ABSTRACT

DIFFERENCE-EXPANSION BASED REVERSIBLE DATA HIDING AND ITS STEGANALYSIS

by
Guangsen Tian

A novel reversible data embedding method was reported in a recent IEEE journal article. The method was based on difference expansion (DE) technique. It used redundancy in digital images to achieve a high embedding capacity, while keeping visual distortion of the stego-image low. In this thesis, this technique was studied and experimentally evaluated. An effective steganalysis scheme for this DE-based reversible data embedding method was proposed, which used 12-dimensional feature vectors and a Bayes Classifier. The proposed steganalysis scheme steadily achieved a correct classification rate of 99%.

DIFFERENCE-EXPANSION BASED REVERSIBLE DATA HIDING AND ITS STEGANALYSIS

by
Guangsen Tian

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

May 2005

APPROVAL PAGE

DIFFERENCE-EXPANSION BASED REVERSIBLE DATA HIDING AND ITS STEGANALYSIS

Guangsen Tian

Dr. Yun Q. Shi, Thesis Advisor — Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Constantine N. Manikopoulos, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Mengchu Zhou, Committee Member — Date
Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Guangsen Tian

Degree: Master of Science

Date: May 2005

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2005

- Master of Engineering in Electrical Engineering,
  Southeast University, Nanjing, P. R. China, 2002

- Bachelor of Engineering in Electrical Engineering,
  Southeast University, Nanjing, P. R. China, 1995

Major: Electrical Engineering

Dedicated to my beloved parents and wife

谨献给我深爱的父母和妻子

ACKNOWLEDGMENT

I would like to express my deep thanks and appreciation to my thesis advisor, Dr. Yun Q. Shi, for his help and support. Dr. Shi helped me keep focused on long-term goals and make progress toward the completion of my work. With his encouragement, I was able to overcome all the difficulties in my research and complete this project successfully. Special thanks are given to Dr. Constantine N. Manikopoulos, and Dr. Mengchu Zhou for actively participating in my committee.

Many thanks are given to my friend Mark Harkness for revising my thesis.

Many of my fellow graduate students in the Multi-Media Research Laboratory are deserving of recognition for their support.

Many thanks to my wife Fu Yan for moral support and for having an open ear every time I needed someone to listen.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Objectives

The objective of this thesis is to analyze a new advantaged reversible data embedding method and propose a new blind steganalysis scheme, which can be used in detecting whether a given medium has hidden message in it.

Recently, digital watermarking has emerged as an increasingly active research area. An invisible watermark is embedded into digital content for purposes of copyright communication and protection, content authentication, counterfeit deterrence or broadcast monitoring, etc. But at the same time it also provides vast opportunities for converting communications. This task is especially urgent for law enforcement to deter the distribution of children pornography images/videos hidden inside normal images/videos, and for intelligence agencies to intercept communications of enemies. Steganalysis is a scientific technology to decide if a medium carries some hidden message. In counter-terrorists activities, steganalysis can serve as an effective way to judge the security performance of steganographic techniques.

1.2 Background Information

In this thesis, a high capacity, high visual quality, reversible data-embedding method is introduced, which is based on a technique called the difference expansion (DE) [1]. And then according to this kind of data embedding method, an effective steganalysis scheme is presented, which is based on a 12-dimensional feature vector and a Bayes classifier.

1.2.1 Reversible Data Embedding Algorithm

Most digital watermarking methods can be categorized as either robust watermarking or fragile watermarking. A robust watermarking should be very resistant to various signal processing operations, while a fragile watermark will be destroyed or degraded in predictable fashion when the digital content is modified. Thus fragile watermarking is a very valuable tool in content authentication. As a special subset of fragile watermarking, reversible data embedding, which is also called lossless, invertible, or erasable data embedding (or watermarking, data hiding), enables the recovery of the original content. With reversible data embedding, one can authenticate the content and restore the original content after it is authenticated. In recent years, a reversible data-embedding method based on difference expansion technique (DE) was put forward. This kind of method calculates the difference values of neighboring two pixel values, and selects some difference values for difference expansion. The original content restoration information, an authentication hash, and additional data (which could be any data, such as date/time information, auxiliary data, etc.) will all be embedded into the difference values [1]. In this thesis, only 8 bits per pixel (bpp) grayscale images are considered.

Reversible data embedding has been a very active research subject in the last few years. In particular, Hongisinger, *et al.* [2], used a robust watermark via spread spectrum and modulo arithmetic to achieve reversible data embedding. Fridrich, *et al.* [3], developed a high capacity reversible data-embedding technique based on embedding information on bits in the status of groups of pixels. Dittmann, *et al.* [4], introduced a media independent protocol designed for reversible data-embedding applications, which required high data integrity. Celik, *et al.* [5], presented a high capacity, low distortion

reversible data-embedding algorithm by compression quantization residues. They employed the lossless image compression algorithm CALIC [6], with quantized values as side-information, to efficiently compress quantization residues to obtain high embedding capacity. Xuan, *et al.* [7], losslessly compressed one (or more) middle bit plane(s) in the wavelet decomposition domain for reversible data embedding. Van Leest, *et al.* [8], presented a reversible data-embedding technique based on a transformation function that introduces "gaps" in the image histogram. Those reversible data embedding algorithms (including the one introduced in this thesis) are fragile: when the embedding image is manipulated and /or lossy compressed, the decoder will find out that it has been tampered and thus there will be no original content restoration.

1.2.2 Steganalysis

An original cover medium and its embedded medium (stego-image) (with hidden message inside) always differ from each other in some aspects since the cover medium is changed during the data embedding. For example, in [9], Fridrich *et al.* discovered that the number of zeros in the block DCT domain of a stego-image will increase if the F5 embedding method is applied to generate the stego-image. There are also some other findings regarding the steganalysis of a particular data hiding method [10] and [11].

Recently, several general steganalysis methods were proposed. In [12], Farid proposed a general steganalysis method based on image high order statistics. These statistics are based on decomposition of images with separable quadrature mirror filters. The subbands high order statistics are obtained from cover images with a certain success rate. In [13], a steganalysis method based on the mass center (the first order moment) of histogram characteristic function is proposed. The second and third order moments are

also considered for steganalysis. Compared with [12], its performance has been improved. However, the performance achieved by [13] is still not high enough since it adopts very limited number of features extracted from the test image. In this thesis, we focus on reversible data embedding methods based on the DE technique and try to get an insight from this to set up a new general steganalysis system with a high correction rate.

By applying this data embedding method, when a difference expansion happens in an image, there will be obvious change in the difference values between original images and stego-images. Compared with the original image, the differences of the stego-image will be enlarged. So a 12-dimensional feature vector is set up for steganalysis by calculating the difference values and separating them into 12 different groups. Finally, a Bayes classifier is built up for classification. All of the 1096 images were used from the coreldraw image database for our extensive experimental work.

The thesis is organized as follows. An introduction about reversible data embedding method based on the DE technique is given in chapter 2. In chapter 3, some discussions and simulation results on this method are presented. A 12-dimensional feature vector and a Bayes classifier are presented in chapter 4. Then the experimental process and results for our steganalysis method are provided in chapter 5. The conclusion is drawn in chapter 6.

CHAPTER 2

REVERSIBLE DATA EMBEDDING METHOD

The reversible data embedding is also called lossless data embedding. Through it data can be embedded into digital images. When needed or authenticated, the original images and embedding data can be restored exactly.

Traditional data embedding is a lossy process. That means that you cannot recover the original image exactly after data embedding. Compared with lossy embedding methods, reversible data embedding embeds a payload into a digital content in a reversible fashion. After embedding, the image should change very little or look no different. That means that the quality degradation on the digital content after data embedding should be low. Another obvious feature of reversible data embedding is the reversibility, that is, when the digital content has been authenticated, one can remove the embedded data to restore the original content (before data embedding). The process can be explained in Figure 2.1 [1]. Here a payload is embedded in a digital image *I*, and the stego-image (embedded image) *I'* is obtained. The quality degradation of *I'* from I should be low. Before sending it to the content authenticator, the image *I'* might or might not have been tampered by some intentional (for example, changing the content) or unintentional (for example, JPEG compression) manipulations. If the authenticator finds that no tampering happened in *I'*, i.e., *I'* is authentic, then the authenticator can remove the payload from *I'* and restore the original image, which results in a new image *I''*. By definition of reversible data embedding, the restored image *I''* will be exactly the same as the original image *I*, pixel by pixel, bit by bit.

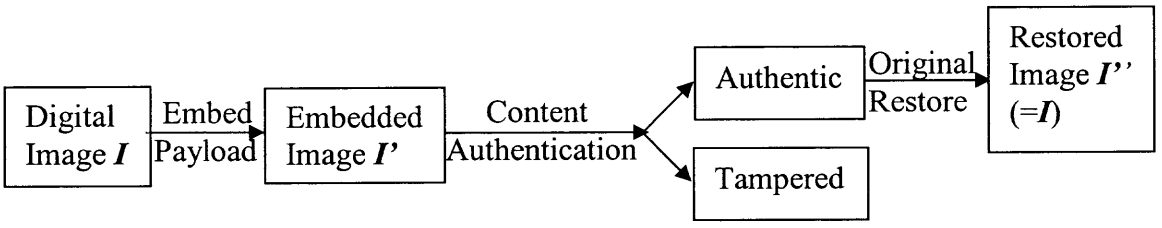

Fig 2.1 Reversible data embedding diagram [1]

A technique called difference expansion (DE) will be introduced in following section. Compared with other reversible data embedding methods, this method can achieve high data embedding capacity and at the same time keep the distortion very low.

2.1 The Difference Expansion Technique

The DE technique [10] reversibly embeds one bit data into two integers, which can be explained as follows. There are lots of redundancies in a digital image. If we change some pixel values to some extent, the appearance of the picture is almost same as the original one. So data embedding can be obtained by changing some pixel values in an image. The DE technique uses the difference between two pixel values to embed one bit. This process can be explained by a simple example shown as Fig. 2.2 [1].

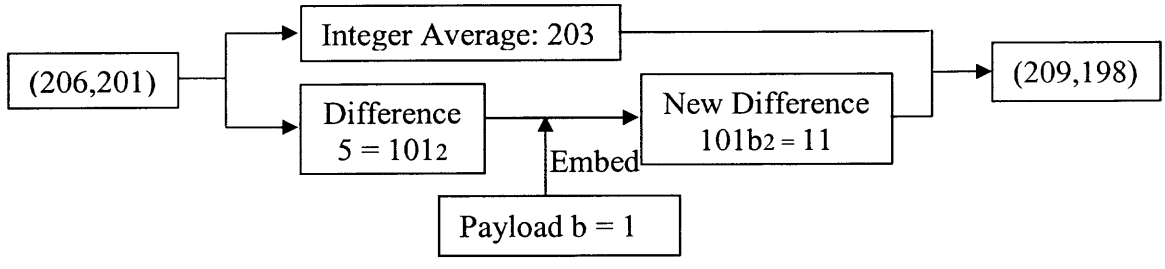

Fig 2.2 A simple example [1]

Assuming there are two grayscale values $x = 206$, $y = 201$, we would like to reversibly embed one bit $b = 1$. First the integer average and difference of x and y are computed,

$$l = \lfloor (x + y)/2 \rfloor \qquad h = x - y \tag{2.1}$$

where the symbol $\lfloor . \rfloor$ is the floor function meaning "the greatest integer less than or equal to". Next the difference value h is represented as its binary representation:

$$h = 5 = 101_2 \tag{2.2}$$

Then embedded bit b is appended into the binary representation of h after the least significant bit (LSB), the new difference value h' will be:

$$h' = 101b_2 = 1011_2 = 11 \tag{2.3}$$

Mathematically, this is equivalent to:

$$h' = 2 * h + b = 2 * 5 + 1 = 11 \tag{2.4}$$

This operation is also called the difference expansion (DE).

Finally the new grayscale values are computed, based on the new difference value h' and the original integer average value l,

$$x' = l + \lfloor (h' + 1)/2 \rfloor = 203 + \lfloor (11 + 1)/2 \rfloor = 209 \tag{2.5}$$

$$y' = 1 - \lfloor h'/2 \rfloor = 203 - \lfloor 11/2 \rfloor = 198 \tag{2.6}$$

and new two pixel values $x = 209$, $y = 198$ are gotten. After finishing this process, one bit is embedded into the two pixel values.

2.2 Data Embedding Process Based on The DE Technique

In brief, the data-embedding algorithm based on the DE technique consists of six steps: 1.Calculating the average and difference values between two pixel values. 2.Partitioning the difference values into four sets. 3.Creating a location map. 4.Collecting the original LSB values. 5.Data embedding by replacement. 6.An inverse integer transforms.

2.2.1 Calculating Average and Difference Value

In this method data is embedded in the difference values of pairs of pixel values. Therefore, the original image is grouped into pairs of pixel values. A pair consists of two neighboring pixel values. For example, two continuous pixel values in horizontal direction $(i, 2j-1)$ and $(i, 2j)$ can be selected as a pair. There are 4 different pairs: horizontal, vertical and two diagonal directions. Reversible integer transforms can be used to get the average and difference matrix.

Here, starting with a simple reversible integer transform. For a grayscale-valued pair (x, y) Where $0 \leq x, y \leq 255$, their integer average and difference are defined as

$$l = \lfloor (x + y)/2 \rfloor, \quad h = x - y \tag{2.7}$$

The inverse transform of the function above is

$$x = l + \lfloor (h + 1)/2 \rfloor, \; y = l - \lfloor h/2 \rfloor \tag{2.8}$$

The reversible integer transforms (2.5) and (2.6) are also called integer Haar wavelet transforms. So for any image a matrix of difference and a matrix of average can be acquired by using equation (2.5). For example, for a 512*512 image, if vertical pair is

selected to calculate average and difference value, a 512*256 difference matrix and a 512*256 average matrix will be found.

2.2.2 Partitioning Difference Values into Four Sets

To prevent overflow and underflow problems, (i.e., to restrict x, y in the range of [0, 255]), before partitioning the difference values two definitions [1] about difference value h are given.

Definition 1: A difference value h is expandable under the integer average value l if

$$|2h + b| \leq min\ (2(255-l),\ 2l + 1) \tag{2.9}$$

for both $b = 0$ and 1. For an expandable difference value h, a bit is embedded by the DE technique. That means one bit 0 or 1 can be embedded into the LSB of difference value h, and at the same time guarantee all pixel values in the stego-image will be in the range of [0, 255].

Definition 2: A difference value h is changeable under the integer average value l if

$$|2\lfloor h/2 \rfloor + b| \leq min\ (2(255-l),\ 2l + 1) \tag{2.10}$$

for both $b = 0$ and 1. Here changeable means data can be embedded by changing the LSB of difference value, meanwhile guarantee all pixel values in the stego-image will be in the range of [0, 255].

Next four disjoint sets of difference values are created, *EZ, EN, CN*, and *NC*:

1. *EZ*: contains all expandable $h = 0$ and expandable $h = -1$.

2. *EN*: contains all expandable *h* values that do not belong to *EZ*.

3. *CN*: contains all changeable *h* values that do not belong to (*EZ* and *EN*).

4. *NC*: contains all non-changeable *h* values.

Each difference value will fall into one and only one of the above four sets. As an expandable difference value is always changeable, the whole set of changeable difference values is $EZ \cup EN \cup CN$.

2.2.3 Creating Location Map

Every difference value *h* in *EZ* will always be selected for difference expansion. For *EN*, depending on the payload size, some difference values will be selected for difference expansion. For convenience, the subsets of selected and non-selected difference values in *EN* as *EN1* and *EN2* are denoted respectively. A one-bit bitmap is created as the location map. For an h in either *EZ* or *EN1*, a value "1" is assigned in the location map; for an *h* in *EN2, CN, or NC*, a value "0" is assigned. Details can be clearly seen from the Table 2.1 [1].

Table 2.1 Embedding on Difference Values [1]

Category	Original Set	Original Value	Location Map	New Value	New Set
Changeable	*EZ or EN1*	*h*	*1*	$2*h + b$	*CH*
	EN2 or CN	*h*	*0*	$2*\lfloor h/2 \rfloor + b$	
Non-changeable	*NC*	*h*	*0*	*h*	*NC*

2.2.4 Collecting Original LSB Values

With the location map, the encoder and the decoder will share the same information on which difference values have been selected for difference expansion. While it is straightforward for the encoder to embed a location map, the decoder needs to know where to collect and decode it.

After difference expansion, the expanded difference value might not be expandable. On the decoder side, to check whether the expanded difference value is expandable does not tell whether the original difference value has been selected for difference expansion during embedding. The expanded difference value is changeable so the decoder could examine each changeable difference value.

In this method, the encoder serves for the decoder. The encoder will select changeable values as the embedding area, so that the decoder will use the same data to decode. During data embedding, this method will modify all changeable difference values, by either adding a new LSB through the DE technique or modifying its LSB. To guarantee an exact recovery of the original image, this method will also embed the original values of those modified LSBs. So for the changeable difference values, the original h values need to be known in preparation for the decoding process. In this method, the original LSBs of difference values are collected in *EN2* and *CN*. For each h in *EN2* or *CN*, LSB (h) will be collected into a bit stream *C*. An exception is when $h = 1$ or -2, nothing will be collected, as its original LSB (1 and 0, respectively) could be determined by the location map.

2.2.5 Data Embedding by Replacement

The location map will be losslessly compressed by a JBIG2 compression or run-length coding. The compressed bit stream is denoted as L. Then L, C, and P are combined together into one binary bit stream B, where P is payload size. The bit stream B is embedded into difference values matrix. This process can be explained by:

$$B = L \cup C \cup P = b_1 b_2 \ldots b_m, \tag{2.11}$$

where $b_i = 0$ or 1. The m is bit length of B. Here C is appended to the end of L, and append P to the end of C. The bit stream B is embedded into the difference values.

2.2.6 Inverse Integer Transforms

When bit stream B is embedded into difference values, this method uses an inverse integer transform shown below to calculate new pixel values, so that a new image with embedding information inside is created. Then the reversible data embedding process is finished.

$$x = l + \lfloor (h + 1)/2 \rfloor, \quad y = l - \lfloor h/2 \rfloor \tag{2.12}$$

2.2.7 Data Embedding Chart Flow

As seen above, only changeable difference values are modified. Non-changeable difference values and all integer average values are unchanged. For a changeable difference value, either a new LSB is embedded by difference expansion or its original LSB is replaced. Thus after embedding, all embedded information is in the LSBs of changeable difference values. By collecting the LSBs of changeable difference values, the decoder will be able to recover the embedded bit steam B. After getting the bit stream

B, decoder can recover original image exactly. This data embedding method based on the DE technique can be clearly shown by chart Fig 2.3.

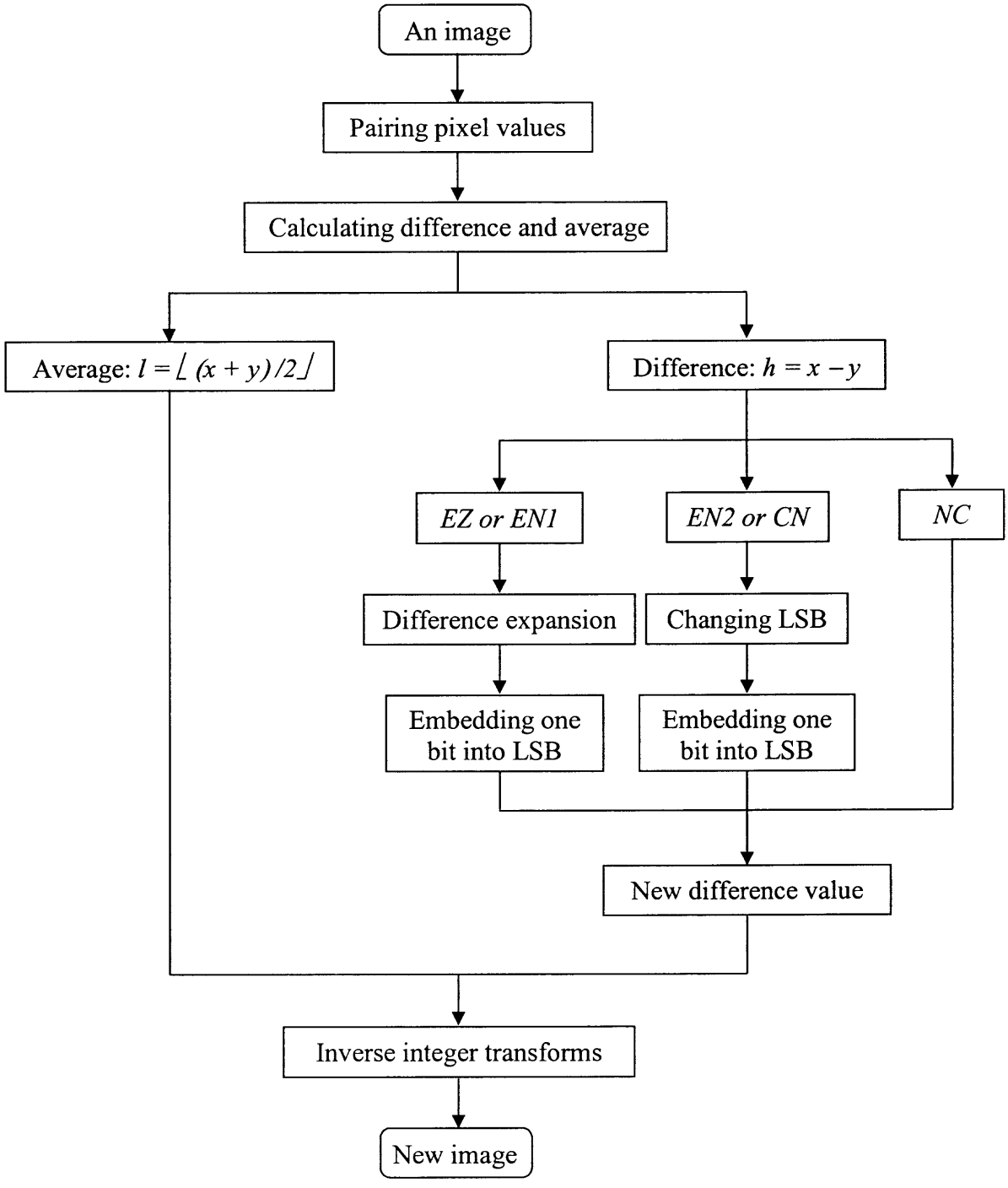

Fig 2.3 Data embedding chart flow

2.3 Decoding and Authentication

Referring to Table 2.1, the embedded bit stream *B* can be retrieved, by collecting the LSBs of all changeable difference values. From stream *B*, the location map stream *L*, the original LSBs stream *C*, and the embedded data (payload) can be decoded. The location map gives the location information of all expanded difference values and changeable difference values. For expanded difference values, an integer division by 2 will give back their original values; for other changeable difference values, their original LSBs can be restored from the bit stream *C*. And for the non-changeable difference values, there are nothing happened on them. After all changeable difference values have been restored the original image can be restored exactly.

Sharing with the information from the data embedding process, the decoding and authentication process has similar working flow as embedding process. It consists of five steps.

1. Calculating the average and difference values.
2. Creating two disjoint sets of difference values, *CH*, and *NC*.
3. Collecting LSBs of all difference values in *CH*, and forming a binary bit stream *B*.
4. Decoding the location map from *B* by JBIG2 decoder.
5. Getting matrix h and recovering the original image.

The process flow chart is shown as Fig 2.4. After getting the original difference values, together with the average values got from step 1, reversible integer transform to get original image can be used.

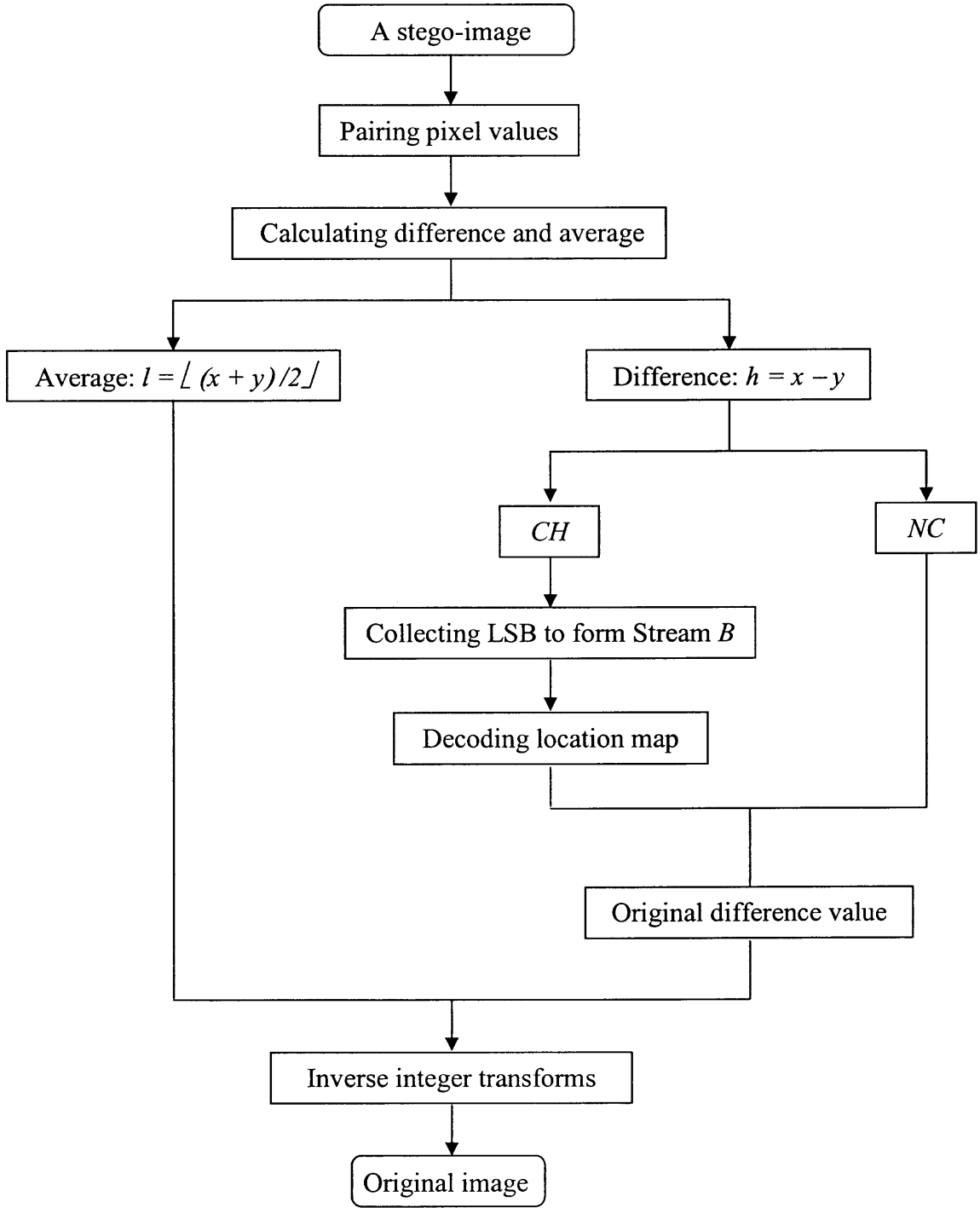

Fig 2.4 Decoding chart flow

CHAPTER 3

DISCUSSIONS AND RESULTS FROM EMBEDDING METHOD

The reversible data embedding method based on the DE technique is a novel data embedding method for digital images. It explores the redundancy in digital images to achieve very high embedding capacity, and at the same time keep the distortion (the quality degradation on digital images after data embedding) low. Some important characteristics of this method will be discussed in this chapter. At the same time, this method is used in some gray level images like "Lena" "Barbara" "Baboon" and "Boat", and get simulation results.

3.1 Discussions and Analysis

Compared with other reversible data embedding methods, the DE method performs well for both capacity limit and visual effect. This method is also very flexible for different payload sizes.

3.1.1 Embedding Capacity Limit

The bit stream B has a bit length of $(|L| + |C| + |P|)$, where $|.|$ is the bit length or numbers of elements of a set. From the data-embedding algorithm [1], the total embedding capacity will be $(|EZ| + |EN1| + |EN2| + |CN|)$. So if we use the DE method to successfully embed some information into an image, there must be

$$|L| + |C| + |P| \leq |EZ| + |EN1| + |EN2| + |CN| \tag{3.1}$$

Assume the total number of 1 and –2 in $EN2$ and CN is N, then

$$|EN2| + |CN| = |C| + N \tag{3.2}$$

Substitute into (3.1),

$$|L| + |C| + |P| \leq |EZ| + |EN1| + |C| + N \tag{3.3}$$

Then we get

$$|P| \leq |EZ| + |EN1| + N - |L| \tag{3.4}$$

Thus the payload size is upper bounded by the sum of the number of expandable difference values equal to 0 and -1, other selected expandable difference values and the number of not selected or not expandable difference value h equal to 1 or –2, minus the bit length of the location map. From equation (3.4) most of the payload capacity is from EZ and $EN1$, which are used for DE in the data embedding process.

3.1.2 Expandable Difference Value Selection

Normally, natural images tend to be continuous and smooth. The correlation between adjacent pixels is very strong, and the difference values of neighboring pixel values are usually small. And from the algorithm introduced in the previous chapter, a small difference value is most likely to be expandable. For example, if the integer average of two neighboring pixel values is in the range of $34 \leq l \leq 220$, and their difference value is in the range of $-34 \leq h \leq 34$, then

$$|2*h + b| \leq 2*|h| + b \leq 2*34 + 1 = 69 \leq min\,(2(255 - l), 2l + 1) \tag{3.5}$$

for both $b = 1$ and 0. So in this case the difference value between two-pixel values h must be expandable. If the integer average of two neighboring pixel values is in the range of $5 \leq l \leq 249$, and their difference value is in the range of $-5 \leq h \leq 5$, then

$$| 2*h + b| \leq 2*|h| + b \leq 2*5 + 1 = 11 \leq min\,(2(255 - l), 2l + 1) \qquad (3.6)$$

for both $b = 1$ and 0. From a statistical survey, natural grayscale images usually have over 99 percent expandable difference values.

The location map is a one-bit bitmap. It can be efficiently compressed by JBIG2, the new international standard for lossless compression of bi-level images. JBIG2 supports model-based coding to permit compression ratios up to three times those of the previous standard for lossless compression. For more detail on JBIG2, we refer to [14].

From the experiment, for any image if all those expandable difference values are selected to do data embedding, most of values in its location map will be "1". After using the JBIG2 compressor, a very high compressible rate will be achieved. Normally it will reach 1:200. For a 1-layer stego-image, this method can get nearly a 0.5-bpp data-embedding rate, and for multi-layer stego-images, this method can get an even higher data embedding rate. This will be discussed in the following section.

When some data at a rate no less than 0.5-bpp need to be embedded, some small differences from the total will be selected to do the data embedding. Here a threshold T, separating EN into $EN1$ and $EN2$, is introduced by

$$EN1 = \{h \in EN: |h| \leq T\},\ EN2 = \{h \in EN: |h| > T\}. \qquad (3.7)$$

For a payload P, we start with a small threshold T, and then increase T gradually until the payload size is satisfied.

Please note that with a different threshold T, the location map L also changes, as does its bits length $|L|$.

3.1.3 Multi-Layer Data Embedding

The multi-layer data embedding for this reversible data embedding method can be applied if the embedding bit rate is more than 0.5-bpp. That means that the reversible data embedding method can be applied to an image more than once for multiple embedding.

For an already embedded image, it can be embedded again with another payload. Even for one payload, the payload can be divided into several parts and use multiple embedding to embed them. In order to keep a low visual distortion rate, the difference between two pixels values should not grow too large after data embedding. So a different pair of pixel values in data embedding step 1 between two continuous data embedding layers should be selected. A recommended approach is to use a complement pairing. For example, if the image is embedded with a vertical pair, then a horizontal pair for the next embedding can be used.

As each embedding has a payload capacity limit less than 0.5-bpp, a multiple embedding will have a payload capacity limit less than *M/2*bpp, where *M* is the number of embeddings. In practice, the payload capacity limit of each embedding will decrease gradually, because the redundancy in pixel values becomes less and less.

As reversible data embedding, in order to help the decoder to determine whether or not it is a multiple embedding and which pair to choose, this method needs to code a 16 bits header and embed it before the location map L. This header can be represented by *H*. The bit stream *B* now becomes

$$B = H \cup L \cup C \cup P.$$

For the original image (1-layer stego-image), H is set to 0. The pairing pattern of the original image will have H at the second embedding (2-layer stego-image). The second embedding will have H at the third embedding (3-layer stego-image), and so on.

During the process of decoding, the B bit stream is extracted from the image. The first 16 bits in B is the pairing pattern H. After the first 16 bits are extracted, the location map is decoded, together with average values, reconstruct a restored image, and authenticate the content. If the content is authentic, H is used as the pairing pattern to decode the restored image again. The decoding process will continue until $H = 0$ or when tampering has been discovered. If $H = 0$, and no tampering has been discovered during the whole decoding process, then the final restored image will be exactly the same as the original image, pixel by pixel, bit by bit.

3.2 Experimental Result from Reversible Data Embedding

Based on the DE technique, the experiments were preformed by embedding and decoding an actual bit stream to verify the correctness of the algorithm. For test purposes, the MATLAB function *rand()* was used to generate pseudo random number, then rounded them to nearest integers to generate a payload.

The experiment was done on the pictures 512*512 "Lena", 512*512 "Barbara", 512*512 "Baboon", and 512*512 "Boat". The embedded payload size, its corresponding bit rate, and the peak signal to noise ratio (*PSNR*) of the embedded image are listed in Table 3.1, 3.2, 3.3, and 3.4, respectively. The experimental data from Jun Tian were obtained, who introduced the DE technique into data embedding and proposed the reversible data embedding method based on the DE technique. Comparisons between my

result and Jun Tian's result [1] on pictures 512*512 "Lena" and 512*512 "Barbara" are listed as Fig 3.1 and Fig 3.2. Here only 1-layer data embedding was considered. A location map example is given, where black dots mean that parts cannot be used for DE. It is listed in Fig 3.3. Finally, 4 stego-images based on the DE technique are presented, 512*512 "Lena", 512*512 "Barbara", 512*512 "Baboon", and 512*512 "Boat", respectively. The comparison between original images and their stego-images are listed from Fig 3.4 to Fig 3.11, respectively.

Table 3.1 Embedded Payload Size vs. PSNR of Embedded "Lena" Image.

Payload Size (bits)	39509	52674	71283	87845	100321	112136	123328	128390
Bit Rate (bpp)	0.1507	0.2009	0.2719	0.3351	0.3827	0.4278	0.4705	0.4898
PSNR (dB)	44.20	43.47	42.37	41.21	40.08	38.70	37.06	35.94

Table 3.2 Embedded Payload Size vs. PSNR of Embedded "Barbara" Image.

Payload Size (bits)	25420	36960	50178	57954	71452	95397	111601	129749
Bit Rate (bpp)	0.097	0.1410	0.1914	0.2211	0.2725	0.3639	0.4257	0.4950
PSNR (dB)	44.17	42.80	41.09	40.10	38.43	35.35	33.25	31.57

Table 3.3 Embedded Payload Size vs. PSNR of Embedded "Baboon" Image.

Payload Size (bits)	8569	23060	47360	66570	82287	106415	119728	124121
Bit Rate (bpp)	0.0327	0.088	0.1807	0.2539	0.3139	0.4059	0.4567	0.4735
PSNR (dB)	34.96	33.44	31.13	29.45	28.21	26.50	25.60	25.27

Table 3.4 Embedded Payload Size vs. PSNR of Embedded "Boat" Image.

Payload Size (bits)	9889	33854	60071	85723	108246	118185	128875
Bit Rate (bpp)	0.0377	0.1291	0.2292	0.3270	0.4129	0.4508	0.4916
PSNR (dB)	41.93	40.55	39.08	37.52	35.72	34.67	33.30

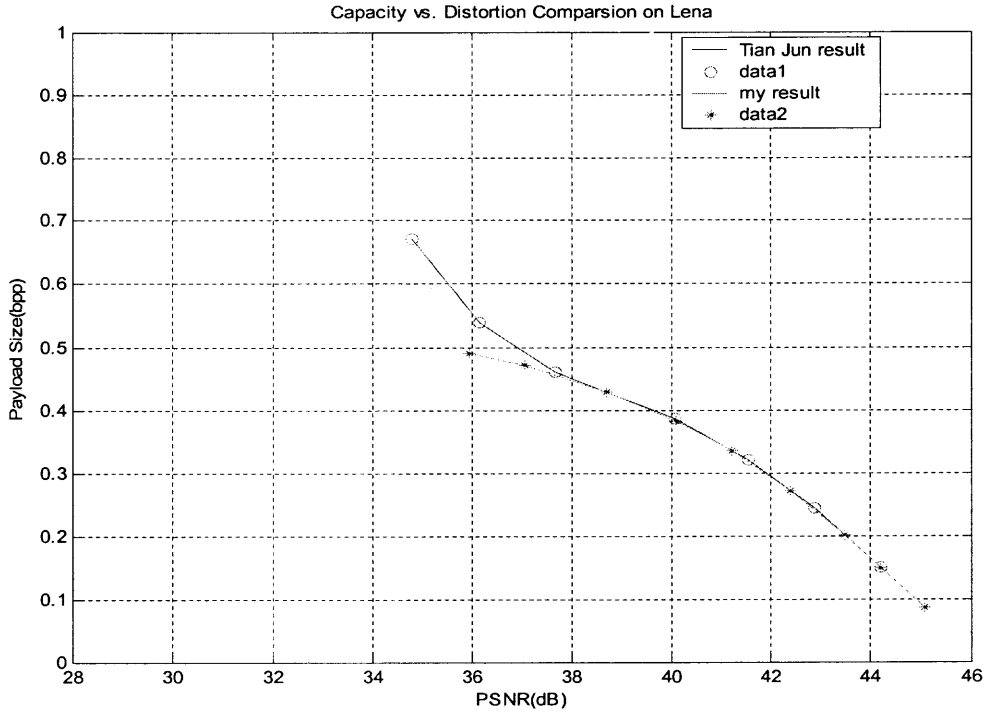

Fig 3.1 Capacity vs. Distortion comparison on "Lena"

In Fig 3.1, our simulation results are very close to the result from Jun Tian. Only when embedding bit rate is above 0.5-bpp, there is a big difference happened. The reason for that is: from that point, Jun Tian used 2-layer data embedding into "Lena" image, but

here we only used one-layer data embedding to get the steg-image. Another comparison on "Barbara" image can be found in Fig3.2.

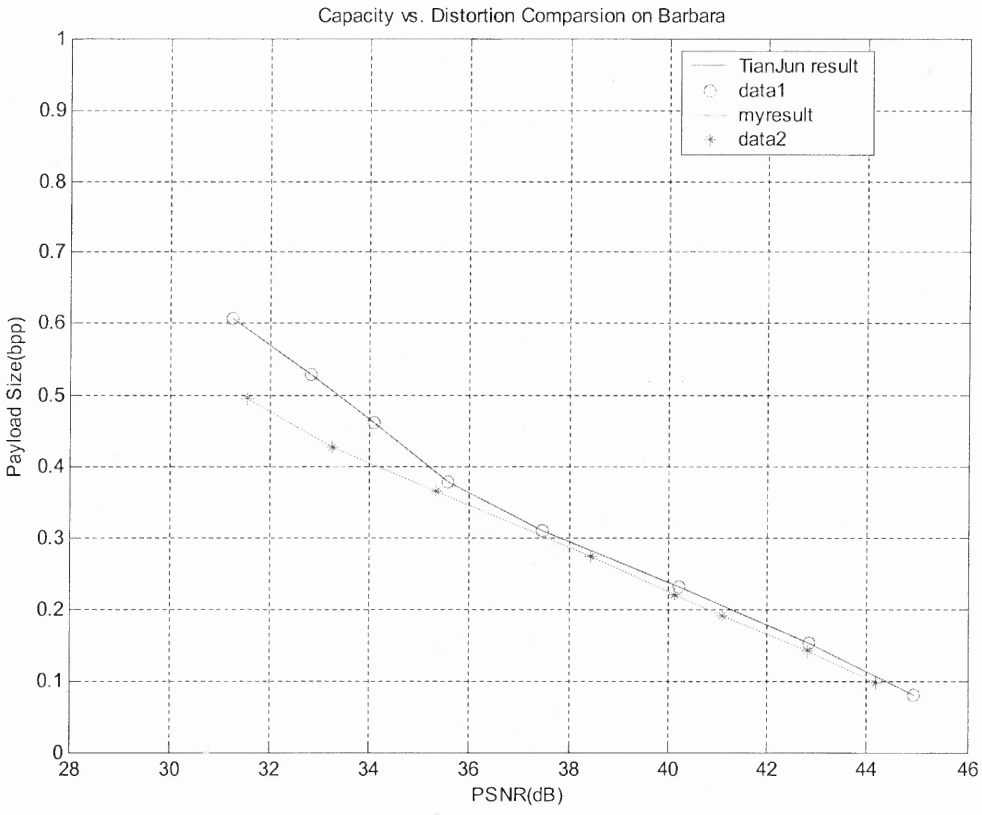

Fig 3.2 Capacity vs. Distortion comparison on "Barbara"

Fig 3.3 Location map of "Lena" with 87,845 bits (0.34bpp) payload

Fig 3.4 Original "Lena" image

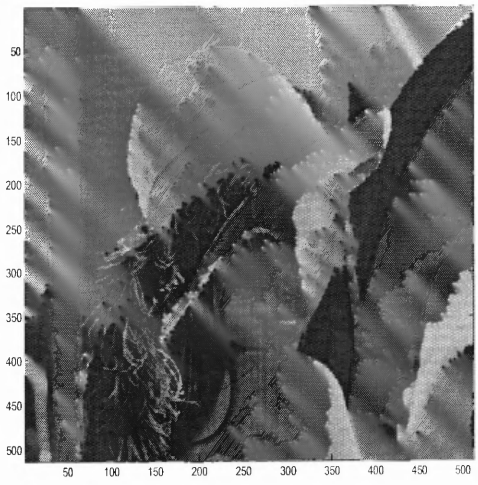

Fig 3.5 Embedded "Lena" with a payload 129946 bits

Fig 3.6 Original "Barbara" image

Fig 3.7 Embedded "Barbara" with a payload 129749 bits

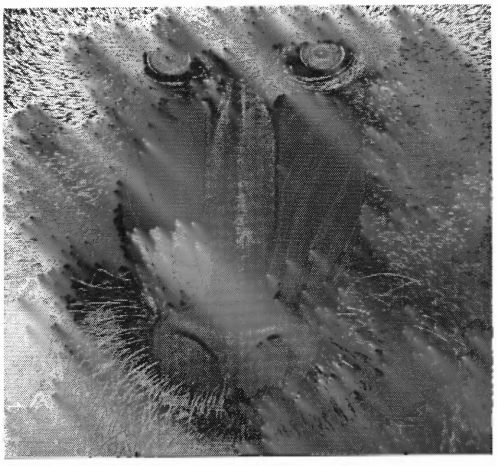

Fig 3.8 Original "Baboon" image

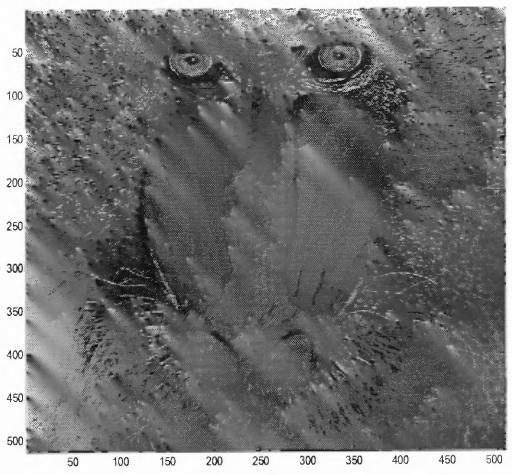

Fig 3.9 Embedded "Baboon" with a payload 124121 bits

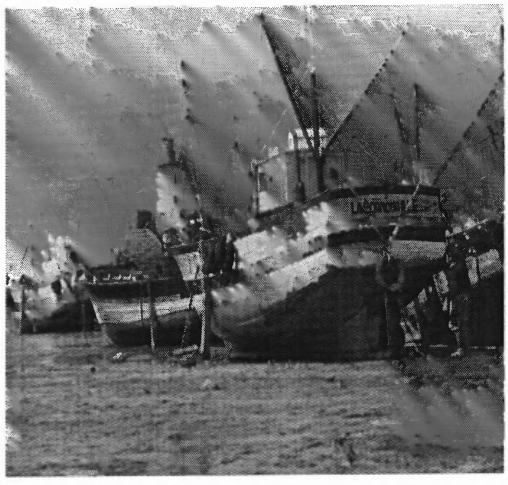

Fig 3.10 Original "Boat" image

Fig 3.11 Embedded "Boat" with a payload 128875 bits

CHAPTER 4

STEGANALYSIS BASED ON DIFFERENCE MOMENTS

In the previous two chapters, a high capacity, high visual quality, reversible data embedding method is introduced, which is based on difference expansion. The basic idea of this kind of data-embedding method is to explore the redundancy in digital image and compute the difference between two pixel values to embed information. As this reversible embedding method will only select the small difference value to do embedding, it gets a good result in keeping distortion at a low level. In order to decode images correctly after data embedding, this method needs to use the JBIG2 image compressor to compress a location map and embed it together with message into difference values. So when decoding, after getting the location map, one can restore the original image exactly. The results in chapter 3 show this method can get very good vision performance after data embedding, when using 1-layer data embedding.

4.1 Overview

In recent years, the concept of image steganalysis has been introduced because of the practical need for detecting the stego-image. Steganalysis is a scientific technology to decide if a medium carries some hidden message. In counter terrorism work, steganalysis can also be served as an effective way to judge the security performance of steganographic techniques. In this chapter, image steganalysis will be discussed, with special focus on the reversible embedding method introduced in previous two chapters.

From the discussion in previous chapters, when there is a difference expansion happened in an image, or when there is data embedded in an image, there will be an obvious change in the difference values between the original images and the stego-images. Compared with the original image, in the stego-image the difference values between two neighboring pixels will be enlarged. Making use of this fact, the difference value was calculated in an image and a 12 dimensional feature vector was obtained for the steganalysis. After observing the performance, a Bayes classifier is set up for classification. All of the 1096 images are used in the coreldraw image database [15] for our extensive experimental work.

4.2 12-D Feature Vector

As discussed previously, steganalysis is a method to decide if an image contains secret messages. It is also a way to classify a given image into two different categories: the original image and the stego-image. In this sense, steganalysis is actually a matter of pattern classification in which a key issue is how to select effective features. Here the name feature is often used in the pattern recognition literature to denote a descriptor.

4.2.1 Multi-dimensional Vector and Integer Haar Wavelet Transform

As may be evident by now, pattern features or vectors can be generated in numerous ways. But not all features can be used for pattern classification. Selecting the features on which to base each component of a pattern vector has a profound influence on the eventual performance of pattern classification (object recognition). A good feature should be sensitive to the hidden message while not sensitive to other operations such as compression. At the same time, the feature should be applicable to all kind of images.

During the experimental process, it is hardly possible to use one single feature to achieve a highly correct classification rate. A multi-dimensional (M-D) feature vector should be used. Each image is a sample point in the M-D feature space. Steganalysis has thus become a pattern classification method in the M-D feature space.

We have already known the data embedding method is based on difference expansion. The basic algorithm is an integer Haar wavelet transform, which is also called a reversible integer transform. The equation can be represented as:

$$l = \lfloor (x + y)/2 \rfloor \quad h = x - y. \tag{4.1}$$

$$x = l + \lfloor (h + 1)/2 \rfloor, \quad y = l - \lfloor h/2 \rfloor. \tag{4.2}$$

In chapter 2 and 3, this method accomplishes the data embedding mostly by adding one bit into LSB. So when an image undergoes difference expansion, it will bring some changes in the difference values in its stego-image. From this point, the integer Haar wavelet transform can be used to get difference values as feature vectors.

4.2.2 Analysis of Difference Values

From statistical survey, more than 99 percent of difference values in any natural image can be used for difference expansion. When an image undergoes difference expansion, it will bring some changes in the difference values. For example, consider the difference expansion operation below:

$$h' = 2 * h + b, \tag{4.3}$$

If the original difference h equals to 0, after difference expansion it will change into a new difference h', which equals 0 (when $b = 0$) or 1 (when $b = 1$). Similarly, if the

original difference h equals to -1, after difference expansion it will change into a new difference h', which will equal -2 (when $b = 0$) or -1 (when $b = 1$).

The location map is always embedded first no matter how much message will be embedded. Nearly all of the small differences, such as where absolute values are less than or equal to 3, will be used for the difference expansion. From experiment, the obvious change can be easily observed in the difference values between the original images and the stego-images, especially for smaller difference values. Some small changes in difference values after embedding (difference expansion) are listed as Table 4.1.

Table 4.1 Difference Values Comparison Between Original and Stego-images.

Original difference value h	Embedded difference value h'
$h = 0$	$h' = 0$ *or* $h' = 1$
$h = -1$	$h' = -2$ *or* $h' = -1$
$h = 1$	$h' = 2$ *or* $h' = 3$
$h = -2$	$h' = -4$ *or* $h' = -3$
$h = 2$	$h' = 4$ *or* $h' = 5$
$h = -3$	$h' = -6$ *or* $h' = -5$
$h = 3$	$h' = 6$ *or* $h' = 7$

From the table, it can be seen that when a difference expansion method is used to embed a random stream into an image, there will be some rules that define the relationship between the original and the embedded difference values. All the differences whose values equal 0 or 1 after embedding come from original difference h whose values equal 0. If *Num* $(h=0)$ is used to represent the total number of original differences h

whose values equal "0", *Num (h'=0)* represent the total number of embedded difference *h'* whose values equal "0", and *Num (h'=1)* to represent the total number of embedded differences *h'* whose values equal "1", there should be the Equation (4.4).

$$Num\ (h=0) = Num\ (h'=0) + Num\ (h'=1) \tag{4.4}$$

And the embedded stream is a random stream, so in an stego-image, *Num (h'=0)* should be nearly equal *Num (h'=1)*. The *Num (h=0)* in an original image is nearly 2 times compared to the *Num (h'=0)* in its stego-image. In mathematics, they can be represented by:

$$Num\ (h'=0) \approx Num\ (h'=1) \tag{4.5}$$

$$Num\ (h=0) \approx 2\ Num\ (h'=0) \tag{4.6}$$

Similarly, there are other rules:

$$Num\ (h=-1) \approx 2\ Num\ (h'=-1) \tag{4.7}$$

$$Num\ (h'=-2) \approx Num\ (h'=-1) \approx 1/2 * Num\ (h=-1) \tag{4.8}$$

$$Num\ (h'=0) \approx Num\ (h'=1) \approx 1/2 * Num\ (h=0) \tag{4.9}$$

$$Num\ (h'=2) \approx Num\ (h'=3) \approx 1/2 * Num\ (h=1) \tag{4.10}$$

$$Num\ (h'=-4) \approx Num\ (h'=-3) \approx 1/2 * Num\ (h=-2) \tag{4.11}$$

$$Num\ (h'=4) \approx Num\ (h'=5) \approx 1/2 * Num\ (h=2) \tag{4.12}$$

$$Num\ (h'=-6) \approx Num\ (h'=-5) \approx 1/2 * Num\ (h=-3) \tag{4.13}$$

$$Num\ (h'=6) \approx Num\ (h'=7) \approx 1/2 * Num\ (h=3) \tag{4.14}$$

4.2.3 12-D Feature Vectors Selection for Steganalysis

In our proposed system, an integer Haar wavelet transform is performed on the image under analysis. After getting the difference values, we will classify them into 12 groups, and calculate the total number in each group. *Num (h=0), Num (h=1), Num (h=-1), Num*

(h=2), Num (h=-2), Num (h=3), Num (h=-3), Num (4≤ |h|≤ 9), Num (10≤ |h|≤ 20), Num (21≤ |h|≤ 30), Num (31≤ |h|≤ 40) and *Num (41≤ |h|≤ 255)* are obtained. If *Num (h)* is used to represent the total number of all differences, then *Rate (Num (h=0))* can be used to represent the ratio of the difference whose value equals "0" to whole differences. In mathematics, it can be shown by below equation

$$Rate\ (Num\ (h=0)) = Num\ (h=0) \,/\, Num\ (h) \tag{4.15}$$

Similarly, *Rate (Num (h=1)), Rate (Num (h=-1)), Rate (Num (h=2)), Rate (Num (h=-2)), Rate (Num (h=3)), Rate (Num (h=-3)), Rate (Num (4≤ |h|≤ 9)), Rate (Num (10≤ |h|≤ 20)), Rate (Num (21≤ |h|≤ 30)), Rate (Num (31≤ |h|≤ 40))* and *Rate (Num (41≤ |h|≤ 255))* are gotten respectively. Those 12 feature ratios are used to build up 12-D feature vector for steganalysis.

Next how to use the proposed 12-D feature vector to classify original images and stego-images is discussed.

4.3 Bayes Classifier

Besides feature selection, the design of the classifier is another key element in pattern recognition. It affects the classification performance in terms of successful classification rate as well as computational complexity, and hence, implementation speed. Therefore, the classifier plays an important role in steganalysis. In this section, a Bayes classifier [16] is introduced because the embedded data basically follows a Gaussian distribution or can be approximated by a Gaussian distribution. The foundational knowledge of the Bayes classifier is given and its application in Gaussian pattern classes is present.

4.3.1 Foundation

The probability that an image x comes from an original image ω_1 is denoted $p(\omega_1 / x)$. If the pattern classifier decides that x came from stego-image ω_2 when it actually came from original image ω_1, it incurs a loss. So the loss incurred in assigning x to stego-image pattern ω_2 is

$$r_2(x) = p(\omega_1 / x) \tag{4.16}$$

If the pattern classifier decides that x came from original image ω_1 when it actually came from stego-image ω_2, it incurs a loss. So the average loss incurred in assigning x to stego-image ω_1 is

$$r_1(x) = p(\omega_2 / x) \tag{4.17}$$

From basic probability theory, $p(A/B) = [p(A)p(B/A)]/p(B)$. Using this expression, equations (4.16) and (4.17) are written in the form

$$r_j(x) = \frac{1}{p(x)} p(x/\omega_i)p(\omega_i) \tag{4.18}$$

where $i = 1$ or $2, j = 1$ or 2 and $i \neq j$. Because $1/p(x)$ is positive and common to $r_1(x)$ and $r_2(x)$, it can be dropped from equation (4.18) without affecting the relative order of these functions from the smallest to largest value. The expression for the average loss then reduces to

$$r_j(x) = p(x/\omega_i)p(\omega_i) \tag{4.19}$$

The classifier has 2 possible classes to choose from for an unknown pattern (original image or stego-image). If it computes $r_1(x)$ and $r_2(x)$ for each pattern x and assigns the pattern to the class with the smallest loss, the total average loss with respect to all decisions will be minimum. The classifier that minimizes the total average loss is called the Bayes classifier. Thus the Bayes classifier assigns an unknown pattern x to class ω_i if $r_i(x) < r_j(x)$. In other words, x is assigned to class ω_i if

$$p(x/\omega_i)p(\omega_i) > p(x/\omega_j)p(\omega_j) \tag{4.20}$$

where $i = 1$ or $2, j = 1$ or 2 and $i \neq j$. And then we see that the Bayes classifier for a 0-1 loss function is nothing more than computation of decision functions of the form

$$d_j(x) = p(x/\omega_j)p(\omega_j) \qquad j = 1 \text{ or } 2 \tag{4.21}$$

where a pattern vector x is assigned to the class whose decision function yields the largest numerical value.

4.3.2 Bayes Classifier for Gaussian Pattern Classes

Here we focus on two pattern classes governed by Gaussian densities, with means m_1 and m_2 and standard deviations σ_1 and σ_2, respectively. From equation (4.21) the Bayes decision functions have the form

$$d_j(x) = p(x/\omega_j)p(\omega_j) \tag{4.22}$$

$$= \frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(x-m_j)^2}{2\sigma_j^2}} p(\omega_j) \tag{4.23}$$

for j = 1 or 2. In the n-dimensional case, the Gaussian density of the vectors in the *j*th pattern class has the form

$$p(x/\omega_j) = \frac{1}{(2\pi)^{n/2} \mid C_j \mid^{1/2}} e^{-\frac{1}{2}(x-m_j)^T C_j^{-1}(x-m_j)} \tag{4.24}$$

where each density is specified completely by its mean vector m_j and covariance matrix C_j, which are defined as

$$m_j = E_j\{x\} \tag{4.25}$$

and

$$C_j = E_j\{(x-m_j)(x-m_j)^T\} \tag{4.26}$$

where $E_j\{\cdot\}$ denotes the expected value of the argument over the patterns of class ω_j. In equation (4.24), n is the dimensionality of the pattern vectors, and $\mid C_j \mid$ is the determinant of the matrix C_j. Approximating the expected value E_j by the average value of the quantities in question yields an estimate of the mean vector and covariance matrix:

$$m_j = \frac{1}{N_j} \sum_{x \in \omega_j} x \tag{4.27}$$

and

$$C_j = \frac{1}{N_j} \sum_{x \in \omega_j} xx^T - m_j m_j^T \tag{4.28}$$

where N_j is the number of pattern vectors from class ω_j, and the summation is taken over these vectors.

According to Equation (4.21) the Bayes decision functions for class ω_j is $d_j(x) = p(x/\omega_j)p(\omega_j)$. This equation can take natural logarithm form.

$$d_j(x) = \ln[p(x/\omega_j)P(\omega_j)]$$

$$= \ln p(x/\omega_j) + \ln P(\omega_j) \tag{4.29}$$

This expression is same as Equation (4.21) in terms of classification performance because the logarithm is a monotonically increasing function. Substituting Equation (4.24) into Equation (4.29) yields

$$d_j(x) = \ln P(\omega_j) - \frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|C_j| - \frac{1}{2}[(x-m_j)^T C_j^{-1}(x-m_j)] \tag{4.30}$$

And the term $\frac{n}{2}\ln 2\pi$ is the same for all classes, so it can be eliminated from Equation (4.30), which then becomes

$$d_j(x) = \ln P(\omega_j) - \frac{1}{2}\ln|C_j| - \frac{1}{2}[(x-m_j)^T C_j^{-1}(x-m_j)] \tag{4.31}$$

for j = 1, 2,..., W. Equation (4.31) represents the Bayes decision functions for Gaussian pattern classes under the condition of a 0-1-loss function. Here x can be represented by the feature vector from an image.

4.3.3 Bayes Classifier Decision for Steganalysis

A Bayes classifier is used in our steganalysis. There are two classes, the original images and the stego-images, and because the term $\ln P(\omega_j)$ is the same for both the original images class and the stego-images class, it can be eliminated from Equation (4.31), which then becomes

$$d_j(x) = -\frac{1}{2}\ln|C_j| - \frac{1}{2}[(x-m_j)^T C_j^{-1}(x-m_j)] \tag{4.32}$$

for j =1, 2. If a features vector is set up as mentioned in the previous section, the covariance matrix for either the original images or the stego-images will be symmetric. Then the equation (4.32) becomes

$$d_j(x) = -\frac{1}{2}\ln|C_j| - \frac{1}{2}x^T C_j^{-1}x + x^T C_j^{-1}m_j - \frac{1}{2}m_j^T C_j^{-1}m_j \tag{4.33}$$

for j = 1,2. Here "1" means original images class, "2" means stego-images class. After calculating Equation (4.33), the Bayes classifier will assign an image to the original class, if $d_1(x) \geq d_2(x)$, otherwise it will assign this image to the stego-image.

CHAPTER 5

EXPERIMENTAL RESULTS

Experiments were performed based on the features vector and the Bayes classifier. A vertical pair embedding method was used on 100 512*512 gray level image. After certified the steganalysis method was efficient, the completed algorithm was set up. There are 4 different ways to perform data embedding based on the difference expansion method. So the data was embedded by pairing vertical, horizontal, clockwise diagonal and counter clockwise diagonal, respectively. A large number of images should be amenable to steganalysis if the steganalysis makes sense and is used practically. Therefore the CorelDraw image database was used as the experimental image set. This database contains 1096 images in total, including images of leisure activities, places, animals, foods, scenery, architecture and so on. In the experiments 896 images were randomly chosen for training purposes. The remaining 200 images were used for testing purposes.

5.1 Vertical Pairing Test

5.1.1 Data Training

In order to test the features vector and Bayes classier, 100 512*512 images were selected, including images of "Lena", "Barbara", "Baboon", "Boat", and so on. Based on the difference expansion method, vertical pairing was picked up to embed random data into the images. Here only 1- layer data embedding was considered. The multi-layer case will be discussed later. After getting 100 stego-images, 70 original images and 70

corresponding stego-images were randomly selected to train the data. The process of data training is as follows:

1. Setting up 70 original images and 70 stego-images for training.

2. In the 140-image training set, using the method mentioned in the previous chapter to set up a 12-D feature vector for every original image and every stego-image, respectively.

3. In those two feature vector groups, calculating the mean features vector $\mathbf{m}_1$ of the original image and the mean features vector $\mathbf{m}_2$ of the stego-image, respectively. In order to simplify the calculation, changing the features vector $\mathbf{m}_1$ and $\mathbf{m}_2$ into natural logarithm form.

4. Using the mean features vector to calculate the covariance matrix $\mathbf{C}_1$ of the original image and the covariance matrix $\mathbf{C}_2$ of the stego-image, respectively. The covariance matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ are 12*12 symmetric matrices, which will be used in the Bayes classifier equation later.

5. Bringing $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{C}_1$ and $\mathbf{C}_2$ into Bayes classifier equation (4.33)

$$d_j(x) = -\frac{1}{2}\ln|C_j| - \frac{1}{2}x^T C_j^{-1} x + x^T C_j^{-1} m_j - \frac{1}{2} m_j^T C_j^{-1} m_j .$$

Below equations were obtained.

$$d_1(x) = -\frac{1}{2}\ln|C_1| - \frac{1}{2}x^T C_1^{-1} x + x^T C_1^{-1} m_1 - \frac{1}{2} m_1^T C_1^{-1} m_1 \qquad (5.1)$$

and

$$d_2(x) = -\frac{1}{2}\ln|C_2| - \frac{1}{2}x^T C_2^{-1} x + x^T C_2^{-1} m_2 - \frac{1}{2} m_2^T C_2^{-1} m_2 \qquad (5.2)$$

Here equation (5.1) and equation (5.2) are called testing equations. Assuming there is an image with the feature vector $\boldsymbol{x}$, we bring $\boldsymbol{x}$ into equation (5.1) and (5.2). If $d_1(x) \geq d_2(x)$, the Bayes classifier will assign this feature vector to the original class, otherwise, it will assign it to the embedding class. The details about the decision-making will be discussed in the next section.

5.1.2 Classification Rate

From previous chapter, there are four pairing ways to calculate difference value to finish data embedding, vertical pairing, horizontal pairing, clockwise diagonal pairing and counterclockwise diagonal pairing, respectively.

In this experiment, vertical pairing was used. In order to efficiently separate original images and embedding, during testing the same pairing way was also used as the process of data embedding to calculate difference value between two continuous pixel values, further get 12-D feature vector of each testing image. 30 original images and 30 stego-images were set up as testing set. So there are totally 30 12-D original feature vectors and 30 12-D embedding feature vectors. Bring those testing vectors into testing equation (5.1) and (5.2) respectively. If $d_1(x) \geq d_2(x)$, the Bayes classifier will say the vector belong to original image, and the corresponding image is assigned to original image. Otherwise, assign the image to stego-image. Compared this result with the practical result (here we have already known which one is original image and which one is stego-image.). The correct classification rate was gotten. After tested 30 original images and 30 stego-images, the correct classification rate is shown in Table 5.1.

Table 5.1 Correct Classification Rate (Embedding Bit Rate from 0.01bpp to 0.5bpp)

Test	60 Images	
Errors	1	
Correct Classification Rate	98.33 %	
	Original Image	Stego-Image
Test	30	30
Errors	0	1
Correct Classification Rate	100 %	96.67 %

The image was checked where the error happened. This image's embedding bit rate is only 0.0153-bpp. The purpose of this experiment is to certify the efficiency of the feature selection and the Bayes classifier for steganalysis. The experiment result showed our steganalysis method based on a 12-D difference feature vector and Bayes classifier is efficient for the data-embedding algorithm using difference expansion.

5.2 Testing Based on Completed Algorithm

During the process of data embedding, there are four ways to calculate the difference value. These are vertical pairing, horizontal pairing, clockwise diagonal pairing and counterclockwise diagonal pairing, respectively. Normally it cannot be known which pairing way is used to do the difference expansion when the image is a stego-image. So in order to make the steganalysis algorithm complete, all possible case must be considered, which means we not only consider four pairing case, vertical pairing, horizontal pairing, clockwise diagonal pairing and counterclockwise diagonal pairing, but also need to consider the multi-layer data embedding case.

5.2.1 Completed Algorithm

Using a 1-layer data embedding algorithm, there are four ways to embed data into an image. These are vertical pairing, horizontal pairing, clockwise diagonal pairing and counterclockwise diagonal pairing. In the previous section, the efficiency of the 12-D feature selection and Bayes classifier for the vertical pairing case were certified. Accordingly, the same idea can be used in other cases and the other three classifiers can be set up. Then four different classifiers were gotten for four different pairing cases,

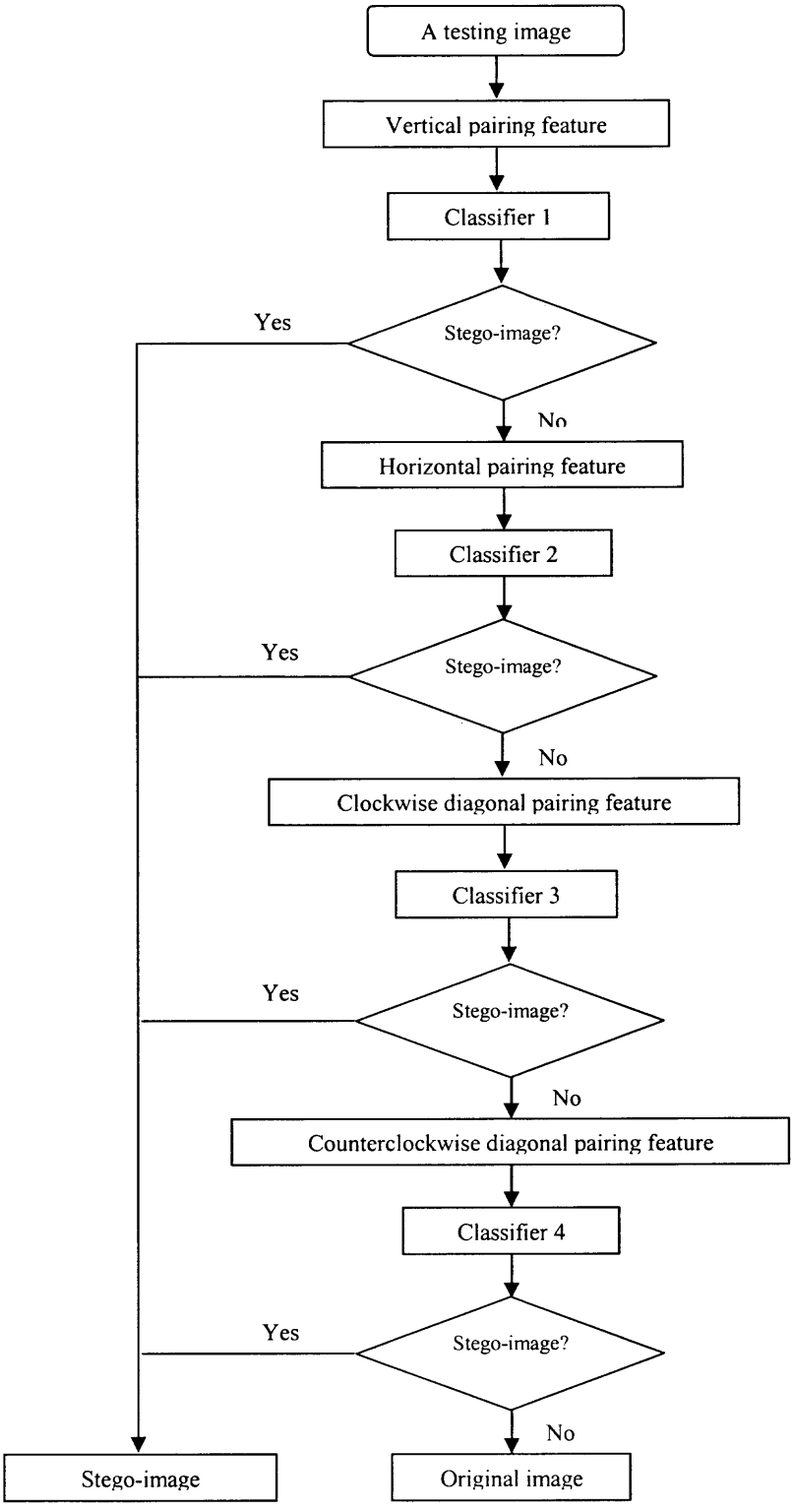

Fig 5.1 Completed algorithm flow

vertical classifier, horizontal classifier, clockwise diagonal classifier and counterclockwise diagonal classifier, respectively. They were named as classifier 1, classifier 2, classifier 3 and classifier 4, respectively. Assuming there was an image that needs to be tested. Four different pair methods can be used to get four different 12-D feature vectors, and then they were brought into the corresponding classifier. For example, the 12-D feature vector obtained using vertical pairing will be put into classifier 1 (vertical classifier), and the feature vector obtained using horizontal pairing will be put into classifier 2 (horizontal classifier), and so on. The final decision was made by using the results from all four different classifiers. The testing image will be assign to original image class only when all results from four different classifiers show this image to be the original image. Otherwise, it will be assigned to the stego-image class. The process of decision-making can be clearly shown in figure 5.1.

5.2.2 Data Training

To develop the complete algorithm, vertical pairing training, horizontal pairing training, clockwise diagonal pairing training and counterclockwise diagonal pairing training were set up respectively.

This time the CorelDraw image database was used for the experimental image set. This database contains 1096 images in total, including images of leisure activities, places, animals, foods, scenery, architecture and so on. In the experiments 300 images were selected randomly to do vertical data embedding, 300 images to do horizontal data embedding, 300 images to do clockwise diagonal data embedding, and 196 images to do counterclockwise diagonal data embedding. Then 250 original images and 250 corresponding vertical stego-images were used for vertical training, 250 original images

and 250 corresponding horizontal stego-images for horizontal training, 250 original images and 250 corresponding clockwise diagonal stego-images for clockwise diagonal training, and 146 original images and 146 corresponding counterclockwise diagonal stego-images for counterclockwise diagonal training. The remaining 200 original images and 200 stego-images were used for testing. The process of data training is as below:

1. Set up 896 original images, 250 vertical stego-images, 250 horizontal stego-images 250 clockwise diagonal stego-images and 146 counterclockwise diagonal stego-images. Build up 8 image groups for training, including vertical original and stego-images groups, horizontal original and stego-images groups, clockwise diagonal original and stego-images groups, and counterclockwise diagonal original and stego-images groups, respectively.

2. In those 8 groups, use the method mentioned in previous chapter to set up 12-D feature vector for every image in each group, respectively.

3. In those 8 groups, calculate the feature mean vector $\mathbf{m}_1$ of the vertical original images and the feature mean vector $\mathbf{m}_2$ of the vertical stego-images, the feature mean vector $\mathbf{m}_3$ of the horizontal original images and the feature mean vector $\mathbf{m}_4$ of the horizontal stego-images, the feature mean vector $\mathbf{m}_5$ of the clockwise diagonal original images and feature mean vector $\mathbf{m}_6$ of clockwise diagonal stego-images, and the feature mean vector $\mathbf{m}_7$ of the counterclockwise diagonal original images and the feature mean vector $\mathbf{m}_8$ of the counterclockwise diagonal stego-images respectively. In order to simplify the calculation, change the feature mean vectors $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{m}_3$, $\mathbf{m}_4$, $\mathbf{m}_5$, $\mathbf{m}_6$, $\mathbf{m}_7$ and $\mathbf{m}_8$ to their natural logarithm forms.

4. Using those mean feature vector to calculate the corresponding covariance matrix $\mathbf{C}_1$, $\mathbf{C}_2$, $\mathbf{C}_3$, $\mathbf{C}_4$, $\mathbf{C}_5$, $\mathbf{C}_6$, $\mathbf{C}_7$ and $\mathbf{C}_8$ respectively.

5. Bring $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{C}_1$ and $\mathbf{C}_2$ into the Bayes classifier equation (4.33) to set up the vertical classifier. Similarly, use $\mathbf{m}_3$, $\mathbf{m}_4$, $\mathbf{C}_3$, and $\mathbf{C}_4$ to set up horizontal classifier, $\mathbf{m}_5$, $\mathbf{m}_6$, $\mathbf{C}_5$ and $\mathbf{C}_6$ to set up clockwise diagonal classifier, and $\mathbf{m}_7$, $\mathbf{m}_8$, $\mathbf{C}_7$ and $\mathbf{C}_8$ to set up counterclockwise diagonal classifier.

5.2.3 Training Results

The training result comparisons between the12-D feature mean vector of the original images and the 12-D feature mean vector of the stego-images are shown in Table 5.2, Table 5.3, Table 5.4 and Table 5.5, respectively.

Table 5.2 Vertical Pairing Training Result Comparison Between 12-D Feature Mean Vector of Original Images and 12-D Feature Mean Vector of Its Stego-Images.

	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6
Vertical Pairing Feature Mean Vector of Original Images	-2.0338	-2.6496	-2.6374	-3.0927	-3.0985	-3.3979
Vertical Pairing Feature Mean Vector of Stego-Images	-2.7246	-2.729	-3.3319	-3.3429	-3.329	-3.3415
	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12
Vertical Pairing Feature Mean Vector of Original Images	-3.4148	-1.5746	-2.0923	-3.2398	-4.0268	-3.9501
Vertical Pairing Feature Mean Vector of Stego-Images	-3.791	-1.6164	-1.8413	-2.418	-2.8474	-3.15

Table 5.3 Horizontal Pairing Training Result Comparison Between 12-D Feature Mean Vector of Original Images and 12-D Feature Mean Vector of Its Stego-Images.

	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6
Horizontal Pairing Feature Mean Vector of Original Images	-1.9016	-2.4297	-2.4307	-2.9455	-2.948	-3.321
Horizontal Pairing Feature Mean Vector of Stego-Images	-2.5932	-2.5955	-3.1257	-3.1186	-3.1214	-3.1231
	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12
Horizontal Pairing Feature Mean Vector of Original Images	-3.324	-1.5984	-2.1661	-3.2449	-3.9466	-3.6279
Horizontal Pairing Feature Mean Vector of Stego-Images	-3.6399	-1.5282	-1.8808	-2.4826	-2.8811	-3.0134

Table 5.4 Clockwise Diagonal Pairing Training Result Comparison Between 12-D Feature Mean Vector of Original Images and 12-D Feature Mean Vector of Its Stego-Images.

	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6
Clockwise Diagonal Pairing Feature Mean Vector of Original Images	-2.0744	-2.6208	-2.5983	-3.0945	-3.0842	-3.4265
Clockwise Diagonal Pairing Feature Mean Vector of Stego-Images	-2.7655	-2.7695	-3.292	-3.3119	-3.2909	-3.314
	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12
Clockwise Diagonal Pairing Feature Mean Vector of Original Images	-3.4316	-1.6015	-2.041	-3.0422	-3.6993	-3.2563
Clockwise Diagonal Pairing Feature Mean Vector of Stego-Images	-3.7782	-1.6315	-1.8631	-2.3431	-2.6759	-2.7111

Table 5.5 Counterclockwise Diagonal Pairing Training Result Comparison Between 12-D Feature Mean Vector of Original Images and 12-D Feature Mean Vector of Its Stego-Images.

	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5	Feature 6
Counterclockwise Diagonal Pairing Feature Mean Vector of Original Images	-2.1508	-2.5694	-2.5778	-3.0698	-3.0562	-3.4307
Counterclockwise Diagonal Pairing Feature Mean Vector of Stego-Images	-2.8418	-2.8437	-3.2733	-3.2598	-3.2682	-3.2635
	Feature 7	Feature 8	Feature 9	Feature 10	Feature 11	Feature 12
Counterclockwise Diagonal Pairing Feature Mean Vector of Original Images	-3.4089	-1.6205	-2.0822	-3.0539	-3.6673	-3.0931
Counterclockwise Diagonal Pairing Feature Mean Vector of Stego-Images	-3.7492	-1.619	-1.8891	-2.3769	-2.6941	-2.6258

5.2.4 Classification Rate

After setting up four classifiers for the completed algorithm, we tested 200 original images and 200 corresponding stego-images. Four different 12-D feature vectors of each image were initially calculated using four different pair ways, then they were brought into corresponding classifiers to judge whether it belongs to the original image class or to the stego-image class. When this result was compared with the practical result, the correct classification rate 99.75% was obtained as shown in Table 5.6. The completed algorithm is efficient and accurate.

Table 5.6 Completed Algorithm Correct Classification Rate (Embedding Bit Rate from 0.01bpp to 0.5bpp).

Test	400 Images				
Errors	1				
Correct Classification Rate	99.75 %				
	Original Images	Vertical Stego-Images	Horizontal Stego-Images	Clockwise Diagonal Stego-Images	Counter-Clockwise Diagonal Stego-Images
Test	200	50	50	50	50
Errors	1	0	0	0	0
Correct Classification Rate	99.5 %	100 %	100 %	100 %	100 %

5.2.5 Discussion of Multi-Layer Case

From chapter 3, the reversible data embedding method based on difference expansion can embed data in multi-layer way. It is necessary for us to further analyze the multi-layer data embedding case to make steganalysis method complete.

Although the multi-layer stego-image will be a little different from a 1-layer stego-image, like the 1-layer data embedding case, after data embedding the difference between two pixel values is enlarged compared with original image. From the analysis in the previous section, the steganalysis method focused on this change in difference value, so the completed algorithm obtained from the 1-layer case will still work in the multi-layer case.

12 original images and 12 two-layer and three-layer stego-images were set up to certificate this point test. The classification rate was 100 percent.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In chapter 2 and chapter 3 of this thesis, a low computational complexity, multi-layer reversible data-embedding algorithm was introduced for digital images. A noticeable difference between this method and other methods is that this method does not need to compress the original values of the embedding area. Due to its hierarchical structure and efficient embedding, both the embedding capacity and visual quality of embedded images are among the best in the literature. In chapter 4 and chapter 5, an efficient steganalysis system was proposed to classify the DE stego-images from the original images. The difference between two neighboring pixel values in an image was calculated, and an M-dimensional (currently M=12) feature vector was set up. A Bayes classifier in this system was built up for decision-making. Extensive experimental work demonstrated that the steganalysis system is effective for separating the DE stego-images from the original images. The correct selection rate was 99.75%.

6.2 Future Work

In this thesis, the steganalysis system was focused on differentiating the stego-images generated by the DE scheme from the original images. Currently there are some other reversible data embedding methods. There are also some lossy data embedding methods.

For future work, many more image data embedding algorithms will be investigated and tested. The feature set will be improved in order to achieve a higher

correct detection rate for other data embedding cases. We will also investigate if we should increase dimensionality and how to optimize that. In addition, further investigation on more powerful classifiers will be conducted in order to enhance the performance. Our final objective is to set up a steganalysis system that can blindly detect stego-images from the original images with a high and reliable success rate and that can handle various images and data embedding algorithms.

REFERENCES

[1] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.,* vol.13, no. 8, pp. 890-896, Aug. 2003.

[2] C. W. Honsinger, P. W. Jones, M. Rabbani, and J. C. Stoffel, "Lossless recovery of an original image containing embedded data," U.S. Patent 6 278 791, Aug. 2001.

[3] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding-new paradigm in digital watermarking,"*EURASIP J. Appl. Signal Processing,* vol. 2002, no. 2, pp. 185-196, Feb. 2002.

[4] J. Dittmann, M. Steinebach, and L. Ferri, "Watermarking protocols for authentication and ownership protection based on timestamps and holograms," in *Security and Watermarking of Multimedia Contents IV-Proc. SPIE,* E. J. Delp III and P. W. Wong, Eds., vol. 4675, Jan. 2002, pp. 240-251.

[5] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Reversible data hiding," in *Proc. Int. Conf. Image Processing,* vol. II, Sept. 2002, pp. 157-160.

[6] X. Wu, "Lossless compression of continuous-tone images via context selection, quantization, and modeling," *IEEE Trans. Image Processing,* vol. 6, no. 5, pp. 656-664, May 1997.

[7] G. Xuan, J. Zhu, J. Chen, Y. Q. Shi, Z. Ni, and W. Su, "Distortionless data hiding based on integer wavelet transform," *IEEE Electronics Letters,* vol. 38, no. 25, pp. 1646-1648, Dec. 2002.

[8] van Leest, M. van der Veen, and F. Bruekers, "Reversible image watermarking," in *Proc. Int. Conf. Image Processing,* vol. II, Sept. 2003, pp. 731-734.

[9] J. Fridrich, M. Goljan and D. Hogea, "Steganalysis of JPEG Images: Breaking the F5 algorithm", *5th Information Hiding Workshop,* 2002, pp. 310-323.

[10] J. Fridrich, M. Golian and R. Du, "Detecting LSB steganography in color and gray-scale images", *Magazine of IEEE Multimedia Special Issue on Security,* Oct-Nov. 2001, pp. 22-28.

[11] R. Chandramouli and N. Memon, "Analysis of LSB based image steganography techniques", *Proc. Of ICIP 2001, Thessaloniki, Greece, Oct. 7-10, 2001.*

[12] H. Farid, "Detecting hidden messages using higher-order statistical models," *Proceedings of the IEEE Int'l. Conf. on Image Processing 02,* vol. 2, pp. 905-908.

[13] J. Harmsen, *Steganalysis of Additive Noise Modelable Information Hiding,* MS thesis, Rensselaer Polytechnic Institute, NY, thesis advisor William Pearlman, April 2003.

[14] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge, "The emerging JBIG2 standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 8, no. 7, pp. 838-848, 1998.

[15] Coreldraw Software, www.corel.com, Jan. 2005.

[16] Rafael Gonzalez and Richard Woods. *Digital Image Processing*, Addison-Wasley, New York, 1993.