

Towards Lower Bounds on Distortion in Information Hiding

A dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy at George Mason University

By

Younhee Kim
Master of Engineering
Ajou University, 2002
Bachelor of Science
Ajou University, 2000

Co-Director: Dr. Zoran Duric, Associate Professor
Co-Director: Dr. Dana Richards, Associate Professor
Department of Computer Science

Fall Semester 2008
George Mason University
Fairfax, VA

Copyright © 2008 by Younhee Kim
All Rights Reserved

Dedication

This dissertation is dedicated to my mother, Park Bong-sook, who has been always believed in me more than I do, and to my father, Kim Tae-kew, who has never stopped dreaming.

Acknowledgments

I owe my gratitude to a great many people who helped me to finish this journey.

My deepest gratitude is to my advisor, Dr. Zoran Duric. There are no words big enough to convey my indebtedness to him for the support, guidance, patience, and encouragement which he gave me throughout the entire process of research. I always feel so fortunate to have Dr. Duric as my advisor.

I am deeply grateful to my co-advisor, Dr. Dana Richards, for his brilliant and insightful comments, and for his help on countless revisions of this dissertation.

I would like to thank my supportive committee members, Dr. Arun K. Sood, Dr. Jana Kosecka, and Dr. Sushil Jajodia for their encouragement and guidance on every step for years.

I owe a special debt to Ms. Sidney L. Johnson, who offered valuable editing assistance. I am truly grateful to her for warm emotional support and her close reading of the entire draft.

I would like to gratefully thank Keith for all his help, encouragement and friendship. I am also grateful to my many friends at George Mason University during the course of my graduate studies: Nooshi, Ann, Therese, John, Christina, Ed, Dan, Nalini, Jaeyong, and all members of Korean Graduate Student Association for the support, warm wishes, and many valuable discussions. I am also thankful to the writing-group members for their encouragement.

I am thankful to the staff at Writing Center for their great help with my writing. I am also grateful to the Department of Computer Science, the Volgenau School of Information Technology and Engineering, and College of Health and Human Services at George Mason University for their various forms of financial support during my Ph.D. I would like to specifically recognize the help of Dr. Henry Hamburger, Dr. Sanjeev Setia, Dr. Daniel A. Menasce, Dr. Igor Griva, Dr. Susan E. Palsbo, and Dr. Naomi Lynn Gerber.

I have much appreciated the steady encouragement provided by my good friends, Kyounglim, Yoon-sook, and Ms. Cho Sun-ok. They have been my constant source of support as they did for last 20 years. I am also very thankful to Dr. Wee Youngcheul and Dr. Kim Dongyun for their continues warm support.

I know this dissertation is completed with the prayers and sacrificial love of my family: Eun-young, Kyoung-jin, Jong-jin, Hyun-jung, Soo-hyung and Danbi. Their support and selfless concern have sustained me through many difficult times. I want to express my deep appreciation and love to Kwon-il for his support, patience, humor, and most of all, always being there for me during the time I needed it most. To my parents Kim Tae-kew and Park Bong-sook, I even do not know how to express my deep gratitude enough, just saying I love you and I know this is not the end, but the start of a new chapter.

Table of Contents

	Page
List of Tables	vi
List of Figures	vii
Abstract	viii
1 Introduction	1
1.1 Introduction to Information Hiding	1
1.1.1 Applications	1
1.1.2 General Information Hiding Scheme	3
1.1.3 Requirements	4
1.2 Overview and Scope of Thesis	6
2 Literature Review	8
2.1 Steganography and Steganalysis	8
2.1.1 Steganography Techniques	9
2.1.2 Steganalysis Techniques	11
2.2 Fragile Watermarking Techniques	14
3 Background	16
3.1 Image Based Information Hiding System	16
3.2 Information Hiding Using JPEG Images	17
3.2.1 JPEG Image Format	18
3.2.2 Distortion in JPEG Images	21
4 Minimizing Distortion in Information Hiding using JPEG Images	24
4.1 Distortion	24
4.2 Rounding Biased Rule	25
4.3 Block-based Coding Techniques	26
4.3.1 Parity Coding	26
4.3.2 Matrix Coding	28
4.3.3 Modified Matrix Coding	29
4.4 Discussion on Shrinkage	31
5 Shrinkage-avoiding Embedding Algorithms	32

5.1	Parity Coding: PB	32
5.1.1	Embedding Algorithm	32
5.1.2	Error Analysis	34
5.1.3	Experimental Results	40
5.2	Modified Matrix Coding: MMx	40
5.2.1	Embedding Algorithm	41
5.2.2	Error Analysis	42
5.2.3	Experimental Results	46
6	Shrinkage-permitting Embedding Algorithms	51
6.1	Parity Coding: PB-s	52
6.1.1	Estimation of Shrinkage	52
6.1.2	Embedding Algorithm	54
6.1.3	Error Analysis	57
6.1.4	Experimental Results	61
6.2	Matrix Coding: MMx-s	66
6.2.1	Estimation of Shrinkage	66
6.2.2	Embedding Algorithm	70
6.2.3	Error Analysis	71
6.2.4	Experimental Results	72
7	Preserving Statistics while Minimizing Distortion	77
7.1	Graph Matching	77
7.1.1	Approximation Algorithm for Maximum Weighted Matching	78
7.1.2	Applying GREEDY-algorithm to Embed a Message	80
7.2	Embedding Algorithm	84
7.3	Error Analysis	86
7.4	Experimental Results	90
8	Algorithm Achieving Absolute Lower Bounds on Distortion	95
8.1	Parity Coding: PB-z	95
8.1.1	Embedding Algorithm	95
8.1.2	Error Analysis	97
8.1.3	Experimental Results	98
8.2	Matrix Coding: MMx-z	103
8.2.1	Embedding Algorithm	104
8.2.2	Error Analysis	105

8.2.3	Experimental Results	107
8.3	Impact of Modifying Zero-valued Coefficients	113
9	Summary	121
10	Conclusion and Research Plan	124
	Bibliography	126

List of Tables

Table		Page
8.1	Steganalysis test and detection rates (%).	114
8.2	Steganalysis test and detection rates (continued).	115
8.3	File sizes of stego images in bytes	119
8.4	The file size ratio between stego images and the corresponding cover images	120
8.5	PSNR comparisons	120

List of Figures

Figure	Page
3.1 An overview of information hiding system	16
3.2 JPEG encoder and decoder	19
3.3 An image is divided into non-overlapped 8×8 blocks	22
3.4 An example of JPEG Encoding	23
4.1 A graphical description for \bar{X}^+ and \bar{X}^-	25
5.1 Test images	35
5.2 Normalized rounding error histograms of non-zero AC JPEG coefficients . .	35
5.3 Probability distribution of rounding errors	36
5.4 Probability distribution of embedding errors	36
5.5 Probability distribution of additional error due to embedding	37
5.6 Distortion density for various values	38
5.7 Embedding error analysis	39
5.8 Error analysis	39
5.9 Average embedding error per message bit using PB	40
5.10 Error analysis using 100 images	41
5.11 Normalized rounding error histograms of the non-zero AC JPEG coefficients	46
5.12 Embedding error analysis for MMx	47
5.13 Comparison of embedding error for F5 and MMx	48
5.14 Average embedding error per message bit using MM2 and MM3	49
5.15 Accuracy of the error analysis on MM2	50
6.1 Histogram comparison of the cover image and the stego image	51
6.2 Histograms of rounding errors	62
6.3 Embedding error analysis for PB-s	63
6.4 Accuracy of the PB-s error analysis	64
6.5 Average embedding error per message bit using PB-s	65
6.6 Embedding distortion per embedding message bit for PB-s	65
6.7 Comparison in histogram modification	67

6.8	The expected block size for embedding a message bit using MM1-s	68
6.9	Embedding error analysis of MM1-s	73
6.10	Average embedding error per message bit using MM1-s	73
6.11	Embedding error per embedding message bit using MM1-s	75
6.12	Comparison of histogram modification	76
7.1	Constructing Graph	82
7.2	Matching	83
7.3	Two data structures (A and B) used in our implementation	87
7.4	Updating A and B after finding a matching	88
7.5	Error analysis of the PB-g method	90
7.6	Examples of test images	91
7.7	Matching rates for PB-g	92
7.8	Average embedding error per message bit using PB-g	93
7.9	The Chi-square distance for PB-g	94
7.10	The Euclidean distance for PB-g	94
8.1	Error analysis of PB-z	98
8.2	Test images used to illustrate PB-z	99
8.3	Comparison of the average distortions caused by PB-z, PB, and F5	100
8.4	Average embedding error per message bit using PB-z	100
8.5	Accuracy of the error analysis on PB-z	101
8.6	The Chi-square distance of PB-z	102
8.7	The Euclidean distance of PB-z	102
8.8	Error analysis of MM2-z	107
8.9	Comparison of the average errors caused by F5, MM2, and MM2-z	109
8.10	Average embedding error per message bit using MM2-z	109
8.11	Comparison of distortion caused by various methods	110
8.12	The median value of distortions using 100 images	111
8.13	Accuracy of the error analysis on MM2-z	111
8.14	Chi-square distance comparisons of MM2, MM2-z, and F5	112
8.15	Euclidean distance comparisons of MM2, MM2-z, and F5	112
8.16	Comparison detection rates	116
8.17	Entropy comparisons of PB stego, PB-z stego, and F5 stego	119
8.18	Entropy comparisons of MM2 stego, MM2-z stego, and F5 stego	119

Abstract

TOWARDS LOWER BOUNDS ON DISTORTION IN INFORMATION HIDING

Younhee Kim, PhD

George Mason University, 2008

Dissertation Co-Director: Dr. Zoran Duric

Dissertation Co-Director: Dr. Dana Richards

Information hiding refers to embedding additional digital data in cover objects—e.g. audio, image or video signals—by modifying the cover objects. Information hiding techniques have been used in a variety of application domains including copyright protection for digital media, content authentication, media forensics, and covert communications. The goal of information hiding techniques is to minimize the effect that the embedding process has on the cover object. Embedding processes introduce distortion to cover images and change the appearance as well as statistics of images. This is undesirable in many applications including fragile watermarking and steganography. In this dissertation, information hiding algorithms using JPEG images have been proposed with three goals: minimizing distortion due to embedding, preserving statistical properties during embedding, and predicting a distortion level for a given message length. Proposed methods use rounding errors created at the JPEG quantization step as side information, and the methods are based on block-based coding techniques known as parity coding and matrix coding. A mathematical analysis predicts the distortion introduced by each proposed algorithm as a function of message length and the rounding error distribution of the cover image. Minimization of distortion in proposed methods was experimentally validated.

Extensive tests using state-of-the-art steganalysis software show that the proposed information hiding methods compare favorably to other published methods for given embedding rates.

Chapter 1: Introduction

1.1 Introduction to Information Hiding

Information hiding refers to the process of embedding additional digital data in host signals, such as image, audio, or video files. The embedded data may vary—copyright information of art works, identification numbers of media files, or information that should be kept secret—depending on the purpose of the application. Digital images are the typical choice for the host signals because of their ubiquitous presence in our digital society. In this thesis, discussion is limited to image-based information hiding. Due to mature information and communication technologies and the rapid growth of the Internet, information hiding techniques have recently drawn the attention of the research community as well as industry.

1.1.1 Applications

Information hiding techniques have been used in many applications [1–5] including copyright protection, traitor tracing, content authentication and covert communication.

Copyright protection

As accessing the Internet and making digital copies of media have become widespread, copyright protection for digital media is more necessary than ever. Copyright information is embedded in media files before distribution, and when needed, it is extracted to prove the ownership for the media. The embedded data, usually called a *watermark*, needs to survive against various attacks. The attacks may occur unintentionally or intentionally. Unintentional attacks include general image processing such as smoothing, sharpening, or compression, while intentional attacks include all attempts to remove the embedded watermark. In order to be useful for copyright protection, the information hiding algorithm

should ensure that the embedded watermark is detectable after going through the various attacks, so that it could authenticate the copyright of the watermarked media.

Traitor tracing

A unique digital ID or signature, usually called a *fingerprint*, is embedded into each piece of distributed digital media. When the distributor detects the unauthorized use of their media, the original user of the unauthorized copies can be traced by the unique fingerprint. In this application, a possible collusion attack between users is a major challenge. For instance, multiple users could dilute or erase the original fingerprint by averaging multiple copies of the media to avoid detection.

Content authentication

Content authentication (tamper detection or proof of integrity) is another important application [6,7] of information hiding techniques. This application requires verifying that media files have not been altered since they were marked. This application can be useful for authentication in automatic video surveillance or driver's licenses [5,8]. The algorithm should be designed in a way that alterations on the media easily destroy the embedded mark, so that the destroyed mark would show that the content has been tampered with and furthermore that where the alteration is located. Fragile watermarking refers to the technique that the mark is designed to be sensitive to any alterations, and semi-fragile watermarking if the mark is sensitive only to malicious alterations (i.e., intentional attacks).

Covert communication

The goal of covert communication is to hide the existence of the hidden message in the communicated medium. While the medium is primary in other applications, the message itself is primary in covert communication. When information hiding technique is used in covert communication, it is called *steganography*, which is of Greek origin and means *covered*, or *hidden writing* [2]. On the other hand, the field of study that is interested in detecting

the steganographic communication is called *steganalysis*.

The main requirement for steganographic systems is undetectability (or security): The goal of steganographic methods is that the object containing the hidden message should not be detected as a suspicious object by a steganalysis. The undetectability is challenging because steganalysis methods use visual and statistical inspections to detect the presence of a hidden message.

Non-security-related applications

Information hiding has many applications other than security-related ones [4]. Data annotation is an example of non-security-related applications. Medical images are currently stored linked to text files written with patients' names and other private medical information in a large database. If the information is embedded in the medical image itself, the database management will become easy [1].

Information hiding technique can also be used for upgrading a legacy system without changing the existing system much. Many analog services are being upgraded to digital service, and information hiding techniques can be used in order to maintain backward compatibility, which bridges to the new digital system while keeping the current system unchanged [9].

1.1.2 General Information Hiding Scheme

The information hiding process has three general processes: generating a message to embed, embedding the message, and extracting the message.

Generating a message: The data to embed is called a *message*, and the message length is called the *payload*. The message could be any digital information including a unique ID for the distributed media, copyright information, image data, or audio data. The decision of what information to embed depends on the applications. For security reasons, the message is usually encrypted using a secret key that has been shared between a sender and a receiver. The encrypted message is assumed to be a binary random sequence, and many

statistical analysis in information hiding have been performed based on the assumption.

Embedding: The embedding process modifies the original medium in order to embed the message. The modification performs in the spatial domain (wherein the image is identified by pixels) or transformation domain (wherein the image is identified by frequency waves such as DCT, DFT, DWT etc.) The issue of selecting the location in the domain is also important. For example, embedding in low frequency coefficients has a different effect from embedding in the high frequency coefficients. It is more difficult for the human visual system to detect modifications of the high frequency coefficients. Therefore, embedding by modifying the high frequency coefficients can achieve more imperceptible embedding rather than ones in the low frequency coefficients. For security reasons, the location of embedding may be determined by a secret key. The original medium before a message has been embedded is termed a *cover object*, and the medium after the message is embedded in it is termed a *stego object*.

Extracting: The extracting process usually inverts the embedding process. When a stego object is sent to the extracting process, the elements of the stego object are arranged in the same order that the embedding process used. This can be accomplished by using a secret key. The usage of the extracted messages varies depending on the applications. For example, the extracted message which is verified as the exactly same to the original message can prove that the content of the stego object has not been modified since marked.

1.1.3 Requirements

Each application using information hiding techniques has different requirements depending on the purpose of the application. Generally, four requirements—robustness, payload, transparency and security—are common to most applications. Because there exist trade-offs between those requirements, it would be very challenging to design an algorithm that satisfies all of four requirements. The tradeoff between payload and security has been dealt with in this thesis.

Robustness: Robustness requires the embedded message to survive against various

types of attacks, such as filtering, resizing, rotation, and intentional attempts to remove the embedded message. To achieve robustness, one considers perceptually significant places for embedding, so that removing the message would result in significant perceptual distortion. Copyright protection application is an example that requires robustness.

Payload: This refers to the number of embedded message bits. The required payload ranges from one bit for a binary decision to a large number of bits for covert communication. Bit rate refers to the payload normalized by the size of media and is used to measure an embedding rate.

Imperceptibility: Many applications of information hiding require the embedded messages to be imperceptible. The transparency requirement conflicts with the robustness and the payload requirements. To achieve the transparency, perceptually insignificant places should be considered for embedding, which is in contrast to the robust embedding technique. It is relatively easy to achieve transparency for small payload applications, but it is challenging for the large payloads.

Security: What distinguishes steganography from other forms of information hiding is the focus on merely detecting presence of a hidden message. In steganography, a third-party is assumed to know the distributions of cover objects and stego objects, and the embedding algorithms (not secret key). Stego objects should look just like innocent cover objects even though they contain hidden messages. In a secure steganographic system, an adversary cannot distinguish whether a sender is sending a legitimate cover object or a stego object. Cachin formalized security of steganography in [10,11]. The security of a steganographic system with a cover object (C) and an object generated by an embedding algorithm (S) is quantified using relative entropy between P_C and P_S . The system is called ϵ -secure against passive adversaries if

$$D(P_C||P_S) = \sum_{x \in C} P_C(x) \log \frac{P_C(x)}{P_S(x)} \leq \epsilon.$$

If $\epsilon = 0$, the stego system is called *perfectly secure* [10].

1.2 Overview and Scope of Thesis

This thesis will present information hiding schemes using JPEG images¹ with three goals: minimizing distortion due to embedding, preserving statistical properties during embedding, and predicting the distortion level with a given message length. Since stego images are created by modifying the cover images, some level of distortion to the cover images is unavoidable; however, the minimized distortion is required in many applications. Especially in high-image-quality-required applications, including fine art works, photographs for court evidence, or surveillance videos that may have to be processed again after stored, minimal distortion to the images should be one of the primary requirements. Another motivation of aiming for minimal distortion is that modification in the image statistics can be minimized by the method of minimizing distortion. Relationship between the distortion and statistical change in information hiding is firstly explored by this thesis. Minimizing distortion due to embedding is the first goal of this thesis.

In addition to producing distortion, embedding messages alters statistical properties of the cover image. When one algorithm focuses only on minimizing distortion, it may disregard the effect of the statistical alteration, which will make the algorithm vulnerable to the steganalysis attack: The altered statistical properties can be used for the steganalysis methods, which are interested in detecting the use of steganographic methods. Preserving statistical properties in images during embedding is the second goal of this thesis.

Another goal regarding distortion in this thesis is predicting the distortion level with a given message length. Distortion varies depending on the payload, the length of message. A high payload increases the level of distortion. There has been very little work on how to choose the payload in terms of the tradeoff between payload and distortion. Instead, trial-and-error methods have been used to determine the payload with a given distortion level. In this thesis, the distortion is predicted as a function of message length and one of the cover data statistics. The mathematical analysis will help to determine the payload automatically. To the best of our knowledge, this is the first mathematical analysis of

¹This can be expanded to other formats.

predicting distortion in the field of information hiding.

In this thesis, information hiding algorithms using JPEG images have been proposed. Chapter 3 will overview of the JPEG encoder including the process of producing the rounding errors, which will provide a background for understanding this thesis. After providing the background about JPEG images, our fundamental minimization scheme, which uses rounding errors and block-based coding, is introduced in Chapter 4. The detailed implementation of the minimization scheme are described as four embedding algorithms from Chapter 5 to 8 depending how to deal with the issues of JPEG information hiding. The algorithms use two block-based coding techniques called parity coding and matrix coding. The expected distortion with each algorithm is mathematically analyzed in the chapters.

Achieving the three goals is validated by measuring distortion. The distortion is measured as a distance between cover data and stego data. When each embedding algorithm is proposed, the distortion for various embedding rates is presented and the various results are demonstrated in each chapter. We have continuously improved our embedding algorithm to decrease distortion and the results show how low a distortion we have reached through the improvement. The accuracy in the estimation of distortion due to embedding is shown by plotting the theoretical one and experimental one in the same graph. To show how closely the embedding algorithm preserve the original statistics, differences between the histograms before and after embedding are measured. Lastly, we have tested our embedding algorithms with the well-known steganalysis tests and showed the results.

Chapter 2: Literature Review

Information hiding is a relatively new research field. Detailed survey of early algorithms and softwares for information hiding techniques can be found in the papers by Petitcolas [2] and Swanson et al. [12]. A comprehensive overview of many applications is provided in the books by Cox, Miller, and Bloom[13], Johnson, Duric, and Jajodia [14], Katzenbeisser and Petitcolas [15], and Wayner[16]. An overview focusing on the principles and mathematical methods on information hiding is provided by Moulin in [5].

The first academic conference related to information hiding is International Workshop on Information Hiding (IH), which has been yearly held since 1996. The IH conference proceedings are considered the primary source for the recent information hiding techniques. Another related conference is IS&T/SPIE Electronic Imaging Security, Forensics, Steganography, and Watermarking of Multimedia Contents (SPIE). In addition, two recent peer reviewed journals are IEEE Transactions on Information Forensics and Security and International Journal of Information Security.

Since the proposed methods in this thesis are applicable to steganography and fragile watermarking, the review will be limited to those two subfields rather than covering all information hiding techniques that have been developed for the last decade.

2.1 Steganography and Steganalysis

People have been using steganography to hide information since Greek times. However, digital steganography is a relatively new research field [15]. Since being undetectable is one of the essential requirements for steganographic applications, steganography and steganalysis techniques are evolving in competition with each other.

2.1.1 Steganography Techniques

The steganography techniques for images can be broadly grouped into two categories: spatial domain methods [17–20] and frequency domain methods [21–25].

In spatial domain steganographic techniques, pixel values of images are used to embed messages. To modify the pixel values, the early steganographic methods replace the least significant bits (LSB) of pixel values with the message bits, which is known as a *LSB replacement technique* [26]. The LSBs of the pixel values are considered redundant parts of images, especially in terms of visual perception. In the LSB replacement technique, it was assumed that the LSBs were completely random so their modification by being replaced with the random message bits would not be detected; however, it was found in [27] that the LSBs are very correlated with the image. It was also found, even though the LSB modification does not change the visual quality of the image much, that the modification changes the statistics of the LSBs significantly, which can be detectable.

Another technique used in the spatial domain methods is called ± 1 *embedding* [18], which modifies the pixel value by incrementing or decrementing, so the LSB of the pixel value become matched to the message bit [17]. More sophisticated method using ± 1 embedding, called *Stochastic Modulation*, was proposed in [18], where noise was added to the pixels with an arbitrary probabilistic distribution to provide the better security.

Second category of image steganographic methods is embedding techniques using transform domain data. The most common transformation is the Discrete Cosine Transform (DCT), which is used in a JPEG encoder. The proposed methods in this thesis use DCT coefficients to embed a message, so they belong to this category. Since the steganalysis methods was successful to detect the spatial domain embedding methods, many transform domain embedding algorithms have proposed methods to preserve statistical properties of cover images. The detailed discussion on how the steganographic methods were proposed to avoid the statistical attacks of various steganalysis methods will be presented in the next section 2.1.2.

Current embedding techniques for JPEG images can be grouped into three groups [28]:

(a) embedding by direct modification of DCT coefficients [21–23], (b) minimal distortion embedding by utilization of side information (e.g., rounding errors occurred at the quantization stage.) [25, 29], (c) converting to JPEG format after robustly embedding a message in another domain [30].

Two different methods of modifying directly DCT coefficients are discussed first: the *F5* [21] and *Outguess* [22] algorithms. Both algorithms were proposed to preserve the histogram of DCT coefficients during the embedding process. *F5* embeds a message by modifying DCT coefficients in a way that avoids becoming the frequency of adjacent coefficients similar, which is noticeable by a simple histogram attack. *F5* decrements the absolute value of the coefficient values instead of replacing the LSB by the message bit. *Outguess* embeds a message using up to a half of all usable coefficients, and correct the discrepancy, which is created by the embedding process, using the remaining coefficients. In results, *Outguess* can preserve the DCT coefficient histogram, but it has a drawback of decreasing the possible maximal length of embedding by half.

Minimal distortion embedding methods [25, 29] utilize side information such as rounding errors that occur during JPEG quantization stage and embed a message while minimizing distortion. Exploiting rounding errors as the side information for minimal distortion was firstly proposed in [29] called *Perturbed Quantization*. *Perturbed Quantization* (PQ) chose those coefficients whose values before rounding are in the middle of the interval as changeable coefficients for low distortion of stego objects. Wet paper code [31] was used to embed a message. The work proposed in this thesis also utilizes rounding errors of DCT coefficients; however, in our methods, (a) the combined distortion due to rounding and embedding has been minimized by considering all coefficients; (b) there are no constraints on choosing usable coefficients. All available coefficients are considered as useable; (c) the solution is always guaranteed; (d) double compression is not necessary.

Another group of JPEG steganographic methods [30, 32] embed messages robustly in images other than the JPEG format and convert the embedded images to the JPEG format.

Besides the methods discussed so far (spatial and transform domains techniques), model

based techniques are important in steganography. *Model-based-steganography* [23, 33] is a hiding method that preserves distributions of individual JPEG coefficients during the embedding process. On the sender’s side the method estimates the distributions of the AC coefficients in JPEG images from the distribution of the most significant bits (MSB) of the coefficients. The estimated distribution is used by an *entropy decoder* to encode the compressed and encrypted message in the LSBs of the coefficients. On the receiver’s side the same distribution is estimated from the MSBs of the coefficients and the message is extracted from the LSBs of the coefficients by an *entropy encoder*.

Another method to preserve the statistics of cover objects is the *graph-theoretical approach*. The graph-theoretical approach technique embeds a message by exchanging rather than overwriting cover data. The method proposed by Hetzl and Mutzel [34] constructs a graph with cover data and finds matched edges. The matched edge is defined as two elements of cover data that can be interchanged to embed a message. This graph-theoretical approach can be applied to various formats of cover data including BMP images, JPEG images, and WAV audio files. Because of the non-uniform distribution of the DCT coefficients, about 3% of coefficients are left unmatched for the block size of 3 in the experiments conducted [34]. However, still the major coefficients are matched, so the histogram is preserved for not being detected by histogram-based attacks [27, 35].

2.1.2 Steganalysis Techniques

The goal of steganalysis methods is to detect the presence of a hidden message, although the message itself does not need to be decoded. The techniques of image steganalysis are categorized into two: algorithm-specific steganalysis and universal steganalysis.

The first approach of steganalysis methods were designed with respect to the specific embedding method, and they can be grouped depending on the image format that the embedding algorithm uses, such as raw images, palette-based images, or JPEG images.

To embed a message in raw images, the simple LSB replacement technique was used in the pixel domain. There are two steganalysis techniques to detect the LSB replacement

embedding: *visual attack* [27] and *chi-square attack* [27].

The visual attack detected changes in the LSB plane of images when the LSBs were replaced by a message. Since the LSBs are correlated with the image, embedding of the random message bits changes the correlation between the LSB plane and the image. For example, after embedding with full capacity of the image (i.e., every LSB is replaced with the message bit), the LSB plane of the image looks like random noise.

The chi-square attack detected changes of image statistics by LSB replacement. The LSB replacement results in producing pairs of adjacent coefficients that their values differ by only 1 (e.g., (2,3), (4,5)). The frequencies of those adjacent coefficients have changed depending on the frequencies of embedding message bits (0 or 1). The chi-square attack [27] found that the frequencies of those adjacent coefficients become similar as the amount of the embedded messages becomes large. Since the chi-square attack was proposed, the LSB replacement techniques have been considered as easily detectable by examining histograms [36–39] or structural analysis [40]. LSB steganalysis approaches that also provided message length estimation were proposed for gray scale images [41,42] and color images [43].

The Hide algorithm [17] increments or decrements the sample value in order to match its LSB to the message bit, which can not be detected by simple LSB steganalysis. This ± 1 embedding technique used in the Hide algorithm was detected by looking at the neighborhood colors for each color value as proposed in [35,44].

The next group of steganalysis is designed with respect to embedding methods for palette-based images. Many steganography methods [20,45] for palette-based images (e.g., GIF images) have been proposed. The early proposed steganalysis method [46] for palette-based images used visual inspection or simple statistics for detection. Later, EzStego-specific steganalysis methods [27,47] explored color pairs, obtained after sorting the palette, that have changed their patterns after the embedding.

The last group of steganalysis is designed with respect to embedding methods for JPEG images. Due to the emergence of many steganographic methods [21,22,48,49] employing JPEG

images, many JPEG steganalysis methods have been proposed. Two steganalysis methods [50,51] have proposed to detect the F5 and Outguess algorithms using DCT coefficients histogram and *blockiness*. F5 employs matrix encoding so as to decrease the number of changes, and it increments (or decrements) the sample value to avoid chi-square attack; however, F5 still alters the histogram of DCT coefficients in a noticeable way. The steganalysis technique in [50] proposed a method of estimating the cover image by cropping the stego JPEG image by 4 columns and then encoding the cropped image using the same quantization table. The histogram of the estimated cover image was compared to the one of the stego image, and then the length of message was estimated.

The steganalysis technique to detect Outguess was proposed in [51]. Since Outguess can preserve the original histogram of DCT coefficients, the technique of estimating the original histogram is not effective. Instead, the steganalysis on Outguess used *blockiness*, which refers the spatial discontinuities along the 8×8 DCT blocks. When the length of the embedding message increases, the *blockiness* also increases.

While the first group of steganalysis methods was designed for a specific embedding method, the second type of steganalysis methods are designed to detect any embedding algorithms, called *Universal Steganalysis* or *Blind Steganalysis Technique* [26]. Universal steganalysis basically designs a classifier that can differentiate the cover images and stego images. It is very important to find the features that can capture the statistical changes introduced in the images after embedding.

A universal steganalysis method proposed in [52,53] decomposed an image using quadrature mirror filters and then high order statistics were calculated from the high frequency bands to be used as features for steganalysis. In [54], the authors used the low-low wavelet subbands to calculate the statistical moments of characteristic functions for their steganalysis.

Another steganalysis method [24] estimated a cover image of the JPEG format and the statistics of the estimated cover image was used as the features to detect data embedded in JPEG images.

The methods in [55,56] detected spread spectrum data hiding method using the inter-pixel dependencies and a Markov chain model. The Markov features were also applied in [57,58] attacking JPEG steganography methods.

2.2 Fragile Watermarking Techniques

A fragile watermark refers a mark that is easily altered or destroyed when the cover image is modified. Fragile watermarking is not suited for enforcing copyright ownership of digital images. Fragile watermarking may be interesting to parties who want to verify that the image has not been edited or altered since it was marked. The sensitivity of fragile marks to modification leads to their use in image authentication. The information to be embedded is a random message, image information (image id, camera id and time stamps) or image summary.

The early fragile technique for authentication used the technique of inserting the mark in the least significant bit (LSB) plane of image pixels [59,60] and the added watermark is a pseudo random sequence, which is not related to the content of the image.

P. Wong describes another fragile marking technique in [61] that calculates a digest of the image using a hash function and embeds it in the cover object. The image ID, image size and user key are hashed and embedded by modifying the LSBs of pixels of the image.

The hybrid method in color images was proposed by Yeung and Mintzer [62]. A random binary sequence was excluded-ORed with all three color channels' hash values, which are related to the image pixel values. In order to spread the modification out, the error diffusion technique was used.

The improved version of the Yeung-Mintzer algorithm [62] was proposed in [63]. It used the neighboring pixels information in generating the authentication code. This algorithm is less localized but more sensitive in detecting alteration compared to the Yeung-Mintzer algorithm.

Fragile watermarking systems using frequency domain data have an advantage that the mark can be embedded in a compressed domain. Wu and Liu [64] described a technique

of changing the quantized DCT coefficients. Kundur and Hatzinakos [65] and Xie and Arce [66] describe techniques based on the wavelet transform. Kundur modified the Haar wavelet transformation coefficients to embed a mark while Xie modified the SPIHT algorithm.

Chapter 3: Background

As the thesis focuses on designing information hiding algorithms using JPEG image, overviews of image based information hiding system and JPEG images are described in this chapter. Information hiding refers to hide information in host signals such as images, audios, and videos. Generally, information hiding methods modify a cover object to embed a message and creates a stego object. A *cover object* refers to the original medium before information has been embedded; a *stego object* refers to the medium after information is embedded in it. The data to be embedded is called a *message* and the message length is called a *payload*.

3.1 Image Based Information Hiding System

Figure 3.1 shows an overview of the information hiding system. As the thesis designed image based information hiding methods, X and \hat{X} in Figure 3.1 are images. A cover image

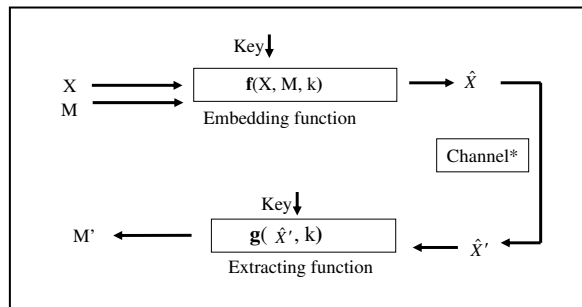


Figure 3.1: An overview of information hiding system.

(X) and a message (M) are sent to the embedding function with a secret key. Typically, the message is assumed to be encrypted for security reasons, and the secret key and the embedding function are assumed to be known to the receiver. An information hiding algorithm

has a pair of functions \mathbf{f} and \mathbf{g} such that

$$\hat{X} = \mathbf{f}(X, M), \quad M = \mathbf{g}(\hat{X}).$$

The typical embedding function finds the insignificant part of the cover image for the human visual system and modifies them to embed a message. The output of the embedding function is the stego image (\hat{X}). It is not easy to tell the stego image from the cover image by the human visual system. The stego image is sent to the receiver through a channel. The channel may modify the stego image by adding noises (e.g., compression, filtering, or malicious modification.) The received image is sent to the extracting function with the key. The extracting function retrieves a message from the stego image. The message may or may not be the same to the message that the sender has sent. The usage of the extracted message varies depending on the applications. For example, the message can be used for content authentication to verify that the content of the image has not been altered: If the extracted message is the same to the original, it verifies that the transferred image has not been altered. Otherwise, the image is reviewed for possible modifications.

3.2 Information Hiding Using JPEG Images

Since the embedding schemes proposed in this thesis use the properties of JPEG encoding, an overview of the JPEG format would help to explain the scheme. JPEG image formatting removes some image details to obtain considerable saving of storage space without much loss of image quality. The savings are based on the fact that humans are less sensitive to changes of the high frequency components than of the low frequency components. Information hiding techniques using JPEG files operate in the frequency domain. One of the benefits of embedding in the frequency domain is that of not being affected by visual attacks [27]. When some stego images are examined by the visual attack technique, embedded messages can be seen on the low-bit plane of the image; this usually happens when messages are

embedded in BMP images using early information hiding techniques.

3.2.1 JPEG Image Format

Suppose a sender has an uncompressed image and wants to embed a message in a JPEG format image. The sender can access every procedure of the JPEG encoder and see all the intermediate data. Figure 3.2 illustrates the JPEG encoder and decoder. At the encoder side each channel is divided into 8×8 blocks and transformed using the two-dimensional discrete cosine transform (DCT). DCT converts spatial domain data ($I(i, j)$) into frequency domain data ($X(u, v)$). The mathematical specifics on DCT are in [67]. Let $I(i, j)$, $i, j = 0, \dots, N - 1$ be an $N \times N$ image block in any of the color channels and let $X(u, v)$, $u, v = 0, \dots, N - 1$ be its value after DCT. The relationship between $I(i, j)$ and $X(u, v)$ is given by

$$X(u, v) = \frac{2}{N} C(u) C(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} I(i, j) \cos\left(\frac{\pi u(2i+1)}{2N}\right) \cos\left(\frac{\pi v(2j+1)}{2N}\right)$$

$$I(i, j) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u) C(v) X(u, v) \cos\left(\frac{\pi u(2i+1)}{2N}\right) \cos\left(\frac{\pi v(2j+1)}{2N}\right),$$

where $C(u) = 1/\sqrt{2}$ for $u = 0$ and $C(u) = 1$ otherwise. The coefficient $X(0, 0)$ is called DC and all others are called AC. The JPEG encoder uses the quantization operation and rounding operation given by:

$$\bar{X}(u, v) = \frac{X(u, v)}{Q(u, v)},$$

$$\tilde{X}(u, v) = \text{Round}(\bar{X}(u, v)) \tag{3.1}$$

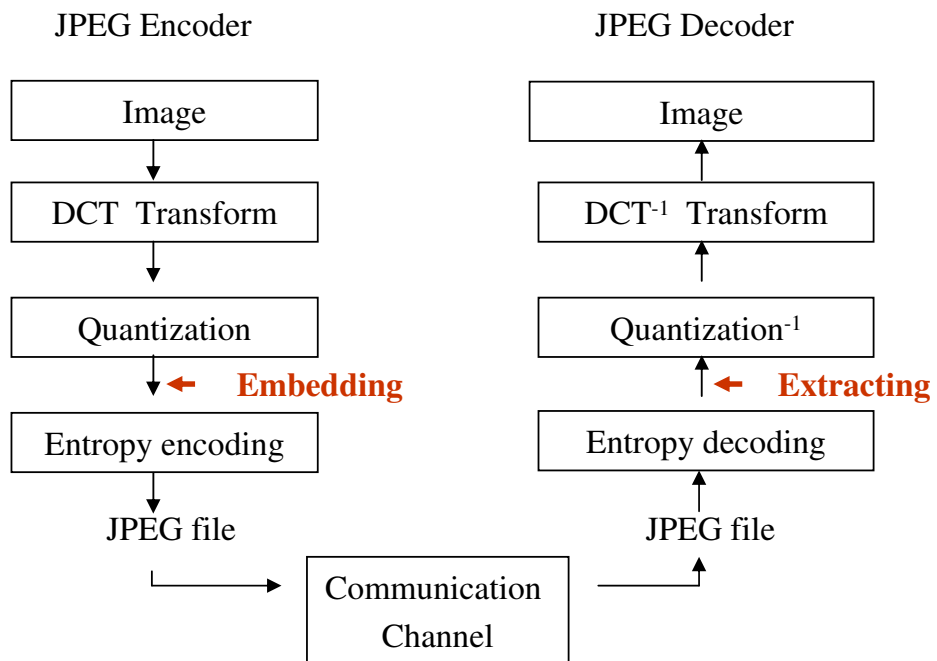


Figure 3.2: JPEG encoder and decoder. The procedure related to information hiding is involved after the quantization procedure and before the entropy coding procedure.

to obtain integer-valued coefficients $\tilde{X}(u, v)$. The process produces quantization errors and rounding errors:

$$\begin{aligned}\delta(u, v) &= \tilde{X}(u, v)Q(u, v) - X(u, v), \\ R(u, v) &= \bar{X}(u, v) - \tilde{X}(u, v).\end{aligned}\tag{3.2}$$

A default quantization table [67] is given by

$$Q(u, v) = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}.$$

Figure 3.3 shows that one channel (Y) of an image is divided into 8×8 -sized blocks, and Figure 3.4 illustrates how the image block is encoded as DCT coefficients. The first matrix in Figure 3.4 is an example of the 8×8 image block; the second matrix is a result of the DCT, the third matrix is a result of the quantization, and the fourth matrix shows the rounded DCT coefficients, which are the information saved in encoded JPEG files. As shown in the fourth matrix in Figure 3.4, usually many coefficients for higher values of $u + v$ become zero, and only a fraction of all coefficients remains non-zero. The coefficients of \tilde{X} are ordered into a linear array by placing higher frequency coefficients (higher values of $u + v$) at the end of the array. Those high frequency coefficients are most likely to be zeroes. Entropy coding is applied to all non-zero coefficients from all blocks in the image; the zero-valued coefficients are encoded separately using run-length encoding. A header that contains the

information about image type, dimensions, compression parameters, and the quantization table is attached to the entropy encoded coefficients. All is packed as a sequence of bits to form a JPEG encoded image file.

On the decoder side, the integer valued coefficients are restored by entropy decoding. The inverse DCT of the inverse-quantized data is performed to obtain the decoded image. The decoded image is not the same with the original image because the rounding operation in the quantization procedure is a lossy operation.

3.2.2 Distortion in JPEG Images

Distortion occurs when an image is encoded as a JPEG file. Figure 3.4 shows the original image block on the top and the decoded image block in the bottom. Because the rounding operation at the quantization stage is lossy, the decoded image block is different from the original image block. In other words, because of δ in Eq. (3.2), the decoded image is not the same as the image before encoded. As shown in Figure 3.2, information hiding techniques for JPEG images modify DCT coefficients, which already contain errors caused from the rounding operation. The process of embedding a message will introduce additional errors to the images. Therefore, the stego image created by the sender in the JPEG format will have distortion, which comes from the rounding operation as well as from the embedding procedure.

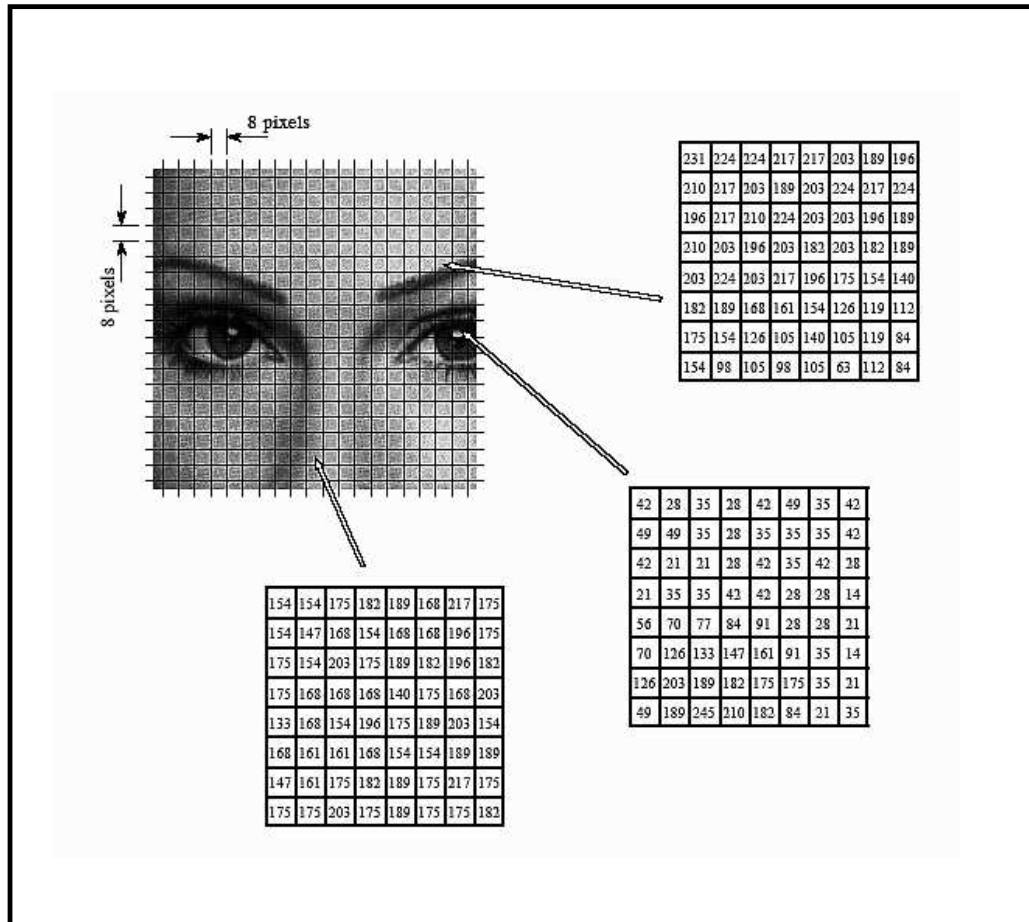


Figure 3.3: An image is divided into non-overlapped 8×8 blocks before DCT. The picture is from [68].

Image pixels

41	29	24	52	91	48	35	51
44	36	34	34	42	45	45	66
44	44	43	44	47	51	42	42
46	45	42	45	38	39	34	41
52	45	30	35	33	32	34	51
76	65	40	33	32	33	43	57
100	123	72	32	33	32	31	38
65	68	58	31	32	33	33	35

After DCT

363.8750	39.8055	38.3418	6.2621	11.8750	-23.9883	-2.4871	-0.8256
-15.6690	-72.7432	-46.3362	12.2684	31.4444	2.8352	3.8513	3.5874
18.5220	10.2478	-7.6543	6.6282	1.9907	-13.7226	-2.5093	11.0946
8.3644	-3.5442	9.8809	7.9056	21.4524	-8.5840	-8.6598	2.3671
-21.8750	-17.5427	-27.8757	3.2590	15.6250	2.4355	0.3167	3.9388
23.3971	32.6065	-10.4669	4.5958	-5.1021	-10.4770	-7.8237	0.5416
-5.9132	-18.2130	-16.7593	8.4949	9.0523	7.1093	0.6543	3.8547
5.4917	11.2672	-5.3799	4.0790	-3.0129	-7.2052	-3.8622	-0.1854

After quantization

22.7422	3.6187	3.8342	0.3914	0.4948	-0.5997	-0.0488	-0.0135
-1.3058	-6.0619	-3.0891	0.6457	1.2094	0.0489	0.0642	0.0652
1.3230	0.7883	-0.4784	0.2762	0.0498	-0.2407	-0.0364	0.1981
0.5975	-0.2085	0.4491	0.2726	0.4206	-0.0987	-0.1082	0.0382
-1.2153	-0.7974	-0.7534	0.0582	0.2298	0.0223	0.0031	0.0512
0.9749	0.9316	-0.1903	0.0718	-0.0630	-0.1007	-0.0692	0.0059
-0.1207	-0.2846	-0.2149	0.0976	0.0879	0.0588	0.0055	0.0382
0.0763	0.1225	-0.0566	0.0416	-0.0269	-0.0721	-0.0375	-0.0019

After rounding

23	4	4	0	0	-1	0	0
-1	-6	-3	1	1	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
-1	-1	-1	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Decoded image pixels

37	41	39	53	72	57	39	49
39	38	26	33	53	49	43	62
54	53	36	33	47	42	37	54
44	52	43	40	51	43	31	41
34	46	38	30	42	42	37	49
78	83	57	28	30	33	37	54
102	106	73	34	27	28	30	44
61	77	58	29	28	30	28	38

Figure 3.4: An image 8×8 block is transformed into JPEG coefficients using DCT. The output of the DCT transformation is quantized and rounded to integers. The integer coefficients are stored in JPEG format file. The decoded image pixel values from the stored JPEG coefficients are different from the original image pixel values.

Chapter 4: Minimizing Distortion in Information Hiding using JPEG Images

Designing information hiding algorithms that minimize distortion is a goal of this thesis. The fundamental scheme that we designed to minimize distortion for JPEG images is described. This chapter describes the formal definition of distortion and rounding errors used in the thesis, and then describes how the rounding errors are utilized for achieving minimal distortion. Block-based coding techniques are also utilized for minimal distortion. Parity coding and matrix coding techniques are two coding techniques used in the proposed methods.

4.1 Distortion

Let \bar{X} and \tilde{X} be the vectors of cover DCT coefficients before and after the rounding operation and \hat{X} be the vector of stego DCT coefficients after embedding. Let R and \hat{R} be:

$$R = \bar{X} - \tilde{X} = (r_1, r_2, \dots, r_l), \quad (4.1)$$

$$\hat{R} = \bar{X} - \hat{X} = (\hat{r}_1, \hat{r}_2, \dots, \hat{r}_l), \quad (4.2)$$

where $-0.5 \leq r_i < 0.5$. We assume that if \bar{X} is given, the corresponding \tilde{X} , R , and \hat{R} can be calculated. Let *distortion* be $D = \|\hat{R}\|_1$ and *rounding error* be R . Note that if $\hat{X} = \tilde{X}$ then $\hat{R} = R$. (i.e., if embedding requires no modification, the total error is equivalent to the rounding error.) Since embedding any message almost always requires changing bits,

the best result that can be obtained is

$$D \geq \|R\|_1.$$

Due to the JPEG encoding properties, embedding a message in a JPEG image yields distortion from two sources: rounding errors and modification due to embedding. The central idea of utilizing rounding error comes from the following question: Can total distortion be minimal by taking into account the rounding errors?

4.2 Rounding Biased Rule

We define *Rounding biased rule*, which is applied when JPEG coefficients are required modification for embedding. By applying the rounding biased rule, the coefficient is changed in a way that its value after embedding (\hat{x}) is close to the original (\bar{x}).

To introduce the rule, the unrounded coefficients are divided into \bar{X}^+ and \bar{X}^- depending on their corresponding rounding errors:

$$\begin{aligned}\bar{X}^+ &= \{\bar{x}_i | \bar{x}_i \in \bar{X} \text{ and } 0 \leq r_i < 0.5\}, \\ \bar{X}^- &= \{\bar{x}_i | \bar{x}_i \in \bar{X} \text{ and } -0.5 \leq r_i < 0\}.\end{aligned}\tag{4.3}$$

Figure 4.1 is a graphical description for \bar{X}^+ and \bar{X}^- . Now, the rule is defined based on the

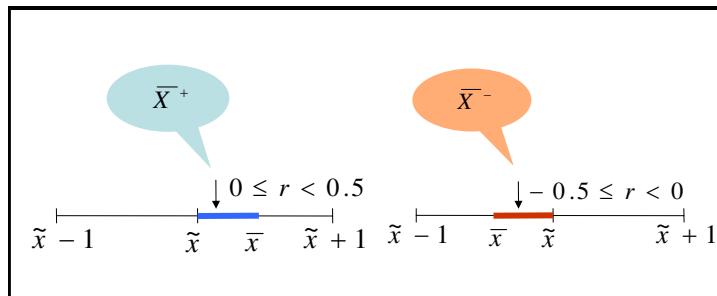


Figure 4.1: A graphical description for \bar{X}^+ and \bar{X}^- .

\bar{X}^+ and \bar{X}^- . If a coefficient \bar{x}_i needs to be changed to \hat{x}_i in order to embed a message, the modification follows the rounding-biased rule given by:

$$\hat{x}_i = \begin{cases} \text{Round}(\bar{x}_i + 0.5), & \text{if } \bar{x}_i \in \bar{X}^+, \\ \text{Round}(\bar{x}_i - 0.5), & \text{if } \bar{x}_i \in \bar{X}^-. \end{cases} \quad (4.4)$$

Since the rounding errors are taken into consideration, a pair of functions \mathbf{f} and \mathbf{g} in this thesis is

$$\hat{X} = \mathbf{f}(\bar{X}, R, M), \quad M = \mathbf{g}(\hat{X}), \quad (4.5)$$

and we seek \mathbf{f} and \mathbf{g} such that $\|\bar{X} - \hat{X}\|_1$ is minimized. With the rounding-biased rule, the distance between \bar{x}_i and \hat{x}_i is always limited to $1 - |r(\bar{x}_i)|$, where $r(\bar{x}_i)$ is the corresponding rounding error when $\text{Round}(\bar{x}_i)$ occurs. And the added distortion due to embedding (called embedding error and denoted by ε in this thesis) is always limited to $1 - 2|r(\bar{x}_i)|$.

4.3 Block-based Coding Techniques

Distortion due to embedding can be minimized using block-based coding techniques. Block-based coding techniques use several elements of cover data in order to embed a message bit and it allows us to choose the minimum one among n (the block size) choices, where minimization is achieved in a greedy way. We have applied two different block-based coding techniques (parity coding and matrix coding) to our methods.

4.3.1 Parity Coding

Exploiting the parity coding technique for the least obtrusive embedding was suggested by Anderson [69]. He suggested that parity coding can reduce the effect that the embedding process has on the image statistics below any arbitrary threshold. Embedding methods based on parity coding take several cover object's elements for a single message bit and embed the bit by matching the parity of the cover data to the message bit.

The parity coding technique is modified in our minimizing distortion scheme. Rather choosing an arbitrary cover coefficient, our methods examine the expected distortion due to embedding based on rounding-biased rule and select the one that causes the minimum distortion for modification. The following describes the detailed steps of embedding a message bit in a n -sized block using parity coding in our rounding-biased rule based embedding scheme.

The embedding process divides \tilde{X} (after rounding) into blocks of length n . To embed a message bit m_i , the i^{th} block $\tilde{X}_i = \{\tilde{x}_{n(i-1)+1}, \dots, \tilde{x}_{ni}\}$ is considered. \tilde{X}_i is the i^{th} block of \tilde{X} , and so forth. If the parity of the LSB(\tilde{X}_i) is equal to m_i , no change is necessary, $\hat{X}_i = \tilde{X}_i$. On the other hand, if the result of parity is different from m_i , the j^{th} coefficient of the block, $\tilde{X}_{i,j} = \tilde{x}_{n(i-1)+j}$ ($1 \leq j \leq n$) needs to be changed so that the LSB's of the block have their parity equal to m_i . Now, the question is which j should be chosen to change. We are interested in minimizing distortion between \bar{X} (before rounding) and \hat{X} (after embedding), therefore, $\hat{r}_j = 1 - |r(\bar{X}_{i,j})|$ for each $\tilde{X}_{i,j}$ is examined, and the one that has the least distortion will be chosen to be changed. Algorithm 1 explains embedding m_i in \tilde{X}_i and producing \hat{X}_i :

Algorithm 1 Proposed embedding algorithm to minimize distortion in a single block using parity coding.

inputs: $\tilde{X}_i, m_i; |\tilde{X}_i| = n;$
 $\tilde{X}_i \leftarrow \text{Round}(\tilde{X}_i);$
if $\text{Parity}(\text{LSB}(\tilde{X}_i)) = m_i$ **then**
 $\hat{X}_i \leftarrow \tilde{X}_i;$
else
 find j such that $\hat{r}_j = \min_{1 \leq k \leq n} \{1 - |r(\tilde{x}_{n(i-1)+k})|\};$
 for $1 \leq k \leq n$ **do**
 if $k = j$ **then**
 $\hat{x}_{n(i-1)+k}$ by the rounding-biased rule;
 else
 $\hat{x}_{n(i-1)+k} = \tilde{x}_{n(i-1)+k};$
 end if
 end for
end if

4.3.2 Matrix Coding

Matrix coding was proposed by Crandall [70] to improve embedding efficiency by decreasing the number of required bit changes. Westfeld [21] proposed F5, a well known steganographic algorithm in which matrix coding was implemented. In F5, cover data are the LSBs of quantized DCT coefficients (\tilde{X}) after rounding. The notation $(1, n, k)$, where $n = 2^k - 1$, denotes embedding k message bits into an n sized block by changing only one bit of it. Let $\tilde{X}_{i,j}$ be the j^{th} coefficient in the i^{th} block of length n and M_i be the i^{th} message block of length k . The advantage of matrix coding is embedding several (k) bits by changing only a single bit. A function \mathbf{b} needs to be defined to find the bit position to change in matrix coding:

$$\mathbf{b}(\tilde{X}_i) = \bigoplus_{j=1}^n (\text{LSB}(\tilde{X}_{i,j})) \cdot j, \quad (4.6)$$

where \oplus is a bitwise exclusive OR operation. The bit position α that needs to be changed is calculated by:

$$\alpha = M \oplus \mathbf{b}(\tilde{X}_i). \quad (4.7)$$

If $\alpha \neq 0$, the LSB of $\tilde{X}_{i,\alpha}$ should be flipped, 1 to 0 or 0 to 1, to embed M_i . The modified block is then given by

$$\hat{X}_i = \begin{cases} \tilde{X}_i, & \text{if } \alpha = 0, \\ \tilde{X}_{i,1}, \dots, \hat{X}_{i,\alpha}, \dots, \tilde{X}_{i,n} & \text{if } \alpha \neq 0, \end{cases} \quad (4.8)$$

where $\hat{X}_{i,\alpha}$ is produced by decrementing the absolute value of $\tilde{X}_{i,\alpha}$ in the F5 implementation.

On the decoder's side, k message bits are extracted from \hat{X}_i by calculating:

$$M_i = \mathbf{b}(\hat{X}_i). \quad (4.9)$$

4.3.3 Modified Matrix Coding

We modified matrix coding in order to be used in our minimizing distortion scheme. The original matrix coding provides only one single choice for embedding k message bits. With the single choice, our minimizing scheme based on rounding-biased rule (Section 4.3.1) can not be utilized.

The modified matrix coding uses (t, n, k) codes, $t \geq 1$, which embeds k messages bits by changing at most t bits of n cover data. For example of $t = 2$, we find pairs of the index number (β, γ) such that $\beta \oplus \gamma = \alpha$. As a result, flipping the bits in the positions of β and γ and flipping the bit in the positions of α produce the same extracted message using Eq. (4.9). Note that for any α , there are $\frac{n-1}{2}$ such pairs which can be enumerated easily. By increasing the number of choice, we can achieve minimization by choosing the one that has the minimum value of expected distortion. The following describes the detailed steps of embedding k message bits in a n -sized block using modified matrix coding $(2, n, k)$ and

rounding-biased rule.

With the given \bar{X}_i and M_i , the corresponding \tilde{X}_i and R_i are calculated from them. We compute α using Eq. (4.7) in the same way that the original matrix coding computes. If $\alpha \neq 0$, we compute $\frac{n-1}{2}$ pairs $(\beta_1, \gamma_1), \dots, (\beta_{\frac{n-1}{2}}, \gamma_{\frac{n-1}{2}})$ such that $\beta_j \oplus \gamma_j = \alpha$, $j = \{1, \dots, \frac{n-1}{2}\}$. Modifying the LSBs of two coefficients in the positions of β_j and γ_j will have the same embedding result with when the α positioned coefficient is changed. Now, the question is which case causes the smallest distortion. We are interested in minimizing distortion between \bar{X}_i and \hat{X}_i , therefore, each expected distortion is examined, and the one that has the least number will be chosen to change. For the case of changing the α position bit, the expected distortion is $1 - |R_{i,\alpha}|$. And for each of the pairs (β_j, γ_j) , the expected distortion is a sum of individual distortion, $(1 - |R_{i,\beta_j}|) + (1 - |R_{i,\gamma_j}|)$. Algorithm 2 describes embedding M_i in \bar{X}_i and producing \hat{X}_i using modified matrix coding.

Algorithm 2 Proposed embedding algorithm to minimize distortion in a single block using modified matrix coding $(2, n, k)$.

```

inputs:  $\bar{X}_i, M_i; |\bar{X}_i| = n, |M_i| = k.$ 
 $\tilde{X}_i \leftarrow \text{Round}(\bar{X}_i);$ 
 $\hat{X}_i \leftarrow \tilde{X}_i;$ 
calculate  $\alpha$  using Eq. (4.7).
if  $\alpha \neq 0$  then
  find  $(\beta_k, \gamma_k)$   $1 \leq k \leq \frac{n-1}{2};$ 
  find  $(\beta_j, \gamma_j)$  corresponding to  $\mu = \min_{1 \leq k \leq \frac{n-1}{2}} (2 - |R_{i,\beta_k}| - |R_{i,\gamma_k}|);$ 
  if  $(1 - |R_{i,\alpha}|) \leq \mu$  then
    create  $\hat{X}_{i,\alpha}$  by the rounding-biased rule;
     $\hat{X}_i = \{\tilde{X}_{i,1}, \dots, \hat{X}_{i,\alpha}, \dots, \tilde{X}_{i,n}\};$ 
  else
    create  $\hat{X}_{i,\beta_j}$  by the rounding-biased rule;
    create  $\hat{X}_{i,\gamma_j}$  by the rounding-biased rule;
     $\hat{X}_i = \{\dots, \hat{X}_{i,\beta_j}, \dots, \hat{X}_{i,\gamma_j}, \dots\};$ 
  end if
end if

```

4.4 Discussion on Shrinkage

Shrinkage refers to the operation of changing a coefficient to 0. Westfeld introduced this term in [21] because the set of coefficients available for embedding shrinks by the shrinkage operation. Typical embedding algorithms using JPEG images avoid changing zero-valued coefficients to non-zero coefficients because modification of zero-valued coefficients may have effects on compression rates and image quality after embedding. Changing non-zero coefficients to 0s are not avoided but needs special care to synchronize the embedding function with the extracting function. If the embedding function makes some coefficients become 0, the extracting function cannot reveal the message bits embedded in the coefficients created by shrinkage because only non-zero coefficients are considered in the extracting function.

F5 [21] proposed the *repeat-embedding* technique to synchronize embedding and extracting functions even if the embedding function creates additional 0 coefficients. In F5, if a coefficient becomes zero during embedding, the embedding of the message bit begins again at the same position; this is repeated until the message bit is successfully embedded without making a coefficient 0. The repeat-embedding technique solves the synchronization problem between the embedding function and the extracting function; however, it results in not only sacrificing capacity but also leaving a trail behind for the detection tool by increasing the frequency of 0 coefficients. This noticeable change in histogram can be used by steganalysis tools [50] to detect the F5 algorithm.

Our minimizing distortion scheme using the rounding-biased rule also encounters the shrinkage problem. When such coefficients that $\tilde{x}_i = 1$ and $\bar{x}_i \in \bar{X}^-$ or $\tilde{x}_i = -1$ and $\bar{x}_i \in \bar{X}^+$ are changed using the rounding-biased rule, they are changed to zero-valued coefficients. Next two chapters will introduce our two embedding algorithms that solve the shrinkage problem while utilizing our minimizing distortion scheme.

Chapter 5: Shrinkage-avoiding Embedding Algorithms

Shrinkage is the operation of changing a non-zero coefficient to 0 and it needs special care in synchronizing the embedding and extracting functions. Some techniques to deal with the shrinkage may cause the increased risk of being detected. In this chapter, our embedding methods that propose to avoid shrinkage and minimize distortion using parity coding and modified matrix coding are described. The modified rounding-biased rule is also proposed. Distortions due to embedding for both methods are formally derived. From the analysis, the amount of embedding distortion is estimated with a given number of message. This prediction may help a sender to choose the embedding rate with the given level of tolerable distortion automatically.

5.1 Parity Coding: PB

We propose a parity-coding-based embedding method called PB that avoids shrinkage and minimizes distortion.

5.1.1 Embedding Algorithm

After computing DCT, all non-zero AC coefficients are marked for possible embedding and collected to form \bar{X} . The corresponding rounding errors R and \tilde{X} are calculated. The embedding process collects non-zero AC coefficients of \tilde{X} and divides it into blocks of length n . To embed a message bit m_i , the i^{th} block $\tilde{X}_i = \{\tilde{x}_{n(i-1)+1}, \dots, \tilde{x}_{ni}\}$ is considered. \tilde{X}_i is the i^{th} block of \tilde{X} , and so forth. If the parity of the LSB(\tilde{X}_i) is equal to m_i , no change is necessary, $\hat{X}_i = \tilde{X}_i$. On the other hand, if the parity is different from m_i , the j^{th} coefficient of block \tilde{X}_i , $\tilde{X}_{i,j} = \tilde{x}_{n(i-1)+j}$ ($1 \leq j \leq n$) needs to be changed so that the LSB's of block

\hat{X}_i have parity equal to m_i .

Since shrinkage should be avoided in this algorithm, the modified rounding-biased rule is proposed. One exception applies to this algorithm when the selected coefficient belongs to the *shrinkable set* S , which is defined by:

$$S = \bar{X}(-1)^+ \cup \bar{X}(1)^-,$$

where $\bar{X}(c) = \{\bar{x} \mid \bar{x} \in \bar{X} \text{ and } \text{Round}(\bar{x}) = c\}$. Recall that the sign refers to the sign of the corresponding rounding error. In order to avoid shrinkage, the *modified rounding-biased rule* is proposed as:

$$\hat{x} = \begin{cases} 2, & \text{Round}(\bar{x}) = 1 \ \& \ \bar{x} \in S \\ -2, & \text{Round}(\bar{x}) = -1 \ \& \ \bar{x} \in S \\ \text{Round}(\bar{x} - 0.5), & \bar{x} \in X^- \ \& \ \bar{x} \notin S \\ \text{Round}(\bar{x} + 0.5), & \bar{x} \in X^+ \ \& \ \bar{x} \notin S. \end{cases} \quad (5.1)$$

When \bar{x} is changed to \hat{x} , the additional error due to embedding ε is given by:

$$\varepsilon = \begin{cases} 1, & \bar{x} \in S \\ 1 - 2|r(\bar{x})|, & \text{otherwise.} \end{cases} \quad (5.2)$$

Since minimizing distortion is primary in this algorithm, each expected distortion with the modified rounding-biased rule is examined, and the process will choose the one that causes the least distortion.

To describe our method, two examples of images are used. We embedded random messages with various parity block sizes in the images, whose rounding error distributions are different, as shown in Figure 5.1. The images are color JPEG images: the left image has the dimensions of 766×1041 , and its rounding errors have a Gaussian-like distribution;

the right image is a size-reduced version of the left image, 536×729 , and its rounding errors have a uniform-like distribution as shown in Figure 5.2. Distortion is analyzed and can be estimated from the probability distribution of the rounding errors.

5.1.2 Error Analysis

We have assumed that all r_i are i.i.d. random variables and their probability density $f_r(x)$ is known. We estimate the rounding error density by normalizing its histogram. The distribution for $\psi = |r|$ is given by

$$F_\psi(x) = \int_{-x}^x f_r(x) dx, \quad x \in [0, 0.5]. \quad (5.3)$$

Figure 5.2 and 5.3 show the normalized rounding error histograms and their probability distributions for two test images of Figure 5.1. Based on the rounding error distribution, we estimate the probability distribution of embedding error that is caused when a coefficient is changed. In this embedding method, there are two kinds of embedding error. One is when the coefficient belongs to S ; the other is when the coefficient does not belong to S .

Let U denote a set of usable coefficients (non-zero AC coefficients in this method). Let S be a shrinkable set and be S^c denote $U - S$. As mentioned previously, ε for $\bar{x} \in S$ is $1 - 2|r(\bar{x})|$ and ε for $\bar{x} \notin S$ is 1. Let $p = \frac{|S^c|}{|U|}$, (i.e., the relative proportion of all coefficients that belong to S^c) and let $q = 1 - p$ (i.e., the relative proportion of S .)

First, the distortion for those that belong to S^c is analyzed. We are looking for a coefficient that will cause the smallest embedding error within every block, \bar{X}_i . If there are $n_p > 0$ coefficients from S^c in a given block, we will choose the coefficient corresponding to the minimum error among the n_p coefficients. Since the error for the coefficients from S is 1, which is always greater than the errors caused by the coefficients from S^c , the remaining $(n - n_p)$ coefficients are not considered.

Probability distribution $F_\nu(x)$ for the embedding errors of the coefficients in S^c is given



Figure 5.1: Images used in this experiment. They have different rounding error distributions. The height and the width of the right image are 75% of the left image dimensions.

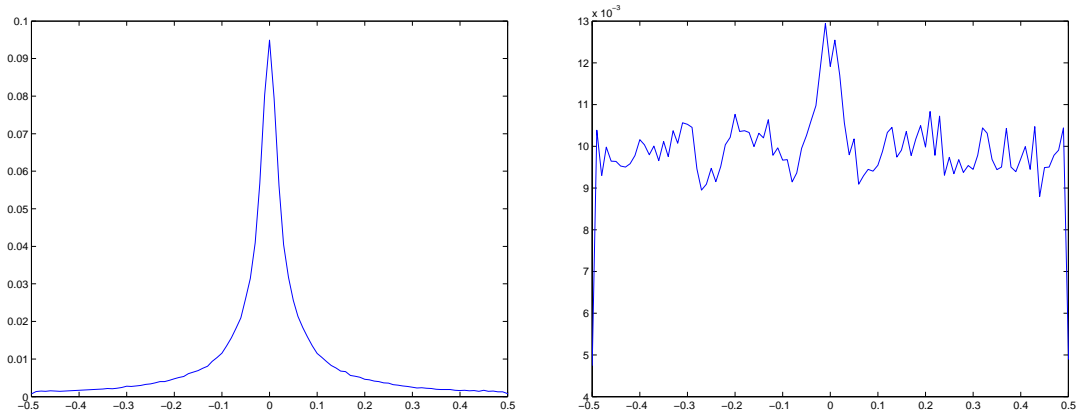


Figure 5.2: Normalized rounding error histograms of non-zero AC JPEG coefficients of the test images in Figure 5.1a (left) and Figure 5.1b (right). We estimate the distribution of the embedding error using the normalized histogram.

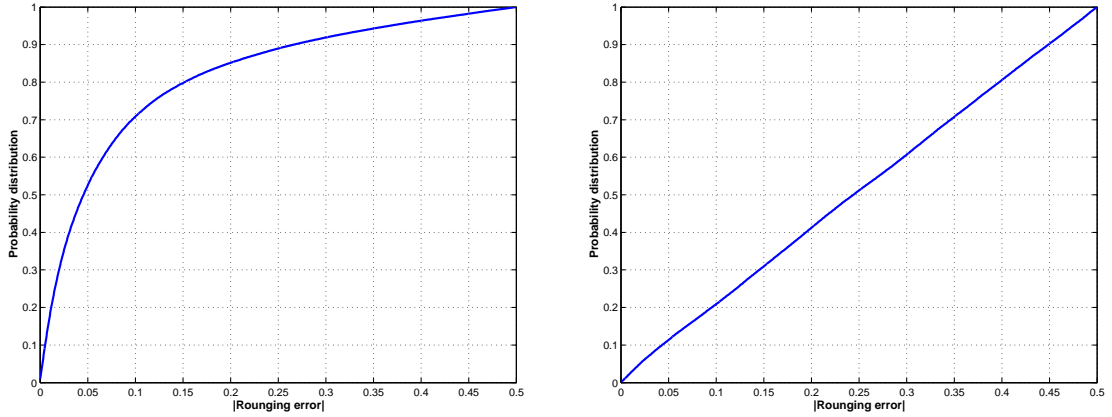


Figure 5.3: Probability distribution of the rounding errors ($F_\psi(x)$) of the test images in Figure 5.1a (left) and Figure 5.1b (right).

by

$$F_\nu(x) = 1 - F_\psi\left(\frac{1-x}{2}\right), \quad \nu \in [0, 1], \quad (5.4)$$

and is shown in Figure 5.4 for two test images of Figure 5.1.

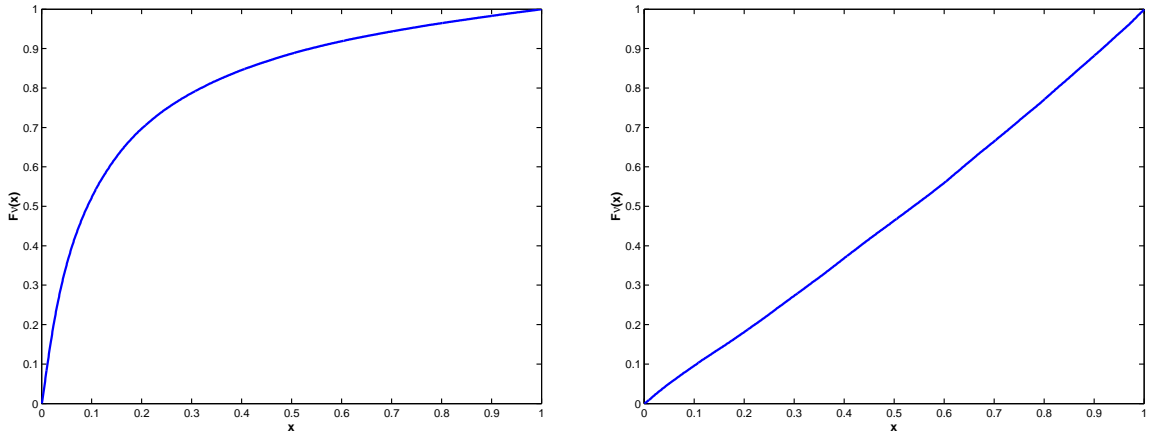


Figure 5.4: Probability distribution of embedding errors ($F_\nu(x)$) of the test images in Figure 5.1a (left) and Figure 5.1b (right).

We are interested in selecting the one that causes the smallest embedding error. We estimate the probability distribution of the smallest embedding error by using order statistics [71]. Given the probability distribution $F_\nu(x)$, the distribution of the smallest distortion

μ when $n_p = s$ is given by

$$F_\mu(x, s) = P_s\{\mu \leq x | n_p = s\} = \begin{cases} U(x - 1), & s = 0 \\ 1 - (1 - F_\nu(x))^s, & s \geq 1, \end{cases} \quad (5.5)$$

where

$$U(x - 1) = 0, \quad x < 1$$

$$U(x - 1) = 1, \quad x \geq 1$$

and $F_\nu(x)$ is given by Eq. (5.4).

The minimal additional error due to embedding is shown in Figure 5.5 in various block sizes with two test images of Figure 5.1. Figure 5.6 shows distortion density with various

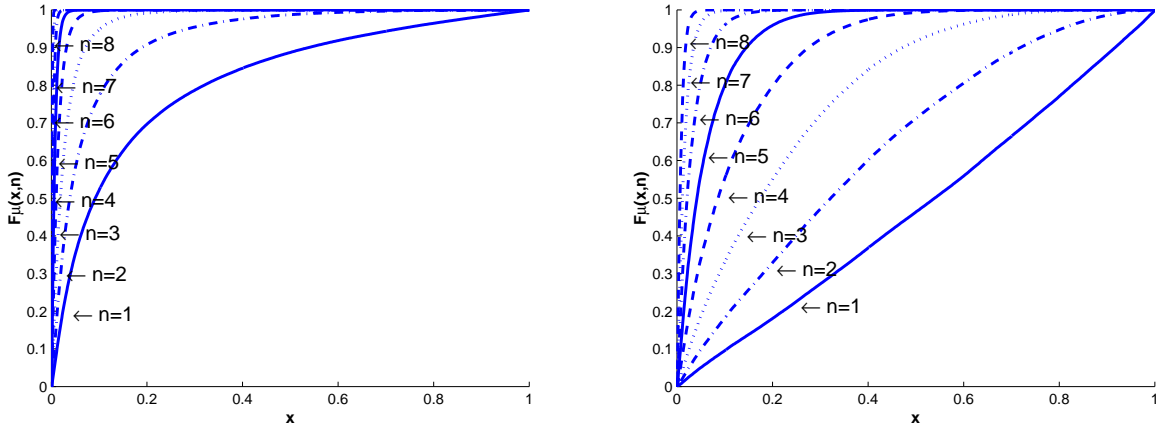


Figure 5.5: Probability distribution of additional error due to embedding ($F_\mu(x, n)$) of the test images in Figure 5.1a (left) and Figure 5.1b (right).

values of n . It can be observed that when $n > 8$, densities look very similar to the density with no embedding. This similarity of densities is shown for both test images, whose distribution of rounding errors are different.

After taking account of all possible combinations of the coefficients, the distribution of

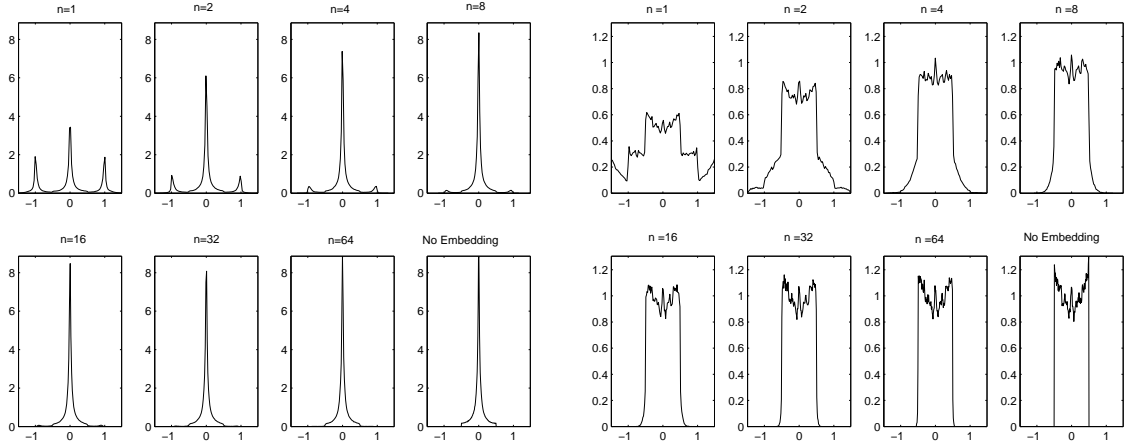


Figure 5.6: Distortion density for various values of n using the test images in Figure 5.1a (left) and Figure 5.1b (right).

additional error will be given by

$$F_{\mu}(x) = \sum_{s=0}^n \binom{n}{s} p^s q^{n-s} F_{\mu}(x, s). \quad (5.6)$$

The expected value of the embedding error will then be given by

$$E[\mu] = \sum_{s=0}^n \binom{n}{s} p^s q^{n-s} E[\mu | n_p = s], \quad (5.7)$$

where

$$E[\mu | n_p = s] = \int_0^{\infty} x dF_{\mu}(x, s), \quad s \geq 0.$$

Figure 5.7 shows the results of the error analysis indicating that the theoretically estimated distortion matches closely to the actual distortion. Figure 5.8 depicts the theoretically estimated error and 100 experimental errors (obtained from embedding 100 different messages using the same image) using a box plot. The box plot indicates that the estimation is very accurate.

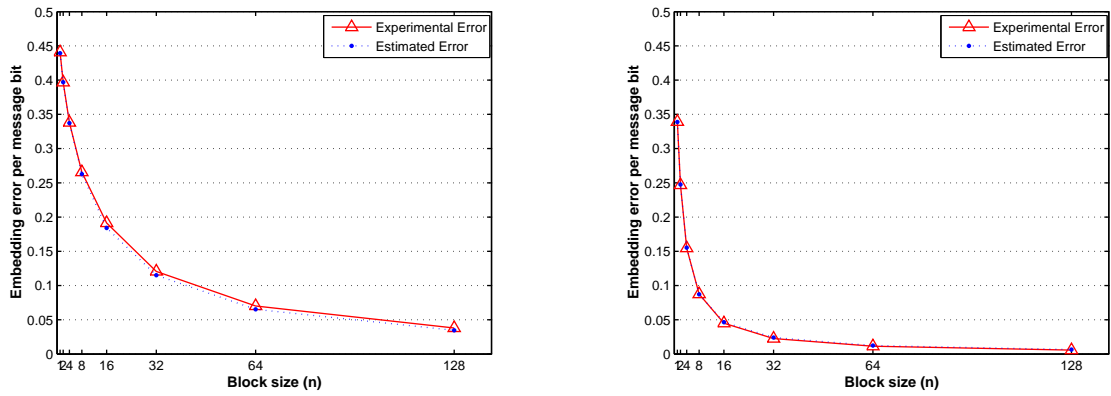


Figure 5.7: Embedding error analysis for the test images in Figure 5.1a (left) and Figure 5.1b (right). The estimated distortion and experimental distortion are very close.

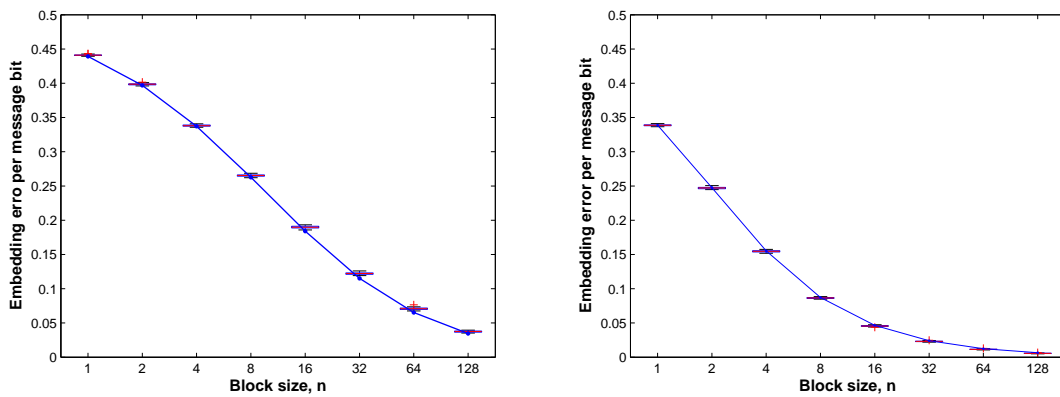


Figure 5.8: We embedded 100 different messages to the test images in Figure 5.1a (left) and Figure 5.1b (right), and compared the actual embedding errors with the predicted error in box plots.

5.1.3 Experimental Results

The extended experimental results using the PB algorithm are presented. Figure 5.9 shows average embedding error per message bit with six different embedding rates. 100 images from UCID [72] database were used to embed messages and the results are shown as a box plot.

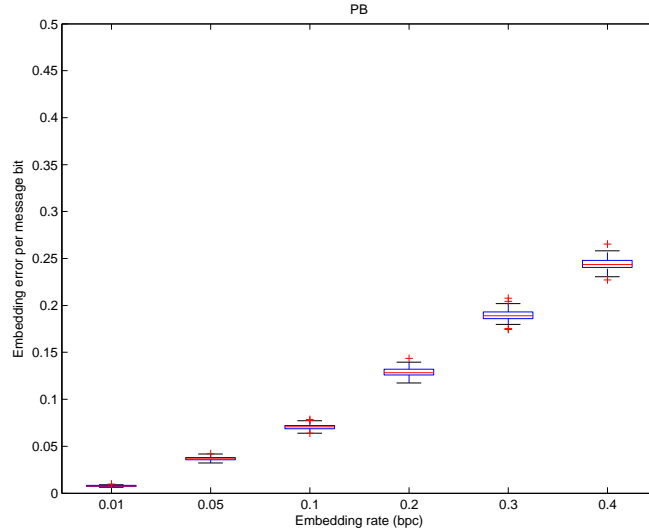


Figure 5.9: Average embedding error per message bit using the PB method with six different embedding rates. 100 images from UCID database were used to embed messages.

To verify accuracy of our error analysis, we repeated the test with 100 different messages using another two images as shown in the first row of Figure 5.10. Figures 5.10 depicts the theoretically-estimated error and 100 experimental errors (obtained from embedding 100 different messages using each image) using a box plot. The box plot shows the accuracy of our estimation.

5.2 Modified Matrix Coding: MMx

We propose a modified-matrix-coding-based embedding method called MMx that avoids shrinkage and minimizes distortion. The details of matrix coding and modified matrix coding are described in Sections 4.3.2 and 4.3.3. As previously mentioned, in (t, n, k) codes

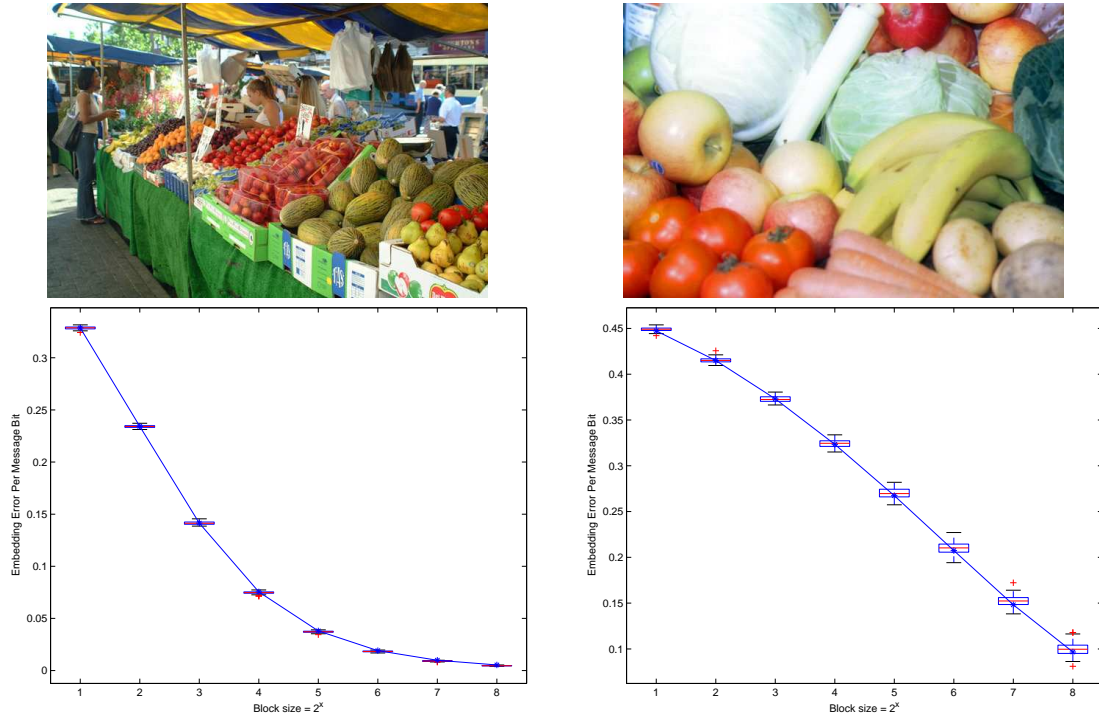


Figure 5.10: Top row: test images; Bottom row: Theoretically estimated embedding error with 100 experimental errors are displayed in a box plot.

where $t \geq 1$, k message bits are embedded by changing at most t coefficients among n coefficients. Depending on t , the number of choice for embedders are determined. For example of $t = 2$ (called MM2), embedders will have additional $\frac{n-1}{2}$ choices in embedding k message bit in n coefficients. If t is increased, the number of choices also is increased. We denote MM3 and MM4 when $t = 3$ and $t = 4$.

5.2.1 Embedding Algorithm

We apply the modified-rounding-biased rule (Eq. (5.1)) to MMx as we did to PB. When the chosen coefficient(s) belongs to the shrinkable set S , we change 1 to 2 or -1 to -2 in order to avoid shrinkage. For example of MM2, two coefficients may change to embed k bits if their changes cause smaller embedding error than the change of α -positioned coefficient.

To simplify notation, let \bar{X} be n -sized coefficients instead of a set of all usable coefficients. Let ε_0 denote the embedding error when the α -positioned coefficient designated by the

original matrix coding is selected for a change, which will be $1 - 2|r_\alpha|$, where r_α denotes the rounding error of α -positioned coefficient. On the other hand, when an alternative pair to α is selected, the added error when changing (β_i, γ_i) -positioned coefficients is one of four cases:

$$\varepsilon_i = \begin{cases} 2, & \text{if } \bar{x}_{\beta_i} \in S \ \& \ \bar{x}_{\gamma_i} \in S \\ 2 - |r_{\beta_i}|, & \text{if } \bar{x}_{\beta_i} \notin S \ \& \ \bar{x}_{\gamma_i} \in S \\ 2 - |r_{\gamma_i}|, & \text{if } \bar{x}_{\beta_i} \in S \ \& \ \bar{x}_{\gamma_i} \notin S \\ 2 - 2(|r_{\beta_i}| + |r_{\gamma_i}|), & \text{otherwise.} \end{cases} \quad (5.8)$$

In order to decide how to create \hat{X} , we find

$$\mu = \min_i \{\varepsilon_i\}, \quad 0 \leq i \leq \frac{n-1}{2}.$$

Given μ , we compute \hat{X} by

$$\hat{X} = \begin{cases} \tilde{X}, & \text{if } \alpha = 0 \\ \{\tilde{x}_1, \dots, \hat{x}_\alpha, \dots, \tilde{x}_n\}, & \text{if } \mu = \varepsilon_0 \\ \{\tilde{x}_1, \dots, \hat{x}_{\beta_i}, \dots, \hat{x}_{\gamma_i}, \dots, \tilde{x}_n\}, & \text{if } \mu = \varepsilon_i, \ i \in \{1, \dots, \frac{n-1}{2}\} \end{cases} \quad (5.9)$$

where \hat{x} is obtained by the modified-rounding-biased rule.

5.2.2 Error Analysis

In MM2 ($t = 2$), we compare ε_0 and ε_i for $i = \{1, \dots, \frac{n-1}{2}\}$ and select the coefficient(s) that will cause the least added error. S^c and S , and their relative proportion p and q were defined in Section 5.1.2.

For a given block of coefficients, $\{\bar{x}_1, \dots, \bar{x}_n\}$ of size n , two cases we should care about

are

$$\begin{aligned}\bar{x}_\alpha \notin S \ \& \ \bar{x}_\beta \notin S \ \& \ \bar{x}_\gamma \notin S, \\ \bar{x}_\alpha \in S \ \& \ \bar{x}_\beta \notin S \ \& \ \bar{x}_\gamma \notin S.\end{aligned}$$

For example, the case of $\{\bar{x}_\alpha \notin S \ \& \ \bar{x}_\beta \in S \ \& \ \bar{x}_\gamma \in S\}$ does not need to be considered because the embedding error caused by changing \bar{x}_α is always smaller than the one caused by changing \bar{x}_β and \bar{x}_γ .

There are $h(= \frac{n-1}{2})$ alternative $(\bar{x}_\beta, \bar{x}_\gamma)$ pairs to change \bar{x}_α in MM2. There will be n_p pairs in which both \bar{x}_β and \bar{x}_γ are not from S , where $0 \leq n_p \leq h$. The probability of a single event that both coefficients are from S^c is p^2 . For any n_p , the probability of both coefficients of the selected pair to be from S^c will be

$$P\{n_p = s\} = \binom{h}{s} (p^2)^s (1 - p^2)^{h-s}. \quad (5.10)$$

As we did in the PB algorithm, the embedding error when $\{\bar{x}_\alpha, \bar{x}_\beta, \bar{x}_\gamma\} \notin S$ is first analyzed. The embedding error when changing \bar{x}_α is $1 - 2|r_\alpha|$ and the error when changing the pair $\{\bar{x}_\beta, \bar{x}_\gamma\}$ will be $2 - 2(|r_\beta| + |r_\gamma|)$. Rounding errors are assumed to be random variables for analysis. Probability density of rounding errors $f_r(x)$ is estimated from the normalized rounding error histogram. Probability distribution for $\psi = |r|$ is given by

$$F_\psi(x) = \int_{-x}^x f_r(x) dx, \quad x \in [0, 0.5].$$

Probability density for $z = |r_1| + |r_2|$ is given by

$$f_z(x) = f_\psi(x) \otimes f_\psi(x), \quad z \in [0, 1],$$

where \otimes stands for convolution. Probability distribution, $F_z(x)$ is given by

$$F_z(x) = \int_0^z f_z(x)dx, \quad z \in [0, 1].$$

Probability distribution for $\nu = 1 - 2\psi$ is given by

$$F_\nu(x) = 1 - F_\psi\left(\frac{1-x}{2}\right), \quad \nu \in [0, 1]. \quad (5.11)$$

Probability distribution for $\omega = 2 - 2z$ is given by

$$F_\omega(x) = 1 - F_z(2-x), \quad \omega \in [0, 2]. \quad (5.12)$$

Embedding error due to the change of $\bar{x}_\alpha \in S^c$ will follow the distribution of $F_\nu(x)$ and the changes of \bar{x}_β and \bar{x}_γ will follow the distribution of $F_\omega(x)$.

In the MMx algorithm, the process will choose \bar{x}_α or one of the pairs of $\{\bar{x}_\beta, \bar{x}_\gamma\}$ by comparing their expected embedding errors. To estimate the probability distribution of the error, order statistics [71] is applied. As the first approximation, the case of $\bar{x}_\alpha \notin S$ is analyzed. The distribution of a minimum embedding error μ when $n_p = i$ for $i \geq 0$ is given by

$$F_\mu(x, i, S^c) = P_{i, S^c}\{\mu \leq x | n_p = i\} = 1 - (1 - F_\nu(x))(1 - F_\omega(x))^i, \quad (5.13)$$

where $F_\nu(x)$ and $F_\omega(x)$ are given by Eq. (5.11) and Eq. (5.12).

On the other hand, the distribution of μ when $n_p = i$ and $x_\alpha \in S$ is given by

$$F_\mu(x, i, S) = P_{i, S}\{\mu \leq x | n_p = i\} = \begin{cases} U(x-1), & i = 0 \\ 1 - (1 - F_\omega(x))^i, & i \geq 1, \end{cases} \quad (5.14)$$

where

$$\begin{aligned} U(x-1) &= 0, & x < 1 \\ U(x-1) &= 1, & x \geq 1. \end{aligned}$$

After taking account of all possible combinations of the coefficients, the distribution of additional error will be given by

$$F_\mu(x) = \sum_{i=0}^h \binom{h}{i} p^{2i} (1-p^2)^{h-i} p F_\mu(x, i, S^c) + q F_\mu(x, i, S). \quad (5.15)$$

The expected value of distortion due to embedding, $E[\mu]$, is given by

$$E[\mu] = \sum_{i=0}^h \binom{h}{i} p^{2i} (1-p^2)^{h-i} p E[\mu|n_p = i, S^c] + q E[\mu|n_p = i, S], \quad (5.16)$$

where

$$\begin{aligned} E[\mu|n_p = i, S^c] &= \int_0^\infty x dF_\mu(x, i, S^c), \\ E[\mu|n_p = i, S] &= \int_0^\infty x dF_\mu(x, i, S). \end{aligned}$$

Since changes occur in $\frac{n}{n+1}$ cases in any block, the expected embedding error per block is $\frac{n}{n+1} E[\mu]$.

We embedded messages using the MM2 method in two examples of images shown in Figure 5.11. The images are color JPEG images: the left image has the dimension of 427×278 and the right image has dimension of 330×222 . Their normalized rounding error histograms of non-zero AC coefficients are shown in the bottom row of Figure 5.11. Figure 5.12 shows the plots of the predicted embedding error and the real experimental embedding error. All tests were performed with six different block-size codes: $(2, 2^{k-1}, k)$, $k = 2, \dots, 7$. The plot for each test image shows that the theoretical prediction accurately

matches the actual embedding error. The theoretical error is calculated by Eq. (5.16).

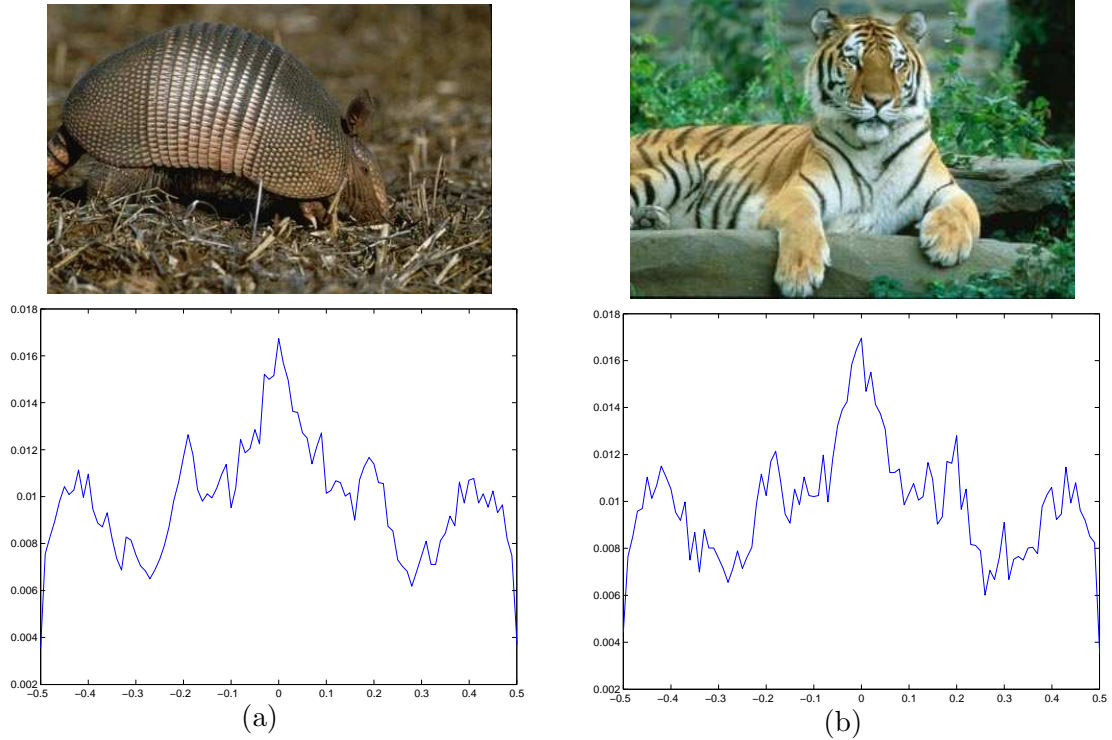


Figure 5.11: Left column: armadillo image. Right column: tiger image. Top row: test images. Bottom row: normalized rounding error histograms of the non-zero AC JPEG coefficients. The histogram is normalized to estimate a probability density of rounding errors.

5.2.3 Experimental Results

The extended experimental results using the MMx algorithm are presented. We extend t to 3 and 4, and denote them by MM3, MM4 respectively. MM3 additionally finds (β, γ, ζ) such that $\beta \oplus \gamma \oplus \zeta = \alpha$ besides finding (β, γ) such that $\beta \oplus \gamma = \alpha$. The embedding error in MM3 is decreased compared to that in MM2; however, the embedding error in MM4 is almost the same as the error in MM3, as illustrated in Figure 5.13. Hence we stopped the extension at MM4.

Figure 5.14 shows the average embedding error per message bit caused by MM2 and MM3. 100 images from UCID database were used to embed messages with six different embedding rates and the results are shown as a box plot.

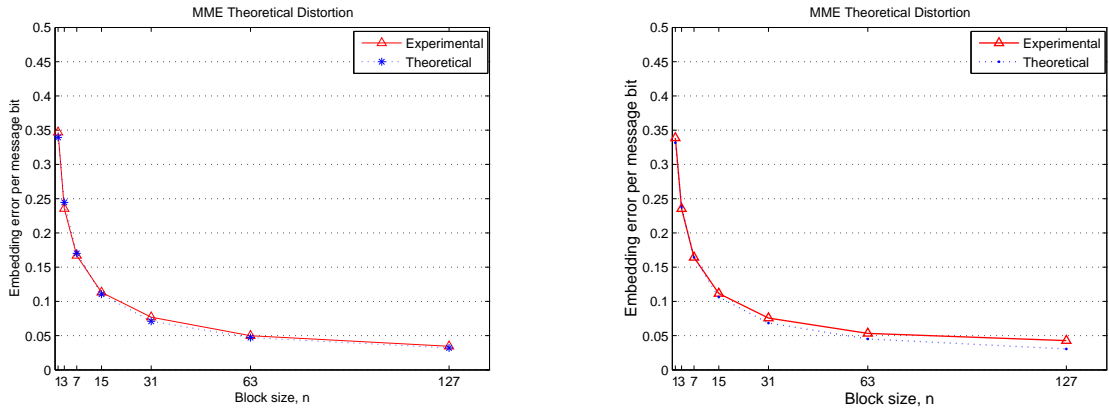


Figure 5.12: Embedding error analysis for the test images in Figure 5.11a (left) and Figure 5.11b (right) for various block sizes, 2^{k-1} for $k = 2, \dots, 7$.

In order to verify our analysis on MM2, we estimated the expected values of embedding errors from 100 images and compared them to experimentally obtained embedding errors. Messages were embedded in six different embedding rates. Figure 5.15 shows the absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors and their results are shown as a box plot.

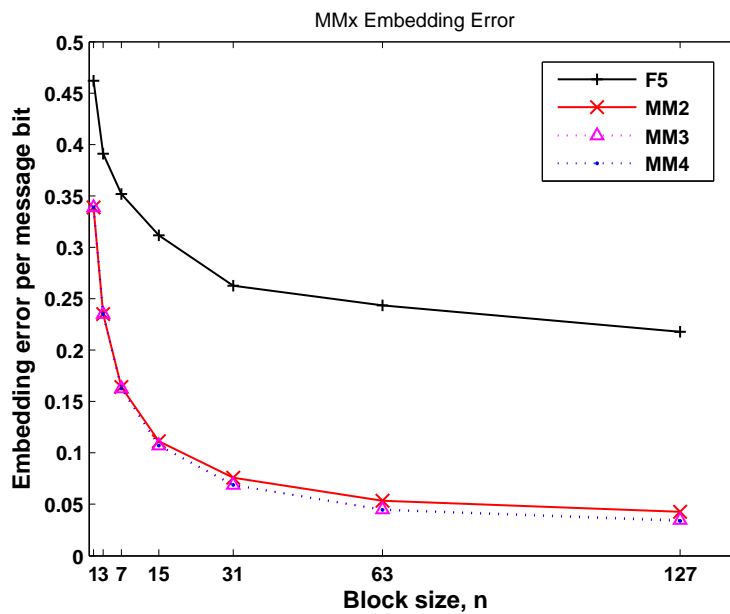
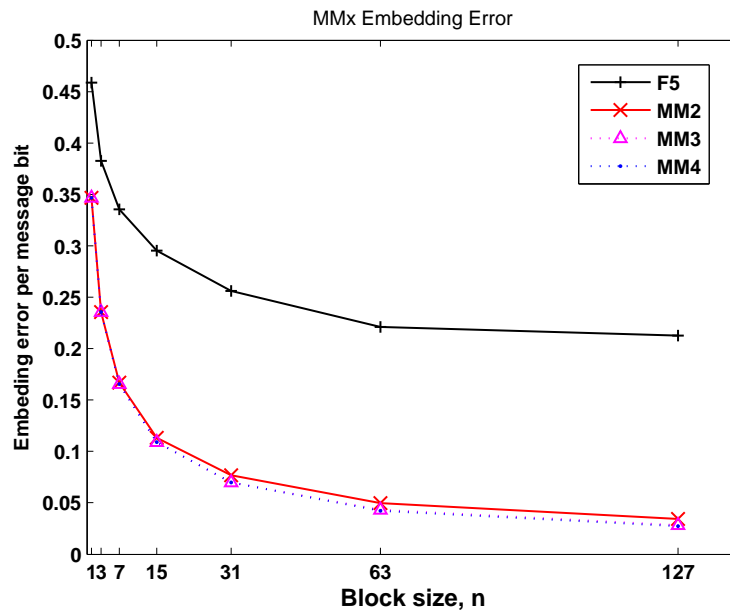


Figure 5.13: Average embedding error per message bit for F5, MM2, MM3, and MM4 using the test images in Figure 5.11a (top left) and Figure 5.11b (top right).

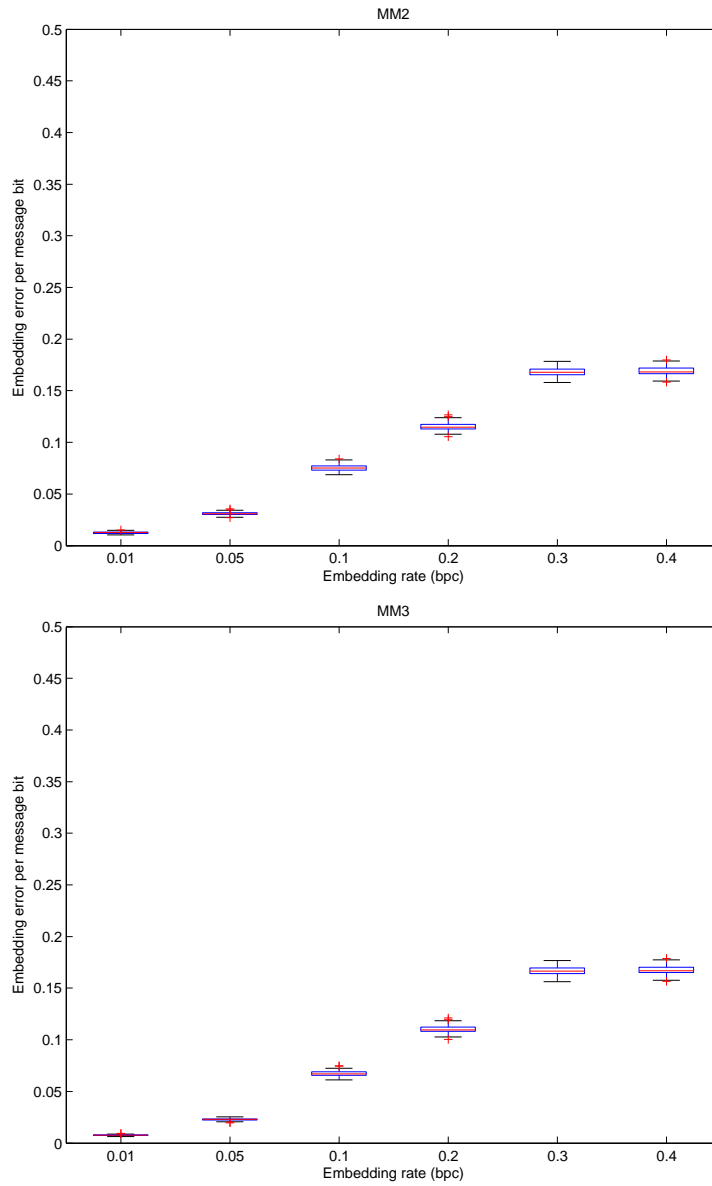


Figure 5.14: Average embedding error per message bit using the MM2 and MM3 methods with six different embedding rates. 100 images from UCID database were used to embed messages.

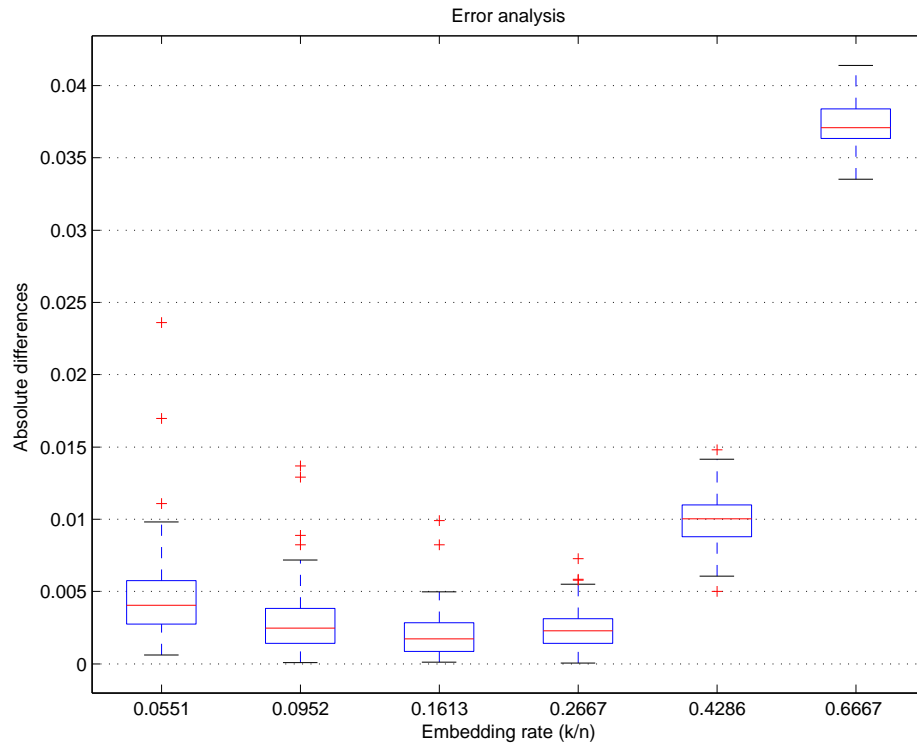


Figure 5.15: Accuracy of the error analysis on MM2. The absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors using 100 images from UCID database.

Chapter 6: Shrinkage-permitting Embedding Algorithms

The shrinkage-avoiding embedding algorithm described in the previous chapter successfully avoids shrinkage while utilizing our distortion minimization scheme.¹ However, since the coefficients 1 and -1 are always changed into 2 and -2 , the algorithm possibly makes a noticeable modification in the coefficient histogram especially with high embedding rates. Figure. 6.1 shows frequencies of coefficients -10 through 10 for the cover image compared with those for the PB stego image in the left graph and those for the MMx stego image in the right graph. It is observable that the frequencies of the coefficients 1 and -1 are decreased and the frequencies of the coefficients 2 and -2 are increased in both stego image histograms. In addition to the modification in the histogram, the exceptional changes from

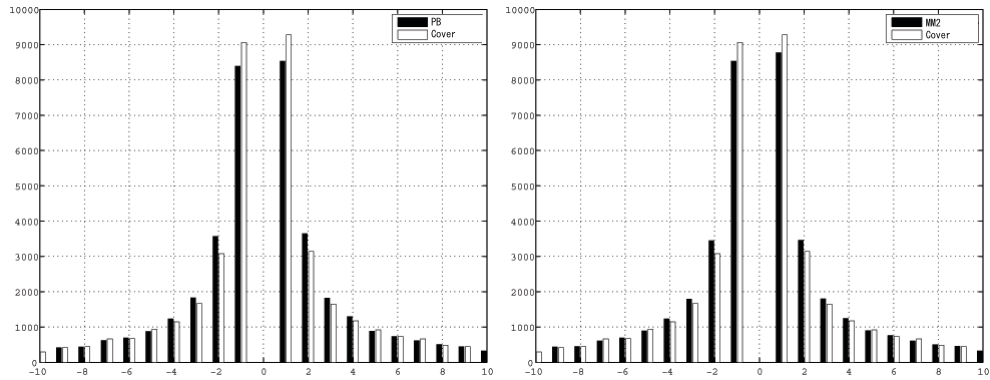


Figure 6.1: Histogram comparison of the cover image and the stego image. The armadillo image (the left image of Figure. 5.11) is used with 0.4 bpc embedding rate. Left: Coefficient histogram of the cover image and the PB stego image. Right: Coefficient histogram of the cover image and the MM2 stego image.

1 and -1 to 2 and -2 cause larger distortion than changing them to 0. For example, the coefficient $\tilde{x}_i = 1$ and $\bar{x}_i \in X^-$, \hat{x}_i will be 2, and the distance between \bar{x}_i and \hat{x}_i will be $1 + |r_i|$ instead of $1 - |r_i|$. Recall that when changing it to 0, the distance would be $1 - |r_i|$.

¹In order to avoid the shrinking operation, the algorithm proposed to change the coefficients 1 and -1 to 2 and -2 respectively. It did not reduce the embedding capacity and could also activate the block-based minimization scheme without shrinkage.

In this chapter, we propose a shrinkage-permitting embedding algorithm that makes the modification of the coefficient histogram less noticeable while applying the rounding-biased rule for minimal distortion. The proposed methods allow the shrinkage operation with the repeat-embedding technique (See Section 4.4) proposed in F5; however, our methods first compensate for the expected modification by shrinkage during the preprocessing stage and then allow the shrinkage operation with the repeat-embedding technique during the embedding stage. Since the repeat-embedding technique will lead to an increase in the frequency of 0 coefficients, the initial compensation will be a decrease in the frequency of 0 coefficients. Therefore, if enough coefficients of 0 have changed to non-zero-valued coefficients in advance, the repeat-embedding caused by shrinkage will not increase the total number of coefficients 0: There will be no net increase in the frequency of 0 coefficients. The algorithm will be described using two coding techniques: parity coding and modified matrix coding.

6.1 Parity Coding: PB-s

The proposed shrinkage-permitting embedding algorithm based on parity coding called PB-s is described in this section. First, the amount of expected shrinkage is estimated. Second, zero-valued coefficients are changed to non-zero coefficients. Third, a message is embedded while permitting shrinkage.

6.1.1 Estimation of Shrinkage

Shrinkage is allowed but limited in the shrinkage-permitting embedding algorithm. In terms of distortion per message bit, shrinkage with repeat-embedding causes larger distortion than choosing an arbitrary coefficient to change even if the coefficient is not the best one in the block. For this reason, shrinkage is allowed only when all elements in the block belong to the shrinkable set S . Recall that the shrinkable set was defined as a union of $\bar{X}(-1)^+$ and $\bar{X}(1)^-$. In other words, shrinkage will not occur if there is any coefficient in the block that does not belong to the shrinkable set. Again, if the amount of shrinking was accurately estimated and

the number of 0 coefficients has decreased by the accurately estimated number, the number of 0 will not be significantly changed after finishing embedding, even though shrinkage creates additional 0.

It is crucial to accurately estimate the amount of shrinkage. Let n_b denote a block size to embed a single message bit successfully including shrinkage. In the parity-coding based algorithm of block size n , n_b will be the same as n if any shrinkage does not occur; n_b will be greater than n if shrinkage occurs.

The estimation of n_b is first described. If either $\text{Parity}(\text{LSB}(\tilde{X}_i)) = m_i$ or shrinkage does not occur, then $n_b = n$. If more than n coefficients (\bar{x}) from S appear consecutively with no parity matching, n_b will be greater than n . The first shrinkage will occur when all n coefficients are from S . Since changing by the shrinkage operation is not considered as embedding a message bit, after the first shrinkage has occurred, a new coefficient will replace the newly created 0 coefficient, and then the parity will again be checked. If the parity does not match m_i and the newly-added coefficient is from S , the second shrinkage will occur. This continues as described in Algorithm 3 below. And so n_b will increase until the message m_i is successfully embedded without causing shrinking.

Let $q = \frac{|S|}{|U|}$, the relative proportion of the coefficients that belong to S over the usable (non-zero AC) coefficients U , and $p = 1 - q$. From p and q , the expected value of n_b is given by:

$$\begin{aligned} E[n_b]_q &= \frac{1}{2}n + \frac{1}{2}((1 - q^n)n + (1 - q) \sum_{i=0}^{\infty} q^{n+i}(n + i + 1)) \\ &= \frac{1}{2}n + \frac{1}{2}((1 - q^n)n + (1 - q)(\frac{q^n}{1 - q}n + \frac{q^n}{(1 - q)^2})). \end{aligned} \quad (6.1)$$

Indeed, $E[n_b]_q - n$ is the expected amount of shrinkage that occurs per block. And the

expected amount of total shrinkage in an image is:

$$N_s = N_b(E[n_b]_q - n), \quad (6.2)$$

where $N_b = \frac{|U|}{E[n_b]_q}$ is the expected total number of blocks.

Suppose that the shrinkable set was increased in size by N_p during the preprocessing stage and shrinking occurs N_s times during the embedding stage. Our goal is to find the N_p that will be the same or close to N_s , so the total change would be zero or minimal. Algorithm 3 describes the procedure to obtain the optimal N_p . To calculate an optimal N_p , k is initially set to 1 and is gradually increased until it reaches the point that $k \simeq N_s$. Increasing k will change p and q gradually, and N_s will need to be calculated with the updated p and q using Eq. (6.1) and Eq. (6.2).

Algorithm 3 Finding the optimal N_p in parity-based minimizing embedding method with shrinkage

The original shrinkable set S_0 and the usable coefficient set U_0 are given.

$N_p = 0; N_s = |U_0|.$

for $k = 1$ to $|U_0|$ **do**

$q = \frac{|S_0|+k}{|U_0|+k};$

$E[n_b]_q$ is calculated using Eq. (6.1);

$N_b = \frac{|U_0|+k}{E[n_b]};$

if $N_b(E[n_b]_q - n) - k < N_s - N_p$ **then**

$N_s \leftarrow N_b(E[n_b]_q - n);$

$N_p \leftarrow k;$

end if

end for

6.1.2 Embedding Algorithm

The embedding algorithm of PB-s has two stages: the preprocessing stage and the embedding stage. Let I be a color image and each color channel of I is divided into 8×8 blocks. While DCT is being done on the blocks, all AC coefficients before and after rounding are collected in a set \tilde{X} and \tilde{X} , respectively. In order to allow shrinkage with less modification

in a histogram, the proposed algorithm takes some coefficients from the set of 0 coefficients ($\bar{X}(0)$) and changes them to the coefficients that will belong to the shrinkable set S before the embedding stage. This preprocessing will compensate for the reduction in the size of usable (non-zero) coefficients caused by shrinking and make the modification in a histogram less noticeable.

Preprocessing Stage: The embedding process starts with finding the optimal N_p , which was described in the previous section. Once N_p is obtained, the process collects the coefficients of $\bar{X}(0)$ in decreasing order by the absolute values of their rounding errors. According to the rounding-biased rule, changing a coefficient that has a larger absolute value of rounding error is beneficial in keeping a low distortion. To keep the size of \bar{X}^+ and \bar{X}^- balanced, $\bar{X}(0)^-$ and $\bar{X}(0)^+$ are separately sorted in decreasing order of absolute values of their rounding errors:

$$\begin{aligned} A &= a_1, a_2, \dots, a_{N_1}, \\ B &= b_1, b_2, \dots, b_{N_2}, \end{aligned}$$

where $a_i \in \bar{X}(0)^-$ and $|r(a_i)| \geq |r(a_{i+1})|$ for $1 \leq i \leq N_1 - 1$ and $b_i \in \bar{X}(0)^+$ and $|r(b_i)| \geq |r(b_{i+1})|$ for $1 \leq i \leq N_2 - 1$. We assume that $N_1 + N_2 > N_p$.

The first $\frac{N_p}{2}$ coefficients of A and B , denoted $A_{\frac{N_p}{2}}$ and $B_{\frac{N_p}{2}}$, are now selected for changing their corresponding rounded coefficients: when $\bar{x}_i \in A_{\frac{N_p}{2}}$, $\tilde{x}_i = -1$, and when $\bar{x}_i \in B_{\frac{N_p}{2}}$, $\tilde{x}_i = 1$. The corresponding rounding errors need to be updated using $r_i = \bar{x}_i - \tilde{x}_i$, which produces $|r| \geq 0.5$. This preprocessing has increased the size of S while decreasing the size of $\bar{X}(0)$ by the size of N_p . In other words, $S = S_1 \cup S_2$; S_1 denotes the original shrinkable set, and S_2 denotes the newly added shrinkable set at the preprocessing stage.

Embedding Stage: The embedding process continues with the preprocessed \bar{X} . At this point, \bar{X} can be rearranged for security in a random order, which is determined by a secret key. As this algorithm embeds a message using non-zero AC coefficients and parity

coding of block size n , n usable coefficients from \bar{X} are used for embedding a bit.

Algorithm 4 describes the detailed steps that the embedding process follows to embed a single message bit, when shrinkage is permitted. First, it collects the next n available coefficients to form $Y = \{y_1, y_2, \dots, y_n\}$. Secondly, it checks the parity of $LSB(Y)$, and, if the parity matches the message bit, the message bit is successfully embedded. Otherwise, the embedding will change one or more coefficients. To examine whether shrinkage can be avoided or not, the process counts how many coefficients belong to S in Y . If there is more than one coefficient that does not belong to S , shrinkage can be avoided in Y . If shrinkage can be avoided, the embedding error that the coefficients not in S would cause are examined. The coefficient that will cause the smallest embedding error will be selected to change. The modification of the selected coefficient follows the rounding-biased rule. In Algorithm 4, let \bar{x}_k and \hat{x}_k be the coefficient corresponding to y_j in \bar{X} and \hat{X} , respectively. The if statement embeds a message bit by changing a coefficient without shrinkage; However, if the process is in the else statement, shrinkage occurs and this changing is not considered as embedding a message bit. After shrinkage occurs, the 0 coefficient is removed from Y and the next available coefficient comes to Y . Shrinkage can occur repeatedly before the message bit is successfully embedded.

Algorithm 4 Embedding a message bit in a block while permitting shrinkage.

$Y \leftarrow$ next n available coefficients from \tilde{X} .

while $Parity(LSB(Y)) \neq m$ **do**
 $n_s \leftarrow |\{x|x \in S \text{ and } x \in Y\}|$.
if $n_s < n$ **then**

find j such that $|r(y_j)| = \max_{1 \leq k \leq n} |r(y_k)|$, where $y_j \notin S$;
find \bar{x}_k that is corresponding to y_j ;
produce \hat{x}_k by the rounding-biased rule;

else

find j such that $|r(y_j)| = \max_k |r(y_k)|$, where $y_j \in S$;
find \bar{x}_k that is corresponding to y_j ;
 $\hat{x}_k \leftarrow 0$;

replace y_j in Y by the next available coefficient;

end if

end while

6.1.3 Error Analysis

In this section, embedding errors caused by the proposed algorithm is analyzed. Based on this mathematical analysis, the message length to embed can be decided without typical trial-error approach. Changing a coefficient from \bar{x}_j by the rounding-biased rule to \hat{x}_j adds always $1 - 2|r(\bar{x}_j)|$ on top of the distortion due to the rounding operation of JPEG.

With the proposed parity-based embedding algorithm, the total error due to embedding is the sum of three kinds of errors ($\varepsilon = \varepsilon_p + \varepsilon_s + \varepsilon_m$):

- (1) ε_p occurs when N_p coefficients that belong to $\tilde{X}(0)$ are changed to the $\bar{X}(-1)$ and $\bar{X}(1)$ at the preprocessing stage.
- (2) ε_s occurs when shrinking occurs at the embedding stage.
- (3) ε_m occurs when coefficients are changed and the changes result in embedding message bits at the embedding stage.

Expectation of ε_p : At the preprocessing stage, the N_p coefficients from $\bar{X}(0)$ that have the highest absolute values of rounding errors will be chosen to change into $\bar{X}(-1)$ or

$\bar{X}(1)$ (see Eq. (6.3)). Let R_p be a set of the rounding errors correspondent to the chosen coefficients: $R_p = \{r_i | r_i = r(\bar{x}_i), \bar{x}_i \in (A_{\frac{N_p}{2}} \cup B_{\frac{N_p}{2}})\}$, where A and B are defined in Eq. (6.3).

Let $\Phi(r)$ denote the absolute value of r , which can be expressed as $\Phi(R_p) = \sqrt{(R_p)^2}$. We have a probability density $f_{R_p}(x)$, and $F_{\Phi}(x)$ is calculated by:

$$F_{\Phi}(x) = \int_{-x}^x f_{R_p} x dx, \quad x \in [0, 0.5], \quad (6.3)$$

then we get

$$E[\Phi(R_p)] = \int_0^{\infty} x dF_{\Phi}(x). \quad (6.4)$$

We then have

$$E[\varepsilon_p] = N_p(1 - 2E[\Phi(R_p)]). \quad (6.5)$$

Expectation of ε_s : Shrinking at the embedding stage will occur in two cases: (a) coefficients that belong to the original shrinkable set S_1 are turned to zero; (b) coefficients that belong to the new shrinkable set S_2 , which was added to \bar{X} at the preprocessing stage. When a coefficient in S_1 causes shrinking, an error is added to ε_p ; however, when a coefficient in S_2 causes shrinking, the error that was added in Eq. (6.5) should be subtracted. Because the coefficients in S_2 were originally zero, coming back to zero should not cause any error.

It is important to note that the coefficients in S_2 always have higher priority in being selected than the coefficients in S_1 , because the embedding error caused by turning a coefficient in S_2 to zero is always smaller than by turning a coefficient in S_1 to zero. As previously mentioned, shrinking occurs only when all coefficients in the block belong to the shrinkable set. Since we found N_p as the closest to the expected amount of shrinkage N_s for a given image, we assume that shrinking occurs N_p times, hence the size of S_2 is N_p .

Shrinking in S_1 will occur when all n coefficients of the block are from S_1 . Let $q_1 = \frac{|S_1|}{|U|}$, the relative proportion of S_1 over usable coefficients denoted (U) in \bar{X} . Let R_{s_1} be a set of

rounding errors for the coefficients in S_1 . The expected total error caused by shrinking in S_1 :

$$E[\varepsilon_{s_1}] = N_p(q_1)^n(1 - 2E[\Phi(R_{s_1})]), \quad (6.6)$$

where n is a block size. $E[\Phi(R_{s_1})]$ can be calculated in a similar way to $E[\Phi(R_p)]$ (see Eq. (6.4)). On the other hand, shrinking in S_2 will occur when any of the coefficients that belong to S_2 exists in the block. The expected total error caused by shrinkage in S_2 is:

$$E[\varepsilon_{s_2}] = -N_p(1 - q_1^n)(1 - 2E[\Phi(R_p)]),$$

which has a negative sign because it should be subtracted from ε_p . R_p was defined when ε_p was estimated. Now, the expected amount of the error from shrinkage is

$$E[\varepsilon_s] = E[\varepsilon_{s_1}] + E[\varepsilon_{s_2}].$$

Expectation of ε_m : This error occurs when the coefficients are changed due to embedding a message, not due to shrinkage or preprocessing. In other words, ε_m occurs when there is at least one coefficient that does not belong to S in a given block. The expected error is

$$E[\varepsilon_m] = \frac{1}{2}N_b E[\mu],$$

where N_b is the number of blocks in U and μ is the minimum error among possible choices when a coefficient changes. The following is a description of calculating $E[\mu]$. Let $q = \frac{|S|}{|U|}$, i.e, the relative proportion of the shrinkable set and $p = 1 - q$. Since we are considering e_m , there will be $1 \leq \pi \leq n$ coefficients not in S among the n available coefficients. For any π , the probability that the selected coefficient is not in S is

$$P\{\pi = i\} = \binom{n}{i} p^i q^{n-i}. \quad (6.7)$$

R is a set of rounding errors of U , a set of usable coefficients. We have assumed that rounding errors are i.i.d. random variables and that their probability density $f_r(x)$ is known. As defined previously, $\Phi(R) = \sqrt{R^2}$ and let the distinct value in $\Phi(R)$ be listed in any order as

$$\{\phi_1, \phi_2, \dots, \phi_l, \dots\},$$

and the probability distribution for $\Phi(R)$ is given by the formula

$$F_\phi(x) = \int_{-x}^x f_r(x) dx, \quad x \in [0, 0.5]. \quad (6.8)$$

To estimate the amount of added distortion when a coefficient is changed, the probability distribution for $\nu = 1 - 2\phi$ is obtained by

$$F_\nu(x) = 1 - F_\phi\left(\frac{1-x}{2}\right), \quad x \in [0, 1]. \quad (6.9)$$

If there are π coefficients not in S ($\pi \geq 1$) in a given block, the algorithm will choose the coefficient corresponding to the minimum embedding error among the π coefficients:

$$\mu = \min_{1 \leq k \leq \pi} \{1 - 2|r_k|\},$$

where r_k indicates a corresponding rounding error of \bar{x}_k among π coefficients. The distribution of μ when $\pi = i$ is given by

$$F_\mu(x, i) = P_r\{\mu \leq x | \pi = i\} = 1 - (1 - F_\nu(x))^i, \quad i \geq 1, \quad (6.10)$$

where $F_\nu(x)$ is given by Eq. (7.2). After taking account of all possible combinations of the

coefficients, the distribution of additional error will be given by

$$F_\mu(x) = \sum_{i=1}^n \binom{n}{i} p^i q^{n-i} F_\mu(x, i). \quad (6.11)$$

The expected value of the embedding error will then be given by

$$E[\mu] = \sum_{i=1}^n \binom{n}{i} p^i q^{n-i} E[\mu|\pi = i], \quad (6.12)$$

where

$$E[\mu|\pi = i] = \int_0^\infty x dF_\mu(x, i), \quad i \geq 1.$$

The estimation of total embedding error caused by the proposed parity-based algorithm with shrinkage is completed as

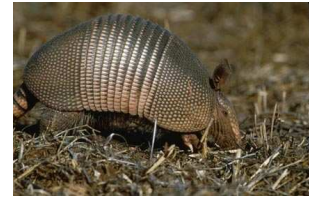
$$E[\varepsilon] = E[\varepsilon_p] + E[\varepsilon_s] + E[\varepsilon_m]. \quad (6.13)$$

6.1.4 Experimental Results

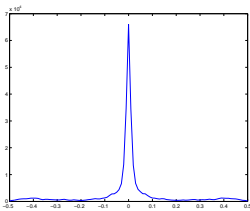
In this section, we demonstrate the experimental results of our method on two test images whose rounding error distributions are different from each other. After computing DCT, all AC coefficients form \bar{X} , and then \bar{X} is preprocessed by increasing the size of the shrinkable set. The implementation follows the algorithm described in Section 6.1.2. Figure 6.2 shows the test images, which are color JPEG images. Rounding error histograms for zero-valued coefficients (R_p) is shown in the second row of Figure 6.2. We make a cumulative histogram with the absolute values of the rounding errors. In the cumulative histogram, a range of the horizontal axis is [0 0.5]. Take the $\frac{N_p}{2}$ below point from the maximum value on the vertical



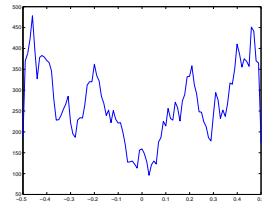
(a) market



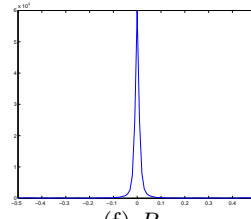
(e) armadillo



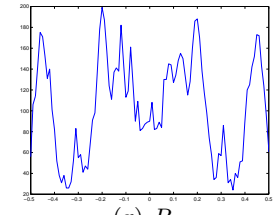
(b) R_p



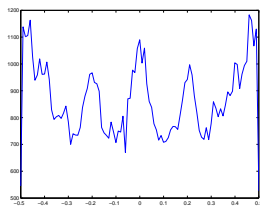
(c) R_s



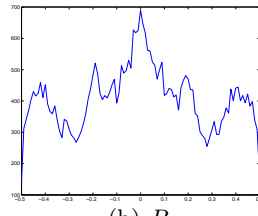
(f) R_p



(g) R_s



(d) R



(h) R

Figure 6.2: Histograms of rounding errors. The left column is for the market image and the right column is for the amarillo image. (b) and (f): histograms of rounding errors for the zero-valued coefficients. (c) and (g): histograms of rounding errors for the shrinkable coefficients. (d) and (h): histograms of rounding errors for the non-zero AC coefficients.

axis. The value of the horizontal axis corresponding to the point is estimated as $E[\Phi(R_p)]$ for the estimation of ε_p in Eq. (6.3). Rounding error histograms for R_s are shown in the third row, and again $E[\Phi(R_{s_1})]$ in Eq. (6.6) is estimated in a similar way to $E[\Phi(R_p)]$. In the fourth row, rounding error histograms for non-zero AC coefficients are shown; we estimate $f_r(x)$ in Eq. (6.8) by normalizing it.

Figure 6.3 shows the theoretical embedding error analysis for PB-s. It depicts the average embedding error per message bit in 7 different block-sizes, and the theoretically estimated values and experimental values are plotted in one graph for comparison. It shows that the theoretical prediction is very close to the actual embedding errors. The tests were accomplished with 7 different block-sizes 2^k ($k = 1, \dots, 7$). Since the embedding rate bpc is the number of embedded message bits divided by the number of non-zero AC coefficients, the larger block size, the smaller bpc.

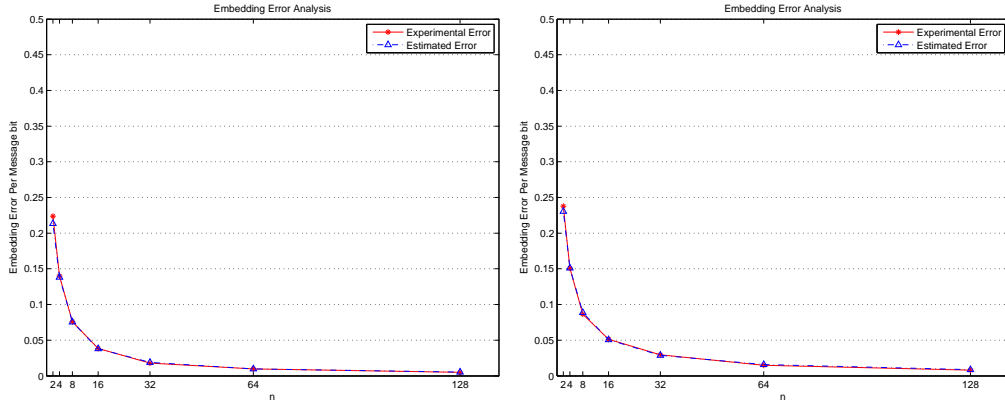


Figure 6.3: Embedding error analysis for the PB-s method in various block sizes (n) using the test images in Figure 6.2a (left) and Figure 6.2e (right). The theoretical embedding errors obtained from our mathematical prediction in Eq. (6.13) and the experimental embedding errors are very close.

In order to verify our analysis on PB-s, we estimated the expected values of embedding errors from 100 images and compared them to experimentally obtained embedding errors. Messages were embedded in six different embedding rates. Figure 6.4 shows the absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors. The result box plot shows that the differences are

very small in various embedding rates.

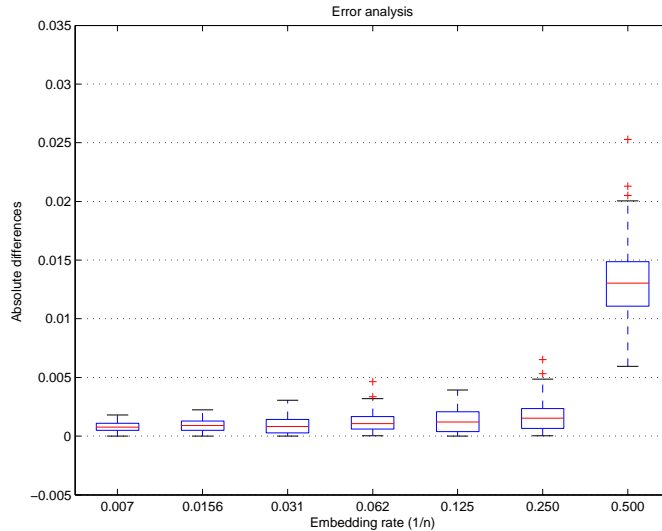


Figure 6.4: Accuracy of the PB-s error analysis. The absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors using 100 images from UCID database.

The extended experimental results using the PB-s algorithm are also presented. Figure 6.5 shows average embedding error per message bit with six different embedding rates. 100 images from UCID database were used to embed messages and the results are shown as a box plot. Obviously the embedding errors increase as the embedding rate gets higher. The result box plot illustrates that the distortion levels caused by PB-s are very low.

We investigated the embedding error caused by the PB-s method compared to other two embedding methods: F5 [21] and PB [49]. Figure 6.6 shows the comparison in embedding errors for those three methods with various embedding rates. The graphs plot the distortions per embedding message bit in increasing embedding rates (bpc), where bpc is the total number of the message bits divided by the usable coefficients (non-zero AC coefficients). PB [49] is an algorithm of trying to minimize distortion due to embedding using parity coding; however, PB proposed the modified-rounding-biased rule in order to avoid shrinkage. The modified-rounding-biased rule contributes to maintain a low level of distortion but leaves a trail in the coefficient histogram by changing the frequencies of the coefficient 2

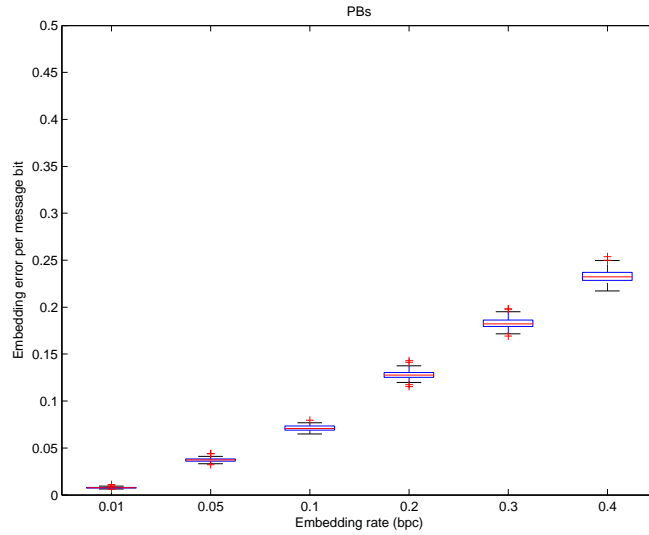


Figure 6.5: Average embedding error per message bit using the PB-s method with six different embedding rates. 100 images from UCID database were used to embed messages.

and -2 in a predictable way. On the other hand, PB-s does not increase the frequencies of 2 or -2 as PB while maintain the low distortion as low as PB as shown in Figure 6.6. The distortions due to both PB and PB-s are noticeably lower than the one due to F5 with the same test images.

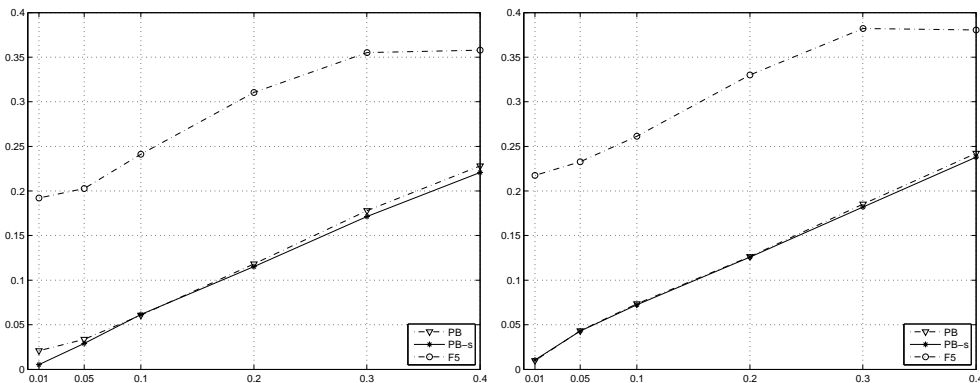


Figure 6.6: Embedding distortion per embedding message bit with various bpc ranging from 0.01 to 0.4 using the test images in Figure 6.2a (left) and Figure 6.2e (right).

We investigated how the PB-s algorithm has changed the distribution of DCT coefficients. For the JPEG steganographic applications, it is necessary to keep the distribution of the stego image's DCT coefficients as close as possible to the cover image in order to

avoid the steganalysis attack; the modified distribution is possibly detected by statistical steganalysis methods. Figure 6.7 shows the histogram modification from the PB-s method with comparison to PB. The differences in the histograms between the stego image and the cover image are shown in $[-10, 10]$ coefficient range in Figure 6.7. These graphs show the differences at various embedding rates: 0.4, 0.3, 0.2, and 0.1 bpc, from the top. The black bars show the difference in the histogram between the PB-s stego image and the cover image, while the white bars show the difference with PB. Note that the histogram modification from PB-s is significantly lower, compared to PB, especially with a high embedding rate. From the results of Figures 6.6 and 6.7, we found that the PB-s algorithm can be capable of not only managing low embedding error in the similar level to PB as shown in Figure 6.6, but also maintaining the statistical properties better than PB as shown in Figure 6.7, which is one of the important requirements for secure steganographic applications.

6.2 Matrix Coding: MMx-s

The shrinkage-permitting embedding algorithm based on matrix coding called MMx-s is proposed. We have applied the algorithm to MM1. The steps of the method are similar to the ones of PB-s. First, the amount of expected shrinking is estimated. Second, zero-valued coefficients are changed to non-zero coefficients. Third, a message is embedded while permitting shrinkage.

6.2.1 Estimation of Shrinkage

The estimation of N_p for the matrix-coding-based algorithm can be described in a similar way to the estimation for the parity-coding-based algorithm. Again, the goal is to find N_p such that it would be the same to the number of actual shrinking, N_s . Let n_b be the average block size for k message bits to be successfully embedded in $(1, n, k)$ matrix coding. The n_b will be the same to n if either $\alpha = 0$ or shrinking does not occur while embedding k bits.

Let $q = \frac{|S|}{|U|}$ (i.e., the relative proportion of the coefficients that belong to the shrinkable

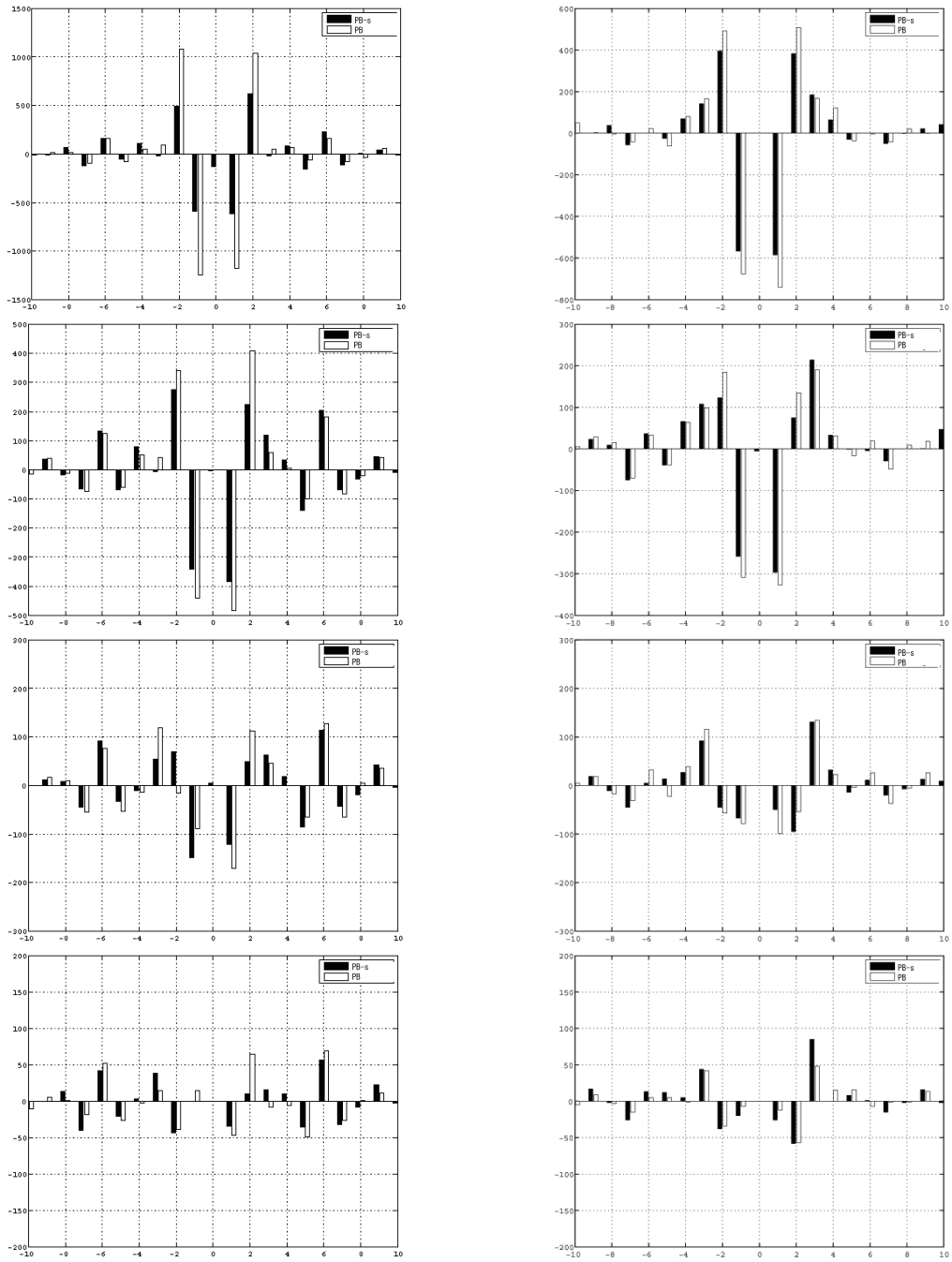


Figure 6.7: Histogram modification from the PB-s method compared to the PB method using the test images in Figure 6.2a (left) and Figure 6.2e (right). The embedding rates are 0.4, 0.3, 0.2, and 0.1 from the top.

set S over the usable coefficients U), and $p = 1 - q$. In the matrix coding, the probability of $\alpha = 0$, which is denoted as u in this section, is $\frac{1}{n+1}$, and let v denote $1 - u$. As shown in Figure 6.8, n_b can be obtained based on p, q, u and v . n_b will be the same to n when either $\alpha = 0$ or shrinking does not occur. The probability of $\alpha = 0$ is u , and the probability that shrinking does not occur is p . Therefore, the probability of $n_b = n$ is $un + vp$. Otherwise, the first shrinking will occur, whose probability is vuq . After the first shrinking occurs, α is calculated again to check whether changing is needed or not. If the α is 0 or the newly added coefficient is not a member of S , then no more shrinking will occur, resulting in $n_b = n + 1$. The probability of $n_b = n + 1$ is $vuq + vuqp$. Otherwise the next shrinking will follow. From p, q, u and v , the expectation of n_b is:

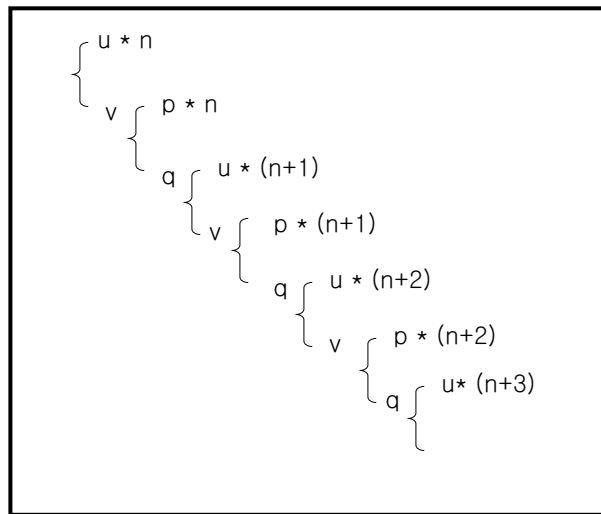


Figure 6.8: The expected block size for embedding a message bit using MM1 with allowing shrinkage. q denotes the probability of shrinkable set over the usable set and $p = 1 - q$. u denotes the probability of $\alpha = 0$ ($\frac{1}{n+1}$) in matrix coding, which is that no change needed for embedding, $v = 1 - u$.

$$\begin{aligned}
E[n_b]_{q,n} &= \sum_{i=0}^{\infty} u(n+i)(vq)^i + \sum_{i=0}^{\infty} vp(n+i)(vq)^i \\
&= un \sum_{i=0}^{\infty} (vq)^i + u \sum_{i=0}^{\infty} (vq)^i i + vpn \sum_{i=0}^{\infty} (vq)^i + vp \sum_{i=0}^{\infty} (vq)^i \\
&= un \frac{1}{1-vq} + u \frac{vq}{(1-vq)^2} + vpn \frac{1}{1-vq} + vp \frac{1}{(1-vq)^2} \\
&= n(u+vp) \frac{1}{1-vq} + (u+vp) \frac{vq}{(1-vq)^2} \tag{6.14}
\end{aligned}$$

As a result, $n_b - n$ is the expected amount of shrinkage that occurs per k message bits and the expected amount of total shrinkage in an image is:

$$E[n_b]_{q,n} = N_b(E[n_b]_{q,n} - n), \tag{6.15}$$

where $N_b = \frac{|U|}{n}$ is the total number of blocks. We assume that the possible maximum message bits are embedded in the image.

The detailed method of finding the optimal N_p for matrix-coding-based algorithm is described in Algorithm 5. Suppose that the shrinkable set has been increased by N_p during the preprocessing stage, and shrinking occurs N_s times during the embedding stage. Our goal is to find N_p such that it is the same or closest to N_s in order to make the total change zero or minimal. To obtain the optimal N_p , i is set to 1 at the beginning and gradually increased until it reaches the point where $i \simeq E[N_s]$. Increasing i will change p and q gradually, and $E[N_s]$ will need to be calculated with updated p and q using Eq. (6.14) and Eq. (6.15). Algorithm 5 describes the procedure to obtain the optimal N_p .

Algorithm 5 Finding the optimal N_p for the matrix-coding-based algorithm

The original shrinkable set S_0 and the usable coefficient set U_0 are given.

$N_p = 0; N_s = |U_0|;$

for $k = 1$ to $|U_0|$ **do**

$q = \frac{|S_0|+k}{|U_0|+k};$

$E[n_b]_{q,n}$ is calculated using Eq. (6.14);

$N_b = \frac{|U_0|+k}{E[n_b]_{q,n}};$

if $N_b(E[n_b]_{q,n} - n) - k < N_s - N_p$ **then**

$N_s \leftarrow N_b(E[n_b]_{q,n} - n);$

$N_p \leftarrow k;$

end if

end for

6.2.2 Embedding Algorithm

The shrinkage-permitting method for the matrix coding is very similar to the one of the parity-coding-based algorithm. It has two stages, the preprocessing and the embedding stage. The steps in the preprocessing stage are identical to the ones of the parity-based algorithm described in Section 6.1.2. In the embedding stage, the matrix-coding-based algorithm checks the bit position to change α instead of checking a parity.

Algorithm 6 describes the detailed steps of embedding k message bits with shrinking enabled. First, next n available coefficients are collected to form $Y = \{y_1, y_2, \dots, y_n\}$. Second, α in matrix coding is calculated using Eq. (4.7), and if $\alpha = 0$, the k message bits are successfully embedded. Otherwise, the coefficient \bar{x}_j that correspondent to y_α is changed by the rounding-biased rule. Since the matrix coding designates the bit position for a change in a block, shrinking occurs if the designated coefficient belongs to the shrinkable set S , which is different from the parity-based algorithm. If shrinking occurs, the message M (k -bit sized) should be embedded again because the coefficients that became zero are not counted as embedding. The embedding process repeats with new Y , which the shrunk y_α is removed and the next available coefficient is added.

In Algorithm 6, let \bar{x}_j be the coefficients corresponding to y_α in \bar{X} . The **if** statement embeds k message bits by changing a coefficient without shrinking; the **else** statement creates

0 by shrinking \hat{x}_j , and this is not considered as embedding the message bits. After shrinking occurs, the 0 coefficient is removed from Y and the next available coefficient comes to Y . Depending on the newly added coefficient, shrinking can occur more than one time before the message bits are successfully embedded.

Algorithm 6 Embedding k message bits in a block while permitting shrinkage.

```

 $Y \leftarrow$  next  $n$  available coefficients from  $\bar{X}$ ;

while  $\alpha \neq 0$ , using  $Y$  do
  if  $y_\alpha \notin S$  then
    find  $\hat{x}_j$  corresponding to  $y_\alpha$ ;
     $\hat{x}_j$  by the rounding-biased rule;
  else
    find  $\hat{x}_j$  corresponding to  $y_\alpha$ ;
     $\hat{x}_j \leftarrow 0$ ;
    replace  $y_\alpha$  in  $Y$  by the next available coefficient;

  end if
end while

```

6.2.3 Error Analysis

Error analysis for the shrinkage enabled algorithm based on the matrix coding is almost identical to the one of parity-based algorithm. With the shrinkage enabled algorithm based on the matrix coding technique, three embedding errors are expected:

- (1) ε_p will occur when N_p coefficients are changed from $\bar{X}(0)$ to either $\bar{X}(-1)^+$ or $\bar{X}(1)^-$ at the preprocessing stage.
- (2) ε_s will occur when shrinkage occurs at the embedding stage.
- (3) ε_m will occur when coefficients are changed and the change results in embedding a message rather than shrinkage at the embedding stage.

It is important to note that this estimation is performed before the preprocessing starts; therefore, the estimation is based on the information obtained from the cover image. S and U are collected from the image before any processing, and N_p is calculated using the

estimation described in Section 6.2.1. The methods to estimate ε_p and ε_s are identical to the one of parity-based algorithm as described in Section 6.1.3. Estimation of ε_m of matrix-based algorithm is described because it is different from one of parity-based algorithm.

Expectation of ε_m : This error is produced when k message bits are successfully embedded with a coefficient changing in an n -sized block. The probability of $\alpha = 0$ is $\frac{n}{n+1}$. Let q be $\frac{|S|+N_p}{|U|+N_p}$, and R_u be rounding errors of the coefficients that do not belong to S .

$$\varepsilon_m = N_b \frac{n}{n+1} (1-q)(1-2E[R_u]),$$

where N_b is the number of times that the process executes the step of calculating α , which is obtained by:

$$N_b = \frac{|U|}{n} + N_p.$$

and $E[R_u]$ is obtained in a similar way to obtain $E[R_p]$ as described in Section 6.1.3.

6.2.4 Experimental Results

In this section, we demonstrate the experimental results of MM1-s on two test images shown in Figure 6.2. First, we examined the accuracy of our theoretical analysis. Figure 6.9 depicts the average embedding error per message bit caused by the MM1-s method in six different block-sizes, and the theoretically estimated values and experimental values are plotted in one graph for comparison. It shows that the theoretical predictions are very close to the actual embedding errors. The tests were accomplished with six different block-sizes, $2^k - 1$, $k = 2, \dots, 7$.

Figure 6.10 shows average embedding error per message bit using MM1-s in six different embedding rates. 100 images from UCID database were used to embed messages and the results are shown as a box plot. The result box plot shows that the distortion produced by MM1-s is very low (below 0.25).

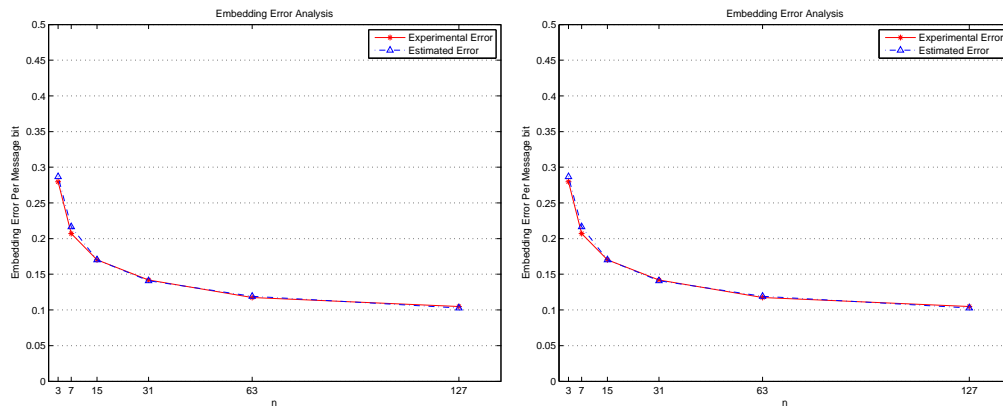


Figure 6.9: Embedding error analysis of the MM1-s method in various block sizes, $(1, n, k)$, the test images in Figure 6.2a (left) and Figure 6.2e (right). The theoretical embedding error obtained from our mathematical prediction as described in Section 6.2.3 and the experimental embedding error are very close.

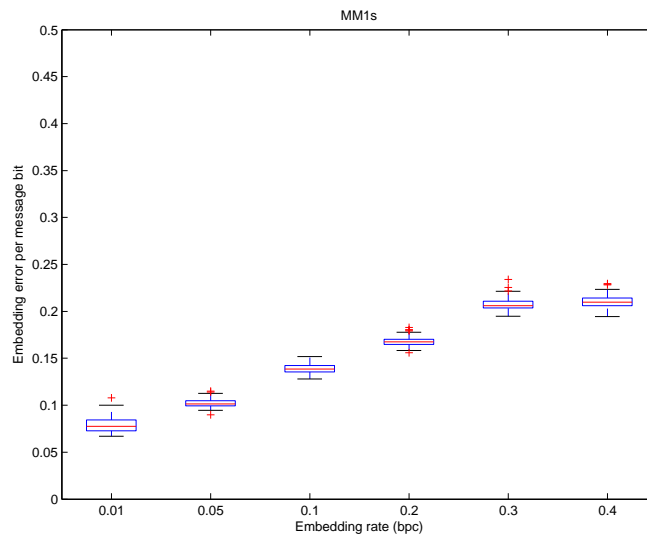


Figure 6.10: Average embedding error per message bit using the MM1-s method with six different embedding rates. 100 images from UCID database were used to embed messages.

The embedding errors caused by the MM1-s method are compared with other two embedding methods, F5 [21] and MMx (MM1) [49]. Figure 6.6 shows the comparisons of the embedding errors for three algorithms at various embedding rates. The distortions per embedding message bit are plotted in increasing embedding rates (bpc), where bpc is calculated by the total number of message bits divided by the usable coefficients (non-zero AC coefficients). MMx [49] is an algorithm that utilizes rounding errors for minimal distortion using modified matrix coding with no shrinkage. Similar to MM1, the proposed algorithm can keep low level of distortion due to embedding as shown in Figure 6.11. The distortions due to both MM1 and MM1-s are noticeably lower than the one due to F5 with the same test images. Figure 6.12 shows the coefficient histogram differences between the cover images and the stego images. The differences in the histograms are shown in the $[-10, 10]$ coefficient range and in various embedding rates: 0.4, 0.3, 0.2, and 0.1 bpc, from the top. The black bars show the differences in the histogram between the MM1-s stego image and the cover image, while the white bars show the differences in the histogram between the stego image of MMx (MM1) and the cover image. MMx proposed an exceptional rule in order to avoid shrinkage: changing coefficient 1 and -1 to coefficient 2 and -2 respectively, no matter what their corresponding rounding errors are. By applying the exception rule, MMx can avoid shrinkage but leaves a trail in the coefficient histogram, which produces a noticeable change in the frequencies of coefficient 2 and -2 .

The results indicate that the proposed shrinkage-permitting algorithm can be capable of managing the low embedding error at the similar level to MMx, as shown in Figure 6.11, as well as maintaining the statistical properties better than MMx, as shown in Figure 6.12, fulfilling one of the important requirements for secure steganographic applications.

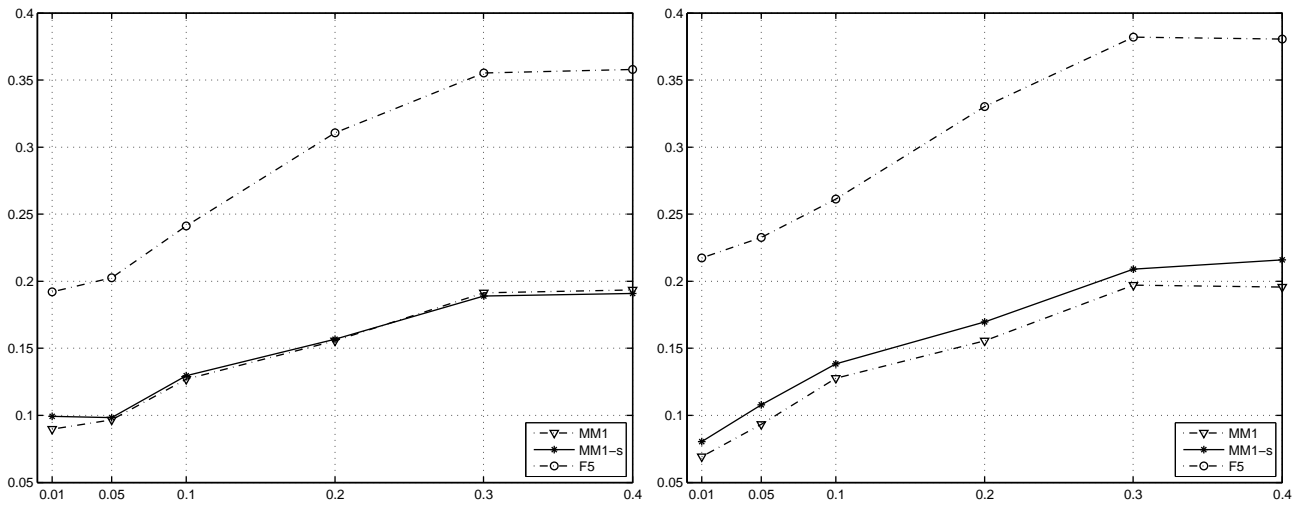


Figure 6.11: Embedding error per embedding message bit using MM1-s in various bpc ranging from 0.01 to 0.4 using the test images in Figure 6.2a (left) and Figure 6.2e (right).

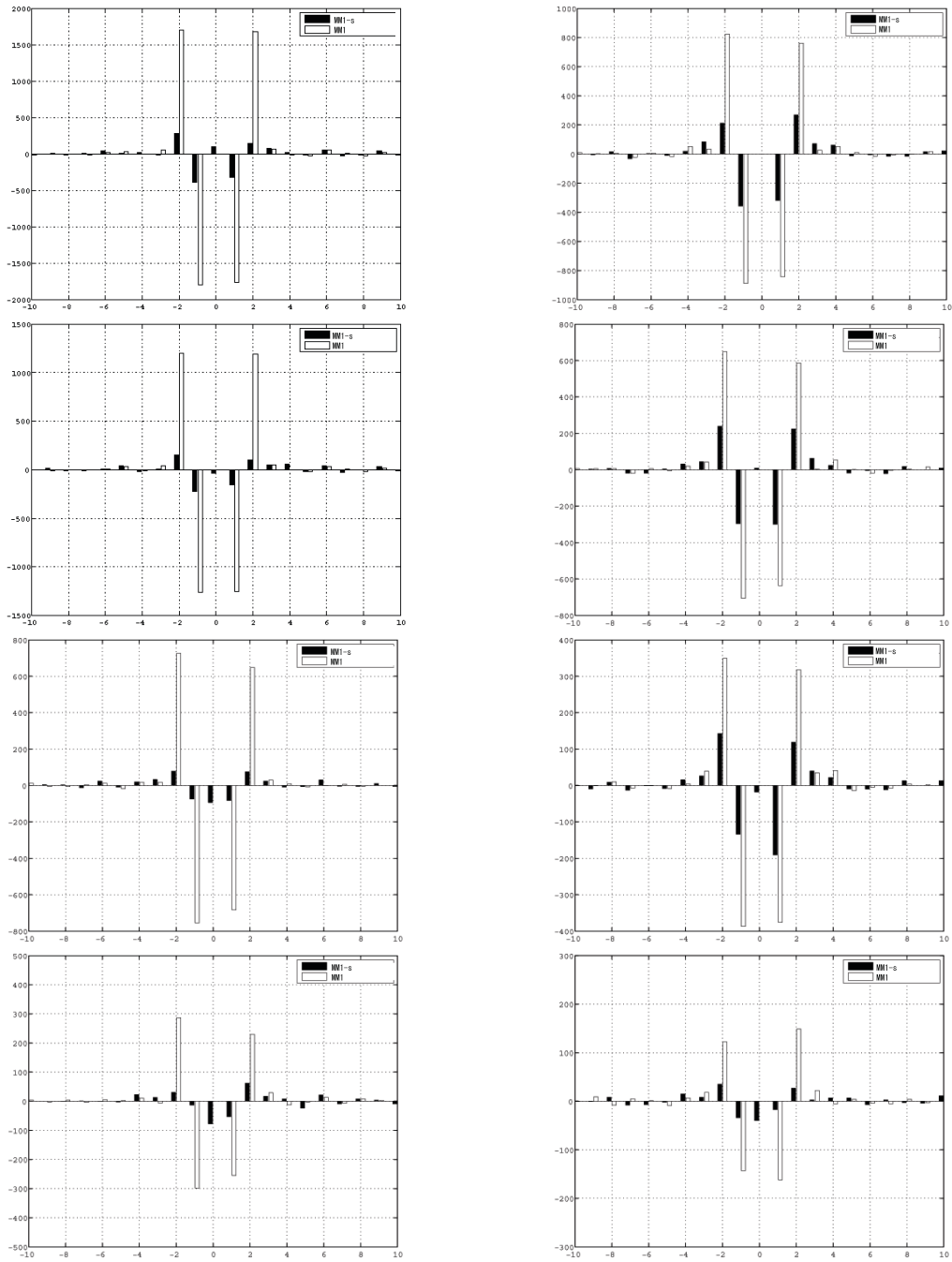


Figure 6.12: Histogram modification from the MM1-s method with comparison to MM1. The embedding rates are 0.4, 0.3, 0.2, and 0.1 from the top using the test images in Figure 6.2a (left) and Figure 6.2e (right).

Chapter 7: Preserving Statistics while Minimizing Distortion

Preserving statistical properties is one of the problems that many researchers have focused on in steganography, because a noticeable modification in statistical properties may be detected by steganalysis methods, which are interested in detecting the existence of hidden information in objects. In this chapter, we propose a parity-coding based embedding algorithm that employs weighted graph matching for preserving coefficient histograms while minimizing distortion and call it PB-g.

7.1 Graph Matching

We employ weighted graph matching from graph theory to preserve coefficient histograms while keeping distortion minimized during embedding. The central idea is to embed a message by interchanging only those coefficients whose changes do not modify the histogram and do guarantee a limited distortion. We have found that *interchangeable pairs*, which result from changing by the rounding-biased rule, are well fitted for the purpose of preserving statistics. The interchangeable pair refers to two adjacent coefficients that can be interchanged without modifying the entire coefficient histogram.

Recall that $\bar{X}(c)^+$ and $\bar{X}(c)^-$ are

$$\begin{aligned}\bar{X}(c)^+ &= \{\bar{x}_i | \bar{x}_i \in \bar{X} \text{ and } \text{Round}(\bar{x}_i) = c \text{ and } 0 \leq r(\bar{x}_i) < 0.5\}, \\ \bar{X}(c)^- &= \{\bar{x}_i | \bar{x}_i \in \bar{X} \text{ and } \text{Round}(\bar{x}_i) = c \text{ and } -0.5 \leq r(\bar{x}_i) < 0\}.\end{aligned}$$

As a result of changing by the rounding-biased rule, the coefficient that belongs to $\bar{X}(c)^+$ is always changed to the coefficient that belongs to $\bar{X}(c+1)^-$; the coefficient that belongs to $\bar{X}(c+1)^-$ is always changed to the coefficient that belongs to $\bar{X}(c)^+$. These two adjacent

coefficients can be paired as an interchangeable pair. As long as interchangeable pairs are involved in embedding, their modifications do not change the coefficient histogram while keeping low distortion by the rounding-biased rule. Given the coefficients, finding the interchangeable pairs is posed as a matching problem.

First, we need to define our problem in terms of graph matching. We embed a message using parity coding [1] with block size of n . We consider a set of n coefficients as a vertex and a connection between the paired interchangeable coefficients as an edge. By assigning a weight based on their rounding errors to each edge, we can achieve the goal of minimizing distortion using weighted graph matching. The weight of an edge is assigned as a sum of absolute values of the rounding errors of the connected coefficients. Since the added distortion when changed by the rounding-biased rule is $1 - 2|r(\bar{x})|$, choosing the largest weighted edges will then lead to minimal distortion.

The proposed algorithm begins by creating a undirect graph, $G = (V, E)$, which does not have self-loops. V is a set of n -coefficient blocks that are units of the parity-coding-based algorithm, and E is a set of edges that links interchangeable pairs. We say $H \subseteq E$ if no two edges of H are adjacent. A vertex v is *matched* if v is incident to an edge of H , otherwise it is *free*. The weight of an edge is $w(\{a, b\}) = |r(a)| + |r(b)|$. Figure 7.1 shows a simple example of the graph. Given a weight function $w : E \rightarrow \mathbb{R}$, we want to find a matching with the maximum weight.

7.1.1 Approximation Algorithm for Maximum Weighted Matching

Many algorithms [73, 74] for matching problems have been proposed, and many of them focus on improving a time complexity for maximum cardinality matching and a maximum weighted matching. However all optimal algorithms up to date have super-linear time complexity and the running time is too costly. Therefore, approximation algorithms for calculating a matching has recently drawn attentions. The approximation algorithms have a smaller time complexity and provide suboptimal solutions.

In our proposed embedding method based on parity coding, the number of vertices

and edges are quite large. For example, if the armadillo image (sized of 427×278 , see Figure 5.11) is used for a cover image, the number of vertices and edges for parity coding $n = 2$ are $|V| = 45,961$ and $|E| = 193,020,048$. The graph grows as the size of image become larger. Hence, we employ a linear-time approximation algorithm with a performance ratio of $\frac{1}{2}$ considering two approximation algorithms: GREEDY-algorithm and Linear Time Approximation algorithm.

Linear Time Approximation Algorithm (LAM)

Preis [75] developed a linear time $\frac{1}{2}$ -approximation algorithm for maximum weighted matching (LAM). LAM can achieve not only at least $\frac{1}{2}$ of the edge weight of a maximum weighted matching but also at least $\frac{1}{2}$ of the cardinality of a maximum cardinality matching in linear time. Algorithm 7 summarizes the LAM algorithm. The locally heaviest edge $\{a, b\}$ is an edge with the highest weight among all adjacent edges. After the locally heaviest edge is found, the vertices incident to a matched edge are contracted.

Algorithm 7 LAM-Algorithm

```

 $M_{LAM} := \emptyset$ 
while ( $E \neq \emptyset$ ) do
  take locally heaviest edges  $\{a, b\} \in E$ ;
  add  $\{a, b\}$  to  $M_{LAM}$ ;
  remove all edges incidents to  $a$  or  $b$  from  $E$ ;
end while

```

To find the locally heaviest edge, the LAM starts with an arbitrary edge and checks the remaining adjacent edges. As long as an adjacent edge with higher weight can be found, the algorithm switches to the new edge and repeats the checking procedure until a locally heaviest edge is reached. The locally heaviest edge is added to the matching and all incident edges to the matched vertices are removed. The algorithm terminates when there are no remaining edges.

Even though LAM runs in time $O(|E|)$, we chose GREEDY-algorithm [76] of $O(|E| \cdot \log|V|)$ time. With our implementation, the proposed algorithm runs in $O(|V| \cdot \log|V|)$

time with GREEDY-algorithm, which is better than $O|E|$, because the number of edges $|E|$ grows as $O(|V|^2)$ in our algorithm.

GREEDY-algorithm

We use the GREEDY-algorithm that can achieve at least $\frac{1}{2}$ of the edge weight of a maximum weighted matching. Avis [76] analyzed that if the edges are sorted by their weights in a preprocessing step, the GREEDY-algorithm runs in $O(|E| \cdot \log|V|)$ time. However, for our graph, it runs in $O(|V| \cdot \log|V|)$. Algorithm 8 describes the GREEDY-algorithm.

Algorithm 8 GREEDY-Algorithm

```

sort the edges in decreasing weight  $w$ ;
 $M_{GREEDY} := \emptyset$ ;
while ( $E \neq \emptyset$ ) do
    take an edge  $\{a, b\} \in E$  with highest weight;
    add  $\{a, b\}$  to  $M_{GREEDY}$ ;
    remove all edges incident to  $a$  or  $b$  from  $E$ ;
end while

```

7.1.2 Applying GREEDY-algorithm to Embed a Message

The goal of the use of weighted graph matching here is to preserve a coefficient histogram after changing coefficients' values and to keep low distortion by utilizing the rounding errors. First, how to construct a graph for the parity-coding-based embedding algorithm will be described followed by the description of how to achieve the maximum matching while minimizing distortion.

Constructing a Graph

Since the proposed method is parity coding based, we consider each block of n coefficients as a vertex in a graph. There are n coefficients of \bar{X} associated with a vertex. We make an edge between two different vertices if each of the vertices has one half of an interchangeable pair. Let $\{\bar{x}_a, \bar{x}_b\} \in E$ be an edge. Each edge has a weight which is a sum of absolute values of the rounding errors of the connected coefficients, $|r(\bar{x}_a)| + |r(\bar{x}_b)|$. Let (\bar{x}_a, \bar{x}_b) be

an interchangeable pair. Suppose that $\bar{x}_a \in \bar{X}^+(c)$ and $\bar{x}_b \in \bar{X}^-(c+1)$. If we change \bar{x}_a to $\hat{x}_a = c+1$ and \bar{x}_b to $\hat{x}_b = c$ as a result of the rounding-biased rule, the sum of embedding errors caused by those two coefficients will be $2 - 2(|r(\bar{x}_a)| + |r(\bar{x}_b)|)$. We try to minimize the errors by finding the maximum weight $w(\{\bar{x}_a, \bar{x}_b\})$ for the edge $\{\bar{x}_a, \bar{x}_b\}$.

Usable coefficients U are collected as forms of $\bar{X}(c)^+$ and $\bar{X}(c)^-$. $\{\bar{X}(c)^+, \bar{X}(c+1)^-\}$ is a set of interchangeable pairs. To make a connection with heaviest weight, $\bar{X}(c)^+$ and $\bar{X}(c+1)^-$ are sorted in decreasing order of absolute values of their rounding errors, let the sorted sets be A and B :

$$\begin{aligned} A &= a_1, a_2, \dots, a_{N_1}, \\ B &= b_1, b_2, \dots, b_{N_2}, \end{aligned}$$

where $a_i \in \bar{X}(c)^+$ and $|r(a_i)| \geq |r(a_{i+1})|$ for $1 \leq i \leq N_1 - 1$ and $b_i \in \bar{X}(c+1)^-$ and $|r(b_i)| \geq |r(b_{i+1})|$ for $1 \leq i \leq N_2 - 1$. An edge $\{a_i, b_i\} \in E$ connects two vertices v_j, v_k ($a_i \in v_j$ and $b_i \in v_k$) and has a weight $w(\{a_i, b_i\}) = |r(a_i)| + |r(b_i)|$. The number of edges created in A and B is equal to $\min\{N_1, N_2\}$. The procedure of creating edges terminates when c reaches $\max(\tilde{X})$. Now the graph is constructed and ready for a matching.

Matching

Let $H \subset E$ be a matching. Our primary goal is to maximize $|H|$ and the secondary goal is to maximize $W(H)$, which is the weight of H : $W(H) := \sum_{\{a,b\} \in H} w(\{a,b\})$. Since distortion due to embedding can be written in terms of the rounding errors as mentioned in the previous section, in order to try minimize the total errors due to embedding, the highest weighted edges are added to a matching. We use the GREEDY-algorithm to find the heaviest edges.

Figure 7.2 a simple example to show how to find a matching using a block size of 2 given by the graph of Figure 7.1. Final modification from \bar{X} to \hat{X} is also shown in the bottom of Figure 7.2, and as a result of \hat{X} , four message bits have been embedded.

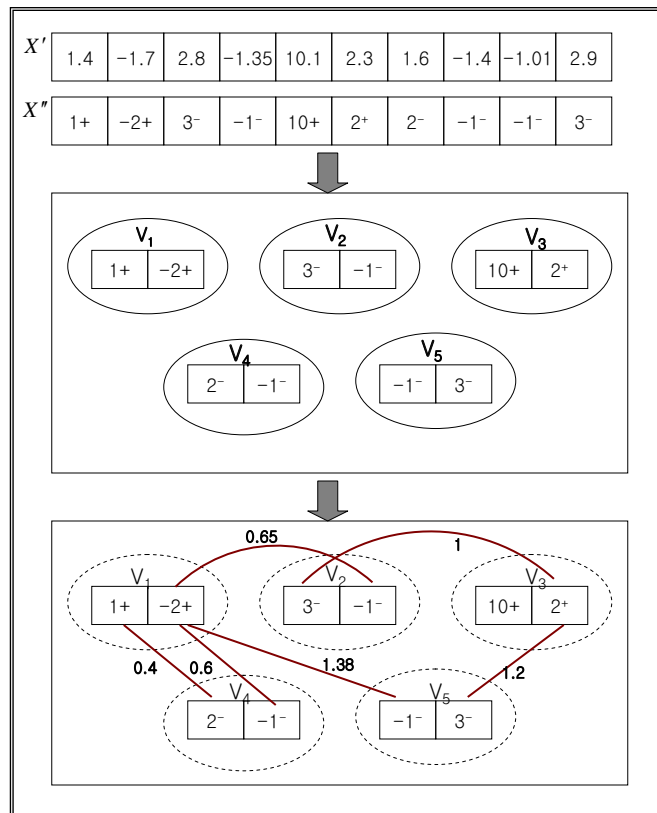


Figure 7.1: Constructing a graph based on parity coding for the block size of 2. Each block is a vertex. The interchangeable pair is linked in an edge. Each edge has a weight based on the rounding errors of the coefficients.

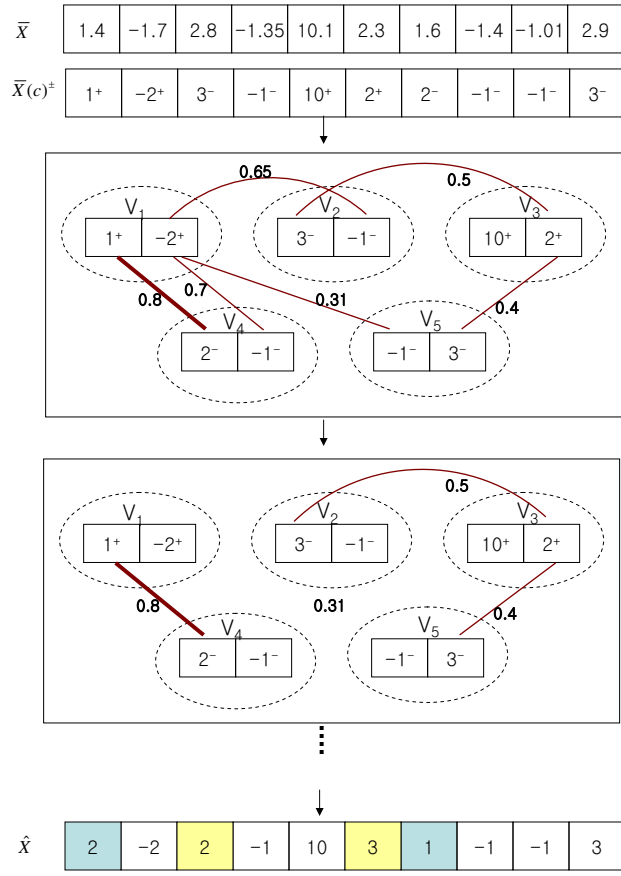


Figure 7.2: Example of finding a highest weighted edge using the GREEDY-algorithm. Choose the heaviest edge and remove all edges incident to the chosen vertices. Move the edge to the matching H . In this figure, only the first matching is marked with in the thicker line, but when the GREEDY-algorithm procedure finishes, one more matching will add to H for this example.

7.2 Embedding Algorithm

The parity coding based embedding algorithm that employs weighted graph matching is described (PB-g). Data structures and detailed steps of the procedure for the implementation is described in this section. For achieving low distortion, any change in the cover coefficient value follows the rounding-biased rule. The graph matching is applied for embedding in order to avoid a substantial modification in a histogram. Unlike other typical embedding algorithms for JPEG images, PB-g uses all AC coefficients including 0. PB-g has three phases. In the first phase, the block parity is compared with the corresponding message bit. In the second phase, the GREEDY-algorithm procedure is activated to find a matching for embedding. In the third phase, embedding is completed by applying the minimization scheme to the unresolved blocks, i.e., those that need to be modified but were unmatched.

1st Phase: Checking Parities

The embedding process compares a parity of each block with the corresponding message bit. The blocks whose parities are the same with the corresponding message bits are designated Type 1 and kept unchanged. Those blocks automatically become message-containing blocks without any changes. On the other hand, the remaining blocks are designated as Type 2 and collected for the next phase embedding.

2nd Phase: Embedding with a Matching

Each Type 2 block is considered as a vertex, and each interchangeable pair are connected as an edge if each half of the pair is in the different vertex. The sum of the absolute values of the connected coefficients' rounding errors weighs the edge. The detailed description on constructing the graph is explained in the previous section.

The following is the detailed description of the implementation. We use data structures that allow us to find the heaviest edges efficiently. In our implementation, two data structure are constructed before matching has started: a *coefficients-by-block* array, A , and a *coefficients-by-value* array, B . A stores all coefficients (in the form of \bar{x}) of Type 2 blocks

such that $A(i, j)$ denotes j^{th} coefficient of the i^{th} block. B contains the same coefficients of A but in a different arrangement. The coefficients in B are arranged by their after-rounding-values and the signs of their rounding errors. For each value of c , there are two separate columns, one is for $\bar{X}(c)^-$ and the other is for $\bar{X}(c)^+$. The coefficients in the same column are sorted in descending order of their absolute value of rounding errors. The constructing of the two arrays in our implementation is illustrated in Figure 7.3 using the same example of Figure 7.2. The interchangeable pairs $\{\bar{X}(c)^+, \bar{X}(c+1)^-\}$ are laid in adjacent columns of this array. Each element of B is filled with the coefficients having the corresponding c and sign from A . In other words, all coefficients located in the same column have the same after-rounding-values and the same sign of the rounding error. Technically, elements of B contains references to the coefficients in A . With the element of B , we can retrieve index (i, j) in A , therefore we can retrieve the coefficient value as well as the information on the block that the referenced coefficient belongs to. Sorting each column of B by the corresponding absolute value of the rounding error in descending order allows us to find the heaviest edges easily. In our implementation, the GREEDY-algorithm starts with the arbitrary edge in the top row of the adjacent columns in B and then checks the adjacent edges by referencing A . If other adjacent edge with higher weight is found, the GREEDY-algorithm switches to the new edge until no adjacent edge has higher weight than the found edge. Using the simple example of Figure 7.2, two coefficients connected with the heaviest edge are marked as a circle in Figure 7.3.

Once the heaviest edge is found, the corresponding two coefficients are modified based on the rounding-biased rule. Embedding is completed for those two blocks. Before searching for another heaviest edge, the arrays A and B are updated by removing the resolved blocks as shown in Figure 7.4. The following three steps are repeated until interchangeable pairs cannot be found anymore in B : searching for the heaviest edge, changing the values of the matched coefficients, and updating the arrays. Figures 7.3 and 7.4 illustrate the steps of the second phase of our algorithm.

3rd Phase: Embedding Based on the Rounding-Biased Rule

After the second phase of embedding have finished, there are some blocks that still remain unresolved. For each such block, a message bit is embedded by choosing a coefficient that will cause the least error and changing it based on the rounding-biased rule.

Suppose $A_i = \{A(i, 1), A(i, 2), \dots, A(i, n)\}$ is an unresolved block. To embed a message bit, the j^{th} ($j \in \{1, \dots, n\}$) coefficient of A_i needs to be changed so that the LSB's of the block after embedding \hat{A}_i would have parity equal to the message bit. Now, the question is which j should be changed. We are interested in minimizing distortion, therefore, $\varepsilon_j = \min_{1 \leq k \leq n} \{1 - 2|r(A(i, k))|\}$ is examined, and the corresponding coefficient that has ε_j will be chosen to be changed as \hat{x}_k , where \bar{x}_k is the coefficient corresponding to ε_j . The modification would be made using the rounding-biased rule.

In the next section, the PB-g algorithm will be evaluated in terms of the level of distortion and modification of the histogram.

7.3 Error Analysis

Since the PB-g method focuses on finding the interchangeable pairs maximally, accurate error analysis based on rounding error distribution is not possible. However, we performed error analysis for showing the limits of distortion due to the PB-g method.

Error analysis for the PB-g method is only applied to those coefficients that are changed during 3rd phase of PB-g. An additional error due to changing a coefficient is given by

$$\varepsilon_j = 1 - 2|r_j|. \quad (7.1)$$

Since we use all AC coefficients for embedding, the R is formed from all AC coefficients. We have assumed that r is an i.i.d. random variable and that its probability density $f_R(x)$ is known. $f_R(x)$ is obtained by normalizing the rounding error histograms. The probability

coefficients-by-block array: A

1.4	-1.7
2.8	-1.35
10.1	2.3
1.6	-1.4
-1.01	2.9

coefficients-by-value array: B

...	-2 ⁺	-1 ⁻	...	1 ⁺	2 ⁻	2 ⁺	3 ⁻	...
	(1,2)	(4,4)		(1,1)	(4,1)	(3,2)	(2,1)	
		(2,2)					(5,2)	
		(5,1)						

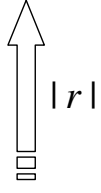


Figure 7.3: Two data structures (A and B) used in our implementation. A stores all coefficients (in the form of \bar{x}) of Type 2 blocks. B contains the same coefficients of A , but they are arranged by their after-rounding-values and the signs of their rounding errors. Each column of B is sorted by the corresponding absolute values of rounding errors in descending order.

coefficients-by-block array: A

4	-1.7
2.8	-1.35
10.1	2.3
1.6	-1.4
-1.01	2.9

coefficients-by-value array: B

...	-2⁺	-1⁻	...	1⁺	2⁻	2⁺	3⁻	...
	(1,2)	(4,4)		(1,1)	(4,1)	(3,2)	(2,1)	
		(2,2)					(5,2)	
		(5,1)						




Figure 7.4: The blocks that have made changes are marked as resolved. All coefficients corresponding to the resolved blocks are removed from both arrays.

distribution for $\psi = |r|$ is given by

$$F_\psi(x) = \int_{-x}^x f_R(x)dx, \quad x \in [0, 0.5].$$

Hence, the probability distribution for $\nu = 1 - 2\psi$ is given by

$$F_\nu(x) = 1 - F_\psi\left(\frac{1-x}{2}\right), \quad x \in [0, 1]. \quad (7.2)$$

Given the probability distribution $F_\nu(x)$ the minimum additional error due to embedding is expressed by

$$\mu = \min_{1 \leq j \leq n} \{\varepsilon_j\}.$$

The distribution of μ is given by

$$F_\mu(x) = P\{\mu \leq x\} = 1 - (1 - F_\nu(x))^n, \quad n \geq 1, \quad (7.3)$$

where $F_\nu(x)$ is given by Eq. (7.2). Finally, the expected value of μ will then be

$$E[\mu] = \int_0^\infty x dF_\mu(x). \quad (7.4)$$

Figure 7.5 illustrates a comparison of the predicted errors due to embedding using PB-g to the real experimental errors. As Figure 7.5 shows, the theoretical error analysis for the PB-g method is not as accurate as our other algorithms focusing on minimizing distortion. However, the above analysis shows the upper bound on distortion by the PB-g method.

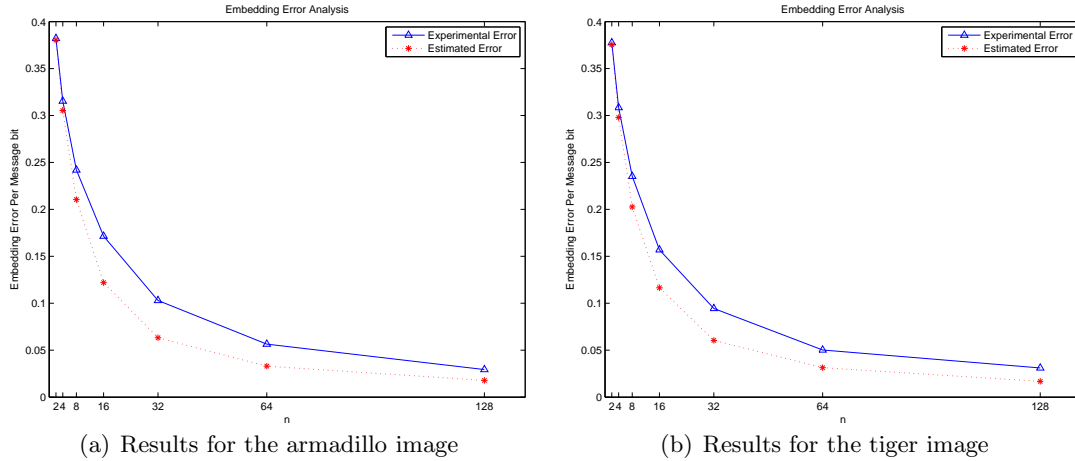


Figure 7.5: Error analysis of the PB-g method in various block size n using the test images in Figure 5.11a (left) and Figure 5.11b (right). The graphs show comparison of the theoretical error to the experimental error.

7.4 Experimental Results

In this section, we demonstrate the experimental results of the PB-g method. The embedding rate bpc shown in the figures is measured by dividing total number of message bits by the number of non-zero AC coefficients. (Even though our algorithm uses all AC coefficients, the measurement based on the number of non-zero AC coefficients is used for a comparison).

Matching Cardinality

To investigate the cardinalities of the matching, we calculated the ratios of resolved blocks by matching among Type 2 blocks. As previously denoted, Type 2 block is a block that needs to be modified for embedding. Figure 7.7 shows the cardinalities of the matching in a box plot using the images shown in Figure 7.6 are used. For the embedding rate up to 0.02 bpc, more than 96% of Type 2 blocks succeed in embedding by matching. In other words, up to 0.02 bpc, more than 99% of embedded message bits are embedded during the 1st and 2nd phases of the PB-g method without modifying the histogram. For the high embedding rate of 0.4, 92% to 93% of embedded message bits are embedded without modifying the

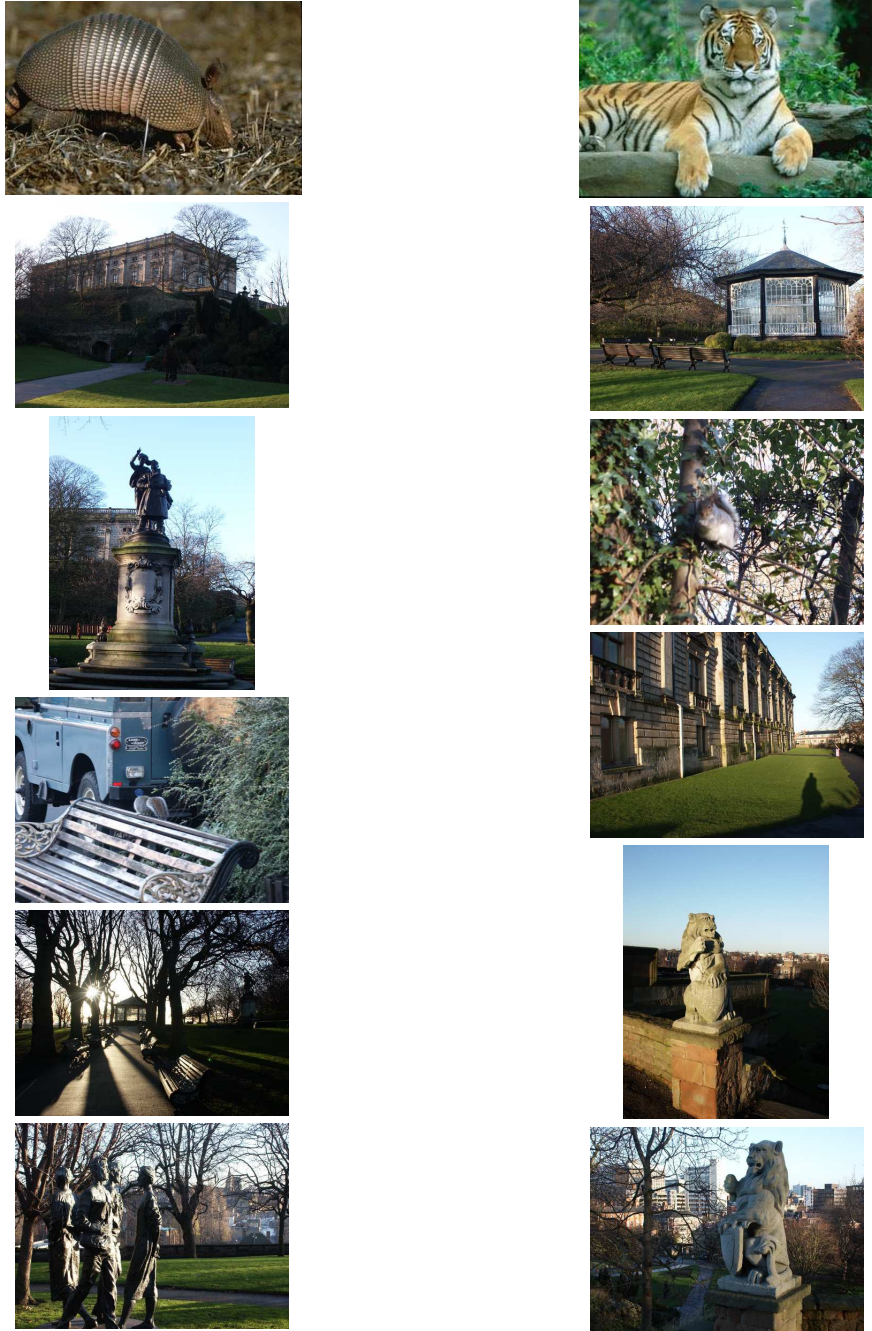


Figure 7.6: Examples of test images

histogram in our experiments.

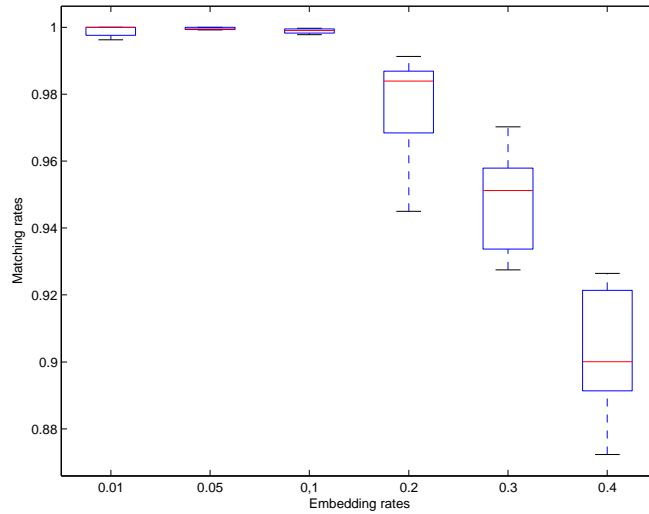


Figure 7.7: Matching rates for six different embedding rates in bpc are shown in a box plot graph. Test images are shown in Figure 7.6.

Distortion

Figure 7.8 shows the average embedding error per message bit caused by PB-g. 100 images from UCID database were used to embed messages at six different embedding rates and the results are shown as a box plot. The result box plot shows that the distortion levels are significantly lowered in PB-g compared to other methods.

Coefficient Histograms

Figures 7.9 and 7.10 show the level of histogram modification caused by the PB-g method compared to PB and F5. The differences of the histograms between the stego images and the cover images are measured in the chi-square distance and the Euclidean distance, respectively. If the distance is small, the method does not change the histogram much during embedding. Note that the histogram modification from the PB-g method is significantly lower compared to the other two methods. The measured difference of PB in the 0.4 bpc is even higher than the ones of F5 as shown in Figure 7.9, which means that PB with the 0.4 bpc changes the histogram larger than F5 does. PB used the modified-rounding-biased rule

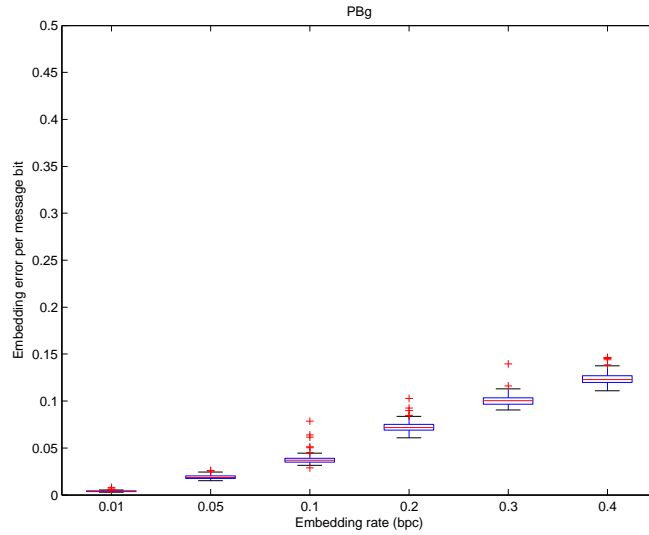
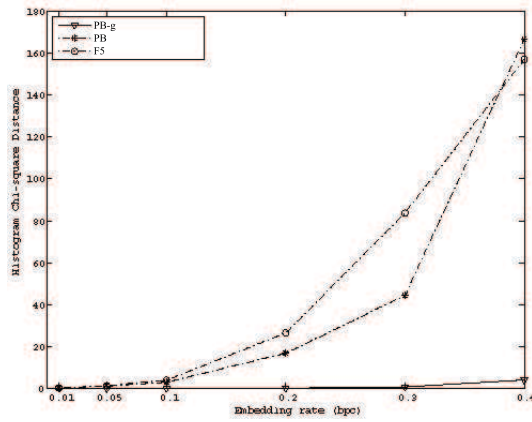


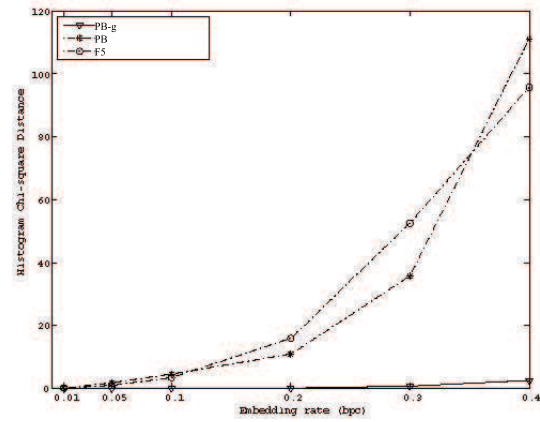
Figure 7.8: Average embedding error per message bit using the PB-g method with six different embedding rates in bpc. 100 images from UCID database were used to embed messages

in order to avoid shrinkage. Because of the modified rule, PB leaves a trail in the coefficient histogram—changing the frequencies for the specific coefficients in a predictable way.

We found that the PB-g algorithm can be capable of not only managing very low embedding error, but also maintaining the statistical properties very well as shown in Figures 7.9 and 7.10, which is one of the important requirements for secure steganographic applications.

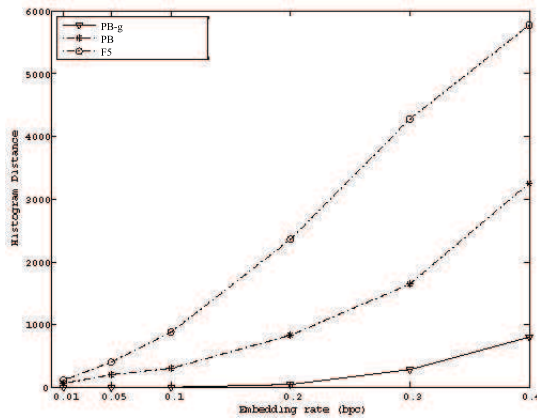


(a) Results for the armadillo image

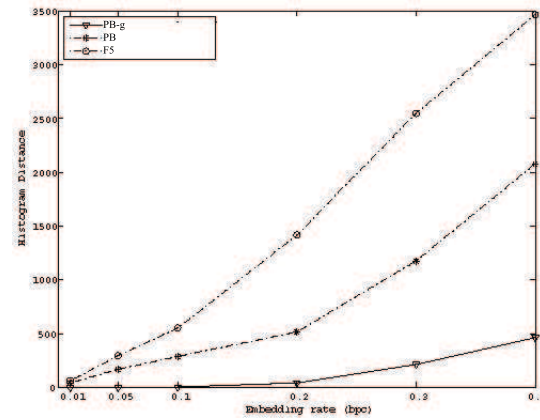


(b) Results for for the tiger image

Figure 7.9: The Chi-square distance between the histogram of a cover image and the histogram of a stego image. Two embedding methods are compared with the PB-g method at the various embedding rates (bpc) using the test images in Figure 5.11a (left) and Figure 5.11b (right).



(a) Results for the armadillo image



(b) Results for the tiger image

Figure 7.10: The Euclidean distance between the histogram of a cover image and the histogram of a stego image. Two embedding methods are compared with the PB-g method in the various embedding rates (bpc) in the various embedding rates (bpc) using the test images in Figure 5.11a (left) and Figure 5.11b (right).

Chapter 8: Algorithm Achieving Absolute Lower Bounds on Distortion

In this chapter, we propose an embedding method that increases the usable JPEG coefficients for embedding by including the zero-valued coefficients. The method will be described using two coding techniques, parity coding and matrix coding, and called PB-z and MMx-z respectively. The increase in usable coefficients leads to an increase of the block size in block-based embedding algorithms, which ultimately results in a decrease in distortion with a given message length for embedding.

8.1 Parity Coding: PB-z

In this section, we proposed an embedding method that uses all AC coefficients including zero-valued coefficients for embedding. We call this method PB-z (using all AC coefficients with parity coding).

8.1.1 Embedding Algorithm

Let I be a color image and each color channel of I is divided into 8×8 blocks. While DCT is being done on the blocks, all zigzag-ordered AC coefficients including 0 before and after rounding are collected in a set \bar{X} and \tilde{X} , respectively. At this point, \bar{X} can be rearranged for security reasons in a random order, which is determined by a secret key. The PB-z method embeds a message using parity coding, n coefficients from \bar{X} are used for embedding a single message bit.

Algorithm 9 describes the detailed steps that the embedding process follows to embed a single message bit. First, it collects the next n available coefficients to form $Y =$

$\{y_1, y_2, \dots, y_n\}$. Second, it checks the parity of $LSB(Round(Y))$ (a set of the least significant bits of rounded Y), and if the parity matches the message bit, the message bit is successfully embedded with no change. Otherwise, the method will examine the embedding error for each of n coefficients. It will choose the coefficient that is expected to cause the smallest embedding error and change it using the rounding-biased rule. In Algorithm 9, let \bar{x}_k and \hat{x}_k be the coefficient corresponding to y_j in \bar{X} and \hat{X} , respectively.

Algorithm 9 Embedding a message bit in a block using PB-z.

$Y \leftarrow$ next n available coefficients from \bar{X} .

if $Parity(LSB(Round(Y))) \neq m$ **then**
 find j such that $|r(y_j)| = \max_{1 \leq k \leq n} |r(y_k)|$;
 find \bar{x}_k that is corresponding to y_j ;
 produce \hat{x}_k by the rounding-biased rule;

end if

The message extracting process is straight forward; it inverts the embedding process. When the stego image is sent to a receiver in a JPEG format, coefficients of the image are extracted. If the embedding process shuffled the coefficients before embedding for security purposes, the decoded coefficients also need to be processed in the same order that the embedding process used. This can be accomplished by using a secret key.

To extract the message correctly in the parity-coding-based embedding method, the block size used for embedding should be known to the extracting process. As a way to communicate the block size between a sender and a receiver, we used a 4-byte-sized header: 2 bytes for the block size and 2 bytes for the embedded message length in bytes. To extract the block size and the length information, LSBs of the first 32 coefficients are examined. Once the block size n is known to the extracting process, each block of n coefficients is taken to extract the messages. The LSBs of the coefficients are extracted and the parity of the LSBs of each block is the corresponding message bit.

8.1.2 Error Analysis

When a sender decides the length of a message to embed, they may be interested in the amount of distortion caused by the length of the message. The analysis described in this section accurately predicts distortion due to embedding and therefore can be used to predict the impact of the embedded message on the cover image before actual embedding. Because we change a coefficient using the rounding-biased rule, the additional error due to changing a coefficient is determined by its corresponding rounding error:

$$\varepsilon_j = 1 - 2|r_j|. \quad (8.1)$$

We need to estimate the probability distribution for $1 - 2|r|$ using the rounding errors given to an embedder. By normalizing the histogram of R , the embedder can estimate its probability density $f_R(x)$. The probability distribution for $\psi = |r|$ is given by

$$F_\psi(x) = \int_{-x}^x f_R(x)dx, \quad x \in [0, 0.5]. \quad (8.2)$$

Hence, the probability distribution for $\nu = 1 - 2\psi$ is given by

$$F_\nu(x) = 1 - F_\psi\left(\frac{1-x}{2}\right), \quad x \in [0, 1]. \quad (8.3)$$

The additional distortion due to embedding is

$$\mu = \min_{1 \leq j \leq n} \{\varepsilon_j\}.$$

We use order statistic to obtain the distribution of μ , which is given by

$$F_\mu(x) = P\{\mu \leq x\} = 1 - (1 - F_\nu(x))^n, \quad n \geq 1, \quad (8.4)$$

where $F_\nu(x)$ is given by Eq. (8.3). Finally, the expected value of μ will then be

$$E[\mu] = \int_0^\infty x dF_\mu(x). \quad (8.5)$$

Figure 8.1 shows the theoretical error analysis results for PB-z. Our theoretical analysis performed for various block sizes. The embedding rate is determined by the block size. The figure illustrates a comparison of the predicted errors due to embedding to the real experimental data. The graphs show that the actual errors are very close to those theoretically predicted for various block sizes for PB-z.

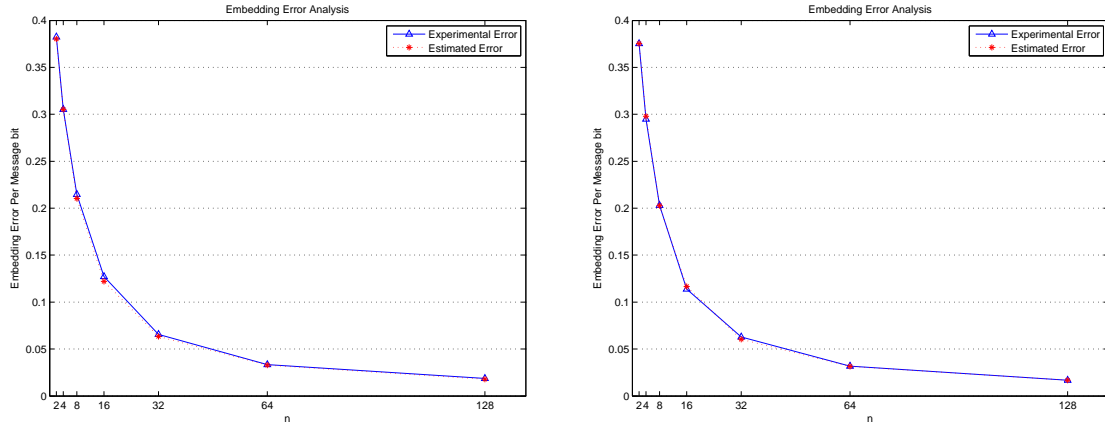


Figure 8.1: Error analysis of PB-z for various block size n using the test images in Figure 8.2a (left) and Figure 8.2b (right). Comparison of the theoretical error to the experimental error is illustrated in the graph.

8.1.3 Experimental Results

The results of experiments using the images shown in Figure 8.2 are presented. The images are color JPEG images and their sizes are 427×278 and 330×222 . The quality factor of the JPEG encoder is set to 80 in our methods. The normalized histograms of rounding error for usable coefficients are shown in the bottom of Figure 8.2. It is important to note that their distributions are different from the ones of non-zero AC coefficients, which are typically usable coefficients in other embedding algorithms. The histograms for coefficients including

0 show that there are many coefficients whose rounding errors are close to 0. Recall that the coefficient whose rounding error is close to 0 produces the larger distortion than the coefficient whose rounding error is close to 0.5 or -0.5.

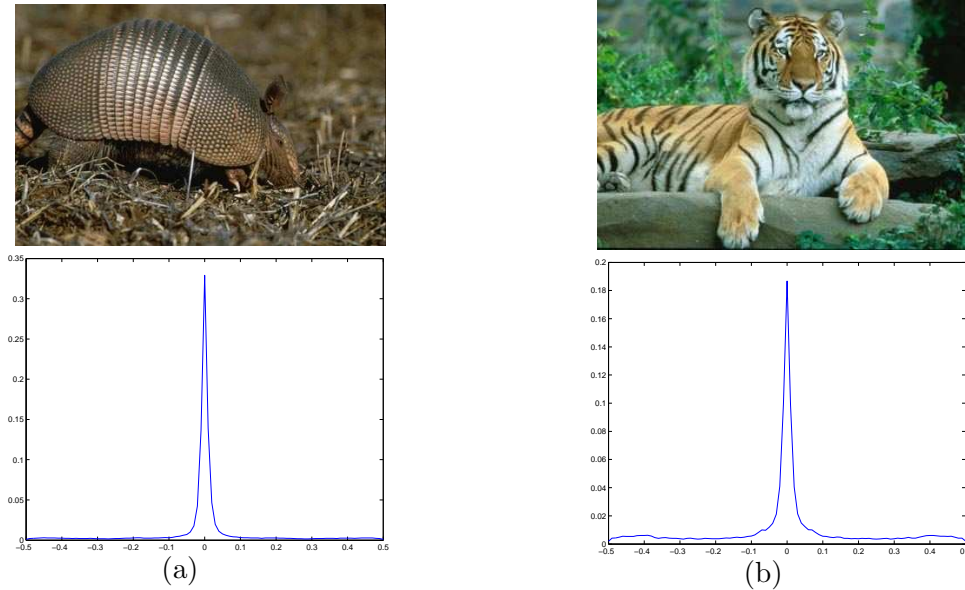


Figure 8.2: Test images used to illustrate PB-z and the normalized histograms of their rounding errors of AC coefficients including zeros.

Distortion

Figure 8.3 presents comparisons of distortion due to embedding for three methods: PB-z, PB and F5; PB-z shows the least error among them. The result box plot demonstrates that the distortion due to embedding by PB-z is very low for all embedding rates. Figure 8.4 shows the average embedding error per message bit caused by PB-z. 100 images from UCID database were used to embed messages at six different embedding rates and the results are shown as a box plot.

Accuracy of Analysis

In order to verify our analysis on PB-z, we estimated the expected values of embedding errors from 100 images and compared them to experimentally obtained embedding errors. Messages were embedded in six different embedding rates. Figure 8.5 shows the absolute

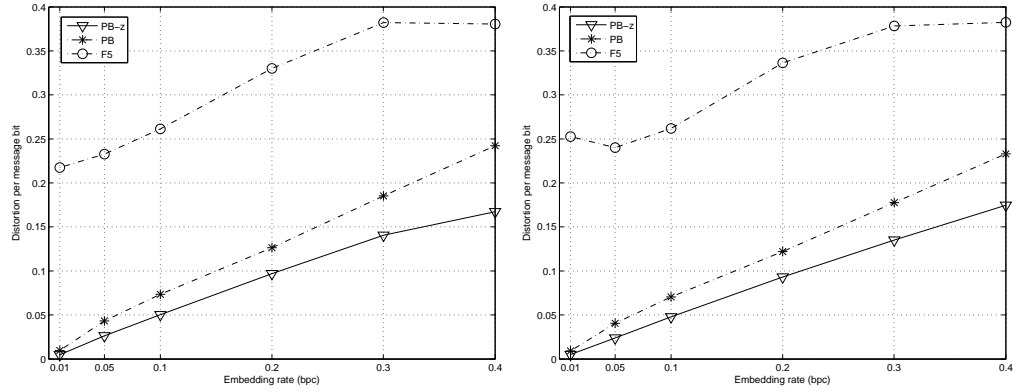


Figure 8.3: The average distortions caused by PB-z, PB, and F5 in various embedding rates (bpc) using the test images in Figure 8.2a (left) and Figure 8.2b (right).

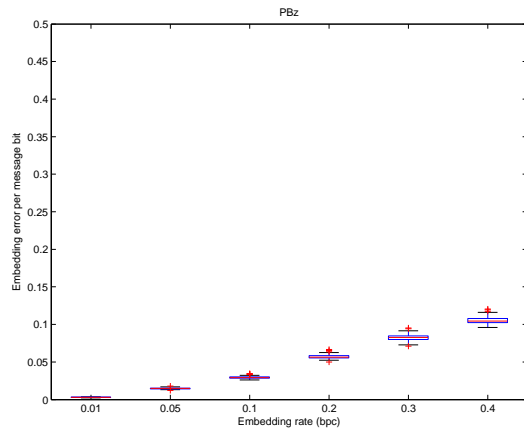


Figure 8.4: Average embedding error per message bit using the PB-z method with six different embedding rates in bpc. 100 images from UCID database were used to embed messages

value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors and their results present that our analysis is very accurate. The difference between two error are mostly below 0.005 for all embedding rates as shown in the box plots.

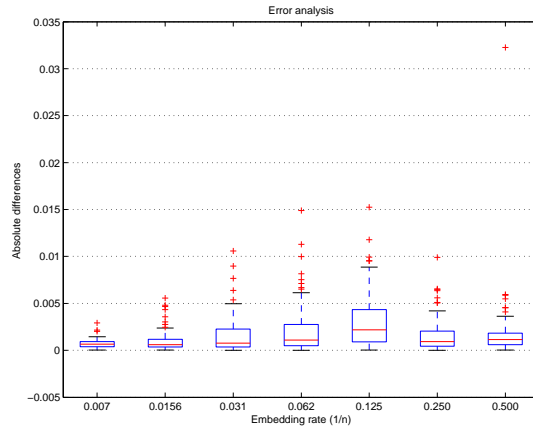


Figure 8.5: Accuracy of the error analysis on PB-z. The absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors using 100 images from UCID database.

Coefficients Statistics

To demonstrate the histogram difference numerically, we measured the Chi-square and Euclidean distances of JPEG coefficient histograms between the cover images and their stego images. For the parity based algorithms, Figures 8.6 and 8.7 illustrate the Chi-square distances and absolute distances between the stego and cover image histograms, respectively. We compared three encoding algorithms: F5, PB (the parity-based-algorithm using non-zero AC coefficients), and PB-z (the parity-based-algorithm using all AC coefficients). For most embedding rates, the distance for PB-z is the smallest, which indicates that PB-z keeps the histogram closer to the original than the other algorithms.

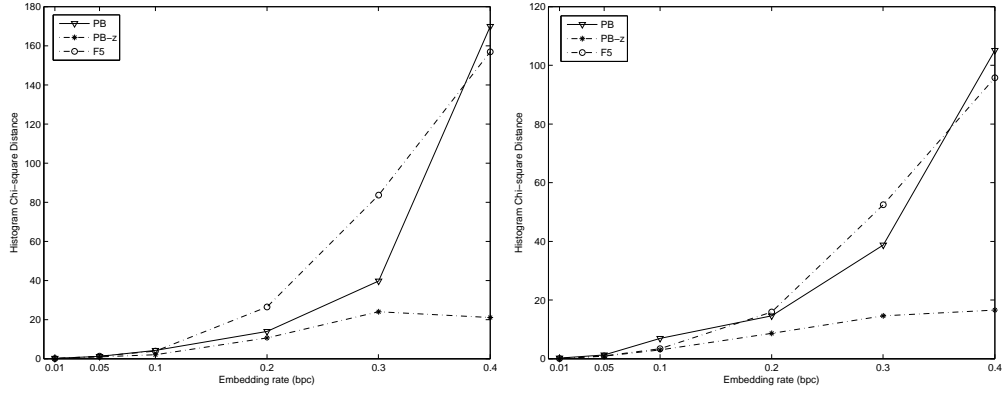


Figure 8.6: The Chi-square distance of PB-z in various bpc using the test images in Figure 8.2a (left) and Figure 8.2b (right).

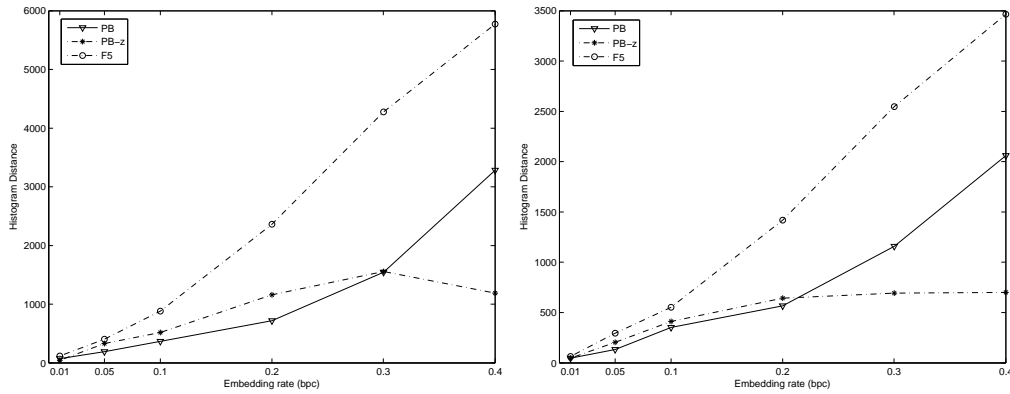


Figure 8.7: The Euclidean distance of PB-z in various bpc using the test images in Figure 8.2a (left) and Figure 8.2b (right).

8.2 Matrix Coding: MMx-z

Since the method proposed in this section uses zero-valued coefficients for embedding and modified matrix coding, we call this method MMx-z. It is obvious that MMx-z can be described as the same manner of MMx. Readers who have read the MMx description in Section 4.3 can skip this section and go to 8.2.3. A full description including error analysis for the MMx-z method are provided in this section for a self contained purpose.

The notation $(1, n, k)$ of the original matrix coding, where $n = 2^k - 1$, denotes embedding k message bits in an n -bit-sized block by changing a single bit of it. For a simple explanation, the size of available coefficients (\bar{X}) is assumed to be n and the size of the message (M) is k . The least significant bits of \bar{X} are extracted to form L : $L = LSB(Round(\bar{X}))$. A function b needs to be defined in matrix coding:

$$b(L) = \bigoplus_{i=1}^n (l_i) \cdot i. \quad (8.6)$$

To calculate the bit position α that needs to be changed for embedding M , we use

$$\alpha = M \oplus b(L). \quad (8.7)$$

If $\alpha \neq 0$, then the bit position α in the block L should be flipped, 1 to 0 or 0 to 1, which means that α^{th} coefficient of \bar{X} should be changed.

On the decoder's side, k message bits are obtained from an n -bit-sized stego data by computing the following:

$$M = b(\hat{X}). \quad (8.8)$$

Since a bit position to change is determined in matrix coding, utilizing our minimizing distortion scheme is more constrained than when using parity coding. We proposed modified matrix coding to increase the number of possible changes in each block. The modified matrix coding use (t, n, k) codes. We change 1 to t bits to embed k -bit message in an n -sized-block.

We describe the MMx-z embedding algorithm using the MM2-z example.

8.2.1 Embedding Algorithm

For an n -size-block, the number of bit-position pairs (β, γ) such that $\beta \oplus \gamma = \alpha$ is $\frac{n-1}{2}$. First, we compute α using Eq. (8.7) as original matrix coding does. We then compute the pairs $(\beta_1, \gamma_1), \dots, (\beta_h, \gamma_h)$ such that $\beta_i \oplus \gamma_i = \alpha$, where $h = \frac{n-1}{2}$.

The embedding error by changing the α -positioned coefficient is given by $\varepsilon_0 = 1 - 2|r_\alpha|$ (see Eq. (8.1)). For each of the pairs (β_i, γ_i) , the embedding error is given by $\varepsilon_i = 2 - 2(|r_{\beta_i}| + |r_{\gamma_i}|)$. In order to decide how to create \hat{X} , we find

$$\mu = \min_j \{\varepsilon_j\}, \quad 0 \leq j \leq h. \quad (8.9)$$

Given μ , we compute \hat{X} by

$$\hat{X} = \begin{cases} \tilde{X}, & \text{if } \alpha = 0 \\ \tilde{x}_1, \dots, \hat{x}_\alpha, \dots, \tilde{x}_n, & \text{if } \mu = \varepsilon_0 \\ \tilde{x}_1, \dots, \hat{x}_{\beta_i}, \dots, \hat{x}_{\gamma_i}, \dots, \tilde{x}_n, & \text{if } \mu = \varepsilon_i, \quad i = 1, \dots, h. \end{cases} \quad (8.10)$$

Algorithm 10 Embedding k message bits in a block using MM2-z.

$Y \leftarrow$ next n available coefficients from \bar{X} .
 $A \leftarrow$ next k available message bits from M .
 $Y' \leftarrow LSB(Round(Y))$.
 $\alpha = A \oplus b(Y')$ using Eq. (8.7).
if $\alpha \neq 0$ **then**
 $\varepsilon_0 = 1 - 2|r_\alpha|$;
 find pairs $\beta_i \oplus \gamma_i = \alpha$ for $i = 1, \dots, \frac{n-1}{2}$;
 compute $\varepsilon_i = 2 - 2(|r_{\beta_i}| + |r_{\gamma_i}|)$ for $i = 1, \dots, \frac{n-1}{2}$;
 find j such that $\varepsilon_j = \min_{0 \leq k \leq \frac{n-1}{2}} \varepsilon_k$;
 if $j = 0$ **then**
 find \bar{x}_k corresponding to y_α ;
 produce \hat{x}_k by the rounding-biased rule;
 else
 find \bar{x}_k, \bar{x}_h corresponding to $y_{\beta_j}, y_{\gamma_j}$;
 produce \hat{x}_k, \hat{x}_h by the rounding-biased rule;
 end if
end if

8.2.2 Error Analysis

We know the histogram of rounding errors R and predict their probability density $f_R(x)$ from the histogram. In MMx-z, R is a set of rounding errors of all AC coefficients. Probability distribution for $y = |r|$ is given by

$$F_y(x) = \int_{-x}^x f_R(x) dx, \quad y \in [0, 0.5].$$

The probability density for $z = |r_1| + |r_2|$ is given by

$$f_z(x) = f_y(x) \otimes f_y(x), \quad z \in [0, 1],$$

where \otimes stands for convolution. The probability distribution is given by

$$F_z(x) = \int_0^z f_z(x)dx, \quad z \in [0, 1].$$

The probability distribution for $\nu = 1 - 2y$ is given by

$$F_\nu(x) = 1 - F_y\left(\frac{1-x}{2}\right), \quad \nu \in [0, 1]. \quad (8.11)$$

The probability distribution for $\omega = 2 - 2z$ is given by

$$F_\omega(x) = 1 - F_z(2-x), \quad \omega \in [0, 2]. \quad (8.12)$$

The embedding error due to the change of x_α will follow the distribution of $F_\nu(x)$ and the changes of x_β and x_γ will follow the distribution of $F_\omega(x)$.

Now, we need to obtain distribution of the smallest error, which is μ in Eq. (8.9). The distribution of μ is given by

$$F_\mu(x) = P\{\mu \leq x\} = 1 - (1 - F_\nu(x))(1 - F_\omega(x))^h, \quad (8.13)$$

where $F_\nu(x)$ and $F_\omega(x)$ are given by Eq. (8.11) and (8.12). Finally, the expected value of μ will then be

$$E[\mu] = \int_0^\infty x dF_\mu(x). \quad (8.14)$$

Since there is no change when $\alpha = 0$, the expected embedding error per k message bits is given by

$$E[\mu] \times \frac{n}{n+1},$$

where $\frac{n}{n+1}$ is a probability of $\alpha \neq 0$.

Figure 8.8 shows the theoretical error analysis results for MM2-z. Our theoretical analysis performed for various block sizes. The embedding rate is determined by the block size. The figure illustrates a comparison of the predicted errors due to embedding to the real experimental data. The graphs show that the actual errors are very close to those theoretically predicted for various block sizes for MM2-z.

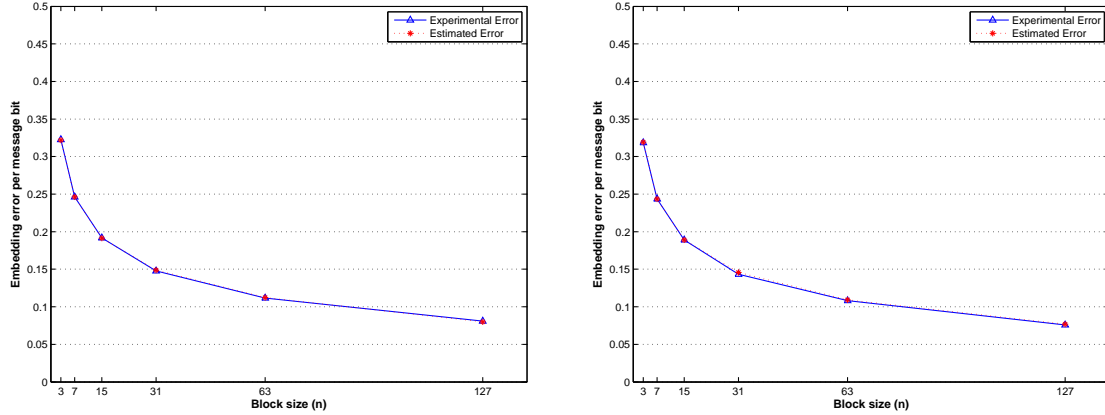


Figure 8.8: Error analysis of MM2-z for various block size n using the test images in Figure 8.2a (left) and Figure 8.2b (right). Comparison of the theoretical error to the experimental error is illustrated in the graph.

8.2.3 Experimental Results

Distortion

As given in Section 4.1, distortion is defined as a sum of distance between the original coefficients before rounding and the coefficients after embedding, which are integers. Let distortion due to embedding be $D_e = \sum |\bar{x} - \hat{x}| - \sum |r|$. For comparison, we calculate the average distortion due to embedding per message bit divided by the embedded message size:

$$\frac{D_e}{|M|}.$$

The distortion MMx-z method is presented using the images shown in Figure 8.2. Figure 8.9 presents comparisons of distortion due to embedding for three methods: MM2-z, MM2 and F5. Using all AC coefficients leads to the least distortion for all tested embedding rates. Figure 8.10 shows the average embedding error per message bit caused by

MM2-z. 100 images from UCID database were used to embed messages with six different embedding rates and the results are shown as a box plot. The box plot demonstrates that the level of distortion due to MM2-z embedding is very low, below 0.1 for 0.4 bpc. Figures 8.11 and 8.12 show the distortion levels by many embedding algorithms including F5 proposed by Westfeld. These graphs demonstrate that the embedding algorithms including zero-valued coefficients (MMx-z and PB-z) introduce lower level of distortion than the shrinkage-avoiding algorithms (MMx and PB) and shrinkage-permitting algorithms (MMx-s and PB-s); The MM3-z method introduces lower distortion than the MM2-z and PB-z methods. All proposed algorithms in this thesis produces lower level of distortion compared to the well-known embedding algorithm, F5.

Accuracy of Analysis

In order to verify our analysis on MM2-z, we estimated the expected values of embedding errors from 100 images and compared them to experimentally obtained embedding errors. Messages were embedded in six different embedding rates. Figure 8.13 shows the absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors and their results are shown as a box plot. The box plots demonstrate that the predicted errors are very close to the actual errors for all embedding rates; the differences are all below 0.005.

Coefficients Statistics

We have investigated how the MMx-z method changes the distribution of DCT coefficients. Changed distributions may give some hints of that the object may contain hidden information. Hence, preserving statistical properties is a very important requirement in steganography applications. To demonstrate the comparison numerically, we measured the Chi-square and Euclidean distances of JPEG coefficient histograms between the cover images and their stego images. Figures 8.14 and 8.15 present the Chi-square and Euclidean distances between the stego and cover image histograms respectively. It is observable that the MMx-z

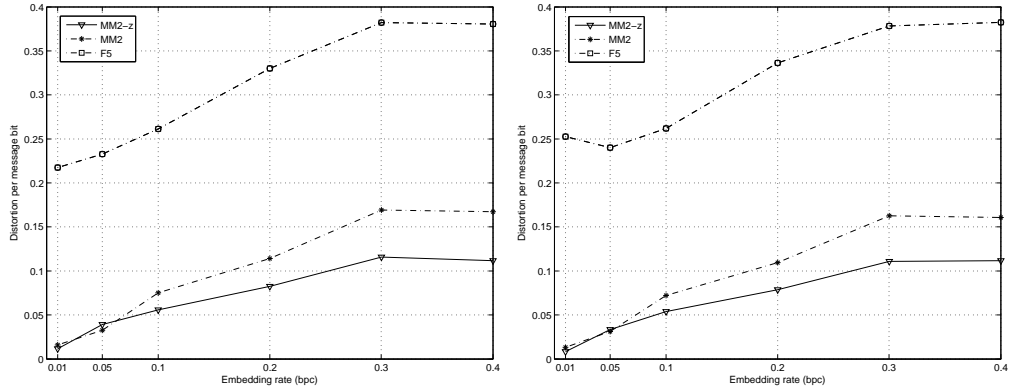


Figure 8.9: Average errors when embedding a single message bit. F5 is compared with MM2 and MM2-z. Embedding distortion per embedding message bit with various bpc ranging from 0.01 to 0.4 using the test images in Figure 8.2a (left) and Figure 8.2b (right).

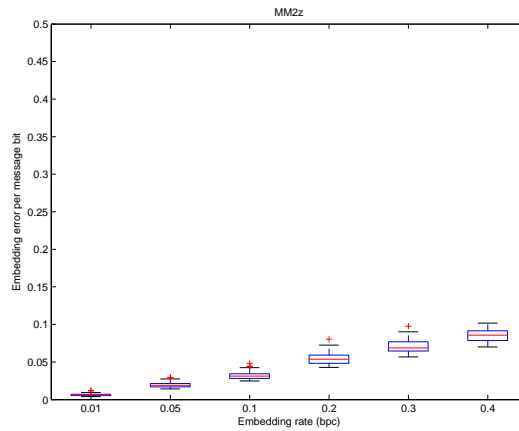


Figure 8.10: Average embedding error per message bit using the MM2-z method with six different embedding rates in bpc. 100 images from UCID database were used to embed messages.

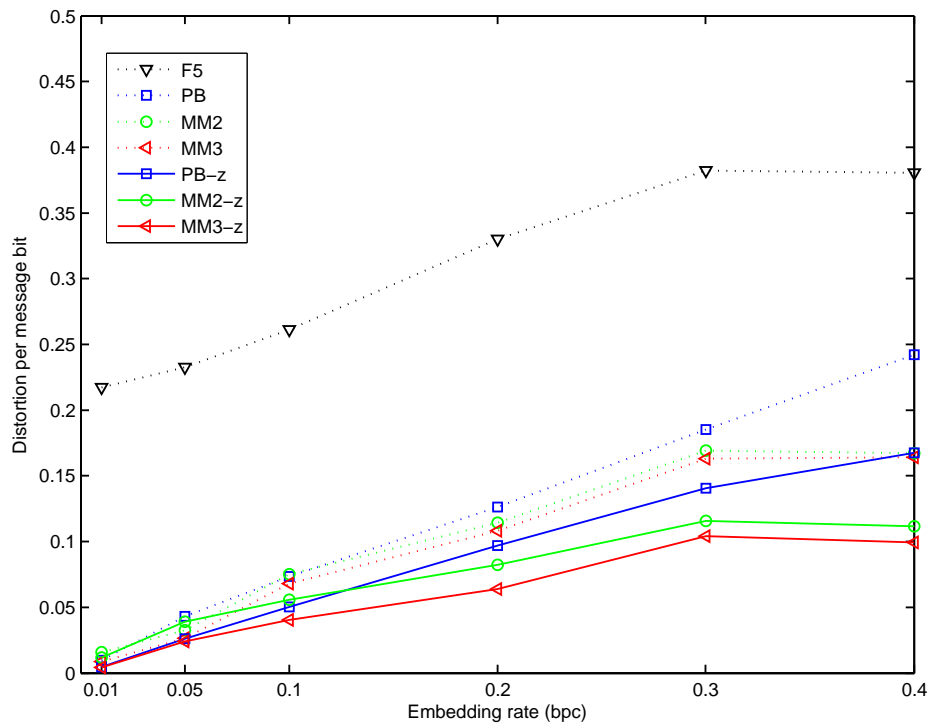


Figure 8.11: Comparison of distortion caused by various methods while embedding the same number of message bits. To generate this graph, the armadillo image (see Figure 8.2) is used. It is a JPEG color image and the number of AC coefficients is 183708.

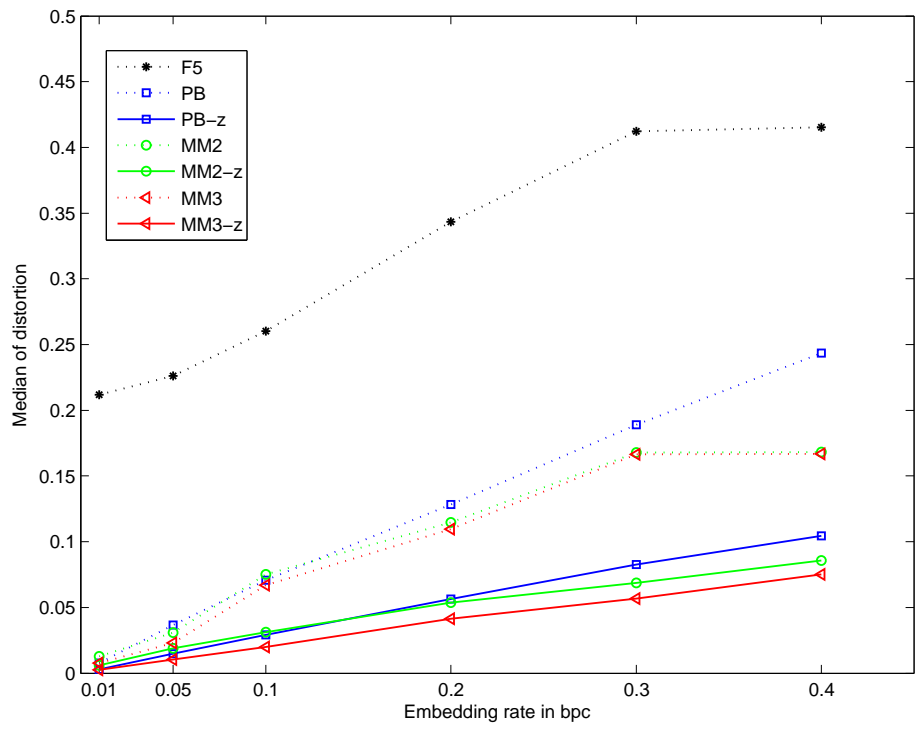


Figure 8.12: The median value of distortions obtained from 100 images of the UCID database.

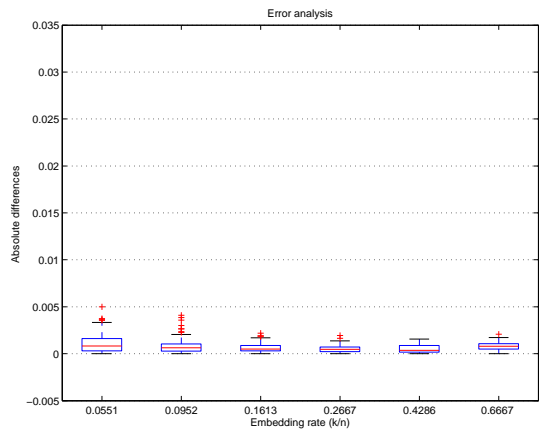


Figure 8.13: Accuracy of the error analysis on MM2-z. The absolute value of difference between the theoretically estimated embedding errors and the experimentally obtained embedding errors using 100 images from UCID database.

method has also improved in preserving coefficients statistics compared to MMx.

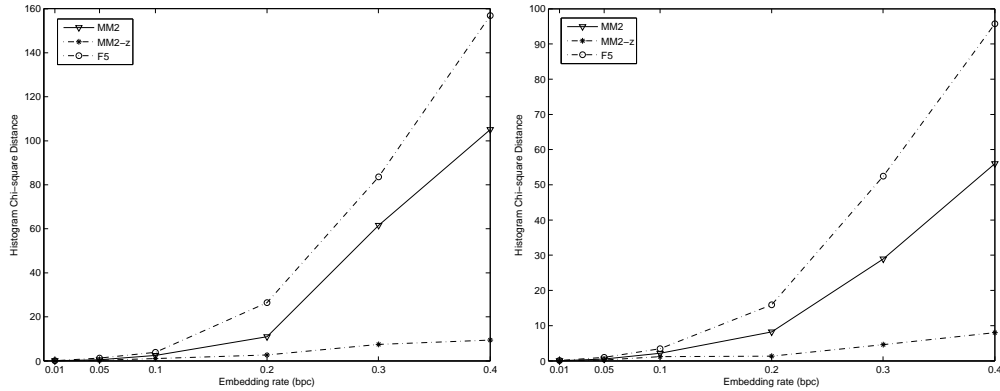


Figure 8.14: Chi-square distance comparisons of the MM2, MM2-z, and F5 algorithms in various bpc using the test images in Figure 8.2a (left) and Figure 8.2b (right).

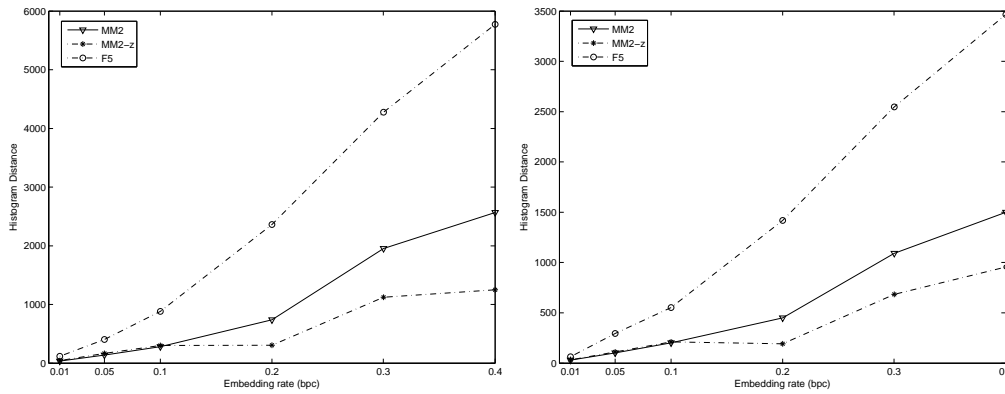


Figure 8.15: Euclidean distance comparisons of the MM2, MM2-z, and F5 algorithms in various bpc using the test images in Figure 8.2a (left) and Figure 8.2b (right).

Steganalysis Tests

We have implemented a steganalysis method proposed by Shi et al. [57], which is well known steganalysis algorithm to attack JPEG steganography. We used 1000 images from UCID (Uncompressed Color Image Database) [72] for cover images. For comparison, we embedded six different embedding rated message in 1000 images using our eight embedding algorithms and two other embedding algorithms, F5 [21] and Model-based steganography [23] (MB). The embedding rates were ranged from 0.01 to 0.4 bpc, where bpc is the number of embedded message bits divided by the number of non-zero AC coefficients.

Tables 8.1 and 8.1 list the detection rates for each algorithm with five embedding rates. The detection rates are calculated in three ways: accurate rate (AC), true-negative rate (TN), true-positive rate (TP). Recall that the detection rate of 50% means that the steganalysis can not distinguish the cover image and the stego image, therefore steganographic methods aim at 50% detection rate in steganalysis tests. As illustrated in Figure 8.16, the PB-z and MMx-z methods—MM3-z is the best—have the lower detection rates and thus enhance a security compared to other compared algorithms.

8.3 Impact of Modifying Zero-valued Coefficients

A typical information-hiding algorithm for JPEG images excludes DC and zero-valued coefficients (0 coefficients) for embedding. Altering DC coefficients may cause large distortion, and changing many zero-valued coefficients to non-zeros values may have effects on compression rates of JPEG files [26]. That is reason the number of bits that can be embedded in JPEG images usually less than the one in raw images (e.g., BMP images). Depending on the texture of JPEG images, the number of non-zero coefficients is various and then the embedding capacity will vary. Rather than following the predetermined assumption, this thesis have examined the disadvantages and advantages of including 0 coefficients for embedding.

There are several benefits in using 0 coefficients for embedding [28]. First, we can increase capacity of embedding for a given image by increasing the number of usable coefficients. The portion of 0 coefficients is relatively large compared to other valued coefficients. For example, the portion of 0 coefficients is 68.7% for the armadillo image (in Figure 8.2) and 68.9% for the tiger image (in Figure 8.2). Excluding 0 coefficients in the embedding process results in a decrease in the number of usable coefficients significantly. The more usable coefficients there are, the more messages can be carried or the less distortion can be caused to carry the same amount of messages.

The second benefit is that we can decrease distortion for a given message length. For

Table 8.1: Steganalysis test and detection rates (%).

Algorithm	BPC	AC	TN	TP
F5	0.01	56.78	52.84	60.95
	0.05	61.22	62.23	60.13
	0.1	71.78	68.35	75.63
	0.2	90.33	87.14	93.57
	0.4	98.11	96.17	100
MB	0.01	57.33	53.22	61.11
	0.05	79.78	79.63	79.89
	0.1	93.33	91.15	95.47
	0.2	99.11	98.63	99.56
	0.4	99.89	99.78	100
PB	0.01	57	54.86	59.12
	0.05	53.56	55.10	52.16
	0.1	64.67	61.72	67.38
	0.2	81.89	82.04	81.75
	0.4	99.56	99.11	100
PB-z	0.01	55.33	53.16	57.74
	0.05	50.67	85.41	18.59
	0.1	53.67	82.60	20.14
	0.2	55.67	60.68	50.96
	0.4	89.67	90.41	88.96
PB-s	0.01	50.2	86.85	13.25
	0.05	58.40	55.78	61.04
	0.1	58.60	64.26	52.99
	0.2	82.80	79.75	85.66
	0.4	99.80	99.60	100.00
PB-g	0.01	50.00	0	100.00
	0.05	50.30	86.00	14.60
	0.1	61.60	58.80	64.40
	0.2	76.90	75.80	78.00
	0.4	95.10	94.00	96.20

Table 8.2: Steganalysis test and detection rates (continued).

Algorithm	BPC	AC	TN	TP
MM2	0.01	57.78	54.65	60.79
	0.05	58.33	57.50	59.32
	0.1	62.33	62.00	62.85
	0.2	88.44	86.97	90.10
	0.4	99.78	99.56	100
MM2-z	0.01	50.6	84.2	17.00
	0.05	52.20	58.17	46.18
	0.1	51.40	55.42	47.41
	0.2	52.20	76.45	29.46
	0.4	92.40	92.49	92.31
MM3	0.01	50.2	81.80	18.60
	0.05	56.00	51.20	60.80
	0.1	56.20	58.23	54.18
	0.2	88.40	86.36	90.31
	0.4	99.80	99.60	100.00
MM3-z	0.01	NA	NA	NA
	0.05	50.50	64.40	36.60
	0.1	54.20	45.42	63.05
	0.2	60.40	59.84	60.98
	0.4	86.60	86.06	87.15

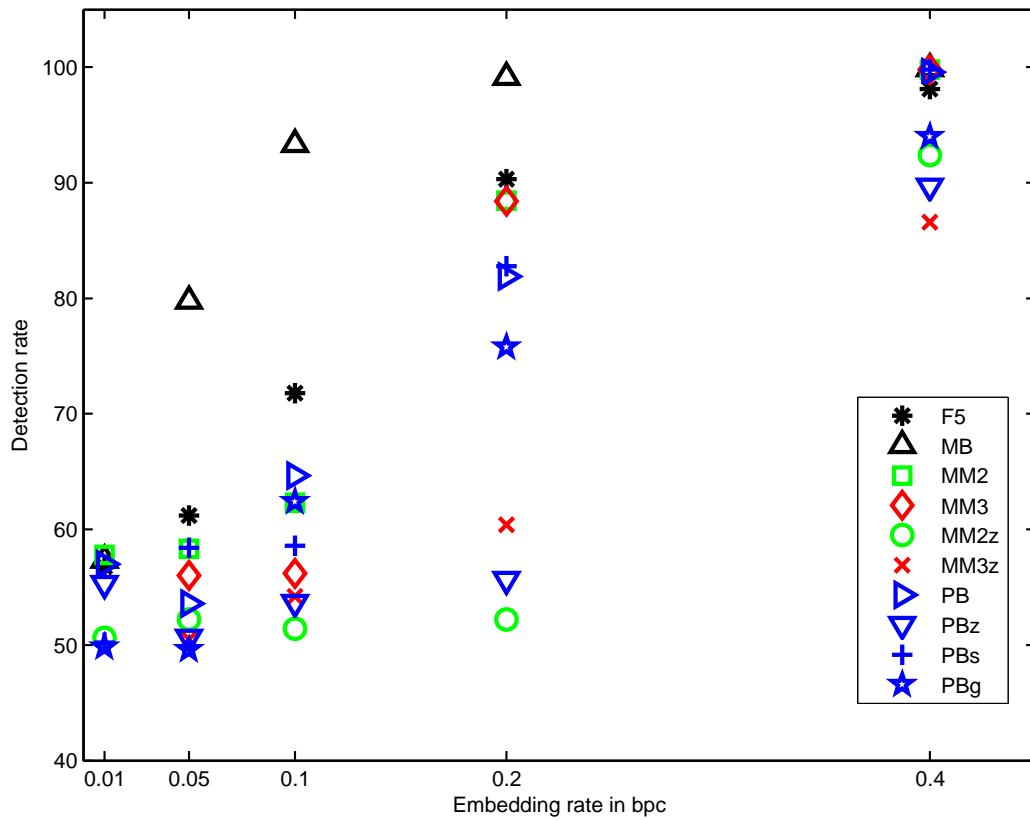


Figure 8.16: Comparison detection rates in accuracy(AC) of various steganographic algorithms: F5, MB, MM2, MM2-z, MM3, MM3-z, PB, PB-z, PB-s and PB-g.

example, in matrix coding [21], we can increase the efficiency of embedding per change by increasing the block size. Therefore, it is very attractive to include 0 coefficients for embedding in terms of distortion and payload.

The third benefit is that it will solve the shrinkage problem. Westfeld first used this term to describe the operation when -1 and 1 coefficients are changed to 0 during the embedding procedure. Because typical information hiding methods are designed not to consider 0 coefficients for embedding a message, several methods were proposed to handle those coefficients. For example, Westfeld proposed the repeat-embedding technique in F5 and we also proposed the shrinkage-avoiding algorithm and shrinkage-permitting algorithm in Chapters 5 and 6. Those methods solve the desynchronizing problem caused by shrinkage with an additional cost of distortion. However if we use 0 coefficients for embedding, the shrinkage does not cause a problem any more.

It is valuable to find a parameter that controls the amount of usable coefficients (including 0 coefficients) and to use the parameter for suppressing the disadvantages of using 0 and enhancing the advantages. We will use a predetermined threshold to limit the allowed number of coefficients for embedding. This threshold does not limit the exact number of 0 coefficients; however, it does eventually control the amount of 0 due to the property of JPEG encoding. After doing DCT in the JPEG encoder, all coefficients $\tilde{x}_{(i,j)}$ are scanned in a zigzag order of the 8×8 block, where i and j are the indexes in the block ($1 \leq i, j \leq 8$). The coefficients close to $\tilde{x}_{(8,8)}$ are more likely to be 0 coefficients. The number of usable coefficient is determined by the threshold θ ($1 \leq \theta \leq 63$). (Then the number of 0 coefficients involved in the embedding process can be roughly determined.) For example, when $\theta = 63$, all AC coefficients are used for embedding. The larger the θ , the more 0 coefficients involved in embedding.

To see possible risks of including 0 coefficients for embedding, entropy and compression ratios are measured and compared to those of other methods that do not use 0 coefficients. The file size and PSNR results show how including 0 coefficients affects the JPEG compression ratio and visual quality of decoded images. From these experiments, we expect to

evaluate benefits and limits of including 0 coefficients for embedding.

Entropy

Modifying 0 coefficients for embedding may cause some concerns including possible increase in entropy of images; increased entropies may cause to being detected by steganalysis tools. We have measured entropies of cover images and stego images. Figures 8.17 and 8.18 show the entropy comparisons between cover images and stego images for the parity coding (PB and PB-z) and matrix coding based algorithms (F5, MMx and MMx-z). Notice that the entropies of PB-z and MMx-z have not changed much compared to PB, MMx, and F5. The entropies of the MM2-z stego image are closer to those of the cover image than the MM2 stego images. These entropy results indicate that including 0 coefficients as a useable coefficients does not have a big impact on the entropy of the stego image with embedding rates up to 0.4 bpc.

File Sizes

In this subsection, file sizes of stego images are measured to examine the effect of including 0 coefficients for embedding. There are concerns about the idea of changing 0 coefficients for embedding that it would possibly decrease compression rates, and that it would increase the file sizes. Table 8.3 lists the file size of the stego images created by three embedding methods in bytes. For easy comparison, Table 8.4 calculates ratios of file sizes of the stego image over the cover image. In F5 with 0.3 and 0.4 bpc, the file sizes are decreased because the number of 0 are increased due to the shrinkage. The increased frequency in 0 will be noticeable by the histogram attack. The results of file size indicate that the change of file size is not very significant up to 0.4 bpc. For example, the change in a file size of PB-z stego image (Armadillo; 32168 bytes for the cover image) is +3 bytes for 0.01 bpc; +17 bytes for 0.05 bpc; -47 bytes for 0.1 bpc; -52 bytes for 0.2 bpc; +192 bytes for 0.4 bpc.

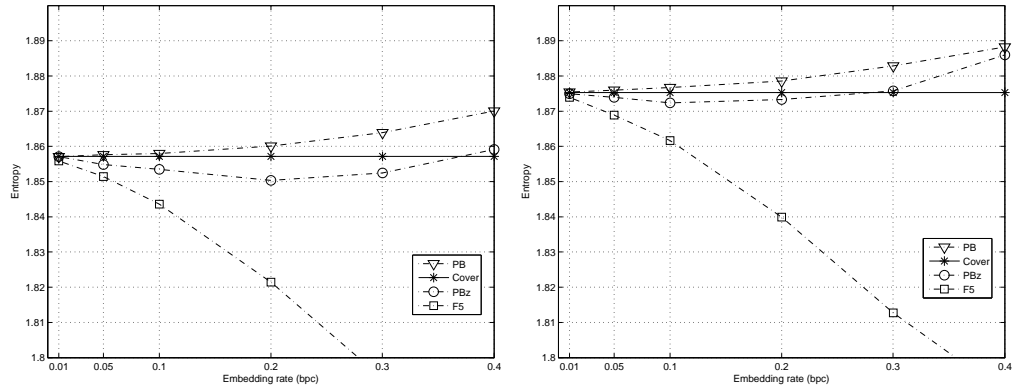


Figure 8.17: Entropy comparisons of PB stego, PB-z stego, and F5 stego in the various embedding rates using the test images in Figure Figure 8.2a (left) and Figure Figure 8.2b.

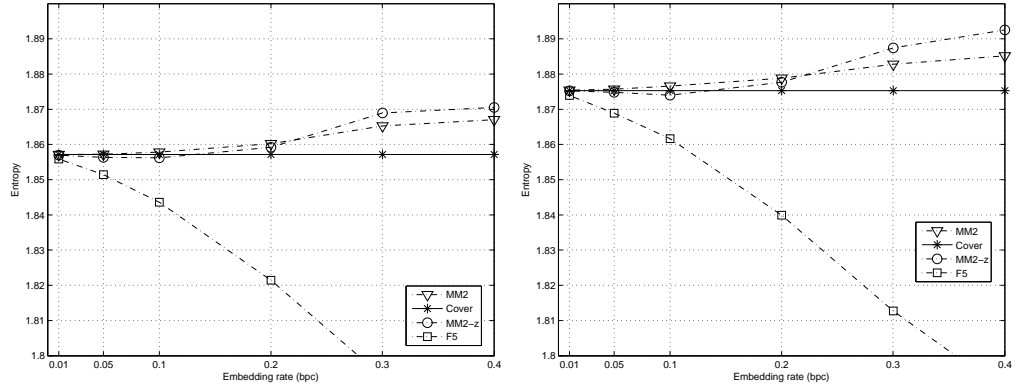


Figure 8.18: Entropy comparisons of MM2 stego, MM2-z stego, and F5 stego in the various embedding rates ranging from 0.01 to 0.4 using the test images in Figure Figure 8.2a (left) and Figure Figure 8.2b.

Table 8.3: File sizes of stego images in bytes

Image	Algorithm	0.01	0.05	0.1	0.2	0.3	0.4
Armadillo (cover: 32168)	F5	32807	32747	32643	32281	31885	31574
	PB	32857	32855	32869	32914	33046	33244
	PB-z	32860	32872	32822	32862	33086	33436
	MM2	32865	32871	32872	32941	33121	33167
	MM2-z	32860	32860	32915	33303	34424	34635
Tiger (cover: 20349)	F5	20783	20751	20666	20475	20237	20035
	PB	20834	20837	20833	20848	20939	21067
	PB-z	20824	20817	20798	20872	21010	21275
	MM2	20838	20827	20833	20876	20961	21002
	MM2-z	20839	20829	20830	21051	21581	21861

Table 8.4: The file size ratio between stego images and the corresponding cover images

Image	Algorithm	0.01	0.05	0.1	0.2	0.3	0.4
Armadillo	F5	1.0199	1.0180	1.0148	1.0035	0.9912	0.9815
	PB	1.0214	1.0214	1.0218	1.0232	1.0273	1.0334
	PB-z	1.0215	1.0219	1.0203	1.0216	1.0285	1.0394
	MM2	1.0217	1.0219	1.0219	1.0240	1.0296	1.0311
	MM2-z	1.0215	1.0215	1.0232	1.0353	1.0701	1.0767
Tiger	F5	1.0213	1.0198	1.0156	1.0062	0.9945	0.9846
	PB	1.0238	1.0240	1.0238	1.0245	1.0290	1.0353
	PB-z	1.0233	1.0230	1.0221	1.0257	1.0325	1.0455
	MM2	1.0240	1.0235	1.0238	1.0259	1.0301	1.0321
	MM2-z	1.0241	1.0236	1.0236	1.0345	1.0605	1.0743

PSNR

To compare the decoded image quality, we measured PSNR of the stego images embedded using PB-z, MMx-z, PB, and MMx. Table 8.5 lists the PSNR of the stego images created by the methods. The PSNR results of the stego images embedded by the algorithm of including 0 coefficients (PB-z and MMx-z) are very similar to the stego images embedded by the algorithm that use only non-zero coefficients.

Table 8.5: PSNR comparisons

Image	Algorithm	0.01	0.05	0.1	0.2	0.3	0.4
Armadillo	F5	36.1147	35.9487	35.6524	34.9477	34.2080	33.7212
	PB	36.1517	36.1286	36.0500	35.8122	35.3102	34.4333
	PB-z	36.1498	36.1105	35.9630	35.5303	34.7713	34.1059
	MM2	36.1546	36.1342	36.0390	35.6724	34.8988	34.6004
	MM2-z	36.1536	36.0682	35.9368	35.2374	33.8699	33.4484
Tiger	F5	34.7265	34.5903	34.4038	33.8379	33.3397	32.9882
	PB	34.7583	34.7410	34.7059	34.5691	34.2370	33.7476
	PB-z	34.7527	34.7223	34.6370	34.2950	33.6637	32.9182
	MM2	34.7578	34.7487	34.6963	34.5022	33.9553	33.7333
	MM2-z	34.7625	34.7065	34.5816	34.0809	32.9725	32.5266

Chapter 9: Summary

Information hiding refers to embedding additional digital data in host signals, and the host signals are JPEG images in this thesis. The requirements of information hiding algorithm are robustness, payload, imperceptibility and security. This thesis has focused on imperceptibility by minimizing distortion and has also dealt with a tradeoff between payload and security.

In the introduction, three goals for this thesis was set: minimizing distortion due to embedding, preserving statistical properties during embedding, and predicting a message length at a given distortion level. Distortion cannot be avoid in the information hiding procedure, but minimal distortion is preferable. Throughout the thesis, knowledge only being available to senders is utilized to achieve minimal distortion. More specifically, information is embedded by modifying JPEG coefficients in such a way that the introduced distortion is minimal. Although the proposed schemes use JPEG images for experiments, other formats can be used.

There is a a tradeoff between the amount of hidden information and distortion due to embedding. The more hidden information, the more distortion. There has been very little work on how to hide information optimally in terms of the tradeoff between distortion and the amount of hidden information. We derive the expected value of the embedding error as a function of the message length and the distribution of the rounding errors at the JPEG quantization stage.

Distortion is defined as a distance between the original data and the modified data after information has been embedded. Embedding messages in the JPEG format yields distortion from two sources: rounding errors that occurs at the JPEG quantization procedure and embedding errors that occurs at the modification of the cover data to embed messages.

We have designed embedding methods that use block-based coding techniques (parity

coding and matrix coding) that allow more than one choice for embedding a single message bit, and that utilize the rounding error for minimal distortion as a criteria for choosing one among those choices. Specifically the modified matrix coding technique was proposed in Chapter 4 to increase the number of choices to utilize the distortion minimization scheme.

There is one significant issue to consider when using JPEG images to embed messages: How to handle zero-valued coefficients? Changing zeros to non-zero values in JPEG information hiding techniques has been avoided because of concerns of compression rates, detectability and image quality, and as a consequence, changing non-zero values to zeros needs a special care to extract correctly the message.

Three different approaches were proposed to handle those zero-valued coefficients in this thesis. The first approach is changing ± 1 coefficients to ± 2 in a manner that avoided creating additional zeros. Our embedding methods using this approach was named PB (parity coding based) and MMx (modified matrix coding based) described in Chapter 5. The second approach compensates by changing to some zero-valued coefficients in advance and then allowing the operations to create new zeros. Our embedding methods using this approach are named PB-s (parity coding based) and MMx-s (modified matrix coding based) described in Chapter 6. The third approach is using all AC coefficients including zeros. There may be a limitation on the payload that can be embedded without noticeable modifications with this approach. We have tested the third approach in various embedding rates ranging from 0.01 bpc to 0.4 bpc, and the experimental results showed that the concerns regarding changing non-zeros to zeros may not cause a big problem with up to a 0.4 bpc embedding rate, which is considered a large embedding rate in steganography applications. Our embedding methods using this approach are named PB-z (parity coding based) and MMx-z (modified matrix coding based) described in Chapter 8.

Preserving statistical properties is one of the problems on which many researchers have focused, especially in the field of steganography. because noticeable modifications in statistical properties will possibly be detected by steganalysis methods as an object of containing hidden information. In the thesis, one algorithm that employs weighted graph matching

to preserve a JPEG coefficient histogram while minimizing distortion was called PB-g. An approximation algorithm for maximum weighted matching was implemented for the fast matching method. The central idea of employing matching is that of embedding a message by interchanging only those coefficients whose changes do not modify the histogram and do guarantee a limited distortion.

In this thesis, broadly two embedding schemes have been proposed: the first embedding scheme aims to minimize distortion by utilizing JPEG rounding errors, and the second scheme aims to preserve statistical properties during embedding. Depending on requirements of applications, implementation can vary by combining some methods from the each proposed scheme. For almost every methods that we have proposed, we derived the expected value of the embedding error as a function of the message length and the distribution of the rounding errors at the JPEG quantization stage. We have presented a theoretical analysis based on order statistics, which uses the rounding error distribution to predict the distortion for a given embedding rate. Our experiments showed that our analysis predicted distortion rates accurately.

We measured distortion and compared to other embedding algorithms. The results demonstrated that our methods introduces a very low level of distortion and achieves a high level of security in the blind steganalysis tests. All methods in this thesis have been implemented in Java and employed a publicly available JPEG encoder/decoder.

Chapter 10: Conclusion

The contribution of this thesis is the exploration of the limits of information hiding: how low a level of distortion can be achieved with a given length of the embedded message. The guiding belief of pursuing minimal distortion is that (a) minimizing distortion is preferable in many applications, especially with current trends of favoring large size high quality pictures, and (b) minimal distortion to stego objects would lead to their being less detectable.

Three goals were established in the introduction: minimizing distortion, predicting the distortion associated with the message length, and preserving statistical properties during embedding. Throughout the thesis, the three goals were successfully achieved. Using data known to the sender (e.g., rounding errors of JPEG quantization and distributions of cover data), a very low level of distortion is achieved. While completing of this thesis, one of the minimizing distortion schemes proposed in the thesis (MM3) was published in a peer-reviewed conference proceedings, and the work was evaluated [77] as currently the best steganographic algorithm in terms of security. Several methods that have better performance than the MM3 method also has been proposed and their experimental results are included in this thesis.

In addition, the mathematical analysis shown in the thesis can be highlighted as a significant contribution. We expect this analysis to be used to change how the message length is chosen. Traditionally, the message length has been decided by trial-error in information hiding methods; however, the thesis has shown that the distortion can be accurately predicted and, this analysis is the first known work for analyzing the distortion in the field of information hiding. Our prediction methods can be easily applied to other embedding algorithms, with some modifications.

Methods to preserve statistical properties were also proposed in the thesis. The methods have successfully preserved the cover data's statistics. Our experimental results showed that

the methods causing less modification of the statistics are more likely to be less detectable with the blind steganalysis tests.

The steganalysis tests showed that minimizing distortion contributes to preserve the second order statistics such as blockiness and Markov features. Mathematical proof of relationship of minimal distortion and second order statistics will be the subject of future research. Additionally experimental proof that embedding methods using our algorithms do not change image features such as edges and corners and that they do not introduce any artifacts will be the subject of future research.

Limitations will also be studied further. Using the methods proposed in this thesis, embedded messages do not survive image processing methods such as recompression. Investigating embedding schemes that survive recompression will be considered for possible future research. In addition, the embedding methods with minimal distortion could be used in image authentication systems. Based on the proposed framework, designing an embedding algorithms that embed authentication data for detecting any tampering in videos while maintaining high quality of the original images will be considered for future research.

Bibliography

Bibliography

- [1] R. J. Anderson and F. A. Petitcolas, "On the limits of steganography," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 474–481, May 1998.
- [2] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1062–1078, July 1999.
- [3] M. Barni, "What is the future for watermarking? (part 1)," *IEEE Signal Processing Magazine*, pp. 55–59, September 2003.
- [4] —, "What is the future for watermarking? (part 2)," *IEEE Signal Processing Magazine*, pp. 53–57, November 2003.
- [5] P. Moulin and R. Koetter, "Data-hiding codes," *Proceedings of the IEEE*, vol. 93, no. 12, pp. 2083–2126, December 2005.
- [6] F. Bartolini, A. Tefas, M. Barni, and I. Pitas, "Image authentication techniques for surveillance applications," *Proceedings of the IEEE*, vol. 89, no. 10, pp. 1403–1418, December 2001.
- [7] E. T. Lin and E. J. Delp, "A review of fragile image watermarks," in *Proceedings of the Multimedia and Security Workshop Multimedia Contents*, October 1999, pp. 25–29.
- [8] B. C. Emin Martinian and G. Wornell, "On authentication with distortion constraints," in *International Symposium on Information Theory*, 2001.
- [9] I. J. Cox and M. L. Miller, "The first 50 years of electronic watermarking," vol. 2002, no. 2, pp. 126–132, Feb 2002.
- [10] C. Cachin, "An information-theoretic model for steganography," in *Proceedings of the Second International Workshop on Information Hiding*. London, UK: Springer-Verlag, 1998, pp. 306–318.
- [11] —, "An information-theoretic model for steganography," *Information and Computation*, vol. 192, no. 1, pp. 41–56, 2004.
- [12] M. Swanson, M. Kobayashi, and A. Tewfik, "Multimedia data-embedding and watermarking technologies," in *Proceedings of the IEEE*, vol. 86, June 1998, pp. 1064–1087.
- [13] I. Cox, M. L. Miller, and J. A. Bloom, *Digital watermarking*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.

- [14] N. Johnson, Z. Duric, and S. Jajodia, *Information Hiding: Steganography and Watermarking — Attacks and Countermeasures*. Boston: Kluwer Academic Publishers, 2000.
- [15] S. Katzenbeisser and F. Petitcolas, *Information Hiding: Techniques for Steganography and Digital Watermarking*. Norwood, MA: Artech House Books, 2000.
- [16] P. Wayner, *Disappearing Cryptography*. San Francisco: Morgan Kaufmann, 2002.
- [17] “Hide 2.1 2001.” [Online]. Available: <http://www.sharpthoughts.org>
- [18] J. Fridrich and M. Goljan, “Digital image steganography using stochastic modulation,” in *Proceedings of SPIE Electronic Imaging*, 2003, pp. 191–202.
- [19] J. Fridrich and R. Du, “Secure steganographic methods for palette images,” in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 1768. Springer-Verlag, 2000, pp. 47–60.
- [20] R. Machado, “Ez stego.” [Online]. Available: <http://www.stego.com>
- [21] A. Westfeld, “F5—a steganographic algorithm: High capacity despite better steganalysis,” in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 2137. Springer-Verlag, 2001, pp. 289–302.
- [22] N. Provos, “Defending against statistical steganalysis,” in *Proceedings of 10th USENIX Security Symposium*, 2001, pp. 323–325.
- [23] P. Sallee, “Model-based steganography,” in *Proceedings of 2nd International Workshop on Digital Watermarking*, ser. LNCS, no. 2939. Springer-Verlag, October 2003, pp. 154–167.
- [24] J. Fridrich, “Feature-based steganalysis for jpeg images and its implications for future design of steganographic schemes,” in *Proceedings of 6th International Workshop on Information Hiding*, ser. LNCS, no. 3200. Springer-Verlag, 2004, pp. 67–81.
- [25] Y. Kim, Z. Duric, and D. Richards, “Modified matrix encoding technique for minimal distortion steganography,” in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 4437. Springer-Verlag, 2007, pp. 314–327.
- [26] M. Kharrazi, H. T. Sencar, and N. Memon, “Image steganography: Concepts and practice,” *Lecture Notes Series, Institute for Mathematical Sciences, National University of Singapore*, 2004.
- [27] A. Westfeld and A. Pfitzmann, “Attacks on steganographic systems,” in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 1768. Springer-Verlag, 1999, pp. 61–75.
- [28] J. Fridrich and J. Kodovský, “Influence of embedding strategies on security of steganographic methods in the JPEG domain,” in *Proceedings of SPIE Electronic Imaging*, 2008.

- [29] J. Fridrich, M. Goljan, and D. Soukal, "Perturbed quantization steganography," *ACM Multimedia and Security Journal*, vol. 11, no. 2, 2005.
- [30] K. Solanki, A. Sarkar, and B. S. Manjunath, "YASS: Yet another steganographic scheme that resists blind steganalysis," in *Proceedings of International Workshop on Information Hiding*, 2007, pp. 16–31.
- [31] J. Fridrich, M. Goljan, and D. Soukal, "Perturbed quantization steganography with wet paper codes," in *Proceedings of ACM Workshop Multimedia and Security*. Magdeburg, Germany: ACM Press, 2004.
- [32] R. E. Newman, I. S. Moskowitz, L. Chang, and M. M. Brahmadessam, "A steganographic embedding undetectable by JPEG-compatibility steganalysis," in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, vol. 2578. Springer-Verlag, 2003, pp. 258–277.
- [33] P. Sallee, "Model-based methods for steganography and steganalysis," *International Journal of Image Graphics*, vol. 5, no. 1, 2005.
- [34] S. Hetzl and P. Mutzel, "A graph-theoretic approach to steganography," in *Proceedings of International Federation for Information Processing*, ser. LNCS, no. 3688. Springer-Verlag, 2005, pp. 119–128.
- [35] A. Westfeld, "Detecting low embedding rates," in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 2578. Springer-Verlag, 2003, pp. 324–339.
- [36] K. Lee, A. Westfeld, and S. Lee, "Category attack for LSB embedding of JPEG images," in *Proceedings of the International Workshop on Digital Watermarking*, ser. LNCS, vol. 4283. Springer-Verlag, 2006, pp. 35–48.
- [37] —, "Generalized category attack - improving histogram-based attack on JPEG LSB embedding," in *Proceedings of the International Workshop on Information Hiding*, 2007, pp. 378–391.
- [38] X. Yu, Y. Wang, and T. Tan, "On estimation of secret message length in jsteg-like steganography," in *Proceedings of International Conference on Pattern Recognition*, vol. 4. Washington, DC, USA: IEEE Computer Society, 2004, pp. 673–676.
- [39] T. Zhang and X. Ping, "A fast and effective steganalytic technique against jsteg-like algorithms," in *SAC '03: Proceedings of the 2003 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2003, pp. 307–311.
- [40] A. Ker, "A general framework for structural analysis of LSB replacement," in *Proceedings of 7th International Workshop on Information Hiding*, ser. LNCS, no. 3727. Springer-Verlag, 2005, pp. 296–311.
- [41] J. Fridrich, M. Goljan, and R. Du, "Detecting LSB steganography in color and gray-scale images," *IEEE Multimedia Magazine*, pp. 22–28, October 2001.
- [42] S. Dumitrescu, X. Wu, and N. D. Memon, "On steganalysis of random LSB embedding in continuous-tone images," in *Proceedings of IEEE International Conference on Image Processing*, 2002, pp. 641–644.

- [43] J. Fridrich, R. Du, and M. Long, “Steganalysis of LSB encoding in color images,” in *Proceedings of IEEE International Conference on Multimedia and Expo*, vol. 3, 2000, pp. 1279–1282.
- [44] J. Fridrich, M. Goljan, and R. Du, “Reliable detection of LSB steganography in color and grayscale images,” in *Proceedings of ACM Workshop on Multimedia and Security*, 2001, pp. 27–30.
- [45] C. Maroney, “Hide and seek.” [Online]. Available: <http://www.rugeley.demon.co.uk>
- [46] N. F. Johnson and S. Jajodia, “Steganalysis of images created using current steganography software,” in *Proceedings of the Second International Workshop on Information Hiding*. London, UK: Springer-Verlag, 1998, pp. 273–289.
- [47] J. J. Fridrich, M. Goljan, D. Hoge, and D. Soukal, “Quantitative steganalysis of digital images: estimating the secret message length,” *Multimedia Syst.*, vol. 9, no. 3, pp. 288–302, 2003.
- [48] D. Upham, “Jsteg.” [Online]. Available: <http://zooid.org/paul/crypto/jsteg/>
- [49] Y. Kim, Z. Duric, and D. Richards, “Towards lower bounds on embedding distortion in information hiding,” in *Proceedings of International Workshop on Digital Watermarking*, ser. LNCS, no. 4283. Springer-Verlag, 2006, pp. 362–376.
- [50] J. Fridrich, M. Goljan, and D. Hoge, “Steganalysis of JPEG images: Breaking the F5 algorithm,” in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 2578. Springer-Verlag, 2002, pp. 310–323.
- [51] —, “Attacking the OutGuess,” in *Proceedings of ACM Workshop on Multimedia and Security*. ACM Press, 2002.
- [52] H. Farid, “Detecting hidden messages using higher-order statistical models,” in *International Conference on Image Processing*, Rochester, NY, 2002. [Online]. Available: www.cs.dartmouth.edu/farid/publications/icip02.html
- [53] S. Lyu and H. Farid, “Detecting hidden messages using higher-order statistics and support vector machines,” in *5th International Workshop on Information Hiding*, Noordwijkerhout, The Netherlands, 2002. [Online]. Available: www.cs.dartmouth.edu/farid/publications/ih02.html
- [54] Y. Q. Shi, G. Xuan, D. Zou, J. Gao, C. Yang, Z. Zhang, P. Chai, W. Chen, and C. Chen, “Image steganalysis based on moments of characteristic functions using wavelet decomposition, prediction-error image, and neural network,” in *ICME*, 2005, pp. 269–272.
- [55] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath, “Steganalysis of spread spectrum data hiding exploiting cover memory,” in *IS&T/SPIE’s 17th Annual Symposium on Electronic Imaging Science and Technology*, January 2005.
- [56] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. Manjunath, “Steganalysis for Markov cover data with applications to images,” *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 2, pp. 275–287, June 2006.

- [57] Y. Q. Shi, C. Chen, and W. Chen, "A Markov process based approach to effective attacking jpeg steganography," in *Proceedings of International Workshop on Information Hiding*, 2006.
- [58] T. Pevný and J. Fridrich, "Merging Markov and DCT features for multi-class jpeg steganalysis," in *Proceedings of SPIE Electronic Imaging*, Jan 2007.
- [59] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proceedings of IEEE International Conference on Image Processing*, vol. 2, November 1994, pp. 86–90.
- [60] R. Wolfgang and E. Delp, "A watermark for digital images," in *Proceedings of IEEE International Conference on Image Processing*, vol. 3, September 1996, pp. 219–222.
- [61] P. W. Wong and N. Memon, "Secret and public key image watermarking schemes for image authentication and ownership verification," *IEEE Transactions on Image Processing*, vol. 10, no. 10, 2001.
- [62] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in *Proceedings of IEEE International Conference on Image Processing*, vol. 2, October 1997, pp. 680–683.
- [63] J. Fridrich, M. Goljan, and A. C. Baldoza, "New fragile authentication watermark for images," in *Proceedings of IEEE International Conference on Image Processing*, September 2000, pp. 446–449.
- [64] M. Wu and B. Liu, "Watermarking for image authentication," in *Proceedings of IEEE International Conference on Image Processing*, vol. 2, October 1998, pp. 437–441.
- [65] D. Kundur and D. Hatzinakos, "Towards a telltale watermarking technique for tamper-proofing," in *Proceedings of IEEE International Conference on Image Processing*, vol. 2, October 1998, pp. 409–413.
- [66] L. Xie and G. Arce, "Joint wavelet compression and authentication watermarking," in *Proceedings of IEEE International Conference on Image Processing*, vol. 2, October 1998, pp. 427–431.
- [67] R. Gonzales and R. Woods, *Digital Image Processing 2nd Edition*. Prentice Hall, 2002.
- [68] [Online]. Available: <http://www.dspguide.com/ch27/6.htm>
- [69] R. J. Anderson, "Stretching the limits of steganography," in *Proceedings of International Workshop on Information Hiding*, ser. LNCS, no. 1174. Springer-Verlag, 1996, pp. 39–48.
- [70] R. Crandall, "Some notes on steganography," 1998. [Online]. Available: <http://os.inf.tu-dresden.de/westfeld/crandall.pdf>
- [71] A. Paopoulos, *Probability, Random Variables, and Stochastic Processes*. Boston, MA: McGraw-Hill, 1991.

- [72] G. Schaefer and M. Stich, “Ucid - an uncompressed colour image database,” in *Proceedings of SPIE, Storage and Retrieval Methods and Applications for Multimedia*, January 2004, pp. 472–480.
- [73] S. Micali and V. Vazirani, “An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs,” in *Proceedings of IEEE Annual Symposium on Foundations of Computer Science*, 1980, pp. 17–27.
- [74] L. Lovász and M. Plummer, *Matching Theory*, ser. Vol. 29 of Annals of discrete mathematics. Publishing House of the Hungarian Academy of Sciences, 1986.
- [75] R. Preis, “Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs,” in *Proceedings of International Symposium on Theoretical Aspects of Computer Science*, 1999, pp. 259–269.
- [76] D. Avis, “A survey of heuristics for the weighted matching problem,” *Networks*, vol. 13, no. 4, pp. 475–493, 1983.
- [77] J. Fridrich, T. Pevný, and J. Kodovský, “Statistically undetectable jpeg steganography: dead ends challenges, and opportunities,” in *Proceedings of the 9th workshop on Multimedia & security*. New York, NY, USA: ACM, 2007, pp. 3–14.

Curriculum Vitae

Younhee Kim received her Bachelor of Science in Computer Science and Master of Engineering in Information and Communication Technology from Ajou University, Suwon, Republic of Korea in 2000 and in 2002, respectively. She has been a software engineer for a video compression company (IMaster, Co.), and a graduate teaching assistant and research assistant for George Mason University. She is a reviewer of Pattern Recognition. Her research interest include information hiding, image/video analysis, and human computer interaction.