REVERSIBLE DATA HIDING USING HISTOGRAM SHIFTING

by

SAI SAKETH NANDAGIRI

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2016

ACKNOWLEDGEMENTS

I would like to sincerely thank my thesis advisor Prof. K. R. Rao, as it is impossible to overstate my gratitude towards him. This work would not have been achievable without his help, support, guidance and encouragement. I would also like to thank my defense committee members for taking the time to review my work and offer their insightful comments and suggestions. I wish to thank the signal processing faculty members and staff for their assistance during my study at University of Texas at Arlington. Last but not the least, I wish to thank my family for their constant support and love without whom I would not be able to carry on and offer my regards to all of those who supported me in any respect during the completion of this project.

ABSTRACT

REVERSIBLE DATA HIDING USING

HISTOGRAM SHIFTING

Sai Saketh Nandagiri, MS

The University of Texas at Arlington, 2016

Supervising Professor: K. R. Rao

With the current state-of-the-art technology, intruders can easily gain access to sensitive information. To avoid this, different methods with improved security to hide information have been developed like, covert channels, hiding data in images, null ciphers, etc. The focus of this work is to enhance the security and data hiding capacity of the current state-of-the-art reversible data hiding algorithms.

The proposed data hiding method is designed such that the original image is losslessly restored after extraction of a payload. The proposed framework allows embedding of a binary bit stream into the host or cover image while retaining the quality of the image. The payload can be extracted at the receiver without any additional information (peak and zero values) about the stego image in which data is hidden. Experimental results indicate that the reversible data hiding scheme outperforms other approaches in the literature in terms of security and marked image quality.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

ABSTRACT iv

LIST OF FIGURES vii

LIST OF TABLES ix

Acronyms and Abbreviations x

Chapter 1 Introduction 11

1.1. Data Hiding 11

1.2. Thesis Outline 12

Chapter 2 Reversible Data Hiding 14

2.1. Introduction 14

2.2. Classification of Reversible Data Hiding Algorithms 15

2.2.1. Those for Fragile Authentication 15

2.2.2. Those for High Data Embedding Capacity 16

2.2.2.1 R-S Scheme 17

2.2.2.2 Difference Expansion Scheme 19

2.2.2.3 Integer Wavelet Transform Based Schemes 20

2.2.3. Those for Semi-Fragile Authentication 22

2.2.3.1 Patchwork-Based Scheme Using Modulo-256 Addition 24

2.2.3.2 Patchwork-Based Scheme without using Modulo-256 Addition 26

2.3. Summary 27

Chapter 3 Reversible Data Hiding using Histogram Shifting 29

3.1 Introduction 29

3.2 Algorithm-I 29

3.2.1 Illustration of Embedding Algorithm Using an Example with One Zero Point and One Peak Point 30
3.2.2 Pseudocode Embedding Algorithm 33
3.2.3 Pseudocode Extraction Algorithm 35
3.2.4 Embedding and Extraction Flow Charts 36
3.3 Algorithm-II 38
3.4 Summary 39
Chapter 4 Proposed method for Reversible Data Hiding 41
4.1 Introduction: 41
4.2 Steps: 42
Chapter 5 Results 46
Part A: 46
Part B: 54
Summary 57
Chapter 6 Conclusions and Future Work 58
Appendix 59
REFERENCES 60

LIST OF FIGURES

Figure 2-1: Comparison of different data hiding methods based on IWT for 21

Figure 2-2: (a) Original medical image, (b) Stego-image with severe salt-and-pepper noise. 746 information bits are embedded into the image of 512x512 with a PSNR of the stego-image versus the original image lower than 10 dB 26

Figure 3-1: Histogram of Lena image..... 30

Figure 3-2: Lena image: (a) original, and (b) marked (PSNR = 48:2 dB)..... 32

Figure 3-3: Histogram of the marked (or) stego Lena image..... 32

Figure 3-4: Data Embedding Algorithm 37

Figure 3-5: Data extracting algorithm for one pair of maximum and minimum points..... 37

Figure 4-1: Proposed method to hide data..... 41

Figure 4-2: Original image..... 44

Figure 4-3: Original image split into four 256 × 256 blocks 44

Figure 4-4: Histogram of individual blocks (see fig.4-3) 45

Figure 4-5: Histogram of inverted blocks (see fig.4-4)..... 45

Figure 5-1 46

Figure 5-2 47

Figure 5-3 47

Figure 5-4 48

Figure 5-5 49

Figure 5-6 49

Figure 5-7 50

Figure 5-8 50

Figure 5-9 51

Figure 5-10 ..... **51**

Figure 5-11 ..... **52**

Figure 5-12 ..... **52**

Figure 5-13 ..... **53**

Figure 5-14 ..... **53**

Figure 5-15 ..... **54**

LIST OF TABLES

Table 5-1 Results for Lena image **54**

Table 5-2 Results for Barbara image **55**

Table 5-3 Results for Goldhill image **55**

Table 5-4 Results for Airplane image **56**

Table 5-5 Results for Tiffany image **57**

Acronyms and Abbreviations

- DCT – Discrete Cosine Transform
- HS-RDH – Histogram shifting based RDH
- ICMP – Internet Control Message Protocol
- IWT – Integer Wavelet Transform
- JPEG – Joint Photographic Experts Group
- LSB – Least Significant Bit
- PSNR – Peak Signal to Noise Ratio
- QIM – Quantization Index Modulation
- RDH - Reversible Data Hiding

Chapter 1

Introduction

Protection of digital multimedia content, over networks has always been a challenge for researchers and engineers. These days, internet offers secure data communication for important messages, secret information, images and documents. But with improved hacking tools, any secure communication can be breached effortlessly. To overcome this problem, two methods have been introduced; cryptography and steganography. In cryptography, the information hidden in the encrypted message can only be extracted with the private key. Even though the attacker gets access to an encryption message, it is not possible to retrieve the content. But if the private key is stolen or broken, this method will no longer protect the data. Data can also be hidden behind a cover image such that an observer is not aware of its existence. This type of data hiding is called steganography.

1.1. Data Hiding

So far, many algorithms [1][3][7][8][9][11][13][17] for data hiding have been proposed but most of them fail to recover the cover image after data extraction. However, in some medical and military applications, it is desired that the original cover media to be recovered losslessly after data extraction. The marking techniques satisfying this requirement are referred to as reversible data hiding techniques.

At present, fragile reversible data-hiding techniques can be conducted in three domains, that is, the spatial domain, the transformed domain, and the compressed domain. Semi-fragile data-hiding techniques can be conducted only in the spatial and

transformed domains, because high-level information about the structure of the data stream usually is not available in compressed streams with embedded secret data. In the spatial domain, the values of the pixels of the cover image are altered directly to embed the data. In the transformed domain, the cover image should be preprocessed by a transform, such as the integer wavelet transform (IWT), the discrete cosine transform (DCT), the discrete wavelets transform, or the discrete Fourier transform, to get the frequency coefficients. Then, the frequency coefficients are modified slightly to embed data, and the stego image can be obtained by using the modified frequency coefficients. In the compression domain, the compression code is altered to embed the data.

Most of the published papers [8][9] related to the reversible data-hiding schemes address fragile techniques because it is difficult to recover the cover image losslessly if the stego image undergoes a lossy modification, such as JPEG compression. A good fragile reversible data-hiding technique should provide high-hiding capacity and low distortion simultaneously. So, the main purpose of fragile techniques is to provide a secure channel to protect communication among legitimate users.

1.2. Thesis Outline

The objective of this research is to improve the security and hiding capacity of the existing reversible data hiding algorithms using histogram shifting. In the conventional data hiding algorithms, the peak point in the histogram is altered after the data is embedded. This demands transmission of original peak value along with the data to be hidden to the receiver to extract the payload. The designed system in this research avoids this problem by not using the peak value for data hiding. The above issue will be further elaborated in the upcoming chapters.

Chapter 2 converses the history and background of reversible data hiding. The basic structure and properties of a general data hiding algorithm are also presented along with present applications of data hiding algorithms.

In chapter 3, algorithms based on RDH-HS are introduced. In RDH-HS, the peak value in the histogram of an image is manipulated to store binary data, i.e. 1 or 0. The hiding capacity and the quality of the stego image are also discussed.

In chapter 4, the proposed algorithm to hide data is explained in detail along with future scope for reversible data hiding.

Chapter 5 presents the results of the proposed algorithm for different test images.

Chapter 6 discusses about the conclusions and future work in the field of reversible data hiding using histogram shifting.

Chapter 2

Reversible Data Hiding

2.1. Introduction

Digital watermarking, often referred to as data hiding, has recently been proposed as a promising technique for the information assurance. Owing to data hiding, however, some permanent distortion may occur and hence the original cover medium may not be able to be reversed exactly even after the hidden data have been extracted out. Following the classification of data compression algorithms, this type of data hiding algorithms can be referred to as lossy data hiding. Here, let us examine three major classes of data hiding algorithms. With the most popularly utilized spread-spectrum watermarking techniques, either in DCT domain [2] or in block 8x8 DCT domain [4], round-off error and/or truncation error may take place during data embedding. As a result, there is no way to reverse the stego-media back to the original without distortion. For the least significant bit-plane (LSB) embedding methods, the bits in the LSB are substituted by the data to be embedded and the bit-replacement is not memorized. Consequently, the LSB method is not reversible. With the third group of frequently used watermarking techniques, called quantization index modulation (QIM) [6], quantization error renders lossy data hiding.

In the next section, different reversible data hiding techniques that have appeared in the literature over the past several years are explained briefly.

2.2. Classification of Reversible Data Hiding Algorithms

Reversible data hiding algorithms can be classified into three categories: 1st, those for fragile authentication, 2nd, those for high embedding capacity, and 3rd, those for semi-fragile authentication. Among each category, one or two prominent algorithms are selected as representative. Their fundamental idea and scheme to achieve reversibility, and their performance are discussed in the following sub-sections.

2.2.1. Those for Fragile Authentication

The first several reversible data hiding algorithms developed at the early stage belong to this category. Since fragile authentication does not need much data to be embedded in a cover medium, the embedding capacity in this category is not large, normally between 1k to 2k bits. For a typical 512 × 512 gray scale image, this capacity as defined in Eq 2.1 is equivalent to a data hiding rate from 0.0038 bits per pixel (bpp) to 0.0076 bpp.

$$\text{Capacity} = \text{number of bits to be hidden}/\text{no. of pixels in the image} \qquad (2.1)$$

In this category, Honsinger et al patent in 2001 [6] describes in detail a reversible data hiding technique used for fragile authentication. Their method is carried out in the image spatial domain by using modulo-256 addition. In the embedding, $Iw = (I + W)$ mod 256, where Iw denotes the marked image, I an original image, W is the payload derived from the hash function of the original image. In the authentication side, the payload W can be extracted from the marked image by subtracting the payload from the marked

image, thus reversibly recovering the original image. By using modulo-256 addition, the issue of over/underflow is avoided. Here, by over/underflow, it is meant that grayscale values either exceeding its upper bound (overflow) or its lower bound (underflow). For instance, for an 8-bit gray image, its gray scale ranges from 0 to 255. The overflow refers to grayscale exceeds 255, while the underflow refers to below 0. It is clear that either case will destroy reversibility. Therefore this issue is often a critical issue in reversible data hiding. Using modulo-256 addition can avoid over/underflow on the one hand. On the other hand, however, the stego-image may suffer from the salt-and pepper noise during possible grayscale flipping over between 0 and 255 in either direction due to the operation of modulo-256 addition. The effect caused by salt-and pepper noise will become clear when an algorithm also using modulo-256 addition is discussed in the third category.

2.2.2. Those for High Data Embedding Capacity

All the reversible data hiding techniques in the first category aim at fragile authentication, instead of hiding large amount data. As a result, the amount of hidden data is rather limited and may not be suitable for applications such as covert communications and medical data systems. Hence, Goljan et al [10] presented a first reversible data hiding technique, referred to as R-S scheme, which is suitable for the purpose of having high data embedding capacity. Later, a difference expansion scheme was developed by Tian [15], which has greatly advanced the performance of reversible data hiding in terms of data embedding capacity versus PSNR of marked (or) stego images with respect to original images. Recently, some integer wavelet transform based reversible data hiding schemes have been developed by Xuan et al [16, 14], which have

demonstrated superior performance over that reported in [15]. These representative schemes are presented in this section.

2.2.2.1 R-S Scheme

The mechanism of this scheme is described as follows. The pixels in an image are grouped into non-overlapped blocks, each consisting of a number of adjacent pixels. For instance, it could be a horizontal block consisting of four consecutive pixels. A discrimination function that can capture the smoothness of the groups is established to classify the blocks into three different categories, Regular, Singular and Unusable. An invertible operation F can be applied to groups. That is, it can map a block from one category to another as F(R) = S, F(S) = R, and F(U) = U. It is invertible since applying it to a block twice produces the original block. This invertible operation is hence called flipping F. An example of the invertible operation F can be the permutation between 0 and 1, 2 and 3, 3 and 4, and so on. This is equivalent to flipping the least significant bit (LSB). Another example is the permutation between 0 and 2, 1 and 3, 4 and 6, and so on, i.e., flipping the second LSB. Apparently, the strength of the latter flipping is stronger than the former. The principle to achieve reversible data embedding lies in that there is a bias between the number of regular blocks and that of singular blocks for most of the images. This is equivalent to saying that there is a redundancy, and some space can be created by lossless compression. Together with some proper bookkeeping scheme, one can achieve reversibility.

The proposed algorithm first scan a cover image block-by-block, resulting in a so called RS-vector formed by representing, say, an R-block by binary 1 and an S-block by binary 0 with the U groups simply skipped. Then the algorithm losslessly compresses this

RS-vector − as an overhead for bookkeeping usage in reconstruction of the original image. By assigning a binary 1 and 0 to R and S blocks, respectively, one bit can be embedded into each R or S block. If the bit to-be-embedded does match the type of a block under consideration, the flipping operation F is applied to the block to obtain a match. The actual embedded data consists of the overhead and the watermark signal (pure payload). In data extraction, the algorithm scans the marked image in the same manner as in the data embedding. From the resultant RS-vector, the embedded data can be extracted. The overhead portion will be used to reconstruct the original image, while the remaining portion is the payload.

While it is novel and successful in reversible data hiding with a large embedding capacity, the amount of data that can be hidden by this technique is still not large enough for some applications such as covert communications. From what is reported in [10], the estimated embedding capacity ranges from 0.022 bits per pixel to 0.17 bits per pixel when the embedding strength is six and the PSNR of the marked image versus the original image is about 36.06 dB. Note that the embedding strength six is rather high and there are some block artifacts in the marked image generated with this embedding strength. On one hand, this embedding capacity is much higher than that in the first category discussed in the previous subsection. On the other hand, however, it may be not high enough for some applications. This limited embedding capacity is expected because each block can at most embed one bit, U blocks cannot accommodate data, and the overhead is necessary for reconstruction of the original image. Another problem with this method is that when the embedding strength increases, the embedding capacity will increase, at the same time the visual quality will drop. Often, block artifacts will take place at this circumstance, thus causing the visual quality of a marked image to decrease.

2.2.2.2 Difference Expansion Scheme

Tian presented a promising high capacity reversible data embedding algorithm in [15]. In the algorithm, two techniques are employed, i.e., difference expansion and generalized least significant bit embedding, to achieve a very high embedding capacity, while keeping the distortion low. The main idea of this technique is described below. For a pair of pixel values x and y , the algorithm first computes the integer average l and difference h of x and y , where $h = x - y$. Then h is shifted to the left-hand side by one bit and the to-be-embedded bit b is appended into the LSB. This is equivalent to $h' = 2 \times h + b$, where h' denotes the expanded difference, which explains the term of difference expansion. Finally the new x and y, denoted by, x' and y' respectively, are calculated based on the new difference values h' and the original integer average value l. In this way, the stego-image is obtained. To avoid over/underflow, the algorithm only embeds data into the pixel pairs that shall not lead to over/underflow. Therefore, a two-dimensional binary bookkeeping image is losslessly compressed and embedded as overhead.

Note that the above-mentioned relationship between the pair of integers x and y versus the pair of integers l and h is implemented in the following manner.

$$\begin{aligned} l &= \lfloor 0.5 \times (x + y) \rfloor & x &= l + \lfloor 0.5 \times (h + l) \rfloor \\ h &= x - y & y &= l - \lfloor 0.5 \times h \rfloor \end{aligned} \tag{2.2}$$

where the floor operation is used. According to the integer Haar transform, it is reversible between these two integer pairs. Apparently, the reversible transformation between integers avoids round-off error. This together with the bookkeeping data mentioned above guaranteed reversibility. It has been reported in [15] that the embedding capacity

achieved by the difference expansion method is much higher than that achieved by [10]. This does not come with surprise since intuitively each pair of pixels can possibly embed one bit, while only each block of pixels can possibly embed one bit.

2.2.2.3 Integer Wavelet Transform Based Schemes

Xuan et al proposed three high capacity reversible data hiding algorithms based on the integer wavelet transform (IWT) [12][16][18]. These three algorithms have three features in common. The first is that they are all implemented in the IWT domain. IWT as a WT is known to be able to de-correlate a signal well in the transform domain. Its main feature is that it is similar to human vision system (HVS). The WT can be implemented efficiently by using a lifting scheme. The IWT can further ensure the reversible forward wavelet transform and the inverse wavelet transform. For these reasons, the IWT has been used in JPEG2000 for lossless compression. It is shown in Xuan et al algorithm that IWT plays an important role in reversible data hiding. The second feature is that these algorithms all contain a pre-processing stage, histogram modification, in order to prevent overflow and underflow. That is, an efficient scheme has been developed to shrink the histogram towards the center, leaving two ends empty. Consequently, the perturbation caused by modification of selected IWT coefficients will not cause overflow and underflow. For reversibility, the histogram modification parameters need to be embedded as overhead. Because of the efficiency of the modification scheme [27], the overhead is not heavy. The third feature is that all of three algorithms embed data in IWT coefficients of high frequency sub-bands. This is because the modification of coefficients in these sub-bands will be imperceptible if the magnitude of the modification is not large.

The first algorithm [12, 27] losslessly compresses some selected middle bit-planes of IWT coefficients in high frequency sub-bands to create space to hide data. Since the bias between binary 1 and 0 in the bit-planes of IWT high frequency coefficients becomes much larger than that in the spatial domain, this method achieves rather higher embedding capacity than [19], that is the counterpart of this algorithm in the spatial domain.

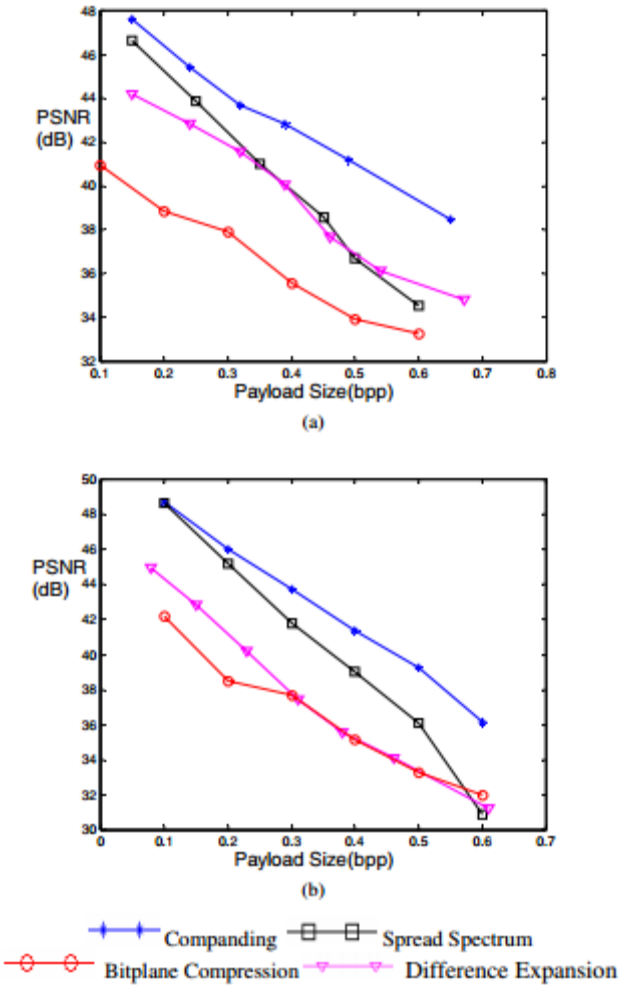

Figure 2-1: Comparison of different data hiding methods based on IWT for Lena (a) and Baboon (b)

The second algorithm [16] uses a spread spectrum method to hide data in the IWT coefficients in high frequency sub-bands. Pseudo bits are used to indicate those coefficients that are not selected for data embedding, thus saving overhead data. As a result, this method is more efficient than the bit-plane compression scheme, described above. The third method [18] uses companding technique for data embedding, which was inspired by [14]. Based on the study of statistical distribution of IWT coefficients in high frequency sub-bands, a piecewise linear companding function is designed. The performance of these three algorithms, applied to two typical test images: Lena and Baboon, are shown in figure 2-1 in terms of data embedding capacity versus PSNR. It is clear that the IWT-based companding algorithm performs best. This can be explained from an investigation of the change in magnitude of the coefficients and how many coefficients are required to embed one bit during the data embedding by these three algorithms. It can be shown that the companding algorithm causes the least amount of change in the selected IWT coefficients among the three algorithms, followed by the spread-spectrum algorithm. Both the spread-spectrum and companding algorithms can embed almost one bit into one IWT high frequency coefficient. Note that both IWT based companding and spread-spectrum algorithms have outperformed the difference expansion algorithm, discussed above in this section. It is noticed that more and more advanced algorithms in this category are being developed. One recent example is shown in [20].

2.2.3. Those for Semi-Fragile Authentication

For multimedia, content-based authentication makes more sense than representation based authentication. This is because the former, often called semi-fragile authentication, allows some incidental modification, say, compression within a reasonable extent, while the latter, called fragile authentication, does not allow any alteration occurred to stego-media, including compression. For instance, when an image goes through JPEG compression with a high quality factor, the content of this image is considered unchanged from the original. Hence, it makes sense to claim this compressed image as authentic. For the purpose of semi-fragile authentication, it is required that reversible data hiding algorithms are robust to compression, maybe called semi-fragile reversible data hiding or robust reversible data hiding. This can be further illustrated by the following scenario. In a newly proposed JPEG2000 image authentication framework [28], both fragile and semi-fragile authentications are included. Within the semi-fragile authentication, both cases of lossy and lossless compressions are considered. In this framework, some features corresponding to an image below a pre-specified compression ratio are first identified. By “corresponding” it is meant that these features will remain as long as the compression applied to the image is below this pre-specified compression ratio. The digital signature of these features is reversibly embedded into the image. Then in the verification stage, if the marked image has not been changed at all, the hidden signature can be extracted and the original image can be recovered. The matching between the extracted signature and the signature generated from the reconstructed (left-over) image renders the image authentic. If the marked image has been compressed with a compression ratio below the pre-specified one, the original image cannot be recovered due to the lossy compression applied, but the hidden signature can still be recovered without error and verify the compressed image as authentic. Obviously, if the marked image goes through a compression with the compression ratio higher than the pre-

specified ratio will render the image un-authentic. Any malicious attack will render the attacked image non-authentic owing to the resultant mismatch between the extracted signature and the signature generated from the received image after the hidden data extraction. A robust lossless data hiding algorithm is indeed necessary for this framework. In general, robust lossless data hiding can be used in a lossy environment.

2.2.3.1 Patchwork-Based Scheme Using Modulo-256 Addition

De Vleeschouwer et al. [21] proposed a reversible data hiding algorithm based on patchwork theory [22], which has certain robustness against JPEG lossy compression. This is the only existing robust reversible data hiding algorithm against JPEG compression. In this algorithm, each hidden bit is associated with a group of pixels, e.g., a block in an image. Each group is pseudo-randomly divided into two subsets of an equal number of pixels. They are defined as Zones A and B. The histogram of each zone is mapped into a circle in the following way. That is, each point on the circle is indexed by the corresponding luminance, and the number of pixels assuming the luminance will be the weight of the point. The mass center of each zone can then be determined. It is observed that in most cases the vectors pointing from the circle center to the mass center of Zones A and B are close (almost equal) to each other because the pixels of Zone A and Zone B are highly correlated for most images. Considering a group, rotating these two vectors in two opposite directions by a small quantity, say, rotating the vector of Zone A counter-clockwise and rotating the vector of Zone B clockwise allows for embedding binary 1, while rotating the vector of Zone A clockwise, and the vector of Zone B counter-clockwise embeds a binary 0. As to the pixel values, rotation of the vector corresponds to a shift in luminance. In data extraction, the angles of the mass center vectors of both

Zone A and Zone B versus the horizontal direction are first calculated, and the difference between these two angles are then determined. A positive difference represents a binary 1, while a negative a binary 0. One major element of this algorithm is that that it is based on the patchwork theory. That is, within each zone, the mass center vector's orientation is determined by all the pixels within this zone. Consequently, the algorithm is robust to image compression to certain extent. Another major element of this algorithm lies in its use of modulo-256 addition to avoid overflow and underflow, thus achieving reversibility. Consequently, as pointed out in section 2.2.1, this algorithm will suffer from the salt-and-pepper noise. One example from our extensive investigation is presented in figure 2.2. In the stego-medical image, the severe salt-and-pepper noise is clear. The PSNR of a stego image versus the original image is below 10 dB when 476 information bits are embedded into this 512x512 image. The salt and pepper noise may be severe for non-medical color images as well. This algorithm was applied to eight JPEG2000 test color images. Four of the eight images suffered from severe salt-and-pepper noise, while the other four had less severe salt-and-pepper noise. The PSNR can less than 20 dB when severe noise exists such as when 1412 information bits are embedded into a color image of 1536x1920x24.

From the above investigation, it can be concluded that all reversible data hiding algorithms based on modulo-256 addition to avoid overflow and underflow, say, in [5] and [20], cannot be applied to many real applications, and hence should be avoided.

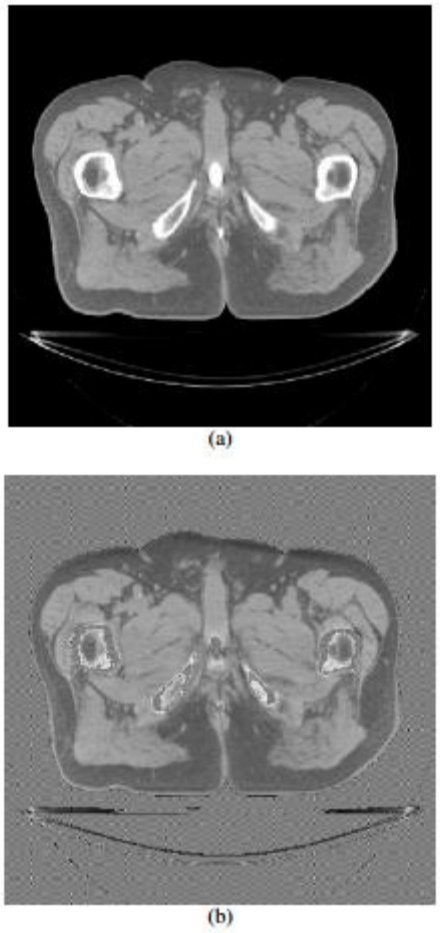

Figure 2-2: (a) Original medical image, (b) Stego-image with severe salt-and-pepper noise. 746 information bits are embedded into the image of 512x512 with a PSNR of the stego-image versus the original image lower than 10 dB

2.2.3.2 Patchwork-Based Scheme without using Modulo-256 Addition

Realizing that modulo-256 addition, though can prevent overflow and underflow and hence achieve reversibility, causes stego-images suffer from annoying salt and pepper noise, Ni et al. have developed a new reversible data hiding scheme that does not use

modulo-256 addition [23]. It is based on the patchwork theory to achieve the semi-fragility in data hiding. Specifically, it identifies a statistical quantity within a block of pixels which is robust to minor alteration of pixel values within the block, and manipulates it to embed a bit into the block. Together with additional measures including error correction coding and permutation, it thus achieves semi-fragility (or in other words, robustness). The reversibility is gained by using a novel strategy in data embedding. That is, it classifies a block of pixels into four different categories according to the distribution of pixel grayscale values within the block. For blocks in different categories, a different embedding strategy is used. This novel semi-fragile reversible data hiding algorithm has achieved superior performance over the semi-fragile reversible data hiding algorithm using modulo-256 addition. It was reported in [23] that their method has been applied to all eight medical test images, including that shown in figure 2-2 (a), to embed the same amount of data (746 information bits in 512x512 images). In all of eight stego-images, there is no salt-and-pepper noise at all. The PSNR of all of eight medical images are above 40 dB, indicating a substantial improvement in the quality of stego-images. Another novel semi-fragile reversible data hiding algorithm using a similar idea as in [23] was implemented in integer wavelet transform domain has been reported in [24]. Some special measures have been taken to avoid overflow and underflow. The framework has been included in the Security Part of JPEG2000 (known as JPSEC), CD 1.0 in April 2004 [28].

2.3. Summary

This chapter presents an investigation on the development of the existing reversible data hiding techniques. These techniques are classified into three different

categories. The principles, merits and drawbacks of each category are discussed. It is shown that the reversible data hiding opens a new door to link two groups of data: cover media data and to-be-embedded data. This will find wide applications in information assurance such as authentication, secure medical data system, and intellectual property protection to name a few. In the next chapter, reversible data hiding using histogram shifting is introduced along with its merits and demerits.

Chapter 3

Reversible Data Hiding using Histogram Shifting

3.1 Introduction

In this chapter, the first reversible data hiding algorithm using histogram shifting proposed by Ni et al. [25] is presented in detail. This method can embed a large amount of data (5–80 kb for a 512 x 512, 8 bits quantized grayscale image) while keeping a very high visual quality for all natural images, specifically, the PSNR of the marked image or stego image versus the original image is guaranteed to be higher than 48 dB. It uses the zero (v_zero) or the minimum point of the histogram (defined below) and slightly modifies the pixel grayscale values to embed data. This technique can be applied to virtually all types of images. Up to now, it has been successfully tested on different types of images, including some commonly used images, medical images, texture images, aerial images, and all of the 1096 images in CorelDRAW database. The computation of the proposed technique is quite simple and the execution time is rather short. Although the proposed lossless data hiding technique is applied to still images, it is also applicable to videos which consist of a sequence of images.

3.2 Algorithm-I

The famous “Lena” image is used as an example to illustrate the algorithm. Then the data embedding and extracting of the proposed algorithm are presented in terms of pseudocode. Finally, some important issues including data embedding capacity are addressed. For a given grayscale image, say, the Lena image of size 512 x 512 with 8 bits quantized, the histogram is generated as shown in fig. 3.1.

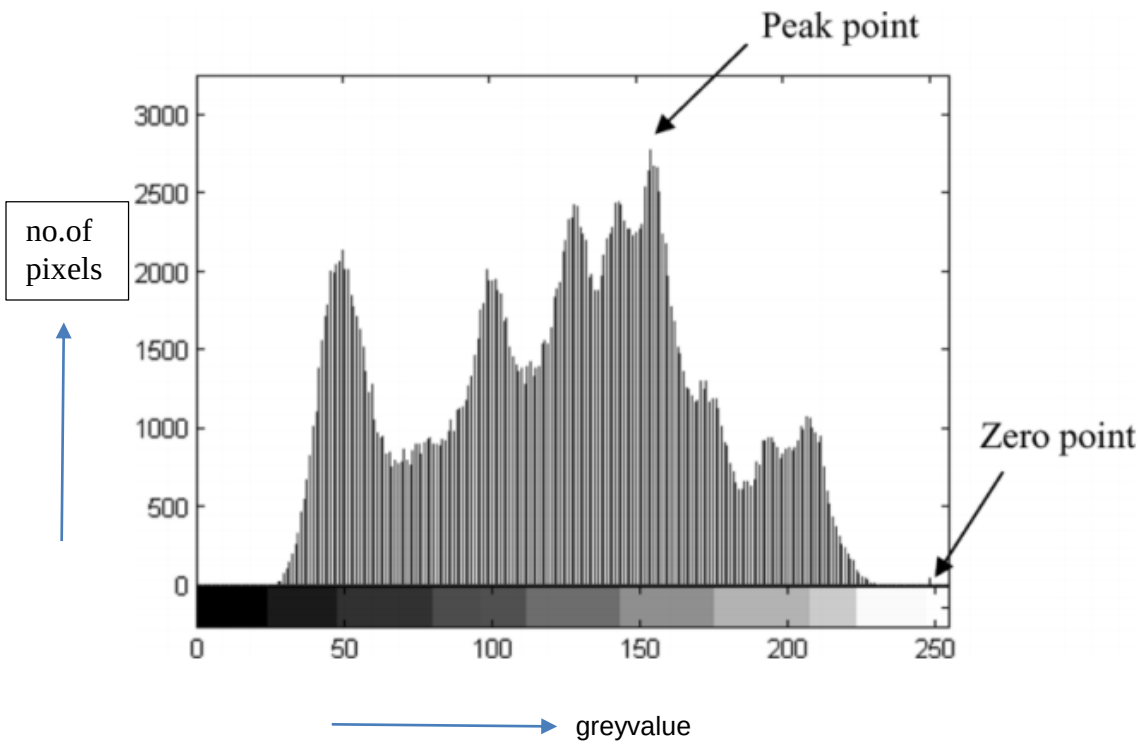

Figure 3-1: Histogram of Lena image

3.2.1 Illustration of Embedding Algorithm Using an Example with One Zero Point and One Peak Point

1) In the histogram, first find the location zero point and peak point. A zero point corresponds to the grayscale value which no pixel in the given image assumes, e.g. $h(255)$, as shown in fig. 3.1. A peak point corresponds to the grayscale value which the maximum number of pixels in the given image assumes, e.g. $h(154)$, as shown in fig. 3.1. For the sake of notational simplicity, only one zero point and one peak point are used in this example to illustrate the principle of the algorithm. The objective of finding the peak point is to increase the embedding capacity as large as possible since in this algorithm, the number of bits that can

be embedded into an image equals to the number of pixels which are associated with the peak point. The implementation of the proposed algorithm with two or more pairs of zero and peak points is further discussed later in this section.

2) The whole image is scanned in a sequential order, say, row-by-row, from top to bottom, or, column-by-column, from left to right. The grayscale value of pixels between 155 (including 155) and 254 (including 254) is incremented by “1”. This step is equivalent to shifting the range of the histogram, [155 254], to the right-hand side by 1 unit, leaving the grayscale value 155 empty.
3) The whole image is scanned once again in the same sequential order. Once a pixel with grayscale value of 154 is encountered, the to-be-embedded data sequence is checked. If the corresponding to-be-embedded bit in the sequence is a binary “1,” the pixel value is incremented by 1. Otherwise, the pixel value remains intact. (Note that this step may be included in Step 2, described above. For illustration purposes, the embedding algorithm is presented in these three steps.)

The above three steps complete the data embedding process. Now the data embedding capacity of this algorithm can be observed when only one pair of zero and peak points is used equals the number of pixels that assume the grayscale value of the peak point as mentioned in Step 1. Fig. 3.2 shows the original and the marked Lena image, respectively. The histogram of the marked Lena image is displayed in fig. 3.3. Note that the peak point, 154, shown in fig. 3.1 has disappeared.

(a) (b)

Figure 3-2: Lena image: (a) original, and (b) marked (PSNR = 48:2 dB)

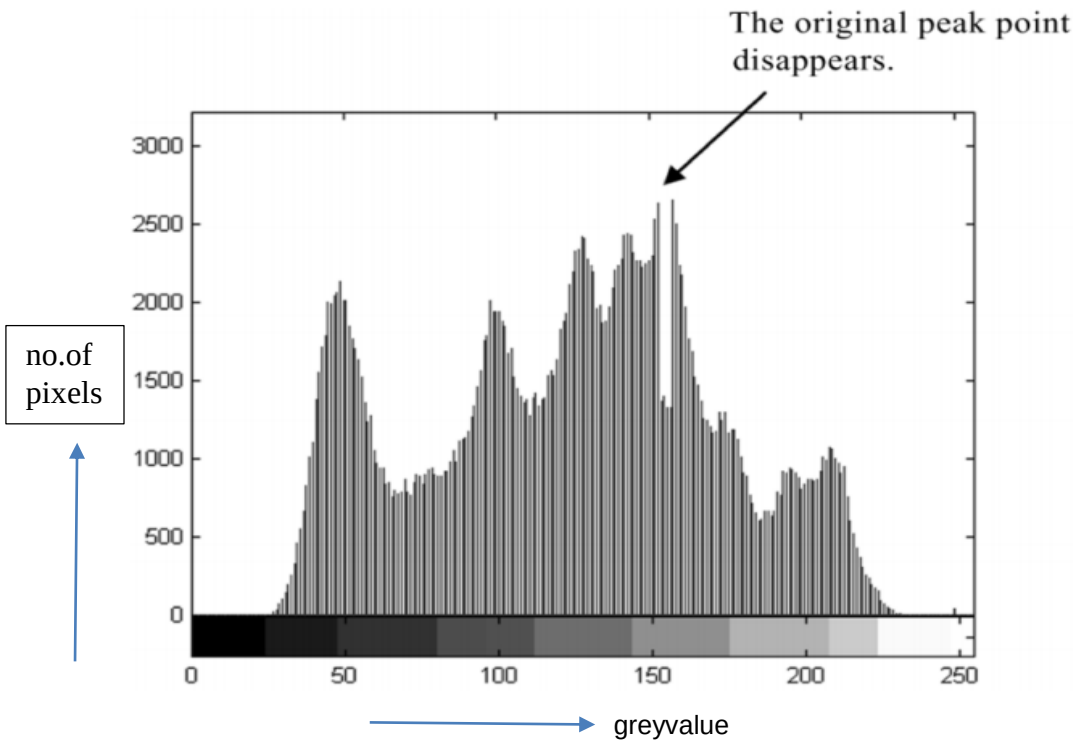

Figure 3-3: Histogram of the marked (or) stego Lena image

3.2.2 Pseudocode Embedding Algorithm

Note that zero point defined above may not exist for some image histograms. The concept of minimum point is hence more general. The term minimum point, implies that a grayscale value, *b*, that a minimum number of pixels assume this value, i.e., $h(b)$ is minimum. Accordingly, the peak point discussed above is referred to as the maximum point. Therefore, in the following discussion, the terms maximum and minimum points are used.

A. *Pseudocode Embedding Algorithm with One Pair of Maximum and Minimum Points: For an M x N image, each pixel grayscale value* $x \in [0,255]$.

   I. Generate its histogram *H(x)*.

   II. In the histogram *H(x)*, find the maximum point *h(a),* $a \in [0,255]$ and the minimum point zero *h(b),* $b \in [0,255]$.

   III. If the minimum point *h(b) > 0*, recode the co-ordinate (i,j) of those pixels and the pixel grayscale value *b* as overhead bookkeeping information (referred to as overhead information for short). Then set *h(b) = 0*.

   IV. Without loss of generality, assume *a* < *b*. Move the whole part of the histogram *H(x)* with $x \in [a, b]$ to the right by 1 unit. This means that all the pixel grayscale values (satisfying $x \in [a, b]$) are added by 1.

   V. Scan the image, once meet the pixel (whose grayscale value is *a*), check the to-be-embedded bit. If the to-be-embedded bit is “1”, the pixel grayscale value is changed to *a + 1*. If the bit is “0”, the pixel value remains *a*.

B. *Actual Data Embedding Capacity (Pure Payload): By using the algorithm mentioned in the above section A, the actual data embedding capacity, C, is calculated as follows:*

$$C = h(a) - O \tag{3.1}$$

Where *O* denotes the amount of data used to represent the overhead information. It is also referred to as pure payload.

Clearly, if the required payload is greater than the actual capacity, more pairs of maximum point and minimum point must be used. The embedding algorithm with multiple pairs of maximum point and minimum point is presented in section C.

C. *Pseudocode Embedding Algorithm With Multiple Pairs of Maximum and Minimum Points:*

Without loss of generality, only a pseudocode embedding algorithm for the case of three pairs of maximum and minimum points is presented below. It is straightforward to generate this code to handle the cases where any other number of multiple pairs of maximum and minimum points is used.

For a *M x N* image with pixel grayscale values $x \in [0,255]$.

I. Generate its histogram H(x).

II. In the histogram H(x), find three minimum points $h(b_1)$, $h(b_2)$, $h(b_3)$. Without loss of generality, assume three minimum points satisfy the following condition: $0 < b_1 < b_2 < b_3 < 255$.

III. In the intervals of $(0, b_1)$ and $(b_3,255)$, find the maximum point $h(a_1)$, $h(a_3)$, respectively, and assume $a_1 \in (0, b_1)$, $a_3 \in (b_3, 255)$.

IV. In the intervals (b_1, b_2) and (b_2, b_3), find the maximum points in each interval. Assume they are h(a_{12}), h(a_{21}), $b_1 < a_{12} < a_{21} < b_2$ and $h(a_{23}), h(a_{32}), b_2 < a_{23} < a_{32} < b_3$.

V. Find a point having a larger histogram value in each of the following three maximum point pairs $\big(h(a_1), h(a_{12})\big), \big(h(a_{21}), h(\,a_{23})\big), and\ \big(h(a_{32}), h(a_3)\big)$, respectively. Without loss of generality, assume $h(a_1), h(a_{23}), h(a_3)$ are the three selected maximum points.

VI. Then $\big(h(a_1), h(b_1)\big), \big(h(a_{23}), h(\,b_2)\big),\ \big(h(a_3), h(b_3)\big)$ are the three pairs of maximum and minimum points. For each of these three pairs, apply steps III to V described in section 3.2.2 (A). That is, each of these three pairs are treated as a case of one pair of maximum and minimum points.

It is apparent that this embodiment of the proposed algorithm may result in suboptimal performance in terms of the actual data embedding capacity versus the visual quality of the marked image.

3.2.3 Pseudocode Extraction Algorithm

For the sake of brevity, only the simple case of one pair of a minimum point and a maximum point is described here because, as shown above, the general cases of multiple pairs of maximum and minimum points can be decomposed as a few one pair cases. That is, the multiple pair case can be treated as the multiple repetition of the data extraction for one pair case.

Assume the grayscale value of the maximum point and the minimum point are *a* and *b*, respectively. Without loss of generality, assume *a* < *b (‘a’ lies*

on the left side of 'b' in the histogram). The marked image is of size *M x N*, each pixel grayscale value $x \in [0,255]$.

I. Scan the marked image in the same sequential order as that used in the embedding procedure. If a pixel with its grayscale value a+1 is encountered, a bit "1" is extracted. If a pixel with its value is encountered, a bit "0" is extracted.

II. Scan the image again, for any pixel whose grayscale value $x \in (a, b]$, the pixel value x is subtracted by 1.

III. If there is overhead bookkeeping information found in the extracted data, set the pixel grayscale value (whose coordinate (*i,j*) is saved in the overhead) as b.

In this way, the original image can be recovered without any distortion.

3.2.4 Embedding and Extraction Flow Charts

In summary, the proposed reversible data hiding and extraction algorithms can be illustrated by the flow charts shown in figs. 3.4 and 3.5, respectively.

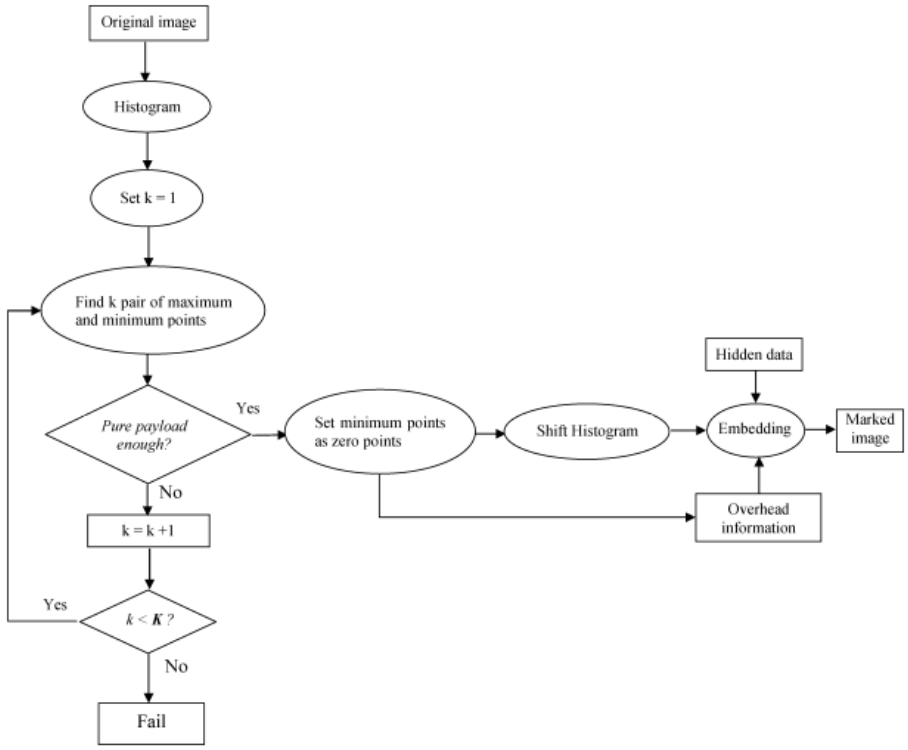

Figure 3-4: Data Embedding Algorithm

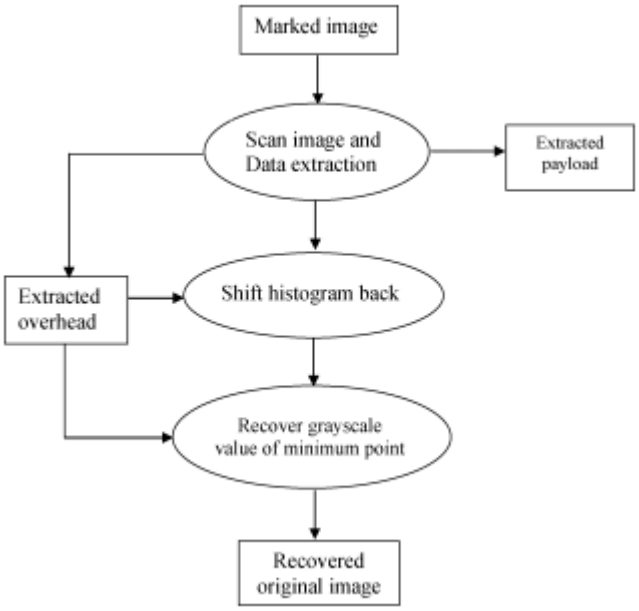

Figure 3-5: Data extracting algorithm for one pair of maximum and minimum points.

3.3 Algorithm-II

The problem with data hiding using the method proposed by Ni et al [25] is that, the histogram of the stego image changes significantly which is undesirable. In the method proposed by Pan et al [26], data is embedded in an image using localization method. This method relies on the idea that, when the data is embedded directly in the cover image, the variation in the histogram of the stego image is high. This can be minimized by splitting the cover image into non-overlapping blocks and then embedding the data into each block. Another added advantage is that, image dependent information like v_{peak} and v_{zero} need not be transmitted because the values adjacent to the peak are used to embed data. So the peak point, v_{peak}, remains unchanged after embedding the data. A brief overview of the embedding procedure using the localization method is described below.

The original (or) cover image, f, is of size $X \times Y$ and 8-bits quantized grayscale image. The cover image f is divided into non-overlapping blocks B of size *s x s*. Each block B is considered as a small cover image and the payload is embedded into each block. For each block B, there exists a peak v_{peak}. During the embedding, the peak point v_{peak} will not be changed. At the extraction end, the peak point v_{peak}, is used to extract secret bits and recover the cover image. For each point in block B, the value of the point between v_{peak}+2 and 254 is increased by 1; the value of the point between 1 and v_{peak}-2 is decreased by 1.

$$B'(x,y) = \begin{cases} B(x,y)+1, & v_{peak}+2 \le B(x,y) \le 254 \\ B(x,y)-1, & 1 \le B(x,y) \le v_{peak}-2 \end{cases} \tag{3.2}$$

If the value of the pixel equals $v_{peak} + 1$, and the embedded secret bit $p(l)\ is\ '0'$, the value of the pixel is unchanged; the value of the pixel is decreased by 1 if the embedded secret bit $p(l)\ is\ '1'$.

$$B'(x,y) = \begin{cases} B(x,y) + 1, & B(i,j) = v_{peak} + 1, p(l) = 1 \\ B(x,y) \quad , & B(i,j) = v_{peak} + 1, p(l) = 0 \end{cases} \tag{3.3}$$

If the value of the pixel equals $v_{peak} - 1$, and the embedded secret bit $p(l)\ is\ '0'$, the value of the pixel is unchanged; the value of the pixel is decreased by 1 if the embedded secret bit $p(l)\ is\ '1'$.

$$B'(x,y) = \begin{cases} B(x,y) - 1, & B(i,j) = v_{peak} - 1, p(l) = 1 \\ B(x,y) \quad , & B(i,j) = v_{peak} - 1, p(l) = 0 \end{cases} \tag{3.4}$$

If the value of the point equals v_{peak}, the value is unchanged.

$$B'(x,y) = B(x,y),\ if\ B(x,y) = v_{peak} \tag{3.5}$$

Where $B(x,y)$ is the value of the pixel at position (x,y) of the block B, $B'(x,y)$ is the value at position (x,y) of the embedded block B'. When all the blocks are processed, an embedded cover image (or) stego image $\tilde{f}$ is generated.

3.4 Summary

In this chapter, the first reversible data hiding algorithm using histogram shifting is discussed in detail and the flowcharts are presented in figs 3.4 and 3.5. Due to reduction in the PSNR of the stego image when Ni et al approach is used, another algorithm based

on the localization method which improves the PSNR of the stego image is briefly explained. In the upcoming chapter the proposed algorithm to hide data is discussed.

Chapter 4

Proposed method for Reversible Data Hiding

4.1 Introduction:

As mentioned in the abstract, the main objective of this thesis is to improve the security and PSNR of the stego image. The algorithm of the proposed method is shown in fig. 4.1.

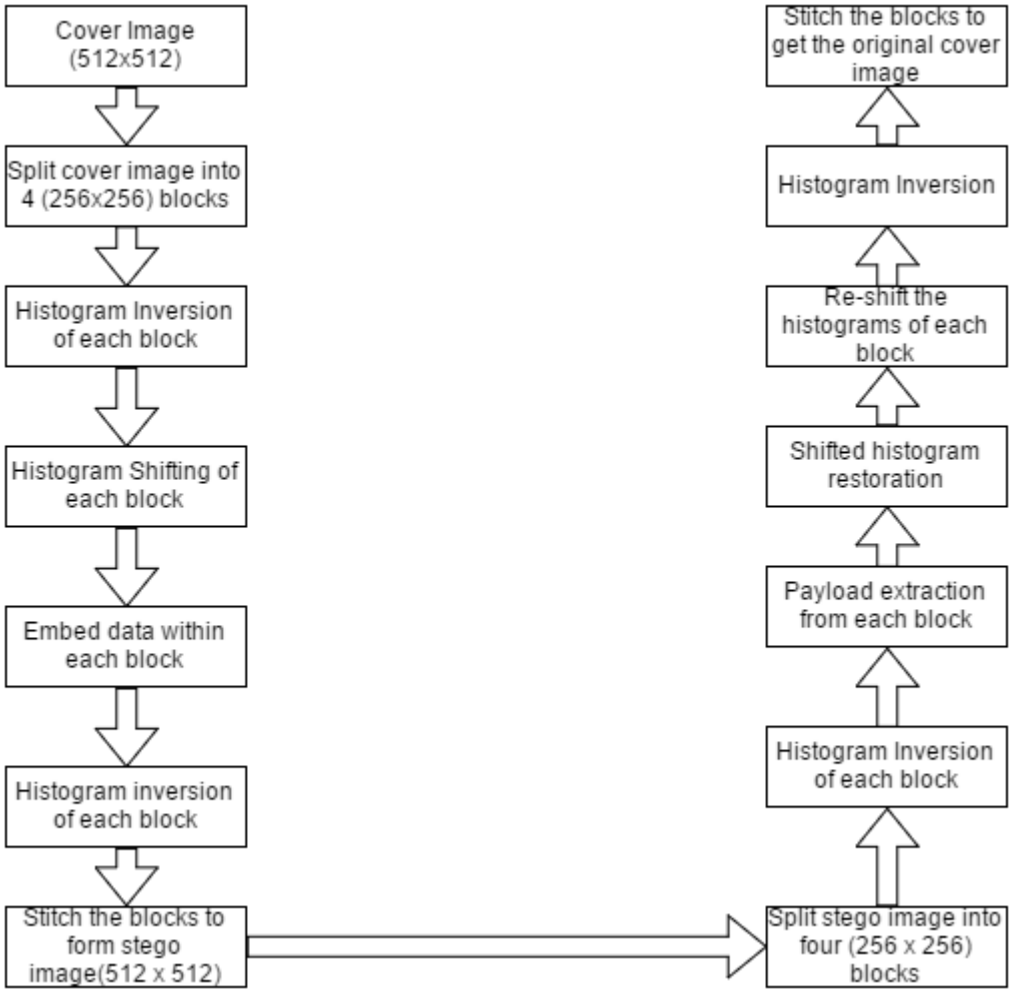

Figure 4-1: Proposed method to hide data

4.2 Steps:

At the transmitter:

1) The (512 × 512) color image as shown in fig. 4.2, is split into four blocks, each of size (256 × 256) as shown in fig. 4.3. The histogram of each block is shown in fig. 4.4.

2) Then the histogram of each block is inverted as shown in fig. 4.5.

$$B(i,j) = 255 - I(i,j), 1 \leq i \leq rows; 1 \leq j \leq columns \tag{4.1}$$

3) Histogram shifting of each block as proposed by Pan et al. [26].

$$B'(i,j) = \begin{cases} B(i,j) + 1, & v_{peak} + 2 \leq B(i,j) \leq 254 \\ B(i,j) - 1, & 1 \leq B(i,j) \leq v_{peak} - 2 \end{cases} \tag{4.2}$$

4) The data is embedded in each block using equation 4.3.

$$B''(i,j) = \begin{cases} B'(i,j) + 1, & B'(i,j) = v_{peak} + 1, p(l) = 1 \\ B'(i,j) - 1, & B'(i,j) = v_{peak} - 1, p(l) = 1 \end{cases} \tag{4.3}$$

5) The histograms of each block are inverted and stitched to form a stego image of size (512 × 512).

$$B'''(i,j) = 255 - B''(i,j), 1 \leq i \leq rows; 1 \leq j \leq columns \tag{4.4}$$

At the receiver:

1) The (512 × 512) stego image is split into four blocks, each of size (256 × 256).

2) Invert the histogram of each block.

$$C(i,j) = 255 - B'''(i,j), 1 \leq i \leq rows; 1 \leq j \leq columns \tag{4.5}$$

3) The data is extracted from each block and shifted histogram is obtained using equation 4.6.

$$C'(i,j) = \begin{cases} C(i,j) - 1, & C(i,j) = v_{peak} + 2, p(l) = 1 \\ C(i,j) + 1, & C(i,j) = v_{peak} - 2, p(l) = 1 \\ C(i,j), & C(i,j) = v_{peak} \pm 1\,, p(l) = 0 \end{cases} \tag{4.6}$$

4) The shifted blocks are re-shifted using equation 4.7.

$$C''(i,j) = \begin{cases} C'(i,j) - 1, & v_{peak} + 3 \leq C'(i,j) \leq 255 \\ C'(i,j) + 1, & 0 \leq C'(i,j) \leq v_{peak} - 3 \end{cases} \tag{4.7}$$

5) The histogram of each block is inverted using equation 4.8.

$$C'''(i,j) = 255 - C''(i,j), 1 \leq i \leq rows; 1 \leq j \leq columns \tag{4.8}$$

6) The blocks are stitched to form cover image of size (512 × 512).

Figure 4-2: Original image

Figure 4-3: Original image split into four 256 × 256 blocks

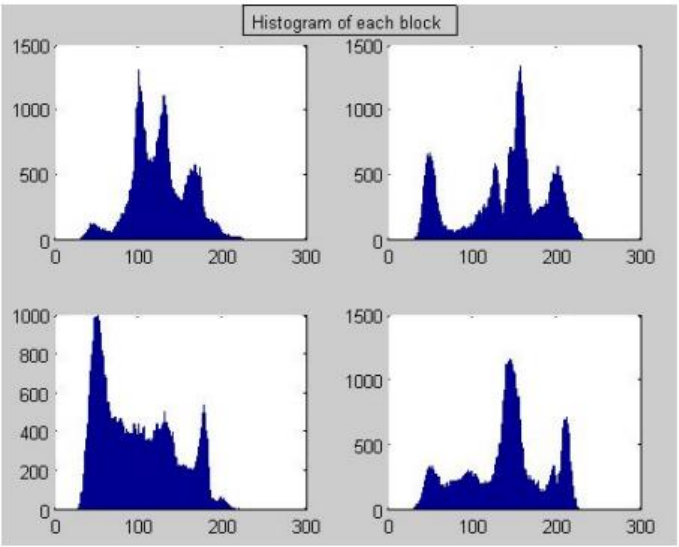

Figure 4-4: Histogram of individual blocks (see fig.4-3)

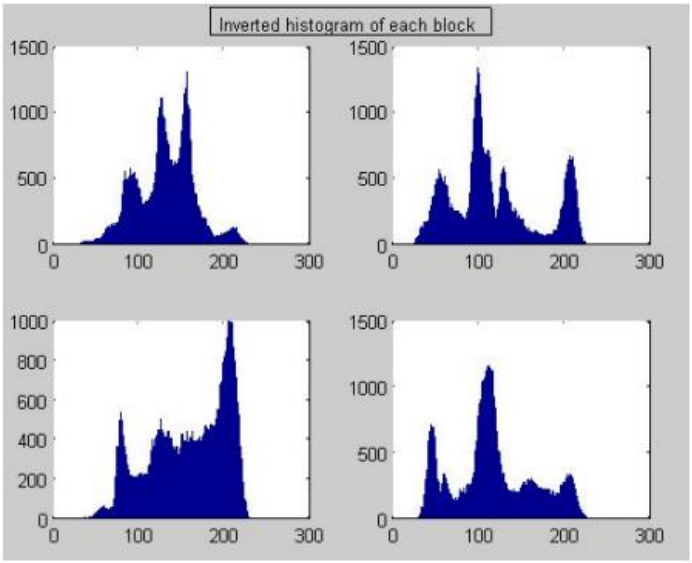

Figure 4-5: Histogram of inverted blocks (see fig.4-4)

Chapter 5

Results

Different grey-scale images are used for the simulation purpose. Quality metrics like PSNR, number of bits that can be embedded are calculated for test images with variable block size. The results are shown in part A and part B. In part A, the overall algorithm is presented in detail. In part B, the maximum number of bits that can be hidden along with the PSNR value of the stego image for different block sizes is presented.

Part A:

In this section, the output of each step in the algorithm is presented. The input image is split into 4 blocks each of size 256 x 256 for this simulation.

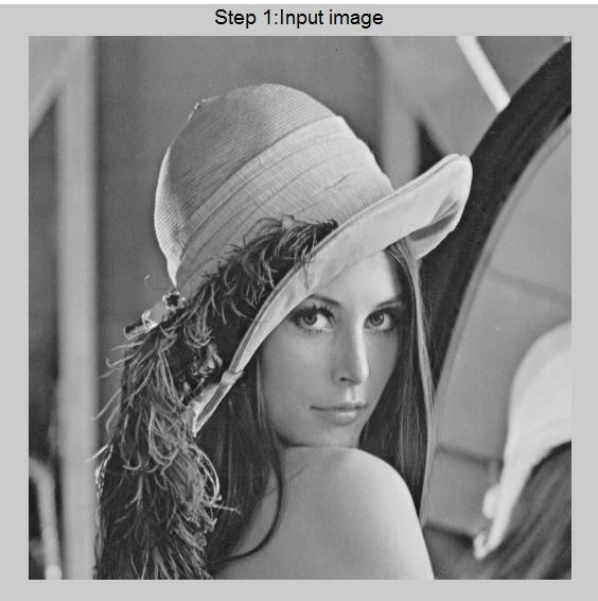

Figure 5-1

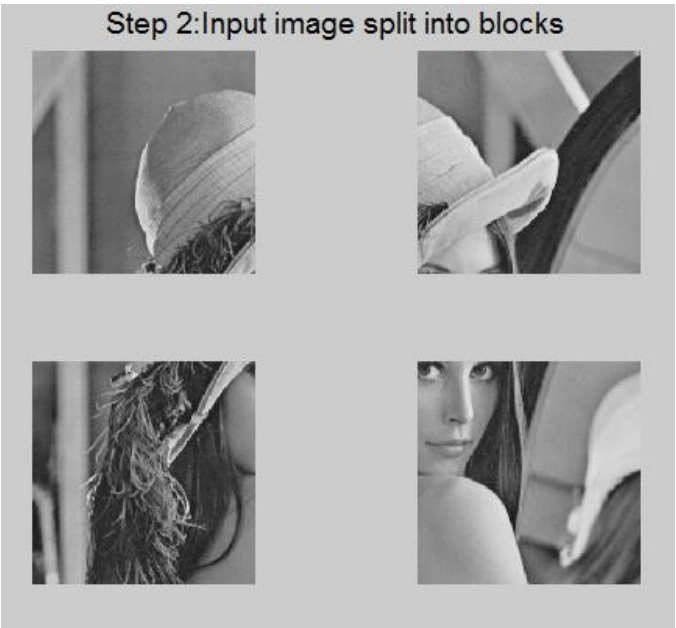

Figure 5-2

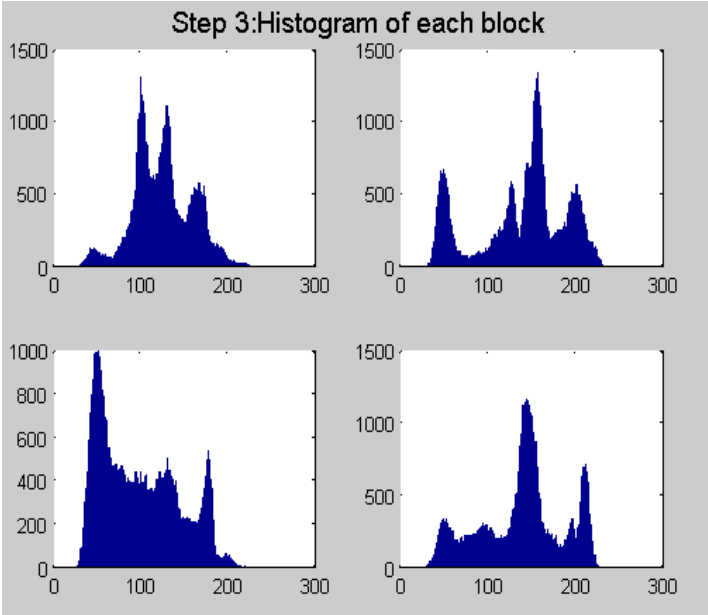

Figure 5-3

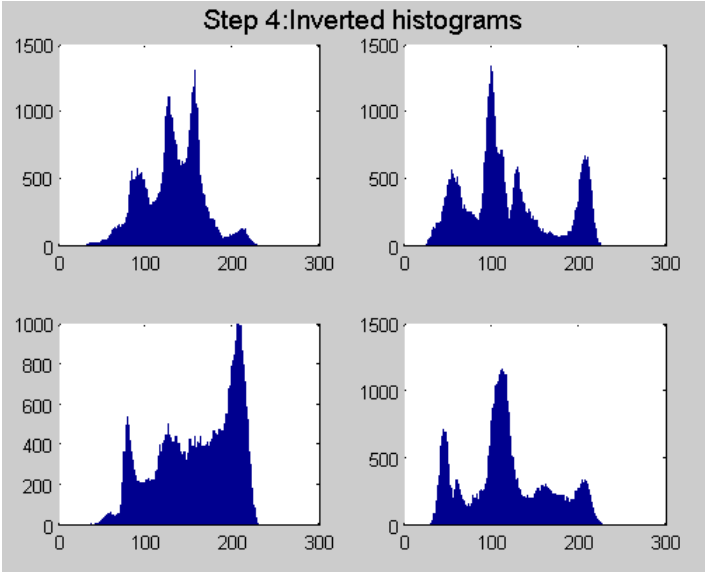

Figure 5-4

Figure 5-5: Inverted lena image split into blocks

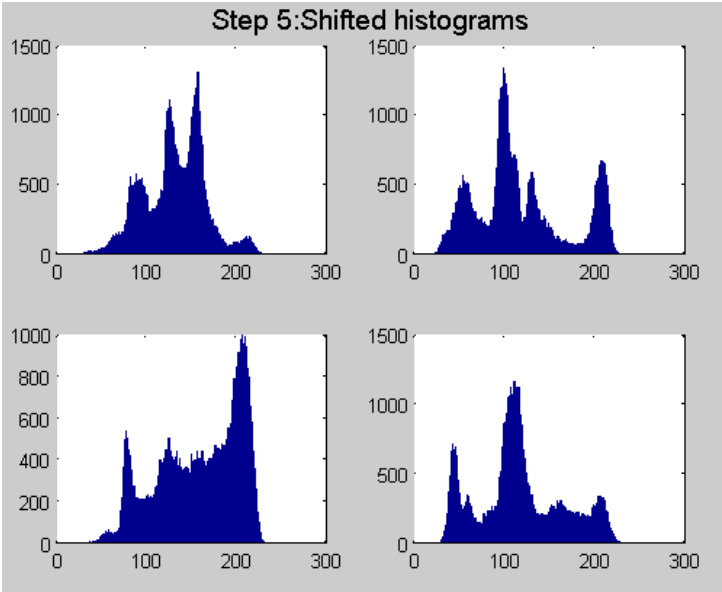

Figure 5-6

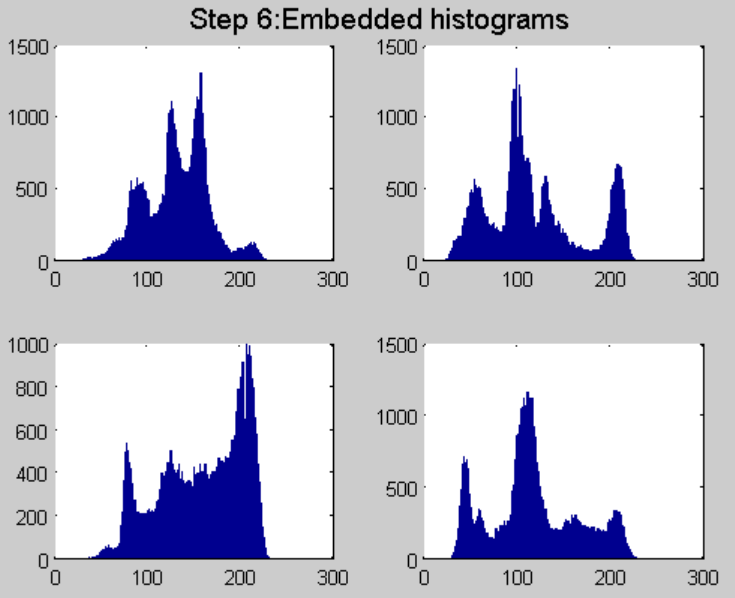

Figure 5-7

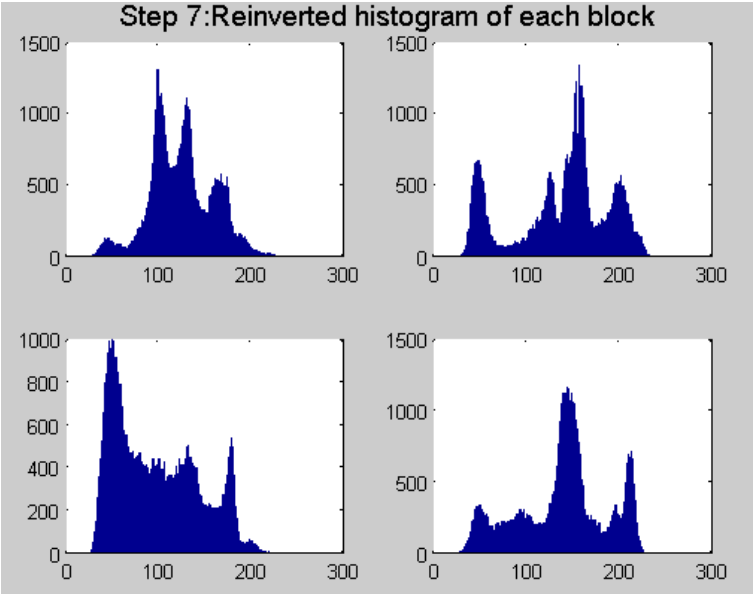

Figure 5-8

Figure 5-9

Figure 5-10

Figure 5-11

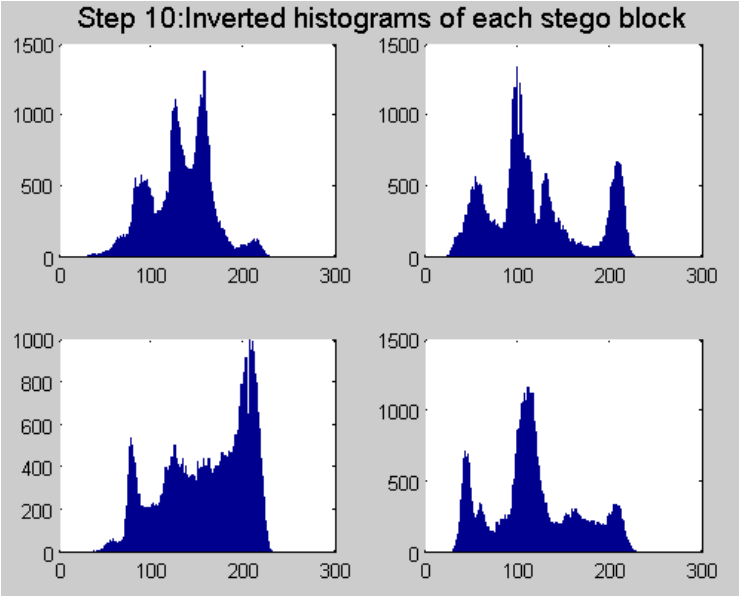

Figure 5-12

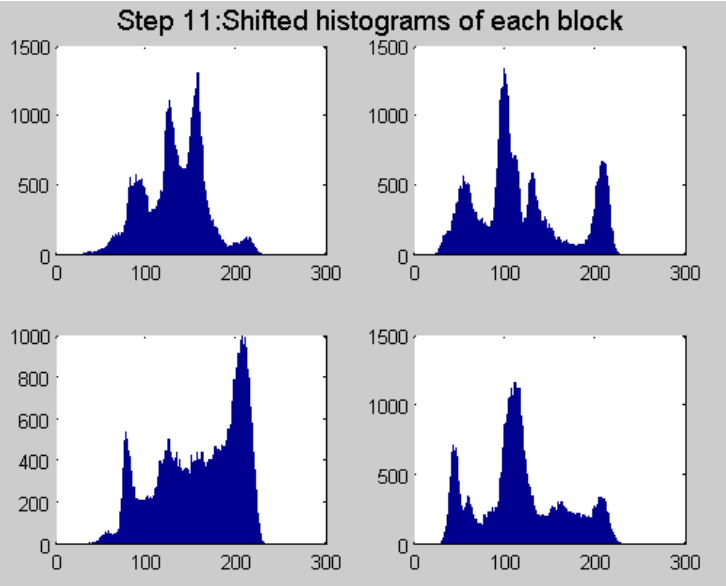

Figure 5-13

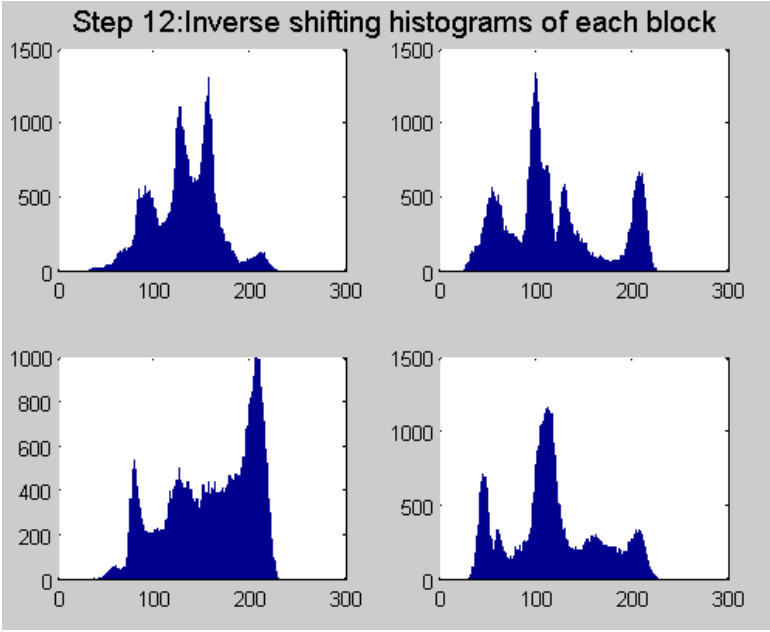

Figure 5-14

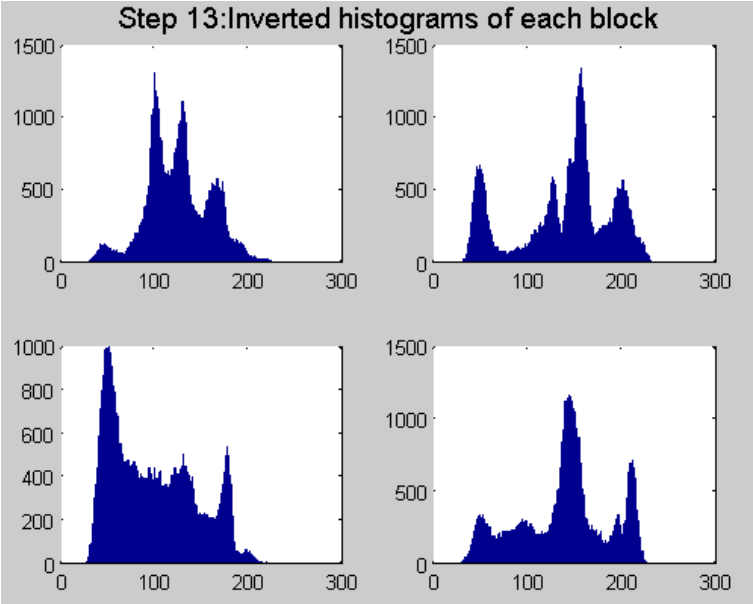

Figure 5-15

Figure 5-16

Part B:

Table 5-1 Results for Lena image

Block size	PSNR of the stego image(in dB)	Max number of bits that can be hidden
256 x 256	48.3141	8987
128 x 128	48.3913	12675
32 x 32	48.6880	24209
16 x 16	48.8930	29579
8x8	49.1838	33960

Table 5-2 Results for Barbara image

Block size	PSNR of the stego image(in dB)	Max number of bits that can be hidden
256 x 256	48.2512	5797
128 x 128	48.3061	8322
32 x 32	48.5328	17438
16 x 16	48.6917	21798
8x8	48.9185	24808

Table 5-3 Results for Goldhill image

Block size	PSNR of stego image(in dB)	Max number of bits that can be hidden
256 x 256	48.2750	7160
128 x 128	48.3358	9966
32 x 32	48.5343	17385
16 x 16	48.6937	21806
8x8	48.9157	24237

Table 5-4 Results for Airplane image

Block size	PSNR of stego image(in dB)	Max number of bits that can be hidden
256 x 256	48.6141	22754
128 x 128	48.7194	27149
32 x 32	49.0965	38959
16 x 16	49.3330	44047
8x8	49.6595	47568

Table 5-5 Results for Tiffany image

Block size	PSNR of the stego image(in dB)	Max number of bits that can be hidden
256 x 256	48.4693	16409
128 x 128	48.6105	21617
32 x 32	49.0057	35272
16 x 16	49.2274	40449
8x8	49.5564	44490

Summary

In chapter 5, results for different test images are presented. From the above results it can be perceived that the PSNR of the stego image and hiding capacity increases with decrease in block size. In the next chapter future work is discussed.

Chapter 6

Conclusions and Future Work

The objective of the thesis is to improve the security of the present data hiding algorithms. This was achieved using histogram inversion before embedding the data in the cover image. In the results section it is clear that the PSNR of the image improves with decrease in block size.

The only tradeoff is, when the block size decreases, there's significant increase in number of blocks to be processed. This leads to an increase in the embedding time.

This thesis is mainly based on the algorithm proposed by Pan et al. [26]. Future work can be, extending the present algorithm for color images and also using the generated stego image to embed data again. The latter is called as multi-layer embedding.

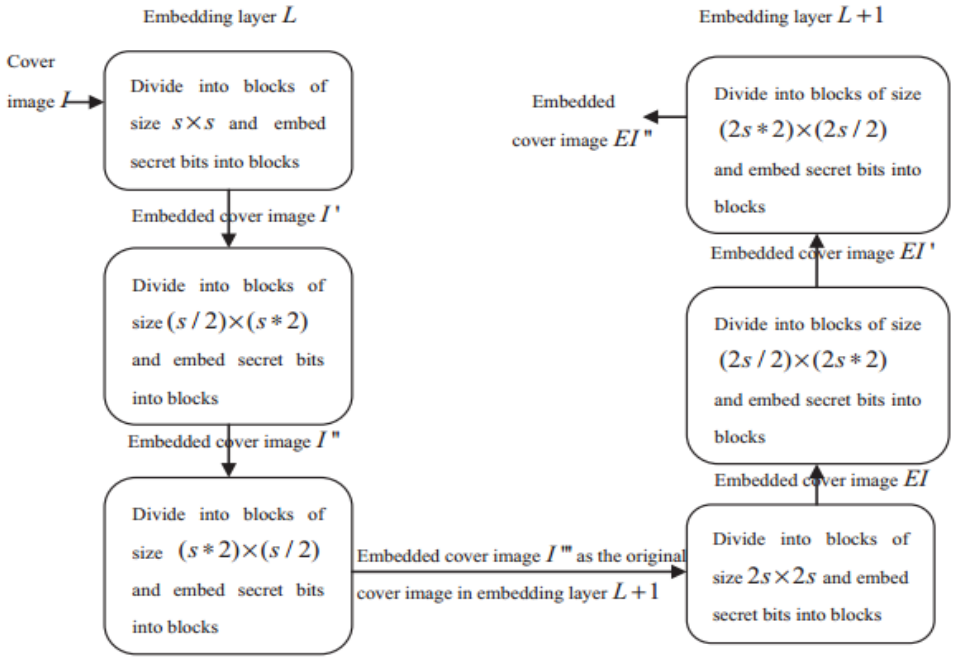

Figure 6-1: Flowchart of multi-layer embedding

Appendix

Test conditions

All the work was done on a system with following configuration:

- Operating System: Windows 10 Home Edition
- Processor: Intel(R) Core(TM) i3-3217U @ 1.80GHz
- RAM: 4.00 GB

System type: 64-bit Operating System, x64-based processor

REFERENCES

[1] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber: Reversible data hiding, *Proceedings of IEEE International Conference on Image* Processing, Rochester, NY, pp. 157-160, 2002.

[2] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, “Secure spread spectrum watermarking for multimedia,” in IEEE Trans. on Image Processing, vol. 6. No. 12, pp. 1673-1687, Dec. 1997.

[3] C. C. Chang, J. Y. Hsiao, and C. S. Chan: Finding optimal LSB substitution in image hiding by dynamic programming strategy, *Pattern Recognition*, vol. 36, no. 7, pp. 1583-1595, 2003.

[4] J. Huang and Y. Q. Shi, "An adaptive image watermarking scheme based on visual masking," *IEE Electronic Letters*, vol. 34, no. 8, pp. 748-750, April 1998.

[5] B. Chen, G. W. Wornell, “Quantization index modulation: a class of provably good methods for digital watermarking and information embedding,” IEEE Transaction on Information Theory, vol. 47, no. 4, pp. 1423-1443, May 2001.

[6] C. W. Honsinger, P. Jones, M. Rabbani, and J. C. Stoffel, “Lossless recovery of an original image containing embedded data,” US Patent: 6,278,791, 2001.

[7] S. Katzenbeissar and F. Petitcolas, Editors, “Information hiding techniques for steganography and digital watermarking,” Artech House, 1999.

[8] Liu, Shao-Hui, Tun-Hang Chen, Hong-Xun Yao, and Wen Gao. “A variable depth LSB data hiding technique in images”. *Proceedings of IEEE International Conference on Machine Learning and Cybernetics,* vol. 7, pp. 3990-3994, Aug., 2004.

[9] Z. M. Lu, J. S. Pan, and S. H. Sun, “VQ-based digital image watermarking method”, Electronics Letters, vol. 36, no. 14, pp. 1201-1202, April, 2000.

[10] M. Goljan, J. Fridrich, and R. Du, “Distortion-free data embedding,” Proceedings of 4th Information Hiding Workshop, pp. 27-41, Pittsburgh, PA, April, 2001.

[11] C. I. Podilchuk and E. J Delp, “Digital watermarking: Algorithms and applications,” IEEE Signal Processing Magazine, pp. 33–46, July, 2001.

[12] G. Xuan et al, “Distortionless data hiding based on integer wavelet transform,” IEE Electronics Letters, pp. 1646-1648, Dec 2002.

[13] H. C. Wu, N. I. Wu, C. S. Tsai and M. S. Hwang, "Image steganographic scheme based on pixel-value differencing and LSB replacement methods," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 152, no. 5, pp. 611-615, 7 Oct. 2005.

[14] B. Yang et al, "Integer DCT Based Reversible Image Watermarking by Adaptive Coefficient Modification" in E. J. Delp, editor, Proceedings of Electronic Imaging, volume VII of Security and Watermarking of Multimedia Contents, SPIE vol. 7, pp. 218-229, Jan, 2005.

[15] Jun Tian, "Reversible data embedding using a difference expansion," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890-896, Aug. 2003.

[16] G. Xuan et al, “Lossless data hiding based on integer wavelet transform”, *IEEE Workshop on Multimedia Signal Processing, 2002*, pp. 312-315, Dec., 2002.

[17] Y. C. Hu, “High capacity image hiding scheme based on vector quantization”, Pattern Recognition, vol. 39, no. 9, pp. 1715-1724, Sep., 2006.

[18] G. Xuan, Y. Q. Shi, and Z. Ni, “Reversible data hiding using integer wavelet transform and companding technique,” Proc. IWDW04, Korea, October 2004.

[19] J. Fridrich, M. Goljan and R. Du, “Invertible authentication,” Proc. SPIE, Security and Watermarking of Multimedia Contents, pp. 197-208, San Jose, CA, January 2001.

[20] M. Thodi and J. J. Rodríguez, “Reversible watermarking by prediction-error expansion,” Proceedings of 6th IEEE Southwest Symposium on Image Analysis and Interpretation, pp. 21-25, Lake Tahoe, CA,USA, March 28-30, 2004.

[21] C. De Vleeschouwer, J. F. Delaigle and B. Macq, “Circular interpretation of bijective transformations in lossless watermarking for media asset management,” IEEE Trans. Multimedia, vol. 5, pp. 97-105, March 2003.

[22] W. Bender, D. Gruhl, N. Mprimoto and A. Lu, “Techniques for data hiding,” IBM Systems Journal, vol. 35, Nos. 3&4, pp. 313-336, 1996.

[23] Z. Ni et al, “Robust lossless data hiding,” IEEE International Conference and Expo, Taipei, Taiwan, June 2004.

[24] D. Zou, Y. Q. Shi and Z. Ni, "A semi-fragile lossless digital watermarking scheme based on integer wavelet transform," *IEEE 6th Workshop on Multimedia Signal Processing*, pp. 195-198, Oct., 2004.

[25] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, “Reversible data hiding,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 3, pp. 354–362, Mar 2006.

[26] Z. Pan, S. Hu, X. Ma, and L. Wang, “Reversible data hiding based on local histogram shifting with multilayer embedding,” Journal of Visual Communication and Image Representation, vol. 31, pp. 64–74, Aug., 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1047320315000838

[27] G. Xuan *et al*., "High capacity lossless data hiding based on integer wavelet transform," *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, Vol.2, pp. II-29-32, May, 2004.

[28] Z. Zhang et al, "A unified authentication framework for JPEG2000," *IEEE International Conference on Multimedia and Expo*, Vol. 2, pp. 915-918, Taipei, June, 2004.

[29] N. Ramaswamy, “Digital signature in H.264/AVC MPEG4 Part 10,” M.S. Thesis, Dept. of Electrical Engineering, University of Texas at Arlington, Aug 2004.

[30] S. Puthussery, “A progressive secret reveal system for color images,” M.S. Thesis, Dept. of Electrical Engineering, University of Texas at Arlington, Aug 2004.

[31] M. T. Sandford II, J. N. Bradley, and T. G. Handel, “Data embedding method,”Proc. SPIE 2615, Integration Issues in Large Commercial Media Delivery Systems, pp. 226–259, Jan, 1996.

[32] C. A. Allen and J. L. Davidson, “Steganography using the minimax eigenvalue decomposition,” Proc. SPIE 3456, Mathematics of Data/Image Coding, Compression, and Encryption, pp. 13–24, Nov., 1998.

[33] S. Takano, K. Tanaka, and T. Sugimura, “Steganograpic image transformation,” Proc. SPIE 3657, Security and Watermarking of Multimedia Contents, pp. 365–374, April, 1999.

[34] L.-C. Huang, L.-Y. Tseng, and M.-S. Hwang, “A reversible data hiding method by histogram shifting in high quality medical images,” Journal of Systems and Software, vol. 86, no. 3, pp. 716–727, March, 2013.

[35] H. J. Hwang, “Reversible Watermarking Method Using Optimal Histogram Pair Shifting Based on Prediction and Sorting,” KSII Transactions on Internet and Information Systems, vol. 4, no. 4, pp. 655–670, Aug, 2010.

[36] J. Gupta, P. Gupta, and S. C. Gupta, “Reversible data hiding technique using histogram shifting,”2nd International Conference on Computing for Sustainable Global Development, pp. 2114–2119, March 2015.

[37] M. Fujiyoshi, “A Histogram Shifting-Based Blind Reversible Data Hiding Method With a Histogram Peak Estimator,” 2012 International Symposium on Communications and Information Technologies (ISCIT), vol. 1, pp. 313–318, Oct., 2012.

[38] F. Shih, “Digital watermarking and steganography: Fundamentals and Techniques,” CRC Press, 2008.

[39] Notes on Steganography, Available: http://www.garykessler.net/library/ steganography.html

[40] S. Jajodia, N. F. Johnson, and Z. Duric, “Information hiding: Steganography and Watermarking-attacks and Countermeasures,” Springer US, 2001.

[41] Y. Q. Shi, “Reversible data hiding”, In Springer Digital Watermarking, pp. 1-12, Oct 30, 2004.