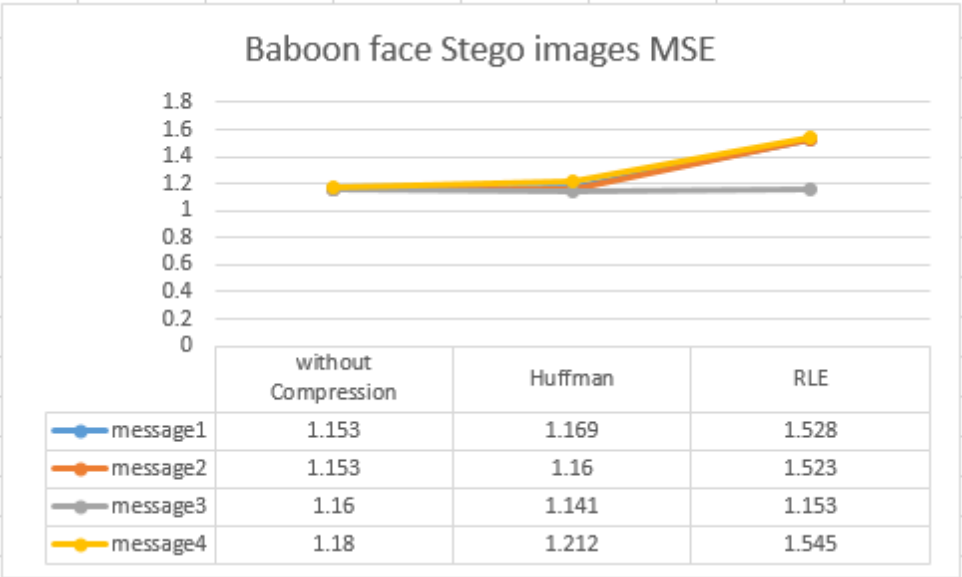


Figure 4.24: MSE values for Baboon face experiments.

4.5Discussion

Experiments were on images with different sizes at first and second stage, and also different size texts were hidden inside them. Lossless compression algorithms are used to compress images. MSE and PSNR were calculated for each image and then came out with these findings:

1-PSNR gives better results without using both compression algorithms.

2- The two compression algorithms doesn't have a major effect on the steganography process , and PSNR values doesn't differ a lot in cases whether compression algorithms are used or not.

3- PSNR values were better when using Huffman algorithm compared to its values when used with RLE compression algorithm.

4- The length of text hidden inside image in first stage doesn't have large effect on the process.

CHAPTER V

CONCLUSION AND RECOMMENDATIONS

5.1Conclusion

Steganography is the art and science of invisible communication. This is accomplished through hiding information in other information, thus hiding the existence of the communicated information. In image steganography the information is hidden exclusively in images.

The main objective is applying the lossless compression algorithms and study the effects of compression on image steganography. In this thesis, Multi-Level Steganography for image steganography, was presented. MLS consists of at least two stenographic methods are utilized respectively, in such a way that one method (called the upper-level) as a carrier for the second one (called the lower-level). The first level has been applied using modified LSB image steganography and the second level applied using pixel value differencing image steganography.

Then applying compression process between first and second level of steganography. For image compression apply two types of lossless compression, Huffman codingand Run-Length Encoding (RLE).MSE and PSNR were calculated for each image to get the results.

5.2 Recommendations

1- Apply another compression techniques.
2- Check the result of proposed algorithm using gray scale image in both levels to compare the performance results.
3- Apply compression on text file.

5.3 Future Work

- Increase the System functionality to hide all other data types like audio, video not only text data and images.
- Trying to enhance the performance of algorithms in both levels to increase the system capacity.

References

[1] T Morkel, JHP Eloff and MS Olivier, "An Overview of Image Steganography", in Proceedings of the Fifth Annual Information Security South Africa Conference (ISSA2005), Sandton, South Africa, June/July 2005.

[2]Shashikala Channalli and Ajay Jadhav, “Steganography An Art of Hiding Data”, Shashikala Channallietal /International Journal on Computer Science and Engineering Vol.1(3),137-141, 2009.

[3]R.Poornima and R.J.Iswarya, “AN OVERVIEW OF DIGITAL IMAGE STEGANOGRAPHY”, International Journal of Computer Science & Engineering Survey (IJCSES) Vol.4, No.1, February 2013.

[4]Mehdi Hussain and MureedHussain, “A Survey of Image Steganography Techniques”, International Journal of Advanced Science and Technology, Vol. 54, May 2013.

[5] Vanitha T , Anjalin D Souza , Rashmi B, Sweeta Dsouza, “A Review on Steganography Least Significant Bit Algorithm and Discrete Wavelet Transform Algorithm”, International Journal of Innovative Research in Computer and Communication Engineering, Vol.2, Special Issue 5, October 2014.

[6]Deepak Singla, RupaliSyal, “Data Security Using LSB & DCT Steganography in Images”, IJCER, Mar-Apr 2012.

[7]El-Sayed M. El-Alfy and Azzat A. Al-Sadi, Pixel-Value Differencing Steganography: Attacks and Improvements, the Second International Conference on Communications and Information Technology (ICCIT), Feb 2012.

[8]WojciechFrączek, WojciechMazurczyk, Krzysztof Szczypiorski, “Multi-Level Steganography: Improving Hidden Communication in Networks”,

[9]Pratishtha Gupta, G.N Purohit, Varsha Bansal, “A Survey on Image Compression Techniques”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, Issue 8, August 2014

[10] S.KHAN, T.KHAN, M.NAEEM and N.AHMAD, “Run-Length Encoding Based Lossless Compressed Image Steganography”, Sindh Univ. Res. Jour. (Sci. Ser.) , Vol.47 (3) 541-544, 2015.

[11] Rahul Jain, Naresh Kumar, “Efficient data hiding scheme using lossless data compression and image steganography”, International Journal of Engineering Science and Technology (IJEST), Vol. 4 No.08, ISSN : 0975-5462, August 2012.

[12] Chintan R. Nagrecha, Prof. Prashant B. Swadas,“Steganography with Various Compression Algorithms to Improve Robustness and Capacity”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 5**,** ISSN: 2277 128X, May 2014.

[13] Anwar J. Moosa, “An Effective Compressed Image WithSteganoghraphy”, Journal of Babylon University/Pure and Applied Sciences/ No. (4)/ Vol. (21): 2013.

[14] Palak Mahajan, Dr. Ajay Koul, “CEET: A Compressed Encrypted & Embedded Technique for Digital Image Steganography”, IOSR Journal of Computer Engineering (IOSR-JCE), e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, PP 44-52, Ver. X (Mar-Apr. 2014).

[15] Kamlesh Lakhwani, Kiran Kumari, “KVL Algorithm: Improved Security & PSNR for Hiding Image In Image Using Steganography”, International Journal of Computational Engineering Research|, Vol. 03, Issue. 10, ISSN:2250-3005, October 2013.