Steganalysis of binary images

Cheng, Jun

2008

Cheng, J. (2008). Steganalysis of binary images. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/13225>

<https://doi.org/10.32657/10356/13225>

STEGANALYSIS OF BINARY IMAGES

Cheng Jun

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2008

*To all in my family,
for their encouragement and love.*

Acknowledgements

First and foremost, I would like to express my deepest appreciation and sincere gratitude to my supervisor, Dr. Alex Kot Chichung, for his invaluable instruction and great support. It is he who gave me strong support and sincere encouragement throughout the tough time of the research. Within these five years, I not only learned specialized knowledge from him, but also learned the spirit of exploring science from him. This work would not have been possible without his excellent guidance and persistent encouragement.

I would like to thank my parents, my grandma, and all the members in my family for teaching me to understand the value of hard work and supporting me during my education as well as in life in general. Their love and support are always the greatest inspiration to me from the first day I came to the world.

I would like to acknowledge the School of EEE, Nanyang Technological University, Singapore, for awarding me the research scholarship and providing me with excellent research facilities. I also thank the anonymous reviewers of my thesis for their valuable comments and suggestions.

Finally, I would like to thank all of my friends, graduate students, the faculty and technicians at Workstation Resource Lab, Media Lab and Information Systems Research Lab. All of my experiences and interactions with these peo-

ple have shaped me and my work, and helped making my graduate experience worthwhile and unforgettable.

Summary

With an increasing use of digital binary images including text, graphic and halftone images in daily life, security of the digital binary images such as certificate, transcript, scanning hard copy to electronic copy etc. is a big concern. Such a concern makes data hiding in binary images an important and attractive research topic in recent years. With the development of the data hiding techniques for binary images, steganalysis of binary images starts receiving attention too. The concern of using binary image data hiding techniques by terrorists for conspiracy makes the research study of steganalysis of binary images necessary and important.

In this thesis, we firstly give an overview of steganography and steganalysis, followed by the introduction of the proposed steganalysis techniques for different types of binary images. Two techniques are introduced for the detection of hidden data in clean and scanned text binary images. An objective distortion measure for binary text and binary graphic images based on edge line segment similarity is proposed and used together with other distortion measures for steganalysis. Filtering based and projection based inverse halftoning methods are applied in the steganalysis of halftone binary images and their performances in detection of hidden data are compared. Finally, conclusions and recommenda-

tions for future work are given.

Detection of data hidden is usually done by examining some features of the images, such as image wavelet statistics, image block statistics which are affected by data hiding. Thus most of steganalysis work is to find out these features and utilize them in the detection. Our focus is on how the embedding processes distort the images, not only visual distortion, but also the distortion to the image features such as the regularities. Exploring the regularities of the images is not only useful for steganalysis, but also beneficial for minimizing the distortions as destroying the regularities of the images may often introduce visible distortion.

Contents

Acknowledgements	i
Summary	iii
List of Figures	ix
List of Tables	xii
1 Introduction	1
1.1 Motivation	3
1.2 A history review	6
1.2.1 Steganography	6
1.2.2 Steganalysis	8
1.3 Objective and contributions	10
1.4 Outline of the thesis	11
2 Literature Review	13
2.1 Steganography and steganalysis for binary document images . .	13
2.1.1 Line and word shifting for text images	14

2.1.2	Pixel flipping technique for text & graphic images	15
2.2	Steganography and steganalysis for halftone images	21
3	Text Images	24
3.1	Introduction	24
3.2	Clean text-font image	25
3.2.1	Observation	25
3.2.2	The feature extraction	27
3.2.3	Detection and message length estimation	28
3.2.4	Experimental results	31
3.3	Scanned text image	34
3.3.1	Common features in embedding process	35
3.3.2	The feature extraction	36
3.3.3	Theoretical analysis	39
3.3.4	Threshold selection	46
3.3.5	Experimental results	50
3.4	Summary	53
4	Binary Graphic Images	55
4.1	Introduction	55
4.2	Edge line segment similarity measure for binary images	57
4.2.1	Background	57
4.2.2	Edge line segment	59
4.2.3	Human visual perception	61

4.2.4	Edge line segment similarity	62
4.2.5	Identifying and mapping of edge line segments	63
4.2.6	Total distortion measure	69
4.2.7	Subjective test	69
4.3	Steganalysis using distortion measures	78
4.3.1	Distortion control in embedding	78
4.3.2	De-noising process	79
4.3.2.1	Morphological smoothing	79
4.3.2.2	De-noising the stair case pattern	82
4.3.3	Selection of distortion measure	88
4.3.4	Support vector machine	91
4.3.5	Experimental results	93
4.4	Summary	96
5	Halftone Image	99
5.1	Introduction	99
5.2	A general method	105
5.2.1	Feature extraction and classification	106
5.3	Filtering based methods	107
5.3.1	Fixed low pass filter	107
5.3.2	Experimental results	111
5.3.2.1	Comparison with other filters	114
5.4	Projection based methods	115

5.4.1	Projection for ordered dithering image	115
5.4.2	Projection for error diffusion image	117
5.4.3	Experimental results	119
5.4.3.1	Comparison with filtering based method at dif- ferent capacity	125
5.5	Summary	127
6	Conclusions and Recommendations	129
6.1	Conclusions	129
6.2	Recommendations for future research work	131
	Author's Publications	134
	Bibliography	136

List of Figures

1.1	Modern steganography system.	7
2.1	Visual effect of embedding.	17
2.2	General methodology for binary image data hiding.	19
3.1	Effect of embedding.	27
3.2	Estimation of embedding rate for different embedding schemes. .	32
3.3	Examples of ‘Super’ pixels in [47]	33
3.4	An example of L -shape pattern.	35
3.5	Examples of isolated and non-isolated COL pixels.	38
3.6	The effect of flipping a COL pixel.	40
3.7	The diagram of each subset division.	40
3.8	Examples of pixels from sets $A_{I,I}$, $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$	41
3.9	The curves of theoretic increase.	45
3.10	The theoretical increment from $\xi_I - \xi_{N,N}$ to $\xi_I^* - \xi_{N,N}^*$ vs. the actual increment.	46
3.11	An original scanned image and a clean image.	47
3.12	The ROC curve.	50

3.13	The detection rates at different embedding rates by a fixed and image dependent threshold.	51
4.1	Two 3×3 patterns with same distortion according to DRDM. .	59
4.2	Illustration of ‘edges’.	59
4.3	Eight types of corners	60
4.4	Two types of crosses.	60
4.5	Edge line segments associated with flipped pixels.	61
4.6	Enlarged “t”.	63
4.7	Pixels and borders in a 3×3 pattern.	64
4.8	Relationship between b_i and the associated edge line segment. The pixels in “grid” are don’t care pixels.	66
4.9	Examples of removing or creating a one-pixel-long edge line segment.	67
4.10	Shifted edge line segments.	68
4.11	Locating the two ends of an edge line segment.	69
4.12	Original images.	70
4.13	One set of distorted images based on the English text image. . .	73
4.14	One set of distorted images based on the Chinese text image. .	73
4.15	One set of distorted images based on Cartoon 1 image.	74
4.16	One set of distorted images based on Cartoon 2 image.	74
4.17	Distribution of ranking scores for English text images.	75
4.18	Distribution of ranking scores for Chinese text images.	75
4.19	Distribution of ranking scores for Cartoon 1 images.	76

4.20	Distribution of ranking scores for Cartoon 2 images.	76
4.21	The diagram of the detector using distortion measures.	78
4.22	Patterns with protrusion and hole.	80
4.23	Digital representation of analog lines.	83
4.24	Consecutive same type corners.	83
4.25	Probability of occurrence of e_k^B in different ranges $[0, 0.5)$, $[0.5, 1)$ and $[1, +\infty)$	86
4.26	Probability of occurrence of e_k^L in different ranges $[0, 0.5)$, $[0.5, 1)$ and $[1, +\infty)$	87
4.27	Some example of test images from the database.	94
5.1	The embedding system.	104
5.2	Flowchart of the detector.	106
5.3	Reconstructed images from ordered dithering based natural and stego halftone image using projection method.	116
5.4	Reconstructed images from error diffusion with Jarvis kernel based natural and stego halftone images using projection method. . . .	120
5.5	Reconstructed images from error diffusion with Steinberg kernel based natural and stego halftone images using projection method.	121
5.6	The reconstructed images from natural and stego images gener- ated by ordered dithering with 4×4 Bay's screen.	123
5.7	The reconstructed images from natural and stego images gener- ated by error diffusion with Steinberg kernel.	124
5.8	Average accuracy of the detectors at different embedding rates.	126

List of Tables

4.1	Numbers of edge line segments associated with pixels A, B and C and subjective distortion.	62
4.2	Experimental results.	72
4.3	F score of ANOVA.	90
4.4	Selected features from SFFS.	91
4.5	The detection results for individual embedding scheme based on the individual features.	95
4.6	The detection results for individual embedding scheme based on the best combination of features.	96
4.7	The detection results for ensemble schemes.	96
5.1	8×8 Bay's dispersed dither matrix.	101
5.2	Javis kernel.	102
5.3	Floyd and Steinberg kernel.	102
5.4	Minimum MSE (left) and Maximum PSNR (dB) (right) achieved by FIR filters with different sizes for three standards images. .	111
5.5	Classification results by the filtering based method.	112

5.6	Classification results using different filters. The classifier are trained by combining different embedding schemes.	115
5.7	Classification results by the projection based method.	122
5.8	Relationship between halftoning noise, embedding noise and de- tectors' average accuracy.	128

Chapter 1

Introduction

Data hiding has been one of the important topics in both academia and industry during the past decade as we can see from the increase in publications and patents appeared in this area. Data hiding is done by hiding the important information into a non-important cover object [26]. The cover object should contain some nonessential information by modifying which would not cause visible distortion. For example, given a color image, the least-significant bit (LSB) of each pixel can be changed to embed the hidden secret information bits [95] without introducing visible distortion.

There are two main reasons that make data hiding attractive. The first reason is that data hiding can be used to protect intellectual property. Digital revolution allows the digital formatting of media such as document, image, video, audio, etc. With the easy reproducing and editing capability of the computer tools as well as the world-wide broadcasting capability of the internet, the security issues related to these digital media such as copyright infringement, illegal distribution, and unauthorized tampering emerge. It becomes more difficult

to protect the intellectual property. Data hiding provides a way to insert a watermark to mark the original owner of the publication [28, 85], or embed a fingerprint or serial number to identify a violator [99, 114], or hide a digital signature of the original copy to protect its integrity [110], etc. The second reason is that it can be used to conduct covert communication for information security [5, 56, 62]. A traditional way to secure the information is cryptography method. Cryptography method encrypts the information with a key and makes people unable to decrypt the encrypted message without knowing the encryption key. However, it has rarely been adequate in practice as surveyed in [89]. The accompanying drawback with encryption is that it also flags the importance of the message and thus attracts cryptanalysts' interests. Data hiding method, however, hides the existence of the important message to avoid causing the opponents' attention. The important message is usually hidden in a non-important cover signal and the existence of the hidden message in the cover signal is hardly perceptible. When data hiding is used to hide the existence of a secret message, it is also called steganography.

Similar to the relationship between cryptography and cryptanalysis, the opponent of steganography is steganalysis. Steganalysis [57] refers to the art and science of discrimination between stego objects and cover objects. It needs to be done without the knowledge of a secret key used for embedding and maybe even without the knowledge of the embedding algorithm. Compared with decoding the message in cryptanalysis where encrypted message is known, decoding the hidden message in steganalysis without the detail knowledge of the embedding algorithm is more difficult. However, since the main objective to use steganography is to conceal the existence of a hidden message, simply detecting the existence of the hidden message from the cover signal defeats the steganogra-

phy. A higher level of attack to steganography may include the estimation of the message length [37, 39], secret key estimation [100], determination of the embedding algorithm and even decoding the hidden message [17]. Nevertheless, detecting the use of steganography is the first and most important step. Once the existence of a covert communication is detected, some precautions can be made to prevent further damage and defeat the conspiracy. It is always good to know things that your adversary does not want you to know.

Depending on the availability of the steganography algorithm for the steganalyst, we categorize the steganalysis techniques into universal steganalysis and specific steganalysis technique:

- Universal steganalysis is designed for the detection of a series of embedding techniques and the knowledge of the detail embedding algorithms is not available. Universal steganalysis is also known as blind steganalysis.
- Specific steganalysis is designed for attacking data hiding by a specific embedding algorithm. Thus the steganography algorithm is available. Specific steganalysis is also known as non-blind steganalysis.

A universal steganalysis technique usually has less accuracy in detection but it is usable on new embedding techniques. Specific steganalysis techniques may use the details knowledge of the embedding algorithm and thus have good detection accuracy, but it is useless to an unknown embedding algorithm.

1.1 Motivation

As data hiding technique is used to protect the intellectual property and conduct covered communication, it creates problems also. The concern in steganalysis

is that the data is hidden. Once the data hiding techniques are developed and published, there is a potential that these techniques are used by anybody including terrorists for conspiracy.

In February 2001, *USA Today* reported that terrorists are using steganography to hide their communication from law enforcement [63]. “It sounds far-fetched, but U.S. officials and experts say it’s the latest method of communication being used by Osama Bin Laden and his associates to outfox law enforcement.” There are widely reported rumors that images on auction sites contain hidden messages [91]. In July 2002, *USA Today* reported about terrorist web sites such as azzam.com, Almuhajiroun.com, and Qassam.net, which contain hidden message in their pictures and texts [64]. The hidden messages contain instructions for al-Qaeda’s next terrorist attacks. In another interesting report [49], Wetstone Technologies studied random images on the webs in their steganography research for the Air Force Research Laboratory and watched a picture of a sewing machine being auctioned on ebay as pixel patterns changed every few days. Due to these threats, developing steganalysis techniques that can detect the existence of the hidden message is necessary and important in anti-terrorism.

On the other hand, since steganalysis is used to detect existence of the hidden message, it can be used to reflect the security level of the steganography algorithm. A possible way to use steganalysis to determine the security level of the embedding scheme is proposed by R. Chandramouli *et al.* as follows [18]: An embedding scheme is called γ secure with respect to the steganalysis technique if $|P_{FA} - P_{Det}| \leq \gamma$, where P_{FA} and P_{Det} are the probability of false alarm and detection rate obtained using the steganalysis technique respectively, $0 \leq \gamma \leq 1$.

If $\gamma = 0$, then the embedding algorithm is perfectly secure with respect to this steganalysis technique. By investigating steganalysis techniques, we may determine the security levels of the existing steganography techniques.

For a long time when people study data hiding as well as steganalysis in digital images or videos, they focus on color or gray level images, videos [81]. In recent years, some researchers have moved their attention to black and white images because of the widely use of these images in documents. A series of data hiding algorithms for binary document images [11, 12, 46, 101, 113] are proposed by different researchers and groups. Data hiding in halftone image is also discussed [41, 42, 44, 66, 109]. With the development of these binary image data hiding schemes, the concern of using these techniques by the terrorists for covert communication becomes involuntary. One reason is that these binary data hiding schemes can provide a high capacity. The second reason is that binary images including document, graphic etc. are widely used in our daily life. Thus it is very convenient to hide quite a lot of information in these widely used binary images. It will be necessary to have precautions to detect covert communication through binary images. Detection of hidden data in binary images may also be necessary in forensic applications. It is expected that electronic documents would be widely used in the future. When an unknown piece of document is given, we may need to first determine whether the document has been hidden with information before we do any further analysis.

1.2 A history review

To understand steganalysis better, it is necessary to know the history of data hiding and steganalysis. The ideas of data hiding can be traced back to ancient times as the form of Steganography. The word “steganography” comes from Greek roots, literally meaning “cover writing”. Steganography is to hide the very existence of the message.

1.2.1 Steganography

As surveyed in [48], several examples of steganography can be found in the works of the ancient historian Herodotus [58]. One example is a man named Harpagus who killed a hare and hid a message inside its body. He sent it with a messenger who pretended to be a hunter. Another example is about how Histiaeus used slaves as carriers for a secret message: He shaved the head of his slave, tattooed the message on the shaved head, waited until the slave's hair grew back, and sent him along.

Aeneas the Tactician [58] proposed a method by putting almost invisible pin pricks above the letters of an innocuous message. This device was used all the way through the Renaissance, and even in World War I the Germans pricked letters in magazines. In other cases, they dotted letters with invisible ink which then had to be heated to show the plain text letters. In World Wars I and II, chemicals were developed and used as secret inks [58]. These chemicals reacted only with very specific other chemicals and the reaction made the secret ink visible. In the 21st century the governments began to use data hiding to protect their currency from being counterfeited. They have employed special

Figure 1.1: Modern steganography system.

inks, dyes, embedded threads and micro-strips which denote the face value of the bill.

In nowadays' digital world, invisible ink and paper are replaced by digital media such as digital document, image, video and audio [4, 8]. Digital media is much more versatile and practical to act as covers for hiding information.

The problems of steganography was formulated for the first time by Simmons as the prisoners' problem in [97]. A simplified framework of the problem is shown in Figure 1.1. In this scenario, Alice and Bob were in jail and wished to hatch an escape plan. Their communication passed through the warden, Wendy. If Wendy detected any suspicious messages, she would do something to frustrate their plan. So they must hide their cipher text in an innocuous looking way so that Wendy would not become suspicious. Robustness, capacity [18], security, and distortion are the major concerns for a steganography system. Limitations of steganography are discussed in [3].

1.2.2 Steganalysis

Steganalysis plays the role of the warden, Wendy. Wendy wants to know if the signal contains hidden message or not. Visual inspection may not be enough if the embedding does not introduce visible distortion. Thus, Wendy resorts to steganalysis techniques. By using steganalysis, Wendy may detect the covert communication that cannot be detected through visual inspection. On the other hand, the fact that Wendy may use steganalysis techniques makes it involuntary for Alice and Bob to try to find out or propose an embedding technique that the current steganalysis technique cannot detect hidden messages. To this purpose, Alice and Bob may need to understand how the steganalysis works in order to avoid detection. We can see that steganalysis is important and necessary for both sides.

Steganalysis is a relative new area of research. So far, the majority of the work in steganalysis are for color or gray level image, video, and audio, etc. Detection of LSB techniques may be the most successful and attractive. A series of steganalysis techniques [29–31, 35–38, 107] have been proposed for LSB embedding techniques by various authors. Pfitzmann and Westfeld introduced a method based on statistical analysis of pairs of values (PoVs) exchanged during message embedding [107]. Pairs of colors that differ in the LSB only, for example, could form these PoVs. This method provides reliable results when we know the message placement (such as sequential). RS steganalysis [35] is proposed by Fridrich *et al.* The image is partitioned into groups of pixels. Each group G is classified as a “Regular” or “Singular” pattern depending on whether the defined discrimination function $f(G)$ is increased or decreased after flipping the LSBs of a fixed set of pixels within the group. The numbers of

regular and singular groups are used to estimate the number of flipped pixel. Another technique by sampling pairs [30, 31] is proposed by Dumitrescu *et al.* for the detection of LSB flipping in continuous tone image. Some sets are defined. Random LSB flipping causes transitions between these sets and alters the relations between their cardinalities. It estimates the embedded message length with high precision. There are also other steganalysis techniques using image statistics [32, 33, 77–79], image quality metrics(IQM) [6], etc. Also, some mathematical frame work can be found in [16, 17].

Although many schemes have been proposed for LSB steganalysis, these techniques are all making use of the fact that the LSB embedding techniques would usually break certain kind of conditions such as smoothness to detect the existence of the hidden message. Moreover, these techniques cannot be applied to binary images which do not have the same smoothness condition as color or gray images. In recent years, researchers start to propose steganalysis techniques for binary images [53, 54]. It is reported in [54] that the compression data rate of clean text-font images would be increased by data hiding and the detection of hidden data in text images can be done by examining the compression data rate. However, this scheme fails for scanned text images. Steganalysis of halftone images is introduced in [53], however, only ordered dithering halftone images are tested. Steganalysis of halftone images generated by other halftoning methods have not been tested.

1.3 Objective and contributions

This thesis focuses on the steganalysis of binary images and also examines the development of a binary image distortion measure and its applications in steganalysis. It should be noticed that most binary data hiding schemes examined in the thesis were originally designed for authentication or annotation purpose, where statistical invisibility is not a concern in the design of these algorithms. One goal of this study is to conceal the visual as well as statistical existence of hidden data in case the existing embedding techniques are used for steganographic applications. Steganalysis could also be useful in forensic applications. It is expected that electronic documents would be widely used in the future. When an unknown piece of electronic document is given, we may need to first determine whether the document has been hidden with information (e.g., for annotation purpose) before we do any further analysis. The contributions of this thesis can be summarized as follows:

- Quantitative steganalysis of data hiding in clean text-font images is studied and an accurate way of message length estimation is proposed. It can defeat the normal pixel flipping technique over the clean text-font images. It actually shows that a clean text-font image is not a good cover signal for hiding information by flipping pixels.
- Detection of data hiding in noisy environment such as scanned text images is studied. Scan images are degraded from clean images or typed images. A systematic threshold selection method is introduced. The patterns in binary images are categorized according to their properties and are used in the detection method.

- Steganalysis technique for binary graphic image using image distortion measures is proposed. The use of an image distortion measure in steganalysis is studied. Two de-noising process for binary graphic images are proposed to generate the reference image, which is used as the reference image to calculate the distortion scores being used in the steganalysis.

- Steganalysis of halftone images is studied and a general method for steganalysis of halftone images is proposed. The idea is to apply inverse halftoning process to reconstruct the gray level images, whose image statistics are used later for steganalysis. Different inverse halftoning process and their applications in steganalysis are investigated and compared.

- A new objective distortion measure is also proposed. The concept of edge line segment is introduced and the relationship between edge line segment and visual distortion is explored. A novel way to quantify the damage to the edges from the flipping is proposed for binary text and binary graphic images. Its advantages over other measures are justified through subjective testing. The proposed distortion measure is applied in the detection of hidden message in a binary cartoon image.

1.4 Outline of the thesis

The rest of thesis is organized as follows:

Chapter 2 An overview of data hiding and steganalysis techniques in binary images are briefly described.

Chapter 3 Detection of hidden data in text image including clean text and scanned text image is discussed. For clean text-font image, which is directly

generated by using a software in a computer, its property is used and two ways to estimate the message length are discussed and compared. For scanned text image, a feature based on mark similarity is extracted and two different thresholds are used in the detection. The performances obtained using two different thresholds are also compared.

Chapter 4 Detection of hidden data in binary graphic image as well as an objective distortion measure for binary images are presented. Based on the concept of edge line segments, a distortion measure for binary image is proposed. Edge line segment similarity between the edge lines in the original and distorted image is calculated to derive the distortion score. The application of distortion measure in steganalysis is explored and two de-noising processes are developed to obtain the reference image for distortion score calculation. The calculated distortion scores are used as features for classifying stego image from original image.

Chapter 5 Detection of hidden data in halftone images is presented. A general method is proposed by applying inverse halftoning process into the steganalysis. Different inverse halftoning processes are used to obtain the reconstructed image for the detection and the performance of the detectors are compared.

Chapter 6 The conclusions are presented and future research directions related to this work are recommended.

Chapter 2

Literature Review

2.1 Steganography and steganalysis for binary document images

For a long time when people study data hiding in digital images, they focus on color or gray level images. Many data hiding schemes have been proposed for color or gray level images such as the work in [69, 80, 84, 90, 103, 106], while not many research work can be found in binary document image data hiding. It could be partly due to the fact that imperceptibility is more difficult to be achieved compared with color or gray level images. For the color or gray level images, we may change the values of a selected group of pixels by a small amount without causing visually noticeable artifact. For example, we can hide information in LSB in color or gray level image. However, with only two colors in binary images, any change can only be done by flipping the pixel from one color to the other color, i.e., black to white or vice versa. It is equivalent to changing the pixel value from 0 to 255 or vice versa in an 8-bits gray level

image. We can see that the flipping of a pixel in a binary image causes a much larger distortion than the flipping of a LSB in an 8-bits gray level image. Thus, arbitrarily flipping pixels in a binary image could cause very noticeable artifacts. In recent years, the increasing usage of digital documents such as electronic transcript, certificate, scanning hard copy to electronic copy etc. makes the security of digital document images important. Data-hiding in document images have received much attention recently. Many techniques have been developed for the protection of binary document images. Line and word shifting techniques and pixel flipping techniques are the two basic types of techniques for hiding information in binary text images.

2.1.1 Line and word shifting for text images

The first approach is to change the values of a group of pixels. In these methods, data hiding in a text image is done by manipulating the line, word or character by shifting the line, word, or character space [9–12, 24, 25, 71, 73, 82]. These schemes are developed mainly for the purpose of copyright protection etc. based on the fact that human visual perception is not sensitive to small modifications on a line, word or character space. However, the capacity of these schemes is very limited. A discussion on the capacity about these approaches is given in [72]. Also, line and word shifting techniques are usually not suitable for scanned text images. This type of embedding techniques can also be used for other purposes such as authentication like the work in [116], which integrates word space and character space to hide information for authentication.

For line or word shifting techniques, the existence of a secret message can be easily detected by examining the line, word or character space. Thus, in

earlier work [73], changing space is used by Low *et al.* for watermarking, where the watermark is inserted to be detected. The detection of hidden message by changing space can be done by the the extraction algorithm. Moreover, these techniques have relatively small embedding capacity, hence the utility for steganography is limited.

2.1.2 Pixel flipping technique for text & graphic images

Another type of techniques for binary image data hiding is to change the value of individually selected pixels, such as the work in [21, 74, 83, 88, 101, 112, 113, 117, 119]. Different from the line, word or character shifting techniques, these techniques hide information in the image by flipping individually selected pixels. Here, flipping means changing the pixel from white to black or vice versa. Thus, we call these techniques as pixel flipping techniques. Perceptual quality is controlled in these pixel flipping techniques to avoid large visible distortions. Most of these pixel flipping techniques manage to only modify pixels along edges because the flipping of non-edge pixels is likely to cause severe visual distortion. The major difference between the line, word, or character shifting technique and the pixel flipping technique lies in the fact that the former changes a group of pixels while the latter flips individual selected pixels. We give below an overview of several typical pixel flipping techniques.

In the scheme [112, 113] by Wu *et al.*, the binary image is shuffled and then divided into non-overlapping blocks. The information is hidden in each block by enforcing the feature of the image block such as the odd-even feature of the number of black pixels in the block. In order to minimize the visible distortion, a flippability score computation method which considers the smoothness and

connectivity of the image is proposed to determine the flippable pixels, i.e., the pixels by flipping which introduce less visual distortion. The smoothness is measured by the horizontal, vertical, and diagonal transitions in a local window (e.g., 3×3), and the connectivity is measured by the number of the black and white clusters. By combining the smoothness and connectivity together, the authors can order all 3×3 patterns in terms of how unnoticeable the change of the center pixel will be. Details of the flippability score computation can be found in [112]. The pixel with highest flippability score in each block is used for flipping. Shown in Figure 2.1 is an example to show the visual effect of the data hiding in binary image. In Figure 2.1(c), the gray pixels represent original pixels and the black pixels represent the flipped pixels.

In the scheme [83] by Mei *et al.*, the contours of an image are extracted and divided into a set of consecutive non-overlapping five-pixel-long segments. Later on, the information bits are hidden in those segments. By using several rules, a total of 100 different pairs of five-pixel-long patterns are considered as “valid” segments. In each pair, there are an ‘Add’ pattern and a ‘Delete’ pattern which differ by one pixel only. Flipping the center pixel from an ‘Add’ pattern will result in the corresponding ‘Delete’ pattern and vice versa. In order to minimize the distortion, only those segments that match one of the 200 “valid” segments are used to hide information. The center pixel of the segment is used as the flippable pixel for flipping. It is easy to hide one bit in a “valid” segment, for example, one Add pattern represents bit 0 and one Delete pattern represents bit 1.

In the scheme [117,119] by Yang and Kot, a set of rules are used to determine flippable pixels. These rules are proposed based on the “VH transitions”, “IR

Figure 2.1: Visual effect of embedding.

transitions” and “C Transitions” as defined in [117]. The image is divided into blocks. Different from Wu’s scheme, the overlapping block mechanism is proposed to increase the capacity and the information bits are only hidden in those embeddable blocks.

It can be seen that the pixels which are chosen for flipping by Wu’s scheme [112] are not easy to relocate after data hiding even if the algorithm is known to public as shuffling is applied. In Mei’s scheme and Yang & Kot’s scheme, the locations of the pixels for hiding information (50% chance of flipping) can be relocated assuming the knowledge of the embedding algorithm. The disadvantage of Wu’s scheme is that it usually needs a block size no smaller than 8×8 or even 16×16 . Otherwise, it may be difficult to guarantee that most of the blocks contain one flippable pixel after random shuffling. The schemes in [83, 117] can avoid this problem as they only hide information in patterns or blocks that contain flippable pixels. The need of a relatively larger block size limits the efficiency of Wu’s scheme in making use of the flippable pixels and thus limits the capacity. As a result, Wu’s scheme usually has a smaller capacity than Yang & Kot’s scheme. In a recent work [111], Wu *et al.* further improved their scheme based on the idea of wet paper codes [40] and provide a

more flexible data hiding scheme with higher capacity.

The scheme proposed in [21] significantly improves the data hiding capacity by using a weighting matrix. By using a weighting matrix, different pixels at different locations are assigned with different weights, the desired feature to embed the message bits is obtained by flipping pixels from different locations. It is shown in [21] that for each $m \times n$ block in the host image, the block can accommodate as many as $\lfloor \log_2(mn + 1) \rfloor$ bits of secret data by changing at most 2 pixels in the block. This is comparatively much more efficient than the techniques shown in [83, 112, 117], which can hide at most one secret bit in each block. The feature used in this scheme is quite different from the odd-even feature by other schemes. For an image block B , the feature is computed as $S = SUM((B \oplus K) \otimes W)$, where K and W are the key matrix and weight matrix with the same size as the block B respectively. The $\oplus$ denotes the bitwise exclusive-OR operation of two equal-size binary matrices. The $\otimes$ denotes the pairwise multiplication operation of two equal-size matrices and SUM means the sum of all elements in a matrix. By flipping at most two pixels from the block, B' can be obtained such that $SUM((B' \oplus K) \otimes W) \equiv b_1 b_2 \dots b_r \pmod{2^r}$, where $b_1 b_2 \dots b_r$ represents the r bits information to be hidden in the block. Details on how to select the two pixels as well as the use of the key matrix and the weight matrix can be found in [21].

The technique in [21] caused serious distortion. Another scheme is proposed in [101] which is an improvement over [21] in terms of the visibility of the hiding effect. By taking into consideration the quality of the image after hiding, the scheme can make the hiding effect more invisible. It ensures that for any pixel that is modified in the original image, the pixel must be adjacent to another

Figure 2.2: General methodology for binary image data hiding.

pixel that has the same value as the flipped pixel. This can be achieved by sacrificing some data hiding capacity.

As a summary, these pixel flipping techniques normally contain two important steps. The first step is to look into how to select a pixel by flipping that introduces the least distortion. The flippability of a pixel is determined by its neighboring pixels in a 3×3 , 5×5 or even larger block. Although different authors come out with different rules to determine flippable pixels, the effects are similar. The next step is to make use of the flippable pixels to maximize the capacity under the constraint that the hidden information can be extracted. Usually, the extraction should not need any knowledge of the original image or any side information that is dependent on the original image. A simplified framework for data hiding in binary images is given in Figure 2.2. Besides the schemes mentioned previously, there are other data hiding schemes proposed in [65, 67, 70, 88, 93, 104, 115, 118, 120]. These schemes also follow a similar procedure to hide information in binary images. In a most recent work [47], Gou and Wu proposed to flip a group of pixels simultaneously, which has taken the relationship between two or more simultaneously flipped pixels into consideration to minimize the visual distortion.

After pixel flipping techniques had been developed by several authors, researchers started paying attention to the detection of hidden message in binary images. Several schemes [54, 55, 121] have been proposed to detect hidden messages in binary images.

In [54], the discrimination between the stego image and the cover image is done by examining the compression data rate [54]. The entropy of a binary image increases as we hide information in the image. Thus, an image with a hidden message has a larger lossless compression data rate. The current best compression method JBIG2 is used to compress the binary image and the size of the resultant JBIG2 image is the estimated compression data rate. This scheme has some shortcomings. Firstly, the compression data rate of the original image is approximately obtained and it is difficult to differentiate stego images from original images if the embedding rate is very low. Secondly, it relies on the knowledge of the embedding algorithm to estimate the hidden message length, although it does not need the knowledge of the embedding algorithm if the purpose is only for the detection of the existence of the message. The detection scheme assumes that using the same embedding scheme to hide information in an area which has already been hidden with secret information, the pixels from the same locations are selected for possible flipping. It is true for embedding algorithms similar to the one in [83], however, this may not be true for other embedding schemes. Finally, the approach is not applicable for scanned text images.

The approach in [55] uses the correlation between a pixel and its neighboring pixels. It is believed that the contours of characters or symbols are polynomial curves. Thus it is possible to estimate the parameters of the polynomial and then estimate the pixel value of an edge pixel according to its neighboring pixels. The parameters are obtained using a large training set of unmarked text image. Although there exists estimation errors for the original images, the flipping of pixels is likely to increase the errors thus leaves the hint to detection. The mean of the variance of the estimation error is approximately linear to the

message length. Thus, with an empirically selected threshold, the algorithm can differentiate a stego image from an original image as well as estimate the message length. The limitation of this approach is the empirical threshold selection. It is likely that different images with different fonts, resolutions etc. may indicate different initial errors. Thus, the accuracy of the detection method would be affected. As mentioned by the author, the polynomial parameters are fonts dependent also. This approach is not applicable for scanned text images.

The scheme in [121] is tested with cartoon images, however, the embedding approaches used the hide information is relatively earlier schemes which have not taken smoothness and connectivity [112] into consideration. It does not apply for the recent data hiding schemes [83, 112, 117] which hide information more invisibly.

2.2 Steganography and steganalysis for halftone images

Although a halftone image is also binary, it is usually obtained by halftoning a gray level image. It is a special type of binary image, different from other binary images such as text or graphic images. There are two basic ways to hide information in a halftone image. The first approach of data hiding in a halftone image is done by flipping the pixels in the halftone image from black to white or vice versa without the knowledge of the original gray level image. In [41, 42], data hiding self toggling (DHST), data hiding pair toggling (DHPT) and data hiding smart pair toggling (DHSPT) are proposed. In DHST, the information is hidden at pseudorandom locations in the halftone image by forcing the pixels

at these locations to be black or white. In DHPT, DHST is applied first and a complementary pixel is chosen randomly for flipping from the 3×3 neighborhood of each flipped pixel during DHST. By doing so, the local intensity is maintained. The DHSPT improves on DHPT by choosing the complementary pixel to achieve smaller salt and pepper clusters. Later in [43] the idea of data hiding with intensity selection is proposed to select the best location from a set of candidate locations for embedding each information bit. This idea is applied to DHST, DHPT and DHSPT to get three new algorithms: data hiding self toggling with intensity selection (DHST-IS), data hiding pair toggling with intensity selection (DHPT-IS) and data hiding smart pair toggling with intensity selection (DHSPT-IS). Generally, DHST is the basic scheme to hide information. It is also discussed in some other papers for multi-resolution binary image embedding [109] or integrity protection and authentication [66] by using DHST.

The second approach of data hiding in a halftone image is done with the knowledge of an original gray level image. Instead of flipping pixels, these data hiding methods usually modify the halftoning method to obtain the halftone image that contains hidden message. In [44], data hiding error diffusion (DHED) and modified data hiding error diffusion (MDHED) are proposed by modifying the error diffusion algorithm. In both methods, the original gray level image is known and the pixel values at the locations determined by the pseudorandom key are first determined according to the message to be hidden. Then the error diffusion algorithm is applied to obtain the binary output for the rest of the pixels. The error due to embedding is also considered in the error diffusion. In DHED, the embedding error is fed forward. In MDHED, the embedding error is fed both forward and backward.

Steganalysis of halftone images is discussed in [53]. In this scheme, an heuristically selected average low pass filter is used to filter the high frequency noise of a halftone image. Then the image statistics [33] of the filtered image is used as features for discriminating the stego image from original image. However, only ordered dithering images are tested. For other halftone images such as error diffusion halftone images, no results have been shown. Also, the performances by using other filters have not been investigated. In this thesis, we will look into these issues.

Chapter 3

Text Images

3.1 Introduction

Depending on the image's origin, binary text images can be categorized into two types. The first type of binary text images is directly generated by using a software in a computer. We call such an image a clean text-font image. The second type of binary text images is the scanned text images obtained by scanning hard copies into electronic copies. As we mentioned in Chapter 2, after pixel flipping techniques had been developed by several authors, researchers started paying attention to the detection of hidden messages in binary images. Several schemes [54,55] have been proposed to detect hidden messages in binary text images. As we have discussed in Section 2.1.2, the entropy based schemes in [54] relies on the knowledge of the embedding algorithm to estimate the hidden message length, moreover, it is not applicable for scanned text images. The correlation based scheme [55] is fonts dependent which implies the scheme needs some side information or a font recognition process to estimate the message

length. In this Chapter, we propose a steganalysis technique to estimate the message length hidden in clean text-font image without the above constraint and a steganalysis technique to detect hidden messages in scanned text images.

In Section 3.2, we present the steganalysis technique for clean text-font images. In Section 3.3, we present the steganalysis technique for scanned text images. The summary is given in the last section.

3.2 Clean text-font image

3.2.1 Observation

For clean text-font images which are converted from text files, the most important property is that all the symbols including characters and letters from the same origin (same symbol and same fonts) are identical after the conversion. Data hiding by flipping some pixels is likely to make those symbols from the same origin no longer identical. In this case, a straightforward way to differentiate a stego image from an original image is to compare the symbols. If we can find two symbols with the same origin not identical, e.g., the first two ‘e’ in Figure 3.1(b), then the image is likely to contain some hidden messages. For this purpose, we need to group the symbols such that all the symbols from the same origin are grouped into one group while the symbols from different origins are grouped into different groups. A general way to group them is by doing the fonts identification and the recognition of individual characters or symbols using an optical character recognition (OCR) software as mentioned in [55]. However, it is time-consuming and the accuracy of the fonts identification and OCR affects the result, not to mention the use of non-standard fonts. In this

section, we propose a simple way to group these symbols.

Obviously, two symbols with same origin should be highly correlated even if some pixels are flipped. When we compare them, the number of mismatched pixel pairs is always limited compared to the total number of pixel pairs, i.e., the percentage of mismatched pixel pairs is usually less than a certain threshold. However, it is difficult to find a perfect threshold in grouping. If the threshold is set too high, two symbols with different origins (but similar) may be grouped together such as ‘o’ and ‘c’, ‘i’ and ‘l’. If the threshold is set too low, the flipped pixels in the data hiding may make two symbols from the same origin to be grouped into different groups.

Recall that most of the pixel-flipping techniques [74, 83, 88, 112, 113, 117] only flip the boundary pixels to avoid large visible distortion. Moreover, these techniques normally do not flip two 8-connected neighboring pixels simultaneously, which implies that in any 2×2 square block in the image, no more than one pixel would be flipped. After data hiding, the same symbols usually no longer match pixel by pixel. However, they normally satisfy the following conditions when we compare them pixel by pixel:

1. The number of mismatched pixel pairs would be at most 2 in any 2×2 square window when they are aligned together as shown in Figure 3.1. It is because both of the square blocks which are from the same symbols originally may contain one flipped pixel.
2. The percentage of mismatched pixel pairs is usually less than a fixed threshold.

In order to get a higher precision in grouping, we make use of the above con-

Figure 3.1: Effect of embedding.

ditions to group them. By taking the first condition into consideration, we can easily differentiate different characters such as ‘o’ and ‘c’, ‘i’ and ‘l’ because they usually do not satisfy the condition.

3.2.2 The feature extraction

The idea of soft pattern matching was proposed to compress binary image in JBIG2 [50, 51]. We use the similar idea here to detect the existence of a hidden message in clean textfont images as well as to estimate the message length.

We first extract all the marks from the whole image. Here, we use the term ‘marks’ instead of the previously used ‘symbols’. Marks [51] refer to letters, ligatures, figures and punctuation symbols and other symbols. These marks can be easily extracted by standard segmentation technique [51]. In our implementation, we segment the image into lines according to the horizontal profile and

then segment each line into marks according to the vertical profile. We sort all the marks according to their locations in an image and then group the similar marks into the same groups as what we have done in [23].

After the grouping, we obtain groups of marks. We average all the marks in each group to get the average mark. The average mark is considered as an estimation of the original mark. Some groups may have only one mark in it, which means the mark in this group can only match itself. We label this as a unique mark.

Suppose a group contains marks M_i , where $i = 1, 2, \dots, N$. The pixel value (1 for black or 0 for white pixel) is represented by $M_i(x, y)$, where (x, y) is the displacement of the pixel from the top-left corner of the mark. Then the average pixel value at (x, y) is

$$avg(x, y) = round\left(\frac{1}{N} \sum_{i=1}^N M_i(x, y)\right) \quad (3.1)$$

where $round(\cdot)$ is the rounding operation to an integer. We then use the average mark as a reference and compare it with all the marks within the same group.

3.2.3 Detection and message length estimation

In each group, we compare all the marks in the group with the average mark. If $M_i(x, y)$ equal to $avg(x, y)$, then pixel $M_i(x, y)$ is a regular pixel, otherwise it is an irregular pixel. Images without irregular pixels are classified as original images, otherwise, stego images. For each group of marks, we first count the number of irregular pixels k_{xy} at location (x, y) and then estimate m_{xy} which is the number of pixels used to hide information among the N pixels $M_i(x, y)$, for

$i = 1, 2, \dots, N$. The aggregation of m_{xy} throughout all the pixels from all the marks results in the estimated hidden message length in the image.

We present two ways to estimate m_{xy} . In the first approach, we estimate m_{xy} by double the number of irregular pixels k_{xy} . It is based on the fact that the embedding algorithms have 50% chance to flip a selected pixel for hiding a random bit of a hidden message. However, we would get biased result due the the estimation errors in estimating the original marks and the message length is usually underestimated.

In the second approach, we estimate m_{xy} based on the probability distribution function of m_{xy} for the observed k_{xy} . Given k_{xy} , suppose the probability that m_{xy} equals to m is $P(m|k_{xy})$. Then m_{xy} can be estimated by $\sum_m mP(m|k_{xy})$. In what follows, it shows how we calculate $P(m|k_{xy})$. Suppose m of the N pixels $M_i(x, y)$, for $i = 1, 2, \dots, N$, are used to hide information. The probability of flipping k pixels is given as $(\frac{1}{2})^m \binom{m}{k}$. In the detection, the observed number of irregular pixels at (x, y) is k_{xy} . The relationship between k_{xy} and the actual number of flipped pixels k is given as:

$$k_{xy} = \min(k, \lfloor \frac{N}{2} \rfloor - k) \quad (3.2)$$

where $\lfloor \cdot \rfloor$ is the floor operation. Then the probability of obtaining k_{xy} irregular pixels after we hide the m bits at the m locations is given by:

$$P(k_{xy}|m) = \begin{cases} (\frac{1}{2})^m \binom{m}{k_{xy}} + (\frac{1}{2})^m \binom{m}{N-k_{xy}}, & 0 \leq k_{xy} < N/2 \\ (\frac{1}{2})^m \binom{m}{k_{xy}}, & k_{xy} = N/2 \\ 0, & \text{otherwise} \end{cases} \quad (3.3)$$

According to the Bayes' theorem,

$$P(m|k_{xy}) = \frac{P(m)P(k_{xy}|m)}{P(k_{xy})} = \frac{P(m)P(k_{xy}|m)}{\sum_n P(k_{xy}|n)P(n)} \quad (3.4)$$

where $P(m)$ is the probability that the message length is m . $P(k_{xy})$ is the probability that the number of irregular pixels is k_{xy} .

To compare the second message estimation method with the first one, we use m'_{xy} to denote the estimation result using the first method, and m^*_{xy} to denote the estimation result using the second method. We have

$$m'_{xy} = 2k_{xy} \quad (3.5)$$

$$m^*_{xy} = \sum_m mP(m|k_{xy}) \quad (3.6)$$

The estimated message lengths by the two methods are obtained through aggregating m'_{xy} and m^*_{xy} . The aggregation of m'_{xy} is straight forward as twice of the total number of irregular pixels. The aggregation of m^*_{xy} is more complicated. We need to calculate the occurrence of $k_{xy} = i$, for $i = 0, 1, \dots, N$, when there is a message hidden among the pixels $M_i(x, y)$, for $i = 1, 2, \dots, N$. When $k_{xy} \neq 0$, we consider that there is a hidden message and we calculate m^*_{xy} for the observed nonzero k_{xy} . When $k_{xy} = 0$, there is no hidden message. However, there is still a small chance that there is a hidden message although $k_{xy} = 0$. In our method, we estimate the occurrence of $k_{xy} = 0$ when there is a hidden

message by:

$$O(k_{xy} = 0) = O(k_{xy} \neq 0) \cdot \frac{P(k_{xy} = 0)}{P(k_{xy} \neq 0)} \quad (3.7)$$

where $O(k_{xy} \neq 0)$ is the occurrence k_{xy} not equal to 0. $P(k_{xy} = 0)$ and $P(k_{xy} \neq 0)$ are the probability $k_{xy} = 0$ and $k_{xy} \neq 0$ when there is a hidden message. Since we have no prior knowledge of the hidden message length, $P(k_{xy} = 0)$ and $P(k_{xy} \neq 0)$ are calculated under the assumption that the hidden message length among the pixels $M_i(x, y)$, for $i = 1, 2, \dots, N$, can be any number from 0 to N with equal probability.

3.2.4 Experimental results

We tested 100 images with different computer generated fonts. The size of these images are A4 size (International Standards Organization (ISO) standard-size typing paper, 210 mm wide and 297 mm tall). By using the soft pattern matching, marks with different fonts can be grouped into different groups as they do not satisfy condition 1 or 2. Although the same marks always satisfy the two conditions discussed previously, there is no guarantee that any two marks which satisfy the two conditions must be originally identical. We have observed two such marks in our testing. The comma (,) and the quotation mark (') may differ by 1 pixel for certain resolution. We can exclude these marks as their size is smaller than an alphabet. We use the schemes in [112] and [83] to embed random bits in the images at different embedding rates R (actual-message-length/maximum-message-length) at 0.1, 0.2, $\dots$, 1.0 and at 0.01, 0.02 and 0.05. We can differentiate all the stego images from the original images. We further estimate the hidden message length l_e in the stego images. The estimation error

(a) Estimation errors of 100 Images for Wu's scheme.

(b) Estimation errors of 100 Images for Mei's scheme.

Figure 3.2: Estimation of embedding rate for different embedding schemes: The solid line represents the mean error; the dotted line and the dash-dotted line represent the maximum positive or negative errors.

Figure 3.3: Examples of ‘Super’ pixels (in black) in [47]. Information can be hidden in the image by changing or not changing the colors of the lines shown in black.

of the hidden message length is defined as $\Delta l = l_e - l = l_e - RC$, where C is the maximum message length. The estimation errors at different embedding rates by the two estimation methods for the two embedding approaches are shown in Figure 3.2. The solid line represents the mean error; the dotted line and the dash-dotted line represent the range of the errors. From the results we can observe that the second hidden message length estimation method provides better estimation results than the first one.

Although the above scheme is proposed for detecting individually flipped pixels, it can be extended for other embedding scheme such as the most recent work in [47]. The scheme in [47] considers the black lines shown in Figure 3.3 as ‘Super’ pixels. These black lines can be flipped to hide information. Let’s say we change some of the black lines of the first letter ‘n’ in Figure 3.3, then it is likely this ‘n’ is no longer identical with other ‘n’s in the same image pixel by pixel.

From the above examples, we can see that hiding data in a clean text-font

image by flipping the pixels is not a good choice unless the pixels belong to a unique mark that appears only once.

3.3 Scanned text image

Besides conversion from a text file, scanning from a hard copy to an electronic copy is another way to generate an electronic text image. Different from clean text-font images where the marks from the same origin are identical, the marks in the scanned images are usually different from each other. As a result, the above method does not work for scanned image. Similarly, the methods in [54, 55] does not work for a noisy scanned image either.

Detection of the existence of a secret message in noisy binary images without a prior knowledge of the embedding algorithm is a difficult problem. One reason is that the locations where the pixels are selected for possible flipping cannot be easily located. The second reason is that the existing noisy pixels in the original scanned image work as a cover of the newly introduced pixels through the embedding process, while the noise level in the original scanned image determined by the scanner may vary largely from an image to another image. The third reason is that the scan noise overwhelms the embedding noise very often. These reasons make it challenging to detect the existence of the embedding noise. The scan noise makes the marks from the same origin no longer identical, however, they are still highly correlated. In this section, we propose a method to detect the existence of a secret message hidden in a scanned text image without detail knowledge of the embedding algorithms by utilizing the similarity of the same characters or symbols within the same text image.

Figure 3.4: An example of L -shape pattern.

3.3.1 Common features in embedding process

To start with, we give a review of the embedding schemes by studying how the embedding algorithms preserve the quality of the images during the embedding process. The authors in [83,88,112,113,117] have studied the flippability of each pixel by comparing it with its neighboring pixels. A flippability score [112] is assigned based on the smoothness and connectivity within 3×3 neighborhoods. In [83], 100 pairs of boundary patterns are chosen with the goal to preserve the overall shape of a character by minimizing noticeable artifacts and distortion. In [117], connectivity is preserved when flipping pixels. The information is hidden by flipping some ‘flippable’ pixels. One of the important observations is that most of these embedding schemes use the center pixel of an L -shape pattern (COL pixel) as shown in Figure 3.4. There is a total of 16 L -shape patterns after taking complement, mirroring and rotation into consideration. Very often, COL pixel is chosen in most of these schemes as flippable pixel to avoid affecting the smoothness and connectivity of the contours [22].

We define set A as the set consisting of all COL pixels. Most embedding schemes choose pixels from the set A for flipping. Some schemes may choose a small number of additional pixels which do not belong to set A . In our steganalysis, no prior knowledge of the embedding process is known, except that these COL pixels are assumed to be the best candidates to be chosen for

flipping to preserve the minimum distortion. In our analysis, we detect the existence of a secret message by studying the difference of the flipped pixels in set A within a character or a symbol in a scanned text image.

3.3.2 The feature extraction

First, we segment the whole image into different marks similar to what we did in Section 3.2. These marks are sorted according to their locations in the image document and the following steps are carried out.

1. Start from the first mark as the current mark.
2. Prescreen all the marks sorted after the current mark. Skip if the mark size is not close to that of the current mark (say the mark size in either dimension differs by more than 2 pixels compared with that of the current mark). Badly distorted marks are excluded.
3. For every potential matched mark, find the alignment with the current mark that gives a minimum number of mismatched pixels and calculate the number of mismatched pixels. If the percentage of mismatched pixels is less than a predefined threshold T , the potential matched mark is considered as a match to the current mark. Inspired by the JBIG2 standard [50, 51], we use T as 21% of the pixel number enclosed in the bounding box of the current mark. All the marks matched to the current mark are grouped together. Marks that do not match any other marks are not included.
4. Select the first mark of the remaining unmatched marks as the new current mark. Repeat Step 2 and 3 until all the possible marks have been grouped.

In Step 3, the alignment is done by first calculating the center of gravity of the mark and followed by searching for the best alignment within a $w \times w$ window centered at the gravity point. For an $n \times m$ binary region B with its pixel value $B(i, j)$, for $i = 1, 2 \dots n$, $j = 1, 2 \dots m$, the center of gravity point (x_c, y_c) is computed using

$$x_c = \text{round}\left(\frac{\sum_{i=1}^n \sum_{j=1}^m iB(i, j)}{\sum_{i=1}^n \sum_{j=1}^m B(i, j)}\right), \quad y_c = \text{round}\left(\frac{\sum_{i=1}^n \sum_{j=1}^m jB(i, j)}{\sum_{i=1}^n \sum_{j=1}^m B(i, j)}\right) \quad (3.8)$$

where $\text{round}(\cdot)$ is the rounding operation to an integer. A reference mark is calculated for every group of marks created. Similar to what we have done in Section 3.2, we consider a group of N similar marks M_i , for $i = 1, 2, \dots, N$, the average pixel value of the average mark Ω_M at (x, y) is obtained using (3.1).

We use the average mark Ω_M as a reference and compare the marks within the group with the reference. The pixel value $\Omega_M(x, y)$ of the reference Ω_M is invariant with high confidence when we flip pixels to hide information since it is obtained by averaging all the marks in each group. However, $M_i(x, y)$ is directly affected by the pixels being flipped in the embedding process and the chance of occurrence of an $M_i(x, y)$ which is not equal to $\Omega_M(x, y)$ will be higher after the embedding process. As we defined before, a pixel at (x, y) from the mark M_i is an irregular pixel if $M_i(x, y) \neq \Omega_M(x, y)$, otherwise, it is defined as a regular pixel. Flipping a regular pixel usually converts it to an irregular pixel or vice versa. We define the COL ratio ξ_0 :

$$\xi_0 = k_0/n_0 \quad (3.9)$$

Figure 3.5: Examples of isolated and non-isolated COL pixels.

where n_0 is the sum of n_b black and n_w white COL pixels and k_0 is the sum of k_b black and k_w white irregular pixels. Experimentally, the COL ratio ξ_0 is found to be $1/3$ approximately, for machine print database (MPDB) images. The value of ξ_0 increases after embedding. However, it is difficult to differentiate stego images with original images based only on ξ_0 , as the change due to the embedding is not significant when compared with the original noise level. By subtracting off the original noise, ξ_0 will become a more dominant feature due to embedding.

We first partition set A of all COL pixels into two subsets A_I and A_N , where A_I consists of all isolated COL pixels as shown in Figure 3.5(a) and A_N consists of all non-isolated COL pixels as shown in Figure 3.5(b). A COL pixel is isolated if none of its 8-connected neighboring pixels is a COL pixel; otherwise, it is defined as non-isolated.

We extract a subset $A_{N,N}$ from A_N , where the element of $A_{N,N}$ is the pixel $a_{N,N}$ which remains as a non-isolated COL pixel when it is flipped. Similar to ξ_0 , we define $\xi_I = k_I/n_I$ and $\xi_{N,N} = k_{N,N}/n_{N,N}$, where k_I and $k_{N,N}$ are the numbers of irregular pixels as we defined earlier in this Section in set A_I and

$A_{N,N}$ respectively. n_I and $n_{N,N}$ are the numbers of pixels in set A_I and $A_{N,N}$ respectively. The value of $\xi_I - \xi_{N,N}$ increases in the embedding process as more COL pixels are used to hide information. Similarly, the original noise level will affect ξ_I and $\xi_{N,N}$. The effects on the two items tend to cancel out with each other and the change of $\xi_I - \xi_{N,N}$ from image to image due to the original noise level is largely reduced. Thus, the difference between ξ_I and $\xi_{N,N}$ can be used to differentiate the stego images from original cover images.

It should be noticed that the number of regular and irregular pixels are related with the noise level of the image. Intuitively, more irregular pixels mean more noise in the image and the vice versa. For example, a clean text-font image should have no irregular pixels while a scanned image should have more irregular pixels. As data hiding introduces more noise, which will affect the number of regular and irregular pixels, we are interested with these pixels. We use ξ_I and $\xi_{N,N}$ to represents the percentage of irregular pixels from two groups of COL pixels, i.e., the isolated COL pixels and non-isolated COL pixels. The value $\xi_I - \xi_{N,N}$ represents the difference of the percentages of irregular pixels from the isolated COL pixels and non-isolated pixels. This difference increases as more data are hidden in the image.

3.3.3 Theoretical analysis

In this section, we show why the value of $(\xi_I - \xi_{N,N})$ increases after data hiding and provide an estimation of such an increase analytically.

The flipping of a COL pixel usually makes its 8-connected neighboring COL pixel (if exists) no longer a COL pixel while the flipped pixel itself is still a COL pixel. At the same time, the flipping of a COL may make one of its neighboring

Figure 3.6: The effect of flipping a COL pixel.

Figure 3.7: The diagram of each subset division.

non-COL pixels become a COL pixel. For example in Figure 3.6, the flipping of pixel b will make the non-COL pixel c become a COL pixel and make the COL pixel a no longer a COL pixel, while pixel b still remains a COL pixel after the flipping.

Subsets of COL pixels We further divide set A_I and A_N as shown in Figure 3.7 into two subsets $A_{I,I}$ and $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$ respectively. In Figure 3.8, we show some examples of pixels which satisfy the definition of sets $A_{I,I}$, $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$. For example, in Figure 3.8(a), the center pixel of the 3×3 square is an isolated pixel. When this pixel is flipped, it still remains as an isolated COL pixel; thus it belongs to the set $A_{I,I}$.

Figure 3.8: Examples of pixels from sets $A_{I,I}$, $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$.

We assume the numbers of elements in the different subsets $A_{I,I}$, $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$ in the original scanned image are $n_{I,I}$, $n_{I,N}$, $n_{N,I}$ and $n_{N,N}$ with the corresponding numbers of irregular pixels $k_{I,I}$, $k_{I,N}$, $k_{N,I}$ and $k_{N,N}$, respectively. We have:

$$n_I = n_{I,I} + n_{I,N} \quad (3.10)$$

$$k_I = k_{I,I} + k_{I,N} = n_{I,I}\xi_I + n_{I,N}\xi_I \quad (3.11)$$

After embedding, the original sets $A_{I,I}$, $A_{I,N}$, $A_{N,I}$ and $A_{N,N}$ are changed and new sets $A_{I,I}^*$, $A_{I,N}^*$, $A_{N,I}^*$ and $A_{N,N}^*$ are created with the corresponding number of pixels and number of irregular pixels denoted by $n_{I,I}^*$, $n_{I,N}^*$, $n_{N,I}^*$, $n_{N,N}^*$, $k_{I,I}^*$, $k_{I,N}^*$, $k_{N,I}^*$ and $k_{N,N}^*$ respectively. The feature used to compare with the threshold is computed using $\xi_I^* - \xi_{N,N}^* = \frac{k_I^*}{n_I^*} - \frac{k_{N,N}^*}{n_{N,N}^*}$. In what follows, we calculate the numbers of irregular pixels and total pixels in the three subsets $A_{I,I}^*$, $A_{I,N}^*$ and $A_{N,N}^*$ in the stego images. From the definition of those subsets, we can see that the status of a COL pixel at (u, v) (which subset it belongs to) can be determined by the 5×5 block centered at pixel (u, v) . We first consider the effects by flipping a COL pixel in the 3×3 block centered at (u, v) .

New set $A_{I,I}^*$ An isolated COL pixel in the set $A_{I,I}$ is the only COL pixel in the 3×3 block centered at this pixel and it will be an isolated COL pixel in set $A_{I,I}^*$ in the stego image except that the pixel value may change, so we have:

$$n_{I,I}^* = n_{I,I} \quad (3.12)$$

In the stego image, the number of irregular pixels $k_{I,I}^*$ in set $A_{I,I}^*$ is given by:

$$k_{I,I}^* = k_{I,I}(1 - \rho_I) + n_{I,I} \frac{\rho_I}{2} \quad (3.13)$$

where ρ_I is the probability an isolated COL pixel being selected for hiding information.

New set $A_{I,N}^*$ The new set $A_{I,N}^*$ consists of the elements from $A_{I,N}$ which are not flipped and the elements from $A_{N,I}$ which are flipped. Thus $n_{I,N}^*$ is given by:

$$n_{I,N}^* = n_{I,N} - n_{I,N} \frac{\rho_I}{2} + n_{N,I} \frac{\rho_N}{2} \quad (3.14)$$

where ρ_N is the probability an non-isolated COL pixel being selected for hiding information. The number of irregular pixels $k_{I,N}^*$ in $A_{I,N}^*$ is given by:

$$k_{I,N}^* = n_{I,N} \left(1 - \frac{\rho_I}{2}\right) \xi_I + n_{N,I} \frac{\rho_N}{2} (1 - \xi_N) \quad (3.15)$$

New set $A_{N,N}^*$ As for the new set $A_{N,N}^*$, it contains at least two COL pixels in the 3×3 block. A pixel $a_{N,N}$ in $A_{N,N}$ will become a pixel in $A_{N,N}^*$ unless its neighboring COL pixel is flipped. On the other hand, flipping a pixel in $A_{I,N}$

or $A_{N,N}$ will make a non-COL pixel become a non-isolated COL pixel in the neighboring pixels of the flipped pixel in the stego image. The newly created COL pixel in the stego image could be an element of $A_{N,N}^*$. We can treat the whole change as: all elements in $A_{N,N}$ move to a temporary $A_{N,N}^*$ and then the temporary $A_{N,N}^*$ exchange with non-COL pixels to form the final set $A_{N,N}^*$. Suppose the exchanging affects the number of elements in $A_{N,N}^*$ by $\Delta n_{N,N}$, then the number of elements in $A_{N,N}^*$ is given by:

$$n_{N,N}^* = n_{N,N} + \Delta n_{N,N} \quad (3.16)$$

Since the exchanged pixels are all original pixels in the scanned images close to or on the edge, their probability (denote as η) to be irregular pixels is close to that of COL pixels. Similar to $k_{I,I}^*$, we have the number of irregular pixels $k_{I,I}^*$ in set $A_{I,I}^*$:

$$k_{I,I}^* = n_{N,N} \xi_{N,N} - n_{N,N} \rho_N \xi_{N,N} + n_{N,N} \frac{\rho_N}{2} + \Delta n_{N,N} \eta \quad (3.17)$$

Combining (3.16) with (3.17) yields:

$$\begin{aligned} \frac{k_{N,N}^*}{n_{N,N}^*} &= \left(\xi_{N,N} (1 - \rho_N) + \frac{\rho_N}{2} \right) \\ &+ \frac{\Delta n_{N,N}}{n_{N,N} + \Delta n_{N,N}} \left(\eta - (1 - \rho_N) \xi_{N,N} - \frac{\rho_N}{2} \right) \end{aligned} \quad (3.18)$$

Since $\Delta n_{N,N} \ll n_{N,N}$ and $|\eta - \xi_{N,N} + \rho_N \xi_{N,N} - \frac{\rho_N}{2}| \ll 1$, as a result, (3.18) can be reduced to:

$$\frac{k_{N,N}^*}{n_{N,N}^*} = \left(\xi_{N,N} (1 - \rho_N) + \frac{\rho_N}{2} \right) \quad (3.19)$$

New $\xi_I^* - \xi_{N,N}^*$ Since most of the isolated COL pixels become non-isolated COL pixels after being flipped, it can be shown experimentally that $n_{I,I} \ll n_{I,N}$. Combining (3.12)-(3.15) with (3.19) and ignoring $n_{I,I}$ yields:

$$\begin{aligned}
 \xi_I^* - \xi_{N,N}^* &= \frac{k_I^*}{n_I^*} - \frac{k_{N,N}^*}{n_{N,N}^*} \\
 &= \frac{n_{I,N} \left(1 - \frac{\rho_I}{2}\right) \xi_I + n_{N,I} \frac{\rho_N}{2} (1 - \xi_N)}{n_{I,N} - \frac{n_{I,N}\rho_I}{2} + \frac{n_{N,I}\rho_N}{2}} \\
 &= \frac{n_{N,N}\xi_{N,N} - n_{N,N}\rho_N\xi_{N,N} + n_{N,N}\frac{\rho_N}{2}}{n_{N,N}} \quad (3.20)
 \end{aligned}$$

For the existing embedding schemes in [83, 88, 112, 113, 117], we can assume the 50% chance a COL pixel being selected for possible flipping is the same as ρ , i.e., $\rho_I = \rho_N = \rho$. Substituting ρ_I and ρ_N with ρ and simplifying $f(\alpha, \rho) = (\xi_I^* - \xi_{N,N}^*) - (\xi_I - \xi_{N,N})$ yields:

$$f(\alpha, \rho) = \frac{\alpha\rho(1 - \xi_N - \xi_I)}{\alpha\rho + 2 - \rho} + \xi_{N,N}\rho - \frac{\rho}{2} \quad (3.21)$$

where $\alpha = n_{N,I}/n_{I,N}$.

For a specific image, α in $f(\alpha, \rho)$ is fixed and can be computed separately. The values of ξ_I , ξ_N and $\xi_{N,N}$ approximately equal to ξ_0 which is around 1/3 experimentally. In Figure 3.9, we show the curves for different α based on (3.21) with ξ_I , ξ_N and $\xi_{N,N}$ assumed to be 1/3.

In the above analysis, we have not considered the influence to a pixel from the flipping of COL pixels outside the 3×3 block and within the 5×5 block centered at this pixel. We find that we can ignore their influence experimentally for the following two reasons. The first reason is that the chance that there exists a

Figure 3.9: The curves of theoretic increase.

COL pixel in this area is small, not to mention the probability that it is selected and flipped. The second reason is that even if we flip this pixel, the chance of affecting the status of the center pixel is much smaller than that when we flip a pixel inside the 3×3 block. In summary, the pixels within the 3×3 block play the key roles to determine the trend that $(\xi_I - \xi_{N,N})$ increases after the flipping of COL pixels. In order to see the difference between the theoretical increase and the actual increase, we use Yang and Kot's scheme [117, 119] (3×3 overlapping window) to generate stego images with different ρ from 0.0 to 0.25 for all the test original images. We calculate the average actual increase using these generated stego images. We also obtain the theoretical average increase for these images. In Figure 3.10, we show both the actual average increase and the theoretical average increase.

Figure 3.10: The theoretical increment from $\xi_I - \xi_{N,N}$ to $\xi_I^* - \xi_{N,N}^*$ vs. the actual increment.

3.3.4 Threshold selection

We compare $(\xi_I - \xi_{N,N})$ with a threshold ϑ . If $\xi_I - \xi_{N,N} > \vartheta$, the image is classified as a stego image, otherwise, it is a clean image. The threshold being used is important. We provide two choices. The first choice of the threshold is a fixed empirically selected threshold. The limitation of a fixed threshold is that we may need different thresholds for different sets of database images. In what follows, we study how to find an image dependent threshold ϑ to differentiate stego images from the original cover images such that most of the original images will not be wrongly classified as stego images. For this purpose, we need to study the value of $(\xi_I - \xi_{N,N})$ for the original scanned images.

Origin of a scanned image We start from the origin of a scanned image. The scanned image can be considered as an image degraded from a corresponding clean image by flipping some pixels from the clean image [60]. For example, the scanned image shown in Figure 3.11(b) can be considered as an image de-

Figure 3.11: An original scanned image and a clean image.

graded from the clean image shown in Figure 3.11(a). A model to describe the flipping of the degradation was discussed in [60] and the validation of the model was discussed in [59]. For convenience, we use I_{scan} and I_{clean} to denote the original scanned image which is used to hide information and the clean image which the original image is degraded from, respectively. During the degradation from I_{clean} to I_{scan} , different pixels may have different probabilities of being flipped. Following the degradation model in [60], we assume that a prior probability to flip a white pixel to black is p_{wb} and a black pixel to white is p_{bw} . The relationship between these two probabilities and the two ratios $\frac{k_b}{n_b}$ and $\frac{k_w}{n_w}$ are important. When p_{wb} and p_{bw} are much smaller than 0.5, the average pixel value $\Omega_M(x, y)$ computed using equation (3.1) normally equals to $I_{clean}(x, y)$. In such a scenario, a pixel being flipped during the scanning usually becomes an irregular pixel in the scanned image, which implies larger p_{wb} and p_{bw} corresponding to larger $\frac{k_b}{n_b}$ and $\frac{k_w}{n_w}$, respectively. Because of the difference between p_{wb} and p_{bw} , the two ratios $\frac{k_b}{n_b}$ and $\frac{k_w}{n_w}$ may differ from each other.

On the other hand, we find that $\frac{k_{I,b}}{n_{I,b}}$ and $\frac{k_{NN,b}}{n_{NN,b}}$ approximately equal to $\frac{k_b}{n_b}$, $\frac{k_{I,w}}{n_{I,w}}$ and $\frac{k_{NN,w}}{n_{NN,w}}$ approximately equal to $\frac{k_w}{n_w}$ experimentally. This suggests that the irregular black pixels and white pixels in the set A are uniformly distributed

among A_I and $A_{N,N}$, respectively. Experimentally, we have verified for most of the test scanned images that it is valid with a difference of no more than ± 0.01 .

$\xi_I - \xi_{N,N}$ for an original scanned image Inspired by the above experimental results, we can rewrite $\frac{k_I}{n_I}$ as:

$$\begin{aligned}
 \frac{k_I}{n_I} &= \frac{k_{I,b} + k_{I,w}}{n_{I,b} + n_{I,w}} \\
 &= \varphi_I \frac{k_{I,b}}{n_{I,b}} + (1 - \varphi_I) \frac{k_{I,w}}{n_{I,w}} \\
 &= \varphi_I \frac{k_b}{n_b} + (1 - \varphi_I) \frac{k_w}{n_w} + \varphi_I \left(\frac{k_{I,b}}{n_{I,b}} - \frac{k_b}{n_b} \right) \\
 &\quad + (1 - \varphi_I) \left(\frac{k_{I,w}}{n_{I,w}} - \frac{k_w}{n_w} \right)
 \end{aligned} \tag{3.22}$$

where $\varphi_I = n_{I,b}/n_I$.

Similarly,

$$\begin{aligned}
 \frac{k_{NN}}{n_{NN}} &= \frac{k_{NN,b} + k_{NN,w}}{n_{NN,b} + n_{NN,w}} \\
 &= \varphi_{NN} \frac{k_{NN,b}}{n_{NN,b}} + (1 - \varphi_{NN}) \frac{k_{NN,w}}{n_{NN,w}} \\
 &= \varphi_{NN} \frac{k_b}{n_b} + (1 - \varphi_{NN}) \frac{k_w}{n_w} + \varphi_{NN} \left(\frac{k_{NN,b}}{n_{NN,b}} - \frac{k_b}{n_b} \right) \\
 &\quad + (1 - \varphi_{NN}) \left(\frac{k_{NN,w}}{n_{NN,w}} - \frac{k_w}{n_w} \right)
 \end{aligned} \tag{3.23}$$

where $\varphi_{NN} = n_{NN,b}/n_{NN}$.

Subtracting (3.23) from (3.22) yields:

$$\begin{aligned} \frac{k_I}{n_I} - \frac{k_{NN}}{n_{NN}} &= (\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right) + \varphi_I \left(\frac{k_{I,b}}{n_{I,b}} - \frac{k_b}{n_b} \right) \\ &\quad + (1 - \varphi_I) \left(\frac{k_{I,w}}{n_{I,w}} - \frac{k_w}{n_w} \right) + \varphi_{NN} \left(\frac{k_b}{n_b} - \frac{k_{NN,b}}{n_{NN,b}} \right) \\ &\quad + (1 - \varphi_{NN}) \left(\frac{k_w}{n_w} - \frac{k_{NN,w}}{n_{NN,w}} \right) \end{aligned} \quad (3.24)$$

Suppose we assume $\left| \frac{k_{I,b}}{n_{I,b}} - \frac{k_b}{n_b} \right| \leq C$, $\left| \frac{k_{NN,b}}{n_{NN,b}} - \frac{k_b}{n_b} \right| \leq C$, $\left| \frac{k_{I,w}}{n_{I,w}} - \frac{k_w}{n_w} \right| \leq C$, and $\left| \frac{k_{NN,w}}{n_{NN,w}} - \frac{k_w}{n_w} \right| \leq C$, (3.24) can be reduced to

$$\begin{aligned} &\left| \left(\frac{k_I}{n_I} - \frac{k_{NN}}{n_{NN}} \right) - (\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right) \right| \\ &\leq \varphi_I \left| \frac{k_{I,b}}{n_{I,b}} - \frac{k_b}{n_b} \right| + (1 - \varphi_I) \left| \frac{k_{I,w}}{n_{I,w}} - \frac{k_w}{n_w} \right| \\ &\quad + \varphi_{NN} \left| \frac{k_{NN,b}}{n_{NN,b}} - \frac{k_b}{n_b} \right| + (1 - \varphi_{NN}) \left| \frac{k_{NN,w}}{n_{NN,w}} - \frac{k_w}{n_w} \right| \\ &\leq 2C \end{aligned} \quad (3.25)$$

Image dependent threshold In (3.25), we can use $(\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right)$ as an estimation of the initial value. As information is hidden in the image, the value $(\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right)$ changes also. We denote the value in the stego as $(\varphi_I^* - \varphi_{NN}^*) \left(\frac{k_b^*}{n_b^*} - \frac{k_w^*}{n_w^*} \right)$. The embedding increases both $\frac{k_b}{n_b}$ and $\frac{k_w}{n_w}$ to $\frac{k_b^*}{n_b^*}$ and $\frac{k_w^*}{n_w^*}$, respectively. The increases in the two items tend to cancel out with each other and make $\left(\frac{k_b^*}{n_b^*} - \frac{k_w^*}{n_w^*} \right)$ close to the original value $(\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right)$, experimentally. We select a threshold $\vartheta = (\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right) + \vartheta'$, where ϑ' is an empirically selected value. As $(\varphi_I - \varphi_{NN}) \left(\frac{k_b}{n_b} - \frac{k_w}{n_w} \right)$ varies from image to image, the threshold ϑ is image dependent.

Figure 3.12: The ROC curve.

3.3.5 Experimental results

To evaluate the proposed steganalysis technique, we have conducted experiments on 150 machine print database (MPDB) images with commonly used font types and sizes and 120 other scanned images. The 120 scanned images were obtained using a HP scan jet 5590 with different resolution setting at 150, 200 and 300 dpi. Each image is A4 size (International Standards Organization (ISO) standard-size typing paper, 210 mm wide and 297 mm tall). We employ four different data hiding schemes in [83, 88, 112, 117] to hide random messages into the original image at different embedding rates (0.1, 0.2, ..., 1.0) to create a set of stego images. In particular, for Wu's scheme and Yang & Kot's scheme, two different block sizes are used. We then compute the value $(\xi_I - \xi_{N,N})$ for all original images and stego images and compare it with the fixed or image dependent threshold. If $(\xi_I - \xi_{N,N}) > \vartheta$, the image is classified as a stego image; otherwise, it is unlikely that there is hidden data in the test image. In order to see the performance of the fixed and image dependent thresholds, we plot

(a) Yang & Kot's scheme(5×5)

(b) Yang & Kot's scheme(3×3)

(c) Wu's scheme(10×10)

(d) Wu's scheme(16×16)

(e) Pan's scheme

(f) Mei's scheme

Figure 3.13: The detection rates at different embedding rates by a fixed and image dependent threshold.

the Receiver Operating Characteristic (ROC) curves for both the fixed and the image dependent thresholds in Figure 3.12. The detection rates are calculated based on all stego images generated by all embedding schemes at all different embedding rates. The ROC curves shows the trade off between true positive (probability of detection rate) and false positive rate (probability of false alarm) by choosing different thresholds. From the ROC curve, we can see that there is still some room for improvement. For example, the proposed method provide a true postive rate of about 84% at a false positive rate of 20%. The acceptable true positive and false positive rates would be application dependent. In particular, Figure 3.13 shows both the detection rates on different embedding schemes using the image dependent threshold and an empirically selected fixed threshold when false alarm rate is 0.05. From the result we can see that the image dependent threshold provides higher detection rate compared with the fixed threshold no matter which embedding algorithm is used to generate the stego images, thus it is more robust. We can also see that the detection rate drops suddently as the embedding decreased to certain rate, e.g., 0.4 for Yang & Kot's scheme(5×5) in Figure 3.13(a) and 0.5 for Wu's scheme(10×10) in Figure 3.13(c). As expected, the performance for different embedding schemes is different because different embedding schemes have different maximum capacities. For other schemes such as the work in [111], our detection scheme still works as long as the COL pixels are used as flippable pixels with high priority. The performance is expected to be related with the capacity. The more pixels being flipped, the more likely we can detect the hidden message.

3.4 Summary

In this Chapter, we first propose a method to detect the hidden message in a clean text-font image. The proposed method is based on the fact that the marks origin from the same source should be identical in the image while the data hiding is likely to change it. It actually implies that computer generated text image may not be good for steganographic application. Quantitative analysis of the message length is also discussed and two methods are proposed to estimate the message length. Accuracy of the two estimation methods has been investigated and compared. We find that the second message length estimation method provides better results than the first one, especially for embedding schemes which randomly select flippable pixels for flipping. The proposed method can detect the existence of the hidden message without the knowledge of the details of the embedding algorithms for most of the boundary pixel-flipping based techniques. The limitation of the method is that it only works for clean text-font image free from scanning errors. It cannot detect the existence of hidden message for very low capacity embedding. For example, we hide several bits in one ‘s’ and make all other ‘s’ identical with the ‘s’ where we hide the information. However, the information can be hidden would be quite lower compared with the method in [83, 112, 117].

We also propose a method to detect the existence of hidden messages embedded by the algorithms that flip COL pixels in a noisy scanned text image. No prior knowledge of other details of the embedding algorithm is needed. The feature being used in the detection is based on the similarity among the similar marks in the text image. By subtracting the factor of the original noise level, we extract the feature as $(\xi_I - \xi_{N,N})$ instead of ξ_0 . A theoretical analysis of the

increase of $(\xi_I - \xi_{N,N})$ has been provided. Two thresholds including fixed and image dependent thresholds are used to classify stego images from original images. The image dependent threshold provides higher detection rate compared with a fixed threshold at the same false alarm rate. Experimental results show that the proposed method work well for both MPDB images and other scanned images when the number of COL pixels being flipped is large enough. The proposed method can detect hidden information by the schemes that mainly use COL pixels without detailed knowledge of the embedding algorithm, but not the message length. Our test results show that the similar marks from scanned image have strong correlations though they are no not identical. Data hiding may affect the correlations. Thus, the next generation embedding scheme should consider the correlation among symbols in selecting flippable pixels.

Chapter 4

Binary Graphic Images

4.1 Introduction

In this Chapter, we aim at studying steganalysis of data hiding in binary graphic images. Pixel flipping techniques [21, 74, 83, 88, 101, 112, 113, 117] can be applied not only on a binary text image, but also on a binary graphic image such as logo, cartoon, line art, and etc., as we have mentioned in Section 2.1.2. A binary text image usually contains similar or same marks which can be used in the steganalysis. However, a graphic image is different from a text image and does not contain high similarity marks in the image. Thus, the detection schemes based on mark similarity proposed in Chapter 3 do not apply to binary graphic images. The scheme [121] mentioned in Section 2.1.2 is tested with cartoon images, however, the embedding approaches used to hide the information are relatively earlier schemes which have not taken smoothness and connectivity [112] into consideration. It does not apply for the recent data hiding schemes [21, 74, 83, 88, 101, 112, 113, 117].

As we know, data hiding usually changes the cover image and thus introduces distortion. When evaluating the image distortions using distortion measures, the scores from a stego image should be different from that of those obtained from its original image. Thus, distortion measures can be useful in steganalysis [6, 87]. The distortion score is usually computed by comparing the distorted image with the original one. However, the original image is assumed to be not available in the steganalysis. Instead, we perform de-noising process to obtain a de-noised image that is close to the original one. The rationale of the technique is that an embedding process creates distortion on the contours of the binary graphic images and often makes the contour lines less smooth after information hiding. Thus, we can de-noise the stego image by predicting and smoothing the edge contours. It is expected that the distortion in a de-noised image from its original image is different from the distortion in a de-noised image from its stego image. Based on this observation, we design a classifier to separate the stego image from original image. A good distortion measure that can reflect the distortions introduced by the data hiding is important in the detection. In this Chapter, we first propose a new objective distortion measure based on edge line segments to evaluate the distortion introduced by flipping individually selected edge pixels for binary text and graphic images in Section 4.2. In Section 4.3, we apply the proposed distortion measure together with other existing distortion measures in steganalysis. The summary is given in the last Section.

4.2 Edge line segment similarity measure for binary images

4.2.1 Background

The distortion measure topic is an important one in image processing including data hiding. Besides their application in steganalysis, distortion measures are also important in evaluating the performance of the data hiding algorithms as well as providing insights into data hiding. Distortion measures can be categorized into subjective measurement and objective measurement [96] categories. As discussed in [61], a subjective distortion measure quantifies the dissatisfaction of the viewer in observing the distorted image instead of the original. A common way to evaluate the dissatisfaction is through subjective testing. In these tests, observer views a series of distorted images and rate them based on the visibility of the artifact. The results of subjective testing depend on various factors such as the observer's background. Subjective measurement is important for image quality evaluation since the images are ultimately viewed by human beings. However, subjective testing is inconvenient, time consuming and expensive. The objective distortion measure gives the distortion between the original and the distorted image mathematically such as mean-square error (MSE), peak-signal to noise ratio (PSNR) and signal to noise ratio (SNR). However, it may not reflect the observer's visual perception of distortion. An objective distortion measure that can accurately reflect the subjective ratings would be quite useful in designing data hiding algorithms. Several authors have discussed the gaps between subjective measurement and objective measurement and they have proposed solutions for multi-level images [15, 61, 68, 86, 98, 105].

Little work has been done on objective measurement for binary text images. As we have mentioned in Chapter 2, data hiding in binary images have received much attention and a series of data hiding schemes for binary images by flipping individually selected edge pixels have been proposed [21, 83, 101, 112, 113, 117] in recent years. These schemes are called pixel flipping techniques. One of the important issues of the pixel flipping technique is how to select the flippable pixels to minimize distortion. Also, the comparison of the distortion introduced by different schemes is important. However, performance evaluation is difficult due to the lack of objective distortion measure that reflects the subjective test results. There is an urgent need to develop a good objective distortion measure.

Distance reciprocal distortion measure (DRDM) proposed in [75, 76] uses a weight matrix to calculate the distortion caused by the flipping of pixels. It shows better correlation with human perception than PSNR [75, 76] and this measure works well if salt and pepper noise is involved. For example, flipping a non-edge pixel usually causes a larger visual distortion than flipping an edge pixel. However, most good data hiding techniques for binary images involve flipping edge pixels only. If connectivity is also taken into consideration, the measure of distortion would be more accurate. Connectivity is important in high quality images such as clean text images. For example, flipping the center pixels of the two patterns in Figure 4.1 have the same distortion score according to DRDM, but the visual distortions by flipping them are quite different. The change in smoothness and connectivity measure (CSCM) by using a modified weight matrix in [22] gives reasonable distortion scores. However, it has a limitation for patterns such as sharp corners and pixels along a straight line. In this section, we introduce a new objective distortion measure for clean text images and high quality binary graphic images which is based on edge line

Figure 4.1: Two 3×3 patterns with same distortion according to DRDM.

Figure 4.2: Illustration of 'edges'.

segment similarity.

4.2.2 Edge line segment

We define an edge line as the common shared 'line' between two immediate neighbor pixels where the pixel values for the two pixels are different. The edge pixel refers to pixels beside the edge line. For example, in the enlarged 'a' in Figure 4.2(a), the edge lines are those black lines shown in Figure 4.2(b). Since every pixel is a square; the edge line can only change its direction by $\pm 90^\circ$. When the edge line changes its direction, it forms a sharp corner. For simplicity, we call it a corner. There are eight different types of corners that make up all possible corners in a 2×2 block as indicated in Figure 4.3.

Figure 4.3: Eight types of corners

Figure 4.4: Two types of crosses.

Figure 4.5: Edge line segments associated with flipped pixels. (a) Enlarged “s”. (b) Edge line segments associated with pixel A. (c) Edge line segments associated with pixel C. (d) edge line segments associated with pixel B.

For a 2×2 block, if any two pixels in the same row or column have different colors as shown in Figure 4.4, the four pixels form a cross at the center point of the block. Corners and crosses divide the edge line into edge line segments. Each edge line segment starts from one corner or cross and ends at another corner or cross without any corner or cross in between. These corners or crosses define the two ends of the edge line segment. An edge line segment is associated with a pixel if flipping the pixel changes the edge line segment.

4.2.3 Human visual perception

From our observation, the edge line segments associated with an individually flipped pixel play an important role in determining the visual distortion caused

Table 4.1: Numbers of edge line segments associated with pixels A, B and C and subjective distortion.

Pixel	Before flipping	After flipping	Changes	Subjective rating
A	One(1)	Five(1',2',3',4',5')	Four	high
B	Four(5,6,7,8)	Two(9',10')	Two	medium
C	Three(2,3,4)	Three(6',7',8')	Zero	low

by the flipped pixel. The first factor is the amount of the related edge line segments. For example in Figure 4.5, flipping pixel A causes larger distortion than flipping pixel B and flipping pixel B causes larger distortion than flipping pixel C. We show in Table 4.1 the numbers of the associated edge line segments before and after flipping these pixels and the corresponding subjective distortion. We can see that the rank of a subjective test is closely associated with the amount of change in the related edge line segments.

The length of each edge line segment also plays an important role in a distortion measure. In Figure 4.6, flipping D, E or F do not change the number of edge line segments. From human perception, flipping D seems to be more distorted than flipping E or F. In this case, the number of edge line segments remains the same but the lengths of some of the segments are changed.

4.2.4 Edge line segment similarity

We use edge line segment similarity to calculate the distortion between two edge line segment before and after flipping. For two edge line segments whose lengths are l_1 and l_2 , we define their edge line segment similarity $ELSS$ as the ratio of

Figure 4.6: Enlarged “t”.

the minimum vs. maximum of their lengths l_1 and l_2

$$ELSS = \frac{\min(l_1, l_2)}{\max(l_1, l_2)} \quad (4.1)$$

For $l_1 = l_2$, $ELSS = 1$, otherwise $ELSS < 1$. Obviously, larger $ELSS$ corresponds to less distortion in edge line segment. We can easily define the distortion d for a particular edge line segment as follows:

$$d = 1 - ELSS \quad (4.2)$$

4.2.5 Identifying and mapping of edge line segments

Flipping an edge pixel may change the lengths of existing edge line segments, create new edge line segments, remove existing edge line segments, separate the existing line segment, and merge two edge line segments to one edge line segment. Identifying and mapping of the associated edge line segments are needed in order to measure the distortion properly.

Figure 4.7: Pixels and borders in a 3×3 pattern.

To identify and map the related edge line segments before and after flipping a pixel, we consider the 3×3 block centered at this pixel. Denote the center pixel of the block as p_c with its pixel value $\bar{p}_c$. Denote the center pixel of the block as p_i , for $i = 0, 1, 2, \dots, 7$, with corresponding pixel value $\bar{p}_i$. Label the one-pixel-long common sharing line segment between p_i and p_{i+1} as b_i , for $i = 0, 1, 2, \dots, 7$ (p_8 refers to p_0), and the one-pixel-long common sharing line segment between p_c and p_i as c_i , for $i = 0, 2, 4, 6$. Figure 4.7 shows the pixels and the labels. A common sharing line is a part of an edge line segment if the values of the two pixels associated with it are different. For later use, we define $p_k = p_{k-8}$, $b_k = b_{k-8}$ and $c_k = c_{k-8}$, for any integer $k \geq 8$.

It can be seen that an edge line segment that is associated with the pixel p_c contains at least one of the twelve common sharing lines. The four line segments c_0 , c_2 , c_4 and c_6 change status from edge line to non-edge line or vice versa when p_c is flipped. Subsequently, the corresponding edge line segment changes. In what follows, we show the conditions associated with the changes in the edge line segments.

For $i = 0, 2, 4, 6$, the line b_i is an part of an edge line segment associated with

p_c if and only if $p_i \neq p_{i+1}$ and $p_{i+1} = p_{i+2}$ (e.g., Figure 4.5 (c) when flipping pixel C). For $i = 1, 3, 5, 7$, the line b_i is an part of an edge line segment associated with p_c if and only if $p_i \neq p_{i+1}$ and $p_{i-1} = p_i$ (e.g., Figure 4.5 (b) when flipping pixel A). We define $\bar{b}_i$ as the value of b_i . $\bar{b}_i = 1$ if b_i is a part of an edge line segment associated with p_c , otherwise $\bar{b}_i = 0$.

These properties can be further demonstrated using Figure 4.8. If $\bar{p}_i \neq \bar{p}_{i+1}$ and $\bar{p}_{i+1} = \bar{p}_{i+2}$, it can be seen that the common sharing line segment c_{i+2} between p_{i+2} and p_c is added to or reduced from the edge line segment which contains b_i when flipping p_c . For example, in Figure 4.8(a), b_i is a part of an edge line segment. After flipping p_c , a new pattern shown in Figure 4.8(b) is created and c_{i+2} becomes a part of the edge line segment containing b_i , thus the edge line segment containing b_i is changed, i.e., b_i is a part of an edge line segment associated with p_c .

If the line segment b_i is a part of an edge line segment associated with p_c , the conditions $p_i \neq p_{i+1}$ and $p_{i+1} = p_{i+2}$ are satisfied. Otherwise, if $p_i = p_{i+1}$, then the line segment b_i is not any part of an edge line segment, which conflicts with the assumption that b_i is a part of an edge line segment. If $p_{i+1} \neq p_{i+2}$ ($p_i \neq p_{i+1}$), the four pixels p_i, p_{i+1}, p_{i+2} and p_c form either a corner or a cross, which is always the end of the edge line segment containing b_i . It means flipping does not change the edge line segment which contains b_i , which implies the segment is not associated with b_i . Thus, we have $p_i \neq p_{i+1}$ and $p_{i+1} = p_{i+2}$. The similar argument goes to the cases for the odd line $b_i, i = 1, 3, 5, 7$.

In the edge line segment mapping process, we map the edge line segment containing b_i in the original image with the edge line segment containing b_i in the distorted image, for any line segment b_i with $\bar{b}_i = 1, i = 0, 1, 2, \dots, 7$. The

Figure 4.8: Relationship between b_i and the associated edge line segment. The pixels in “grid” are don’t care pixels.

length of the edge line segment containing b_i can be computed by locating the two ends of the edge line segment.

In Figure 4.9(a), c_i is a one-pixel-long edge line segment. After flipping p_c , c_i is no longer an edge line segment as shown in Figure 4.9(b). In order to compute the edge line segment similarity based on its length, we map the one-pixel-long edge line segment with the zero-length-long edge line segment after flipping p_c . Similarly, by flipping p_c in Figure 4.9(b), we map the zero-length edge line segment before flipping p_c with the one-pixel-long edge line segment after flipping p_c . Same rules can be applied to all similar patterns with one-pixel-long edge line segment removed or created.

Consider a special case in Figure 4.10 that flipping p_c creates a new one-pixel-long edge line segment at c_{i+4} by shifting the one-pixel-long edge line segment at c_i by one pixel length. Shifting a one-pixel-long edge line segment at c_i by one pixel length to c_{i+4} causes very minimum distortion that can be ignored under human visual perception (e.g., Figure 4.5(c), when flipping pixel C, and Figure 4.6, when flipping pixel D, E, or F), although the change actually involves

Figure 4.9: Examples of removing or creating a one-pixel-long edge line segment. Flipping p_c in (a) removes the segment. Flipping p_c in (b) creates a segment.

creating one and removing one edge line segment. Such a scenario occurs if and only if $\bar{p}_{i+1} = \bar{p}_{i+2} = \bar{p}_{i+3}$, $\bar{p}_{i+5} = \bar{p}_{i+6} = \bar{p}_{i+7}$, $\bar{p}_i \neq \bar{p}_c$ and $\bar{p}_i \neq \bar{p}_{i+4}$. This implies that the two ends of the edge line segment at c_i before flipping have the same corner as the two ends of the edge line segment at c_{i+4} after flipping correspondingly.

We summarize our procedures below to identify and map the edge line segments before and after flipping:

1. Set $\bar{b}_i = 1$, for $i = 0, 1, 2, \dots, 7$, if b_i is a part of an edge line segment associated with p_c , otherwise set it to be 0.
2. Locate the two ends of the edge line segment which contains b_i with $\bar{b}_i = 1$, map and compute l_1 and l_2 of the edge line segment length in the original and distorted image (e.g., edge line segments 2, 4, 5, 8 are mapped to 6', 8', 9', 10' in Figure 4.5(c)(d) respectively. Edge line segment 1' and 5' are all mapped to 1 in Figure 4.5(b)).
3. Map all the one-pixel-long edge line segments removed (or created) by

Figure 4.10: Shifted edge line segments.

flipping p_c to zero-length edge line segments to get $l_1 = 1$ and $l_2 = 0$ (or $l_1 = 0$ and $l_2 = 1$) [e.g., the created edge line segments 2',3',4',7' in Figure 4.5(b)(c) and the removed edge line segments 3, 6,7 in Figure 4.5(c)(d) are mapped to zero-length segments].

4. Discard all the mappings for the one-pixel-long edge line segments created or removed by shifting a one-pixel-long edge line segment by one pixel. The two ends of the one-pixel-long edge line segment c_i before flipping should have the same corner type as the two ends of the new edge line segment c_{i+1} [e.g., 3 and 7' in Figure 4.5(c)].

In the algorithm, the identification of the edge line segment is done by locating the two ends of the edge line segment. To locate the two ends of an edge line segment that b_i belongs to, we start from the two ends of the border b_i and trace to the two sides as shown in Figure 4.11. For each side, we stop when a corner or cross is met.

Figure 4.11: Locating the two ends of an edge line segment.

4.2.6 Total distortion measure

When mapping of the edge line segments before and after flipping one pixel are done properly, distortion for each pair of edge line segments can be computed. The edge line distortion score ELD_k calculated based on the edge line segments for the k^{th} flipped pixel is given by:

$$ELD_k = \sum_i \left(1 - \frac{\min(l_{1i}^k, l_{2i}^k)}{\max(l_{1i}^k, l_{2i}^k)} \right) \quad (4.3)$$

where, l_{1i}^k and l_{2i}^k are the lengths of the i^{th} pair of mapped edge line segments that are associated with the k^{th} flipped pixel with l_{1i}^k for the segment in the original image and l_{2i}^k from the distorted image.

When a total of N pixels are flipped in a given image, the overall distortion ELD_{total} can be obtained by:

$$ELD_{total} = \sum_{k=1}^N \sum_i \left(1 - \frac{\min(l_{1i}^k, l_{2i}^k)}{\max(l_{1i}^k, l_{2i}^k)} \right) \quad (4.4)$$

4.2.7 Subjective test

Experiments similar to the setup in [75] were carried out to test the performance of the proposed distortion measure using the two binary text images and two

These *are* testing. 请根据所观察到的失真
 Rank: 1 2 3 4? 排列出图像的失真大小
Thank You! 非常感谢!

(a) English text (207×94 pixels)(b) Chinese text (243×87 pixels) (c) Cartoon 1 (200×140 pixels) (d) Cartoon 2 (100×100 pixels)

Figure 4.12: Original images.

binary cartoon images shown in Figure 4.12.

The subjective tests were carried out using these two original clean text images and two original binary cartoon images. The two clean text images consists of characters in different languages with different fonts. The characters in different languages are all based on curve, which appears in other languages as well. Thus, these images are good representative. The two cartoon images are high quality images with smooth contours. These images can represent high quality binary graphic images where smooth contours appear often. For each original image, we generated a number of random test images with different visual distortion by adding noise to the original image. As we were evaluating the distortion by flipping edge pixels, we only added noise along the edges. A total of 200, 400, 500 and 150 edge pixels were flipped from the original English text image, original Chinese text image, original Cartoon 1 image, and original Cartoon 2 image, respectively. A series of distorted images were generated by

flipping different edge pixels. All the distorted images generated from the same original image were divided into four groups based on ELD_{total} value; with group 1 having lowest ELD_{total} and group 4 having highest ELD_{total} . Four images were chosen randomly with each image from each of the four groups to form one set of English text images, Chinese text images or cartoon images as shown in Figure 4.12, Figure 4.13, Figure 4.15 and Figure 4.16.

In all the cases, the subjective assessments were done by 60 subjects. Each subject was given the original image and four sets of test images, which were printed on an 80 GSM quality paper using HP LaserJet 5000 printer. The subject was asked to rank the quality of the four images in each set according to the distortions that he or she perceived in normal viewing condition.

There are four rankings (1, 2, 3 and 4) with score 1 for the least distortion and 4 for the most distortion perceived. The ranking scores are analyzed and compared with the ranking according to the distortion score computed by the new proposed measure. The results are shown in Table 4.2. PSNR for every image is of the same value as the same number of pixels is being flipped in each image. All the scores by distance reciprocal distortion measure (DRDM) [75,76], change of smoothness and connectivity measure (CSCM) [22] that we have reviewed in Section 4.2.1 and the proposed method are shown in Table 4.2, where the smaller score indicates less distortion. It can be seen that the proposed method correlates well with the subjective ranking by visual perception while the PSNR and DRDM do not. CSCM gives the same ranking as the proposed one but it has some limitation for patterns [75]. As we know, flipping a pixel along a straight line is normally considered as a large distortion; however, CSCM assigns a low distortion score instead. Another limitation of CSCM as well as

Table 4.2: Experimental results.

	Group	PSNR	DRDM	CSCM	ELD_{total}	Subjetive ranking
English Text	1	19.9	104	42	150	1.15
	2	19.9	108	66	326	2.03
	3	19.9	106	76	504	3.05
	4	19.9	104	94	700	3.77
Chinese Text	1	17.2	244	144	280	1.08
	2	17.2	236	172	616	2.06
	3	17.2	240	188	1100	3.03
	4	17.2	228	204	1420	3.83
Cartoon 1	1	17.5	255	41	382	1.03
	2	17.5	252	108	875	2.10
	3	17.5	256	183	1347	3.06
	4	17.5	256	236	1742	3.80
Cartoon 2	1	18.2	79	11	125	1.02
	2	18.2	79	42	308	2.12
	3	18.2	78	60	467	3.04
	4	18.2	80	85	604	3.82

DRDM is that it cannot tell the difference between the pixels shown in Figure 4.6 while the proposed method can differentiate the two. The proposed method uses the 3×3 block centered at each flipped pixel to identify the edge line segments associated with the flipped pixel. It may need to extend to area outside the 3×3 block to calculate the length of these associated edge line segments. The size of the area to be extended into is determined by these edge line segments automatically. Thus, the size of area that determines the distortion score is more flexible.

The distributions of the subjective ranking scores for the images are shown in Figure 4.17, Figure 4.18, Figure 4.19, and Figure 4.20. In these figures, the abscissa represents four ranking scores (1, 2, 3 and 4), and the coordinate shows the occurrence of the corresponding ranking scores given by four groups of 60 human subjects. Since each of the 60 subjects is given four sets of test images,

Figure 4.13: One set of distorted images based on the English text image.

Figure 4.14: One set of distorted images based on the Chinese text image.

Figure 4.15: One set of distorted images based on Cartoon 1 image.

Figure 4.16: One set of distorted images based on Cartoon 2 image.

Figure 4.17: Distribution of ranking scores for English text images.

Figure 4.18: Distribution of ranking scores for Chinese text images.

Figure 4.19: Distribution of ranking scores for Cartoon 1 images.

Figure 4.20: Distribution of ranking scores for Cartoon 2 images.

there are 240 scores in total for each group. From the figures, we can see that the quality of each image is distinctively different.

It should be noticed that our test is based on high quality images such as clean text images or high quality graphic images. For other type of images such as scanned images, it is hard to perform the subjective test as the “original” image is already distorted with scan noise.

There exists some relationship between the contour based view measure and the pixel level view measure. We take the ‘L’ shape pattern shown in Figure 4.10 as an example. From contour based view point, flipping the center pixel shifts a one-pixel-long edge line segment by one pixel while keeping the total number of edge line segments unchanged. From pixel level view point, the transition from black to white or white to black is changed while the total number of transition remains the same. However, these are not equivalent. In the examples shown in Figure 4.6, the shape of the letter “t” remains the same after flipping the pixels ‘D’ and ‘E’, thus from contour based view point, the distortion is small. However, from pixel level view point, it is a relatively large distortion as the number of transitions reduced. Another difference between the contour based view point and pixel level view point is that the contour based measure uses the 3×3 block centered at each flipped pixel to identify the edge line segments associated with the flipped pixel. It may need to extend to area outside the 3×3 block to calculate the length of these associated edge line segments. The size of the area to be extended into is determined by these edge line segments automatically. On the other hand, the pixel level considers a small region e.g., a 3×3 square as in Wu’s scheme or Yang’s scheme. The extension to a larger region (say 5×5) is difficult. The contour based measure can tell the difference

Figure 4.21: The diagram of the detector using distortion measures.

between the pixels ‘D’ and ‘E’ shown in Figure 4.6 while the pixel level pattern based measure cannot.

4.3 Steganalysis using distortion measures

As we have mentioned in Section 4.1 that distortion measures are useful in steganalysis [6,87]. The rationale of the technique is that an embedding process creates distortion on the contours of the binary graphic images and often makes the contour lines less smooth after information hiding. The distortion score is usually computed by comparing the distorted image with the original one. However, the original image is assumed to be not available in the steganalysis. We need to generate a reference image to calculate the distortion scores. We perform de-noising process to obtain a de-noised image that is close to the original one. The main idea of the detection using distortion measure can be summarized into the diagram in Figure 4.21.

4.3.1 Distortion control in embedding

To start with, we give a brief review of the embedding schemes by studying how the visual distortion is minimized. Arbitrarily flipping of pixels in a binary image may cause visible distortion, especially when flipping a pixel from pure white or black region. The scheme in [21] uses a condition that only pixels along edges

can be flipped. However, this may not be adequate to preserve the quality of the image. Flipping pixels along smooth contours such as straight lines may change the smoothness of the edge contour and cause visible distortion in an image. Recall that in [67, 83, 88, 112, 117], the authors have studied the flippability of each pixel by comparing with its neighboring pixels and more strict criteria are proposed based on smoothness and connectivity. As we mentioned before COL pixels as shown in Figure 3.4 are chosen in most of these schemes as flippable pixel with high priority as it does not affect the smoothness and connectivity of the contours [22].

4.3.2 De-noising process

In order to compute distortion score for an image, a reference image that is close to the original image is needed. It is done by applying a de-noising algorithm to the image. Since the final purpose is to differentiate stego image with original image, the de-noising process should possibly recover the original pixels which are flipped in the embedding. Also, it should lower the chance to change original pixels without hidden information. After studying the existing pixel flipping schemes, the following two de-noising processes are applied to remove noise in different patterns due to the embedding process such that a de-noised image close to the original can be obtained.

4.3.2.1 Morphological smoothing

The flipping of some pixels may change the smoothness of the image. It creates holes or protrusions as shown in Figure 4.22. We propose a morphological smoothing process to fill holes and remove protrusions based on the two ba-

Figure 4.22: Patterns with protrusion and hole.

basic morphological operations: erosion and dilation [45, 52]. The mathematical definitions of the morphological erosion and dilation are reproduced as follows: Suppose X is the set of foreground pixels defined in $\mathbb{Z}^2$. A Structure Element (SE) is a set B , also defined in $\mathbb{Z}^2$. Let B_{xy} denote the translation of B so that its origin is located at (x, y) . Then the erosion of X by B is defined as the set of all points (x, y) such that B_{xy} is included in X . That is,

$$X \ominus B \triangleq \{x : B_{xy} \subset X\} \quad (4.5)$$

Similarly, the dilation of X by B is defined as the set of all point such that B_{xy} hits X , that is, they have a nonempty intersection:

$$X \oplus B \triangleq \{(x, y) : B_{xy} \cap X \neq \Phi\} \quad (4.6)$$

Based on the two basic operations, the opening process and the closing process [45, 52] are defined as erosion followed by dilation and dilation followed by erosion respectively, with the same SE used for both erosion and dilation. That is,

$$X \circ B \triangleq X \ominus B \oplus B \quad (4.7)$$

$$X \bullet B \triangleq X \oplus B \ominus B \quad (4.8)$$

Opening process filters out the details of the image and removes the noise smaller than SE while the closing process fills gaps and holes smaller than SE. As a result, we apply a smoothing process as an opening process followed by a closing process on the image to remove protrusions as well as to fill holes:

$$X \otimes B \triangleq X \circ B \bullet B = X \ominus B \oplus B \oplus B \ominus B \quad (4.9)$$

Since the holes or protrusions created by the pixel flipping techniques are usually no more than one pixel width, the following SE is used.

$$\text{SE} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \quad (4.10)$$

It can be seen that the opening and closing process with the above SE can fill holes and remove protrusions due to the embedding; however, it may also remove the original one-pixel-width strokes in the original image. To preserve the one-pixel-width strokes, any two 8-connected neighboring pixels that are flipped simultaneously by the above smoothing process are flipped back. Denote the set of pixels flipped by the flipping-back process as F .

$$F \triangleq \{(x, y) : (x, y) \in D \text{ and } N_{xy} \cap D \neq \emptyset\} \quad (4.11)$$

where $D = \overline{X \otimes B} \cap X + X \otimes B \cap \overline{X}$ and N_{xy} is the set of 8-connected neighboring points of (x, y) . Finally, the proposed morphological smoothing (MS) technique is defined as an opening process followed by a closing process and then the above

flipping-back process:

$$MS(X, B) \triangleq (X \otimes B \cap \overline{F}) \cup (\overline{X \otimes B} \cap F) \quad (4.12)$$

We do the flipping-back process as above because the chance to flip two 8-connected pixels simultaneously by the embedding algorithm is very small [101, 112]. By doing so, we can preserve most of the one-pixel-width strokes in the original image. The trade off is that the embedding noise along the one-pixel-width strokes cannot be removed by the morphological smoothing process. The limitation of the morphological smoothing is that it cannot recover some flipped pixels such as COL pixels. For an embedding scheme which makes use of COL pixels, morphological smoothing does not work well.

4.3.2.2 De-noising the stair case pattern

Removing noisy pixels by not changing the smoothness and connectivity is difficult. In order to find a way to possibly recover the flipped pixels among these pixels such as COL pixels, we look into how the flipping of these pixels affects the edge contours. Very often, we find stair case patterns in cartoon images. The digital contour can be considered as a line digitized from an analog curve. As the digitization of a diagonal or curved line, the stair case patterns appear. For example, the pattern shown in Figure 4.23 is a possible digital representation of a straight slope. Very often, COL pixels appear in these patterns.

The locations of the corners, i.e., the locations where the edge lines change their directions, are important in the stair case patterns. As we have discussed in Section 4.2.2, there are eight different types of corners including four black and four white corners to make up all possible corners in a pixel block as indicated

Figure 4.23: Digital representation of analog lines.

Figure 4.24: Consecutive same type corners.

in Figure 4.4.

As the consecutive corners of the same type along one contour are correlated very often when they are close, we can predict the location of a corner based on its nearby preceding and subsequent corners. For example, the type IV corners in Figure 4.23 can be predicted well by a linear equation. On the other hand, the corners are shifted due to the flipping in the embedding. Suppose the prediction error of the corner location before flipping is very small, then the resultant corner after flipping tends to stay further to the predicted location. Based on this observation, we propose a de-noising process by shifting the corner back closer to the predicted location. The shifting is done by flipping a pixel along stair case pattern. Since the locations of black and white corners are dependent with each other, we use black corners only. Without lost of generality, we first study the case that the corners are shifted horizontally.

We first obtain the locations of the corners in the sequence determined by the contour tracing. Next, the corners are divided into groups with each group formed by same type black corners consecutively. Suppose a group with K corners with locations for the k^{th} corner at (x_k, y_k) , $k = 1, 2, \dots, K$, as shown in Figure 4.24. Since the corners are shifted horizontally, we consider y_k as a function of x_k . Assume that the coordinates of the corner at (x_k, y_k) together with its neighboring corners can be approximated by a polynomial function $f(x)$ with highest degree $2M - 1$. The function $f(x)$ can be written as :

$$f(x) = \sum_{i=0}^{2M-1} a_i x^i \quad (4.13)$$

To estimate the column coordinate y_k for the k^{th} corner, we need to calculate the coefficients a_i , for $i = 0, 1, \dots, 2M - 1$, based on $2M$ sampling points from $f(x)$. Obviously, one of the choices is the M preceding and M subsequent corners. Assume that these $2M$ corners are the sampling points from the curve $f(x)$. We have the following $2M$ equations:

$$\left\{ \begin{array}{lcl} y_{k-M} & = & a_0 + a_1 x_{k-M} + \dots + a_{2M-1} x_{k-M}^{2M-1} \\ y_{k-M+1} & = & a_0 + a_1 x_{k-M+1} + \dots + a_{2M-1} x_{k-M+1}^{2M-1} \\ \vdots & & \\ y_{k-1} & = & a_0 + a_1 x_{k-1} + \dots + a_{2M-1} x_{k-1}^{2M-1} \\ y_{k+1} & = & a_0 + a_1 x_{k+1} + \dots + a_{2M-1} x_{k+1}^{2M-1} \\ y_{k+2} & = & a_0 + a_1 x_{k+2} + \dots + a_{2M-1} x_{k+2}^{2M-1} \\ \vdots & & \\ y_{k+M} & = & a_0 + a_1 x_{k+M} + \dots + a_{2M-1} x_{k+M}^{2M-1} \end{array} \right. \quad (4.14)$$

Solve the above $2M$ equations to obtain the $2M$ coefficients a_i , for $i =$

$0, 1, \dots, 2M - 1$ and to estimate the column coordinate y_k for the given row x_k . Denote the estimated y_k for the given x_k based on corners from both sides as y_k^B . The prediction error is given by:

$$e_k^B = |y_k^B - y_k| \quad (4.15)$$

In order to determine M , we first look into how the pixel flipping affects the prediction errors. Let's say the original error $e = |f(x_k) - y_k|$ and the flipping shifts a corner (x_k, y_k) to (x_k, y'_k) . The new error e' would be

$$e' = \begin{cases} e + 1 & \text{if } f(x_k) < y_k < y'_k \text{ or } f(x_k) > y_k > y'_k \\ |e - 1| & \text{otherwise} \end{cases} \quad (4.16)$$

If e is smaller than 0.5, the new error e' is always larger than the original error e ; otherwise, there is no guarantee that the original error is larger. Recall that we de-noise the image by making the prediction error smaller based on the assumption that the prediction error before flipping is small enough. As we mentioned earlier, we should lower the chance to flip the original pixels in the image when we de-noise the image in order not to confuse the classifier. Thus we can see that we expect to find a prediction method that can predict the corner location with error smaller than 0.5 for the original image. In practice, an ideal prediction method that gives error e for original images always smaller than 0.5 hardly exists. Instead, we expect that the prediction gives more error e smaller than 0.5. We study the distributions of the prediction errors from all the original testing images using polynomials with different M . Shown in Figure 4.25 is the distribution for e_k^B in different ranges $[0, 0.5)$, $[0.5, 1)$ and $[1, +\infty)$. From the distribution, we can see that the linear prediction using $M = 1$ is the

Figure 4.25: Probability of occurrence of e_k^B in different ranges $[0, 0.5)$, $[0.5, 1)$ and $[1, +\infty)$.

best choice as it has most errors less than 0.5.

Besides the above estimation, we may also estimate y_k based on $2M$ preceding or $2M$ subsequent corners. These two estimations are important because of the existence of singular points [14]. Around a singular point, the edge contour may be well predicted based on corners from one side but not from both sides. Correspondingly, we can obtain the estimated column coordinate denoted as y_k^L and y_k^R , respectively. Similarly, the distance between the estimated corners and the actual corner is denoted as e_k^L and e_k^R can be calculated by

$$e_k^L = |y_k^L - y_k| \quad (4.17)$$

$$e_k^R = |y_k^R - y_k| \quad (4.18)$$

We also study the distribution of the prediction errors e_k^L and e_k^R . Figure 4.26 shows the distribution of e_k^L . The distribution of y_k^R is similar to the distribution

Figure 4.26: Probability of occurrence of e_k^L in different ranges $[0, 0.5)$, $[0.5, 1)$ and $[1, +\infty)$.

of y_k^L . We observe that $M = 1$ is also the best choice.

Suppose we can flip a pixel to shift the corner at (x_k, y_k) to (x_k, y'_k) without creating a new type of black corner within the 3×3 block centered at the pixel to be flipped. We compute the three new estimation errors $\hat{e}_k^L = |y_k^L - y'_k|$, $\hat{e}_k^B = |y_k^B - y'_k|$ and $\hat{e}_k^R = |y_k^R - y'_k|$ after the shifting. We shift all the black corners along the stair case pattern if the shifting satisfies the following conditions: After shifting, $\hat{e}_k^L < e_k^L$, $\hat{e}_k^B < e_k^B$ and $\hat{e}_k^R < e_k^R$.

It should be noticed that some of the three estimations may not be applicable in some cases and the corresponding errors cannot be computed. In that case, we ignore the inapplicable errors. When only one of the three estimations is applicable, we do not shift the corner unless the estimation errors become 0 after shifting. For example, we only have $\hat{e}_k^L$ and e_k^L for the last corner in each group, we do not shift the corner unless $\hat{e}_k^L = 0$. The additional condition here is to reduce the chance of flipping original pixels in the end points.

When a corner is shifted vertically by flipping a pixel, we consider x_k as a function of y_k instead. We then predict x_k based on the preceding and subsequent corners to obtain the prediction error. Similar rules as above is used in the de-noising process to de-noise the image.

4.3.3 Selection of distortion measure

In order to select distortion measures for feature extraction as well as to see the effectiveness of the de-noising process in recovering the original pixels as we showed in Figure 4.21, we apply Analysis of Variance (ANOVA) similarly to [6]. ANOVA [94] is used to test the hypotheses below about the differences between two or more means.

$$\begin{aligned} H_0 &: \mu_1 = \mu_2 = \mu = \dots \mu_k \\ H_1 &: \text{at least one } \mu_i \neq \mu_j, \text{ for } i \neq j \end{aligned}$$

The value $\mu_1, \mu_2, \dots, \mu_k$ are the means of samples from the first, second, and k^{th} set of samples. The F test is to test the hypothesis that these mean values are equal. In our problem, we have $k = 2$. We use F test to see how the features from stego images differ from those from original image statistically. We perform the ANOVA test for each embedding scheme separately. The purpose is to see whether or not the features calculated based on the distortion measures will be affected statistically by the embedding schemes. We use F test as the test statistics with $k - 1$ and $N - k$ degree of freedom, where N is the total number of test images [94]. We reject H_0 if the score is larger than the critical value related with the confidence level. More details of this test can be found in [94].

In our problem, we use F test to evaluate how the features from stego images differ from those from original image statistically.

We apply the tests for the following embedding schemes: Tseng's scheme [101], Wu's scheme [112], Yang & Kot's scheme [117], Kim's scheme [67], Pan's scheme [88] and Mei's scheme [83]. In each of our tests, a vector of features calculated from cover images and a vector of features calculated from stego images using the same distortion measure are tested. The stego images are generated by hiding randomly generated message bits into the cover images by the embedding scheme. It is expected that the features from cover images are statistically different from those in the stego images. The following four distortion measures are used to calculate the distortion score: mean-square error (MSE), edge line segment similarity measure (ELSSM), distance-reciprocal distortion measure (DRDM) [75], change of smoothness and connectivity measure (CSCM) [22]. The results of ANOVA are given in Table 4.3. In this table, the ranking of the F score in each row indicates the ranking of the features in evaluating the difference between stego and cover images, where the highest score indicates the distortion measure based feature that responds most strongly. The confidence level we choose is 0.95. The critical value at this confidence level is around 4. Since all the scores are much larger than the critical value, we can make a conclusion that the features calculated from stego images are quite different from that from original images. From Table 4.3 we can see that the score calculated based on ELSSM is the highest in each row, which implies that the distortion score based on ELSSM is the best individual feature for discrimination.

Although ANOVA can test each individual measure, it has not taken the inter-correlation of these measures into consideration and some of the features

Table 4.3: F score of ANOVA.

Embedding scheme	MSE	ELSSM	DRDM (m=3)	DRDM (m=5)	CSCM (m=3)	CSCM (m=5)
Tseng's (16×16)	33361	39518	37986	39353	30950	25680
Yang & Kot's (3×3)	972	1746	1144	1148	946	823
Yang & Kot's (4×4)	1244	2149	1491	1466	1237	1135
Yang & Kot's (5×5)	1090	1818	1247	1237	1020	918
Wu's (8×8)	3816	6079	4585	4831	4401	2658
Wu's (12×12)	2299	3645	2891	3033	2507	1723
Wu's (16×16)	1850	3098	2256	2353	2033	1489
Mei's	256	296	271	284	100	73
Pan's	481	1023	568	569	649	524
Kim's	610	632	312	308	443	341

may be redundant. We use sequential forward floating search method (SFFS) [92] to search for the combination of image distortion measures that yields the best classification results. Suppose k features are selected to form set $F_k = \{f_1, f_2, \dots, f_k\}$. We have the following steps.

1. Select the most significant feature f_{k+1} from the available features to form new set F_{k+1} .
2. From the remaining features, add each feature by including which the classification result improves the most.
3. Find the least significant feature in the current feature set. If the last added feature is the least significant feature, then return step 2. Otherwise, remove the least significant feature to form a new set and repeat this step.
4. Stop when the classification result cannot be improved.

Table 4.4: Selected features from SFFS.

Embedding scheme	MSE	ELSSM	DRDM (m=3)	DRDM (m=5)	CSCM (m=3)	CSCM (m=5)
Tseng's (16×16)		✓				
Yang & Kot's (3×3)	✓	✓	✓	✓	✓	✓
Yang & Kot's (4×4)	✓	✓	✓	✓	✓	✓
Yang & Kot's (5×5)	✓	✓	✓	✓	✓	✓
Wu's (8×8)	✓	✓	✓	✓	✓	✓
Wu's (12×12)	✓	✓	✓	✓	✓	✓
Wu's (16×16)	✓	✓	✓	✓	✓	✓
Mei's	✓	✓	✓	✓		
Pan's	✓	✓	✓	✓	✓	✓
Kim's	✓	✓	✓	✓	✓	✓

In Table 4.4, we use '✓' to show the features that are selected by the sequential forward floating search method.

4.3.4 Support vector machine

The support vector machines (SVM) [27] are classification tools for solving machine learning problems based on recent advance in statistical learning theory. They are powerful tools for two-class classification. Denote the training data pair $\{\mathbf{f}_i, c_i\}$, $i = 1, 2, \dots, N$, $\mathbf{f}_i$ is a feature vector and c_i is either +1 or -1 for positive or negative patterns. Each $\mathbf{f}_i$ is a vector of scaled value to avoid variables in greater numeric ranges dominate those in smaller numeric ranges. Suppose a hyper-plane which separates the two classes exists, the data point $\mathbf{f}_i$ which lies on the hyper-plane satisfies

$$\mathbf{w}^T \mathbf{f}_i + b = 0 \quad (4.19)$$

where $\mathbf{w}$ is the normal to the hyper-plane [13, 122]. Let d_1 and d_2 be the distance from the hyper-plane to the nearest positive and negative data point respectively. The margin is defined as $d = d_1 + d_2$. In a linearly separable case, the support vector machine searches for the hyper-plane by maximize the margin d . It can be formulated as follows: if a separating hyper-plane exists, then all the training data satisfy the following constraints [13, 122]:

$$\mathbf{w}^T \mathbf{f}_i + b \geq 1 \text{ if } c_i = +1 \quad (4.20)$$

$$\mathbf{w}^T \mathbf{f}_i + b \leq -1 \text{ if } c_i = -1 \quad (4.21)$$

By introducing positive Lagrange multiplier γ_i , the following Lagrangian formulation can be constructed [13, 122]:

$$L = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^l \gamma_i [c_i (\mathbf{f}_i \cdot \mathbf{w} + b) - 1] \quad (4.22)$$

From the gradient of L with respect to $\mathbf{w}$ and b , the optimal value of $\mathbf{w}$ and b can be given by:

$$\mathbf{w} = \sum_{i=1}^N \gamma_i \mathbf{f}_i c_i \quad (4.23)$$

$$b = \frac{1}{N} \sum_{i=1}^N (c_i - \mathbf{w}^T \mathbf{f}_i) \quad (4.24)$$

Once we obtained the trained SVM, we use it to classify a given test image with feature $\mathbf{x}$. If $\mathbf{w}^T \mathbf{x} + b \geq 0$, the image is classified as stego image, other-

wise classified as original image. For non-linearly separable case, the learning machine maps the input feature vectors to a higher dimensional space where a linear hyper-plane is located. The transformation from the non-linear feature space to linear higher dimensional space is done by using kernel function. In our experiment, radial basis function is adopted.

4.3.5 Experimental results

To evaluate the proposed steganalysis technique, we have conducted experiments on more than 5000 images download from the website of the Laboratory for Engineering Man/Machine Systems (LEMS), Brown University [2]. A set of sampling test images are shown in Figure 4.27. The following six different embedding schemes are tested in our experiments: Tseng's scheme [101], Wu's scheme [112], Yang & Kot's scheme [117], Kim's scheme [67], Pan's scheme [88] and Mei's scheme [83]. In particular, for Wu's scheme and Yang & Kot's scheme, three different block sizes are used. Random messages are generated and embedded by different schemes to generate the stego images. We apply both de-noising methods to remove noise in different patterns to obtain the reference image for calculating the distortion score. We first evaluate the system with each embedding algorithm individually. We randomly select half of the original images and the corresponding stego images for the training. The rest are used for testing using the trained SVM. We do multiple tests to get the average of the test. The SVM implementation from LIBSVM [19] is used. The classification results from the two best individual features are given in Table 4.5. Our experiments show that other individual features give poor results.

The results based on the combination of features that yields the best results

Figure 4.27: Some example of test images from the database.

Table 4.5: The detection results for individual embedding scheme based on the individual features.

Embedding Scheme	ELSS		DRDM(m=5)	
	False Alarm	Missing Rate	False Alarm	Missing Rate
Tseng's (16×16)	0.0%	0.0%	0.0%	0.0%
Wu's (8×8)	5.4%	8.9%	7.7%	12.4%
Wu's (12×12)	8.3%	14.5%	10.5%	17.3%
Wu's (16×16)	13.4%	23.2%	16.4%	24.5%
Yang & Kot's (3×3)	13.2%	22.0%	16.1%	22.4%
Yang & Kot's (4×4)	11.1%	15.9%	15.3%	19.3%
Yang & Kot's (5×5)	12.5%	14.7%	17.5%	18.8%
Kim's	15.7%	33.4%	19.2%	35.7%
Pan's	14.8%	30.6%	18.9%	33.5%
Mei's	25.2%	40.3%	25.3%	39.8%

are given in Table 4.6. From the results, we can see that we can classify all stego images created by Tseng's method. It implies the limitation of the flipping that does not take smoothness into consideration. For other schemes which mainly use COL pixels, the capacity is the major reason that affects the results. For Mei's scheme, the detection performance is the worst as the payload is the lowest.

We also evaluate the steganalysis system for the ensemble of the different schemes. Ten different sets of original images with 500 images per set are embedded with the ten embedding scheme as shown in Table 4.6 (including using different block size). Half of the original images and the stego images created by various embedding schemes are used for training and the other half are used for testing. The results are given in Table 4.7. We can see that the trained steganalyzer can detection messages hidden by different embedding schemes. However, stego images generated by algorithms with relatively lower embedding payload such as Mei's scheme make the detection of hidden message

Table 4.6: The detection results for individual embedding scheme based on the best combination of features.

Embedding Scheme	False Alarm	Missing Rate	Average Flipping
Tseng's (16×16)	0.0%	0.0%	35
Wu's (8×8)	2.5%	4.9%	93
Wu's (12×12)	3.6%	10.0%	39
Wu's (16×16)	9.8%	19.4%	21
Yang & Kot's (3×3)	9.6%	19.0%	27
Yang & Kot's (4×4)	7.1%	13.9%	33
Yang & Kot's (5×5)	8.7%	16.1%	31
Kim's	10.7%	26.1%	15
Pan's	11.8%	22.6%	18
Mei's	22.8%	38.3%	13

Table 4.7: The detection results for ensemble schemes.

Embedding Scheme	False Alarm	Missing Rate
Ensemble	9.7%	19.9%

difficult as we can see from the missing rate at 19.9%.

4.4 Summary

In this Chapter, we first propose a new objective distortion measure for binary images by calculating the edge line segment similarity. This measure is proposed based on how the overall shape of edges are affected by flipping some of the pixels. Compared with DRDM which works well when flipping non-edge pixels is involved, our approach emphasizes the importance of connectivity which is a good distortion measure for flipping edge pixels. Another advantage is that the new measure considers a block with flexible size automatically instead of fixed 3×3 or 5×5 block which is often used in CSCM and DRDM. The distortion score by the proposed method is calculated based on the similarity between the edge

line segments in the original and distorted images. Our experimental results show strong correlation with subjective assessment. Besides using the measure to evaluate the performance of the pixel flipping algorithms in [21, 83, 101, 112, 113, 117], we can use this measure to select the most flippable pixel from clean text images or high quality binary graphic images when hiding information, provided that the selected pixels are at least one pixel apart from each other. This measure is suitable for evaluating distortion from individually flipped pixel only. It does not work for the cases where two 8-connected neighboring pixels are flipped simultaneously. It does not work for halftone images as halftone images as the edge lines in halftone images are different from those in text or cartoon images.

After the proposed distortion measure, we propose a steganalysis technique for binary cartoon images. The proposed detection approach is based on the hypothesis that the embedding of the message makes the image less smooth or predictable. The proposed method is suitable for the steganalysis of high quality cartoon images. Two de-noising process for binary images are proposed to remove noisy pixels added into different image patterns. The first de-noising process based on opening and closing process is applied to fill holes and remove protrusions due to data hiding. The second de-noising method by shifting corners in the stair case patterns is proposed to recover the flipped pixels. By flipping these pixels, corners are shifted. The new proposed edge line segment similarity measure together with other binary image distortion measures are used to calculate the distortion scores, which are used as the features for classification. ANOVA is used to analyze the features to distinguish original cover images and stego images. The ANOVA results show that the distortion score between an original image and its de-noised version is different from that be-

tween a stego image and its de-noised version. Thus the proposed de-noising process are effective in removing the noise. A sequential forward floating search method is used to search for the combination of the features that yields the best classification results. Support vector machines are used to classify the stego images from original images. The experimental results show that the steganalysis of schemes that mostly use COL pixels is more difficult. However, for binary graphic images which contain stair case patterns, we can use of the location correlation among close corners for the detection. It implies that, besides the smoothness and connectivity, the selection of flippable pixels should also take the location correlation among consecutive corners along the same edge into consideration. The performance could be improved with a better de-noising technique or a better distortion measure. As we assume the edge contour of an image should be smooth before flipping, the detection scheme is not suitable for scanned images.

Chapter 5

Halftone Image

5.1 Introduction

As we have discussed in Section 2.2, different data hiding schemes for halftone images are proposed while detection of data hiding in halftone image is only discussed in [53]. The proposed method in [53] is based on an average low pass filtering followed by a wavelet statistic feature extraction and a fisher linear discrimination classifier. A low pass filtering is used instead of inverse halftoning based on the comment that “inverse halftoning techniques do not give much benefit since they often smooth away the ‘noise’ caused by the embedded message” [53]. However, we find that this is not always the case and some inverse halftoning algorithms can highlight the embedding noise instead of smoothing it away and thus provide more accurate results in detecting hidden messages.

In this Chapter, we present a steganalysis technique for halftone image based on inverse halftoning. Halftone image is a special type of binary image different from binary cartoon image or text image where edge lines are distinctive.

Halftone image is generated by applying halftoning process on a gray level image to make it look like a gray level image.

There are two main kinds of halftoning methods: ordered dithering [7] and error diffusion [34]. Denote an 8-bits $M \times N$ natural gray level image as $\mathbf{f} = \{f(i, j), i = 0, 1, \dots, M - 1, j = 0, 1, \dots, N - 1\}$ and the corresponding output halftone image of the halftoning process as $\mathbf{b} = \{b(i, j), i = 0, 1, \dots, M - 1, j = 0, 1, \dots, N - 1\}$. In ordered dithering, the pixel $f(i, j)$ of the gray level image $\mathbf{f}$ is compared with a threshold pattern or screen to determine the binary output, i.e.,

$$b(i, j) = \begin{cases} 255 & f(i, j) > T(i, j) \\ 0 & \text{otherwise} \end{cases} \quad (5.1)$$

where $T(i, j)$ is a varying threshold determined by the dither screen and 0 and 255 represent the values of black and white pixels respectively. A commonly used dithering screen in [20] is shown in Table 5.1. There are two major variants of ordered dithering [7, 102]:

1. Dispersed dot dithering: If consecutive threshold values are placed far from each other within the matrix, halftoning of an image will produce dispersed micro-dots.
2. Clustered dot dithering: If consecutive threshold values are placed in a special sequence within the matrix, halftoning of an image will produce larger clustered dots.

Halftoning by error diffusion is more complicated. The halftoning error is fed forward to its adjacent neighbors using a kernel, and the sum of image pixel

Table 5.1: 8×8 Bay's dispersed dither matrix.

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	52	20	62	30	54	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	55	23	61	29	53	21

intensity and the error from the past pixels is compared with a fixed threshold T to determine the binary output. To implement error diffusion, one can compute the following three equations for each pixel in the scanning order [34]:

1. Calculate $u(i, j)$ as:

$$u(i, j) = f(i, j) + \sum_{m, n} k(m, n) e(i - m, j - n) \quad (5.2)$$

where $k(m, n)$ is determined by the kernel being used. Two commonly used kernels, i.e., the Jarvis kernel and the Floyd and Steinberg kernel [102], are shown in Table 5.2 and Table 5.3, respectively.

2. Compute binary output as:

$$b(i, j) = \begin{cases} 255 & u(i, j) > T \\ 0 & \text{otherwise} \end{cases} \quad (5.3)$$

3. Compute the halftone error at pixel (i, j)

$$e(i, j) = u(i, j) - b(i, j) \quad (5.4)$$

Table 5.2: Jarvis kernel.

			7	5
3	5	7	5	3
1	3	5	3	1

Table 5.3: Floyd and Steinberg kernel.

		7
3	5	1

As we mentioned in Chapter 2, data hiding in a halftone image can be classified into two groups depending on the knowledge of the original gray level image. The first kind of approaches of data hiding in a halftone image is done by flipping the pixels in the halftone image from black to white or vice versa without the knowledge of original gray level image. In [41, 42], data hiding self toggling (DHST), data hiding pair toggling (DHPT) and data hiding smart pair toggling (DHSPT) are proposed. In DHST, the information is hidden at pseudorandom locations in the halftone image by forcing the pixels at these locations to be black or white. In DHPT, DHST is applied first and a complementary pixel is chosen randomly for flipping from the 3×3 neighborhood of each flipped pixel during DHST. By doing so, the local intensity is maintained. The DHSPT improves on DHPT by choosing the complementary pixel to achieve smaller salt and pepper clusters. Later in [43] the idea of data hiding with intensity selection is proposed to select the best location from a set of candidate locations for embedding each information bit. This idea is applied to DHST, DHPT and DHSPT to get three new algorithms: data hiding self toggling with intensity selection (DHST-IS), data hiding pair toggling with intensity selection (DHPT-IS) and data hiding smart pair toggling with intensity selection (DHSPT-IS). Generally, DHST is the basic scheme to hide information. It is also discussed in some other papers

for multi-resolution binary image embedding [109] or integrity protection and authentication [66] by using DHST. These DHST based schemes can be applied to most of halftone images; however, when they applied to clustered dot halftone images, clusters in the halftone image will be destroyed and visible distortion such as holes will be created in the clustered dot very often.

The second kind of approaches of data hiding in a halftone image is done with the knowledge of the original gray level image. Instead of flipping pixels, these data hiding methods usually modify the halftoning method to obtain the halftone image that contains hidden message. In [44], data hiding error diffusion (DHED) and modified data hiding error diffusion (MDHED) are proposed by modifying the error diffusion algorithm. In both methods, the original gray level image is known and the pixel values at the locations determined by the pseudorandom key are first determined according to the message to be hidden. Then the error diffusion algorithm is applied to obtain the binary output for the rest of the pixels. The error due to the embedding is also considered in the error diffusion. In DHED, the embedding error is fed forward. In MDHED, the embedding error is fed both forward and backward. Although the original gray level image is needed in the embedding, the extraction of hidden message is the same as that for DHST, i.e., the extraction of hidden message only needs the stego halftone image with hidden information and the pseudorandom locations used to hide the message. As this kind of data hiding approaches do not rely any other side information in the extraction, steganalysis is worthwhile with the same reason as for the first kind of data hiding approaches. These two types of data hiding algorithms can be simplified and summarized as shown in Figure 5.1. The input and output of the halftoning process $H(\cdot)$ are denoted as $\mathbf{f}$ and $\mathbf{b}$ respectively. Both $\mathbf{b}$ and $\mathbf{f}$ are possibly the input of the embedding process

Figure 5.1: The embedding system.

$E(\cdot)$, whose output is denoted as $\mathbf{s}$. We have:

$$\mathbf{b} = H(\mathbf{f}) \quad (5.5)$$

$$\mathbf{s} = E(\mathbf{f}, H(\mathbf{f})) \quad (5.6)$$

Since a halftone image is the output of the halftoning process, it may not be appropriate to consider any halftone image as the original image. Instead, we consider a halftone image output of a predefined halftoning process as a natural halftone image when the input is a natural gray level image. The halftone image with a hidden message is called unnatural or stego halftone image.

In this Chapter, we propose a general technique for the steganalysis of halftone images. We focus on dispersed dot ordered dithering and error diffusion images with the commonly used screens or kernels. For halftone image such as clustered dot ordered dithering image, there is no good data hiding algorithm for clustered dot ordered dithering image to the best of our knowledge. The reason is that randomly flipping of pixels from clustered dot image may create holes in the clusters of the image, as we have discussed earlier. In section 5.3, filtering based inverse halftoning and its performance in steganalysis is investigated. In section 5.4, projection algorithm is applied and its performance in steganalysis is also investigated. The summary is given in the last section.

5.2 A general method

Inspired by the steganalysis techniques [53, 77] for gray level image and the fact that a halftone image is obtained by applying halftoning process on a gray level image, we extend the scheme in [53] to a general method as shown in Fig. 5.2 for the steganalysis of halftone images. We apply inverse halftoning on the halftone image to reconstruct the gray level image. The original natural gray level image has certain image statistics which may still be retained in the reconstructed image, while the data hiding process in a halftone image is likely to change some of these statistics. In what follows, we focus on the selection of different inverse halftoning processes and study their application in steganalysis. We first define the inverse halftoning process as $G(\cdot)$ with halftone image $\hat{\mathbf{b}}$ and inverse halftoning gray level image $\hat{\mathbf{f}}$ as the input and output of $G(\cdot)$ respectively.

$$\hat{\mathbf{f}} = G(\hat{\mathbf{b}}) \quad (5.7)$$

In the steganalysis, $\hat{\mathbf{b}}$ is either a natural halftone image $\mathbf{b}$ or a stego image $\mathbf{s}$. The purpose of steganalysis is to identify if the input image $\hat{\mathbf{b}}$ is a natural halftone image $\mathbf{b}$.

We define the halftone noise $n_H = G(H(\mathbf{f})) - \mathbf{f}$ and embedding noise $n_E = G(E(\mathbf{f}, H(\mathbf{f}))) - G(H(\mathbf{f}))$. After applying inverse halftoning process, the focus of steganalysis is now on gray level image, i.e., differentiate $G(H(\mathbf{f}))$ from $G(E(\mathbf{f}, H(\mathbf{f})))$. The difficulty is due to the introduction of halftoning noise. The task here is to determine the existence of embedding noise with the presence of halftoning noise. A good inverse halftoning process for the use of steganalysis should keep the trace of embedding, i.e, the embedding noise. As halftoning

Figure 5.2: Flowchart of the detector.

noise could be mistaken as the embedding noise, it could be beneficial to minimize the halftoning noise by choosing a proper inverse halftoning algorithm.

5.2.1 Feature extraction and classification

It has been shown in [77] that the wavelet image statistics defined in [77] are effective for the steganalysis of gray level images. In this Chapter, we use the same statistic features to differentiate the reconstructed gray level images based on natural halftone images from the reconstructed gray level images based on stego halftone images.

These are basically subband/wavelet image statistics obtained by decomposing the image recursively based on quadrature mirror filters (QMF). The image is first decomposed into vertical, horizontal, diagonal and low frequency subbands by convolving the image with the low-pass and high-pass QMF along the two image axes. The decomposition is obtained next by first down-sampling the last step low frequency subband by 2 and then followed by the QMF. The vertical, horizontal, and diagonal subbands at each step i are denoted as V_i , H_i , D_i respectively, for $i = 1, 2, \dots, K$. A set of predicted coefficients EV_i, EH_i, ED_i for V_i , H_i , D_i are also calculated. The mean, variance, skewness and kurtosis of these $6K$ coefficients including V_i , H_i , D_i , EV_i , EH_i , and ED_i , for $i = 1, 2, \dots, K$, are then calculated as the features. In our implementation, we use $K = 3$ to obtain 72 dimension feature as used in [77].

Similar to Chapter 4, we use SVM [27] as a two-class classifier and we use the implementations in [19] in our tests. Our problem is to differentiate stego images from natural halftone images by examining the statistic features from their reconstructed gray level images.

5.3 Filtering based methods

One type of inverse halftoning algorithms is the filtering based techniques. Methods based on this technique usually aim at producing a gray level image $\hat{\mathbf{f}}$ such that it appears similar to the original image $\mathbf{f}$ by minimizing the error in a statistical sense between $\hat{\mathbf{f}}$ and $\mathbf{f}$.

5.3.1 Fixed low pass filter

A basic form of inverse halftoning is to use a finite impulse response (FIR) low pass filter. We first observe the detection performance when a low pass filter is applied in the inverse halftoning process. In order to minimize the halftone noise n_H , we find a suitable filter that minimizes the mean square error (MSE) between $\hat{\mathbf{f}}$ and $\mathbf{f}$. Suppose a low pass filter with an impulse response $\mathbf{W} = \{W(m, n), -N_W \leq m, n \leq N_W\}$ is applied to a binary image $\hat{\mathbf{b}}$, we have

$$\hat{f}(i, j) = \sum_{m=-N_W}^{N_W} \sum_{n=-N_W}^{N_W} W(m, n) \hat{b}(i - m, j - n) \quad (5.8)$$

When evaluated throughout all possible pixels (i, j) from an $M \times N$ image, (5.8) can be expressed in matrix representation as:

$$\hat{\mathbf{f}} = \mathbf{Q}\mathbf{w} \quad (5.9)$$

where the vector $\hat{\mathbf{f}}$ and $\mathbf{w}$ are formed by stacking all possible elements of $\hat{\mathbf{f}}$ and $\mathbf{W}$, respectively. The matrix $\mathbf{Q}$ contains all the coefficients $\hat{\mathbf{b}}$ as specified in (5.8).

$$\begin{aligned} \mathbf{w} &= \begin{bmatrix} w(0) \\ \vdots \\ w(i) \\ \vdots \\ w(4N_W^2 + 4N_W) \end{bmatrix} \\ &= \begin{bmatrix} W(-N_W, -N_W) \\ \vdots \\ W(\lfloor \frac{i}{2N_W+1} \rfloor - N_W, \text{mod}(i, 2N_W + 1) - N_W) \\ \vdots \\ W(N_W, N_W) \end{bmatrix} \end{aligned} \quad (5.10)$$

where $\lfloor \cdot \rfloor$ is the floor operation and $\text{mod}(x, y)$ is the modular operation of x over y , $w(i)$ is the i^{th} element of $\mathbf{w}$ and $W(x, y)$ represents the element of $\mathbf{W}$ at

(x, y) .

$$\begin{aligned}
 \hat{\mathbf{f}} &= \begin{bmatrix} \hat{\mathbf{f}}(0) \\ \vdots \\ \hat{\mathbf{f}}(i) \\ \vdots \\ \hat{\mathbf{f}}((M - 2N_W)(N - 2N_W) - 1) \end{bmatrix} \\
 &= \begin{bmatrix} \hat{f}(N_W, N_W) \\ \vdots \\ \hat{f}(\lfloor \frac{i}{N-2N_W} \rfloor + N_W, \text{mod}(i, N - 2N_W) + N_W) \\ \vdots \\ \hat{f}(M - N_W - 1, N - N_W - 1) \end{bmatrix} \quad (5.11)
 \end{aligned}$$

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_0, & \mathbf{q}_1, & \dots, & \mathbf{q}_j, & \dots, & \mathbf{q}_{4N_W^2+4N_W} \end{bmatrix} \quad (5.12)$$

where

$$\begin{aligned} \mathbf{q}_j &= \begin{bmatrix} q(0, j) \\ \vdots \\ q(i, j) \\ \vdots \\ q((M - 2N_W)(N - 2N_W) - 1, j) \end{bmatrix} \\ &= \begin{bmatrix} \hat{b}(2N_W - \lfloor \frac{j}{2N_W+1} \rfloor, 2N_W - \text{mod}(j, 2N_W + 1)) \\ \vdots \\ \hat{b}\left(\lfloor \frac{i}{N-2N_W} \rfloor - \lfloor \frac{j}{2N_W+1} \rfloor + 2N_W, \right. \\ \left. \text{mod}(i, N - 2N_W) - \text{mod}(j, 2N_W + 1) + 2N_W\right) \\ \vdots \\ \hat{b}\left(M - 1 - \lfloor \frac{j}{2N_W+1} \rfloor, N - 1 - \text{mod}(j, 2N_W + 1)\right) \end{bmatrix} \end{aligned}$$

The unknown $\mathbf{w}$ is determined by the least square estimation similar to [77] to get

$$\mathbf{w} = (\mathbf{Q}^T \mathbf{Q})^{-1} \mathbf{Q}^T \mathbf{f} \quad (5.13)$$

where $\mathbf{f}$ is formed by stacking all possible elements of the gray level image $\mathbf{f}$.

Although different images may yield different $\mathbf{w}$'s, or equivalently, $\mathbf{W}$'s, to get the minimum mean square error, the difference is small and a low pass filter obtained from one image can be applied to another image [20]. Similar to [20], we use the standard 512×512 Lena, Pepper and Jet images to obtain $\mathbf{W}$. The size of the FIR filter $\mathbf{W}$ needs to be selected first. Table 5.4 shows the mean square error (MSE) and the peak signal to noise ratio (PSNR) between the original gray level images and the reconstructed gray images from different

Table 5.4: Minimum MSE (left) and Maximum PSNR (dB) (right) achieved by FIR filters with different sizes for three standards images.

Filter size	5×5		7×7		9×9		11×11	
Lena	99.6	64.89	93.7	65.50	92.0	65.69	91.9	65.70
Pepper	98.1	65.04	90.6	65.84	88.7	66.05	87.8	66.15
Jet	117.1	63.27	114.0	63.54	109.8	63.91	109.4	63.95

halftone images when different sizes of the FIR filter are selected. Generally, larger size filters produce smaller MSE, however, when the size is larger than 9×9 , the improvement is minimal. In our implementation, we choose to use the 9×9 size and obtain three filters based on the standard Lena, Pepper and Jet images. The average of the three filters is used. It should be noted that the low pass filters that minimize the MSE or Maximize the PSNR between the natural and stego halftone images generated by different halftoning methods are obtained separately.

5.3.2 Experimental results

We use the 2812 natural continuous tone gray level image from the website of Vision Research Lab, University of California, Santa Barbara [1] as the test images. These gray level images are used to generate natural and stego halftone images. Three halftoning methods including one ordered dithering algorithm using the 8×8 Bay's Dither matrix in [20] and the two error diffusion algorithms with Jarvis kernel and Steinberg kernel are used to generate the natural halftone images. For ordered dithering images, the previously mentioned six embedding algorithms including DHST, DHPT, DHSPT, DHST-IS, DHPT-IS, and DHSPT-IS are used to generate stego images for the detection. For error diffusion images, two additional embedding schemes DHED and MDHED men-

Table 5.5: Classification results by the filtering based method.

Halftoning method	Ordered dithering		ED(Javis)		ED(Steinberg)	
Stego algorithm	TN	TP	TN	TP	TN	TP
DHST	99.29	98.72	91.39	93.31	95.30	94.88
DHPT	100.00	99.29	89.33	90.97	98.72	98.36
DHSPT	100.00	99.29	91.25	92.60	94.52	93.88
DHST-IS	99.36	98.36	92.82	92.11	95.09	94.81
DHPT-IS	100.00	99.29	90.11	89.69	93.17	91.96
DHSPT-IS	100.00	99.22	91.75	91.39	93.46	92.67
DHED	-	-	77.24	78.81	86.56	85.06
MDHED	-	-	78.88	82.86	84.14	83.64
Combine	99.64	98.57	86.62	88.34	91.89	90.11

tioned earlier are tested, in addition to the above 6 embedding algorithms. We keep the same capacity as that in [44], i.e., $\frac{4096}{512^2}MN = \frac{1}{64}MN$ bits for image size of $M \times N$. We first evaluate individually the fixed low pass filter based detection scheme with each embedding algorithm and select randomly half of the natural halftone images and their corresponding stego images for training. The rest of the natural and stego halftone images are kept for testing using the trained SVM classifier. The results are shown in Table 5.5, where the true negative (TN) is the probability that a natural image is classified as natural image and the true positive (TP) is the probability that a stego image is classified as stego. From these results we can see that the detection of data hidden by DHED and MDHED is more difficult than the detection of data hidden by other schemes which hide data without the availability of original gray level image. This is expected as DHED and MDHED schemes make use of the original gray level image and the resultant stego image contains less embedding noise.

As the trained SVM classifier based on natural halftone images and stego halftone images generated based on one embedding scheme may not be ap-

plicable for detecting hidden data embedded by other embedding scheme, a combining test for different embedding schemes is important. In a testing of K embedding algorithms, all the natural images are divided into K groups with the same number of images. These K groups of natural images are embedded with the K embedding schemes respectively to generate K groups of stego images. Half of the natural images from each group and their corresponding stego images are used for the training and the other half are used for the testing. We perform tests for the halftone images generated by different halftoning methods. The results are shown in the last row in Table 5.5. From these results, we can see that the performances of the detector is related to the halftoning method and the detection of data hidden in ordered dithering images is easier than that in error diffusion images.

It is noted that the trained SVM classifier based on halftone images generated by one halftoning method may not work well for halftone images generated by another halftoning method and the halftone method is assumed to be known when the trained SVM classifier is applied in the above tests. Steganalysis without knowing the halftoning method used is more difficult as the statistics for halftone images generated by one halftoning method could be quite different from those generated by another halftoning method. To remove such a constraint in making the trained SVM classifier more robust, we need to use SVM to train halftone images generated by different halftoning methods, i.e., a two-class classifier with training images in each class including images by different embedding schemes combined with different halftoning approaches. We combine the natural halftone images generated by different halftoning methods as well as the corresponding stego halftone images generated by various embedding scheme in the training. In our experiments, we perform tests on the three

halftoning methods combined with 8 embedding schemes(excluding DHED and MDHED for ordered dithering images as DHED and MDHED are for error diffusion halftone images only) and get an overall detection accuracy of 92.05% for natural image and 92.87% for stego image. The results from the test are slightly worse than that the result of the average of all the individual tests; however, the trained SVM classifier can work for all the halftoning methods.

5.3.2.1 Comparison with other filters

A fixed filter has its limitation in minimizing the mean square error, instead, adaptive inverse halftoning algorithm using a varying filter is proposed in [20], where image pixels are first classified into low, middle and high variance pixels according to their local variances and then three different filters are applied on the three groups of pixels respectively. It has been shown in [20] that partitioning the pixels into three categories including low, middle and high variance is sufficient to decrease the mean square error. Besides the varying low pass filter, other filters such as an average filter is proposed in [53] where heuristically selected average filter is used for steganalysis of the DHST scheme [66] in an ordered dithering image.

Table 5.6 shows the classification results (it combines all different embedding schemes) when an average low pass filter and a varying low pass filter are used. Comparing Table 5.6 with the result in the last row in Table 5.5, the average filter and varying filter provide worse results than those obtained using the fixed FIR filter.

Table 5.6: Classification results using different filters. The classifier are trained by combining different embedding schemes.

Halftoning method	Ordered dithering		ED(Javis)		ED(Steinberg)	
Filter	TN	TP	TN	TP	TN	TP
Average filter	93.55	94.47	80.13	81.34	83.11	84.97
Varying filter	96.72	96.56	86.91	87.69	88.67	89.46

5.4 Projection based methods

Different from filtering based method, the inverse halftoning problem in the projection approach is defined as follows. For a given halftone image $\mathbf{b}$ obtained from $\mathbf{f}$, the purpose of the projection based inverse halftoning is to find a gray level image $\hat{\mathbf{f}}$ such that it satisfies

$$H(\hat{\mathbf{f}}) = \mathbf{b} \quad (5.14)$$

in order to obtain the reconstructed image close to the original.

5.4.1 Projection for ordered dithering image

Inspired by the projection algorithm for error diffusion [108], we apply a projection algorithm for an ordered dithering image. Let $\tilde{\mathbf{f}}$ be an approximation of $\mathbf{f}$. The projection algorithm includes the following steps:

1. Set (i, j) to be the first location in the scan direction of ordered dithering.
2. Perform one step of ordered dithering on $\tilde{f}(i, j)$ to obtain:

$$\tilde{b}(i, j) = \begin{cases} 255 & \tilde{f}(i, j) > T(i, j) \\ 0 & \text{otherwise} \end{cases} \quad (5.15)$$

Figure 5.3: Center 100×100 pixel portion of reconstructed images from ordered dithering based natural and stego halftone image using projection method. The images in (a)-(f) are reconstructed from stego images with 4096 bits hidden by DHST, DHPT, DHSPT, DHST-IS, DHPT-IS and DHSPT-IS respectively. The image in (g) is reconstructed from natural ordered dithering image.

where $T(i, j)$ is the threshold determined by the halftoning kernel or the dithering screen.

3. If $\tilde{b}(i, j) = b(i, j)$, go to step 2 for the next pixel until the last pixel of the image.
4. If $\tilde{b}(i, j) \neq b(i, j)$, adjust $\tilde{f}(i, j)$ such that

$$\tilde{f}(i, j) = \begin{cases} T(i, j) + 1 & \text{if } b(i, j) = 255 \\ T(i, j) & \text{if } b(i, j) = 0 \end{cases} \quad (5.16)$$

The adjustment in step 4 sets $\tilde{b}(i, j) = b(i, j)$. We apply the varying low pass filter in [20], i.e., different filtering for each of the low, medium, and high-frequency regions, to obtain the approximated $\tilde{\mathbf{f}}$ of $\mathbf{f}$ as it provides better approximated $\tilde{\mathbf{f}}$ for a natural image $\mathbf{f}$ than other two filters based on the results in [20]. In the following, we show how the projection method helps differentiate the natural images with stego images. Figure 5.3 shows the center 100×100 pixel portion of reconstructed images from natural and stego ordered dithering halftone images. It can be observed clearly that the reconstructed images in Figure 5.3(a)-5.3(f) from stego halftone images contain more visible high frequency noise than the reconstructed image in Figure 5.3(g) from natural halftone image.

5.4.2 Projection for error diffusion image

A projection method is proposed in [108] to perform inverse halftoning for the error diffusion halftone images. The key of the projection method is based on the following projection algorithm. Let $\tilde{\mathbf{f}}$ be an approximation of $\mathbf{f}$, the projection algorithm can be described as follows [108]:

1. Set (i, j) to be the first location in the scan direction of error diffusion
2. Perform one step of error diffusion using $\tilde{f}(i, j)$ to obtain

$$\tilde{b}(i, j) = \begin{cases} 255 & u(i, j) > T \\ 0 & \text{otherwise} \end{cases} \quad (5.17)$$

where

$$u(i, j) = \tilde{f}(i, j) + \sum_{m, n \in S} k(m, n) e(i - m, j - n) \quad (5.18)$$

and

$$e(i - m, j - n) = u(i - m, j - n) - \tilde{b}(i - m, j - n) \quad (5.19)$$

3. If $\tilde{b}(i, j) = b(i, j)$, go to step 2 for the next pixel until the last pixel of the image.
4. If $\tilde{b}(i, j) \neq b(i, j)$, adjust $\tilde{f}(i, j)$ such that

$$\begin{aligned} u(i, j) &= \tilde{f}(i, j) + \sum_{m, n \in S} k(m, n) e(i - m, j - n) \\ &= \begin{cases} T + 1 & \text{if } b(i, j) = 255 \\ T & \text{if } b(i, j) = 0 \end{cases} \end{aligned} \quad (5.20)$$

5. Replace $e(i, j) = u(i, j) - b(i, j)$ and repeat step 2 for the next pixel.

As suggested in [108], a single-pass projection is performed by a half-band low pass filtering, a nonlinear statistical smoothing and the projection algorithm given above. A k -passes projection can be applied by repeating the single pass

projection k times. A good approximation of the original $\mathbf{f}$ is usually needed before applying the projection. In our implementation, we use the varying low pass filter in [20] to obtain an approximation of $\mathbf{f}$ as it provides better approximated $\tilde{\mathbf{f}}$ for a natural error diffusion image $\mathbf{f}$ than other two filters based on the results in [20]. After that, 8-passes projection used in [108] is applied.

Figure 5.4 and Figure 5.5 show the center 100×100 pixel portion of the reconstructed images obtained from natural and stego error diffusion halftone images using projection based inverse halftoning method. It can be observed that the reconstructed images in Figure 5.4(a)-5.4(g) and Figure 5.5(a)-5.5(g) from stego halftone images contain more visible high-frequency noise than the reconstructed images in Figure 5.4(i) and 5.5(i). For the reconstructed images in Figure 5.4(h) and 5.5(h), we cannot clearly observe the differences, especially the former one.

5.4.3 Experimental results

Although the visual inspection on the reconstructed images can differentiate natural halftone images from stego ones for most cases, computer detection is still needed. We still use the same subband/wavelet image statistics of the reconstructed images obtained previously in Section 5.2.1. We perform similar tests using SVM as before. The results from the individual tests on different embedding schemes are shown in Table 5.7, where the true negative (TN) is the probability that a natural image is classified as natural image and the true positive (TP) is the probability that a stego image is classified as stego. The results when combine different embedding schemes in the training are shown in the last row in Table 5.7. From the results, we observe that the projection algo-

Figure 5.4: Center 100×100 pixel portion of reconstructed images from error diffusion with Jarvis kernel based natural and stego halftone images with error diffusion using Jarvis kernel using projection method. The images in (a)-(h) are reconstructed from stego images with 4096 bits hidden by DHST, DHPT, DHSPT, DHST-IS, DHPT-IS, DHSPT-IS, DHED and MDHED respectively. The image in (i) is reconstructed from natural error diffusion image with Jarvis kernel.

Figure 5.5: Center 100×100 pixel portion of reconstructed images from error diffusion with Steinberg kernel based natural and stego halftone images using projection method. The images in (a)-(h) are reconstructed from stego images with 4096 bits hidden by DHST, DHPT, DHSPT, DHST-IS, DHPT-IS, DHSPT-IS, DHED and MDHED respectively. The image in (i) is reconstructed from natural error diffusion image with Steinberg kernel.

Table 5.7: Classification results by the projection based method.

Halftoning method	Ordered dithering		ED (Javis)		ED(Steinberg)	
Stego algorithm	TN	TP	TN	TP	TN	TP
DHST	100.00	99.93	93.10	95.23	99.64	99.50
DHPT	100.00	100.00	97.30	97.37	99.50	100.00
DHSPT	100.00	100.00	97.16	97.58	99.36	99.36
DHST-IS	100.00	99.93	93.88	96.02	99.72	99.93
DHPT-IS	100.00	100.00	97.01	96.73	99.08	99.28
DHSPT-IS	100.00	100.00	97.44	97.37	99.29	99.15
DHED	-	-	95.38	96.30	99.64	99.57
MDHED	-	-	86.20	89.05	94.31	95.95
Combine	100.00	99.93	93.81	93.60	97.65	97.15

algorithm increases the classification accuracy for all embedding schemes especially for DHED and MDHED.

Noticed that in the above, we have assumed that we have the knowledge of the halftone kernel (the dither screen or the error diffusion kernel) being used in the projection procedure. In practice, this information may not be available and we may need alternative solutions. We first show what happens if the halftone image is obtained with one kernel and the reconstruction is done based on another kernel. Figure 5.6 shows the reconstructed images using 8×8 Bay's screen [20] in the projection for natural and stego ordered dithering halftone image generated using 4×4 Bay's screen [7] in the halftoning. It shows that using a different dither screen in the reconstruction may result in high frequency noise even if the image does not contain hidden message. However, the noise appears periodic while the high frequency noise due to embedding appears random. Thus, it is not a must to have the exact same form of the dither screen as that has been used in the halftoning process when applying the projection algorithm for ordered dithering images. Here, it should be noticed

Figure 5.6: The reconstructed images from natural and stego images generated by ordered dithering with 4×4 Bay's screen. (a)Reconstruct from natural image with 4×4 screen being used in the projection. (b)Reconstruct from natural image with 8×8 screen being used in the projection. (c)Reconstruct from stego image with 4×4 screen being used in the projection. (d)Reconstruct from stego image with 8×8 screen being used in the projection.

that we can observe some visible high frequency noise in the reconstructed image in Figure 5.6(a) while we cannot observe similar noise in Figure 5.3(g). It is because the varying filter being used to obtain the reconstructed image in Figure 5.6(a) is obtained based on the halftone image generated by the 8×8 dither screen instead of 4×4 dither screen as we assume we do not know the halftone kernel. In our implementation, we have done the tests for the 4×4 and 8×8 screens and we can get almost the same results compared with that in Table 5.7.

We perform similar analysis on error diffusion images. Figure 5.7 shows the reconstructed images using Jarvis kernel in the projection for halftone im-

Figure 5.7: The reconstructed images from natural and stego images generated by error diffusion with Steinberg kernel. (a)Reconstruct from natural image with Steinberg kernel being used in the projection. (b) Reconstruct from natural image with Jarvis kernel being used in the projection. (c) Reconstruct from stego image with Steinberg kernel being used in the projection.(d) Reconstruct from stego image with Jarvis kernel being used in the projection.

ages generated by Floyd and Steinberg kernel. It is observed that when a different kernel is applied in the reconstruction of a natural halftone image, the reconstructed image contains small noise distributed throughout the image. Moreover, for an unnatural image source, high frequency noise can be observed. When the right kernel is used, the reconstructed image is usually very smooth if the halftone image does not contain hidden message. If the halftone image contains hidden message, the reconstructed image contains high frequency noise in smooth regions. Similar to the case for ordered dithering images, we train the SVM classifier that works for L different error diffusion kernels. We fix the kernel in the projection algorithm to be Jarvis kernel. Then L different kernels are used to generate natural halftone images for the training. Our tests on the Floyd and Steinberg kernel shows that the classification results are slightly worse (2-3% less) when compared with the results in Table 5.7.

5.4.3.1 Comparison with filtering based method at different capacity

In the above, we have done the tests for a fixed capacity. In order to evaluate the performance of the detectors for different capacity as well as the comparison with filtering based method, we carry out the tests on the same 2812 images from the website of Vision Research Lab, University of California, Santa Barbara [1] at different embedding rates (actual-message-length/maximum-message-length) at 0.1, 0.2, ..., 0.9 as well as at 0.01, 0.02 and 0.05, where the maximum message length is $\frac{1}{64}MN$ bits.

Figure 5.8 shows the average accuracy $\frac{1}{2}(\text{TP}+\text{TN})$ of the fixed filter based and projection based detectors at different embedding rates. We can see that the projection algorithms improve the detectors' performance for halftone images

Figure 5.8: Average accuracy of the detectors at different embedding rates.

generated by different halftoning methods at different embedding rates. The performance of the filtering based detector drops faster than that of projection based detector as the embedding rate decreases. Especially for ordered dithering image, the projection based detector can still provide a high accuracy for low embedding rate as 0.01.

Finally, let's look at how the different inverse halftoning techniques affect the halftone noise and embedding noise as well as the average accuracy of the detectors. Table 5.8 show the noise level of halftone noise (MSE between $\mathbf{f}$ and $G(\mathbf{b})$) and embedding noise (MSE between $G(\mathbf{s})$ and $G(\mathbf{b})$) as well as the accuracy for different inverse halftoning methods. The projection based method improves the accuracy largely because it exaggerates the embedding noise especially for ordered dithering image. In fact, the projection method is a filtering method followed by the projection algorithm, where the filtering method tends to smooth away the halftone noise as well as part of the embedding noise and the projection algorithm recovers some embedding noise.

Although we show the advantages of projection based detection scheme over filtering based detection scheme through experimental results, it is important to know the essential reason behind it. When a specific halftoning kernel is used to generate a halftone image, it will leave some information to the halftone image. Let us use an ordered dithering image as an example. Recall that an ordered dithering image is generated by binarizing the value of each pixel in a gray level with a varying threshold of a periodically used halftoning kernel. When a gray pixel is compared with a relatively low threshold, it has larger probability to be 1 and the vice versa. Thus, the probability distribution of 1s and 0s in a halftone image is determined by the halftoning kernel. However, existing halftone image data hiding schemes have not considered this point and the selection of flipping pixels is normally done using a pseudorandom key, independent with the halftoning kernel. Thus, the data hiding will change the probability distribution of 1s and 0s in the halftone image. From this, we can see that the next generation steganographic techniques should consider the probability of the pixels be 1 and 0 when select flippable pixels.

5.5 Summary

In this Chapter, we have proposed a general method for steganalysis of halftone images. The proposed method utilizes inverse halftoning algorithms to reconstruct the gray level images. The wavelet statistic features [77] calculated from the reconstructed images are used in the detection with SVM as the classifier. The experimental results show that these wavelet statistic features from the reconstructed gray level images are useful for steganalysis of halftone images. The filter based and projection based inverse halftoning algorithms have been

Table 5.8: Relationship between halftoning noise, embedding noise and detectors' average accuracy.

Halftoning	G	n_H	n_E	n_E/n_H	Accuracy (%)
Ordered dither	Average filter	233	14	0.06	94.01
	Fixed filter	112	10	0.09	99.14
	Varying filter	110	9	0.08	99.00
	Projection	113	109	0.96	99.96
ED (Javis)	Average filter	176	50	0.28	80.74
	Fixed filter	35	29	0.83	89.65
	Varying filter	31	24	0.77	87.30
	Projection	30	30	1.00	93.71
ED (Floyd and Steinberg)	Average filter	132	38	0.29	84.04
	Fixed filter	46	33	0.71	91.00
	Varying filter	54	21	0.38	89.06
	Projection	61	67	1.10	97.40

applied to reconstruct the images and the detection accuracy of the detectors based on different inverse halftoning algorithms are compared. The experimental results show that the projection based inverse halftoning algorithm exaggerates the embedding noise and provides better detection results than the filtered based inverse halftoning algorithm. Also, the performance of the detectors for halftone images generated by different halftoning methods are different. Detection of data hidden in ordered dithering images is easier than those hidden in error diffusion images and the detection of data hidden in Javis kernel based error diffusion images is more difficult than that in Steinberg kernel based error diffusion images. Our experimental results show that maximizing n_E/n_H is helpful in improving the accuracy of the detectors, which can be used to search for a better inverse halftoning algorithm to further improve the accuracy of the detectors. Including more features such as the recent work in [79] may further improve the detectors' performance.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

This thesis deals with the problem of evaluating the distortion in binary text and binary graphic images and detecting hidden message in most forms of binary images. A distortion measure and several steganalysis schemes are proposed.

Based on the property of the identical marks in computer generated clean text-font images, a universal steganalysis technique is proposed to defeat pixel flipping in clean text-font image. It can detect the existence of a secret message more reliably than earlier work. The special property of clean text-font image makes the detection of hidden data in clean text-font image easier, which implies that clean text-font image is not a good choice to hide data.

Compared with steganalysis of clean text-font images, steganalysis of noisy text images is much more difficult as noisy scanned images usually do not have

such a property of identical marks as in clean text-font image. However, similar marks are still correlated and the embedding of the hidden message may reduce the correlation. Based on this observation, a universal detection scheme is developed by analyzing the COL pixels. An image dependent threshold is adopted to classify stego image from original image and its superiority over fixed threshold is shown from the ROC curve. The results show that the detection scheme works well when the embedding rate is high for most of the schemes.

Inspired by the experience in evaluating the distortion of the distorted image compared with the original image, a novel way to calculate the distortion by comparing the edge line segments from the original and distorted image is proposed for binary text and binary graphic images. Subjective tests show that the proposed distortion measure is strongly correlated with human visual perception.

A steganalysis technique for binary cartoon image is also proposed based on the fact that the edge line of cartoon image is very often a stair case pattern where the locations of the neighboring corners are highly correlated. Two denoising processes are proposed to remove different types of noise in the image for steganalysis. The morphological smoothing process is proposed to fill holes and remove protrusions created by data hiding. De-noising along stair case pattern process is proposed to remove noise among pixels such as COL pixels. Distortion measures are used in the feature extraction for the detection. Experimental results show that we can combine different distortion measures in the steganalysis to get better classification results. From the results, we can see that detection of hidden data by the embedding algorithm that creates holes or protrusions is much easier than the detection of hidden data by flipping COL pixels. It implies

that the selection of flippable pixels should take smoothness and connectivity into consideration. The de-noising along stair case pattern process implies that the selection of flippable pixels should also take the location correlation among close corners into consideration.

Inspired by the fact that halftone image is an output of a halftoning process originated from natural gray level image, a general method for steganalysis of halftone image by applying inverse halftoning process to reconstruct the gray level image is proposed. Different inverse halftoning processes including filtering based and projection based inverse halftoning processes have been used to obtain the reconstructed gray image for classification. The application of these inverse halftoning processes in steganalysis is investigated and the performance by using different inverse halftoning is compared. The experimental results show that the projection based inverse halftoning can capture more evidence of hidden data than the filtering based inverse halftoning.

6.2 Recommendations for future research work

This thesis has proposed several steganalysis techniques for binary images and an objective distortion measure for all major types of binary text images and binary graphic images. The research and development in this area have been and will remain as one of the most important branches for stenography and steganalysis. So far there are still many unsolved problems in this research area which need further investigation. More work should be done to develop better structures or algorithms to deal with the steganography and steganalysis of binary images. Based on the present work, future research works are

recommended as below.

- The proposed distortion measure in this thesis are mainly based on evaluating the distortion by flipping individual selected pixels. It is recognized that modifying several neighboring pixels while keeping the shape of the contours may not necessary introduce a larger distortion than flipping a single pixel along the straight line. Thus, how to evaluate the distortion when a combination of several pixels are used could be interesting. We may need to apply this to [101] to reduce the visibility of data hiding.

Distortions are important for both the steganalysis and the data hiding. As we know, distortions will be introduced into a image when we hide information in it. At the same time, these distortions can be utilized for the steganalysis, as we can see from steganalysis work in Chapter 4. In fact, visual examination is a direct way of steganalysis. The more distortions we introduce, the more suspicious of the hidden message and vice versa. The word distortions refer to not only the visual distortion, but also the damage on other features such as certain regularities of the images. Very often, we find these regularities are related with visual distortion. The less damage on the regularities, the less likely we observe visible distortion. Thus, to design the next generation data hiding schemes, we may need to consider not only the visual distortion, but also the damages on the regularities of the images. Thus, it would be beneficial to think of the possible regularities of the images and try not to change these regularities during the data hiding.

- We focus on detection of hidden message in text, graphic, and halftone images in Chapter 3, Chapter 4 and Chapter 5 respectively. Except for clean text-font image, estimation of the message length, i.e., quantitative steganalysis,

has not been further investigated. How to estimate the message length would be interesting. In the steganalysis of graphic and halftone images, we use a high-dimensional feature in the classification. The relationship between the message length and the high-dimensional feature is not straightforward. Future work could be focused on looking for one-dimensional feature which is highly correlated with the message length. For scanned text image, the feature being used is one-dimensional, however, its correlation with message length is not strong enough to provide a good estimation of the message length. Thus, a more robust feature is needed. Also, a more robust feature provides better classification results in the detection.

- Besides the scanned text images, scanned graphic images and halftone images can be used to hide information. Our detections schemes in Chapter 4 and Chapter 5 would fail to detect messages hidden in these images as the detector would mistake the scan noise as the message. Also, there is no mark similarity within the image and our detection scheme in Chapter 3 would fail too. How to differentiate the hidden message from scan noise would be an interesting problem.

- With the development of the steganalysis techniques for binary images, we may utilize these techniques to design more secure data hiding scheme to prevent being detected. There is little doubt that even with the development of a suite of detection techniques, more sophisticated steganographic techniques will emerge, which in turn will lead to the development of more detection tools, and so on, thus making the steganography increasingly more difficult.

Author's Publications

Journal

1. J. Cheng and A. C. Kot, "Objective distortion measure for binary text image based on edge line segment similarity," *IEEE Trans. on Image Processing*, vol. 16, pp. 1691-1695, June 2007.
2. J. Cheng and A. C. Kot, "Steganalysis of halftone images using inverse halftoning," (under review).
3. J. Cheng and A. C. Kot, "Steganalysis of binary graphic images using distortion measures," (in preparation).

Conference

1. J. Cheng, A. C. Kot and Susanto Rahardja, "Steganalysis of binary cartoon images using distortion measure," in *Proc. of IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. II, pp. 261-264, Honolulu, Hawaii, USA, April 2007.
2. J. Cheng, A. C. Kot, J. Liu and H. Cao, "Detection of data hiding in binary text images," in *Proc. of IEEE Int. Conf. Image Processing*, vol. III, pp. 73-76, Genoa, Italy, Sep. 2005.
3. J. Cheng, A. C. Kot, J. Liu and H. Cao, "Steganalysis of binary text images," in *Proc. of IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. IV, pp. 689-692, Philadelphia, PA, USA, March 2005.
4. J. Cheng, A. C. Kot, J. Liu and H. Cao, "Steganalysis of data hiding in binary text images," in *Proc. of IEEE International Symposium on Circuits and Systems*, pp. 4405-4408, Kobe, Japan, May 2005.

-
5. J. Cheng and A. C. Kot, "Objective distortion measure for binary images," in *Proc. of IEEE TENCON*, pp. 355-358, Chiang Mai, Thailand, Nov. 2004.
 6. H. Lu, A. C. Kot. and J. Cheng, "Secure data hiding in binary document images for authentication," in *Proc. Of IEEE Int. Symp. Circuits and Systems*, vol.3, pp. 806-809, Bangkok, Thailand, May 2003.

Bibliography

- [1] http://vision.ece.ucsb.edu/~sullivak/Research_img/.
- [2] A large binary image data base. <http://www.lems.brown.edu/dmc/>.
- [3] R. J. Anderson and F. A. P. Petitcolas. On the limites of steganography. *IEEE Journal on Selected Areas in Communications*, 16:474–481, 1998.
- [4] R. J. Anderson(eds). *Information Hiding: first international workshop, Lecture notes in computer science*. Springer Verlag, 1996.
- [5] D. Artz. Digital steganography:hiding data within data. *IEEE Internet Computing*, 5:75–80, 2001.
- [6] I. Avcibas, N. Memon, and B. Sankur. Steganalysis using image quality metrics. *IEEE Trans. on Image Processing*, 12:221–229, 2003.
- [7] B. E. Bayer. An optimum method for two level rendition of continuous tone pictures. in *Proc. of IEEE. Int. Conf. Communication*, pages 2611–2615, 1973.
- [8] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. Techniques for data hiding. *IBM Systems Journal*, 35:313–336, 1996.
- [9] A. K. Bhattacharjya and H. Ancin. Data embedding in text for a copier system. In *Proc. IEEE. Int. Conf. Image Processing*, volume 2, pages 245–249, 1999.
- [10] J. T. Brassil, S. H. Low, and N. F. Maxemchuk. Copyright protection for the edectornic distribuion of text documents. *Proce. of the IEEE*, 87(7):1181–1196, 1999.
- [11] J. T. Brassil, S. H. Low, N. F. maxemchuk, and L. O’Gorman. Marking text features of document images to deter illicit dissemination. In *Proc. IEEE Int. Conf. Pattern Recognition*, volume 2, pages 315–319, 1994.

- [12] J. T. Brassil and L. O’Gorman. Watermarking document images with bounding box expansion. In *Proc. First Int. Workshop on Information Hiding*, volume 1174, pages 227–235, 1996.
- [13] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. In *Data Mining and Knowledge Discovery Archive*, volume 2, pages 121–167, 1998.
- [14] C. A. Cabrelli and U. M. Molter. Automatic representation of binary images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12:1190–1196, 1990.
- [15] N. Chaddha and T. H. Y. Meng. Psycho-visual based distortion measures for monochrome image and video compression. In *The 27th Asilomar Conf. Signal, Systems and Computers*, pages 841–845, 1993.
- [16] R. Chandramouli. A mathematical approach to steganalysis. In *Proc. SPIE Security and Watermarking of Multimedia Contents IV*, volume 4675, page 14C25, 2002.
- [17] R. Chandramouli. A mathematical framework for active steganalysis. In *ACM/Springer Multimedia Systems Special Issue on Multimedia Watermarking*, pages 303–311, 2003.
- [18] R. Chandramouli and N. Memon. Steganography capacity: A steganalysis perspective. In *Proc. SPIE Security and Watermarking of Multimedia Contents*, volume 5020, pages 173–177, 2003.
- [19] C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines. 2001. <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [20] L. M. Chen and H.M. Hang. An adaptive inverse halftoning algorithm. *IEEE Trans. Image Processing*, 6:1202–1209, 1997.
- [21] Y. Y. Chen, H. K. Pan, and Y. C. Tseng. A secure data hiding scheme for two-color images. In *Proc. IEEE Int. Symp. Computers and Communications*, pages 750–755, 2000.
- [22] J. Cheng and A. C. Kot. Objective distortion measure for binary images. In *Proc. IEEE Tencon*, volume 1, pages 355–358, 2004.
- [23] J. Cheng, A. C. Kot, J. Liu, and H. Cao. Steganalysis of binary text images. In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, volume 4, pages 689–692, 2005.

- [24] N. Chotikakamthorn. Electronic document data hiding technique using inter-character space. In *Proc. of IEEE Asia-Pacific Conf. on Circuits and Systems*, pages 419–422, 1998.
- [25] N. Chotikakamthorn. Document image data hiding technique using character spacing width sequence coding. In *Proc. IEEE Int. Conf. Image Processing*, volume 2, pages 250–254, 1999.
- [26] E. Cole. *Hiding in plain sight*. John W. Wiley, 2003.
- [27] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.
- [28] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoan. Secure spread spectrum watermarking for multimedia. *IEEE Trans. Image Processing*, 6:1673 – 1687, 1997.
- [29] O. Dabeer, K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath. Detection of hiding in the least significant bit. *IEEE Trans. on Signal Processing*, 10:3046–3058, 2004.
- [30] S. Dumitrescu and X. Wu. Steganalysis of lsb embedding in multimedia signals. In *Proc. IEEE Int. Conf. Multimedia and Expo*, volume 1, pages 581–584, 2002.
- [31] S. Dumitrescu, X. Wu, and Z. Wang. Detection of lsb steganography via sampling pairs. *IEEE Trans. Signal Processing*, 51:1995–2007, 2003.
- [32] H. Farid. *Detecting steganographic messages in digital images*. Technical Report:TR2001-412, Dartmouth college, 2001.
- [33] H. Farid. Detecting hidden messages using higher order statistical models. In *Proc. IEEE Int. Conf. Image Processing*, volume 2, pages 905–908, 2002.
- [34] R.W. Floyd and L. Steinberg. An adaptive algorithm for spatial grayscale. in *Proc. SID*, pages 75–77, 1976.
- [35] J. Fridrich, R. Du, and M. Long. Steganalysis of lsb encoding in color images. In *Proc. IEEE Int. Conf. Multimedia and Expo*, volume 3, pages 1279–1282, 2000.
- [36] J. Fridrich and M. Goljan. Practical steganalysis-state of the art. In *Proc. SPIE Photonics Imaging, Security and Watermarking of Multimedia Contents*, volume 4675, pages 1–13, 2002.

- [37] J. Fridrich, M. Goljan, and R. Du. Detecting lsb steganography in color and gray scale images. *Magazine of IEEE Multimedia, Special Issue on Security*, 8:22–28, 2001.
- [38] J. Fridrich, M. Goljan, and D. Hoge. Attacking the outguess. In *Proc. ACM Workshop Multimedia and Security*, pages 3–6, 2002.
- [39] J. Fridrich, M. Goljan, D. Hoge, and D. Soukal. Quantitative steganalysis of digital images: Estimating the secret message length. *ACM Multimedia Systems Journal*, 9:288–302, 2003.
- [40] J. Fridrich, M. Goljan, and D. Soukal. Wet paper codes with improved embedding efficiency. *IEEE Trans. on Information Forensics and Security*, 1:102–110, 2006.
- [41] M. S. Fu and O.C. Au. Data hiding by smart pair toggling for halftone images. in *Proc. of IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 4:2318–2321, 2000.
- [42] M. S. Fu and O.C. Au. Data hiding for halftone images. in *Proc. of SPIE Conf. On Security and Watermarking of Multimedia Contents II*, 3971:228–236, 2000.
- [43] M. S. Fu and O.C. Au. Improved halftone image data hiding with intensity selection. in *Proc. IEEE Int. Conf. Circuits System*, 5:243–246, 2001.
- [44] M. S. Fu and O.C. Au. Data hiding watermarking for halftone images. *IEEE Trans. Image Processing*, 11:477–484, 2002.
- [45] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Publishing Company, Inc., 1992.
- [46] H. Gou and M. Wu. Data hiding in curves with applications to fingerprinting maps. *IEEE Trans. on Signal Processing, Special Issue on Secure Media*, 53:3988–4005, 2005.
- [47] H. Gou and M. Wu. Improving embedding payload in binary images with ‘super-pixles’. In *Proc. of IEEE Int. Conf. on Image Processing*, volume 3, pages 277–280, 2007.
- [48] S. Hetzl. A survey of steganography. <http://steghide.sourceforge.net>.
- [49] C. Hosmer and G. Gordon. Steganography detection and recovery toolkit. Technical report, WetStone Technology Inc. Air Force Research Laboratory, Mar. 2000.

- [50] P. G. Howard. Text image compression using soft pattern matching. *The Computer Journal*, 40:146–156, 1997.
- [51] P. G. Howard, F. Kossetini, B. Martins, S. Forchhammer, and W.J. Rucklidge. The emerging JBIG2 standard. *IEEE Trans on Circuit and Systems for Video Technology*, 8:838–848, 1998.
- [52] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.
- [53] M. Jiang, E.K. Wong, N. Memon, and X. Wu. Steganalysis of halftone images. in *Proc. of IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2:793–796, 2005.
- [54] M. Jiang, X. Wu, E. K. Wong, and N. Memon. Quantitative steganalysis of binary images. In *Proc. IEEE Int. Conf. Image Processing*, volume 1, pages 29–32, 2004.
- [55] M. Jiang, X. Wu, E. K. Wong, and N. Memon. Steganalysis of boundary based steganography using autoregressive model of digital boundaries. In *Proc. IEEE Int. Conf. Multimedia and Expo*, volume 2, pages 883–886, 2004.
- [56] N. F. Johnson. *Information hiding : steganography and watermarking : attacks and countermeasures*. Kluwer Academic, 2001.
- [57] N. F. Johnson and S. Jajodia. Steganalysis: The investigation of hidden information. In *Proc. of IEEE Information Technology Conference*, 1988.
- [58] D. Kahn. The history of steganography. In *Proc. of Information Hiding, First Int. Workshop*, volume 1174, 1996.
- [59] T. Kanungo, R. M. Haralick, and W. Stuezel H. Barid, and D. Madigan. A statistical, nonparametric methodology for document degradation model validation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:1209–1223, 2000.
- [60] T. Kanungo and Q. Zheng. Estimating degradation model parameters using neighborhood pattern distributions: an optimization approach. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26:520–524, 2004.
- [61] S. A. Karunasekera and N. G. Kingsbury. A distortion measure for blocking artifacts in images based on human visual sensitivity. *IEEE Trans. on Image Processing*, 4(6):713–724, 1995.

- [62] S. Katzenbeisser. *Information Hiding Techniques for Steganography and Digital watermarking*. Artech House, 2000.
- [63] J. Kelley. Terror groups hide behind web encryption. *USA Today*, Feb. 2001.
- [64] J. Kelley. Militants wire web with links to jihad. *USA Today*, Oct. 2002.
- [65] H. Y. Kim. A new public-key authentication watermarking for binary document images resistant to parity attacks. In *Proc. IEEE Int. Conf. Image Processing*, volume 2, pages 1074–1077, 2005.
- [66] H. Y. Kim and A. Afif. Secure authentication watermarking for binary images. in *Proc. of XVI Brazilian Symp. on Computer Graphics and Image Processing*, pages 199–206, 2003.
- [67] H. Y. Kim and R. L. Queiroz. A public-key authentication watermarking for binary images. In *Proc. IEEE Int. Conf. Image Processing*, volume 2, pages 3459–3462, 2004.
- [68] K. M. Kim, C. S. Lee, E. J. Lee, and Y. H. Ha. Color image quantization using weighted distortion measure of hvs color activity. In *Proc. IEEE Inf. Conf. Image Processing*, pages 1035–1039, 1996.
- [69] M. Kwan. Gifshuffle. <http://www.darksized.com.au/gifshuffle>.
- [70] T. H. Liu and L. W. Wang. An adaptive data hiding technique for binary images. In *Proc. Int. Conf. Pattern Recognition*, volume 4, pages 831–833, 2004.
- [71] Y. Liu, J. Mant, E. Wong, and S. H. Low. Marking and detection of text documents using transform-domain techniques. In *Proc. SPIE Security and Watermarking of Multimedia Contents*, volume 3657, pages 317–327, 1999.
- [72] S. H. Low and N. F. Maxemchuk. Capacity of text marking channel. *IEEE Signal Processing Letters*, pages 345–347, 2000.
- [73] S. H. Low, N. F. Maxemchuk, J. T. Brassil, and L. O’Gorman. Document marking and identification using both line and word shifting. In *Proc. Infocom*, volume 2, pages 853–860, 1995.
- [74] H. Lu, A. C. Kot, and J. Cheng. Secure data hiding in binary document images for authentication. In *Proc. IEEE Int. Symp. Circuits and Systems*, pages 806–809, 2003.

- [75] H. Lu, A. C. Kot, and Y. Q. Shi. Distance-reciprocal distortion measure for binary document images. *IEEE Signal Processing Letters*, 11:228–231, 2004.
- [76] H. Lu, J. Wang, A. C. Kot, and Y. Q. Shi. An objective distortion measure for binary document images based on human visual perception. In *Proc. IEEE Int. Conf. Pattern Recognition*, volume 4, pages 239–242, 2002.
- [77] S. Lyu and H. Farid. Detecting hidden messages using higher order statistics and support vector machines. In *Fifth Int. Workshop on Information Hiding*, volume 2578, pages 340–354, 2002.
- [78] S. Lyu and H. Farid. Steganalysis using color wavelet statistics and one class support vector machines. In *SPIE Symp. on Electronic Imaging*, volume 5306, pages 35–45, 2004.
- [79] S. Lyu and H. Farid. Steganalysis using higher order image statistics. *IEEE Trans. on Information Forensics and Security*, 1:111–119, 2005.
- [80] R. Machado. Ezsteg. 2001. <http://www.stego.com>.
- [81] K. Matsui and K. Tanaka. Video-steganography: how to secretly embed a signature in a picture. *Proc. IMA Intellectual Property Project*, 1:187–205, 1994.
- [82] N. F. Maxemchuk and S. H. Low. Marking text document. In *Proc. IEEE Int. Conf. Image Processing*, volume 3, pages 13–16, 1997.
- [83] Q. Mei, E. K. Wong, and N. Memon. Data hiding in binary text documents. In *Proc. SPIE Security and Watermarking of Multimedia Contents III*, volume 4314, pages 369–375, 2001.
- [84] C. Moroney. Hide and seek. <http://www.rugeley.demon.co.uk/security/hdsk50.zip>.
- [85] J. O’Ruanaidh and T. Pun. Rotation, scale and translation invariant spread spectrum digital image watermarking. *Signal Processing*, 66:303–317, 1998.
- [86] W. Osberger, A. J. Maeder, and D. Mclean. An objective quality assessment technique for digital image sequences. In *Proc. IEEE Int. Conf. Image Processing*, volume I, pages 897–900, 1996.

- [87] H. Ozer, I. Avciabas, B. Sankur, and N. Memon. Steganalysis of audio based on audio quality metrics. In *Proc. SPIE*, volume 5020, pages 55–66, 2003.
- [88] G. Pan, Y. Wu, and Z. Wu. A novel data hiding method for two-color images. In *Lecture notes in Computer Science*, volume 2229, pages 261–270, 2001.
- [89] F. Petitcolas, R. Anderson, and M. Kuhn. Information hiding-a survey. In *Proc. of IEEE*, volume 87, pages 1062–1078, 1999.
- [90] N. Provos. Defending against statistical steganalysis. In *Proc. 10th USENIX Security Symp.*, pages 323–335, 2001.
- [91] N. Provos and P. Honeyman. Detecting steanographic content on the internet. Technical report, CITI, University of Michigan, 2001.
- [92] P. Pudil, J. Novovicova, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, 1994.
- [93] N. B. Puhan and A. T. S. Ho. Binary document image watermarking for secure authentication using perceptual model. In *Proc. IEEE Int. Symp. Signal Processing and Information Technology*, pages 393–398, 2005.
- [94] A. C. Rencher. *Methods of Multivariate Analysis*. J. Wiley, 2nd edition, 2002.
- [95] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osorne. A digital watermark. In *Proc. IEEE Int. Conf. Image Processing*, volume 2, pages 86–90, 1994.
- [96] Y. Q. Shi and H. Sun. *Image and Video Compression for Multimedia Engineering: Fundamental, Algorithm and Standards*. CRC Press LLC, 1999.
- [97] G. J. Simmons. Prisoners problem and subliminal channel. In *Advance in Cryptology: Proc. of CRYPTO83*, pages 51–67, 1984.
- [98] P. C. Teo and D. J. Heeger. Perceptual image distortion. In *Proc. IEEE Int. Conf. Image Processing*, pages 982–986, 1994.
- [99] W. Trappe, M. Wu, Z. J. Wang, and K. J. R. Liu. Anti-collusion fingerprinting for multimedia. *IEEE Tran. on Singal Processing*, 51:1069–1087, 2003.

- [100] S. Trivedi and R. Chandramouli. Secret key estimation in sequential steganography. *IEEE Trans. on Signal Processing*, 53:746–757, 2005.
- [101] Y. C. Tseng and H. K. Pan. Data hiding in 2-color images. *IEEE Trans. on Computers*, 51:873–878, 2002.
- [102] R. Ulichney. *Digital halftoning*. Cambridge MA: MIT Press, 1987.
- [103] D. Upham. Jpeg-jsteg. <ftp://ftp.funet.fi/pub/crypt/steganography/jpeg-jstegv4.diff.gz>.
- [104] H. Wang. Data hiding techniques for printed binary images. In *Proc. IEEE Int. Conf. Information Technology: Coding and Computing*, pages 55–59, 2001.
- [105] A. A. Webster, C. T. Jones, M. H. Pinson, S. D. Voran, and S. Wolf. An objective video quality assessment system based on human perception. In *Proc. SPIE Human Vision Visual Processing and Digital Display IV*, pages 15–26, 1993.
- [106] A. Westfeld. F5-a steganographic algorithm: high capacity despite better steganalysis. In *Lecture Notes in Computer Science*, volume 2137, pages 289–302, 2001.
- [107] A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. In *Proc. Information Hiding, Third Int. Workshop*, volume 1768, pages 61–69, 1999.
- [108] P. W. Wong. Inverse halftoning and kernel estimation for error diffusion. *IEEE Trans. Image Processing*, 4:486–498, 1995.
- [109] P. W. Wong. Multi-resolution binary image embedding. in *Proc. of SPIE Security and Watermarking of Multimedia Contents*, 5020:423–429, 2003.
- [110] P. W. Wong and N. Memon. Secret and public key image watermarking schemes for image authentication and ownership verification. *IEEE Trans. Image Processing*, 10:1593–1601, 2001.
- [111] M. Wu, J. Fridrich, M. Goljan, and H. Gou. Handling uneven embedding capacity in binary images: A revisit. In *Proc. of SPIE Conf. on Security, Watermarking and Steganography*, volume 5681, pages 194–205, 2005.
- [112] M. Wu and B. Liu. Data hiding in binary image for authentication and annotation. *IEEE Trans. on Multimedia*, 6:528–538, 2004.

- [113] M. Wu, E. Tang, and B. Liu. Data hiding in digital binary image. In *Proc. IEEE Int. Conf. Multimedia and Expo*, pages 393–396, July 2000.
- [114] M. Wu, W. Trappe, Z. J. Wang, and K. J. R. Liu. Collusion-resistant fingerprinting for multimedia. *IEEE Signal Processing Magazine*, 21:15 – 27, 2004.
- [115] H. Yang and A. C. Kot. Data hiding for bi-level documents using smoothing techniques. In *Proc. IEEE Int. Symp. Circuits and Systems*, volume 5, pages 692–695, 2004.
- [116] H. Yang and A. C. Kot. Text document authentication by integrating inter character and word spaces watermarking. In *Proc. IEEE Int. Conf. Multimedia and Expo*, volume 2, pages 955–958, 2004.
- [117] H. Yang and A. C. Kot. Data hiding for text document image authentication by connectivity-preserving. In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, volume 2, pages 505–508, 2005.
- [118] H. Yang and A. C. Kot. Two-layer binary image authentication with tampering localization. In *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, volume 2, pages 309–312, 2006.
- [119] H. Yang and A. C. Kot. Pattern-based data hiding for binary image authentication by connectivity-preserving. *IEEE Trans. on Multimedia*, 9:475–486, 2007.
- [120] H. Yang, A. C. Kot, and J. Liu. Semi-fragile watermarking for text document images authentication. In *Proc. IEEE Int. Symp. Circuits and Systems*, volume 4, pages 4002–4005, 2005.
- [121] X. Yu, Y. Wang, and T. Tan. Steganalysis of data hiding in binary images. In *Proc. IEEE Int. Conf. Pattern Recognition*, volume 4, pages 877–880, 2004.
- [122] D. Zou, Y.Q. Shi, W. Su, and G. Xuan. Steganalysis based on markov model of thresholded prediction-error image. In *Proc. IEEE Int. Conf. Multimedia and Expo*, pages 1365–1368, 2006.