MULTIMEDIA DATA HIDING

Min Wu

A DISSERTATION
PRESENTED TO THE FACULTY
OF PRINCETON UNIVERSITY
IN CANDIDACY FOR THE DEGREE
OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE
BY THE DEPARTMENT OF
ELECTRICAL ENGINEERING

June 2001

To Mom and Dad

Abstract

The digital information revolution has brought about profound changes in our society and our lives. The many advantages of digital information have also generated new challenges and new opportunities for innovation. This thesis discusses the issues regarding multimedia data hiding and its application to multimedia security and communication, addressing both theoretical and practical aspects, and tackling both design and attack problems.

In the fundamental part, we identify a few key elements of data hiding through a layered structure. Data hiding is modeled as a communication problem where the embedded data is the signal to be transmitted. Various embedding mechanisms target different robustness-capacity tradeoffs. We study this tradeoff for two major categories of embedding mechanisms. In addition, we have found that the unevenly distributed embedding capacity brings difficulty in data hiding. We propose a comprehensive solution to this problem, addressing the considerations for choosing constant or variable embedding rate and enhancing the performance for each case.

In the design part, we present new data hiding algorithms for binary images, grayscale and color images, and videos, covering such applications as annotation, tamper detection, copy/access control, fingerprinting, and ownership protection. The designs provide concrete examples regarding the choice of embedding mechanisms, the selection of modulation/multiplexing technique(s) for hiding multiple bits, and the handling of uneven embedding capacity. Data hiding can also be used in video communication to convey side information for additional functionalities or better performance. This is demonstrated by the novel approaches for real-time transcoding and error concealment.

Because many data hiding applications are in a competitive environment where an adversary has an incentive to obliterate the embedded data, testing the systems' robustness and security via attacks is important. In the attack part, we discuss a number of attacks and countermeasures for data hiding systems. Our investigation begins with three specific types of watermarking schemes, in which full knowledge of the watermarking algorithms is available. We then study the attack problems for digital music under a unique competitive environment, in which the watermarking algorithms are unknown to analysts.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Bede Liu, for his guidance, nurturing, encouragement, and support in every stage of my graduate study. His knowledge, kindness, patience, openmindedness, and vision have provided me with lifetime benefits.

I am grateful to Profs. Peter Ramadge and Sanjeev Kulkarni in my thesis committee for their valuable comments and suggestions on the thesis drafts, as well as for the experience either as a TA or as a student with these two outstanding teachers. I would also like to thank all faculty members of the ISS and CE groups in Princeton's Electrical Engineering Department for their academic guidance and encouragement. Many course works taught by them have provided the background and foundations for my thesis research. Special thanks to Profs. S-Y. Kung, Wayne Wolf, Mike Orchard, and Margaret Martonosi for their guidance in several independent projects that have expanded my vision. Also thank CS Profs. Adam Finkelstein, Perry Cook, and Edward Felten for the enlightening discussions. Beyond technical matters, I greatly appreciate the experience with Prof. Ed Zschau in his High-tech Entrepreneurship classes, and with the instructors of Princeton's Japanese Language Program.

I want to take this opportunity to thank Drs. Ingemar Cox, Matt Miller, Jeffery Bloom, and Harold Stone for their support and collaborations both during my internship at NECI and thereafter. The interaction with them have tremendously improved my understanding in data hiding and have nurtured me in the course of becoming a professional researcher. I have also benefited a lot from the interaction with Prof. Nasir Memon and Mr. Shimizu. Their thoughtful comments and encouragement on my research and thesis are highly appreciated. Special thanks to Dr. Wenjun Zeng for the valuable discussions and helps since the very beginning of my research, and to Dr. Heather Yu for the collaborations, encouragement, and friendship. Thank Drs. Bo Tao, Andrew McMellips, and Y-P. Tan for their enlightening discussions in my early research, my officemates Drs. Xiaodong Wang and Chris Yu for sharing their vision in graduate studies, and Dr. Minerva Yeung for her valuable advices regarding research and career.

My memory of graduate studies at Princeton can never be complete without mentioning my fellow graduate students in the Information Sciences and Systems group and the Computer Engineering group. The survivor story would be much tougher if without the accompanying by the students coming to Princeton in the same year as me. Special thanks to Zhifan Sheng for his kindness and countless advices, to Pei Chen for her friendship, and to Rich Radke for the enjoyable discussions in such a wide range. I would also like to express my appreciation to Peng Yin, Rob Joyce, Scott Craver, and Minghui Xia for the enjoyable interactions and collaborations, some of which can be found in this thesis. In addition, I would like to thank Mr. Jay Plett for the tireless computing support, and Ms. Stephanie Costantini, Ms. Karen Williams, and Mr. Thomas Roddenbery for their assistance.

I am so grateful to Xiang Yu, my husband, for his love and encouragement during our years at Princeton. His broad technical knowledge and skills in computer science have provided endless support in my study. I also want to thank my host family at Princeton and my extended family in China for the years' care and support received from them. This thesis draws a period for my 20-year education in schools. In addition to the thanks to my teachers and classmates over the past two decades, I must mention two persons without whose love, nurturing, and support I could never accomplish all these. Thank Mom for her care and encouragement toward the pursuit of excellence. And thank Dad, not only for his patience and sacrifice, but also for being such a great role model for his daughter. I dedicate this thesis to my parents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The digital information revolution has brought about profound changes in our society and our lives. The many advantages of digital information have also generated new challenges and new opportunities for innovation. Along with powerful software, new devices, such as digital camera and camcorder, high quality scanners and printers, digital voice recorder, MP3 player and PDA, have reached consumers worldwide to create, manipulate, and enjoy the multimedia data. Internet and wireless network offer ubiquitous channels to deliver and to exchange information. The security and fair use of the multimedia data, as well as the fast delivery of multimedia content to a variety of end users/devices with guaranteed QoS are important yet challenging topics. The solutions to these problems will not only contribute to our understanding of this fast moving complex technology, but also offer new economic opportunities to be explored. This thesis addresses the issues regarding multimedia data hiding and its applications in multimedia security and communication.

With the ease of editing and perfect reproduction in digital domain, the protection of ownership and the prevention of unauthorized tampering of multimedia data (audio, image, video, and document) become important concerns. Digital watermarking and data hiding, schemes to embed secondary data in digital media, have made considerable progress in recent years and attracted attention from both academia and industry. Techniques have been proposed for a variety of applications, including ownership protection, authentication, access control, and annotation. Data hiding is also found useful as a general tool to send side information in multimedia communication for achieving additional functionalities or enhancing performance. Imperceptibility, robustness against moderate processing such as compression, and the ability to hide many bits are the basic but rather conflicting requirements for many data hiding applications. In addition, a few other important problems encountered in practice, such as the *uneven embedding capacity* for image/video and the perceptual models for binary images, have received little attention in literature. The works included in this thesis intend to contribute toward the understanding of multimedia data hiding, addressing both theoretical and practical aspects, and tackling both design and attack problems. Various issues of data hiding are studied with new principles and techniques being proposed. This introductory chapter first provides a brief overview of recent technologies and progress of data hiding, then outlines the problems addressed in this thesis and summarizes its major original contributions.

1.1 Overview of Multimedia Data Hiding

The ideas of information hiding can be traced back to a few thousand years ago. As surveyed in [29], simply obscuring the content of a messages by encryption is not always adequate in practice. In many rivalry environments, concealing the existence of communication is desirable to avoid suspicion from adversaries. The word "steganography", which originated from Greek and is still in use today, literally means "covered writing". Stories of covert communications have been passed for generations, but they were mainly used by military and intelligence agencies. It is until the recent decade that information hiding began receiving wide attention from research community and information technology industry, with hundreds of publications and dozens of patents coming out in the past few years. Digital information revolution and the thriving progress in network communication are the major driving forces of this change. The perfect reproduction, the ease of editing, and the Internet distribution of digital multimedia data have brought about concerns of copyright infringement, illegal distribution, and unauthorized tampering. Techniques of associating some imperceptible data with multimedia sources via embedding started to come out to alleviate these concerns. Interestingly, while most such techniques embed data imperceptibly to retain the perceptual quality and value of the host multimedia source, many of them were referred as *digital watermarking* whose traditional counterpart is not necessarily imperceptible. The analogy emphasizes on the applications: as a technique in the art of paper making, the paper watermarks usually indicate the origin, the ownership, and/or the integrity of the document printed on the associated pieces of paper, in addition to artistic decorations. As the application domain of embedding data in digital multimedia sources becomes broaden, several terms are used by various groups of researchers, including *steganography*, *digital watermarking*, and *data hiding*. Some explanation and comparison of terminologies related to information hiding were presented in [29, 30]. Rather than playing with the terminologies, this thesis uses the two terms *data hiding* and *digital watermarking* almost interchangeably, referring to embedding secondary data into the primary multimedia sources. The embedded data, usually called *watermark(s)*, can be used for various purposes, each of which is associated with different robustness, security, and embedding capacity requirements. The principal advantage of data hiding versus other solutions is its ability to associate secondary data with the primary media in a seamless way. As we shall see later in this thesis, the seamless association is desirable in many applications. For example, compared with cryptographic encryptions, the embedded watermarks can travel with the host media and assume their protection functions even after decryption. With the only exception of visible watermarks that will be discussed below, the secondary data are expected to be imperceptible.

There are many ways to categorize data hiding techniques. A straightforward classification is according to the type of primary multimedia sources, giving us data hiding systems for perceptual and non-perceptual sources. This thesis is primarily concerned with perceptual multimedia sources, including audio, binary image, color or grayscale image, video, and 3-D graphics. Among digital sources, the major difference between perceptual and non-perceptual data is that the non-perceptual data, like text and executable codes, usually requires lossless processing, transmission and storage. Flipping a single bit may lead to different meaning. Perceptual data, however, has a perceptual tolerance range, which allows minor change before being noticed by human. This perceptual property enables data embedding as well as lossy compression either imperceptibly or with a controllable amount of perceptual degradation. Although many general techniques of data hiding can be applied to audio, image, video, and 3-D graphics [30, 124], there are unique treatments associated with each type of perceptual sources. The main reason is that they are related to particular sense(s) and the way we see things is quite different from the way we hear. Proper perceptual models have to be exploited to ensure the host data is changed in such a way that does not introduce noticeable difference. Dimensionality and causality are another two reasons leading to different treatments. The techniques and resources required for processing 1-D data would be quite different from that for 2-D and for 3-D. Similar argument holds for non-progressive data like image versus progressive data like audio and video. We shall clarify that the perceptual property is not a necessity for hiding data. There are changes that can be made to non-perceptual data while preserving the semantic meaning. For example, a word can be changed to its synonyms, special pattern of spaces and blank lines can be added to computer source code, and jumping instructions in assembly code can be rearranged [125]. These changes can be used to enforce certain relations (either deterministically or statistically) to encode secondary data, as we do for hiding data in perceptual sources. The detailed discussion of data hiding in non-perceptual sources is beyond the scope of this thesis.

In terms of perceptibility, data hiding techniques can be classified into two groups, perceptible and imperceptible hiding. Perceptible watermarks are mainly used in image and video. A visually meaningful pattern, such as a logo, is overlaid on an image or video, which is essentially an image editing or synthesis problem. The visible watermarks explicitly exhibit the copyright, ownership information, or access control policies so as to discourage the misuse of the watermarked images. For example, semi-transparent logos are commonly added to TV programs by broadcasting networks, and to the preview images accessible via World Wide Web by copyright holders. In [122], a visible watermarking technique is proposed by modifying the luminance of the original image according to a binary or ternary watermark pattern. The amount of modification is adaptive to the local luminance to give a consistent perceptual contrast [16]. In addition, the modification is modulated by a random sequence to make it difficult to systematically remove the visible marks via an automated algorithm. Video can be visibly marked using similar ideas [123]. The majority of current data hiding research concerns with imperceptible watermarking. It is also the focus of this thesis. As we mentioned earlier, perceptual models need to be explored to ensure the changes imposed by an embedding system are imperceptible to retain the perceptual quality and value of the multimedia sources.

Application domain is another criterion to categorize data hiding techniques. Classic applications include ownership protection, authentication, fingerprinting, copy / access control, and annotation. We shall briefly explain the design requirement of each application:

- *Ownership Protection*: a watermark indicating ownership is embedded in the multimedia source. The watermark, known only to the copyright holder, is expected to survive common processing and intentional attack so that the owner can show the presence of this watermark in case of dispute to demonstrate his/her ownership. The detection should have as little ambiguity and false alarm as possible. The total embedding capacity, namely, the number of bits that can be embedded and extracted with small probability of error, does not have to be high in most scenarios.

- *Authentication or Tampering Detection*: a set of secondary data is embedded in the multimedia source beforehand, and later is used to determine whether the host media is tampered or not. The robustness against removing the watermark or making it undetectable is not a concern as there is no such incentive from attacker point of view. However, forging a valid authentication watermark in an unauthorized or tampered media source must be prevented. In practical applications, it is also desirable to locate the tampering, and to distinguish some changes (such as the non-content change incurred by moderate lossy compression) from some other changes (such as content tampering). The embedding capacity has to be high in general to accommodate these needs. The detection should be performed without the original unwatermarked copy because either this original is unavailable or its integrity has not been established yet. This kind of detection is usually called *non-coherent detection* or *blind detection*.

- *Fingerprinting or Labeling*: the watermark in this application is used to trace the originator or recipients of a particular copy of multimedia source. For example, different watermarks are embedded in different copies of multimedia sources before distributing to a number of recipients. The robustness against obliterating and the ability to convey a non-trivial number of bits are required.

- *Copy Control & Access Control*: the embedded watermark in this case represents certain copy control or access control policy. A watermark detector is usually integrated in a recording/playback system, like the proposed DVD copy control [121] and the on-going SDMI activities [140]. Upon detection, the policy is enforced by directing certain hardware or software actions such as enabling or disabling a recording module. The robustness against removal, the ability of blind detection, and the capability of conveying a non-trivial number of bits are required.

- *Annotation*: the embedded watermark in this application is expected to convey as many bits as possible without the use of original unmarked copy in detection. While the robustness against intentional attack is not required, a certain degree of robustness against common processing like lossy compression may be desired.

More generally, data hiding is a tool to convey side information while retaining the original appearance. This property is found useful in some multimedia communication scenarios [129] to achieve additional functionalities or better performance.

From a theoretical point of view, data hiding can be considered as a communication problem where the watermark is the signal to be transmitted. Many communication theories and techniques are found very useful in studying data hiding. A fundamental problem along this direction is the total embedding capacity. It is impossible to answer how many bits can be embedded if without specifying the required robustness. This is not hard to understand from the aspects of information theory, where the capacity is tied with a specific channel model and is a function of the channel parameters. Although the classic results of channel capacity in information theory [7], including the capacity theorem in terms of an optimization of mutual information between the channel input and output, the AWGN channel capacity, the capacity of parallel channels, and the zero-error capacity, have been found beneficial toward the understanding of data hiding capacity [41, 89, 90, 91, 92, 93, 94, 95, 96], there are many important differences between data hiding and conventional communication: (1) the noises incurred by processing or intentional attack are diverse and rather complicated to model, and (2) the shape and parameter constraints of watermark signals are determined by human perceptual system, which is far more sophisticated than a simple L_2 model and has not been completely understood yet. These differences limit the direct application of information theoretical results to practical scenarios.

In addition to the total embedding capacity, we notice another fundamental problems associated with data hiding. Due to the non-stationary nature of perceptual sources, the amount of data that can be embedded varies significantly from region to region. This *uneven embedding capacity* adds great difficulty to high-rate embedding. The problem does not receive much attention in literature because a highly suboptimal approach is generally used in practice by embedding a predetermined small number of bits to each region. Although the low constant rate embedding seems work well in experiments involving only a few test sources, where the embedding rate can be tuned toward this small test set, it hardly works in practical systems that need to accommodate much more diverse sources. The simple constant rate embedding not only wastes much embedding capacity in regions that are capable of hiding many bits, but also create dilemma in regions that can hardly embed any bits without introducing noticeable artifacts. Solutions to this problem would substantially improve the performance of many practical systems.

1.2 Thesis Organization and Contributions

This thesis is organized into three parts: *Fundamental Issues* (Part I), *Algorithm and System Designs* (Part II), and *Attacks and Countermeasures* (Part III). We conclude the thesis with final remarks and suggestions of avenues for further study in Chapter 11.

1.2.1 Fundamental Issues and Solutions

We begin our discussion with a general framework and a list of key elements associated with almost all data hiding problems in Chapter 2. A layered view analogous to network communication is presented to show the relations among those key elements. This view point of data hiding motivates the divide-and-conquer strategies for data hiding problem so that the general approaches for each element can be pursued, based on which solutions to specific applications can be systematically found.

In Chapter 3, we consider data hiding as a communication problem where the embedded data is the signal to be transmitted. Different embedding mechanisms target at different robustness-capacity tradeoff. We study this tradeoff for two major categories of embedding mechanisms, including the capacity of simplified channel models and the set-partitioning nature. This study serves as a guideline for selecting an appropriate embedding algorithm given the design requirements of an application, such as the proposed data hiding algorithms for binary images (Chapter 5) and data hiding applications in video communication (Chapter 8). It also serves as a foundation of multi-level data hiding (Chapter 6), a new embedding framework/algorithm with improved performance. In addition, we discuss a number of modulation/multiplexing techniques for embedding multiple bits in multimedia sources. While many data hiding publications use one or several modulation techniques, there is little systematic study and justification regarding how to embed multiple bits given a set of design requirements. Our work compares the advantages and disadvantages of various techniques in a quantitive way. The principles discussed here are used intensively in our algorithm and system designs.

Due to the non-stationary nature of natural multimedia source such as digital image, video and audio, the number of bits that can be embedded varies significantly from segment to segment. This unevenly distributed embedding capacity adds difficulty in data hiding: using constant embedding rate generally wastes embedding capability, and using variable embedding rate requires sending additional side information that sometimes forms an expensive overhead. Unfortunately there are few solutions in literature. In Chapter 4, we address this problem and propose a comprehensive solution. Specifically, when the total number of bits that can be embedded is much larger than the number of bits needed to convey how many bits are embedded, we choose variable embedding rate and hide the side information to facilitate detection without wasting too much embedding capability. When the two bit numbers are comparable, we hide data using constant embedding and shuffling. We will show via analysis and experiments that shuffling is an efficient and effective tool to equalize the uneven embedding capacity. The solutions to uneven embedding capacity problem are applied to many of our designs presented in Part II.

1.2.2 Algorithm and System Designs

In Part II, we present new data hiding algorithms for binary images, grayscale and color images, and videos. For each design, we follow the list of key elements discussed in the fundamental part, explaining how these elements are handled. We shall see concrete examples regarding the choice of embedding mechanism, the selection of modulation/multiplexing technique(s) for hiding multiple bits, and the handling of uneven embedding capacity via techniques like random shuffling.

We begin with designing data hiding algorithms for binary images in Chapter 5. Embedding data in binary images are generally considered difficult because there is little room to make invisible changes. Very few works in literature discuss human visual model for binary images. The few existing data hiding works for binary images are usually only applicable to a specific type of binary image, and the number of bits that can be embedded is limited. We propose a novel algorithm to hide data in a wide variety of binary images, including digitized signature, text document, and drawings. The algorithm consists of several modules, addressing (1) how to identify flippable pixels while maintaining visual quality, (2) how to use flippable pixels to embed data, (3) how to handle uneven embedding capacity from region to region. The proposed algorithm can be used to annotate and authenticate binary images, which is demonstrated by three examples. The embedded data can be extracted not only from a digital copy, but also from a printed hard copy. The conditions and the method to recover the original digital image after printing and scanning are discussed, along with security issues and a few other practical considerations.

We have discussed in the fundamental part the tradeoff between robustness and embedding capacity for a specific embedding mechanism. When designing a data hiding system, considering a single tradeoff setting, which is common in practice, may either overestimate or underestimate the actual distortions. We propose novel multi-level embedding in Chapter 6 to allow the amount of extractable data to be adaptive according to the actual noise condition. When the actual noise condition is weak, many bits can be extracted from a watermarked image; when the noise is strong, a small number of bits can still be extracted with very small probability of error. Analysis is presented to support this idea. A multi-level data hiding system for grayscale and color images is designed with experimental results being presented. The design also uses a refined human visual model that is revised from a work in literature with reduced artifacts and a reasonable amount of computation. We then extend the work to hide a large amount of data in video. The design of hiding data in video shows concrete examples of handling the uneven embedding capacity from region to region within a frame and also from frame to frame. A small amount of side information, so-called control bits, are crucial for handling uneven embedding capacity and for combating frame jitter that may occur during transcoding or intentional attacks. We shall explain how to convey these bits via various modulation/multiplexing techniques.

We mentioned earlier that editing digital multimedia data is much easier than the traditional analog version. In many occasions, determining whether a digital copy has been tampered or not is very important. In Chapter 7, we discuss data hiding algorithms for detecting tampering of grayscale and color images. Many general data hiding techniques, such as the embedding mechanism and shuffling, are used in this specific application. In the mean time, issues that are unique to authentication need to be addressed, including what to authenticate, how to authenticate, and security considerations. We present a framework for watermark-based authentication covering these aspects. Following this framework, we design a specific image authentication system, aiming at signaling and locating tampering as well as at distinguishing non-content changes like moderate lossy compression from content tampering. This distinguishment is important because many digital images and videos are stored in lossy compressed format for efficient storage and transmission, and excessive fragility of an authentication system that is unable to tolerate the change incurred by compression is undesirable. Our design uses a transform domain table look-up embedding mechanism to embed a visually meaningful pattern and a set of content features in pre-quantized DCT coefficients. The detection of tampering utilizes both the semi-fragility of embedding mechanism and the meaning conveyed by the embedded data. This provides adjustability in the degree of distinguishment for content vs. non-content changes, hence is suitable to accommodate a variety of authentication applications. Our experimental results also show that applying shuffling helps to embed more data, enabling better distinguishment between non-content and content changes while preserving visual quality. Extensions to color image and video are discussed at the end of the chapter.

Besides the classic use in ownership protection, authentication, and copy/access control, data hiding serves as a general tool to convey side information. In Chapter 8, we propose novel applications of data hiding in video communication, where the embedded side information helps to achieve additional functionalities or better performance. We start with the problem of real-time transcoding, where a new video bitstream with lower bit rate is generated from an existing one to cope with the bandwidth limitation. The reduction of spatial resolution can significantly reduce the bit rate, but the processing, mostly used for motion estimation and motion compensation, is rather involved. We propose a fast compressed-domain approach to obtain from an MPEG stream a new MPEG stream with half the spatial resolution. The key idea to alleviating the bottleneck of motion estimation and motion compensation is to directly use as much information as possible from the original full size video. Our solution is supported by a novel standard-and-customized decoding framework based on data hiding. That is, the transcoded bit stream still maintains standard compliant appearance and can be decoded by standard decoder with reasonable visual quality; in the mean time, better image quality will be obtained if a customized decoder that can extract the embedded information is available. We present justifications regarding the advantage of data hiding versus other methods for conveying the side information. We then move onto the error concealment problem that is commonly used to compensate the perceptual quality reduction caused by transmission errors. After discussing the connections between error concealment and data hiding and reviewing a few related works, we present an error concealment system that consists of a data hiding module to protect P-frame motion vectors by embedding motion parity bits in the DCT coefficients of I-frames.

1.2.3 Attacks and Countermeasures

Many applications of data hiding, like ownership protection, copy/access control, and authentication, stay in rivalry environment where an adversary has incentives to obliterate the embedded data. Testing the robustness and security of a data hiding system via attacks is as important as the design process and can be viewed as its inseparable element in a broad sense. In Part III, we discuss a number of attacks and countermeasures for data hiding systems, aiming at not only identifying the weaknesses of existing design algorithms and suggesting improvement, but also obtaining a better understanding of what data hiding can do and cannot do for the above mentioned applications.

We begin our study with three specific types of watermarking schemes in Chapter 9, to which analysts have full knowledge of the watermarking algorithms and are able to perform attack experiments without much limitation. The novel block replacement attack in Section 9.1 targets on removing robust watermarks embedded locally in an image. The attack has discovered an important weakness of block-based embedding mechanism that has been neglected in literature. Possible causes of the vulnerability against the proposed attack are analyzed, along with the discussion of countermeasures. In Section 9.2, we shall present a countermeasure against geometric attacks on robust image watermarks, which have been considered as a big challenge. Our novel solution embeds and detects watermarks in a domain that is related to special properties of Fourier transform and is resilient to rotation, scale, and translation. Many important implementation issues are discussed, followed by experimental results on thousands of images. The chapter is concluded with a double capturing attack for forging fragile watermarks in Section 9.3. This attack touches a fundamental aspect of image authentication, i.e., the authenticity is always relative with respect to a reference. Countermeasures of embedding additional data are proposed, aiming at detecting multiple captures or non-natural captures.

Chapter 10 discusses attacks under a unique emulated rivalry environment, in which analysts have no knowledge of the watermarking algorithms. This interesting study and experimental results are based on our participation in the recent public challenge in the form of attacking four audio watermarking technologies organized by the Secure Digital Music Initiative (SDMI). We begin our discussion with the challenge setup, commenting on a few unrealistic aspects that made the challenge much more difficult than a real-world scenario. General approaches for tackling the attack problem are proposed. Following this general framework, we take two successful attacks as examples to demonstrate our attack strategies, to describe the specific implementation, and to present analysis in detail. For completeness, other successful attacks are also briefly explained. While the challenge is designed to test robust watermarks, we notice that an SDMI system may consist of both robust and fragile watermarks. Having found that the fragile watermark serves a purpose of special tamper detection and that its security is important to an SDMI system, we present a potential attack on fragile watermarks and a countermeasure to conclude the chapter.

Part I

Fundamental Issues

Chapter 2

Preliminaries

2.1 Data Hiding Framework

A typical data hiding framework is illustrated in Fig. 2.1. Starting with an original copy of digital media, or *original media* (I_0) in short [1], an embedding module puts in it a set of secondary data to be embedded ($\underline{b}$), which is referred as *embedded data.* The output of the embedding module is perceptually identical to the original but with data hidden in. It is often referred as *marked media* (I_1). The difference between I_1 and I_0 is referred as *embedding distortion*, i.e., the distortion introduced by the embedding process.

In most cases, the hidden data is a collection of bits, which, depending on the application, may come from an encoded character string, from a pattern, from some executable agents, or others. The bit-by-bit accuracy in extracting these hidden data from the marked media is desirable in these cases and is the focus of this thesis. The embedded data may form a perceptual source itself, such as the application of "image in image" and "video in video" [69]. Some moderate decay of the hidden data is tolerable in this case.

[1] I_0 is also commonly referred as *host media* or *cover media.*

The marked media may be subjected to various kinds of processing and attacks before feeding into a detector. The input media to the detector is referred as *test media* (I_2), and the difference between I_2 and I_1 is referred as *noise*. The data extracted from I_2 is referred as *extracted data* ($\hat{\underline{b}}$). In such applications as ownership protection, fingerprinting [2] and access control, accurate decoding of hidden data from distorted test media is preferred. They are commonly referred as *robust data hiding / watermarking*. In other applications such as authentication and annotation, robustness against processing and attacks are not a principal requirement in general. We will discuss the specific design requirement in the later chapters.

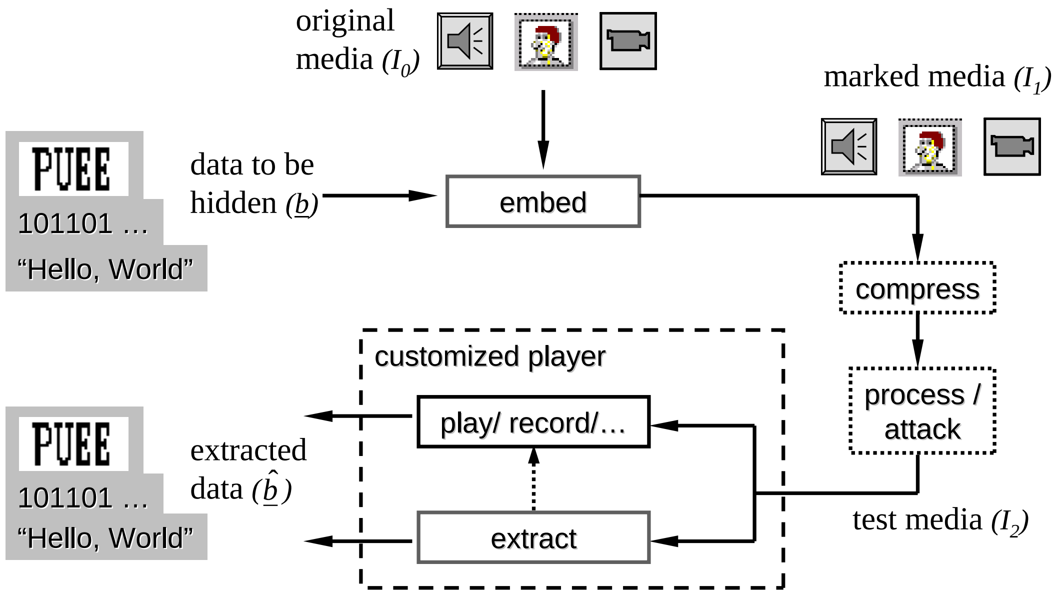


Figure 2.1: General framework of data hiding systems

[2]The *fingerprinting* here refers to the application where different labels are embedded in copies of the same media content before distributing to multiple recipients and the hidden labels are used for tracing each recipient. See also the discussion in Chapter 1.

2.2 Key Elements and A Layered View

Data hiding can be viewed as a communication problem, in which the hidden data is to be delivered and the media host serves as a carrier or as part of the channel. Techniques such as matched filtering, spread spectrum communication, modulation, and error correction coding are widely used in data hiding. In addition, a layered structure helps to prioritize and compartmentalize various design issues.

The key elements in many data hiding systems include:

1. A perceptual model that ensures imperceptibility;
2. How to embed one bit;
3. How to embed multiple bits via modulation/multiplexing techniques;
4. How to embed in those parts of cover media which are difficult to embed, or more generally, how to handle uneven embedding capacity;
5. How to enhance robustness and security;
6. What data to embed.

The elements can be viewed in layers, analogous to the layered structure in network communication (Fig. 2.2). The "physical layer" of data hiding deals with how one or multiple bits are embedded imperceptibly in the original media. This layer has three key elements, namely, (1) the mechanism for embedding one bit, (2) the perceptual model to ensure imperceptibility, and (3) the modulation/multiplexing techniques for hiding multiples bits. Protocols for achieving additional functionalities are built on top of this "physical layer", for example, to handle uneven embedding capacity, to enhance robustness and approach capacity via error correction coding, and to

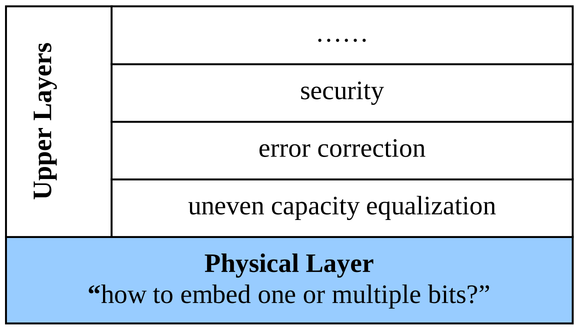

Upper Layers
……
security
error correction
uneven capacity equalization
Physical Layer
"how to embed one or multiple bits?"

Figure 2.2: Layered structure of a data hiding system.

incorporate additional security measures. In the remaining chapters of Part I, we shall use data hiding in images as an example to discuss several elements in detail.

Chapter 3

Classification and Capacity of Embedding Mechanisms

In classic communication, the gap between the theoretical Shannon channel capacity and the limitation in practice are filled by systematic and implementable studies on such issues as modulation, coding/decoding, and equalization [5]. Because data hiding problems have close connection with communication, filling the gap between theoretical embedding capacity and practical limitations is important and is the focus of the fundamental issue part (Part-I) of this thesis. More specifically, we follow the classification described in our work [167] to study two major categories of embedding mechanisms that are practically realizable. We shall begin with simplified models, then gradually loosen the assumptions to consider practical situations. The capacity achievable by the two types of schemes are compared, from which the conditions under which each type of schemes is superior to the other one are identified. We consider the following problems and propose solutions:

- *Distortion during and after embedding*: depending on applications, the allowable change introduced by embedding (a.k.a *embedding distortion*) may be smaller, sometimes, much smaller, than the distortion introduced to the watermarked image (a.k.a *noise*). This is especially the case when the applications preside in a rivalry environment. For example, when watermark is used for ownership protection or access control, the quality of watermarked image/video determines the commercial and artistic value of the digital art works hence the embedding distortion should be well constrained to maintain superb imperceptibility. On the other hand, an adversary who wants to obliterate the watermark may be willing to tolerate some quality degradation. Under this scenario, the noise introduced by an adversary can be significantly larger than the embedding distortion.

- *Actual noise conditions*: an embedding system is generally designed to survive certain noise conditions. This single robustness-capacity tradeoff has limitation in practical applications. First, for applications presiding in a rivalry environment, the actual noise condition may vary dramatically. Second, a watermarked image/video may be compressed or transcoded to different bit rate in order to be delivered through different kinds of communication channels. The desirable amount of information extracted from the image/video could be different depending on the level of compression, as will be explained in Chapter 6. In addition, the information to be embedded usually requires unequal error protection (UEP). Some bits, such as the ownership information and a small amount of control information facilitating the decoding of a larger amount of payload bits, are required to be embedded more robustly than others.

- *Non-stationary property*: due to the non-stationary nature of perceptual sources, the amount of data that can be embedded varies significantly from region to region. This *uneven embedding capacity* adds difficulty to high-rate embedding, especially in practical systems that need to accommodate diverse multimedia content.

In this chapter, we study the robustness-capacity tradeoff for two major categories of embedding mechanisms. The embedding capacity of simplified channel models for these two kinds of embedding are compared. These studies serves as a guideline for selecting an appropriate embedding algorithm given the design requirements of an application, and as a foundation of multi-level data hiding (Chapter 6), a new embedding framework/algorithm with improved performance. We also discuss in this chapter a number of modulation/multiplexing techniques for embedding multiple bits, quantitively comparing the advantages and disadvantages of various techniques. The discussion of the uneven embedding capacity problem will be presented in the next chapter.

3.1 Two Types of Data Embedding

The mechanism for embedding one bit in original media is the most basic element in a data hiding system. Many embedding approaches have been proposed in the literature and there are many ways to classify them. For example, some schemes work with the multimedia signal samples while others work with transformed data. We found it beneficial to study the existing embedding approaches under noise-free conditions (i.e., directly passing a watermarked media to a detector) and to examine whether knowledge of the original host media will enhance the detection performance, regardless of whether a detector uses such knowledge or not [167]. Many existing embedding approaches would then fall in one of the following two categories.

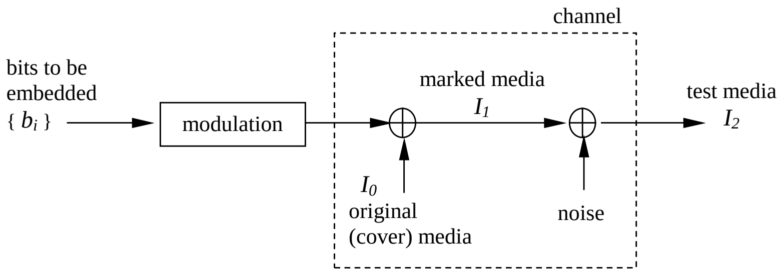


Figure 3.1: Channel model of Type-I embedding.

In the first category (Type-I), the secondary data, possibly encoded, modulated, and/or scaled, is added to the host signal, as illustrated in Fig. 3.1. The addition can be performed in a specific domain or on specific features. Considering the embedding of only one bit, the difference between marked signal I_1 and the original host signal I_0 is a function of b, the bit to be embedded, i.e., $I_1 - I_0 = f(b)$. Although it is possible to detect b directly from I_1 [53], I_0 can be regarded as a major noise source in such detection. Therefore, the knowledge of I_0 will enhance detection performance by eliminating the interference. Additive spread spectrum watermarking is a representative of this category [44, 46].

In the second category (Type-II), the signal space is partitioned into subsets which are mapped by a function $g(\cdot)$ to the set of values taken by the secondary data (e.g., $\{0, 1\}$ for binary hidden data), as illustrated in Fig. 3.2. The marked value I_1 is then chosen from the subset which maps to b, so that the relationship of $b = g(I_1)$ is deterministically enforced. To minimize perceptual distortion, I_1 should be as close to I_0 as possible, where the distance measure is chosen using perceptual models. Unlike the first category, the detector for this type of scheme does not need the knowledge of original value I_0 because the information regarding b is solely carried in I_1.

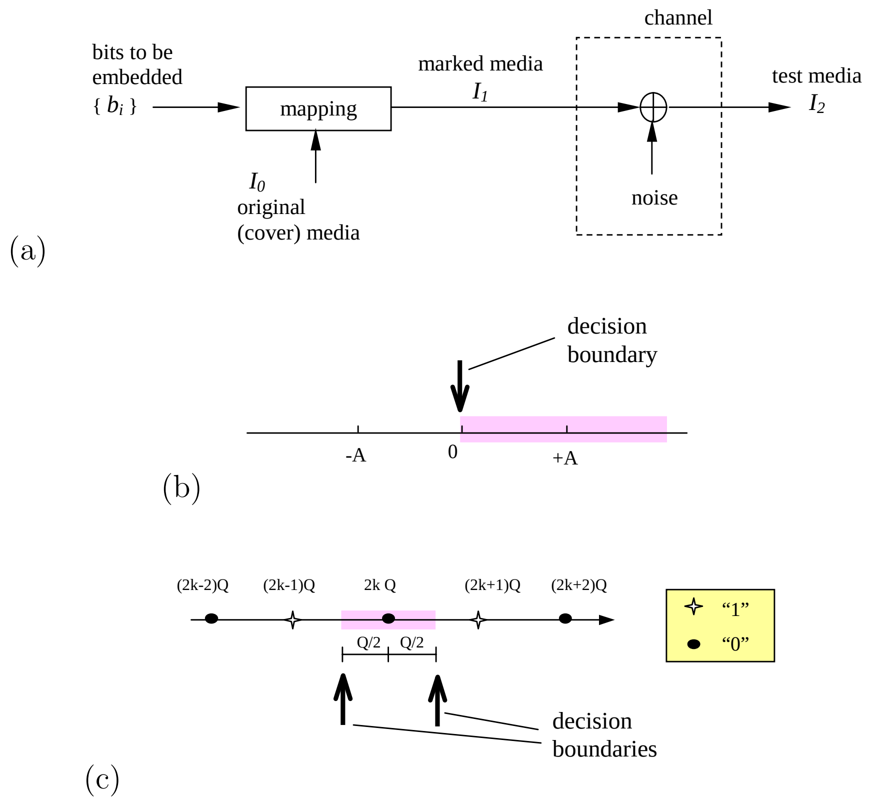


Figure 3.2: Channel model and decision boundaries of Type-II embedding: (a) channel model; (b) single-sided detection decision for sign enforcement with "0" as threshold; (c) detection decision for odd-even enforcement with two boundaries.

A simple example of Type-II is the so called *odd-even embedding*: we choose an even number as I_1 to embed a "0" and an odd number to embed a "1". Data hiding can also be achieved by enforcing a rather global relationship. For example, one may change the sum of several source components to a nearby even number to encode a "0", and to an odd number to encode a "1". This is equivalent to reducing the bits allocated for representing the original vectors and to re-allocate them for conveying side information. By moving from 1-D space to a space of higher dimension, the magnitude of the introduced distortion per dimension is reduced.

Also, there are more choices to select a new signal vector with desired bits embedded in, which allows embedding to be performed in such a way that the human-visual-model-weighted distortion is minimized. On the other hand, the embedding bit rate is reduced, showing a tradeoff between embedding rate and invisibility [1]. The odd-even embedding can be viewed as a special case of the table-lookup embedding [78, 163], which uses a lookup table to determine the mapping between the possible values of a media component and the data to be embedded. There are many other possible ways to partition the space and to enforce a desired relationship. One can enforce the ordering of a pair of samples or coefficients v_1 and v_2. For example, we generate marked coefficients v'_1 and v'_2 close to v_1 and v_2 such that $v'_1 > v'_2$ to embed a "1" and $v'_1 \leq v'_2$ to embed a "0" [63]. One can also enforce signs to embed a "1" or "0", as used in [64, 65]. Extending the basic ways of enforcement, more sophisticated schemes can be designed and/or analyzed [96]. Many proposed schemes in the literature that claimed to have the ability of non-coherent detection [2] belong to this category. It is the deterministically enforced relationship on I_1 that removes the need of using original signal I_0. For the convenience of discussion, we shall refer the collection of image pixels or coefficients on which the relation is enforced as an *embedding unit.* If the enforcement is performed on a quantity derived from the embedding unit (e.g., the sum of a few coefficients, the signs of a coefficient, etc.), we shall refer the quantity as a *feature.*

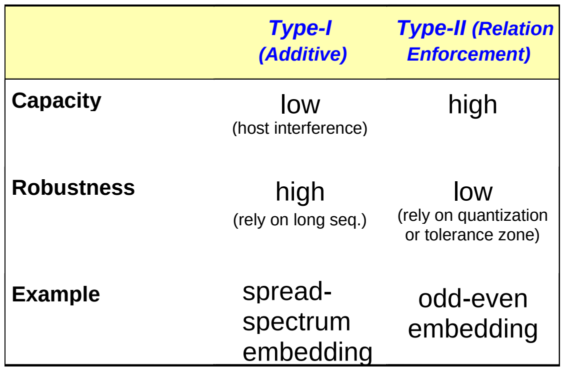
[1]Equivalently, if the embedding distortion per dimension is fixed, the total distortion that can be introduced increases when moving to higher dimensions. This aggregated energy enables embedding more reliably via quantization, as will be discussed in Sec. 3.1.1.

[2]Non-coherent detection in data hiding refers to being able to detect the embedded data without the use of the original unwatermarked copy. It is also called "blind detection".

3.1.1 Comparison

The two types of schemes reveal different characteristics in terms of robustness, capacity and distortion (introduced by embedding), as shown in Table 3.1. For Type-I schemes, hypothesis testing is a tool for verifying what hidden data is present in the test media. Spread spectrum embedding, a representative of Type-I, has been demonstrated with excellent robustness and invisibility when the original host media is available in detection [44, 46]. In non-coherent detection, the interference from host signal exists even when there is no subsequent processing or intentional attack [3].

Table 3.1: Comparison of two types of embedding mechanisms



	Type-I (Additive)	*Type-II (Relation Enforcement)*
Capacity	low (host interference)	high
Robustness	high (rely on long seq.)	low (rely on quantization or tolerance zone)
Example	spread-spectrum embedding	odd-even embedding

We can analyze the detection performance via the following simplified additive model is:

$$\begin{cases} H_0 : Y_i = -S_i + M_i \quad (i = 1, ..., n) \qquad \text{if } b = -1 \\ H_1 : Y_i = +S_i + M_i \quad (i = 1, ..., n) \qquad \text{if } b = +1 \end{cases} \tag{3.1}$$

where $\{S_i\}$ is a deterministic sequence (sometimes called *watermark*), b is one bit to be embedded and is used to antipodally modulate S_i, M_i is the noise, and n

[3]Recently, Cox *et al.* modeled the additive embedding as communication with side information and proposed techniques of "informed embedding" to reduce (but not completely eliminate) the negative impact from host interference [54, 55].

is the number of samples/coefficients to carry the hidden information. We further assume b is equally likely to be "-1" and "+1". In coherent detection where the original source is available, M_i comes from the processing and/or attack applied to the marked copy; in non-coherent detection, M_i consists of noise from the host media and processing/attack. For simplicity, M_i is usually modeled as i.i.d. gaussian distribution $N(0, {\sigma_M}^2)$, for which the optimal detector is a (normalized) correlator with S_i according to the classic detection theory [2]:

$$T_N = \underline{Y}^T \underline{S} / \sqrt{{\sigma_M}^2 \cdot ||\underline{S}||^2} \tag{3.2}$$

where $\underline{Y}$ and $\underline{S}$ are column vectors. This test statistic is gaussian distributed with unit variance and the following mean

$$\begin{aligned} E(T_N) &= b \cdot \sqrt{||\underline{S}||^2 / {\sigma_M}^2} && (3.3) \\ &= b \cdot \sqrt{n \cdot (\frac{1}{n}||\underline{S}||^2) / {\sigma_M}^2} && (3.4) \end{aligned}$$

We then compare T_N with zero, and decide H_1 if it is positive and H_0 otherwise. The probability of error is $\mathcal{Q}(E(T_N))$, where $\mathcal{Q}(x)$ is the probability of $P(X > x)$ of a gaussian random variable $X \sim N(0,1)$. Because $\mathcal{Q}(\cdot)$ is monotonically decreasing, one should raise the ratio of total watermark energy $||\underline{S}||^2$ to noise power ${\sigma_M}^2$ to get a lower probability of error. Given the same noise power, this can be achieved by adding watermark to more elements, and/or by raising the watermark power (per element). A watermark with higher power introduces more distortion on the host media. The maximum watermark power is generally determined by perceptual models so that the changes introduced by the watermark are below the *just-noticeable-difference* (JND). Therefore, the remaining way to achieve robustness is to use large n, that is, to collect energy from many weak components of a long signal and to use such a long signal to represent one bit. A longer watermark vector in turn reduces the *capacity* (i.e., the number of secondary information bits that can be encoded and extracted with very small probability of error).

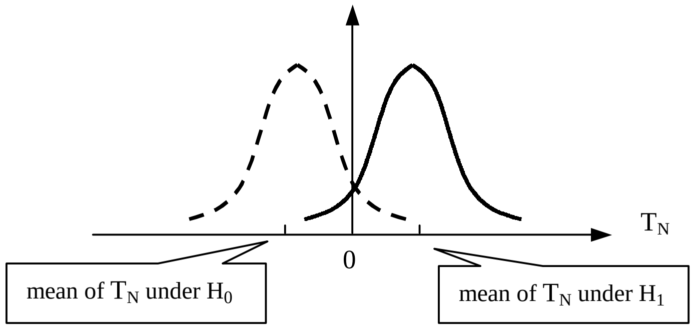


Figure 3.3: Illustration of the distribution of detection statistics (Type-I). Small mean value results in large probability of error.

It is worth mentioning that the hypothesis testing model in Eq. 3.1 is concerned with embedding one bit of information using antipodal modulation on a signal $\underline{S}$. Another popular model considers the case of a watermark being present versus being absent:

$$\begin{cases} H_0 : Y_i = M_i & (i = 1, ..., n) \quad \text{if watermark is absent} \\ H_1 : Y_i = S_i + M_i & (i = 1, ..., n) \quad \text{if watermark is present} \end{cases} \tag{3.5}$$

The watermark signal $\underline{S}$ often represents ownership information [44, 46]. The detection statistics of this hypothesis problem is the same as the previous antipodal model. While the threshold can be set according to the Bayesian rule to minimize the overall probability of error as in the previous case, the Neyman-Pearson criterion is often adopted to minimize miss detection probability $P($ choose $H_0|H_1$ is true$)$ while keeping the false alarm probability $P($ choose $H_1|H_0$ is true$)$ below a bound.

Unlike Type-I, the Type-II schemes are free from the interference from host media and have the ability of coding one bit in only a small number of host components hence high capacity. Their robustness against processing and attacks generally comes from quantization and/or tolerance zones. For schemes enforcing ordering or sign, instead of making minimal changes to enforce $v'_1 > v'_2$, the embedding mechanism may force $v'_1 > v'_2 + \delta$, where δ is the size of a tolerance zone. As long as distortion is smaller than δ, $v'_1 > v'_2$ can still be retained. For other enforcements, we may apply quantization to obtain robustness [167] [4]. For example, in the odd-even embedding, we pre-quantize the host signal I_0 by step Q, then enforce the quantized value to be an even number to embed a '0' and an odd number to embed a '1'. As shown in Fig. 3.2(c), any further distortion within (-Q/2, +Q/2) will not cause errors in detection. The larger the Q is, the more tolerance we obtain, but also the larger distortion an embedding process may introduce. This is because the mean squared error introduced by embedding, as illustrated in Fig. 3.4, is

$$
\begin{aligned}
MSE &= \frac{1}{2} \cdot \frac{Q^2}{12} + \frac{1}{2} \cdot [\frac{1}{Q} \int_{-\frac{Q}{2}}^{0} (x+Q)^2\, dx + \frac{1}{Q} \int_{0}^{\frac{Q}{2}} (x-Q)^2\, dx] \qquad (3.6)\\
&= \frac{1}{3} Q^2, \qquad (3.7)
\end{aligned}
$$

where the host components within $\pm Q$ of kQ are assumed to follow uniform distribution, giving an overall MSE quadratic with respect to Q. The uniform distribution is a good approximation when Q is small. After embedding, noise N is introduced to marked components by processing/attack, then an inverse quantization is performed by a watermark detector. If $|N| < Q/2$, no error will be introduced in detection. If N is uniformly distributed between $-M/2$ and $M/2$ where $M > Q$, the probability of error can be expressed on interval basis:

$$
P_e = \begin{cases} 1 - \frac{(2k-1)Q}{M} & M \in [(4k-3)Q, (4k-1)Q] \\ \frac{2kQ}{M} & M \in [(4k-1)Q, (4k+1)Q] \end{cases} \qquad (3.8)
$$

[4] The enforcement with quantization is formulated in a slightly different way by Chen *et al.* [72] and is referred as *Quantization Index Modulation (QIM)*.

where k is a positive integer. Here P_e fluctuates around $1/2$ and converges to $1/2$ as k goes large. While a more sophisticated set partition may achieve a better tradeoff between the perceptual distortion introduced by embedding and the tolerance against certain processing or attacks, one can see that the tolerance is always limited and is achieved by pre-distortion in the embedding step. By incorporating a proper human visual model, Type-II schemes are suitable for high-rate data hiding applications that do not have to survive severe noise.

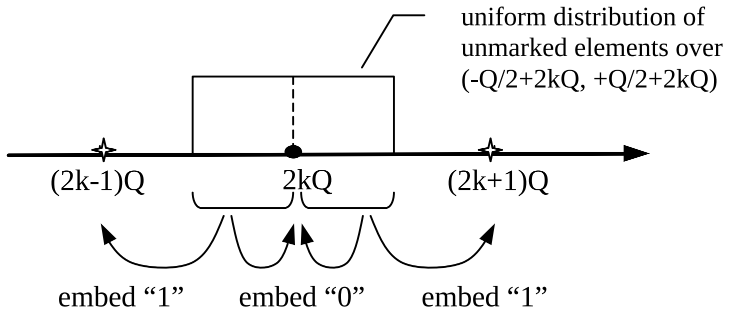


Figure 3.4: Computing MSE distortion by odd-even embedding

For both types of embedding, when the probability of detection error per embedding unit (such as in one or a set of image pixels or coefficients) is non-zero, proper channel coding like error correction codes (ECC) can be applied to achieve reliable embedding under certain noise conditions. Properly constructed codes are ways to approach the embedding capacity to be discussed in Section 3.2.

3.1.2 A Unified View on Two Types

A unified view of the two embedding types can be obtained in terms of set partitioning: both types partition signal space into several subsets, each of which represents a particular value of hidden data. We have already explained the set partitioning for Type-II. Now considering Type-I, for example, the additive spread spectrum scheme, we can see that the signal space for detection is also partitioned into two parts according to the sign of test statistics T_N which usually takes the form of correlation or a variation of correlation: the positive part represents a '1' and the negative part represents a '0'. The difference is that for Type-I, enforcement is not done on the marked media deterministically. The additive embedding alone still leaves non-zero probability for marked components not to be enforced to the desired set in non-coherent detection. Because the host media is a major noise source, we have to rely on a statistical approach (e.g., spreading watermark signal to many components and taking average) to suppress noise and to obtain detection result with small probability of error. This effort is needed even when there is no noise coming from processing/ attack.

Most recently, attentions have been paid to Costa's theoretical work in the early 1980s on the channel capacity under two additive gaussian noise sources with one noise source being known to the sender [97]. Costa showed that the channel capacity is equal to the capacity in the absence of the known noise source and that the optimal transmitter adapts its signal to the state of the known noise source rather than attempting to cancel it. Incorporating Costa's work into the data hiding problem, as suggested by Moulin *et al.* and Chen *et al.*, provides an alternative unified view on the two embedding types [93, 72].

3.2 Quantified Capacity Study

The difference between the two types of schemes in terms of robustness-capacity tradeoff can be quantified using an additive channel model. For simplicity, we assume additive white gaussian noise (AWGN) for both types. Other additive noise conditions such as additive white uniform noise (AWUN) and colored noise can be studied similarly by applying whitening and/or re-computing the capacity based on information theory. It is important to notice that the capacity is tied to a channel model with specific noise distribution and watermark signal constraints. The capacity would be different if the channel is modeled differently, and it is a function of the parameters of the noise distribution and watermark constraints, such as the power of the noise and of the watermark.

Capacity for Type-I Embedding

The channel model of Type-I schemes shown in Fig. 3.1 has continuous input and continuous output (CICO). The additive noise consists of two parts, namely, the interference from the host signal and the noise due to other processing/distortion. For the moment, we assume that (1) the host signal is independent of the processing noise, and (2) both are i.i.d. gaussian distributed. The embedding capacity under the overall AWGN noise is achieved with gaussian distributed input and is equal to

$$C_{CICO} = \frac{1}{2}\log_2(1 + \frac{A^2}{{\sigma_I}^2 + \sigma^2}). \tag{3.9}$$

where ${\sigma_I}^2$ is the power of the original host signal, A^2 is the power of the embedded signal, and σ^2 is the power of additive processing noise. In general, the interference from host signal is much stronger than the processing noise, i.e., ${\sigma_I}^2 >> \sigma^2$.

Capacity of Type-II Embedding

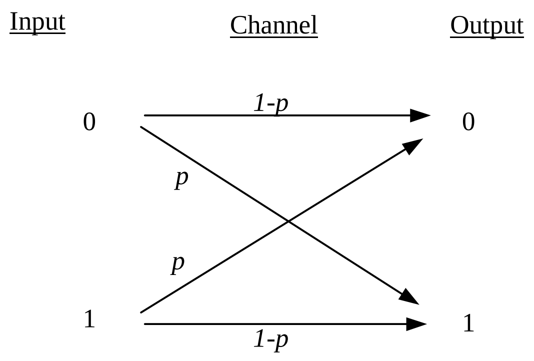


Figure 3.5: A binary symmetric channel (BSC) with a flipping probability of p.

The channel of Type-II schemes has discrete input with the decision boundary being either single sided or double sided, as illustrated in Fig. 3.2. The single-sided case generally corresponds to the embedding relying on sign enforcement, and the double-sided case is common to denser enforcement such as odd-even and lookup table embedding. We shall first study the single sided case, and extend the result to the double-sided case later. Regarding the output of the channel, if a hard decision is used, the channel will be the discrete-input discrete-output (DIDO) binary symmetric channel (BSC) shown in Fig. 3.5. The capacity of this type of channel has been well studied and is given by

$$C_{DIDO} = 1 - h_p \tag{3.10}$$

achieved by equiprobable input, where h_p is the binary entropy

$$h_p = p \cdot log(\frac{1}{p}) + (1-p) \cdot log(\frac{1}{1-p}). \tag{3.11}$$

The probability of bit error p for AWGN and AWUN noise are

$$p_{AWGN} \quad = \quad \mathcal{Q}(\frac{A}{\sigma}) = \int_{\frac{A}{\sigma}}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}\, dt, \tag{3.12}$$

$$p_{AWUN} = \begin{cases} \frac{1}{2} - \frac{A}{M} & A < M/2 \\ 0 & A \geq M/2 \end{cases}, \quad \text{with } \sigma = M/\sqrt{12}. \tag{3.13}$$

where the AWUN noise is uniformly distributed between $-M/2$ and $+M/2$ (noise variance $\sigma^2 = M^2/12$), If a soft decision is used, which offers a detector the knowledge of how wrong the decision of a particular element would be, the channel will be discrete-input continuous-output (DICO), having a capacity of

$$C_{AWGN,DICO} = 1 + \frac{A^2}{\sigma^2} \log_2 e - E[\log_2(e^{\frac{2AY}{\sigma^2}} + 1)], \tag{3.14}$$

where $E[\cdot]$ is the expectation with respect to Y, whose probability density function is

$$f(y) = \frac{1}{2\sqrt{2\pi\sigma^2}} e^{-\frac{(y+A)^2}{2\sigma^2}} + \frac{1}{2\sqrt{2\pi\sigma^2}} e^{-\frac{(y-A)^2}{2\sigma^2}}. \tag{3.15}$$

When the noise is AWUN between $-M/2$ and $+M/2$, we can show that:

$$C_{AWUN,DICO} = \begin{cases} \frac{2A}{M} & A < M/2 \\ 1 & A \geq M/2 \end{cases} \Rightarrow C_{AWUN,DICO} = \begin{cases} \frac{A}{\sqrt{3}\sigma} & \frac{A}{\sigma} < \sqrt{3} \\ 1 & \frac{A}{\sigma} \geq \sqrt{3} \end{cases} \tag{3.16}$$

The soft decision allows the *watermark signal-to-noise ratio* A/σ^2 being $2 \sim 5$dB lower for the same capacity than the hard decision under AWGN or AWUN noise, as shown in Fig. 3.6. The derivations of $C_{AWGN,DICO}$ and $C_{AWUN,DICO}$ are included as an appendix in Sec. 3.5.

Up to now, we have discussed a simple case with only two signal points $-A$ and $+A$ conveying one bit information. Capacity of different channel models, namely, DICO and DIDO, have been studied and compared. With a few small variations, we can obtain the capacity of several typical Type-II schemes. For practical schemes that enforces signs with a tolerance zone A, the signal are generally enforced to be greater than $+A$ or less than $-A$ to encode one bit, rather than being enforced exactly to

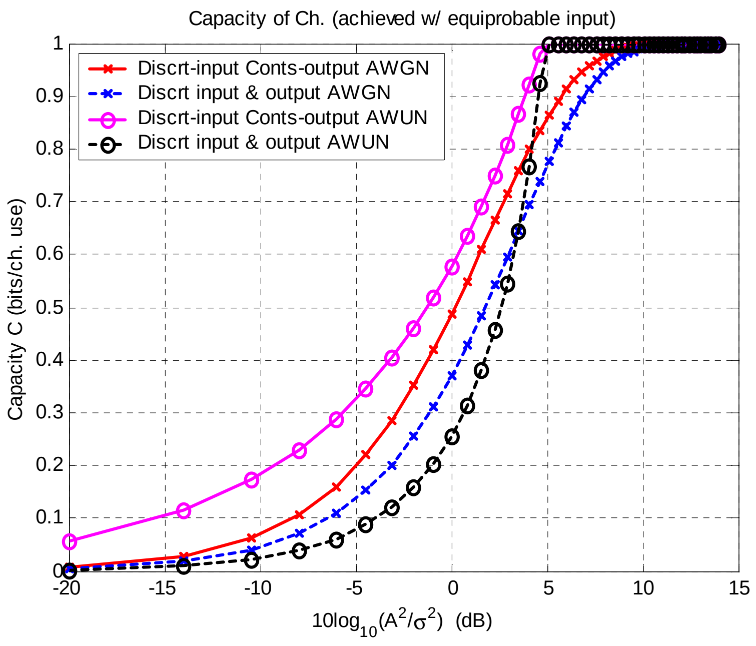


Figure 3.6: Capacity of DICO and DIDO channels under AWGN and AWUN noise.

$\pm A$ in our simplified model. This implies that the capacity should be higher than that of our model. Regarding the odd-even embedding mentioned earlier, the error regions are two-sided rather than single-sided (Fig. 3.2). We denote the quantization step size as Q and consider the channel input $X = kQ$ (i.e., the enforced values that carry hidden information), error will be incurred when the output Y is in the regions $Y > (k + 1/2)Q$ or $Y < (k - 1/2)Q$. Thus for the model of DIDO channel with AWGN noise, the bit error probability p is:

$$\begin{aligned} p_{AWGN} &= \min\{1/2,\, 2\cdot\sum_{k=0}^{+\infty}\mathcal{Q}(\frac{(4k+1)Q}{2\sigma}) - \mathcal{Q}(\frac{(4k+3)Q}{2\sigma})\} \\ &= \min\{1/2,\, 2\cdot\sum_{k=0}^{+\infty}\int_{\frac{(4k+1)Q}{2\sigma}}^{\frac{(4k+3)Q}{2\sigma}}\frac{1}{\sqrt{2\pi}}e^{-\frac{t^2}{2}}\,dt\}, \end{aligned} \tag{3.17}$$

For high watermark-to-noise ratio ($\frac{Q}{\sigma}$), we may ignore the regions that are far away from kQ but map to the same bit value as kQ and approximate the probability with a upper bound:

$$p_{_{AWGN}} \approx \min\{1/2, 2 \cdot \mathcal{Q}(\frac{Q}{2\sigma})\} = \min\{1/2, 2 \cdot \int_{\frac{Q}{2\sigma}}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}\, dt\}, \qquad (3.18)$$

This approximation is based on the fast decay of the tails of gaussian distribution. Plugging the result of $p_{_{AWGN}}$ into Eq. 3.10 yields the channel capacity.

We have discussed *DIDO* and *DICO* channel models that reflect the practically implementable Type-II embedding. The channel model for Type-II embedding can be further improved. Motivated by Costa's techniques in proving the channel capacity [97], Chen *et al.* proposed to incorporate multiplicative scaling into quantization-based enforcement embedding. The enforcement is then linearly combined with the host signal to form a watermarked signal. The scaling factor is a function of watermark-to-noise ratio and has the capability of enhancing the number of bits that can be embedded. Interested readers may refer to [40, 72] for details.

Capacity Comparison for Type-I & Type-II

Fixing the mean squared error introduced by the embedding process as E^2, we compare the capacity of Type-I and Type-II schemes under AWGN noise with the following simplification. For Type-I, we consider a CICO channel model and assume that the AWGN noise consists of gaussian processing noise (with variance σ^2) and host interference (with standard deviation 10 times as much as that of the watermark signal, i.e., $\sigma_I = 10E$). For Type-II, we consider a DIDO BSC channel with p derived in Eq. 3.17 for odd-even embedding with such quantization step Q that the embedding MSE distortion equals to E^2, i.e., $Q = \sqrt{3}E$ according to Eq. 3.7. The capacity is thus obtained as:

$$C_I = \frac{1}{2}\log_2(1 + \frac{E^2}{(10E)^2 + \sigma^2}) \qquad (3.19)$$

$$C_{II} = 1 - h_{\min\{1/2,\, 2\cdot\sum_{k=0}^{+\infty} \mathcal{Q}(\frac{(4k+1)Q}{2\sigma}) - \mathcal{Q}(\frac{(4k+3)Q}{2\sigma})\}} \tag{3.20}$$

We plot capacity vs. different watermark-to-noise ratio E^2/σ^2 in Fig. 3.7. It shows that the capacity of Type-II is much higher than that of Type-I until the watermark-to-noise ratio (WNR) falls negative, confirming our previous analysis regarding the host interference of Type-I and the pre-distortion nature of Type-II. The comparison suggests that Type-II is useful under low noise condition while Type-I is suitable for severe noise. The capacity of both Type-I and Type-II can be approached via channel coding, such as RS / BCH codes used in [61, 167].

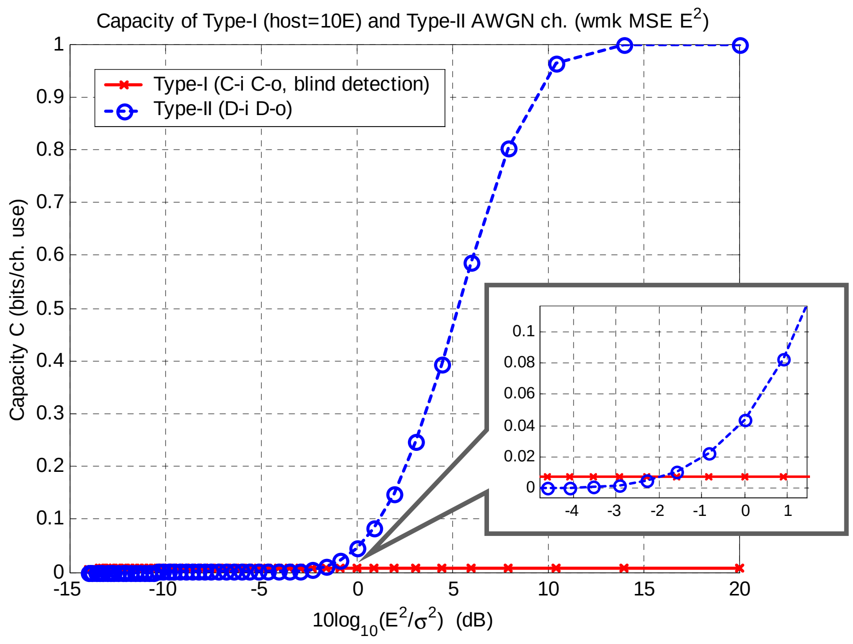


Figure 3.7: Capacity of Type-I (CICO channel) and Type-II (DIDO channel) embedding under AWGN noise.

A Few Extensions

We have discussed the two possible channel models for Type-II embedding, namely, DICO and DIDO channels, and have compared them with the CICO channel model for Type-I embedding. Throughout the above discussion, we assumed that the noise is additive and white, and the watermark signal power is the same on all media components. Generalization is possible on two aspects. First, We should consider the so-called *unembeddable* components which have to be left untouched by the embedding mechanism to meet imperceptibility requirement. As we shall see in Chapter 4, these unembeddable components reduce the data hiding capacity. Second, we should consider each media components might incur different host interference, be able to watermarked with different strength, and/or sustain different noise. The channel model can be modified into a parallel channel, with L bands per unit or channel use. The ratio of watermark to interference-plus-noise is different for in each band, but the total noise is assumed to be independent from band to band and is i.i.d. from unit to unit (Fig. 3.8). The channel capacity per unit becomes:

$$C = \max_{P_{X^{(L)}}} I(X^{(L)};Y^{(L)}), \tag{3.21}$$

where

$$\begin{aligned} I(X^{(L)};Y^{(L)}) &= I([X_1,...,X_L];[Y_1,...,Y_L]) = h(Y^{(L)}) - h(Y^{(L)}|X^{(L)}) \\ &= h(Y^{(L)}) - h(N_1,...,N_L) = h(Y_1,...,Y_L) - \sum_{i=1}^{L} h(N_i) \\ &\leq \sum_{i=1}^{L} h(Y_i) - \sum_{i=1}^{L} h(N_i) = \sum_{i=1}^{L} I(X_i;Y_i). \end{aligned} \tag{3.22}$$

If $\{X_i\}$ are independent of each other, the equality of Eq. 3.22 holds, indicating that the total capacity is the sum of the capacity achieved by each individual band. The capacity of an individual band can be determined by the noise power and the just-noticeable-difference of the band, using the approach we discussed in the earlier part of this section. Studying the capacity under correlated noise from unit to unit and/or non-gaussian noise is more involved. It is a direction of future work.

A noteworthy issue regarding the parallel channels is that in classic communication literature, the capacity of L parallel AWGN channels follows a so-called *Water-filling Theorem*, where a constraint on total power is imposed and the power needs to be shared among all channels. The theorem suggests an optimum allocation of the power among L parallel channels, which is analogous to water-filling. Though meaningful in the cases of telecommunication, the constraint on the total power may not be realistic in watermarking problems where the power constraint for each individual channel (possibly in the form of a frequency band or a local region) is determined by perceptual models such as those in [101, 36, 48]. Even though the perceptual constraints may have dependency among several channels, it does not appear to fit the simple constraints on the total power. How to better incorporate the dependency among channels is another direction to be explored.

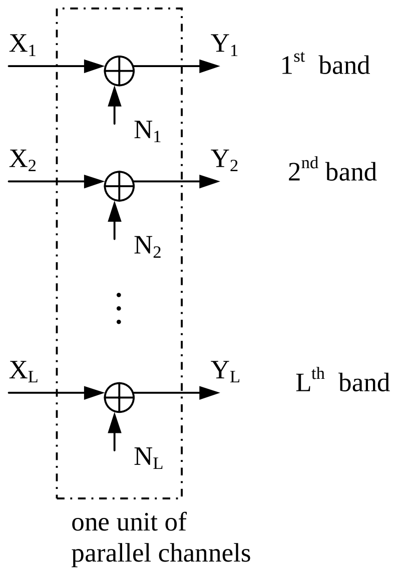


Figure 3.8: Parallel Channel Model With L Bands. Noise in each band is independent but with different variance.

We shall also note from the above analysis that the discrete-input channel model for Type-II has zero probability of detection error if the support of noise distribution is within the decision boundary shown as shaded area in Fig. 3.2. Under the specific model considered there, the channel is able to convey 1 bit per channel use with no error. If we revise the channel model to allow freedom in choosing the input alphabet, the channel capacity is determined by Shannon's zero-error capacity [99, 7], as suggested in [41]. The capacity may exceed 1 bit per channel use for a specific noise distribution and specific power constraint on the watermark.

3.3 Bandwidth via Data Hiding

A specious argument about data hiding is that it could provide *additional* bandwidth to convey secondary information. Actually the bandwidth for conveying secondary information is at an expense of either reducing the bandwidth for conveying host media or increasing the total bandwidth of conveying the watermarked media, depending on whether the quality of host media is reduced or not. Considering the simplest case of embedding one bit in an image (Fig. 3.9), the entire image space $\mathcal{S}$ is always partitioned into two disjoint subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ with $\mathcal{S} = \mathcal{S}_1 \bigcup \mathcal{S}_2$, regardless of the specific embedding algorithm being used. When a detector sees an image belonging to $\mathcal{S}_1$, it will output the embedded bit value as "0"; when it sees an image belonging to $\mathcal{S}_2$, it will output the embedded bit value as "1". Assuming the embedded bit takes "0" and "1" equiprobably, the probability of an image falling into the first and the second subset equals to 1/2, respectively. To code any image in the space, one bit is spent on specifying to which subset the image belongs. In other words, one of the bits used in coding an image is actually for conveying the embedded bit.

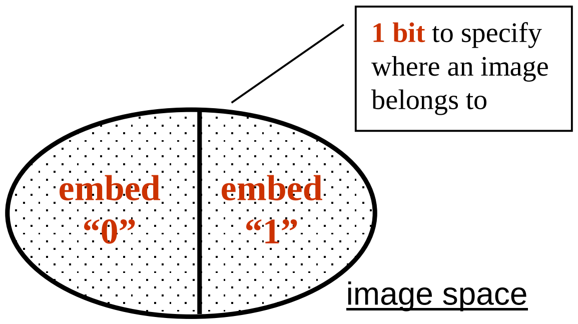


Figure 3.9: Illustration of the bit re-allocation nature of data hiding.

For odd-even embedding in a single image element, the two subsets are obtained by a partition according to the least significant bit (LSB); for an odd-even enforcement applied to the sum of two components, the embedded bit is related to the LSB of both components. For spread spectrum additive embedding, the boundary between the two subsets is commonly determined by a correlator-type detector. If the total number of bits for representing an image do not change during the embedding process, one bit is logically reallocated from representing the image to representing the embedded data, even though there may be more than one bits physically related to the embedded data. In this case, the absolute quality of the image is reduced because of the fewer bits effectively used in image representation (unless there is redundancy in the previous representation). While this indicates that data hiding does not have an advantage in terms of saving bit rate when compared with attaching the secondary data separately to the host media, it does have quite a few advantages, including the ability to associate the secondary data with the host media in a seamless way as well as the standard compliant appearance and the low computation complexity in practical applications. We will discuss this in Chapter 8.

3.4 Modulation and Multiplexing Techniques for Embedding Multiple Bits

An embedding scheme generally specifies a particular way to hide one-bit information in multimedia source. Modulation/multiplexing techniques can be applied on top of it to embed multiple bits. Evolving from classic communication [5], the following strategies are commonly used:

- *Amplitude modulo modulation* (for Type-II embedding [5])

  In general, B bits can be embedded in each embedding unit by enforcing a feature derived from this unit into one of M subsets, where $B = \log_2 M$. A straightforward example extended from odd-even embedding is to enforce the relation via modulo-M operation to hide B bit per element. That is,

$$I_1 = \arg_I \min_{\text{s.t. } I=kQ, k\in\mathbf{Z}, \text{mod}(k,M)=m} |I - I_0| \tag{3.23}$$

  where $[\cdot]$ represents the rounding operation, $m \in \{0, 1, ..., M-1\}$ represents the B-bit information to be embedded, I_0 is the original image feature, I_1 is the watermarked feature, and Q is the quantization step size for obtaining robustness. Assuming I_0 follows uniform distribution in each quantization interval $(kQ - \frac{Q}{2}, kQ + \frac{Q}{2})$ where k is an integer, we can show that the MSE distortion introduced by embedding is $Q^2M^2/12$. This indicates that with the minimal separation Q between the M subsets being fixed, larger embedding distortion will be introduced by a larger M; with fixed MSE embedding distortion, the

[5]The Type-I additive embedding formulated in Eq. 3.1 (the antipodal modulation) and Eq. 3.5 (the on-off modulation) can be viewed as amplitude modulation. For blind detection of additive embedding that is subject to host interference, using amplitude modulation to convey more than two constellation points are rare in practice. We therefore focus on the amplitude modulo modulation that is applicable to Type-II embedding.

enforced relation with a larger M has smaller separation hence tolerates less distortion. The idea is easily extensible to table lookup embedding or other enforcement scheme and the analysis can be done similarly.

- *Orthogonal & Biorthogonal modulation*

  Mainly used with Type-I additive embedding, the *orthogonal modulation* uses M orthogonal signals to represent $B = \log_2 M$ bits by embedding one of the M signals into the host media. A detector computes the correlation with respect to all M signals. The signal that gives the largest correlation and exceeds some threshold will be decided as the signal embedded by the sender and the corresponding B-bit value will be determined accordingly. A variation, so-called *biorthogonal modulation*, encodes $\log_2 2M = (B+1)$ bits by adding or subtracting one of M signals. The computational complexity of detection is exponential with respect to the number of bits being conveyed, therefore is inefficient except for small M.

- *"TDMA" type modulation*

  For data hiding in images, this type of modulation partitions an image into non-overlapped regions and hides one or several bits in each region. For audio, it means to partition an audio into time segments and to hide one or several bits in each segment. A video can be partitioned into regions within each frame and into time segments across frames. "TDMA" type modulation is a simple way to realize orthogonal embedding for both Type-I and Type-II, i.e., the bits embedded in different regions or segments do not interfere with each other. However, it could suffer from the problem of uneven embedding capacity, which will be explained in Section 4.

- *"CDMA" type modulation*

  For Type-I additive embedding, encoding B bits to a watermark signal $\underline{w}$ in the following way provides more efficient detection than orthogonal modulation in terms of the computational complexity:

  $$\underline{w} = \sum_{k=1}^{B} b_k \cdot \underline{u}_k, \tag{3.24}$$

  where $b_k = \pm 1$. In general, the vectors $\{\underline{u}_k\}$ are chosen to be orthogonal to each other. The orthogonality implies that the total signal energy is the sum of the energy allocated for each bit. If a fixed amount of energy is uniformly allocated to each bit, the energy per bit will be reduced as B increases, implying a decrease in detection reliability and more generally, a limit on the total number of bits that can be hidden for low error rate extraction. "TDMA" is a special case with the supports of $\underline{u}_k$ being non-overlapped with each other in the sample domain (i.e., the pixel domain for image and the time domain for audio). Alternatively, we can choose orthogonal but overlapped $\{u_k\}$, similar to CDMA in communication [6]. Uneven embedding capacity is no longer a concern as we can choose $\{u_k\}$ such that each bit is spreaded all over the media. But B orthogonal sequences have to be generated and shared with a detector, which may be non-trivial for large B. The TDMA and CDMA approach can be combined to encode multiple bits.

  For Type-II embedding, multiple bits can be embedded by enforcing relations deterministically along multiple directions that are orthogonal to each other. Swanson *et al.* and Alghoniemy *et al.* proposed to embed multiple bits in an image block by enforcing relations on the projections of a feature vector along several orthogonal directions [66, 68]. The total modification introduced by embedding is the sum of the change along each direction, implying a tradeoff among capacity, robustness, and imperceptibility.

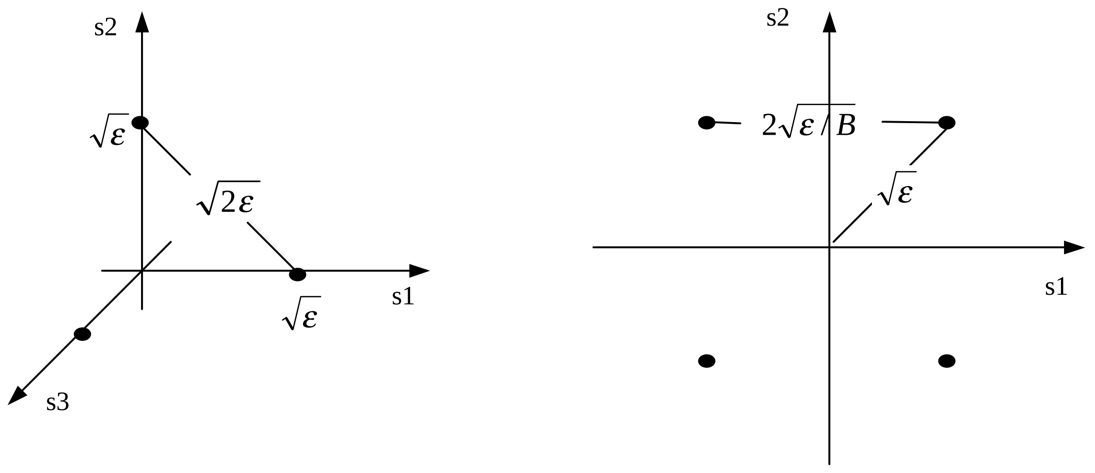


Figure 3.10: Comparison of distance between signal constellation points for orthogonal modulation (left) vs. TDMA/CDMA-type modulation (right) with total signal energy being fixed at $\mathcal{E}$.

The orthogonal modulation and TDMA/CDMA-type modulation can be compared by studying the distance between signal constellation points that represent the secondary data (Fig. 3.10). Considering the case of conveying B bits using total energy $\mathcal{E}$. The minimum distance between signal points is $\sqrt{2\mathcal{E}}$ for orthogonal modulation, and is $2\sqrt{\mathcal{E}/B}$ for TDMA/CDMA. When $B > 2$, orthogonal modulation gives smaller probability of error at a cost of detection complexity.

Expanding a bit more, we summarizes the quantified comparison among the modulation/multiplexing techniques discussed above [6] in Table 3.2. The modulation/multiplexing is applied to one embedding unit of S elements. The quantity

[6]The modulo-M modulation extended from odd-even embedding is taken as a representative of amplitude modulation.

$\mathcal{W} = \frac{\mathcal{Y}}{\mathcal{X}\cdot\mathcal{Z}^2}$ measures the energy efficiency of embedding, where $\mathcal{Y}$ is the MSE distortion per element introduced by embedding, and $\mathcal{Z}$ is the minimum separation between the enforced constellation points hence reflects the robustness against noise. Because $\mathcal{W}$ describes the MSE embedding distortion per bit per unit squared separation distance, a smaller value is more preferable. We can see that except for very small S and B, biorthogonal techniques has the smallest $\mathcal{W}$ values, while the amplitude modulo technique gives large $\mathcal{W}$ values as M goes larger – it equals to $\frac{1}{3}$ for $M = 2$, and to $\frac{2}{3}$ for $M = 4$. TDMA and CDMA modulation, being applicable to both Type-I and Type-II embedding under blind detection and having a constant $\mathcal{W}$ value of $\frac{1}{4}$, show a good balance between energy efficiency and detection complexity, therefore are suitable for many applications. Further, TDMA or CDMA can be combined with orthogonal or biorthogonal modulation to enhance the embedding rate while balancing the detection complexity.

3.5 Appendix - Derivations of Type-II Embedding Capacity

In this appendix section, we derive the capacity under DICO channel model for Type-II embedding. We shall consider AWGN and AWUN noises, and show the capacity under these channels follow Eq. 3.14 and Eq. 3.16, respectively.

According to information theory [7], the channel capacity is

$$C = \max_{p(x)} I(X;Y) \tag{3.25}$$

where $I(X;Y)$ is the mutual information between two random variables X and Y.

Table 3.2: Comparison of Modulation/Multiplexing Techniques

(S elements per embedding unit, $B \le S$)

	Amplitude Modulo	*TDMA / CDMA*	Orthogonal	Biorthogonal
Type-I embed.		v	v	v
Type-II embed.	v	v		
$\mathcal{X}$ # embedded bits per element	$\frac{\log_2 M}{S}$	$\frac{B}{S}$	$\frac{\log_2 B}{S}$	$\frac{\log_2 2B}{S}$
$\mathcal{Y}$ MSE distortion per element	$\frac{Q^2 M^2}{12S}$	$\frac{\mathcal{E}}{S}$	$\frac{\mathcal{E}}{S}$	$\frac{\mathcal{E}}{S}$
$\mathcal{Z}$ minimum separation	Q	$2\sqrt{\frac{\mathcal{E}}{B}}$	$\sqrt{2\mathcal{E}}$	$\sqrt{2\mathcal{E}}$
$\mathcal{W} = \frac{\mathcal{Y}}{\mathcal{X} \cdot \mathcal{Z}^2}$	$\frac{M^2}{12 \log_2 M}$	$\frac{1}{4}$	$\frac{1}{2 \log_2 B}$ $\left(\ge \frac{1}{2 \log_2 S}\right)$	$\frac{1}{2(1+\log_2 B)}$ $\left(\ge \frac{1}{2(1+\log_2 S)}\right)$

For a channel with continuous outputs, we have

$$I(X;Y) = h(Y) - h(Y|X) = h(Y) - h(X+Z|X) = h(Y) - h(Z) \tag{3.26}$$

where $h(\cdot)$ is the differential entropy of a continuous random variable, $h(\cdot|\cdot)$ is the conditional differential entropy, and Z is additive noise that is independent of the channel input. Consider first the case of AWGN noise $N(0, \sigma^2)$, whose differential entropy is known as $\frac{1}{2}\log(2\pi e \sigma^2)$. We have

$$I(X;Y) \quad = \quad E[-\log f_{_Y}] - \frac{1}{2}\log(2\pi e \sigma^2) \tag{3.27}$$

where the expectation $E[\cdot]$ is with respect to the random variable Y whose probability density function (p.d.f.) f_Y is a bimodal gaussian (?), i.e.,

$$f_Y(y) = P(X=-A)\cdot\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(y+A)^2}{2\sigma^2}} + P(X=+A)\cdot\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(y-A)^2}{2\sigma^2}} \tag{3.28}$$

By symmetry, the capacity is achieved by equiprobably input, i.e., $P(X=-A) = P(X=+A) = 1/2$. We now have

$$\log f_Y(y) = -\log 2 - \frac{1}{2}\log(2\pi\sigma^2) - \frac{A^2}{2\sigma^2}\log e - \frac{y^2}{2\sigma^2}\log e + \log(e^{-B}+e^{B}) \tag{3.29}$$

where $B = yA/\sigma^2$. The term $\log(e^{-B}+e^{B})$ can be simplified as

$$\log(e^{-B}+e^{B}) = \log\frac{e^{2B}+1}{e^{B}} = \log e^{\frac{2yA}{\sigma^2}} + 1 - \frac{yA}{\sigma^2}\log e \tag{3.30}$$

We take expectation with respect to Y on every term in $-\log f_Y(y)$ and obtain

$$h(Y) = \log 2 + \frac{1}{2}\log(2\pi\sigma^2) + \frac{A^2}{2\sigma^2}\log e + \frac{\log e}{2\sigma^2}E(Y^2) - E[\log e^{\frac{2AY}{\sigma^2}} + 1] \tag{3.31}$$

where the term $E[\frac{YA}{\sigma^2}\log e]$ vanishes because Y has zero mean. With $E(Y^2) = \sigma^2$ and some more rearrangement, we arrive at

$$C_{AWGN,DICO} = \log 2 + \frac{A^2}{\sigma^2}\log e - E[\log(e^{\frac{2AY}{\sigma^2}} + 1)]. \tag{3.32}$$

Therefore, the channel capacity in unit of bit per channel use under AWGN noise is

$$\boxed{C_{AWGN,DICO} = 1 + \tfrac{A^2}{\sigma^2}\log_2 e - E[\log_2(e^{\frac{2AY}{\sigma^2}} + 1)]} \tag{3.33}$$

For AWUN noise between $-M/2$ and $+M/2$ (noise variance $\sigma^2 = M^2/12$), the differential entropy of noise is

$$h(Z) = \int_{-M/2}^{M/2}\frac{1}{M}\log M\,dz = \log M \tag{3.34}$$

The shape of the output Y's distribution depends on the relations between M and A. We can show that

$$h(Y) = \begin{cases} \frac{2A}{M}h(p) + \log M & A < M/2 \\ h(p) + \log M & A \geq M/2 \end{cases} \tag{3.35}$$

where p is the probability of the channel input $p = P(X = -A)$, and $h(p)$ is the binary entropy defined as $h(p) = -p \cdot \log p - (1-p) \cdot \log(1-p)$. By noticing $h(p)$ assumes its maximum at $p = 1/2$, we have

$$\boxed{C_{AWUN,DICO} = \begin{cases} \frac{2A}{M} & A < M/2 \\ 1 & A \geq M/2 \end{cases}} \tag{3.36}$$

where the capacity is achieved by equiprobable inputs.

Chapter 4

Handling Uneven Embedding Capacity

We have pointed out in previous chapters that the design of a data hiding system involves several conflicting requirements, such as imperceptibility, robustness / security, and capacity. Depending on the specific applications, these requirements are given different weights and in general a tradeoff has to be made. Compared with this widely discussed tradeoff, another challenge, the so-called *uneven embedding capacity*, has received little attention in literature. It is, however, an important and unavoidable problem that every data hiding system has to consider. This chapter discusses and proposes a few ways to handle uneven embedding capacity.

The unevenly distributed embedding capacity for multimedia data hiding comes from the non-stationary nature of perceptual sources. Taking an image as an example, we can see that some regions of an image are smooth while others are busier with edges and textures. Changes made in smooth areas are easier to be perceived than those in texture areas; in terms of data hiding, fewer bits can be embedded in smoother regions, yielding unevenly distributed embedding capacity from region to region. For the convenience of discussion, we refer to a pixel or coefficient of the media source as *embeddable* if it can be modified by more than a predetermined amount without introducing perceptible distortion, where the predetermined amount of modification is usually determined by both robustness and imperceptibility requirements. For example, a DCT coefficient whose magnitude is smaller than a threshold may be considered as *unembeddable*. The uneven distribution of embeddable coefficients from region to region is one reflection of the uneven embedding capacity.

While it is desirable to embed as many bits as possible in each region (i.e., using a variable embedding rate), the side information regarding how many bits are actually embedded in each region has to be conveyed to a detector in order to achieve accurate decoding. Under blind detection where a detector does not have the original unwatermarked copy, accurately estimating how many bits are embedded in each region is not always easy, especially when the watermarked image may experience some distortion. An error in this estimation may not only incur errors in determining the data embedded in the associated region but also incur synchronization errors that affect the data extracted from the following regions. Unless the number of bits that can be embedded in each region is large, conveying side information is quite expensive and sometimes even exceeds the number of bits that can be reliably embedded. Embedding a fixed number of bits in each region, which is popular in the literature, eliminates the need of sending much side information. But using a constant embedding rate wastes embedding capabilities in regions that are capable of hiding more bits. In addition, the size of each region has to be large enough to accommodate the worst case (i.e., the smoothest regions). Large region size reduces the total number of bits that can be embedded.

We propose a comprehensive solution to handling uneven embedding capacity. More specifically, if the total number of bits that can be embedded is much larger than the number of bits to convey how many bits are embedded, we propose to use a variable embedding rate and to select appropriate multiplexing technique hide the side information as *control bits* to facilitate detection without introducing too much overhead. If the two bit numbers are comparable, we propose constant rate embedding with two approaches handling uneven embedding capacity, namely, backup embedding and shuffling. The backup embedding may be viewed as a special case of shuffling. We will show via analysis and experiments that shuffling is an efficient and effective tool to equalize uneven embedding capacity. In later chapters, we will demonstrate how the proposed solution is applied to specific design problems. Multi-level data hiding for video (Chapter 6) shows our adaptive choice of embedding rate in each video frame, while shuffling is adopted both within a video frame and in image data hiding works (Chapter 5 and 7).

We start our discussion with a quantitive model of the uneven embedding capacity in a natural image in Section 4.1. We then discuss constant and variable rate embedding in Section 4.2 and Section 4.3, respectively. Three examples from the design of later chapters are briefly outlined in Section 4.4 to demonstrate how the proposed approaches can be used for designing practical data hiding systems.

4.1 Quantitive Model for Uneven Embedding Capacity

To quantitively illustrate the uneven embedding capacity and to facilitate the discussion later in the paper, we consider an image of size $S = M_1 \times M_2$ is block-DCT transformed, and each transform coefficient is labeled as "embeddable" or "unembeddable". The embeddability here is determined by a human visual model. The coefficients whose magnitude is smaller than a perceptual threshold are left unchanged to avoid artifacts [36, 163]. For simplicity, we also leave DC coefficients unchanged because the change, unless very small, is likely to introduce blocky artifacts (especially in smooth regions). With these perceptual considerations, a smooth block may have no embeddable coefficients at all because all AC coefficients are small. In a typical natural image such as the one shown in Fig. 4.1, about 20% of the 8×8 blocks are smooth and have no embeddable coefficients at all. This is illustrated in Fig. 4.2.

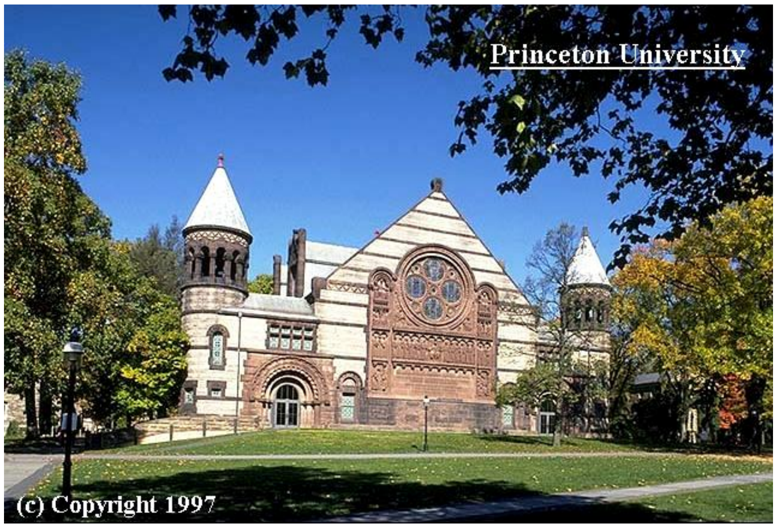


Figure 4.1: An original unmarked 640×432 image *Alexander Hall* (stored in JPEG format with a quality factor of 75%). Watermarking and related studies are performed on its luminance components.

More abstractly, we assume that a fraction of p among a total of S coefficients are embeddable, thus the total number of embeddable coefficients is $n = p \cdot S$. Concatenating coefficients in all blocks into a single string of length S, we divide it into segments of equal length q, obtaining a total of $N = S/q$ segments. Let m_r be the number of segments having r embeddable coefficients, where $r = 0, 1, 2, ..., q$. Then $\{\frac{m_r}{N}\}$ forms a histogram with each bin representing the fraction of segments having

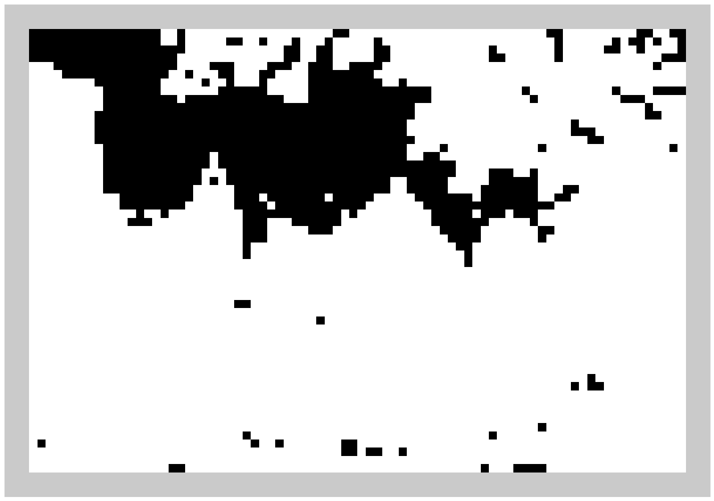


Figure 4.2: Smooth blocks of Fig. 4.1 (shown in black)

i embeddable coefficients. For the image in Fig. 4.1 and block size $q = 8 \times 8 = 64$, we plot the histogram of $\frac{m_r}{N}$ vs. r as the dash line in Fig. 4.3. It can be seen that about 20% of the blocks have no embeddable coefficients at all, while a small number of blocks have as many as 25 embeddable coefficients, showing large deviation in the distribution of embeddable coefficients.

4.2 Constant Embedding Rate (CER)

In the introduction section, we have explained the dilemma in choosing an embedding rate under uneven embedding capacity. On one hand, using a variable embedding rate generally requires sending side information about the embedding rate, which could be an expensive overhead; on the other hand, using a constant embedding rate may waste embedding capabilities. In this section, we shall focus on a constant embedding

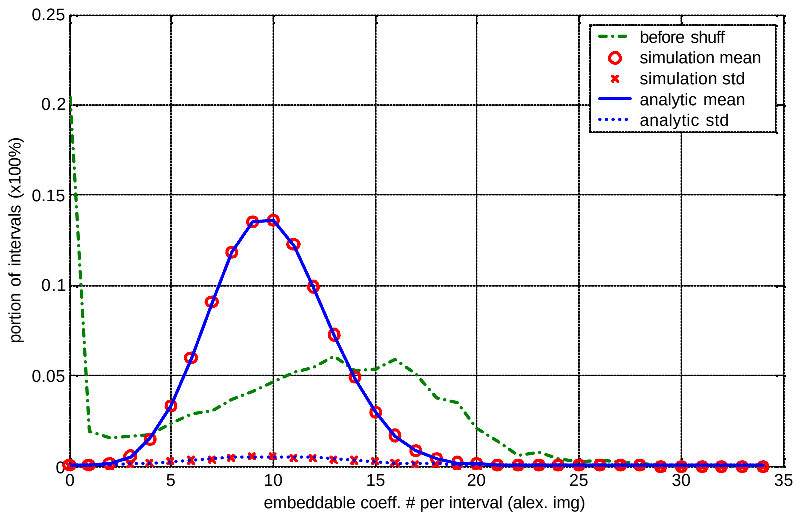


Figure 4.3: Histogram of embeddable coefficients per block for the luminance components of Fig. 4.1 before and after shuffling. (dash-dot line) - histogram before shuffling. All others are after shuffling: (solid line) - mean of simulation; (dot lines) - std of simulation; (circle) - mean from analytic study; (cross) - std from analytic study.

rate and explore approaches that can improve the value of the constant embedding rate.

The simplest case of constant embedding rate is to embed one bit in each segment or block by either Type-I or Type-II mechanisms discussed in Chapter 3. Taking an image as an example, the blocks are normally obtained by a regular partition on an image, which retains the original geometric layout of the image. As illustrated earlier in Fig. 4.3, unless the block size is large, blocks in a smooth area may have no embeddable coefficients at all. Under a constant embedding rate, a large block size reduces the total number of bits we can embed and wastes a large amount of embedding capability. Hence approaches that can embed more data via a smaller block size are more desirable. In the following, we will discuss two ways to achieve this, namely, backup embedding and shuffling.

4.2.1 Backup Embedding

The idea of backup embedding is to embed the same data in multiple locations. The locations are identified deterministically by a rule known to both the embedder and the detector. Illustrated in Fig. 4.4 is a special case where we not only embed one bit in (i, j) block, but also put a backup copy in $(i, j + \frac{H}{2})$ block that is half way apart, where H is the number of blocks along the vertical direction. We shall call this *symmetric backup*. Assuming a block consists of q components and the number of locations holding the same information is L, the equivalent block size for embedding one bit is Lq, implying that an increase in L will reduce the total number of bits being embedded. The difference between backup embedding with L locations and simply increasing the block size by L times is that the former one is more likely to allow most bits being embedded. This is because if the multiple locations are sufficiently further apart from each other, the probability of each location being smooth tends to be independent of each other, therefore the probability that all of them are smooth is greatly reduced, enabling to hide more data than the approach that simply enlarges the block size [1]. With a proper choice of the location patterns, the independence condition is more likely to hold. The shuffling approach discussed next can be viewed as a generalization of backup embedding where each block is reduced to contain only one coefficient and the multiple locations are specified by a permutation function.

[1]Similar ideas to backup embedding has been used in data hiding systems that use the embedded data to recover corrupted regions [41, 88].

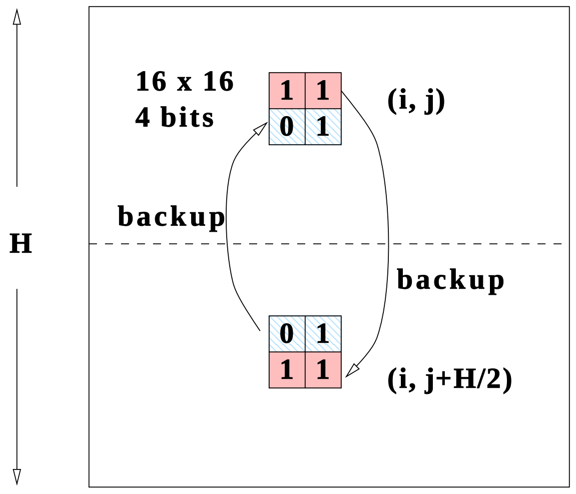


Figure 4.4: Symmetric backup embedding for handling smooth region. One bit is embedded in a block and its companion block half image apart. The effective embedding rate is two bits per 16×16 macroblock which involves four blocks.

4.2.2 Equalizing Embedding Capacity Via Shuffling

The effectiveness of simple backup embedding like the symmetric backup shown in Fig. 4.4 may still be dependent on the structure of the host image. For example, an entire column or row of an image may be smooth and therefore data cannot be hidden in that column or row. To achieve statistical independency with respect to the image structure, we consider shuffling the coefficients, where the shuffle can be viewed as a bijective mapping of the coefficient indexes $f : \{1, 2, ..., S\} \rightarrow \{1, 2, ..., S\}$, where S is the total number of image coefficients. As illustrated in Fig. 4.5, an original sequence of coefficients is shuffled to obtain the second sequence. After the embedding process is performed in the shuffled domain, the coefficients are inversely shuffled, which results in the fourth sequence. The same shuffling needs to be performed at detection.

Shuffling can be considered as a general permutation covering the following cases:

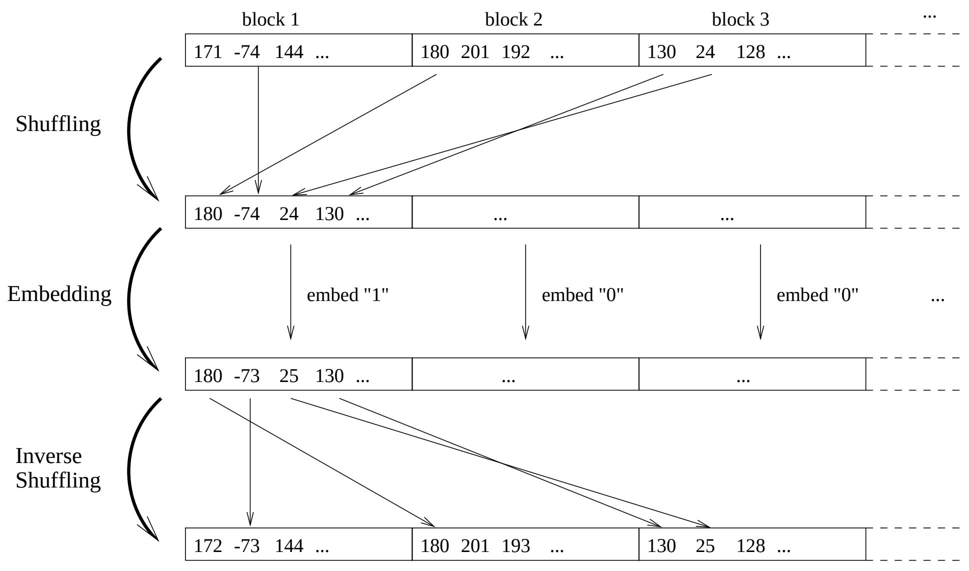


Figure 4.5: Incorporate shuffling with an embedding mechanism

(1) random permutation, whose effectiveness can be studied analytically and is independent of the specific distribution of embeddable elements; the permutation can be performed among all elements (*complete random permutation*) or among elements in the same frequency band (*in-band random permutation*), (2) non-random interleaving or pairing, for example, to embed the i-th bit of a total of B bits to $\{kB + i\}$-th coefficients, where k is a positive integer. We will present our analysis for the case of complete random permutation, where all permutations are equiprobable hence the probability of each permutation is $1/S!$. The focus is on the distribution of embeddable coefficients after a random permutation is performed on all block-DCT coefficients.

Analysis

As defined earlier, $\frac{m_r}{N}$ is the fraction of segments having r embeddable coefficients. Because computing the marginal distribution of $P(m_r)$ from the joint probability distribution of the histogram $\{m_0, m_1, ..., m_{\frac{S}{N}}\}$ may experience high complexity unless $q = \frac{S}{N}$ is very small, we adopt *moment approach* [3] to study the mean and variance of each normalized bin $\frac{m_r}{N}$ of the histogram. For each bin m_r with $r = 0, ..., q$, we have obtained

$$E\left[\frac{m_r}{N}\right] = \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} \tag{4.1}$$

$$Var\left[\frac{m_r}{N}\right] = \frac{1}{N} \cdot \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} + \left(1 - \frac{1}{N}\right) \frac{\binom{q}{r}\binom{q}{r}\binom{S-2q}{n-2r}}{\binom{S}{n}} - \left[\frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}}\right]^2 \tag{4.2}$$

The detailed derivation and further analysis of these two quantities can be found in Appendix 4.7. It can be seen that the distribution of embeddable coefficients per segments after shuffling depends only on the global parameters p (percentage of embeddable coefficients) and q (segment size taken after shuffling), and does not depend on the detailed distribution before shuffling. Other variations besides random shuffling may also be adopted, such as performing shuffling/ interleaving on smaller groups, with a tradeoff between the effectiveness of equalization and such cost as memory usage and delay.

Simulation and Verification

We use the Alexander Hall image in Fig. 4.1 as an example to illustrate the effectiveness of shuffling. Among a total of $S = 640 \times 432$ coefficients, $p = 15.49\%$ of all coefficients are embeddable. The block size of $q = 8 \times 8 = 64$ is used for the cases both with and without shuffling. By Eq. 4.1 and Eq. 4.2:

$$E\left[\frac{m_0}{N}\right] \approx 0.002\%, \quad Var\left[\frac{m_0}{N}\right] \approx 4.85 \times 10^{-9}$$

which indicates that the average fraction of segments with no embeddable coefficients is reduced by 4 orders of magnitude, from the original 20% to 0.002%. The expected number of blocks with no embeddable coefficient after shuffling is only $0.002\% \times N = 0.002\% \times 640 \times 432/64 \approx 0.086$. The very small value of $Var\left[\frac{m_0}{N}\right]$ suggests that very few shuffles among the $S!$ possibilities will cause the histogram significantly deviate from the mean.

To verify the analysis, we perform 1000 times of random permutation on the image shown in Fig. 4.1, then compute the mean and variance of each bin of the histogram $\{\frac{m_r}{N}\}$. The dashed line of Fig. 4.3 is the histogram of $\{\frac{m_r}{N}\}$ before the shuffling, showing that 20% of the blocks have no embeddable coefficients. The solid line of Fig. 4.3 is the average fraction of blocks having a given number of embeddable coefficients together, and the dotted line shows one standard deviation away from the average. The simulation shows that after shuffling, most blocks have between 5 and 15 embeddable coefficients, and the number of blocks which have no embeddable coefficients has been significantly reduced. The figure also shows that the result of theoretical analysis and that of 1000-time simulation match each other very well. As discussed as appendix in Section 4.7, the averaged histogram $\{E\left(\frac{m_i}{N}\right)\}$ is an arch-shaped hypergeometric distribution function and can be approximated by binomial, Poisson, and normal distributions with mean pq. Notice that q does not have to be the same block size as that of the transform (8×8). It should be chosen to get the desired mean, pq, of the histogram $\{E\left(\frac{m_i}{N}\right)\}$, and to ensure the left tail of the histogram is smaller than a desired bound. For images that contain a large fraction of embeddable coefficients (i.e., large p), the segment size can be chosen to be small;

while for images in which most regions are smooth, the segment size should be large enough to ensure enough decay at the left tail.

Since shuffling significantly reduces the segments which have no embeddable coefficient at all, we can embed one bit in each shuffled segment. This enables us to embed the bits which are intended to but are not able to be inserted in smooth regions without shuffling. The capability achieved by shuffling to embed in smooth region is in a *logical sense*. No bits are actually inserted into the smooth regions since this is forbidden by the invisibility constraint. Embeddable coefficients from complex regions are dynamically allocated to hold the data which are intended to be put in smooth regions yet without the need of sending much side information. This also indicates that as long as the criterion for identifying embeddable coefficients is unchanged, adding shuffling step will not compromise the perceptual quality.

The equalization of embedding capacity via shuffling requires little additional side information. The embedding and extraction processes need only agree on the segment size and the shuffle table that can be generated from a key. This side information also enhances security because the detection of the embedded data requires the knowledge of the shuffle table or the key for generating the table. Using shuffling to add uncertainty is a common practice in cryptographic and security systems.

4.2.3 Practical Considerations

In this section, we discuss a few practical considerations associated with shuffling.

Generating Shuffle Table A shuffling table can be generated from a key and the generation is with linear complexity proportional to the number of entries. An algorithm of this kind is discussed as appendix in Section 4.6.

Handling Bad Shuffle While our analysis shows that the statistical effectiveness of random shuffling assures very small probability for getting a bad random shuffle, it is still possible that a particular shuffle is unable to efficiently equalize the embedding capacity for a specific image. The problem of bad shuffles can be handled by the two approaches.

The first approach addresses the problem that a specific instance of random shuffle could be good for most images and bad for some images. Notice that an image-independent shuffle is desirable for marking many images without the need of conveying much additional side information of the shuffling. We propose to generate a set of candidate shuffles which are significantly different from each others, then select and use the best shuffle when hiding data in a given image. The probability that all shuffles are bad for the image shall decrease exponentially from the already low probability in the single shuffle case. In practice, we may select two shuffles, a primary one and a secondary one. We use the primary one most of the time, and only switch to use the secondary one when the primary one is not suitable for a given image. An important issue is how to let a detector know what shuffle is used for each image. This is the additional side information to be conveyed to a detector. Because of some similarities between this and the side information in variable rate embedding, we postpone the discussion till Section 4.3.

The second approach targets at the case that even for good shuffles, there could still be some blocks with no flippable pixel. The bits to be embedded in those blocks can be treated by a detector as *erasure bits*. These erasure bits can be handled by encoding the data via error correction coding [8] before embedding them in an image. Because the number of blocks with no flippable pixel is very small (hence the small number of erasure bits), a little error correction capability would be sufficient, while it has to be much larger if without shuffling.

Adaptive Block Size The block size q gives a measure of how many bits will be embedded in an image and is determined by p, the percentage of embeddable pixels: if p is small, the block size has to be large to ensure enough flippable pixels are included in each shuffled block. Because the percentage of flippable pixels can be quite different from image to image, it is desirable to choose the block size adaptively according to the content and the type of each image. Similar to the handling of bad shuffles, a key problem for using adaptive block size is how to convey such side information to a detector. We will discuss this in Section 4.3.

4.2.4 Discussion

We would like to comment on a few issues related to shuffling. First, we mentioned earlier that uneven embedding capacity occurs when multiple bits are embedded in non-overlapped segments in the sample domain of a perceptual source. As discussed in Section 3.4, this kind of insertion of secondary data is analogous to TDMA in communication: multimedia source is partitioned into segments (spatially for image, temporally for audio, etc) and one or more bits are embedded in each segment. An alternative way is to embed multiple bits using the CDMA approach, possibly combined with spread spectrum embedding. For example, we can insert into multimedia source two or more random sequences with same support, each of which is modulated by one bit from the secondary data. CDMA-type modulation requires that both the embedder and the detector keep copies of all random sequences and that the sequences are orthogonal or approximately orthogonal to each others. On the contrary, the orthogonality is much easily fulfilled by TDMA-type modulation with little side information. The simplicity in implementation contributes to the wide adoption of TDMA-type modulation, especially for the high rate embedding via the Type-II mechanisms discussed in Chapter 3. The shuffling approach proposed in this chapter serves as a tool to equalize the uneven embedding capacity problem that is generally suffered by TDMA-type modulation. In addition, TDMA-type modulation is more suitable than CDMA for hiding data in sequential sources like audio and video.

Second, shuffling may increase the sensitivity against intentional attacks that target at rendering watermark undetectable. For example, a quite different sequence may be obtained after shuffling if the image is shifted, rotated, or scaled, which implies the bit error rate can be very high at detection. While this is a potential shortcoming for some applications, it is not a real concern for applications in which users can benefit from the hidden data and/or are not willing to make the hidden data undetectable, such as using watermark to detect tampering or to convey bilingual audio tracks. Furthermore, the robustness against sample dropping, warping, scaling, and other distortion/ attack is a major challenge for robust data hiding [61, 62, 169], regardless of whether the shuffling is performed or not.

4.3 Variable Embedding Rate (VER)

In this section, we explore issues associated with variable embedding rate. Compared with CER, VER may enable embedding more data by better utilizing the embedding capability. However, the side information regarding how many bits are embedded in each segment must be conveyed. This indicates that the gain of VER over CER is significant under the following two conditions: (1) the total number of bits that can be hidden should exceed the amount of side information for almost all segments – this ensures there is sufficient room to convey side information; (2) the average overhead for side information is relatively small compared with the average embedding capacity per segment. A key issue is how to tell a detector the number of bits being embedded in each segment. More generally, we would like to explore mechanisms to convey additional side information to a detector so as to facilitate the extractions of the embedded data payload. The side information could be the number of bits being embedded in each segment, or could be an index signaling which shuffle and/or what segment size is used in the constant-rate embedding discussed in Section 4.2.3. The latter scenario also indicates a connection between CER and VER: while for a set of segments such as all the blocks of an image, we may use CER for each segment, the parameter settings like the segment size could vary for different sets of segments (e.g., different images) due to the fact that the embedding capacity of different sets of segments may vary significantly. On the set level, it is more suitable to apply VER rather than CER because the two conditions described above are likely to hold.

4.3.1 Conveying Additional Side Information

The additional side information can be conveyed using either the same embedding mechanism as that for the user payload or different embedding mechanisms. In both cases, the side information consumes part of the energy by which the host image can be changed imperceptibly. The difference lies only on the specific way to achieve orthogonality, similar to the discussion of TDMA and CDMA multiplexing in Section 3.4.

Considering first the embedding of side information via the same embeddding mechanism as that for the user payload, we choose a strategy similar to the training sequence in classic communication. That is, part of the embedded data (such as the first several bits) are pre-determined and/or be designed to be self-verifiable. The self-verifiability can be obtained by hash function (message digest function), or certain error detection/correction codes. For example, in order to let a detector know which shuffle is used for each image, one may choose the first 15 bits of hidden data to be a predetermined label, or a label plus its hash. The detector tries to decode the hidden data using all candidate shuffles. The shuffle that accurately decodes the first 15 bits is identified as the one used by the embedder. When we decode the embedded data using a shuffle table that is significantly different from the one used by the embedder, the decoded bits are approximately independent of each other and equiprobable to be "1" or "0". The probability of matching the pre-determined pattern or passing the verification test decreases exponentially with the number of bits used for identifying which shuffle is used. Similarly, to let decoder know what block size is used by the embedding process, we can select a finite number of candidate block sizes, and choose a suitable one to embed data. Again, part of the embedded data is per-determined or self-verifiable. A detector will try out candidate block sizes and find the one that successfully passes the verification. To limit the searching complexity, we may have one primary block size which is suitable for a large number of images, and a couple of secondary sizes that are larger or smaller than the primary one and are used for handling special images.

For grayscale/color images and videos, it is possible to find some other domains or mechanisms to hide the additional side information. These mechanisms are often orthogonal to that for embedding the user payload to avoid interference. The popular spread spectrum additive embedding is one feasible approach for this purpose because their statistical properties make it easy to generate additional "watermarks" orthogonal or approximately orthogonal to the watermarks for the user payload. In addition, spread spectrum embedding has been proven to be robust to a number of distortions. The robustness is necessary since the accuracy in determining such information as how many bits are embedded and what shuffle is used is crucial for correctly extracting the user payload. The watermarks for conveying side information share part of the total energy that can be allocated to all embedded data while preserving perceptual quality. Allocating more energy to the side information gives higher robustness in extracting them but reduces the amount of user payload. It is desirable to both limit the amount of side information and use energy efficient modulation techniques to embed multiple bits of side information. Several commonly used energy-efficient modulation techniques such as orthogonal and biorthogonal modulation have been discussed and compared in Sec. 3.4.

4.4 Outline of Examples

Several design examples in the following chapters will be used to explain how the approaches described in the previous sections are used in designing practical watermarking algorithms. Experimental results are reported to demonstrate the effectiveness of our proposed approaches. More specifically, the data hiding in binary images (Chapter 5) and the watermark-based authentication for grayscale/color images (Chapter 7) show the effectiveness of shuffling in equalizing uneven embedding capacity from region to region. The multi-level data hiding in video (Chapter 6) is a prototype design incorporating almost all solutions we discussed in Part-I. It adopts CER within a video frame and uses VER from frame to frame with adaptive embedding rate.

4.5 Chapter Summary

In summary, this chapter addresses the problem of unevenly distributed embedding capacity and proposes a set of feasible solutions. Depending on the overhead relative to the total embedding capacity, we choose between a constant embedding rate and a variable embedding rate. For a constant embedding rate, shuffling is proposed to dynamically equalize the distribution of embeddable coefficients, allowing for the hiding of more data. We demonstrated, via analysis and experiments, that shuffling is effective and is applicable to many data hiding schemes and applications. For variable embedding rate, we discussed how to convey the additional side information to a detector to ensure the correct detection of the embedded user payload. Three design examples and experimental results will be presented in the following chapters to illustrate the handling of uneven embedding capacity in practical data hiding systems.

4.6 Appendix - Generating Shuffling Table From A Key

The generation of shuffling table relies on random number generator. The security strength of the generator determines that of the shuffling table. For efficient implementation, we adopt pseudo random number generator with key(s) or seed(s) determining its output sequence. A simple way of generating an N-entry shuffling table is to sort N random numbers and to use the sorting index to construct the table. This approach is used by Mathworks for its Matlab function "randperm" [24]. More specifically, let $\{r_k\}$ denote the sequence of N random numbers ($k = 1 \sim N$), and

$\{r'_k\}$ denote the sorted sequence with $r'_k = r_{i_k}$ and $r'_{k_1} \leq r'_{k_2}$ for any $k_1, k_2 \in \{1, ..., N\}$ such that $k_1 < k_2$. The mapping T of the shuffling table is then obtained as $T(k) = i_k$. Since the best sorting we can get has the complexity of $O(N \log N)$, the complexity of this algorithm for generating the shuffling table is $O(N \log N)$.

A better algorithm quantizes the random number with monotonically increasing step sizes and makes use of carefully selected data structure, reducing the complexity to $O(N)$. The basic idea is as follows: we start with a set $S_1 = \{1, ..., N\}$, and generate one random number per step. At the k^{th} step, we uniformly partition the output range of random number generator into $N - k + 1$ non-overlapped segments; if the random number generated at this step falls in the j_k^{th} segment, we pick the j_k^{th} element in the set S_k, fill the value in the k^{th} entry of shuffling table, and cross out the element from the set, denoting the new set of $N - k$ element as S_{k-1}. We continue the process until the shuffling table is fully filled. To achieve linear complexity and to allow in-place storage (i.e., no additional storage is needed for every new S_k), we implement the set S_k based on hashing and swapping the elements in an array. The detailed algorithm is summarized below:

(1) Initialization. Set up two N-element array $T[i]$ and $s[i]$ ($i = 1 \sim N$) for storing shuffling table and for keeping the above mentioned set S_k, respectively. Let $s[i] = i$ and the step index $k = 1$.

(2) Generate a random number r_k, and denote j_k as the index of the segment that it falls in. More specifically, if the range of the random number is $[L, U)$, then

$$j_k = \left\lfloor \frac{r_k - L}{U - L} \times (N - k + 1) \right\rfloor + 1.$$

(3) $T[k] = s[j_k]$, then swap the content of $s[N - k + 1]$ and $s[j_k]$.

(4) $k = k + 1$. If $k \geq N$, let $T[N] = s[1]$, then stop; otherwise, go back to (2).

Notice that after the above process, $T[k] = s[N-k+1]$, implying that $s[\cdot]$ contains an inversely ordered version of $T[\cdot]$ hence even the array $T[\cdot]$ is not needed. The following example further illustrates the algorithm, assuming the output of random number generator is within $[0, 1)$ and $N = 10$.

$$\begin{array}{lcl} & & s[\cdot] = [1,2,3,4,5,6,7,8,9,10] \\ r_1 = 0.46 \rightarrow j_1 = 5 & \longrightarrow & s[\cdot] = [1,2,3,4,10,6,7,8,9,\underline{5}], T[1] = 5 \\ r_2 = 0.70 \rightarrow j_2 = 7 & \longrightarrow & s[\cdot] = [1,2,3,4,10,6,9,8,\underline{7,5}], T[2] = 7 \\ r_3 = 0.51 \rightarrow j_3 = 5 & \longrightarrow & s[\cdot] = [1,2,3,4,8,6,7,\underline{10,7,5}], T[3] = 10 \\ & \cdots \quad \cdots & \end{array}$$

4.7 Appendix - Analysis of Shuffling

The detailed analytic study of shuffling introduced in Section 4.2.2 is presented as follows. For the simplicity of discussion, we formulated the problem of analyzing the distribution of embeddable coefficients after shuffling in terms of a ball game illustrated in Fig. 4.6. Consider we have a total of S balls, and a fraction p of them or a total of $n = pS$ balls are *blue* which represent the embeddable coefficients in our data hiding problem. The balls are to be placed in S holes randomly with one ball for each hole. Further, every q holes are grouped together to form a cluster, and the total number of clusters is $N = S/q$. It is important to note that S is a very large number and $q << S$. For simplicity, we assume n and N are integers. We are interested in studying m_r/N, the percentage of clusters each of which has exactly r blue balls, for $r = 0, ..., q$. Since the blue balls are the center of focus, we can view the game as putting all blue balls in a bag, then take out one ball at a time and randomly throw it to the unfilled holes. The ball has equal probability falling in each unfilled holes. The game continues until all blue balls are thrown.

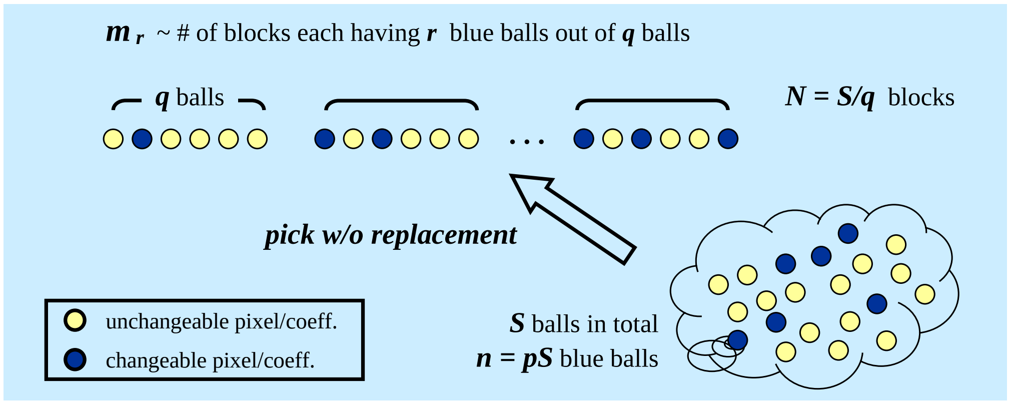

m_r ~ # of blocks each having r blue balls out of q balls
q balls
$N = S/q$ blocks
pick w/o replacement
● unchangeable pixel/coeff.
● changeable pixel/coeff.
S balls in total
$n = pS$ blue balls

Figure 4.6: Illustration of random shuffling in terms of a ball game.

4.7.1 Joint Probability of Histogram

Traditionally, we start with the joint probability of the histogram $\{m_0, m_1, ..., m_q\}$, which can be found as

$$P\left([m_0, ...m_q] = [y_0, ..., y_q]\right) \quad = \quad \frac{\left[\binom{q}{0}\right]^{y_0} ... \left[\binom{q}{q}\right]^{y_q} \times \frac{N!}{y_0!...y_q!}}{\binom{S}{n}} \tag{4.3}$$

The denominator $\binom{S}{n}$ is the number of ways to throw n balls into S holes, while the numerator indicates how many of them result in the same histogram of $[y_0, ..., y_q]$. While it is possible to sum up the distribution of histogram under the constraints

$$\begin{cases} \sum_k y_k = N \\ \sum_k k \cdot y_k = n \end{cases} \tag{4.4}$$

to get the marginal distribution of each bin $P(m_r)$, the computation may involve high complexity unless q is very small. For this reason, we adopt the *moment approach* [3] suggested by Kolchin *et al.* to study the mean and variance of each normalized bin of the histogram $\{m_r/N\}$ [2].

4.7.2 Mean and Variance of Each Bin

Considering the bin of m_r where r is an integer between 0 and q, we perform the following decomposition

$$m_r = \theta_{r,1} + \theta_{r,2} + \ldots + \theta_{r,N} \tag{4.5}$$

where $\theta_{r,i}$ is an indicator function defined as

$$\theta_{r,i} = \begin{cases} 1 & \text{if } i^{th} \text{ cluster has } r \text{ balls} \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Computing the mean of $\theta_{r,i}$ is equivalent to getting the probability that the i^{th} cluster has r balls, i.e.,

$$\begin{aligned} E[\theta_{r,i}] &= P[\theta_{r,i} = 1] && (4.7) \\ &= \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} && (4.8) \end{aligned}$$

Since the mean of θ_i is independent of i, we have

$$E\left[\frac{m_r}{N}\right] = E\left[\frac{\sum_{i=1}^{N} \theta_{r,i}}{N}\right] = E(\theta_{r,1}) = \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} \tag{4.9}$$

This quantity indicates the average portion of clusters each having exactly r balls. The variance is obtained by observing the following relationship:

$$\theta_{r,i}{}^2 = \theta_{r,i} \tag{4.10}$$

[2]Interested readers may refer to [3] for the analysis strategies and results of several related random allocation problems with simpler conditions.

from which we obtain

$$m_r{}^2 = \sum_{k=1}^{N} \theta_{r,k}{}^2 + \sum_{i \neq j} \theta_{r,i}\theta_{r,j} = \sum_{k=1}^{N} \theta_{r,k} + \sum_{i \neq j} \theta_{r,i}\theta_{r,j} = m_r + \sum_{i \neq j} \theta_{r,i}\theta_{r,j} \quad (4.11)$$

For $i \neq j$,

$$\begin{aligned} E[\theta_{r,i}\theta_{r,j}] &= P(\theta_{r,i} = \theta_{r,j} = 1) && (4.12) \\ &= \frac{\binom{q}{r}\binom{q}{r}\binom{S-2q}{n-2r}}{\binom{S}{n}} && (4.13) \end{aligned}$$

indicating that the expected value of $\theta_{r,i}\theta_{r,j}$ is the probability that two different clusters, the i^{th} and the j^{th}, each has r blue balls. Since this probability is independent of i and j, we have

$$E[m_r{}^2] = E[m_r] + E[\sum_{i \neq j} \theta_{r,i}\theta_{r,j}] = E[m_r] + N(N-1)E[\theta_{r,1}\theta_{r,2}] \quad (4.14)$$

Therefore,

$$\begin{aligned} Var\left[\frac{m_r}{N}\right] &= \frac{1}{N^2}E[m_r{}^2] - \left[E\left(\frac{m_r}{N}\right)\right]^2 && (4.15) \\ &= \frac{1}{N}E\left[\frac{m_r}{N}\right] + \left(1 - \frac{1}{N}\right)E[\theta_{r,1}\theta_{r,2}] - \left[E\left(\frac{m_r}{N}\right)\right]^2 && (4.16) \\ &= \frac{1}{N} \cdot \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} + \left(1 - \frac{1}{N}\right)\frac{\binom{q}{r}\binom{q}{r}\binom{S-2q}{n-2r}}{\binom{S}{n}} - \left[\frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}}\right]^2 && (4.17) \end{aligned}$$

In summary, the mean and variance of the r^{th} bin is

$$\boxed{\begin{aligned} E\left[\frac{m_r}{N}\right] &= \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} \\ Var\left[\frac{m_r}{N}\right] &= \frac{1}{N} \cdot \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} + \left(1 - \frac{1}{N}\right)\frac{\binom{q}{r}\binom{q}{r}\binom{S-2q}{n-2r}}{\binom{S}{n}} - \left[\frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}}\right]^2 \end{aligned}} \quad (4.18)$$

We have presented the simulation result in the main text (Fig. 4.3), showing that the analytic study and the simulation on the mean and variance match very well.

4.7.3 More About $E[\frac{m_r}{N}]$ - A Hypergeometric Distribution

The relation of $E[\frac{m_r}{N}]$ with respect to r, which in our data hiding problem describes the spread of embeddable coefficients after shuffling, is what we are mostly interested in. We noted that the distribution $P[\theta_{r,i} = 1]$ (in Eq. 4.7) which the mean is equal to is known as a *hypergeometric* distribution [1]. Given a population of S balls with n of them are blue, a hypergeometric distribution $H(r; S, n, q)$ describes the probability of getting r blue balls among a sampling of q balls, where the sampling is performed without replacement. Denoting a random variable following this distribution as Y, we have seen that its probability mass function takes the form of

$$P(Y = r) = H(r; S, n, q) = \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} \tag{4.19}$$

for $r = 0, ..., \min(q, n)$. In our problem, r takes values from 0 to q as $q << n$. Noticing the following relationship

$$\sum_{r=0}^{q} \binom{q}{r}\binom{S-q}{n-r} = \binom{S}{n} \tag{4.20}$$

from $\sum_{r=0}^{q} P(Y = r) = 1$, we compute the mean of Y

$$E[Y] = \frac{1}{\binom{S}{n}} \sum_{r=1}^{q} r \cdot \binom{q}{r}\binom{S-q}{n-r} = q \cdot \frac{\binom{S-1}{n-1}}{\binom{S}{n}} = p \cdot q \tag{4.21}$$

where p is the portion of blue balls in the population and $p = n/S$. Similarly, we obtain the second moment of Y

$$\begin{aligned} E[Y^2] &= \frac{1}{\binom{S}{n}} \sum_{r=1}^{q} r^2 \cdot \binom{q}{r}\binom{S-q}{n-r} && (4.22) \\ &= \frac{1}{\binom{S}{n}} \left[\sum_{r=2}^{q} r(r-1) \cdot \binom{q}{r}\binom{S-q}{n-r} + \sum_{r=1}^{q} r \cdot \binom{q}{r}\binom{S-q}{n-r}\right] && (4.23) \\ &= q(q-1) \cdot \frac{\binom{S-2}{n-2}}{\binom{S}{n}} + E[Y] && (4.24) \end{aligned}$$

$$= \quad p \cdot q \cdot \left[\frac{(n-1)(q-1)}{S-1} + 1 \right] \tag{4.25}$$

from which the variance of Y can be computed

$$Var[Y] = E[Y^2] - (E[Y])^2 = p \cdot q \cdot \frac{(S-q)(1-p)}{S-1} \tag{4.26}$$

To study the relations of $H(r; S, n, q)$ with respect to r, we simplify the notation as H_r and study the ratio

$$\frac{H_r}{H_{r-1}} = \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{q}{r-1}\binom{S-q}{n-r+1}} \tag{4.27}$$

$$= \frac{(q-r+1)\cdot(n-r+1)}{r\cdot(S-q-n+r)} \tag{4.28}$$

$$= 1 + \frac{(q+1)(n+1) - r(S+2)}{r(S-q-n+r)} \tag{4.29}$$

Defining r_0 as

$$r_0 = \frac{(q+1)(n+1)}{(S+2)} = p \cdot q + p + \frac{(q+1)(1-2p)}{S+2} \tag{4.30}$$

we have

$$\begin{cases} H_r > H_{r-1} & \text{if } r < r_0 \\ H_r < H_{r-1} & \text{if } r > r_0 \\ H_r = H_{r-1} & \text{if } r_0 \in \mathcal{Z} \text{ and } r = r_0 \end{cases} \tag{4.31}$$

This indicates that with r varying from 0 to q, H_r first monotonically increases then monotonically decreases, achieving its maximum value at $r = \lfloor r_0 \rfloor$ except that if r_0 is an integer, the maximum values is achieved at both r_0 and $(r_0 - 1)$. Such a relation has been confirmed by a numerical evaluation of H_r shown in Fig. 4.3 and Fig. 4.7 with the parameter setting taken from the image in Fig. 4.7. In this case, $S = 640 \times 432$, $q = 64$, $p = 15.49\%$, so $r_0 = 10.0687$, implying H_r reaches maximum at $r = 10$. This is the same as what we have observed in the numerical evaluation.

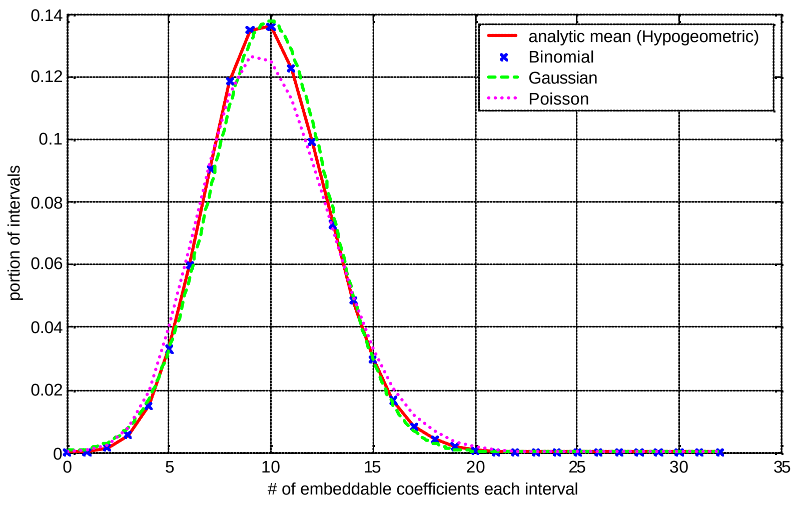


Figure 4.7: Various approximations to the hypergeometric distribution. Experiments are performed on the Alexander Hall image (Fig. 4.1).

4.7.4 Approximations for Hypergeometric Distribution

In [1], Feller pointed out the close relations among the hypergeometric distribution, the binomial distribution, the Poisson distribution, and the normal (gaussian) distribution. Their probability mass functions or probability density function (for the normal distribution) are summarized as follows:

$$\begin{cases} \text{hypergeometric} & H(r;S,n,q) = \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} \\ \text{binomial} & b(r;q,p) = \binom{q}{r} p^r (1-p)^{q-r} \\ \text{Poisson} & P(r;\lambda) = \frac{\lambda^r}{r!} e^{-\lambda} \\ \text{normal} & f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \end{cases} \tag{4.32}$$

More specifically, as the population S goes to infinity, $H(r; S, n, q)$ is approximated by a binomial distribution $b(r; q, p)$, as long as q is much smaller than S so that $\lim_{S\to\infty} \frac{q}{S} = 0$. This approximation is shown as follows:

$$\begin{aligned}
H(r; S, n, q) &= \frac{\binom{q}{r}\binom{S-q}{n-r}}{\binom{S}{n}} && (4.33)\\
&= \binom{q}{r}\prod_{j=0}^{r-1}\left(\frac{n-j}{S-j}\right)\prod_{k=0}^{q-r-1}\left(\frac{S-n-k}{S-r-k}\right) && (4.34)\\
&= \binom{q}{r}\prod_{j=0}^{r-1}\left(\frac{\frac{n}{S}-\frac{j}{S}}{1-\frac{j}{S}}\right)\prod_{k=0}^{q-r-1}\left(\frac{1-\frac{n}{S}-\frac{k}{S}}{1-\frac{r+k}{S}}\right) && (4.35)\\
&\approx \binom{q}{r}p^r(1-p)^{q-r} && (4.36)\\
&= b(r; q, p) && (4.37)
\end{aligned}$$

Intuitively, this means that when taking a small number of samples from a large population, the statistical outcomes of sampling without replacement is approximately the same as that with replacement (or equivalently, from an infinite population). Since the above-mentioned conditions hold in our data hiding problem, we have obtained a very good binomial approximation to $E[m_r/N]$, as shown in Fig. 4.7.

Because the behavior of binomial distribution is well studied [1], the binomial approximation enables our making use of many existing results to understand the behavior of random shuffling. For large q and small p, a binomial distribution $b(r; q, p)$ can be approximated by a Poisson distribution with mean $\lambda = p \cdot q$. A binomial distribution can also be approximated by a normal distribution (with mean $p \cdot q$ and variance $p(1-p)q$) for large q and small r. While the q in our data hiding problem is generally not very large, the Poisson approximation and the gaussian approximation are still pretty good numerically. Fig. 4.7 shows the hypergeometric distribution with the parameters taken according to the Alexander Hall image (Fig. 4.1) and the corresponding binomial, Poisson, and normal approximations. In addition, the tail of a binomial distribution is known to be bounded by

$$\begin{cases} \sum_{k=0}^{r} b(k;q,p) \leq \frac{p(q-r)}{(pq-r)^2} & \text{if } r < pq \\ \\ \sum_{k=r}^{\infty} b(k;q,p) \leq \frac{r(1-p)}{(r-pq)^2} & \text{if } r > pq \end{cases} \tag{4.38}$$

4.7.5 More About $Var[\frac{m_r}{N}]$

The joint probability $P(\theta_{r,i} = \theta_{r,j} = 1)$ which is an important term for computing the variance can also be approximated by sampling with replacement (or equivalently, from an infinitely large population). That is,

$$P(\theta_{r,i} = \theta_{r,j} = 1) \approx [b(r;q,p)]^2 = \left[\binom{q}{r} p^r (1-p)^{q-r}\right]^2 \tag{4.39}$$

Therefore,

$$\begin{aligned} Var[\frac{m_r}{N}] &\approx \frac{1}{N} b(r;q,p) + \Big(1 - \frac{1}{N}\Big)[b(r;q,p)]^2 - [b(r;q,p)]^2 && (4.40) \\ &= \frac{1}{N} \cdot b(r;q,p) \cdot [1 - b(r;q,p)] && (4.41) \end{aligned}$$

This approximation takes the form of $f(x) = C \cdot x(1-x)$ with x being replaced by $b(r;q,p)$ and C is a constant. The function $f(x)$ describes an arch shaped curve in an x-y Cartesian coordinate with the maximal value sits at $x = 1/2$. Because the binomial distribution $b(r;q,p)$ has small values over its support except for very small q, $b(r;q,p)$ is generally smaller than 1/2 for all r. Therefore, $Var[\frac{m_r}{N}]$ is monotonically increasing with $b(r;q,p)$ for all practical cases. This implies that the trend of $Var[\frac{m_r}{N}]$ with respect to r is the same as that of $b(r;q,p)$ which has an arch shape with the maximum value around pq. Our numerical evaluation shown in Fig. 4.8 confirms this analysis. Note that special care should be taken in the numerical evaluation of $Var[\frac{m_r}{N}]$ because it involves taking the difference between two comparable terms (Eq. 4.18).

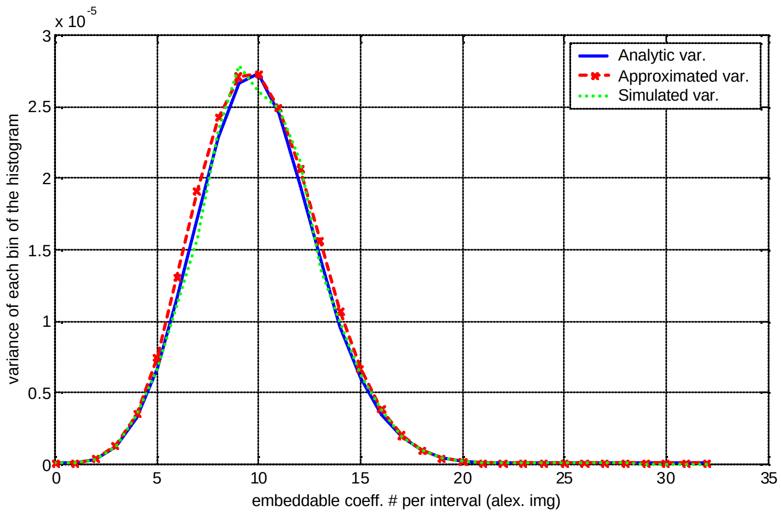


Figure 4.8: Comparison of analytic, approximated, and simulated variance of the histogram of the embeddable coefficients after shuffling. Experiments are performed on the Alexander Hall image (Fig. 4.1).

Part II

Algorithm and System Designs

Chapter 5

Data Hiding in Binary Images

5.1 Introduction and prior art

An increasingly large number of digital binary images have been used in everyday life. Handwritten signatures captured by electronic signing pads are digitally stored and are being used as the records for credit card payment by many department stores in the U.S. and for parcel delivery by major courier services such as the United Parcel Service (UPS). Word processing software like Microsoft Word allows a user to store his/her signature in a binary image file for inclusion at specified locations of a document. The documents signed in such a way can be sent directly to a fax machine or be distributed across a network. The unauthorized use of a signature, such as copying it onto an unauthorized payment, is becoming a big concern. In addition, a variety of important documents, such as social security records, insurance information, and financial documents, have also been digitized and stored. Because it is easy to copy and edit digital images via software tools, the annotation and authentication of binary images as well as the detection of tampering are very important. This chapter discusses data hiding techniques for these purposes as an alternative to or in conjunction with the cryptographic authentication approach. While we expect the embedded data to have some robustness against minor distortion and preferably to withstand printing and scanning, the robustness of embedded data against intentional removal or other obliteration is not a primary concern because there is little incentive to do so in the targeted applications of annotation and authentication.

Most prior works on image data hiding are for color or grayscale images in which the pixels may take on a wide range of values. For those images, changing pixel values by a small amount is generally unnoticeable under normal viewing conditions. This property of human visual system plays a key role in watermarking of perceptual media data [44, 46]. For images in which the pixels take on only a limited number of values, hiding data without causing visible artifacts becomes more difficult. In particular, flipping white or black pixels that are not on the boundary is likely to introduce visible artifacts in binary images. Before we present our solutions to the challenging issues of hiding data in binary images, we shall give a brief review of the prior art.

Several methods for hiding data in specific types of binary images have been proposed in literature. Matsui *et al* [106] embedded information in dithered images by manipulating the dithering patterns and in fax images by manipulating the run-lengths. Maxemchuk *et al* [108] changed line spacing and character spacing to embed information in textual images for bulk electronic publications. These approaches cannot be easily extended to other binary images and the amount of data that can be hidden is limited. In [107], Koch and Zhao proposed a data hiding algorithm which enforces the ratio of black vs. white pixels in a block to be larger or smaller than 1. Although the algorithm aims at robustly hiding information in binary image, it is not robust enough to tolerate many distortions/attacks, neither is it secure enough to be directly applied for authentication or other fragile use. Only a limited number of bits can be embedded because the particular enforcing approach has difficulty in dealing with blocks that have low or high percentage of black pixels. In spite of these weaknesses, their idea of enforcing properties of a group of pixels via the local manipulation of a small number of pixels can be extended as a general framework of data embedding. Another approach of marking a binary document is proposed in [110] by treating a binary image as a grayscale one and by manipulating the luminance of dark pixels slightly so that the change is imperceptible to human eyes yet detectable by scanners. This approach, targeted on intelligent copier systems, is not applicable to bi-level images hence is beyond the scope of this paper. The bi-level constraint also limits the extension of many approaches proposed for grayscale or color images to binary images. For example, applying the spread spectrum embedding, a transform-domain additive approach proposed by Cox *et al* [44], to binary image could not only cause annoying noise on the black-white boundaries, but also have reduced robustness hence limited embedding capacity due to the post-embedding binarization that ensures the marked image is still a bi-level one [109]. For these Type-I additive embeddings, hiding a large amount of data and detecting without the original binary image is particularly difficult. In summary, these previously proposed approaches either cannot be easily extended to other binary images, or can only embed a small amount of data.

We propose a new approach that can hide a moderate amount of data in general binary images, including scanned text, figures, and signatures. The hidden data can be extracted without using the original unmarked image, and can also be extracted after high quality printing and scanning with the help of a few registration marks. The approach can be used to verify whether a binary document has been tampered with or not, and to hide annotation labels or other side information.

We shall discuss three key issues of hiding data in binary image in Section 5.2, along with our proposed solutions. In Section 5.3, we demonstrate three applications and the experimental results of the proposed data hiding approach. A variety of discussions are given in Section 5.4, including robustness analysis and security considerations. Issues such as recovering hidden data from high quality printing-and-scanning is also addressed.

5.2 Proposed Scheme

There are two basic ways to manipulate binary images for the purpose of data hiding, namely, by changing the values of individual pixels and by changing a group of pixels. The first approach flips a black pixel to white or vice versa. The second approach modifies such features as the thickness of strokes, curvature, and relative positions, which generally depends more on the types of images (e.g., text, sketches, signatures, etc.). Since the number of parameters that can be changed by the second approach is limited, especially under the requirements of blind detection (i.e., without using the original image in detection) and invisibility, the amount of data that can be hidden is usually limited except for special types of images.

We focus in this paper on using the first approach. An image is partitioned into blocks and several bits are embedded in each block by changing some pixels in that block. For simplicity, we shall show how to embed one bit in each block. Three issues will be discussed below: (1) how to select pixels for modification so as to introduce as little visual artifacts as possible, (2) specific means to embed data in each block using these flippable pixels, and (3) why to embed the same number of bits in each block and how to enhance the efficiency. The entire process of embedding and extraction is illustrated in Fig. 5.1.

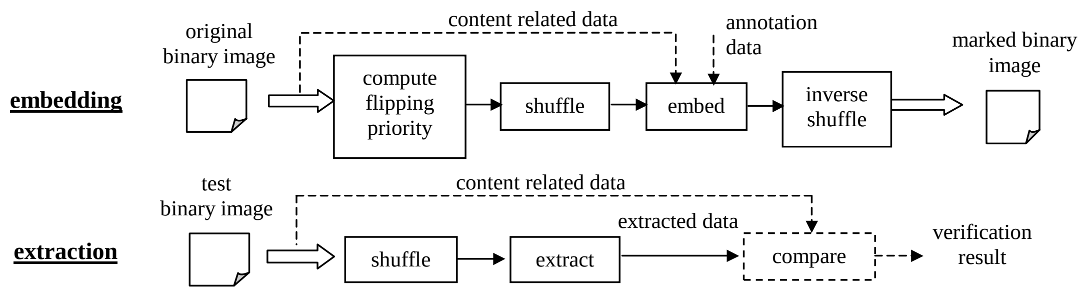


Figure 5.1: Block diagram of the embedding and extraction process in binary images for authentication and/or annotation.

5.2.1 Flippable Pixels

While a human visual model is a key element in data hiding systems, there is little discussion on a human visual model for binary images. A simple criterion, proposed in [107], is to flip boundary pixels for high contrast image such as text image and to only create rather isolated pixels for dithered image. Our work takes the human perceptual factor into account by studying the flippability of each pixel. More specifically, we examine the pixel and its neighbors to establish a continuous score of how unnoticeable such a change will be. The score is from 0 to 1 with 0 meaning absolutely no flipping. Flipping pixels with higher scores generally introduces less artifacts than flipping a lower one.

Manual score assignment, though possible, has two weaknesses. First, except for small neighborhood such as 3×3, storing the score of every pattern involves a non-trivial amount of storage – storage on the order of mega bytes is needed for a 5×5 neighborhood and the storage increases exponentially as the neighborhood gets larger.

Second, as a pure subjective process, the outcomes of manual score assignment could vary significantly from person to person because different persons may emphasize different visual artifacts. This is especially the case when the score is preferably continuous or to involve many discrete levels.

To overcome the problems with manual score assignment, we look for causes that make some flipping more visible to human eyes and measure them in a subjective way. The continuous score can be obtained by applying a set of perceptual rules on these raw measures. In our work, the score is hierarchically determined and is arrived at by considering the change in smoothness and connectivity. 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. For example, the flipping of the center pixel in Fig. 5.2(b) is more noticeable than that in Fig. 5.2(a) because the connectivity of (b) changes and human eyes are sensitive to this change. In this manner, we obtain a list of all 3×3 patterns ordered in terms of how unnoticeable the change of the center pixel will be. The list can be implemented using a look-up table. We then look at a larger neighborhood like 5×5 to refine the score. Special cases are also handled in such large neighborhood so as to avoid introducing noise on special patterns such as sharp corners. The details of our proposed score computation can be found in the appendix of this chapter (Sec. 5.6).

5.2.2 Mechanics of Embedding

When designing an embedding mechanism, it is important to consider how to extract the embedded data without the original image. Directly encoding the hidden information in flippable pixels (e.g., set to black if to embed a "0" and to white if to embed a "1") may not work since the embedding process may change a flippable pixel

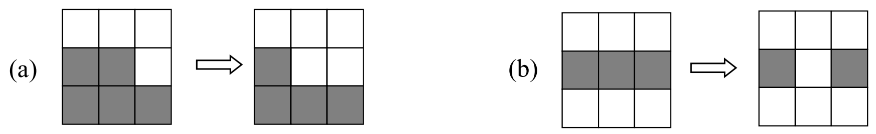


Figure 5.2: Two examples of 3×3 neighbourhood, for which flipping the center pixel to white in (a) is less noticeable than that in (b).

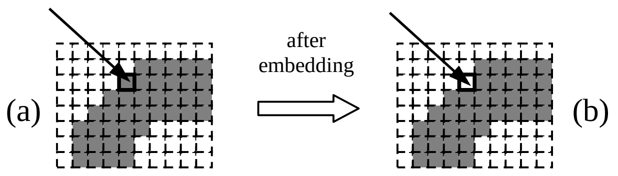


Figure 5.3: If assuming only black pixels that are directly adjacent to white pixels are considered as "flippable", the boundary pixel indicated by an arrow becomes a "non-flippable" one after embedding. This simple example demonstrates that directly encoding the hidden information in flippable pixels may not work since the embedding process may change a flippable pixel in the original image to a pixel that may no longer be considered as flippable.

in the original image to a pixel that may no longer be considered as flippable. As a simple example, suppose only black pixels that are directly adjacent to white pixels are considered as "flippable", and the flippable pixel marked by thick boundary in Fig. 5.3(a) is changed to white to carry a "1". Fig. 5.3(b) shows that after embedding, this pixel is no longer considered as flippable if using the same rule. In this case, it is hard for detector to correctly identify which pixel carries what hidden information without knowing the original image.

Instead of directly encoding the hidden information in flippable pixels, we apply the Type-II embedding discussed in Chapter 3. That is, we embed the data by manipulating flippable pixels so that a certain relationship on features of a group of pixels is enforced. One possible feature is the total number of black pixels. To embed a "0" in a block, we may change some pixels so that the total number of black pixels in that block is an even number. Similarly, to embed a "1", the number of black pixels is enforced to an odd number. An alternative approach is to choose a "quantization" step size Q and to force the total number of black pixels in a block to be 2kQ (for some integer k) in order to embed a "0", and to be (2k+1)Q to embed a "1". As discussed in Chapter 3, larger Q gives higher robustness against noise because any perturbation smaller than $Q/2$ will not affect the accuracy in decoding; however, as a tradeoff, the changes introduced by the embedding process also increases and the image quality may be reduced. The "odd-even" method can be viewed as a special case of the table lookup approach which is similar to those in [78, 163] and in Chapter 7. These two approaches are illustrated in Fig. 5.4, where each possible quantized number of black pixels per block is mapped to 0 or 1. The marked image is generated by manipulating pixels with high flippability score in such a way that the number of black pixels in each block is enforced to match the bit to be embedded via a prescribed mapping. That is,

$$v'_i \quad = \quad \arg \min_{x:T(x)=b_i, x=kQ} |x - v_i| \tag{5.1}$$

where v_i is the i^{th} feature to be enforced (in the above case, the total number of black pixels of the i^{th} block), v'_i is the feature value after embedding, b_i is the bit to be embedded in i^{th} feature, and $T(\cdot)$ is a prescribed mapping from feature values to hidden data values {0,1}, which may be represented in closed-form or by a lookup table. Detection is done by checking the enforced relationship - for the above cases, to examine the odd/even properties, or to perform a table lookup. That is,

$$\hat{b}_i = T(v''_i) \tag{5.2}$$

where v_i'' is the feature extracted from the i^{th} block of a test image, and $\hat{b}_i$ is the estimated value of the embedded bit in the i^{th} block. If one bit is repeatedly embedded in more than one block, majority voting is performed to determine which bit has been hidden. More sophisticated coding than simple repetition may be used to enhance the performance.

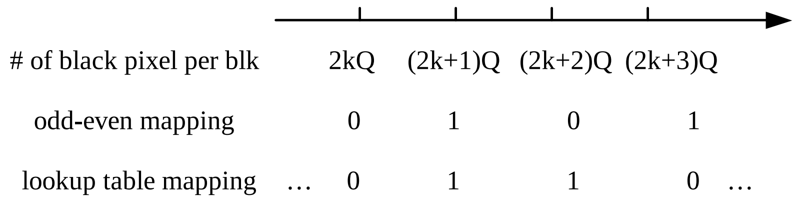


Figure 5.4: Illustration of odd-even mapping and table lookup mapping from the quantized number of black pixels per block to the binary data to be embedded. One bit can be embedded in a block by enforcing the number of black pixels to a value that matches the bit to be embedded via a prescribed mapping. The enforcement involves the change of flippable pixels in the block, if necessary.

While other relationship enforcing techniques are certainly possible, we shall in this chapter, for simplicity of discussion, use the enforcing of odd or even number of black pixels.

5.2.3 Uneven Embedding Capacity and Shuffling

As outlined earlier, we embed multiple bits by dividing an image into blocks and hiding one bit in each block via the enforcement of the odd-even relationship. However, the distribution of flippable pixels may vary dramatically from block to block in a binary image. No data can be embedded in the white or black uniform regions, while regions with text, drawing, and dithered images may have quite a few flippable pixels, especially on the non-smooth boundary. This uneven embedding capacity can be seen from Fig. 5.5 where the pixels with high flippability scores, indicated by black dots, are on the rugged boundaries.

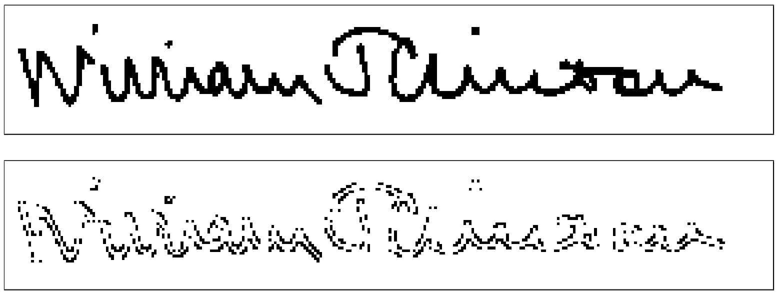


Figure 5.5: A binary image (top) and its pixels with high flippability scores (bottom, shown in black).

General approaches to handling uneven embedding capacity have been discussed in Chapter 4. Regarding the uneven embedding capacity in a binary image, using variable embedding rate from block to block is not feasible because (1) a detector has to know exactly how many bits are embedded in each block, and any mistake in estimating the number of embedded bits is likely to cause errors in decoding the hidden data for the current block and the error may propagate to the other blocks, and (2) the overhead for conveying this side information via embedding is quite significant and could be even larger than the actual number of bits that can be hidden. For these reasons, we adopt constant embedding rate (i.e., to embed the same number of bits in each region) and use shuffling to equalize the uneven embedding capacity from region to region.

As shown in Fig. 5.6, the flippable pixels distribute more evenly after a random permutation of all pixels. This is also illustrated in the histogram of the number of flippable pixels in one 16×16-pixel block (Fig. 5.7). Before shuffling, the distribution

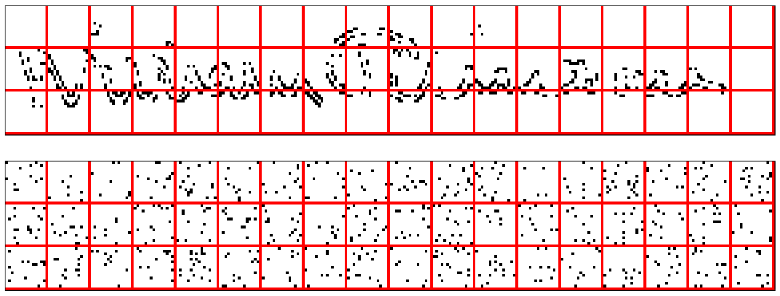


Figure 5.6: Distributions of flippable pixels per 16x16-pixel block of the binary image in Fig. 5.5, before shuffling (top) and after shuffling (bottom).

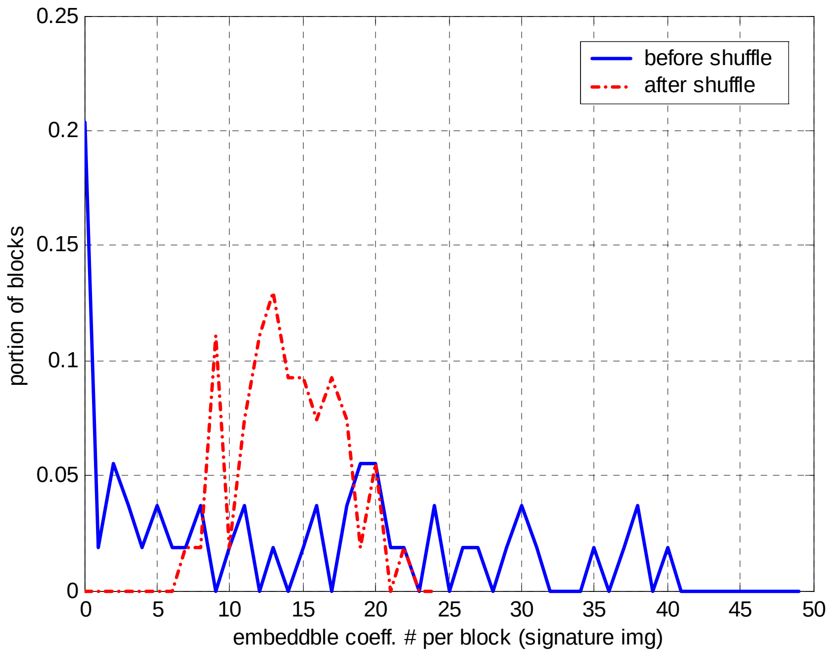


Figure 5.7: Histogram of flippable pixels per 16x16-pixel block of the binary image in Fig. 5.5, before shuffling (solid line) and after shuffling (dotted-dash line).

extends from 0 to 40 flippables per block and that about 20% of the blocks do not have any flippable pixels. This implies that either we embed nothing in those blocks or we have to introduce significant artifacts to hide data there. The distribution after shuffling, shown as the dotted line, concentrates from 10 to 20, and ALL shuffled blocks have flippable pixels. This equalization function of shuffling has been analyzed in Chapter 4. Plugging into Eq. 4.1 and Eq. 4.2 the parameters of the binary signature image of Fig. 5.5:

$$\begin{cases} \text{block size} & q = 16 \times 16 \\ \text{image size} & S = 288 \times 48 \\ \text{block number} & N = S/q = 18 \times 3 \\ \text{flippable percentage} & p = 5.45\% \end{cases}$$

we compute the mean and the standard deviation of the histogram. The analytic results are shown in Fig. 5.8, along with the simulation results from 1000 random shuffles. Table 5.2.3 also shows the behavior of blocks with no or few flippables, which are of most concern for data hiding problems. We can see that the analysis and simulation conform with each other very well, and the percentage of blocks with no or few flippables is extremely low. For the problem that we may encounter a small number of blocks with no flippable pixel to carry hidden information, applying error correction encoding with a little correction capability would be sufficient to handle this. As can be seen from the block diagram in Fig. 5.1, the embedding of one bit per block of fixed size described in Section 5.2.2 is performed in the shuffled domain, and inverse shuffling is performed to get a marked image.

We have also discussed in Chapter 4 that shuffling does not produce more flippable pixels. Instead, it dynamically assigns the flippable pixels around the active regions and rugged boundaries to carry more data than the less active regions, yet without the

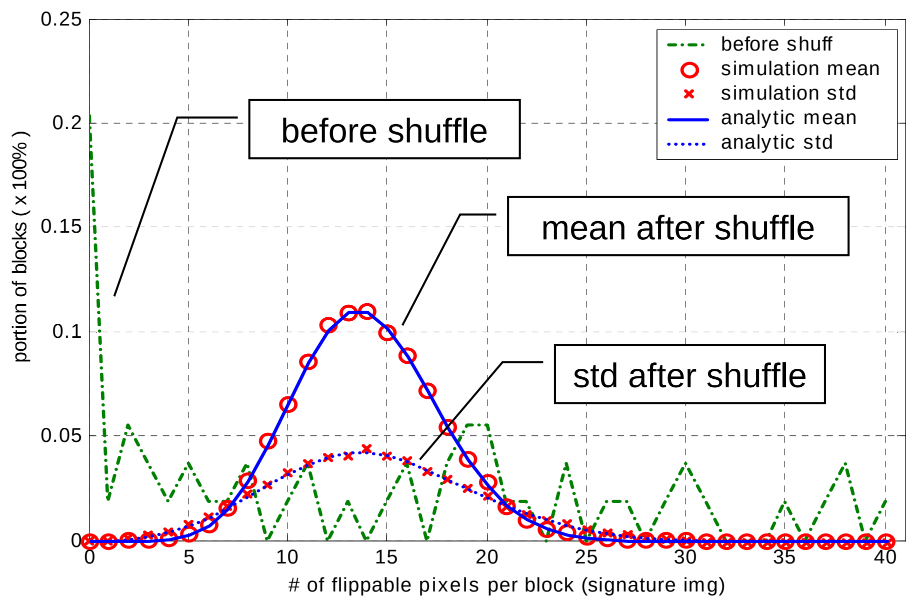


Figure 5.8: Analysis and simulation of the statistical behavior of shuffling for the binary image in Fig. 5.5.

Table 5.1: Analysis and simulation of the blocks with no or few flippable pixels before and after shuffling for the binary image in Fig. 5.5.

	before shuffle	mean after shuffle		std after shuffle	
		analysis	simulation	analysis	simulation
m_0/N (0th bin)	**20.37%**	**5.16×10^{-5} %**	**0 %**	**9.78×10^{-5}**	**0**
m_1/N (1st bin)	**1.85%**	**7.77×10^{-4} %**	**0 %**	**3.79×10^{-4}**	**0**
m_2/N (2nd bin)	**5.56%**	**5.81×10^{-3} %**	**5.56×10^{-3} %**	**0.0010**	**0.0010**

need of specifying side information that is image dependent. Shuffling also enhances security since the shuffling table or a key for generating the table is needed to correctly extract the hidden data.

5.3 Applications and Experimental Results

In this section, we present three applications of the proposed data hiding method for binary images along with experimental results.

5.3.1 "Signature in Signature"

We mentioned before that unauthorized use is a potential concern for the increasingly popular use of digitized signature. "Signature in Signature" has been proposed as a tool for annotating the signer's signature with the data that is related to the signed documents, so that the unauthorized use of a signature can be detected [111]. Here the second "signature" refers to the actual digital version of a person's signature, while the first "signature" refers to a checksum related to the document content or other annotation information. The data hiding method proposed in this paper can be applied to annotating a signature in such applications as faxing signed documents and storing digitized signatures as transaction records. Compared with the traditional cryptographic authentication approach [11] that has been used in secure communication, the proposed data embedding based approach has the advantage of being user-friendly, easily visualized, and integrating the authentication data with the signature in a seamless way, hence is suitable for the general public.

An example is demonstrated in Fig. 5.9, in which up to 7 characters (approximately 50 bits) can be embedded in a 287×61 signature of Fig. 5.9(top). The embedding rate is 1 bit per block of 320 pixels. Fig. 5.9(middle), which has 7 letters embedded, differs very little from the original one, as indicated by black pixels in Fig. 5.9(bottom) [1].

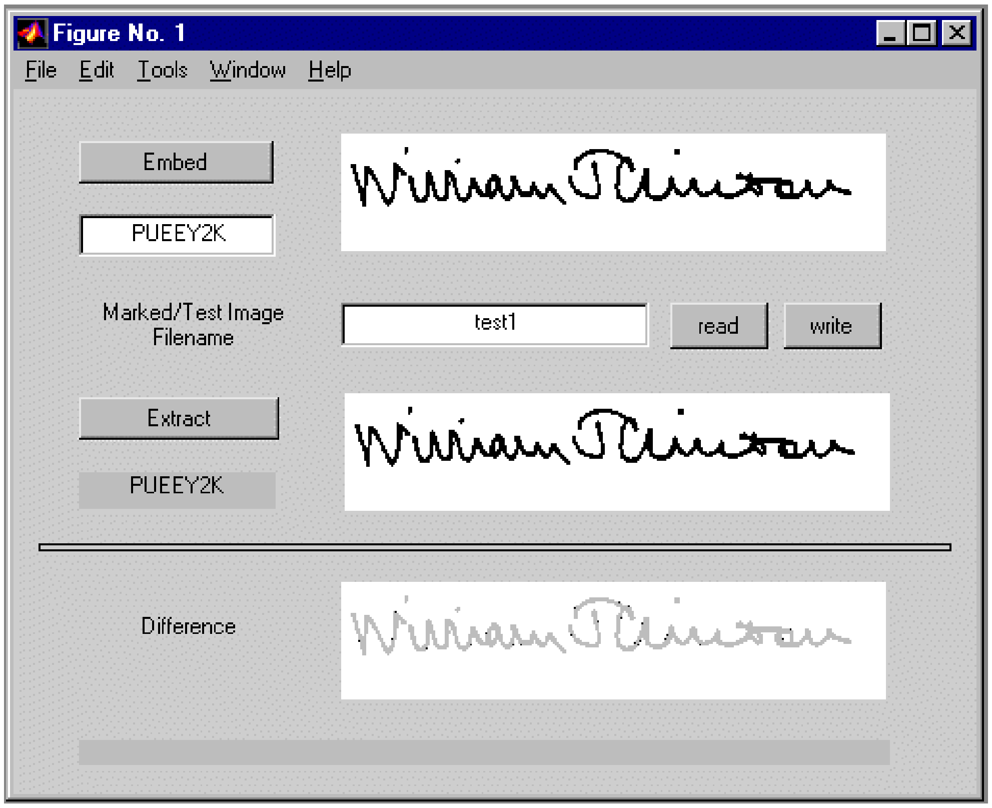


Figure 5.9: “Signature in Signature”. (top) the original image, (middle) a marked copy with 7 letters (approximately 50 bits) embedded in, (bottom) the difference between the original and the marked (shown in black).

[1]The gray areas in Fig. 5.9(bottom) and Fig. 5.11(d), visualizing the strokes and the background, respectively, are for assisting viewers to associate the difference between the original and the marked image with their precise location in the images. They don’t indicate the pixelwise differences that are indicated by black pixels.

5.3.2 Invisible Annotation for Line Drawings

In artistic applications, one would usually prefer to annotate the artwork with information like the creation date and location in such a way that the annotation data interfere with perceptual appreciation to the minimal extent. Our proposed approach can be used to invisibly annotating artistic line drawings like the binary comic picture (120×150) shown in Fig. 5.10. In this example, a character string of date information "01/01/2000" is embedded in Fig. 5.10(middle). We can see that the annotation does not interfere with perceptual appreciation in any perceivable way.

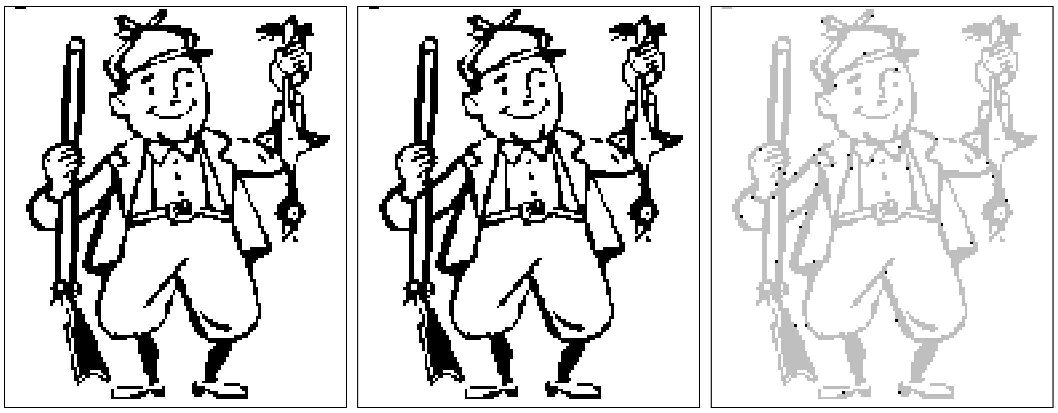


Figure 5.10: Invisible annotation for line drawings: (left) the original image, (middle) a marked copy with 10-letter date information (70 bits) embedded in, (right) the difference between the original and the marked (shown in black).

5.3.3 Tamper Detection for Binary Document

As we have mentioned, a large number of important documents have been digitized and stored for records. Because it is easy to edit digital images, the authentication of these digital documents as well as the detection of possible tampering is a very important concern. The data hiding techniques proposed in this paper can be applied for such purposes, as an alternative to or in conjunction with the cryptographic authentication approach.

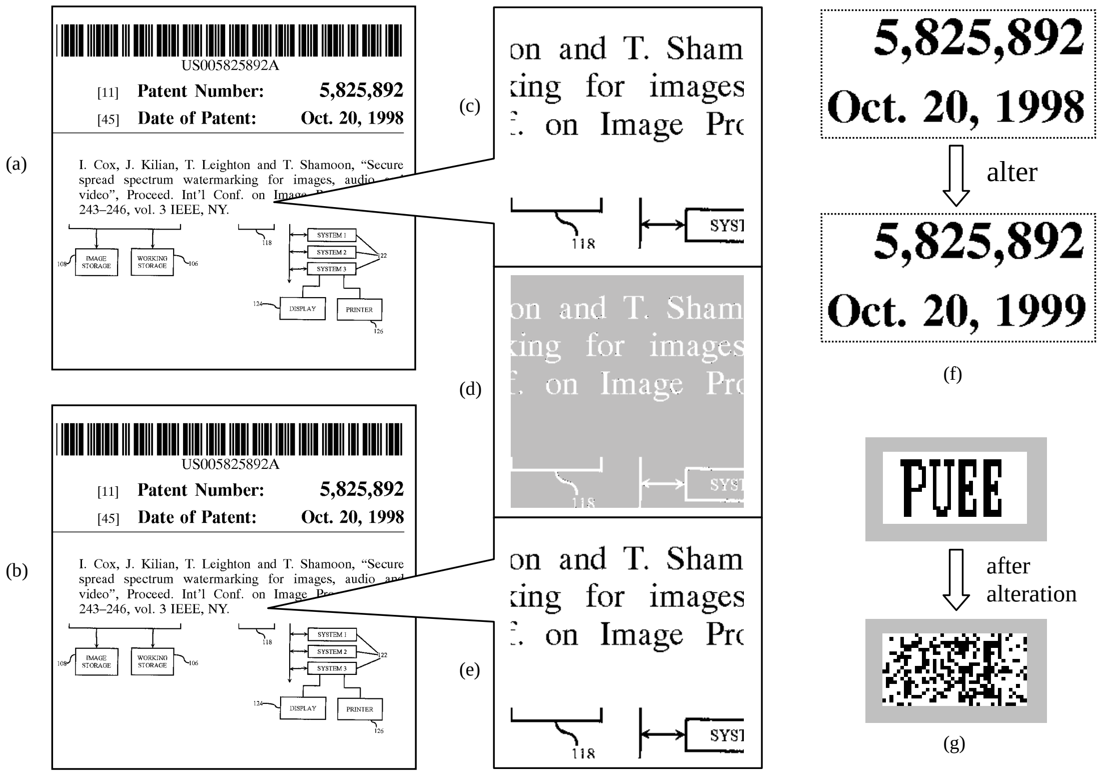
The basic idea of authentication is the same as that for grayscale and color image [163]. Data is embedded in an image in such a fragile way that it will be obliterated if the image is altered and/or it no longer matches some properties of the image. The hidden data may be an easily recognized pattern and/or some features/digest related to the content of host image. Shown in Fig. 5.11(a) is a part of a U.S. Patent, consisting of 1000×1000 pixels. This binary image contains a variety of patterns such as texts, drawings, lines, and bar codes. Fig. 5.11(b) is a visually identical figure, but with 976 bits embedded in it using the proposed techniques. In this particular example, 800 bits of the embedded data forms a "PUEE" pattern shown in Fig. 5.11(g). If the date "1998" on the top is changed to "1999", the extracted data will be the random pattern shown in Fig. 5.11(g) and significantly different from the originally embedded one. This is an indication that alteration was made on the document.

5.4 Discussions

In this section, we will discuss the robustness and security issues of the proposed scheme. Other considerations associated with shuffling, such as the methods for handling bad shuffles and for adaptively choosing the block size, can be found in Chapter 4.

5.4.1 Analysis and Enhancement of Robustness

In Section 5.2.2, we discussed possible ways of embedding secondary data via manipulating pixel values to enforce certain relationships (i.e., Type-II embedding). The


US005825892A

[11] Patent Number: 5,825,892

[45] Date of Patent: Oct. 20, 1998

I. Cox, J. Kilian, T. Leighton and T. Shamoon, "Secure spread spectrum watermarking for images, audio and video", Proceed. Int'l Conf. on Image Pr... 243–246, vol. 3 IEEE, NY.

(a)
(b)
(c)
(d)
(e)
(f)
(g)
IMAGE STORAGE
WORKING STORAGE
SYSTEM 1
SYSTEM 2
SYSTEM 3
DISPLAY
PRINTER
118
5,825,892
Oct. 20, 1998
alter
5,825,892
Oct. 20, 1999
PUEE
after alteration

Figure 5.11: Data hiding in binary document image. (a) Original copy, (b) a marked copy with 976-bit embedded in, (c) magnified original image, (d) difference between original and marked (shown in black), (e) magnified marked image, (f) a portion of the image where alteration is done (on the marked image) by changing "1998" to "1999", (g) among the 976-bit hidden data, 800 bits forms a "PUEE" pattern; the 800-bit data patterns extracted after alteration is visually random and significantly different from the embedded "PUEE".

robustness against noise is quite limited, and generally depends on whether and how much quantization or tolerance zone we applied. Let us consider the simple odd-even case with no quantization, i.e., the total number of black pixels is enforced to an even number to embed a "0", and to an odd number to embed a "1". When a single pixel gets flipped due to noise, the bit embedded in the block to which the pixel belongs to will be wrongly decoded. When several pixels in an embedding block are subject to be changed, whether or not the bit can be decoded correctly depends on how many pixels being flipped; if the change is independent from pixel to pixel and is with probability p for each of n pixels where $n \geq 1$, the probability of getting a wrongly decoded bit is

$$P_{e_1} = \sum_{k=1, k \text{ odd}}^{n} \begin{pmatrix} n \\ k \end{pmatrix} p^k (1-p)^{n-k} = \frac{1-(1-2p)^n}{2}. \tag{5.3}$$

The error probability P_{e_1} is small for small p and small n. In this case, error correction encoding can be applied to correct errors if accurate decoding of hidden data is preferred. When p is close to 0.5, so is P_{e_1}, implying the difficulty in embedding and extracting data reliably. Notice that because of shuffling, the assumption of independent change is likely to hold even if the noise involves nearby pixels since adjacent pixels in the original image will be distributed to several blocks. If the total number of changed pixels in the whole image is small (no matter whether they are close to each other in the original image or far away), it is likely that most of those pixels are involved in different embedding blocks hence the extracted bits from those blocks will be wrong; on the other hand, if many pixels have been changed, each embedding block may include several of these pixels and the decoded bit from each block is wrong with approximately 0.5 probability. This implies that the decoded data are rather random, like what we have seen in Fig. 5.11(g). The case of incorporating quantization or tolerance zone can be analyzed similarly, combining the above strategies and those in Chapter 3.

Besides the noise involving flipping of individual pixels, misalignment is another cause of decoding errors. For this matter, using shuffling has the disadvantage of increasing the sensitivity against geometric distortion such as translation. This is due to the shift-variant property of the shuffling operation, i.e., the shuffling result of a shifted image is very different from that of the non-shifted one. To alleviate the sensitivity with respect of translation, we can hide secondary data in a cropped part of the image, as shown in Fig. 5.12. Without loss of generality, we consider the case of black foreground and white background. The upper-left point of the data hiding region is determined by the uppermost and leftmost black pixel, and the lower-right point is by the lowermost and rightmost black pixel. The data hiding region therefore covers all black pixels. This approach can reduce the shifting sensitivity as long as both embedding and detection system agree on the protocol and no cropping or addition of the outermost black pixels is involved.

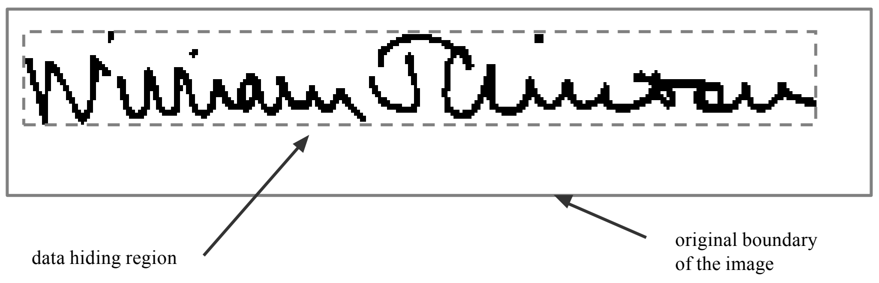


Figure 5.12: Achieving robustness against small translation. Here we use the outermost black pixel to determine a data hiding region (indicated by a dash box) covering all black pixels.

In addition to the above approach, adding registration marks helps to survive high-resolution printing and scanning. Recovering the image from printing and scanning with precision as high as one pixel is a challenging task, because this D/A-A/D process may result in small rotation, up-scaling of an unknown factor, and noisy boundary. If one original pixel in the image corresponds to a very small number of pixels in the scanned version (e.g., corresponding to one or less than one pixel), it will be very difficult to combat the distortion introduced by the D/A-A/D process. On the other hand, if significant oversampling is performed so that one original pixel corresponds to a large number of pixels in the scanned version, it would be possible to sample at the center of each "original" pixel, averaging out the noise introduced on the boundary and/or by the rounding errors in de-skewing. The registration marks help to identify the boundary and the size of the original image as well as to correct skewing. We noted that while the size of one original pixel represented in the scanned image may be estimated from a well-designed registration mark (e.g., we may estimate that one original pixel corresponds to 8×8 pixels in a scanned image), minor errors in such estimation could be accumulated when determining the width and height of the original image up to single pixel precision. For this reason, we impose constraints on the width and height of original images, for example, to be multiples of 50. As shown in Fig. 5.13(a), one possibility is to add cross-shape marks at four corners and at four sides at an interval of 50 pixels horizontally and of 25 pixels vertically, serving as a ruler. In our experiment, we printed out the signature image via the Microsoft Word program (with default image importing resolution 72dpi) using a HP 2100TN laser printer, and scan back with 600dpi precision and 256 gray levels using a Microtek 3600 scanner. The image is binarized using the mean of the maximum and minimum of scanned luminance value. We use the registration marks to determine the image boundary, to perform de-skewing, and to compute the proper scaling factor. The estimated centers of each original pixel on the scanned version are shown in Fig. 5.13(b). Sampling at those pixels can recover the original digital image perfectly from the scanned one hence allow the embedded data to be extracted correctly. In the Appendix Section 5.7, we shall present more detailed discussion on the recovery of binary image from printing and scanning.

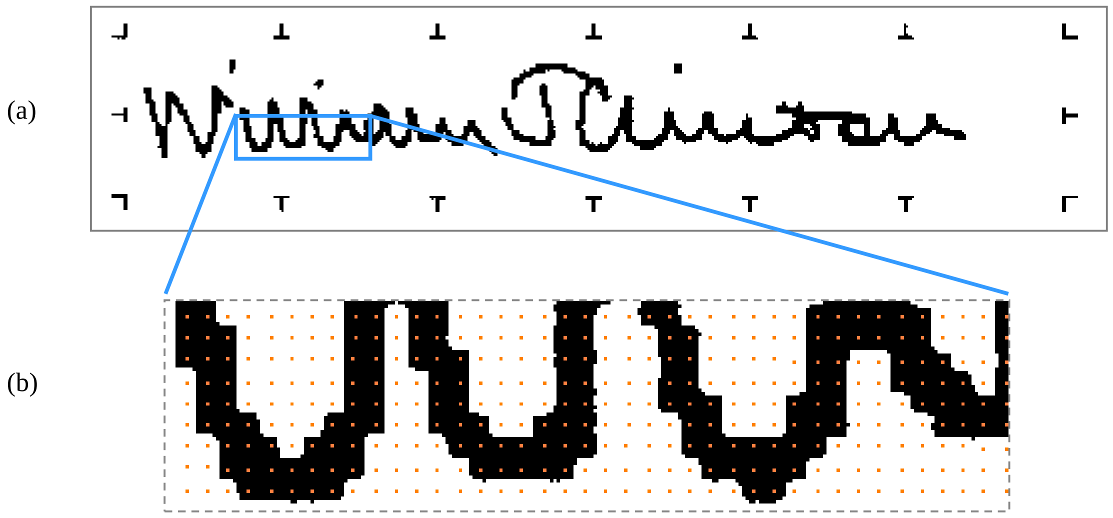


Figure 5.13: Recovering binary image from high quality printing and scanning. (a) Cross-shape marks are added at four corners and at four sides at an interval of 50 pixels horizontally and of 25 pixels vertically, helping to determine the boundary, the scale, and the skewing angle of a scanned image; in addition, the width and height of original images are constrained to be multiples of 50; the image is imported to Microsoft Word at 72dpi, printed out via a laser printer, and scanned in with 600dpi and 256 grey levels; the size of the scanned image is 2028x444. (b) The estimated centers of each original pixel are shown in light color; sampling at those centers can recover the original binary image perfectly from a scanned one.

5.4.2 Security Considerations

We have demonstrated how to embed data in a binary image and illustrated two possible applications in Section 5.3. Drawing an analogy between data hiding and communication, the embedding methods serve as physical communication layer, on top of which other functionalities and features can be built. For instance, security issues may be handled by top layers in authentication applications. Under this scenario, the major objective of an adversary is to forge authentication data so that an altered document can still pass the authentication test. One could use traditional cryptography-based authentication to produce a cryptographical digital signature and to embed it in the binary image. This traditional approach relies on a cryptographically strong hash function to produce a digest of the document to be signed as well as on public-key encryption to enable verification without giving up the encryption keys, hence only authorized person can produce a correctly encrypted signature [11]. By using embedding, we not only save room that is needed for storing and/or displaying the cryptographical data separately, but also obtain additional capability to associate the authentication data with the media source in a seamless way.

Although a cryptographical signature can be adopted as (part of) the embedded data, the embedding approach proposed in this paper has the potential of allowing plain text to be embedded since secret information such as keys/seeds have already been incorporated via shuffling and/or lookup table. However, envisioning potential attacks [139], the following security issues have to be considered. More specifically, for the application of authentication, it is important to study the following two problems, assuming that the attacker has no knowledge about any secret keys: (1) the probability of making content alterations while preserving the m-bit embedded authentication data, and (2) the possibility for an adversary to hide specific data in an image, assuming he/she has no knowledge about any secret keys.

For the first problem, we have discussed in Section 5.4.1 that an n-pixel alteration on a marked image would change the decoded data. If n is small compared to the total number of blocks m, there are approximately n bits in the decoded data that will be different from the originally embedded one; if n is large, the probability of getting the decoded data to be exactly the same as the originally embedded one is approximately 2^{-m}, which is very small as long as m is reasonably large. Therefore, the threat of making content alterations while preserving the m-bit embedded authentication data is weak.

For the second problem, it depends on whether watermarked versions of the same image with different data embedded are available to an adversary. For convenience, we shall call these as "multiple copies". When multiple copies are not available, it is extremely hard for an adversary to embed specific data in an image, even if he/she knows the algorithm. This is due to the secrecy in the shuffling table. However, in applications such as "signature in signature", an adversary may be able to obtain multiple copies, for example, signatures with different signing date or payment amount embedded. This is similar to the scenario of plaintext attack in cryptography [11]. We would like to know whether he/she can derive information regarding which pixels carrying which bit by studying the difference between those copies hence create new images with specific data embedded (e.g., specific date or payment amount). If the embedding imposes the minimal necessary changes to enforce a desirable relationship (for example, in the odd-even case, at most one pixel will be flipped in each embedding block), the pixels that differ among the multiple copies are those used for embedding hidden information. Assuming an adversary collects sufficiently many copies and knows what data is embedded in each copy, he/she will be able to identify which pixels carrying which bit and to hide his/her desired data by manipulating the corresponding pixels.

To prevent the above-mentioned attack, we have to introduce more uncertainty. One approach is to use a different shuffling table, for example, choose one table from K candidate ones, similar to the approach used for handling bad shuffles in Section 4.2.3. Another approach is that instead of making minimal changes for hiding one bit in each embedding block, we also flip, with probability of 0.5 in each block, an additional pair of flippable pixels. More specifically, to embed a "0", if the number of black pixels in a shuffled block is already an even number, with probability of 0.5 we flip an additional pair of pixels selected arbitrarily from 3 highly flippable pixels; if the number of black pixels is an odd number, with probability of 0.5:0.5 we flip all 3 pixels or flip one pixels selected arbitrarily from the 3 ones. When more than three highly flippable pixels are available, we may make the above selection from a larger pool. Now if we look at two image copies whose hidden data differ in just one bit, the difference between the two images via minimal-change embedding is just at one pixel, while that via the above-mentioned randomization involves many other pixels randomly. In the latter case, if a total of N bits are embedded, on average there will be $(4N + 1)/3$ pixels being different. The computation is sketched as follows: we first consider the embedding of the different bit between the two images, i.e., in one image, 0 or 2 pixels are flipped while in the other image, 1 or 3 pixels are flipped. The four combinations (0-1, 2-1, 0-3, 2-3) are equally likely, giving the average number of different pixels as a result for embedding this bit as $5/3$. Similarly, the average number of different pixels for embedding the rest $(N - 1)$ identical bits is $4(N - 1)/3$, giving the overall average $(4N + 1)/3$. When N is sufficiently large, it is difficult for an adversary to identify which pixels are associated with which bits. As a tradeoff, the randomization requires three flippable pixels to be available for many shuffled blocks and introduces more pixel changes at the embedding step. Note that both countermeasures assume that for any given hidden data, only one copy of a marked image is available to an attacker, otherwise he/she may be able to average out the randomization and to compromise our solutions.

5.5 Chapter Summary

This chapter addresses the problem of data hiding for binary images. Technical challenges in such embedding are discussed with solutions proposed. In particular, we propose a new data hiding method for binary images. The method manipulates "flippable" pixels to enforce a specific block-based relationship in order to embed a significant amount of data without causing noticeable artifacts. Shuffling is applied before embedding to equalize the uneven embedding capacity. The hidden data can be extracted without using the original image, and with the help of a few registration marks, they can also be accurately extracted after high quality printing and scanning. The algorithm can be applied to detect unauthorized use of signatures in binary image format and to detect alterations on documents.

In terms of future work, the flippability model can be refined for different types of binary images such as texts, figures/drawings, and dithered images. The approach for recovering binary image from high quality printing and scanning may be improved by using the grayscale information from the scanned image. Comparative study of various embedding mechanisms will also provide insights on the improvement of data hiding in binary images.

Acknowledgement

The connectivity criterion for generating flippability scores was revised from a proposal by Ed Tang of Princeton Summer Institute '99. The application of "signature in signature" was proposed by Computer Science Prof. Adam Finkelstein.

5.6 Appendix - Details of Determining Flippability Scores

In this appendix section, we describe the details of a 5-step procedure for computing flippability scores. For simplicity, we shall illustrate the evaluation method for non-dithered binary image. The scores will be used to determine which pixel will be flipped with high priority during the embedding process.

Step-1 Compute smoothness and connectivity of 3×3 pattern.

The smoothness of the neighborhood around pixel (i, j) is measured by the total number of horizontal, vertical, diagonal, and anti-diagonal transitions in the 3x3 window, respectively, using a differential operator along the corresponding directions:

$$\begin{aligned}
\text{horizontal} \quad N_h(i,j) &= \sum_{k=-1}^{1} \sum_{l=-1}^{0} I(\{p_{i+k,j+l} \neq p_{i+k,j+l+1}\}), && (5.4)\\
\text{vertical} \quad N_v(i,j) &= \sum_{k=-1}^{1} \sum_{l=-1}^{0} I(\{p_{i+l,j+k} \neq p_{i+l+1,j+k}\}), && (5.5)\\
\text{diagonal} \quad N_{d_1}(i,j) &= \sum_{k,l \in \{-1,0\}} I(\{p_{i+k,j+l} \neq p_{i+k+1,j+l+1}\}), && (5.6)\\
\text{anti-diagonal} \quad N_{d_2}(i,j) &= \sum_{k \in \{0,1\}, l \in \{-1,0\}} I(\{p_{i+k,j+l} \neq p_{i+k-1,j+l+1}\}). && (5.7)
\end{aligned}$$

where $I(\cdot)$ is the indicator function taking value from $\{0, 1\}$, and $p_{i,j}$ denotes the pixel value of the i^{th} row and j^{th} column of the whole image. These computations are also illustrated in Fig. 5.14. Note that regular patterns such as straight lines, have zero transition along at least one direction, as shown in Fig. 5.15.

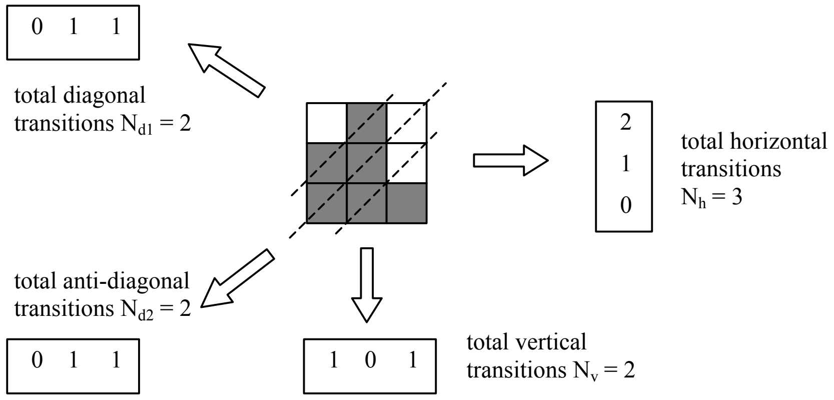

0 1 1
total diagonal transitions N_d1 = 2
2
1
0
total horizontal transitions N_h = 3
total anti-diagonal transitions N_d2 = 2
0 1 1
1 0 1
total vertical transitions N_v = 2

Figure 5.14: Illustration of transitions in four directions, namely, horizontal, vertical, diagonal, and anti-diagonal. The number of transitions is used to measure the smoothness of the 3×3 neighbourhood.

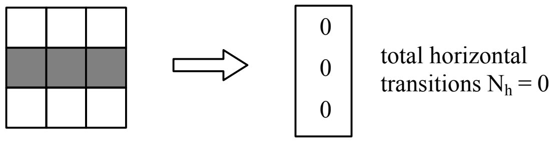


Figure 5.15: Regular patterns such as straight lines have zero transition along at least one direction. Showing here is part of a horizontal line with zero horizontal transition.

The connectivity is measured by the number of the black and white clusters. For this purpose, the connectivity criterion between two pixels needs to be prescribed. A commonly used criterion, illustrated in Fig. 5.16, considers the pixels that touch each others by 90-degree (i.e., $(i, j \pm 1)$ or $(i \pm 1, j)$) or by 45-degree (i.e., $(i + 1, j \pm 1)$

or $(i-1, j\pm 1)$) and that have the same pixel value as *connected* [2] Depending on the specific constraints of visual artifacts, 45-degree touching may not always be considered as connected. Using the criterion, we can build a graph for black (or white) pixels. In the graph, each vertex represents a black (or white) pixel, and there is an edge between two vertices if and only if the two corresponding pixels are connected. An example is shown in Fig. 5.17 with five black pixels forming two clusters and four white pixels forming one cluster. The number of clusters can be automatically identified by traversing the graph using *depth-first search* strategy. Here we present a stack-based implementation of non-recursive depth-first search algorithm, adapted from [4]. We assume that there are M pixels in total (counting both white and black), and the final value of "counter" indicates the number of clusters.

(1) Initialization: let $p[k]$ store the value of k^{th} pixel and q be the pixel value of interest (i.e., q is black if to find black clusters, and vice versa); set up an empty stack and an M-element array $label[\cdot]$ for storing the index of the cluster that each pixel belongs to; set $label[k]=0$ for all $k=1,...,M$; $i=1$; $counter=0$.

(2) If $label[i]\neq 0$ (i.e., it has already been visited) or $p[i]\neq q$, go to (7).

(3) $counter = counter + 1$; push node-i into the stack.

(4) If the stack is empty, go to (7).

(5) $k = \text{pop}(\ )$ from stack; $label[k] = counter$.

(6) Find all pixels directly connected with k. For each connected pixel j, if $label[j] = 0$ (i.e., it has not yet been visited or pushed into stack), assign $label[j] = -1$,

[2]In some references, 90-degree touching is known as *four-connectivity*, and 90-degree or 45-degree touching is known as *eight-connectivity* [17].

and push node-j into stack [3]. Go back to (4).

(7) $i = i + 1$; if $i > M$, stop, otherwise go to (2).

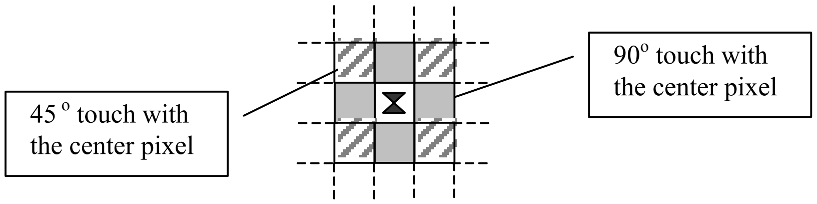

45° touch with the center pixel
90° touch with the center pixel

Figure 5.16: The pixels that touch each others by 90^o (i.e., $(i, j \pm 1)$ or $(i \pm 1, j)$) or by 45^o (i.e., $(i+1, j \pm 1)$ or $(i-1, j \pm 1)$) and that have the same pixel value as *connected*. The lightly shaded pixels in this figure are considered as touching the center pixel by 90^o, while those with stripes touch the center by 45^o.

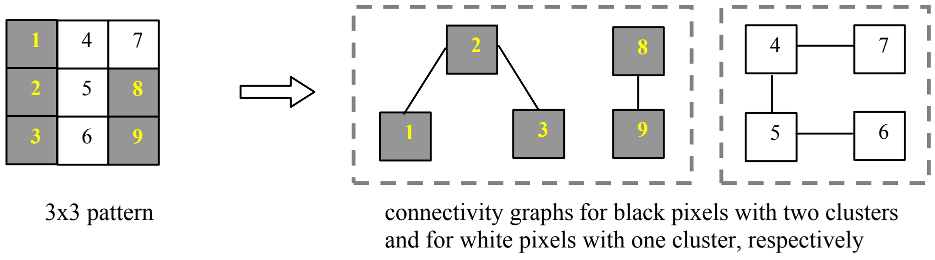

1
4
7
2
5
8
3
6
9
3x3 pattern
2
8
1
3
9
4
7
5
6
connectivity graphs for black pixels with two clusters and for white pixels with one cluster, respectively

Figure 5.17: Graph representation of the connectivity for black and white pixels. Showing here is an example of 3x3 pattern with five black pixels forming two clusters and with four white pixels forming one cluster. Only 90-degree touching is considered as connected in this example.

Step-2 Compute flippability score.

The smoothness and connectivity measures are passed into a decision module to come up with a flippability score. Main considerations when designing this module

[3]Note that by the definition of *connected*, $p[j] = q$.

are: (1) whether the original pattern is a very smooth pattern, (2) whether flipping will increase non-smoothness by a large amount, (3) whether flipping will cause the change of connectivity. These changes or the artifacts on these patterns are generally more significant. For each pixel, the followings are the major rules used in our decision module:

(1) the lowest score (i.e., not flippable) is assigned to uniform white or black regions as well as to the isolated single white or black pixels. These trivial cases are handled first.

(2) if the number of transitions along horizontal or vertical direction is zero, i.e., the pattern is very smooth and regular, assign zero flippability as a final score for the current pixel. Otherwise, assign to the pixel a base flippability score S_B and proceed to the next rule.

(3) if the number of transitions along diagonal or anti-diagonal direction is zero, reduce the flippability. Otherwise, if the minimum transition point along any one of the four directions is below a given threshold T_1, which means the pattern is rather smooth, reduce flippability by a small amount. Note that we treat smooth horizontal/vertical patterns and diagonal/anti-diagonal patterns differently because the artifacts along the horizontal/vertical patterns are likely to attract more attention from viewers.

(4) if flipping the center pixel does not change the transition points, increase the flippability. Otherwise, if flipping results in the increase of transition points (i.e., reduces smoothness and makes the pattern noisy), decrease flippability.

(5) if flipping changes the number of black clusters or white clusters, reduce the flippability.

After going through these rules, a lookup table of all 3×3 patterns can be obtained and ordered in terms of how unnoticeable the change of the center pixel will cause. For small neighborhood such as 3×3, this table has a small number of entries ($2^{3\times3} = 512$) hence can be off-line computed. The flippability score of every pattern in an image can then be determined by looking up the stored table. When larger neighborhood is involved, for example, 5×5 neighborhood, the table size increases exponentially ($2^{5\times5} = 2^{25} \approx 32$mega), possibly exceeding the available memory size for particular applications. This problem can be solved by online computing the flippability for each pattern. More efficiently, we may adopt hierarchical approach, namely, obtaining preliminary flippability measure based on a small neighborhood (e.g., 3×3) by table lookup, then if necessary, refining the measure by on-line computing based on a larger neighborhood.

Step-3 Handle special cases.

We handle some special cases that involve larger neighborhood. For example, we detect particular patterns such as sharp corners to avoid introducing annoying artifacts on them.

Step-4 Impose minimum distance constraint between two flippable pixels.

Up to now, the flippability evaluation is done independently for the pattern revealed in a moving window centered at each pixel, assuming that any pixels other than the center one will not be flipped. Pixels that are close to each others may be considered flippable by this independent study, but simultaneously flipping them could cause artifacts. We handle this problem by imposing constraints on the minimum distance of two pixels that can be flipped and pruning the pixels with relatively low flippability in its neighborhood.

Step-5 Assign a predetermined score to the remaining boundary points (optional).

Edge pixels that have not yet been assigned non-zero flippability will be given a small flippability value. These pixels serve as a bottom line for hiding a particular bit when there is no pixel with higher flippability available to carry the hidden data. Adding this step helps to achieve a high embedding rate while keeping visual quality reasonably good for the extreme cases.

The above procedures can be further refined by studying a larger neighborhood and by using more extensive analysis, especially for Step-2. Shown in Fig. 5.18 is one possible lookup table for 3×3 patterns, excluding the symmetric cases of rotation, mirroring, and complement. Here we set the threshold $T_1 = 3$, the base flippability score $S_B = 0.5$, and the flippability adjustments in Step-2 are multiples of 0.125. For dithered image, some criterions and parameters need to be revised, for example, a pixel is given high flippability if its flipping does not cause larger relative change in local intensity, and the connectivity is given less consideration. The techniques in lossy bi-level image compression like those in JBIG2 activities [20] may provide further insights to data hiding, and the methods used in data hiding may also contribute to compression.

5.7 Appendix - Details on Recovering Binary Images After Printing and Scanning

In Section 5.4.1, we described adding cross-shape marks at four corners and at four sides to serve as a ruler for registration purpose. Identifying the cross points of these

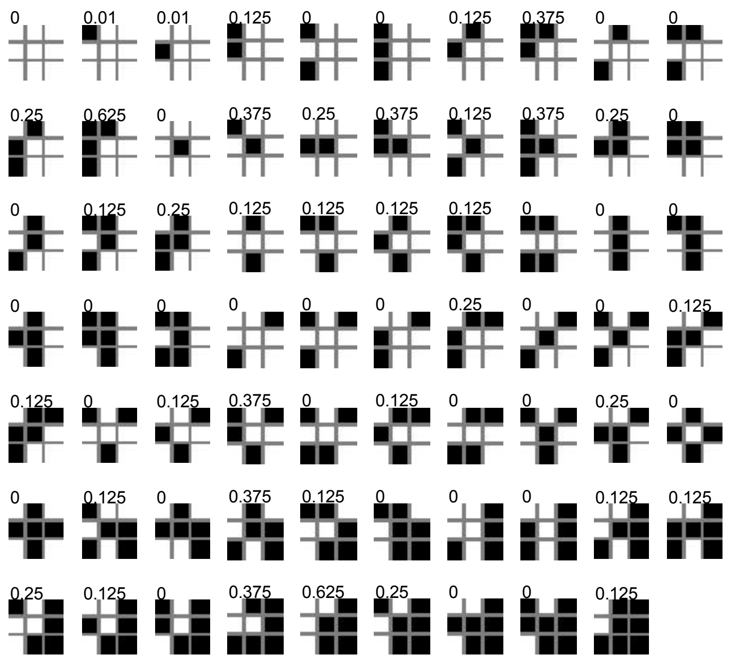


Figure 5.18: One possible flippability lookup table for 3×3 pattern, excluding symmetric cases of rotation, mirroring, and complement. Larger value indicates that the change of center pixel is less noticeable hence the change is more likely to be made for hiding information.

marks in a scanned image is the first step in recovering binary image from high quality printing and scanning. Here we propose a projection based approach under the assumption that the approximate region of the mark to be recovered has already been specified. For white background, the range should include the entire mark and preferably no other black pixels. A white outer layer of fixed width is added to the source image to facilitate the identification of the mark regions, as shown in Fig. 5.19. The approximate region containing the mark can be either manually specified via an interactive interface or automatically determined via pattern matching. For simplicity, the manual approach is used in our experiment, and reasonable effort is made during scanning so that the skewing of each mark is negligible.

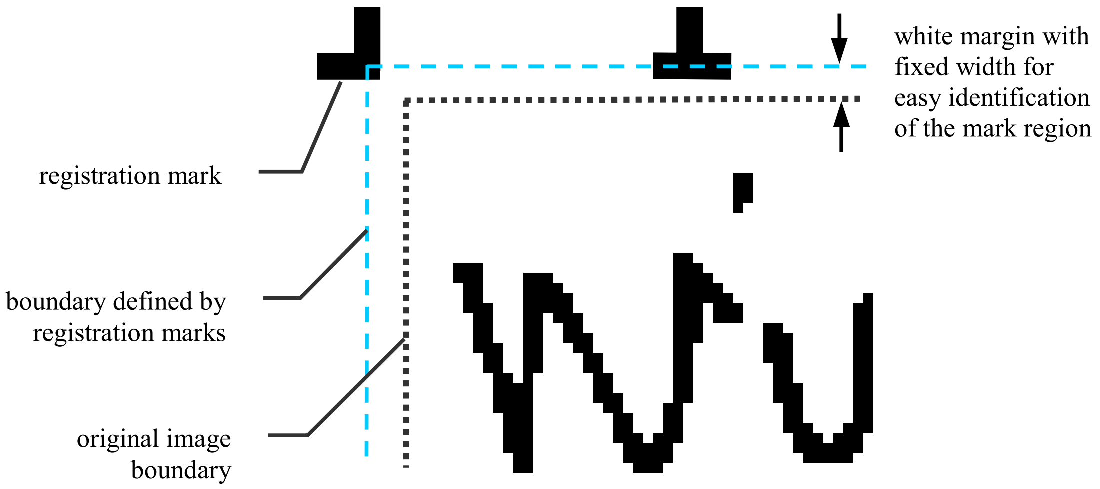

registration mark
boundary defined by registration marks
original image boundary
white margin with fixed width for easy identification of the mark region

Figure 5.19: Illustration of registration marks. A white outer layer of fixed width is added to facilitate the identification of the approximate region of each mark during print-and-scan.

To determine the cross point of a mark, we perform horizontal and vertical projections and get two profiles each of which has a unique "plateau" corresponding to the horizontal and vertical stroke, respectively. As illustrated in Fig. 5.20, the centers of the two plateaus determine the y- and x- coordinates of the cross point.

Using the identified cross points of registration marks, we can determine the skewing angle α of the entire scanned image, as illustrated in Fig. 5.21. The scaling factors can be estimated as follows: assuming the original image size has been determined as $N_w \times N_h$ and the scanned image size is $W \times H$, all measured in pixels [4]. We further assume the coordinate of the upper-left pixel in both the scanned image and

[4]Recall that we have impose constraints that the width and height of original binary image has to be multiples of 50. The actual multiplication factor can be determined by registration marks that serve as a ruler. Alternatively, the multiplication factor can be determined by estimating from the width of registration marks how many pixels in the scanned image correspond to one pixel in the original. Any additions such as the white outer layer in Fig. 5.19 need also be counted.

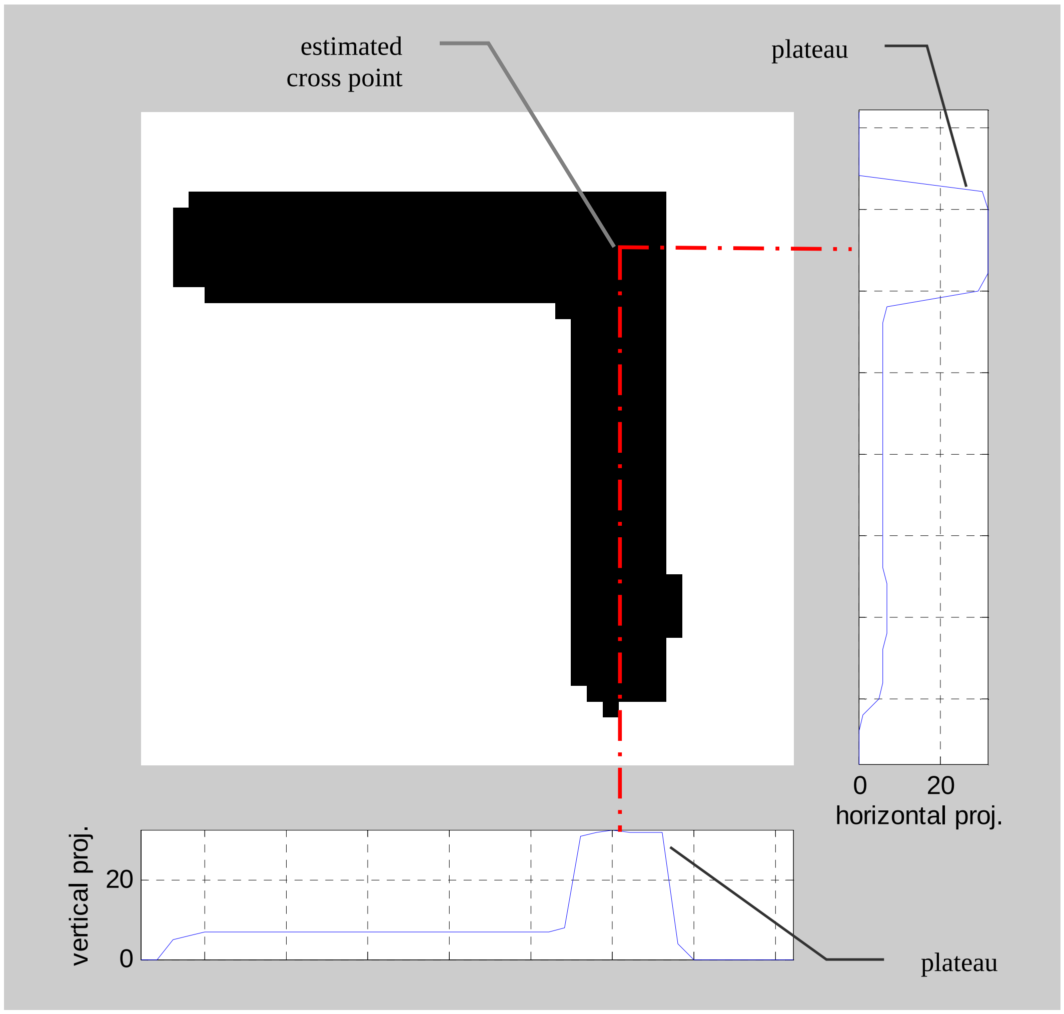


Figure 5.20: Determining the cross point of a registration mark by performing horizontal and vertical projection. The centers of the two projection plateaus are used as y- and x- coordinates of the cross point.

the original image is $(0, 0)$. Considering a pixel (x', y') in the original image, we would like to find the center of this pixel in the scanned version. We first perform a scaling operation:

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} \frac{W-1}{N_w-1} & 0 \\ 0 & \frac{H-1}{N_h-1} \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \frac{(W-1)x'}{N_w-1} \\ \\ \frac{(H-1)y'}{N_h-1} \end{bmatrix}. \tag{5.8}$$

where $(W - 1)$, $(H - 1)$, $(N_w - 1)$ and $(N_h - 1)$ are used because the coordinate of the first pixel starts from $(0, 0)$. We then perform rotation of $-\alpha$ degree and get the coordinate (x, y) of the estimated pixel center:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}. \tag{5.9}$$

If the estimation is well centered in the pixel and the scanning resolution is sufficiently high so that one original pixel corresponds to many scanned pixels (like those shown in Fig. 5.13), sampling at the estimated centers will recover the original image. Improvement may be done by considering the surrounding pixels as well as the grayscale information obtained from scanning, especially when a printed image has noisy boundaries and/or slightly blurred.

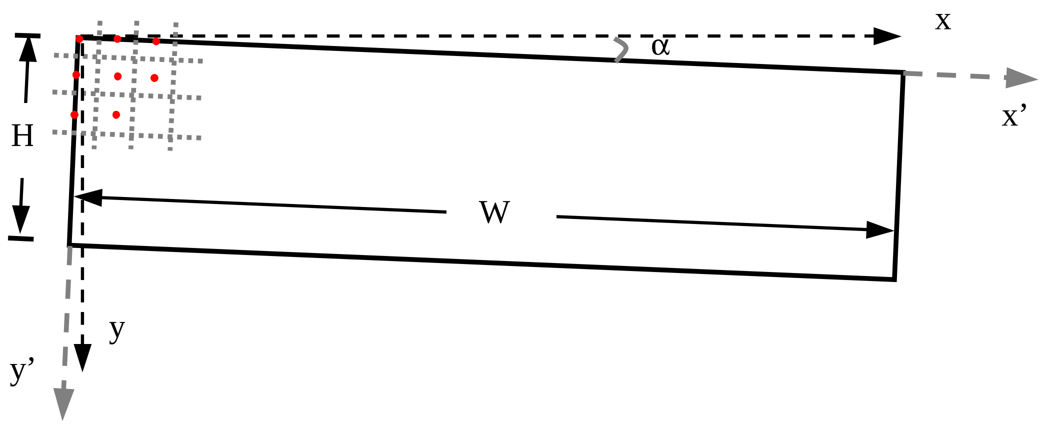


Figure 5.21: Using coordinate conversion to perform scaling and de-skewing. Coordinate x-y is for the scanned image, and Coordinate x'-y' is for the original image. Skewing angle between the two coordinates is represented by α; the lightly dotted squares represent original pixels, and the round dots are the corresponding centers.

Chapter 6

Multilevel Data Hiding for Image & Video

6.1 Introduction and Prior Art

We have mentioned that imperceptibility, robustness against moderate compression and processing, and the ability to hide many bits are the basic but rather conflicting requirements for many data hiding applications. The traditional way to handle this is to target at a specific capacity-robustness pair. Some approaches choose to robustly embed just one or a few bits [44, 46], while others choose to embed a lot of bits but to tolerate little or no distortion. However, a single robustness threshold generally overestimates the noise condition in some situations and/or underestimates in some other situations. It is desirable to design a data hiding system that is able to convey secondary data in high rate when noise is not severe and can also convey some data reliably under severe processing. This chapter proposes multi-level data hiding, which can achieve the above goal by combining multiple levels of embedding, each associated with different robustness-capacity tradeoff. The proposed scheme is motivated by a two-category classification of embedding schemes that have been discussed in Chapter3 and by a study on the performance of non-coherent detection of the popular spread spectrum watermarking to be presented in Sec. 6.2.1.

The work presented in this chapter can be used for such applications as robust annotation, content-based authentication, access/copy control, and fingerprinting. The design objective of this work is to survive common processing in transcoding and scalable/progressive transmission, such as compression with different ratio and frame rate conversion in the case of video. Malicious attack of making the watermark undetectable is not a major concern either because there is no incentive to do so in applications like annotation and authentication, or the threat can be alleviated by other means such as a well-determined business model.

6.2 Multi-level Embedding

An embedding scheme with a specific parameter setting generally target at a specific pair of [robustness, capacity]. The relation between the watermark-to-noise ratio (WNR) x and the maximum number of bits C that can be embedded is illustrated by solid lines in Fig. 6.1(a). Focusing on the two types of embedding mechanisms discussed in Sec. 3, the curve $C(x)$ is essentially a profile of the capacity curves for Type-I and Type-II in Fig. 3.7. For a watermarking algorithm targeting at surviving a specific level of noise (i.e., a specific watermark-to-noise ratio x_1), the maximum number of bits we can extract with very small probability of error under different actual noise conditions is indicated by the solid line in Fig. 6.1(b), which forms a step function with the jump occurring when the actual watermark-to-noise ratio (WNR) is x_1. By varying the targeted WNR, we get different curves for the amount of extractable data. These curves imply that targeting at surviving weak noise has the weakness of having no extractable data when the actual noise is strong, while targeting at surviving strong noise wastes the capability of conveying more data when the actual noise is weak. We would like to explore the possibility of choosing two WNR $[x_1, x_2]$ as our target where $x_1 < x_2$. More specifically, we apply the embedding strategy for surviving x_1 on a fraction α_1 of the total number of media components and apply the embedding strategy for surviving x_2 on the remaining portion α_2 with $\alpha_1 + \alpha_2 = 1$. The maximum number of extractable bit versus the actual noise conditions of this combined embedding follows a 2-step curve $C_{II}(x)$ in Fig. 6.1(c). The combined approach allows more bits to be extractable than both $C_{I,1}(x)$ (with targeted WNR as x_1) when $x \geq x2$, and $C_{I,2}(x)$ (with targeted WNR as x_2) when $x < x2$.

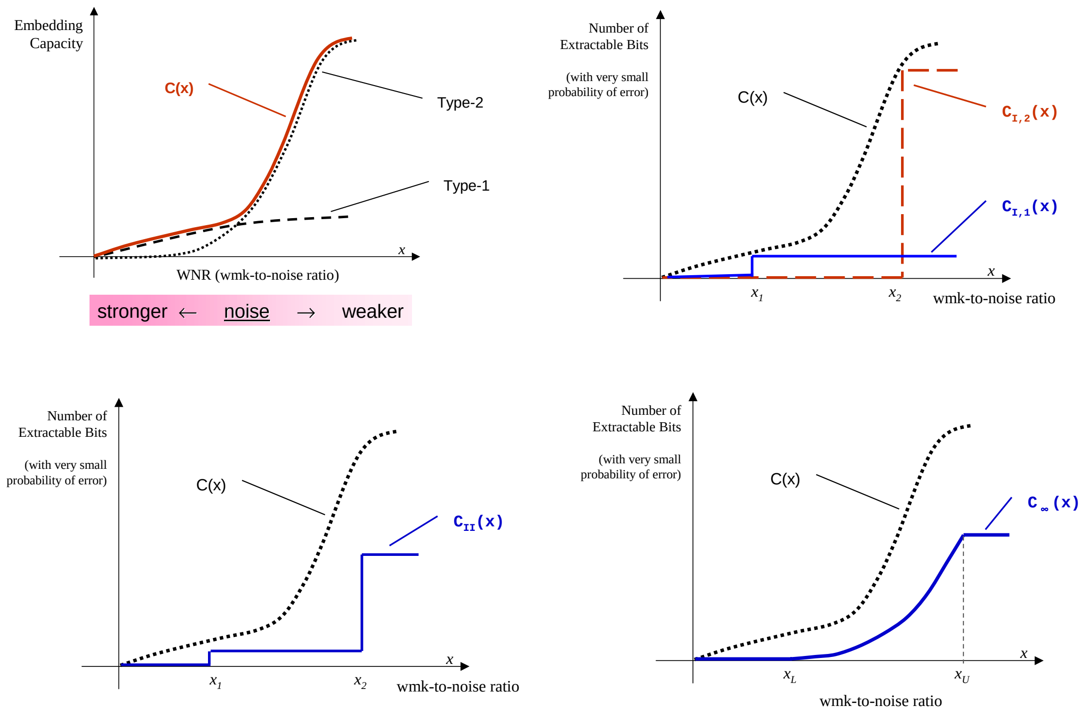


Figure 6.1: $\frac{a|b}{c|d}$ Amount of extractable data by single-level & multi-level embedding: (a) embedding capacity versus watermark-to-noise ratio, (b)-(d) the number of extractable bits by single embedding level, by two embedding levels, and by infinitely many embedding levels.

More generally, for a combined embedding with targeted WNR as $[x_1, x_2, ..., x_M]$ and the associated fraction as $[\alpha_1, \alpha_2, ..., \alpha_M]$ where $x_1 < x_2 < ... < x_M$ and $\sum_{i=1}^{M} \alpha_i = 1$, the maximum number of extractable bits $C_M(x)$ is:

$$C_M(x) = \begin{cases} \sum_{i=1}^{M} \alpha_i C_{I,i}(x_i) & \text{if } x > x_M; \\ \sum_{i=1}^{k} \alpha_i C_{I,i}(x_i) & \text{if } x_k < x < x_{k+1}, k = 1, ..., M-1; \\ 0 & \text{if } x < x_1. \end{cases} \tag{6.1}$$

If the amount of media components allocated for each embedding level x_k is $\alpha_i = \frac{1}{M}$, the above equation becomes

$$C_M(x) = \begin{cases} \frac{1}{M} \sum_{i=1}^{M} C_{I,i}(x_i) & \text{if } x > x_M; \\ \frac{1}{M} \sum_{i=1}^{k} C_{I,i}(x_i) & \text{if } x_k < x < x_{k+1}, k = 1, ..., M-1; \\ 0 & \text{if } x < x_1. \end{cases} \tag{6.2}$$

To study the case of large M, let $x_L = x_1$, $x_U = x_M$, and $x_{i+1} - x_i = (x_U - x_L)/(M-1)$. We further assume that the lower and upper bounds x_L and x_U are kept constant and the profile $C(x)$ is Riemann integrable. Then as M going to infinity, we have

$$C_\infty(x) = \begin{cases} \frac{1}{x_U - x_L} \int_{x_L}^{x_U} C(t)dt & \text{if } x > x_U; \\ \frac{1}{x_U - x_L} \int_{x_L}^{x} C(t)dt & \text{if } x_L \le x \le x_U; \\ 0 & \text{if } x < x_L. \end{cases} \tag{6.3}$$

This is illustrated in Fig. 6.1(d). We can see that combining many embedding levels gives a smoothly decayed amount of extractable information as the actual noise gets strong. We shall call this *multi-level embedding*. In practice, both the fractions, $\{\alpha_i\}$, and the targeted WNRs, $\{x_i\}$, can be non-uniform to allow different emphasis toward different noise conditions.

The graceful change of the amount of extractable information is desirable in many applications. The information to be embedded usually requires unequal error protection (UEP). Some bits, such as the ownership information and a small amount of control information facilitating the decoding of a larger amount of payload bits, are required to be embedded more robustly than others. For applications in which access or copy control policies, often in the form of a non-trivial number of bits, are embedded in the audio or video source, the policy cannot be enforced until it is decoded. Because lightly compressed audio/video has higher perceptual quality hence higher value than strongly compressed one, it is desirable to put the policy in effect much sooner to protect the rights of copyright holders. This implies that for a watermarked audio/video which could be later transcoded to various rates, the amount the extractable information should be adaptive with respect to the actual noise condition. Although there are a few works in literature proposing to perform multiple watermarking with different robustness (e.g., add a robust ownership watermark and a multiple-bit tracking label), the amount of the information conveyed by one or both levels is often no more than a few bits. Their applications are mainly for the control information versus user payload, which will be discussed in Sec. 6.4.2. In this chapter, we shall study the multi-level embedding problem in a more general sense, aiming at conveying several sets of data with different robustness, each set having a non-trivial number of bits.

While the two types of embedding discussed in Chapter 3 can be used to realize multi-level data hiding, a key issue is to determine what part of the host signal to be used for each capacity-robustness level. We have performed an analysis on the performance of non-coherent detection of spread spectrum approach which has good tradeoff between robustness and imperceptibility but has limited data hiding capacity under non-coherent detection. This analysis, summarized as follows, provides a guideline to the partitioning of host signal spectrum for multi-level data hiding.

6.2.1 Justification of Spectrum Partition

In Chapter 3, we have discussed the hypothesis testing formulation of Type-I additive watermarking:

$$\begin{cases} H_0 : Y_i = -S_i + M_i \quad (i = 1, ..., n) \qquad \text{if } b = -1 \\ H_1 : Y_i = +S_i + M_i \quad (i = 1, ..., n) \qquad \text{if } b = +1 \end{cases} \tag{6.4}$$

where $\{S_i\}$ is a deterministic known sequence (i.e., the *watermark*), b is one bit of data to be embedded and is used to antipodally modulate S_i and is equally likely to take "+1" and "-1" values, M_i is the noise, and n is the number of samples/coefficients to carry the hidden information. In literature, M_i is usually modeled as i.i.d. gaussian distribution $N(0, {\sigma_M}^2)$ for simplicity. The optimal detection statistics under this assumption is a (normalized) correlator with S_i:

$$T_N = \underline{Y}^T \underline{S} / \sqrt{{\sigma_M}^2 \cdot ||\underline{S}||^2} \tag{6.5}$$

This test statistics is gaussian distributed with unit variance and the mean

$$E(T_N) = b \cdot \sqrt{n \cdot (\frac{1}{n}||\underline{S}||^2) / {\sigma_M}^2} \tag{6.6}$$

According to Bayesian rule, setting the threshold to zero gives the minimum probability of error. That is, we compare T_N with zero, and decide on H_1 if it is positive and on H_0 otherwise. The probability of error is $\mathcal{Q}(E(T_N))$, where $\mathcal{Q}(x)$ is the probability of $P(X > x)$ of a gaussian random variable $X \sim N(0, 1)$.

In non-coherent detection, M_i consists of the interference from host media and the noise processing/attack. The high power of host media causes a large value in ${\sigma_M}^2$, reducing $E(T_N)$ and increasing the probability of detection error. Because the host media serves as a major noise source in non-coherent detection, researchers have been studying ways to reduce the interference from host signal. One proposal, based on the observation that the low band coefficients of the host media generally have much higher power than the mid-band, suggests marking only mid-band coefficients to reduce the interference [62]. The watermark detector there is the commonly used correlator. The "mark mid-band only" proposal conflicts with the common understanding in detection theory that under gaussian noise, the detection performance should be enhanced with more independent observations. The confliction comes from the noise model. While the i.i.d. gaussian distribution is the noise condition that leads to the minimum Euclidean distance or maximum correlation being the optimal detection [2], this noise model is not always the case in practical applications, for example, different bands of block DCT coefficients have different variance. The observation that low-band coefficients have higher power than the mid-band in the above-mentioned prior work is a reflection of this fact. A more realistic model assumes the noise being independent gaussian distributed but with different variance from band to band. For this kind of noise, the simple correlator is no longer an optimal detector. Instead, we should first normalize the observations by the corresponding standard deviation to make the noise distribution i.i.d, then take the correlation. That is,

$$T'_N = \sum_{i=1}^{N} \frac{Y_i \cdot S_i}{{\sigma_{M_i}}^2} \Big/ \sqrt{\sum_{i=1}^{N} \frac{{S_i}^2}{{\sigma_{M_i}}^2}} \tag{6.7}$$

This optimal detector can be understood as a weighted correlator with more weight given to the less noisy components.

There are two implications associated with the weighted correlator. On one hand, leaving any band out in embedding will reduce the magnitude of the detection statistics hence enhance the probability of error. This result is different from what was claimed in literature because the detector there, though simple and very popularly used, is not optimal for non-i.i.d gaussian noise. On the other hand, the contribution from the noisy bands is little, as they have been scaled by the inverse of the noise variance. The noisy bands include the low bands which have strong interference from original host signal. Leaving them out in spread-spectrum embedding will not cause significant performance degradation in detection. For a more general gaussian noise model in which the components of the host media and/or the noise may be dependent, whitening and normalization should be performed before applying the minimum Euclidean distance detector or maximum correlation detector [2].

This analysis has been confirmed by our experimental study on 114 natural photo images. We implemented the block-DCT domain spread spectrum algorithm proposed by Podichuk-Zeng [46] and use the *q-statistics* proposed by Zeng-Liu [53] in detection. The q-statistics is a correlation statistics with variance being normalized to 1. We shall denote q' and q as the detection statistics with and without weighting according to an estimation of the total noise conditions:

$$\begin{aligned} q = & \frac{M_Z}{\sqrt{V_Z/n}} && (6.8) \\ & \text{where} \quad Z_i = Y_i \cdot S_i, M_Z = \frac{1}{n}\sum_{i=1}^{n} Z_i, V_Z = \frac{1}{n-1}\sum_{i=1}^{n}(Z_i - M_Z)^2; \\ q' = & \frac{M_{Z'}}{\sqrt{V_{Z'}/n}} && (6.9) \\ & \text{where} \quad {Z'}_i = Y_i \cdot S_i/c_i, M_{Z'} = \frac{1}{n}\sum_{i=1}^{n} {Z'}_i, V_{Z'} = \frac{1}{n-1}\sum_{i=1}^{n}({Z'}_i - M_{Z'})^2. \end{aligned}$$

The weight $\{c_i\}$ reflects the impact of the noise variance term in Eq. 6.7. The noise variance is not easy to be estimated accurately because (1) the precise power of the cover signal is unknown, and (2) the variance of processing noise varies dramatically depending on what kind of distortion/attack is applied to the media. For the first problem, one may make an estimation based on the statistics of the current test media; for the second problem, a set of known signal may be added to predetermined locations of the cover signal, serving as a training sequence to facilitate the noise estimation [38]. In our experiment, we choose $\{c_i\}$ based on the variance of host signal and potential processing noise of the frequency band which Y_i is in. They are empirically determined via a statistical study on a number of natural images.

Using both q and q' as detection statistics, we studied the above-mentioned 114 natural images, each of which was tested using 3 different watermarks. We vary the frequency band from which watermark begins to be put in and this change is in a zigzag order shown in Fig. 6.2. The q and q' values are computed under several distortion conditions including zero distortion, JPEG with different quality factors, and low pass filtering [1]. The average normalized q and q' are shown in Fig. 6.3. It can be seen that the q value assumes maximum when the band from which the embedding starts is around $6to10$ (i.e., the third diagonal lines of AC coefficients), and q decreases when either more or less frequency bands are involved. As predicted by our analysis, q' gives larger value hence smaller probability of error than q. In addition, q' is monotonically decreasing when fewer bands are used in embedding, but the decrease when leaving the lowest 5 bands out is insignificant.

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

Figure 6.2: Zig-zag ordering of DCT coefficients in an 8×8 block.

[1]For each image, we also normalize q and q' with respect to the number of *embeddable* coefficients that can be watermarked without introducing perceptual distortion so that the detection values for smoother images and for more complex images are comparable.

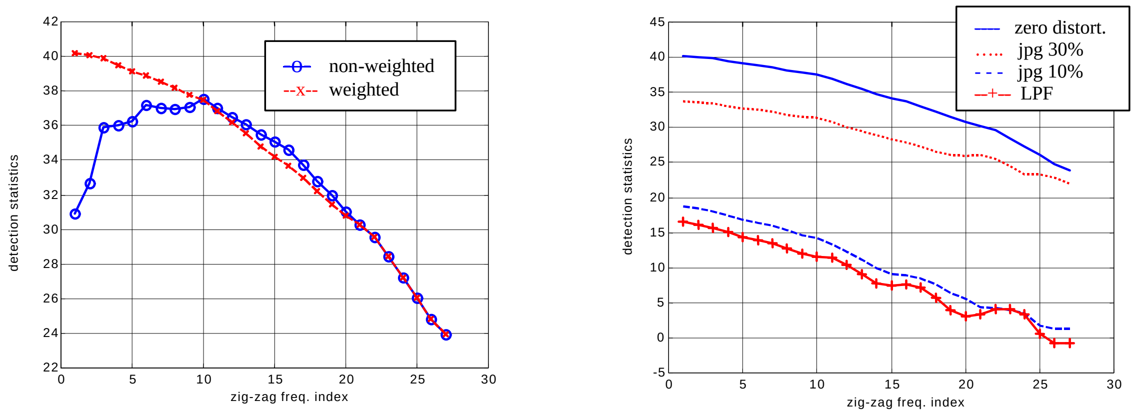


Figure 6.3: $\underline{a|b}$ Comparison of average detection statistics for the correlator and the weighted correlator from 114 natural images, both are normalized: (a) detection statistics of non-weighted and weighted correlator q and q' under zero distortion, with x-axis indicates the frequency band from which watermark starts to be put in using a zigzag order; (b) detection statistics of weighted correlator q' under different distortions.

6.3 Multi-level Image Data Hiding

With the analysis in previous section on non-coherent detection of spread spectrum watermarking and the two-category classification of embedding mechanisms in Chapter 3, the answer regarding which part of the host signal should be used for each embedding level becomes clear. The multi-level data hiding shall take advantage of the "sweet points" of both types of schemes: we apply Type-I spread spectrum embedding to mid-band coefficients to reach high robustness at a cost of capacity, and apply Type-II to low-band to reach high capacity with moderate robustness. With such multi-level embedding, we can hide many bits and decode them successfully when image experiences little or moderate distortion, and can convey a few bits very robustly even when image is distorted significantly.

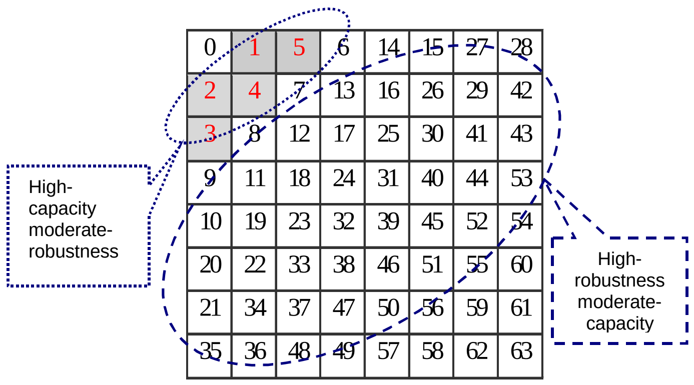
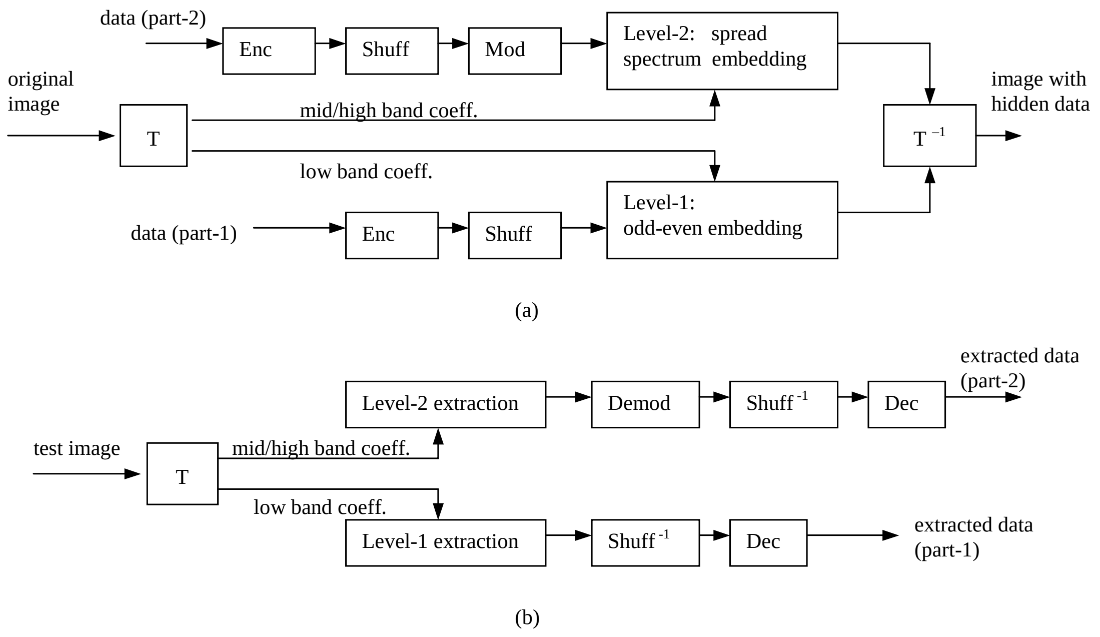


Figure 6.4: Illustration of Two-level Data Hiding in Block-DCT Domain.

A two-level image data hiding in block-DCT domain (Fig. 6.4) is demonstrated here in detail, while it is possible to extend the idea to more than two levels. The first level uses odd-even embedding, an example of Type-II to embed the first set of data in low band, though it is straightforward to replace it with other schemes. Based on the justification regarding the spectrum partition, we take the first two diagonal lines of AC coefficients as low band, i.e., the first 5 AC coefficients, shaded in Fig. 6.4. In addition, we perform the embedding with quantization to enhance robustness, as discussed in Chapter 3. The quantization step sizes we have used are equivalent to JPEG compression with quality factor 50%. During this process, the just-noticeable-difference (JND) is computed and any embedding which may result in noticeable difference is withheld. The second level uses spread spectrum embedding to hide the second set of data in mid-band. Antipodal modulation is used here, i.e., to add or subtract a spread spectrum signal to denote one bit (Eq. 6.4). The watermark strength is also adjusted by JND. Both sets of data may be error correction encoded, and each bit is embedded in a region that is not overlap with those for other bits. Shuffling is performed on this TDMA type of embedding to handle the uneven embedding capacity from region to region, as discussed in Sec. 3.4 and Sec. 4.

The block diagram in Fig. 6.5 summarizes the overall system. Before we present the experimental result, we shall discuss briefly a human visual model refined from those available in literature. The model has been used in our system for computing the JND.



Notation

T:	transform	T^{-1}:	inverse transform
Enc:	error correction encoding	Dec:	error correction decoding
Mod:	modulation	Demod:	demodulation
Shuff:	shuffling	$Shuff^{-1}$:	inverse shuffling

Figure 6.5: Block diagram of multi-level data hiding for images: (a) embedding process, (b) extraction process.

6.3.1 Refined Human Visual Model

Almost all watermarking algorithms on grayscale and color images utilize human visual model to ensure imperceptibility, either implicitly or explicitly. In a classic work of spread spectrum watermarking, Cox *et al.* [44] pointed out the importance to embed watermark in perceptually significant components to achieve robustness and made use of the perceptual tolerance of minor changes to embed watermark in those significant components. In their implementation, the watermark is embedded in the DCT domain of the image and a simplified scaling model is used to set the watermark power about a magnitude lower than that of the cover image. By explicitly utilizing human visual models known as frequency-domain masking, the works by Podichuck-Zeng [46] and Swanson *et al.* [48] embed the watermark in block-DCT domain and use the masking model to tune the watermark strength in each block. Swanson *et al.* also incorporated spatial-domain masking in their work.

The block DCT domain is a popular embedding domain in literature. It is compatible with the commonly used image and video compression techniques such as JPEG, MPEG, and H.26x, making it possible to perform compressed domain embedding and to make use of various techniques already developed for that domain (such as human visual model for JPEG compression [100, 101]). The block-based domain also has the advantage of fine-tuning watermark strength for each local region to achieve a good tradeoff between imperceptibility and robustness against distortion. However, this popular domain has a few major weaknesses both on imperceptibility and on robustness. We shall focus on the perceptual problem in this section and postpone the discussion regarding robustness till Chapter 9. The perceptual problem with block DCT domain embedding is the ringing artifacts introduced on edges. The previously proposed frequency-masking model has not taken this issue into account [46]. The only way for those models to reduce artifacts is to attenuate the whole watermark signal, which leads to less robustness and data hiding capacity. Tao *et al.* proposed to apply block classification to reduce artifacts by classifying image blocks into six categories (i.e., edge, uniform with moderate luminance, uniform with either high or low luminance, moderately busy, busy, and very busy), and adjusting the watermark strength differently for each category [50]. The classification, involving enumerations of many possibilities, could be computationally expensive. We propose a refined human visual model with less computational complexity than [50] while introducing fewer artifacts than [46].

Before presenting the details of our refinement, we shall explain a bit more about the frequency domain masking model used in [46], on top of which our refinement is applied. The masking model is based on the following observations of human visual system: first, different frequency bands have different just-noticeable levels, and generally the just-noticeable-difference (JND) in high frequency bands is higher than that in low bands; second, in a specific frequency band, a stronger signal can be modified by a larger amount than a weak signal without introducing artifacts. Because the blocks with edges and textures have larger coefficient values (in magnitude) than the smooth blocks, the JND of the non-smooth blocks obtained by this model is generally larger than the smooth ones. The model reflects little difference between the two non-smooth cases, namely, edge block and texture block. These two cases, however, have significant visual difference: with modification of the same strength in block DCT domain, the artifacts is more likely to be revealed in an edge block than in a texture one. The possible reason lies on two aspects: (1) the modification in block-DCT domain is equivalent to add or subtract the corresponding 2-D DCT basis images shown in Fig. 6.6, and (2) busy non-structural pattern on or near a structural feature such as an edge is quite obvious to human eyes, while many textures themselves involve more or less random patterns hence the add-on busy artifacts get swamped and undistinguishable to eyes. Our refinement tries to distinguish edge and texture blocks so that we can adjust the preliminary JND computed by the simple masking model to achieve better invisibility. In other words, we try to protect the edge block from over-modified. Furthermore, we observed that compared with the edges between two non-smooth regions, the boundary block between a smooth region and another region (either smooth or not) should be protected more, even though the edge in that block may be soft and weak.

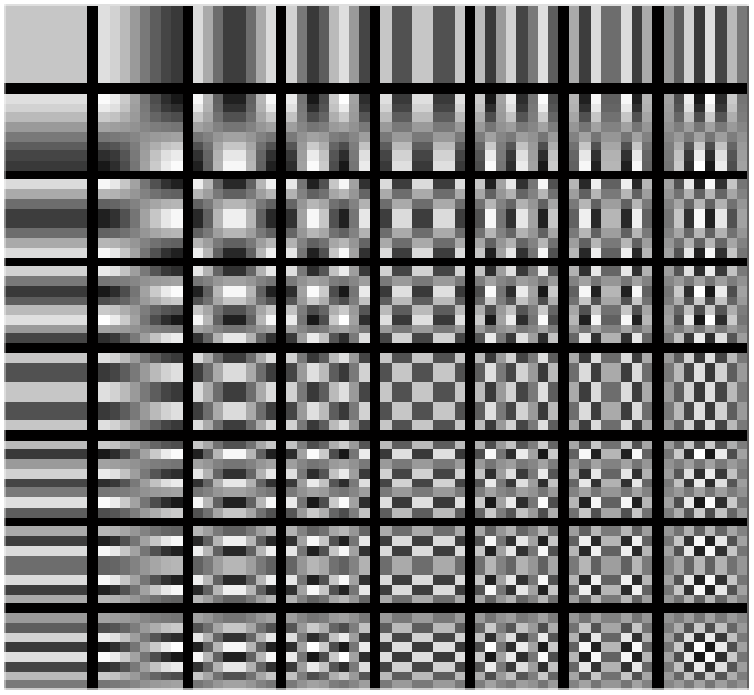
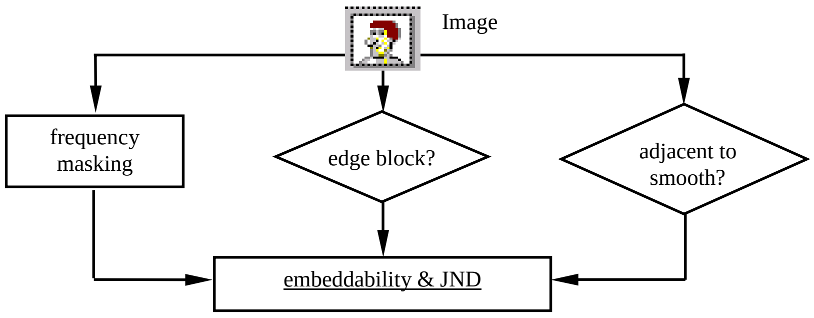


Figure 6.6: 2-D DCT basis images of 8×8 blocks. The upper-left corner is the DC basis image.

The refined HVS model, illustrated by the block diagrams in Fig. 6.7, includes



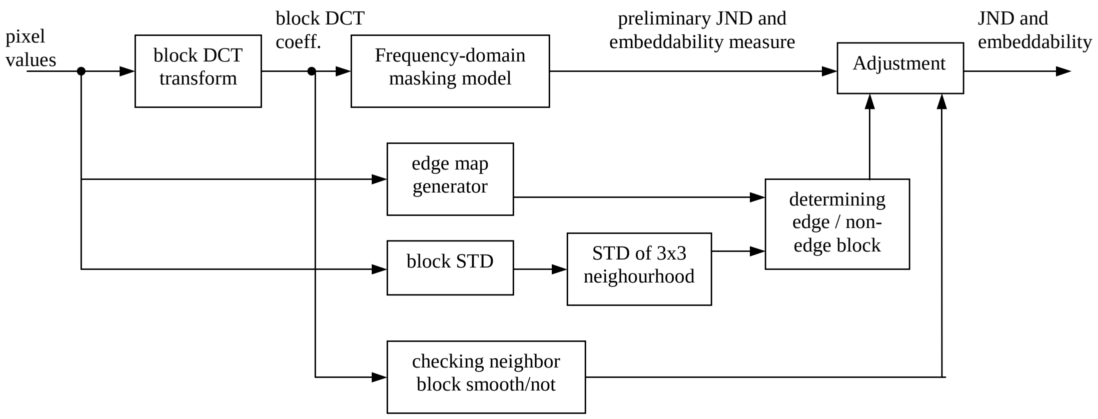

pixel values
block DCT transform
block DCT coeff.
Frequency-domain masking model
preliminary JND and embeddability measure
Adjustment
JND and embeddability
edge map generator
determining edge / non-edge block
block STD
STD of 3x3 neighourhood
checking neighbor block smooth/not

Figure 6.7: Block diagram of the refined 3-step HVS model: (top) basic modules, (bottom) detailed procedures.

the following three steps:

Step-1: Frequency domain masking

The first step of our perceptual analysis makes use of block-DCT domain masking result and computes preliminary embeddability and just-noticeable-difference (JND) for each coefficient, which determine whether a coefficient can be modified and if so, by how much amount it can be changed. As mentioned above, this step is similar to what has been proposed in [46] and forms a basis for further adjustment.

Step-2: Edge-block detection

The algorithm we use for determining which block contains edges has two sub-steps. First, we use edge detection algorithm (e.g., Haar filtering) to produce an edge map. Second, we compute the standard deviation (STD) of pixel values in each block (i.e., we obtain one value for each block, measuring its activeness), then compute the standard deviation of these standard deviations in a neighborhood (e.g., 3 blocks by 3 blocks). The second sub-step helps to eliminate many unwanted edges obtained in the first sub-step, such as those in texture regions.

The reasoning for such double STD measure is that in a texture region, although the STD of each block is large, the STDs of adjacent blocks in the same texture region are similar hence does not have large deviation when computing the second round STD. On the other hand, the STD of a edge block is likely to be very different from the majority of its neighbor blocks. Double STD computation can be easily implemented.

At the end of this step, we combine the edge map with the double STD result and output an edge measure that indicates whether there is an edge across the block and if so, how strong the edge is. The edge measure is then used to adjust the just-noticeable-difference. The adjusted JND will ultimately be used to control the watermark strength so that weaker watermark will be applied to edge blocks than to texture ones.

Step-3: Identifying blocks adjacent to smooth region

As we mentioned, artifacts by block-DCT domain embedding are more visible in blocks that are adjacent to smooth region than in other blocks, even if the block contains such weak edges that the watermark may not be attenuated sufficiently by Step-2. A relatively stronger watermark can be added if an edge block is not adjacent to smooth region than the contrary case. To protect the blocks adjacent to smooth region from artifacts, we attenuate the JND of a block that is adjacent to smooth block so that the watermark applied there will be weaker. The smoothness of a block is determined by the magnitudes of its AC coefficients.

Fig. 6.8 demonstrates the difference between applying step-1 only (similar to HVS model in [46], and denote as "HVSpz") and our new 3-step model (denoted as "HVSedge"), for both Lenna image (containing lots of smooth region and sharp edges) and Baboon image (containing lots of textures and also a dark border at bottom). The image quality and detection statistics of single additive spread-spectrum watermark are summarized in Table 6.1. Note that other parameters such as scaling factor for watermark strength are kept same in this experiment so that the only difference is the HVS model. From the image details shown in Fig. 6.8, we can see that the proposed 3-step HVS model has fewer artifacts, yet the detection statistics is still high enough to encode multiple bits.

Image	HVS type	Detection statistics	PSNR (dB)	Subjective image quality
Lenna (512x512)	HVS edge	25.50	42.51	good img quality
	HVS pz	35.96	40.76	artifacts along edges (e.g., shoulder)
Baboon (512x512)	HVS edge	58.49	33.59	good img quality
	HVS pz	62.81	33.10	obvious artifacts along bottom dark border

Table 6.1: Comparison of the proposed HVS model ("HVSedge") and a model used by Podichuk-Zeng ("HVSpz").

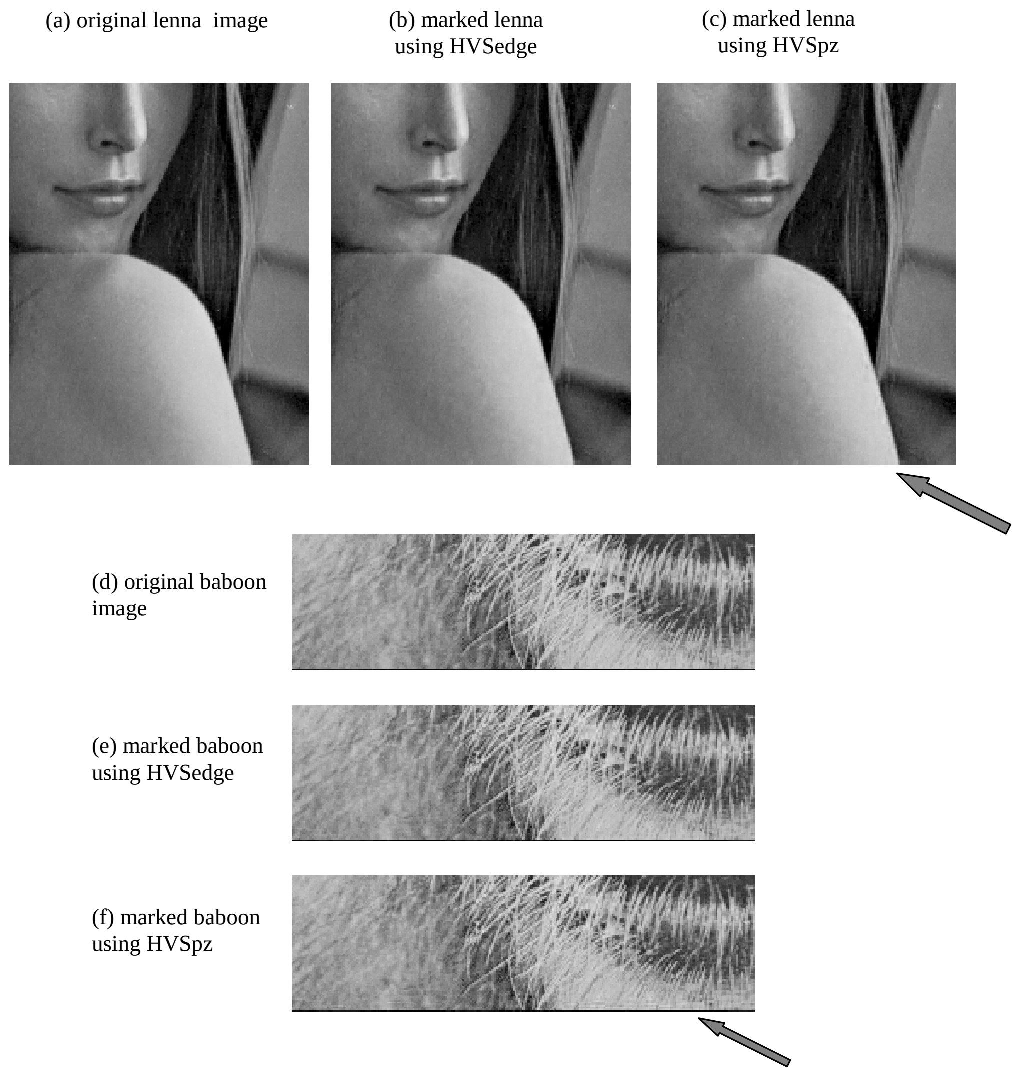

(a) original lenna image
(b) marked lenna using HVSedge
(c) marked lenna using HVSpz
(d) original baboon image
(e) marked baboon using HVSedge
(f) marked baboon using HVSpz

Figure 6.8: Examples of images watermarked by the proposed HVS model ("HVSedge") and a model used by Podichuk-Zeng ("HVSpz"). The artifacts by HVSpz model are indicated by gray arrows.

6.3.2 Experimental Results

The two-level data hiding scheme has been applied to 512$\times$512 Lenna image (Fig. 6.9). With respect to the original unmarked image, the PSNR of the image with hidden data is 42.5dB. Incorporating error correction codes and shuffling, we embed a 32×32 binary pattern of PINTL-Matsusita logo in low band and it can be extracted correctly even when the image experiences JPEG compression of quality factor 45%. We also use spread spectrum approach to embed the ASCII code of a character string "PINTL" in mid band, which can be extracted without error even when the image is JPEG compressed with quality factor 20% or blurred. The embedding rate can be higher for images which contain more complex contents and/or which are larger than Lenna images, and can be lower otherwise. For example, we can embed a longer string of "Panasonic Tech." and 32×32 pattern in baboon image [2], as shown in Fig. 6.10. With the help of the refined human visual model discussed in the previous section, the marked image has no visible artifacts and has a PSNR of 33.6dB with respect to the original image. The lower PSNR of the baboon image than the Lenna image is another indication that stronger watermarks can be embedded invisibly in images with more textures.

6.4 Multi-level Video Data Hiding

Video has been the center of interests in the world of multimedia. If a picture is considered better than 1000 words, a video or a sequence of image frames can convey even more information. Content providers such as movie industry and news agencies

[2]The embedding rate of baboon image in Type-II level can be higher than Lenna image. For the ease of visualizing the hidden data, our experiment hid the same PINTL-Matsusita pattern of 1024 bits in both images.

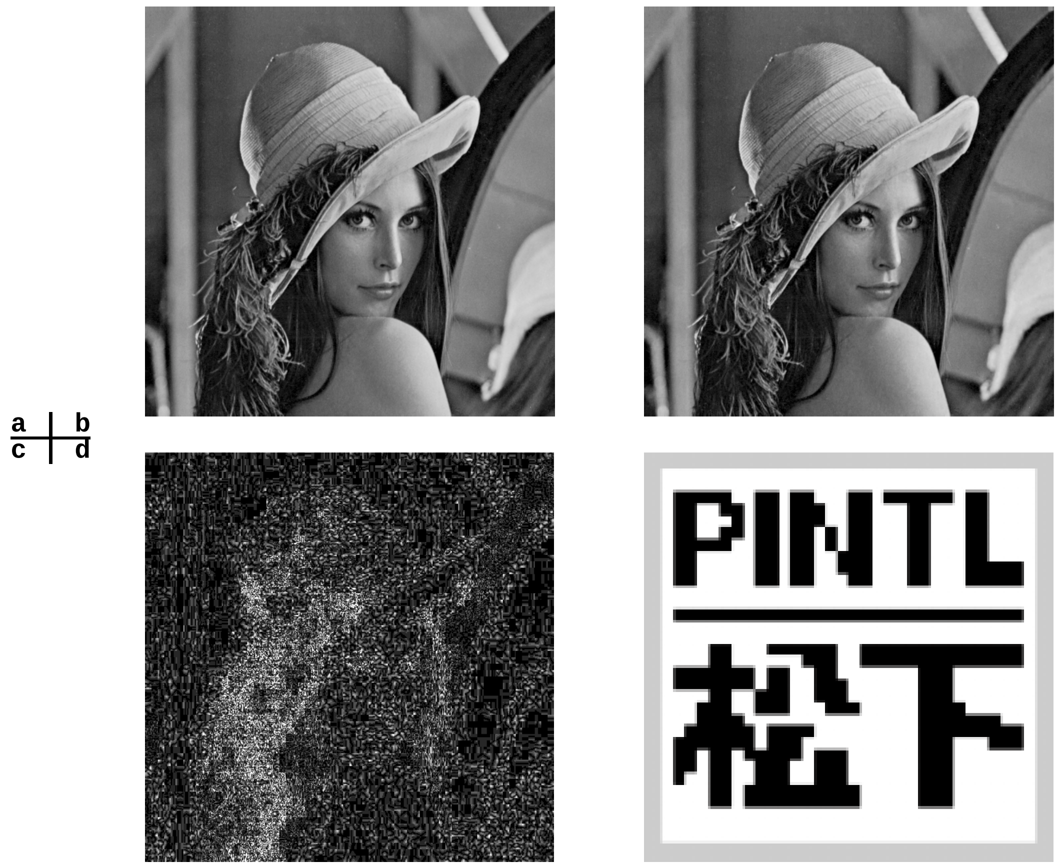


Figure 6.9: $\frac{a|b}{c|d}$ Multi-level data hiding for Lenna image (512x512). (a) original image; (b) image with hidden data; (c) enlarged difference between (b) and (a) with black denoting zero difference; (d) extracted 32x32 PINTL-Matsusita pattern embedded via high-capacity embedding level.

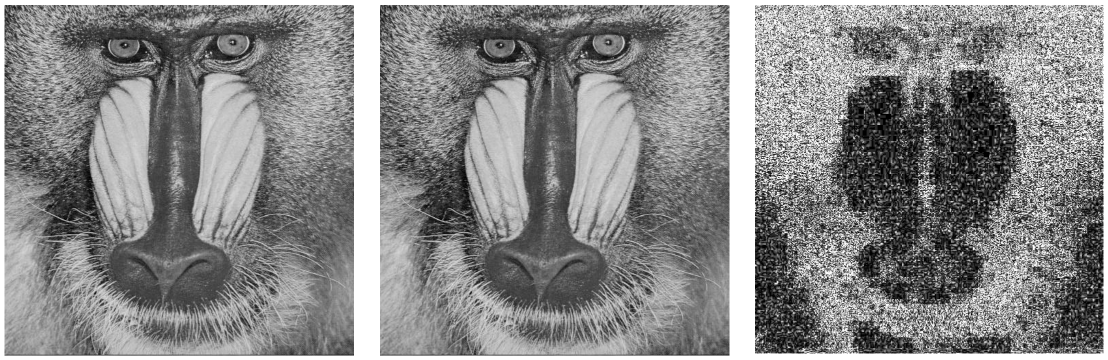


Figure 6.10: $\underline{a|b|c}$ Multi-level data hiding for Lenna image (512x512). (a) original image; (b) image with hidden data; (c) enlarged difference between (b) and (a) with black denoting zero difference.

have imposed high demand of the ownership protection, alteration detection, access control, and source tracking (fingerprinting) on digital video. Embedding many bits in high-quality digital video achieving both imperceptibility and robustness can be used for these purposes. Besides the large data volume and high computation complexity involved in processing video, we shall discuss in this section other challenges of video watermarking and propose solutions. Specifically, extending the multi-level data hiding from image to video, we need to address several issues, including the embedding domain and the uneven embedding capacity. A summary of our system design is shown in Fig. 6.11. The details of the modules that perform data embedding/extraction to/from each frame are similar to the multi-level image data hiding.

6.4.1 Embedding Domain

Because consecutive video frames look similar (except at scene change and fast motion) and each frame can be viewed as a standalone unit, it is possible to add or drop

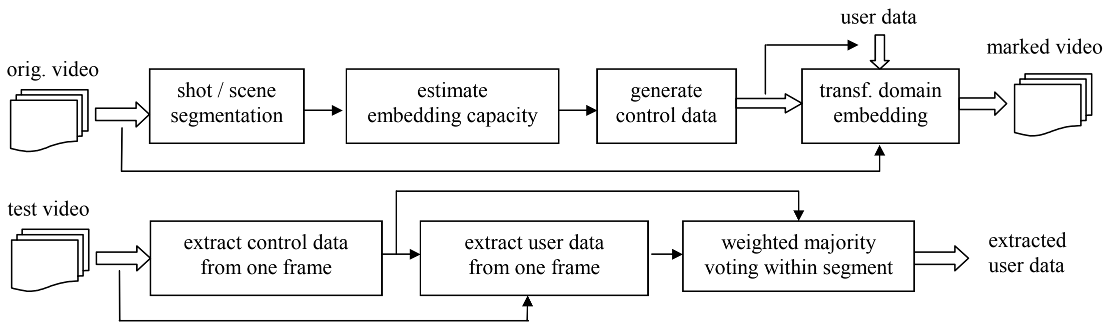

user data
marked video
orig. video
shot / scene segmentation
estimate embedding capacity
generate control data
transf. domain embedding
test video
extract control data from one frame
extract user data from one frame
weighted majority voting within segment
extracted user data

Figure 6.11: Block diagram of the proposed video data hiding system.

some frames, or switch the order of adjacent frames without causing noticeable difference. These manipulations are potential attacks that have to be considered in robust data hiding systems. Adding redundancy and/or searching for frame-jitter invariant domain are common ways to handle these attacks. We focused on the redundancy approach due to its effectiveness and computational simplicity.

In particular, we adopt two methods to handle frame jittering, as illustrated in Fig. 6.12. The first one is to partition a video into segments, each of which consists of similar consecutive frames, then to hide the same data in every frame of one segment. This approach allows us to tolerate frame dropping which involves a small number of isolated frames. Repetition also helps to combat noise from processing/attacks so as to achieve higher detection accuracy. Extraction can be done via weighted majority voting, giving larger weights to the frames experiencing less distortion.

For combating frame reordering, frame addition, and frame dropping of larger units, the first method alone is not sufficient. Our second method addresses these issues by embedding a shortened version of segment index in each frame. This information is called *frame synch* and is part of the control bits that will be discussed next. The frame synch information can assist us in detecting and locating frame jittering. When used with redundancy approaches such as repeatedly embedding data in separate parts of a video, this method can enhance the robustness against attacks of frame reordering and dropping.

In short, we apply image data hiding approach to each video frame and hide the same user data as well as frame synch index in every frame of the same segments to combat frame jittering.

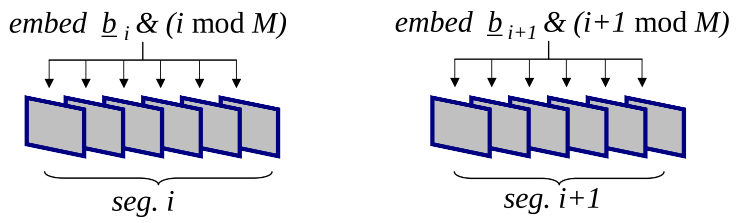

embed $\underline{b}_i$ & (i mod M)
embed $\underline{b}_{i+1}$ & ($i+1$ mod M)
seg. i
seg. $i+1$

Figure 6.12: Methods for handling frame jittering by the proposed video data hiding system.

6.4.2 User Data vs. Control Data

In practice, there are two layers of data we would like the video to carry. Layer-1 is the actual information (possibly error correction encoded) for the purpose of copy protection (e.g., a copyright label of "(c) Panasonic 2000"), access control, authentication, annotation, and other applications. We shall refer to the data for this layer as *user data*. Layer-2 is the other information that needs to be conveyed to facilitate extraction, for example, the information regarding synchronization and the number of bits embedded in each frame. This layer's data is referred as *control data*.

To achieve the multiple tradeoff levels between robustness and capacity, we use multi-level embedding for user data and adopt TDMA approach with shuffling to hide them. For control data, we use the robust spread spectrum embedding because in general the total number of control bits is not large but their accuracy is critical for extracting user data. The random vectors for hiding each control bits are orthogonal to each other and are also orthogonal to those used for embedding user data.

Typical control data include the total number of embedded bits of user data, frame synch information, and a constant watermark vector that can be used for image registration. We shall take frame synch as an example to demonstrate the embedding of control data. Frame synch, as introduced before, is a shortened video segment index to help combating frame jittering attack. The range of frame synch index is from 0 to $K-1$, and the i-th segment is labeled with an index $\text{mod}(i, K)$. A tradeoff has to be made here: on one hand, the larger the K is, the more accurately the segments' ordering information can be obtained, hence the higher tolerance with respect to frame jittering; on the other hand, the control data is considered as overhead hence its amount should be limited, which implies that a small K is desirable. Considering these factors, we choose $K = 8$. That is, we index video segments in a round-robin fashion from 0 to 7 and each index is represented by 3 bits ($2^3 = 8$). For these 3 bits, the orthogonal modulation discussed in Sec. 3.4 is used due to its energy efficiency: if the synch index is j, we embed the j-th sequence of K pre-selected orthogonal random sequences. As long as K is not too large, we can find sufficiently many orthogonal sequences and keep a reasonable detection computational complexity.

6.4.3 Variable vs. Constant Embedding Rate (VER vs. CER)

We have mentioned that one issue to be addressed when extending the multi-level data hiding idea from image to video is the uneven embedding capacity both from region to region within a frame and from frame to frame. Because of the potentially large overhead by using VER, we alleviate the intra-frame unevenness via CER and the shuffling techniques as explained in Part-I. Regarding inter-frame unevenness, we adopt VER approach by embedding additional side information because the overhead can be made relatively small compared to the total number of bits that can be embedded in most frames. The details of VER implementation are explained below.

It has been observed that the number of bits that can be embedded in each frame generally varies quite significantly, from very few bits for smooth/uniform frames to dozens or even hundreds bits for frames containing lots of contents and textures. Representing the side information of how many bits being embedded would need many bits on average. Denoting the number of bits embeddable in a given frame as C, it is desirable to (1) assign a shorter representation or code to smaller values of C to reduce the relative overhead, (2) reduce the average code length for representing C. We adaptively determine the embedding rate according to the relationship between the estimated capacity $\hat{C}$ of the current frame and two thresholds T_1 and T_2 where $T_1 < T_2$, and convey this side information using spread spectrum embedding that is orthogonal to other embedded data. The estimation $\hat{C}$ is obtained from the energy of DCT coefficients, the number of embeddable DCT coefficients, and the detection statistics of an embedding trial that hides a single spread spectrum watermark in the video frame. As shown in Table 6.2, we do not embed any user data when the $\hat{C}$ is small, and embed a predetermined small number of bits when $\hat{C}$ is moderate. The overhead for conveying these two cases is well constrained by two orthogonal spread spectrum sequences $+\underline{s}_1$ and $+\underline{s}_2$. When $\hat{C}$ is large, we switch to a full VER mode signaled by a spread spectrum sequence $-\underline{s}_1$, and use orthogonal modulation via several other spread spectrum sequences to convey the number of bits embedded in the frame to ensure correct decoding of user payload data. To reduce the overhead for conveying and detecting this side information, we require that the number of bits embedded can only be chosen from a pre-determined finite set, for example, {0, 1, 4, 8, 16, 40, 75, 100, 150, 200 bits}. Little embedding capacity will be wasted if the finite set is wisely chosen.

Table 6.2: Adaptive embedding rate for video.

estimated capacity $\hat{C}$	*user data*	*control data*
$\hat{C} \leq T_1$	not to hide any bits	add $\underline{s}_2$
$T_1 < \hat{C} < T_2$	CER - constant rate (hide a predefined number of bits)	add $+\underline{s}_1$
$\hat{C} \geq T_2$	VER - variable rate (the number of embedded bits is determined by $\hat{C}$)	add $-\underline{s}_1$ and # bits embedded

6.4.4 Experimental Results

We tested the proposed algorithm on luminance components of digital video. A same character string denoting access information (without error correction coding) is hidden in two embedding levels. For simplicity, we use equal-length segments, each containing 6 consecutive frames. One test video is the first 60 frames of "flower garden" raw sequence, which has a frame size of 352 x 240 and a frame rate of 30 frames per second. After data hiding, the video is encoded using MPEG-2 compression. 18 characters can be extracted accurately under 1.5Mbps compression with a GOP structure of IBBPBBI. For less strong compression such as 4.5Mbps encoding, a longer string of 91 characters can be successfully extracted. This indicates that the system is able to extract the whole string from only a small number of frames when the video goes through light compression hence with high quality, while also being able to extract it when the video goes through heavy compression, though the processing of more frames is needed in the latter case. Fig. 6.13 shows the 1st and 30th frames of original and watermarked frames as well as their enlarged difference.

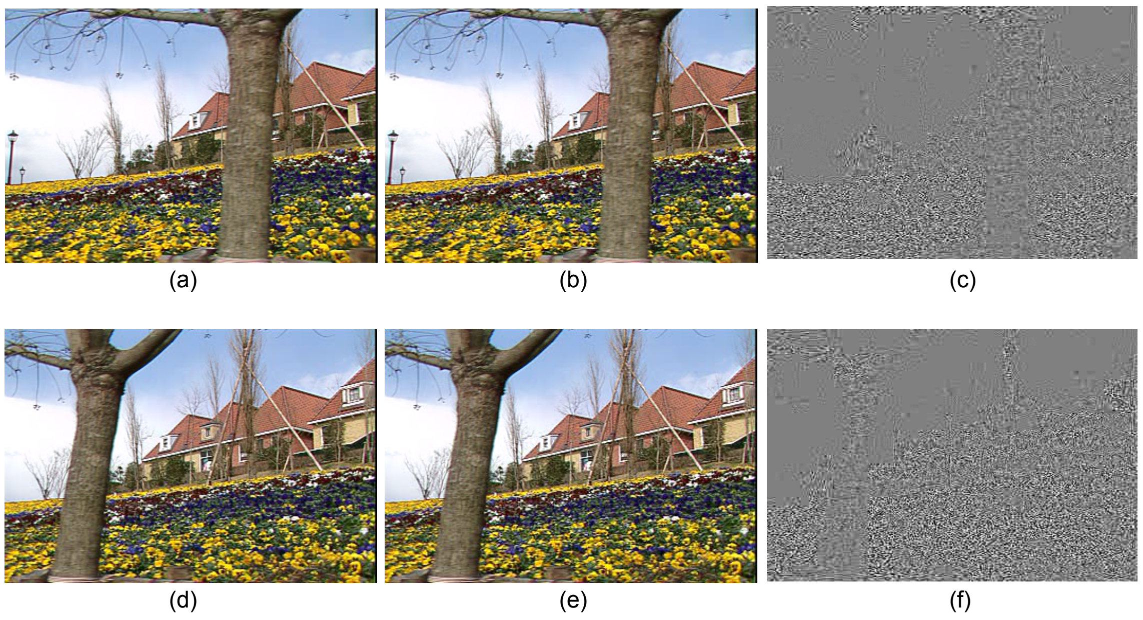


Figure 6.13: Multi-level data hiding for flower garden video sequence: (a)-(c) the original 1st frame, the watermarked version, and their difference, respectively; (d)-(f) the original 30th frame, the watermarked version, and their difference, respectively. Both videos are compressed using MPEG-2 4.5Mbps, and the differences are enlarged with gray denoting zero difference and black/white denoting large difference.

We also tested a longer and more diverse sequence with 660 frames. It is a concatenated version of “flower garden”, “football”, and “table tennis” sequences. A total of 3032 bits are embedded via high capacity level and 1266 bits via high robustness level. All 4298 bits can be extracted accurately after 4.5Mbps MPEG-2 compression. When the video is compressed at 1.5Mbps, the 1266 bits at high robustness level can still be correctly extracted, though the detector shows low detection confidence on

3 bits (0.2%) under such severe distortion. In practice, error correction codes such as BCH/RS codes or Turbo codes can be incorporated to correct a small percentage of errors, if any. The experimental results of both image and video data hiding are summarized in Table 6.3.

Table 6.3: Summary of experimental results for the proposed multi-level image and video data hiding system

	Level-1: high capacity		Level-2: high robustness		Notes
	embedding rate	robustness	embedding rate	robustness	
512x512 lenna	32x32 pattern (1024 bits)	JPEG Q ≥ 45%, moderate additive noise.	"PINTL" (35 bits)	JPEG Q ≥ 20%; low pass filtering; additive noise.	PSNR = 42.5 dB
512x512 baboon			"Panasonic Tech." (105 bits)		PSNR = 33.6 dB
60-frame 352x240 flower garden sequence	640 bits (91 char.)	Mpeg-2 4.5Mbps; frame dropping	132 bits (18 char.)	Mpeg-2 1.5Mbps; frame dropping	some control info. is also hidden, such as how many number of bits are embedded.
660-frame 352x240 concatenated video sequence	3032 bits		1266 bits		

Acknowledgement

Most of the work presented in this chapter was performed with Dr. Heather Yu while the author was a summer intern at Panasonic Information and Networking Laboratories.

Chapter 7

Data Hiding for Image Authentication

7.1 Introduction

For years, audio, image, and video have played important role in journalism, archiving, and litigation. A coincidentally captured video of the well-known 1993 Rodney King incident played an important role in prosecution, a secretly recorded conversation between Monica Lewinsky and Linda Tripp touched off the 1998 presidential impeachment, just to name a few. Keeping our focus on still pictures, we have no difficulty in realizing that the validity of the old saying "Picture never lies" has been challenged in the digital world of multimedia. Compared with the traditional analog media, making seamless alteration is much easier on digital media by software editing tools. With the popularity of consumer-level scanner, printer, digital camera, and digital camcorder, detecting tampering becomes an important concern. In this chapter, we discuss using digital watermarking techniques to partially solve this problem by embedding authentication data invisibly into digital images.

In general, authenticity is a relative concept: whether an item is authentic or not is relative to a reference or certain type of representation that is regarded as authentic. Authentication is usually done by checking whether certain rules and relationship which are supposed to be found for an authentic copy are still hold in the test material. In traditional digital network communication, a sophisticated checksum, usually known as hash or message digest, is used to authenticate whether the content has been altered or not. The checksum is encrypted using such cryptographic techniques as public-key encryption to ensure that the checksum cannot be generated or manipulated by unauthorized persons. This is known as *digital signature* in cryptography [11]. For traditional type of source data such as text or executable codes, the checksum is stored or transmitted separately since even minor changes on this kind of data may lead to different meaning. Perceptual data such as digital image, audio and video are different from the traditional digital data in that perceptual data can be slightly changed without introducing noticeable difference, which offers new room for authenticating perceptual data. For example, we can imperceptibly modify an image so that the least significant bit of the representation of each pixel is always the checksum of other bits. In other words, the checksum is *embedded* into the image instead of having to be stored separately as for traditional data. Such embedding approach falls in the category of digital watermarking / data-hiding. For example, *fragile watermarking* [26] can be used to insert into an image some special data which will be altered when the host image is manipulated.

Many general techniques of data hiding can be applied to this specific applications, such as the general approaches discussed in Part I and the image data hiding approaches presented in Chapter 6. But the algorithm design has to be aware of a few unique issues associated with authentication, including the choice of what to embed and the security considerations to prevent forgery or manipulation of embedded data. The following features are desirable to construct an effective authentication scheme:

1. to be able to determine whether an image has been altered or not;
2. to be able to integrate authentication data with host image rather than as a separate data file;
3. the embedded authentication data be invisible under normal viewing conditions;
4. to be able to locate alteration made on the image;
5. to allow the watermarked image be stored in lossy-compression format, or more generally, to distinguish moderate distortion that does not change the high-level content vs. content tampering.

This chapter presents a general framework of watermarking for authentication and proposes a new authentication scheme by embedding a visually meaningful watermark and a set of features in the transform domain of an image via table look-up. The embedding is a Type-II embedding according to Chapter 3. Making use of the pre-distortion nature of Type-II embedding, our proposed approach can be applied to compressed image using JPEG or other compression techniques, and the watermarked image can be kept in the compressed format. The proposed approach therefore allows distinguishing moderate distortion that does not change the high-level content versus content tampering. The alteration made on the marked image can be also localized. These features make the proposed scheme suitable for building a "trustworthy" digital camera. We also demonstrate the use of shuffling (Chapter 4) in this specific problem to equalize uneven embedding capacity and to enhance both embedding rate and security.

7.2 Previous Work

The existing works on image authentication can be classified into several categories: digital signature based, pixel-domain embedding, and transform-domain embedding. The latter two categories are invisible fragile or semi-fragile watermarking.

Digital signature schemes are built on the ideas of hash (or called message digest) and public key encryption that were originally developed for verifying the authenticity of text data in network communication. Friedman [74] extended it to digital image as follows. A signature computed from the image data is stored separately for future verification. This image signature can be regarded as a special encrypted checksum. It is unlikely that two different natural images have same signature, and even if a single bit of image data changes, the signature may be totally different. Furthermore, public-key encryption makes it very difficult to forge signature, ensuring a high security level. After his work, Schneider *et al.* [75] and Storck [79] proposed content-based signature. They produce signatures from low-level content features, such as block mean intensity, to protect image content instead of the exact representation. Another content-signature approach by Lin *et al.* developed the signature based on a relation between coefficient pairs that is invariant before and after compression [41, 76]. Strictly speaking, these signature schemes do not belong to watermarking since the signature is stored separately instead of embedding into images.

Several pixel-domain embedding approaches have been proposed. In Yeung *et al.*'s work, a meaningful binary pattern is embedded by enforcing certain relationship according to a look-up table. Their work allows the tampering that is confined in some local areas to be located [78]. Walton proposed an approach by embedding data via enforcing relationship between sets of pixels [77]. Another pixel-domain scheme was proposed by Wong [80]. This scheme divides an image into blocks, then copies the cryptographic digital signature of each block in the least significant bits of the pixels for future verification. However, images marked with these pixel-domain *Invisible-Fragile Watermarking* schemes cannot be stored in lossily compressed format like JPEG compression, which is commonly used in commercial digital camera.

In addition to pixel-domain approaches, several block DCT-domain schemes may be used for authentication purpose. Swanson *et al.* [66] round coefficients to multiples of just-noticeable difference or mask value, then add or subtract a quarter to embed one bit in an 8×8 block. Koch *et al* [63] embed one bit by forcing relationships on a coefficient pair or triplet in mid-band. The two approaches achieve limited robustness via pre-distortion, and the embedding is likely to introduce artifacts in smooth regions. Similar problem exists in other approaches, including a DCT-domain quantization approach by Lin *et al.* [41], and a Wavelet-domain quantization approach by Kundur *et al.* [82], both of which embed a signature in transform domain by rounding the quantized coefficients to an odd or even number. Additional data can be embedded to recover some tampered or corrupted regions, such as the self-recovery watermarking proposed by Fridrich *et al.* in [88] and by Lin *et al.* in [41].

Recalling the desirable requirement for image authentication presented in the previous section, we find that many existing approaches in literature cannot satisfy all requirements. The digital signature proposed in[74], as well as the content based signature reported in [75] and [79] do not satisfy the requirements 2 and 4. The pixel-domain scheme [78, 77, 80] can not be stored in lossy compression format. In addition, transform-domain schemes [41, 82, 63, 66] do not handle the uneven embedding capacity problem raised in Chapter 4, therefore may either introduce artifacts in smooth region or embed fewer authentication bits than our proposed approach.

7.3 Framework for Authentication Watermark

We proposed a general framework including the following elements for authentication watermark:

1. what to authenticate

2. what data to be embedded for authentication purpose

3. how to embed data into an image

4. how to handle uneven embedding capacity

5. security considerations

The first element is fundamental and affects the other elements. We have to decide whether to authenticate the exact representation of an image, or to have some tolerance toward certain processing such as compression, cropping and scaling. In addition, we need to determine other functionalities we would like to achieve, such as locating alteration. The next two elements, namely, what and how to embed, are designed based on the answer to the first element. More specifically, we can either mainly rely on the fragility of the embedding mechanism to detect tampering (e.g., to put zeros in the least significant bits of all pixel values and later to check whether such properties still hold or not on test images), or rely on the embedded data (e.g., to robustly embed a 64-bit check sum of the image features such as the block mean intensity or image edge map, and later to check whether the extracted check sum matches the one computed from the test image), or use both. For local embedding schemes such as the TDMA type modulation discussed in Chapter 4, special handling with smooth region, or in general, the uneven embedding capacity is necessary to achieve high embedding rate and to locate alteration. Besides an appropriate design of what and how to embed, the detailed implementation must take security issues into account in order to meet the demands in practical applications, for example, to make it difficult for an attacker to forge valid authentication watermark in a tampered image.

Following the above framework, we discuss our proposed authentication watermarking approach based both on the fragility of embedding mechanism and on matching the embedded features with the features extracted from a test image. The detection of tampering relies on both the embedding mechanism and the embedded data. The alteration can also be located unless there is global tampering or the tampering involves large areas. We shall present our approach in the context of grayscale images with JPEG compression. The extension to images compressed using other means such as Wavelet and to color images will be briefly discussed in Section 7.7. A block diagram of the embedding process is given in Fig. 7.1 which, aside from the block labeled "embed", is identical to that of the JPEG compression [18]. Watermarks are inserted into the quantized DCT coefficients via a look-up table. Explained below are two aspects of watermark-based authentication, namely, to embed what data and how to embed them.

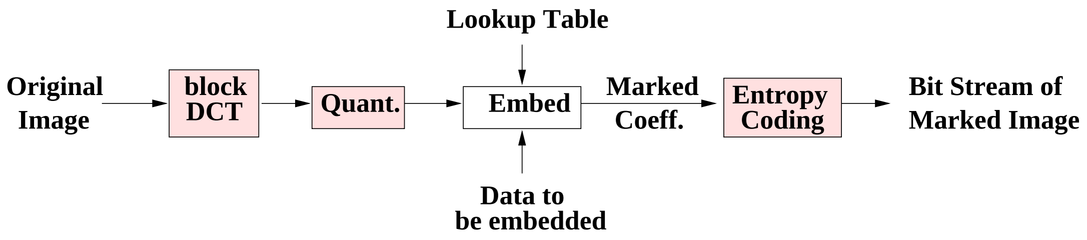


Figure 7.1: Block diagram of embedding process for authentication watermarking. The "Quant." module stands for the quantization step.

7.4 Transform-domain Table Lookup Embedding

The data for authentication purpose is generally embedded using a Type-II approach discussed in Chapter 3 for its high embedding capacity and fragility, both of which are useful in authentication. Here we present a Type-II embedding using a look-up table in transformed domain. This transform domain look-up table embedding is an extension of the pixel-domain scheme proposed by Yeung *et al.* [78]. The embedding is performed on the quantized coefficients with a set of pre-selected quantization step sizes, which are known to the detector because the extraction of hidden data must be performed in the same quantized domain. As discussed in Chapter 3, this quantization is a pre-distortion step to obtain limited robustness against compression and other distortion.

A proprietary look-up table (LUT) is generated beforehand by the owner of the image or the digital camera manufacturers. The table maps every possible value of JPEG coefficient randomly to "1" or "0" with a constraint that the runs of "1" and "0" are limited in length. To embed a "1" in a coefficient, the coefficient is unchanged if the entry of the table corresponding to that coefficient is also a "1". If the entry of the table is a "0", then the coefficient is changed to its nearest neighboring values for which the entry is "1". The embedding of a "0" is similar. This process can be abstracted into the following formula where v_i is the original coefficient, v_i' is the marked one, b_i is the bit to be embedded in, $Q(\cdot)$ is the quantization operation [1], and

[1]For a uniform quantizer with quantization step size q, the quantization operation $Q(x)$ is to round x to the nearest integer multiples of q.

$LUT[\cdot]$ is the mapping by a look-up table:

$$
v_i{}' = \begin{cases} Q(v_i) & \text{if } LUT[Q(v_i)] = b_i \\ v_i + \delta & \text{if } LUT[Q(v_i)] \neq b_i, \text{ and} \\ & \quad \delta = \min_{|d|}\{d = Q(x) - v_i \text{ s.t. } LUT[Q(x)] = b_i\} \end{cases} \tag{7.1}
$$

The extraction of the signature is simply by looking up the table. That is,

$$
\hat{b_i} = LUT[Q(v_i{}')] \tag{7.2}
$$

where $\hat{b_i}$ is the extracted bit.

The basic idea of the embedding process is also illustrated by the example in Fig. 7.2. Here, zeros are to be embedded in two quantized AC coefficients with values "-73" and "-24" of an 8×8 image block. The entry in the table for coefficient value "-73" is "1". In order to embed a "0", we have to change it to its closest neighbor for which the entry is "0". In this example, "-73" is changed to "-74". Since the entry for coefficient value "24" is already "0", it is unchanged.

As mentioned earlier, the detection of tampering is based on both the embedding mechanism and the embedded data. The clues provided by the embedding mechanism is as follows: when a small part of a watermarked image is tampered without the knowledge of the look-up table, the extracted bit from each tampered coefficient becomes random, i.e.,

$$
P(\hat{b_i} = 0) = P(\hat{b_i} = 1) = \frac{1}{2}
$$

implying that it is equally likely to be the same as or be different from the bit b_i originally embedded. For the moment, we assume that detector has knowledge of the originally embedded data $\{b_i\}$ and the justification of this assumption will be presented later. From a single coefficient, it is not reliable to determine whether tampering occurs or not because there is a 50/50 chance that the extracted data

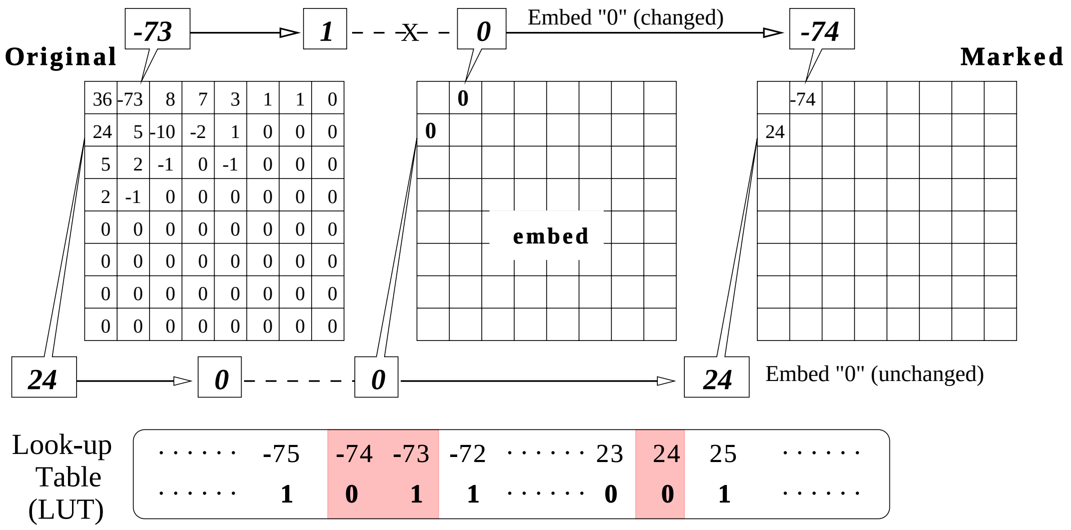


Figure 7.2: Frequency-domain Embedding Via Table Lookup: zeros are embedded to two quantized DCT coefficients "-73" and "24" by enforcing relationship according to a look-up table. After embedding, the coefficients in the watermarked image are "-74" and "24".

matches the originally embedded one, i.e., $\hat{b_i} = b_i$. However, if the tampering affects several coefficients in a block and/or coefficients in several blocks, the chances of miss detection (i.e., all decoded data of altered region happen to match the originally embedded ones) are reduced exponentially:

$$P([\hat{b_1}, ..., \hat{b_n}] = [b_1, ..., b_n]) = 2^{-n}$$

where n is the number of coefficients affected by tampering. For example, miss detection probability is around 0.00098 when n is equal to 10.

According to Chapter 3, the table lookup embedding, being an example of Type-II embedding, relies on deterministic relationship enforcement. From set partition point of view, all possible values of a quantized coefficient are divided into two subsets each of which conveys special meaning and the partition rule is set by the table. One subset contains values which map to "1" according to the lookup table, and the other subset contains those that map to "0". The embedding process introduces minimal necessary changes to force a quantized coefficient to take value from the subset that maps to the binary data to be embedded.

7.4.1 Considerations for Imperceptibility & Security

Several steps are taken to ensure that the embedding is invisible:

- The run of "1" and "0" entries in the LUT is constrained to avoid excessive modification on the coefficients;
- The DC coefficient in each block is not changed to avoid blocky effect unless the quantization step is very small [2];
- Small valued coefficients (mostly in high frequency bands) are not modified to avoid large relative distortion.

Coefficients that are allowed to be changed according to these constraints are called *embeddable* or *changeable*. The number embeddable coefficients vary significantly, which has been referred as "uneven embedding capacity" in Chapter 4. Also, extraction error may occur due to image format conversion, rounding, and other causes involving no content changes. As discussed in Chapter 4, we apply shuffling to equalize the unevenly distributed embedding capacity. A proper block size is first determined according to the overall embedding capacity measured by the total number of changeable coefficients. The side information regarding the block size can be

[2]This constraint may be loosened to allow the DC coefficients in texture regions to be modified, as the change there is likely to be invisible.

conveyed using the approaches discussed in Chapter 4 and 6, for example, using additive spread spectrum embedding. Then, one bit is embedded in each shuffled block by repeatedly embedding the same bit to all embeddable coefficients in the shuffled block. The bit is extracted by a detector in the same shuffled domain via majority voting. The repetition may be replaced by more sophisticated coding techniques for better performance.

The algorithm shown in Table 7.1 is for generating an L-entry look-up table $T[\cdot]$ with maximum allowed run of r and positive index $i \in \{1, ..., L\}$.

Table 7.1: An Algorithm for Generating Look-up Table with Limited Runs

Step-1:	$i = 1$.
Step-2:	If $i > r$ and $T[i-1] = T[i-2] = ... = T[i-r]$, then $T[i] = 1 - T[i-1]$. Otherwise, generate a random number from $\{0, 1\}$ with a probability $0.5 : 0.5$ and set $T[i]$ to this value.
Step-3:	Increase i by 1. If $i > L$, stop. Otherwise go back to Step-2.

To analyze the minimum secure value of r, we start with the case of $r = 1$, which has only two possibilities:

$$T[i+1] = 1 - T[i], \qquad T[0] \in \{0, 1\}, i \in \mathbf{N}$$

or equivalently,

$$T[i] = \begin{cases} 0 & \text{(i is even)} \\ 1 & \text{(i is odd)} \end{cases} \qquad or \qquad T[i] = \begin{cases} 1 & \text{(i is even)} \\ 0 & \text{(i is odd)} \end{cases}$$

This implies that the odd-even embedding discussed in Chapter 3 is a special case of table-lookup embedding. Since there is very little uncertainty in the table, it is easy for unauthorized persons to manipulate the embedded data, and/or to change the coefficient values while retaining the embedded data unchanged. Therefore, $r = 1$ is not a proper choice if no other security measure such as a careful design of what data to embed is taken.

When r is increased to 2, the transition of the LUT entries has Markovian property, as shown in Fig. 7.3. It is easy to show that starting from "0" or "1", the number of possible LUTs with i elements long, F_i, forms a *Fibonacci series*:

$$F_i = F_{i-1} + F_{i-2}, \qquad F_0 = 1, F_1 = 1. \tag{7.3}$$

The total number of possible sequences with length $L = 256$ is on the order of 10^{53}. Although this number is smaller than the number of possible sequences without run length constraint, which is 2^{256}, or the order of 10^{77}, the table still has enough uncertainty, and the probability of obtaining table by guessing is very, very small. Thus from embedding mechanism point of view, the minimum secure choice for r is 2.

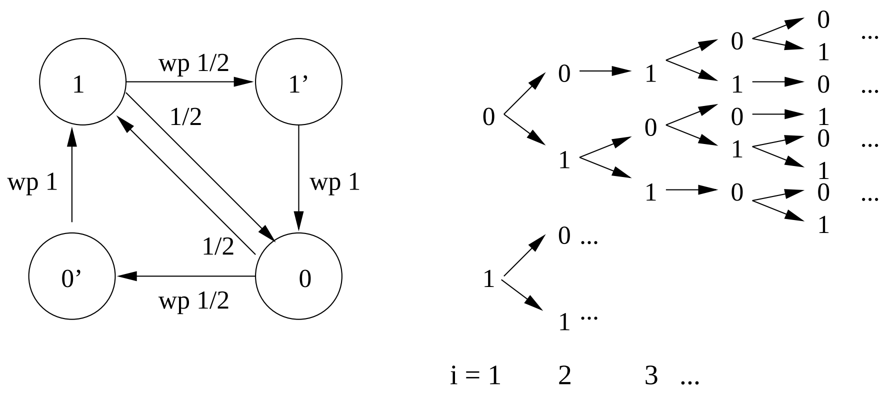


Figure 7.3: (left) Markov property of restricted LUT generation with maximum run of 2, where "wp" stands for "with probability"; (right) Expansion graph illustrating the generation of restricted LUTs with i elements.

The mean squared error incurred by table-lookup embedding with $r = 2$ is computed as follows. First, we consider the error incurred purely by quantization, i.e., rounding the input coefficient in the range of $[(k-1/2)Q, (k+1/2)Q)$ to kQ:

$$\text{MSE(quantize to } kQ \text{)} \quad = \quad \frac{Q^2}{12} \tag{7.4}$$

This is the case if the entry corresponding to the original quantized coefficient in the table happens to be the same as the bit to be embedded. We then consider the case of having to shift the coefficient to $(k-1)Q$ in order to embed the desired bit:

$$\begin{aligned} \text{MSE(shift to } (k-1)Q \text{)} \quad &= \quad \int_{(k-\frac{1}{2})Q}^{(k+\frac{1}{2})Q} [x-(k-1)Q]^2 \times \frac{1}{Q} dx \\ &\underset{\text{let} \quad y = x-(k-1)Q}{=} \quad \frac{1}{Q} \times \int_{\frac{Q}{2}}^{\frac{3}{2}Q} y^2 dy \quad = \quad \frac{13}{12} Q^2 \end{aligned} \tag{7.5}$$

By symmetry, the MSE for shifting to $(k+1)Q$ is same as above. Hence the overall MSE is:

$$\begin{aligned} \text{overall MSE} \quad &= \quad \frac{\text{MSE(to kQ)}}{2} + \frac{\text{MSE(to (k-1)Q) + MSE(to (k+1)Q)}}{4} \\ &= \quad \frac{1}{2} \times \frac{Q^2}{12} + 2 \times \frac{1}{4} \times \frac{13}{12} Q^2 = \frac{7}{12} Q^2 \end{aligned} \tag{7.6}$$

This is achievable since look-up table with $r = 2$ allows at most one-Q-step modification away from kQ. The MSE is a little larger than the case of $r = 1$ (i.e., the odd-even embedding in Chapter 3), which gives an MSE of $Q^2/3$ and is equivalent to double the quantization size as far as the distortion is concerned.

7.4.2 More on Determining Embedded Data & Coefficient Changes

Earlier when explaining the extraction of embedded data, we have assumed that a detector has the knowledge of the originally embedded data $\{b_i\}$. In practice, this knowledge is not a necessity, especially with the incorporation of majority voting, error correction coding and shuffling.

Assuming that the tampering is restricted to a small portion of an image, the changed coefficients tend to occur in small clusters and the number of them is not large. After shuffling, these coefficients will be diffused to many blocks in shuffled domain. Because each shuffled block is unlikely to receive many too many changed coefficients by the nature of shuffling [3], the few changed coefficients in each shuffled block will not affect $\{\hat{b}_i\}$, which is the bit ultimately extracted from that block via table lookup detection and error correction coding. This implies that the extracted data, $\{\hat{b}_i\}$, can be regarded as a good estimate of $\{b_i\}$, the originally embedded data. Using $\{\hat{b}_i\}$ as "ground truth", we can determine what bit value is supposed to be embedded into each embeddable coefficients by an embedding system. By comparing the supposedly embedded data with the data actually extracted from each coefficients of a test image, we are able to identify the changed coefficients.

The change identification by this two-step process relies on the fragility of embedding, namely, the tampering may change the originally embedded data. In the next section, we shall see a second way to detect tampering, which relies on the meaning and the value of embedded data. By then, we will get a more complete picture of the authentication framework introduced in Section 7.3.

7.5 Design of Embedded Data

We mentioned earlier that the embedded data can be used to detect tampering. The authentication data we propose to embed consists of a visually meaningful binary

[3]This is supported by the same analysis on shuffling in Section 4.7, with the percentage of blue balls (the balls of interest), p, being very small.

pattern and content features. As we shall see, the combination of these two types data is suitable for applications such as image authentication for "trustworthy" digital cameras.

7.5.1 Visually Meaningful Pattern

The visually meaningful pattern, such as letters and logos, serves as a quick check for signaling and locating tampering. Shown in Fig. 7.4 is a binary pattern "PUEE". The decision on whether an image is altered or not can be made by (automatically) comparing the extracted pattern with the original one, if available, or by human (e.g., jury in court) based on visualizing the extracted pattern. The latter case uses a reasonable assumption that human can distinguish a "meaningful" pattern from a random one. It is also possible to make such decision automatically, e.g., computing a randomness measure.

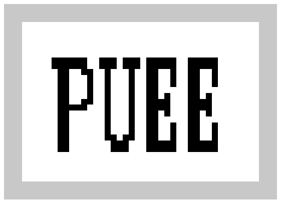


Figure 7.4: A binary pattern as part of the embedded data

7.5.2 Content-based Features

Content features offer additional security to combat forgery attack and help distinguish content tampering vs. minor, non-content change. Due to the limited embedding bit rate, content features have to be represented using very few number of bits.

An example of content features is the most significant bit of macroblock mean intensity. Another example is the sign of intensity difference between a pair of blocks or macroblocks. These features bring dependence on images to the data to be embedded, so it is effective against forgery attacks that rely on the independence [139]. Other features, such as the edge map and luminance/color histogram, can also be used. A general formulation of local features of the block (i,j) is

$$b_{i,j} = f(\underline{v}_{i-k_1,j-k_2}, ..., \underline{v}_{i-1,j}, \underline{v}_{i,j}, \underline{v}_{i+1,j}, ..., \underline{v}_{i+k_3,j+k_4})$$

where $k_1, k_2, k_3, k_4 \in \mathbf{N}$, $\underline{v}_{i,j}$ represents a collection of all the coefficients in the block (i,j), and $f(\cdot)$ is a deterministic function which is known to both the embedder and the detector. The detector compares the features computed from the test image and the ones extracted by table look-up (i.e., the features embedded by the embedder). A mis-match between the two sets of features is an indication that the image has been tampered.

Content features are especially useful in authenticating smooth regions. As discussed in Section 7.4.1, no data can be embedded in smooth regions without introducing visible artifacts, hence it is impossible to rely on the embedding mechanism to signal the tampering of these regions, i.e., it is impossible to embed data with certain regularity and later check the alteration of such regularity at detector. The tampering associated with smooth regions includes the case of altering a smooth block to another smooth one (e.g., changing luminance or color) and the case of altering a complex block to a smooth one. Changing smooth block to complex block is easy to detect because when the detector sees the complex block, it assumes some data have been embedded, but the extracted data from these altered block will be random as an attacker has no knowledge of the secrets used in embedding. Although there are limited meaningful changes that can be applied by changing original blocks (either smooth or complex) to smooth ones, we believe that authentication schemes should take smooth region authentication into account instead of risking miss detection of possible meaningful alterations. Since the embedded data can be used to detect tampering, alterations in smooth regions can be detected by relying on the embedded data, i.e., we embed features derived from smooth regions in the embeddable regions and later check the match between the features computed from a test image and those extracted by the watermark detection module. In addition, the features based on block mean intensity are useful in detecting intentional alterations such as the possible meaningful tampering by only changing some DC coefficients, because the embedding scheme preferably leaves DC coefficient untouched to avoid blocky effect.

7.6 Experimental Results

We first present the results of an earlier design that does not use shuffling and embeds less data than the shuffling approach. A JPEG compressed image of size 640×432 is shown in Fig. 7.5. Fig. 7.6(a) is the same image but with a 40×27 binary pattern "PUEE" of Fig. 7.6(b) and the MSB of macroblock mean intensity embedded in. This image is visually indistinguishable from the original. In terms of PSNR with respect to original uncompressed image, the watermarked one is only 1dB inferior to the image with JPEG compression. The smooth regions of the image are mainly in the sky area. For these blocks, *backup embedding* is used, namely, the data associated with the i-th block are embedded in both the i-th block and a companion block (see Fig. 4.4).

The marked image is modified by changing "Princeton University" to "Alexander Hall" and "(c) Copyright 1997" to "January 1998", shown in Fig. 7.7(a). This image

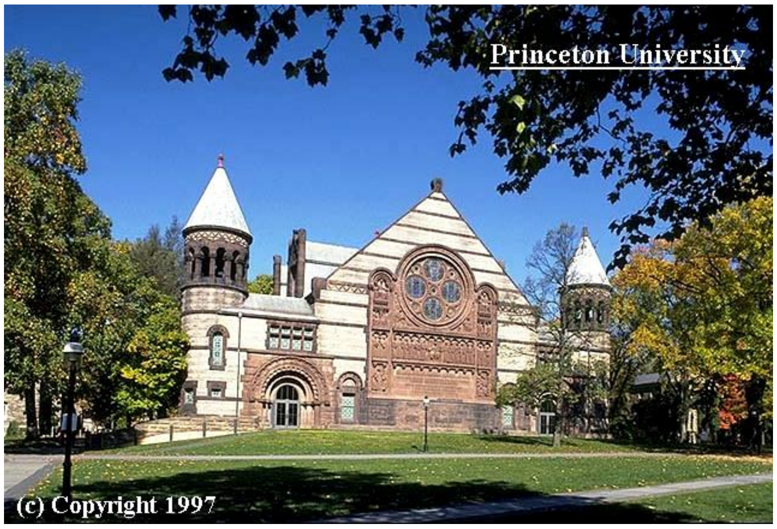


Figure 7.5: An original unmarked 640 × 432 image *Alexander Hall* (stored in JPEG format with a quality factor of 75%). Watermarking is performed on its luminance components.

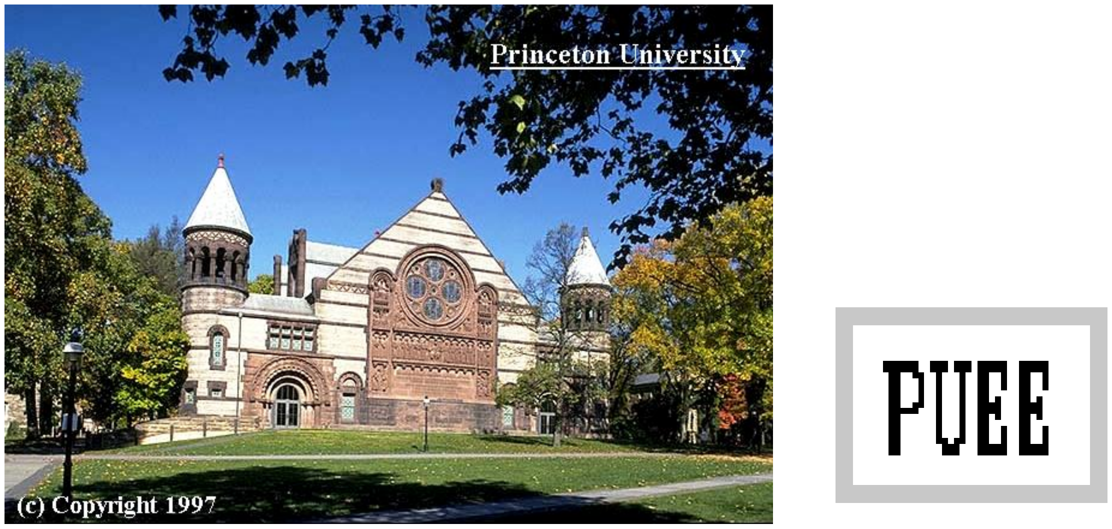


Figure 7.6: [$a|b$] Authentication result with watermark embedded without shuffling: (a) watermarked image; (b) embedded binary pattern.

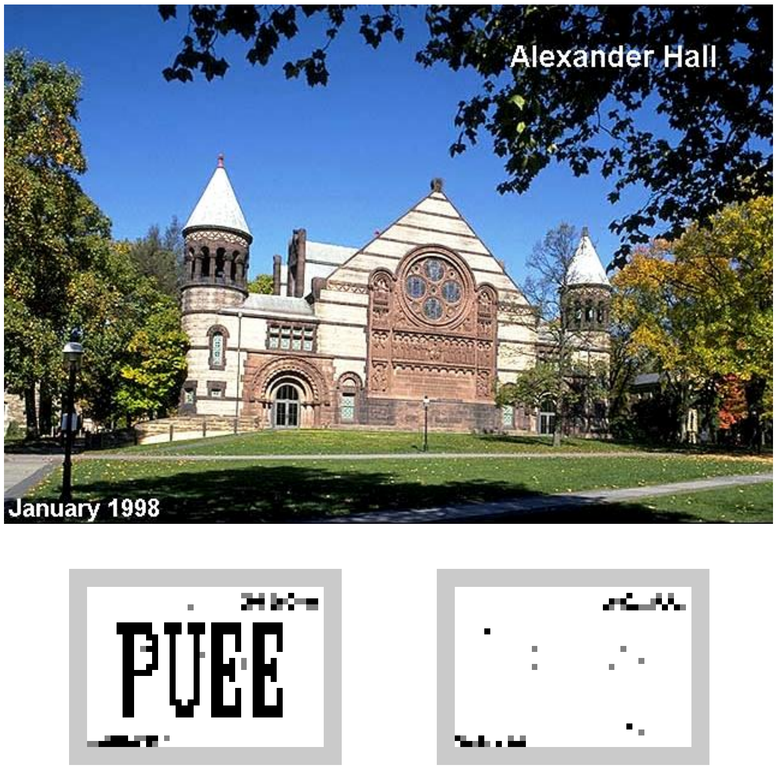


Figure 7.7: $[\frac{a}{b|c}]$ Authentication result with watermark embedded without shuffling: (a) an altered version of the watermarked image (stored in 75% JPEG); (b) extracted binary pattern from edited image; (c) feature matching result.

is again stored in the JPEG format. For the ease of visualizing embedded pattern, we shall denote the block that embeds “0” as a black dot, the block that embeds “1” as a white dot, and the block with no obvious majority being found in detection as a gray dot. Similarly, to visualize the feature matching result, we use a black dot for the unmatched block, a white for the matched one, and a gray for the block with which we do not found the obvious majority in detection hence have low confidence in determining a match or unmatch. With these notations, Fig. 7.7(b) and (c) show the extracted binary pattern and content feature matching result of the tampered image.

The random pattern corresponding to the tampered blocks are clearly identifiable. We can see that using the backup embedding, there are very few unembeddable bits at an expense of the reduced embedding rate. Also notice that the round-off error may occur during the verification and tampering, which contributes to the few unexpected dots out of the altered regions.

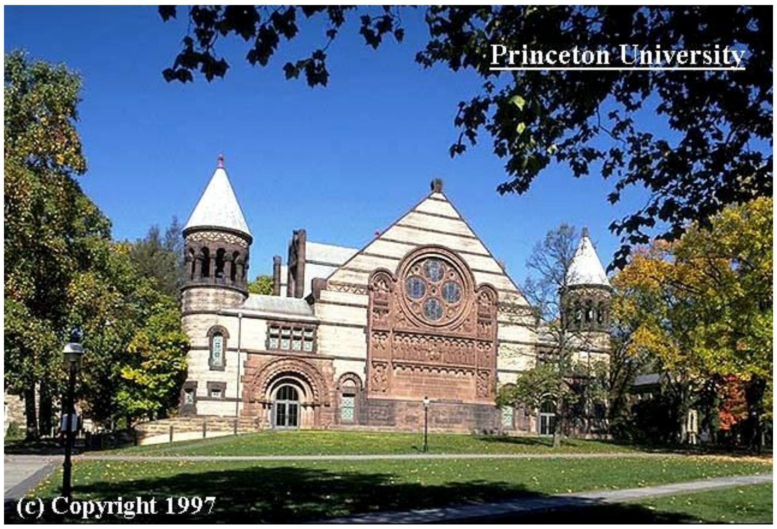
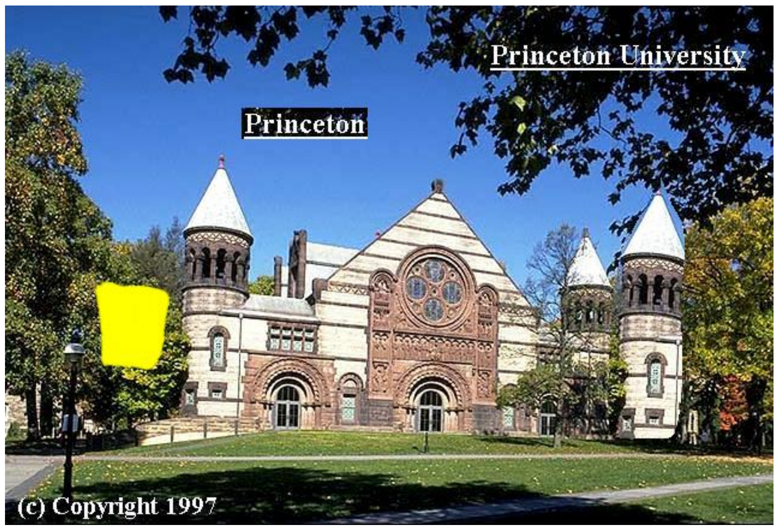


Figure 7.8: An image with authentication watermark embedded using shuffling.

As discussed in Chapter 4, shuffling equalizes the uneven embedding capacity and allows embedding more data. The example shown in Fig. 7.8 has a BCH encoded version of the "PUEE" pattern and multiple content features embedded in its luminance components. There is no visual difference between this watermarked image and the original unwatermarked copy in Fig. 7.5. The embedded content features include the most significant bit of the average macroblock intensity and the smoothness of each macroblock. The combined result of pattern comparison and feature matching provides information regarding both content tampering and minor non-content changes

(a)


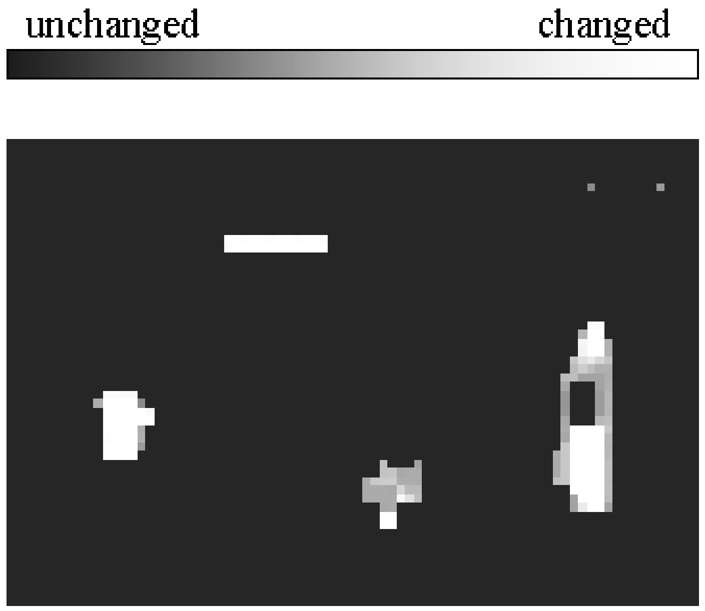
(b)


Figure 7.9: Authentication result with watermark embedded using shuffling: (a) an altered version stored in 75% JPEG format of Fig. 7.8; (b) authentication result with whiter dot denoting higher likelihood of content manipulation for the corresponding area in (a).

such as those introduced by rounding or recompression. Fig. 7.9(a) shows various content alterations made to the watermarked image. A comprehensive report combining both pattern comparison and feature matching result is shown in Fig. 7.9(b), with whiter pixel indicates higher likelihood of content changes. We can see that the improved authentication system using shuffling is able to identify content changes with little false alarm. The quantization domain used by the embedding step is the same as JPEG with quality factor 75%, implying the system is able to tolerate compressions and other distortions that are comparable or less severe than JPEG 75%.

7.7 Extensions

Multi-level Data Embedding and Unequal Error Protection

We mentioned in Section 7.5 that two sets of data, namely, a meaningful visual pattern and a set of low-level content features, are embedded in the image for authentication purpose. More generally, the idea of multilevel data hiding in Chapter 6 can be incorporated to embed several sets of data with different levels of error protection and using embedding mechanisms with different robustness. The embedded data sets could be image features at different resolutions: the coarser the level is, the more it is protected. The multi-resolution features with unequal error protection can help us authenticate an image in a hierarchical way.

Other Compression Format

Besides JPEG compressed image, we also performed some simple tests in Wavelet compression and found our approach effective in detecting tampering. In addition to the advantage in efficient coding, Wavelet domain offers convenience to implement the above-mentioned hierarchical authentication. Since wavelet is selected as the core of the new JPEG2000 compression standard, a complete authentication system design targeted at JPEG2000 is a promising new direction.

Color Images

For color images, we currently work in YCrCb coordinates and use the proposed approach to mark luminance components while leaving chrominance components unchanged. A better way is to apply the approach to chrominance components as well to embed more data and to detect potential tampering of colors. We may also work in other color coordinates, such as RGB. However in practice, due to the limited computation precision, we are likely to find a few pixels whose YCrCb or RGB values may change after one or more rounds of color coordinate conversion [152]. This is similar to the rounding and truncating errors incurred by going to and forth between pixel domain and transform domain. Pre-distortion via quantization and error correction codes are ways to combat these errors.

Videos

A system for authenticating MPEG compressed digital video can be designed by marking I-frames of video streams using our proposed scheme. In addition, I-frame serial number can be used as part of embedded data to detect modification such as frame reordering and frame dropping. Alternatively, these frame jitters can be detected via a spread-spectrum watermark in each frame, which is embedded using the same approach as the embedding of frame synch index in Section 6.4. P- and B-frames can be similarly watermarked but with larger quantization step size and more error protection in embedding to survive motion compensation during moderate compression. This practical consideration is similar our implementation of multilevel data hiding for video in Section 6.4. Compressed domain embedding in these predicted P- and B- frames are also possible by manipulating the residues of motion compensation.

Chapter 8

Data Hiding for Video Communication

Motivated by the traditional watermarks in paper, ownership verification and tampering detection are the initial purposes of embedding digital watermarks in multimedia source. Examples of these watermark systems are shown in Chapter 5–7. In general, data hiding provides a way to convey side information that can also be used for many other purposes. For example, Silverstein *et al.* proposed to embed into an image a map indicating the regions for which an enhancement scheme effectively improves the perceptual quality and later to use this embedded information to direct the selective enhancement only in these regions [126, 127]; Song *et al.* used embedding in motion vectors to facilitate key distribution and key updating in secure multimedia multicast [42, 129]. In this chapter, we discuss the applications of data hiding in video communication, where the side information helps to achieve additional functionalities or better performance.

Delivery of digital video through network and wireless channels is becoming increasingly more common. Limited bandwidth and channel errors are two major challenges in video communication. Transcoding a video to a lower rate helps to cope with the bandwidth limitation by gracefully degrading visual quality, while concealing corrupted regions of an image/video is commonly used to compensate the perceptual quality reduction caused by transmission errors. In the followings, we will explain how to apply data hiding to these problems to enhance performance.

8.1 Transcoding by Downsizing Using Data Hiding

A number of bit rate reduction techniques have been proposed for transcoding, including frame dropping, suppressing color, discarding high order DCT coefficients and re-quantizing DCT [154]. The reduction of spatial resolution can significantly reduce the bit rate [153], but the processing is rather involved. More specifically, most videos are stored in compressed domain involving DCT transform and motion compensation. As many applications require real-time transcoding, it is desirable to carry out the transcoding in the compressed domain and the approach should aim at reducing the computational complexity while achieving a reasonable visual quality. Typically, motion estimation consumes 60% of encoding time [153], while motion compensation consumes 11%. In order to transcode with low delay, we need to concentrate on reducing the complexity in these two steps.

8.1.1 Overview of Proposed Approach

We propose a fast approach to obtain from an MPEG stream a new MPEG stream with half the spatial resolution. That is, each macroblock (16×16) in the original video becomes one block (8×8) in the reduced-resolution video. We work directly in the compressed domain to avoid the computationally expensive steps of converting to the pixel domain. Two problems need to be addressed in order to generate a standard-compliant bit stream for the reduced-resolution video: (1)For the I-frames, how to produce an 8×8 DCT block for the reduced-size video from four 8×8 DCT blocks of the original video? (2)For P- and B-frames, how to produce a motion vector and residues for the new 16×16 macroblock from four 16×16 macroblocks? For the first problem, several computationally efficient solutions have been proposed in [155]. For the second problem, we need to compute one motion vector from four motion vectors and to produce the DCT of the residues for one macroblock in the reduced-resolution video given the DCT of the residues for four macroblocks in the original video. In [153], an algorithm was proposed to estimate the motion vector of the reduced-resolution video by using a weighted mean of the original motion vectors. The DCT of the residues is computed by reconstructing the P- and B-frames for both the original and the reduced resolution, and by using the estimated motion vectors. The computation of this closed-loop DCT domain approach is still rather costly.

We will focus in this thesis on the P-frames; B-frames can be treated similarly. We first propose an adaptive motion estimation scheme to approximate the motion vectors of the reduced-resolution video using the motion information from the corresponding blocks in the original full-resolution video as well as their neighboring blocks. This idea is similar to the overlapped block motion compensation in [156]. We then propose a transform domain open-loop approach to produce the DCT of the residue, thus eliminating the need to reconstruct the frames as in [153]. After downsizing the original four-block residues to one-block, we use subblock motion information to improve the image quality. As the subblock motion information is not compatible with MPEG-like standards, it is sent as side information using data hiding so as to be compliant with video encoding standard. The transcoded bit stream can be decoded by standard decoder with reasonable visual quality. Better image quality can be obtained, however, with a customized decoder after extracting the hidden information. Because the residue is computed in an open-loop manner, there is a tendency of error accumulation, particularly when there is considerable motion and/or large GOP. To overcome this, the GOP structure is modified to reduce the number of frames between two successive I-frames.

In the following, we shall elaborate the use of data hiding for conveying subblock motion information.

8.1.2 Embedding Subblock Motion Information

When the motion vectors are changed, obtaining accurate residue information usually requires reconstructing the entire frames. Directly downsizing residues saves computation in frame reconstruction, but since it is different from the accurate residue, motion compensation may not produce good result if only a single motion vector is used (as shown in Figure 8.1(b)). If all four motion vectors can be used, as shown in Figure 8.1(c), the resulting motion compensation would be better and image quality can be significantly improved. This is similar to the use of subblock motion compensation in [157]. For our current problem, we would like to send u, as well as the differences $u - u_i$ where $i = 1 \ldots 4$. However, the syntax of MPEG does not allow subblock motion information to be included. To maintain compatibility with MPEG-like standards, the subblock motion information can be sent in the user data part of the stream. This would maintain image quality but at the expense of increasing the bit rate. A standard decoder would give reasonable visual quality, while a customized decoder would be able to extract the side information and to produce improved images.

We propose to send the subblock motion information as side information using *data hiding*. Specifically, we embed the subblock information in the DCT residues.

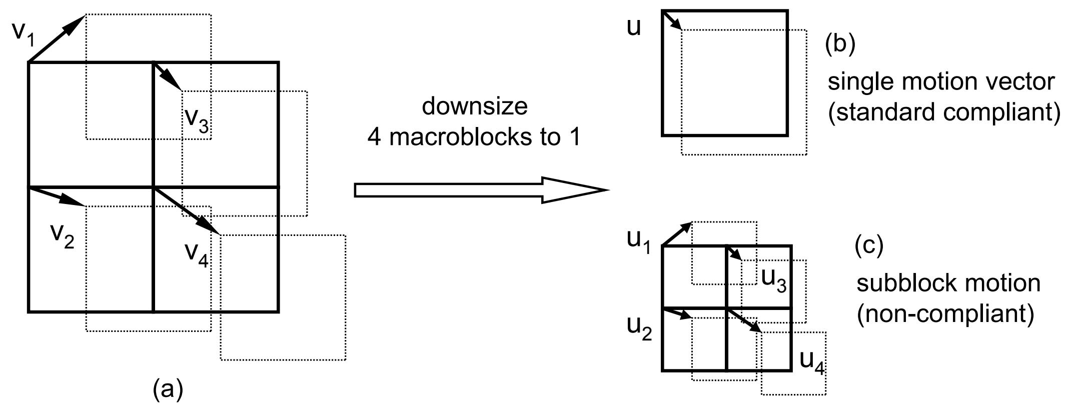


Figure 8.1: Relationship among motion vectors of an original picture and its downsized version

Since modifications of high frequency DCT coefficients tend to produce less noticeable artifacts, we can embed the side information in these coefficients, keeping DC coefficients and low order AC coefficients unchanged. One way to send the difference between u and u_i is to encode it in the highest frequency coefficient of the i^{th} subblock DCT residue whose quantized value is non-zero, i.e., we replace the coefficient with the motion vector difference. This embedding preserves the efficiency in run-length coding, hence introduces little overhead in terms of the bit rate of the video. The horizontal and vertical motion components are encoded separately by spliting the coefficients of a block into two parts and encode one motion component in one part.

8.1.3 Advantages of Data Hiding

We mentioned above two ways of conveying side information, namely, attaching it separately such as in user data field, and embedding in the media source. In Chapter 3, we discussed the bit reallocation nature of data hiding and explained that embedding does not offer "free" bandwidth. Instead, the most obvious advantage of data hiding is that the hidden data is carried with the source in a seamless way. This close association enables conveying of side information while maintaining compliant appearance when no user data field is available. It also enhances security since the existence and/or the location of the hidden data can be made difficult to identify. Thus, for robust data hiding, an unauthorized person has to distort the host media by a significant amount to remove the hidden data.

The efficiency in encoding side information is another advantage of data hiding. It is possible to embed the secondary data for a particular region $\mathcal{A}$ in $\mathcal{A}$ or another region that has deterministic relation with $\mathcal{A}$. Such association can help us save overhead in encoding the region index if the secondary data is sparsely distributed and separately encoded. In addition, when the side information is directly put in user data field (if available), the total number of bits will increase (Fig. 8.2-top). To keep the total number of bits almost identical to the original one, the media source has to be transcoded into a lower rate (Fig. 8.2-middle). Such transcoding is not a trivial task because sophisticated rate control may have to be involved for a good tradeoff between bit rate and visual quality. As illustrated in Fig. 8.2 (bottom), hiding side information in the media source via a properly selected embedding mechanism is a more convenient way to preserve the total bit rate due to its bit re-allocation nature.

8.1.4 Experimental Results

Our implementation is performed on MPEG-1, while the extension to other DCT based hybrid video coding is straightforward. We tested our approach using the two well-known sequences "football" and "table tennis" with a 15-frame GOP. The picture size is 352×224 for the original sequences and 176×112 for the reduced-resolution

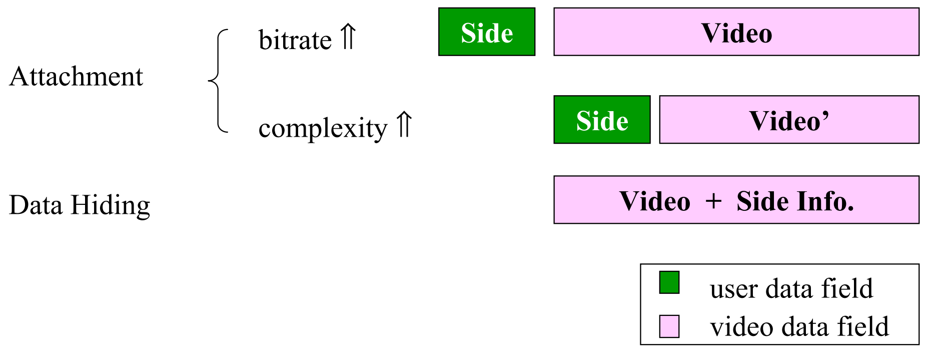
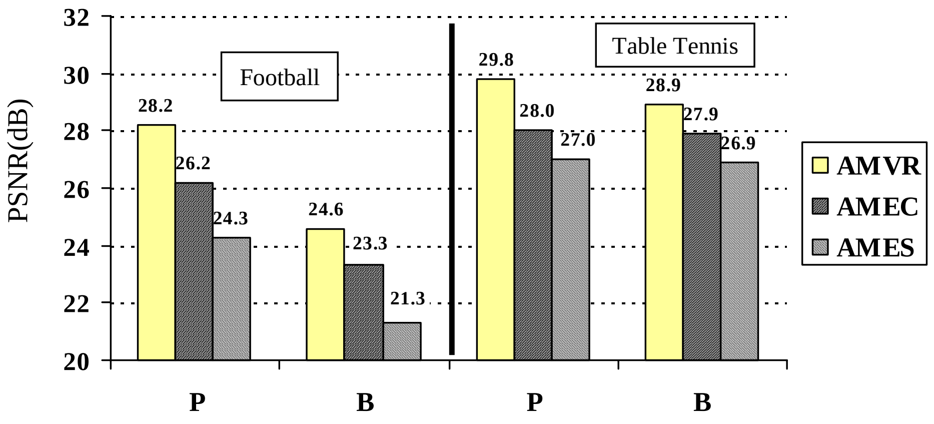


Figure 8.2: Comparison of sending side information via data hiding vs. attaching to user data field.

sequences. The quantization scaling factor for I-frames is 8, for P-frames 10, and for B-frames 25. We compare three schemes listed in Table 8.1, where AMVR is proposed in [153] and the rest are our proposals.

Fig. 8.3(a) shows the average PSNR for the prior art AMVR, our approach decoded by standard MPEG decoder (AMES) and by a customized MPEG decoder (AMEC). AMEC has a PSNR gain up to 2 dB over AMES due to the use of embedded subblock motion vectors extracted from DCT residues. However, when compared with the more complex AMVR, AMEC loses up to 2 dB. This not only shows a tradeoff among quality, complexity and bit rate, but also demonstrates the limitation of using the open-loop method to compute the DCT residues. However, when the original video is encoded at a high bit rate and is of high quality, the gap of PSNR between using AMVR and AMEC would be much smaller. In either case, no obvious visual difference is observed between AMEC and AMVR, as shown in Fig. 8.3(b). The figure also shows that some artifacts may appear in the video decoded by a standard decoder (the AMES case) due to the motion compensation residue not matching the motion

(a)



(b)

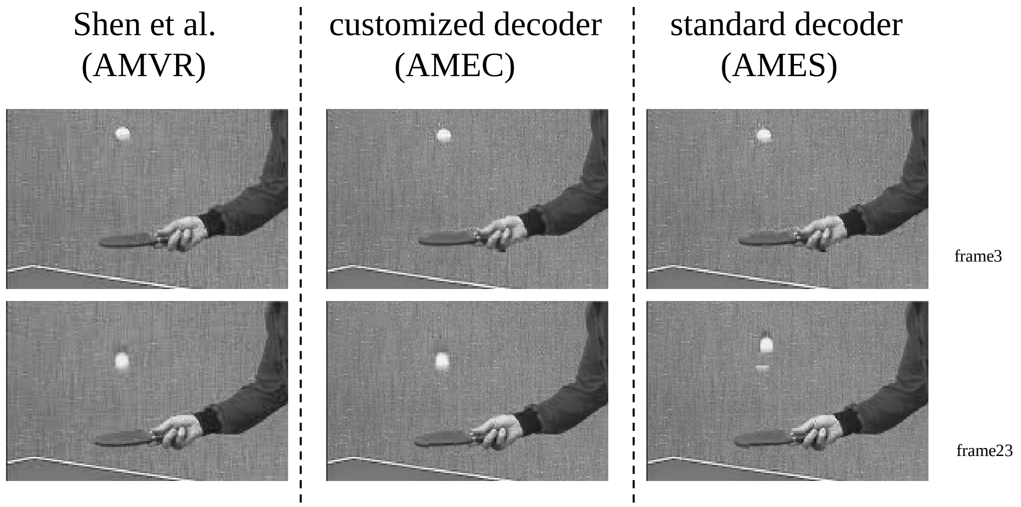


Figure 8.3: Performance comparison of various transcoding methods: (a) PSNR, and (b) visual effect.

Table 8.1: List of three schemes for experimental comparison

	full name	*use what motion vectors*	*use what residues*	*decoder*
AMVR	adaptive motion vector resampling	weighted average of 4 original motion vectors	accurate residue by reconstructing P/B frames	standard compliant
AMEC	adaptive motion with embedding & customized decoder	weighted average of 4 original motion vectors and neighbours	downsized residues from 4 orig. blocks with embedded motion	customized, using embedded info. to reconstruct frames
AMES	adaptive motion with embedding & standard decoder	weighted average of 4 original motion vectors and neighbours	downsized residues from 4 orig. blocks with embedded motion	standard compliant without using embedded info.

vector. This is expected since the core idea of the proposed scheme is not to optimize the quality obtained by a standard decoder. Despite of the artifacts, the overall visual quality by a standard decoder is reasonable with about 30% computation being saved over the prior art AMVR. In addition, the quality is improvable when a customized decoder is available.

8.2 Error Concealment and Data Hiding

We mentioned earlier that error resilience is important when transmitting image and video over unreliable networks such as the Internet and the wireless channels. The corrupted regions usually take the form of blocks or strips due to the block coding nature of the popular image/video codecs. Techniques for combating transmission errors have been classified into three categories [159, 160]: (1) sender-side techniques to make the encoded bit stream resilient to potential errors, (2) receiver-side techniques to conceal or alleviate the negative effects incurred by errors, and (3) interactive techniques involving both sender and receiver. Among the receiver-side techniques, temporal and spatial interpolations are principal tools to conceal small corrupted regions based on the inherent temporal and spatial correlation within the multimedia sources. Various interpolation approaches for the purpose of concealment have been proposed, each with different tradeoff between complexity and recovered perceptual quality. In the case of spatial interpolation, namely, to conceal the corrupted blocks from the surrounding uncorrupted blocks, the simple bilinear or bicubic interpolation has such problems as blocky artifacts and blurring edges. Edge directed interpolation proposed in [144, 145] improves the perceptual quality of recovered images/videos by estimating the major edges in the corrupted blocks and by avoiding interpolation across edges. This basic idea is illustrated in Fig. 8.4, and an example of concealing lost blocks in Lenna image is shown in Fig. 8.5. Note that the improvement in visual quality is at an expense of computation complexity. About 30% computation in the error concealment algorithm of [145] is for estimating edge information from surrounding blocks [130].

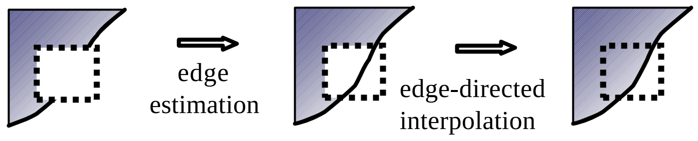


Figure 8.4: Illustration of edge directed interpolation for concealing lost blocks

8.2.1 Related Works

One of the first associations between error concealment and data hiding is found in [36], where data hiding is used to store a score in each block when encoding an

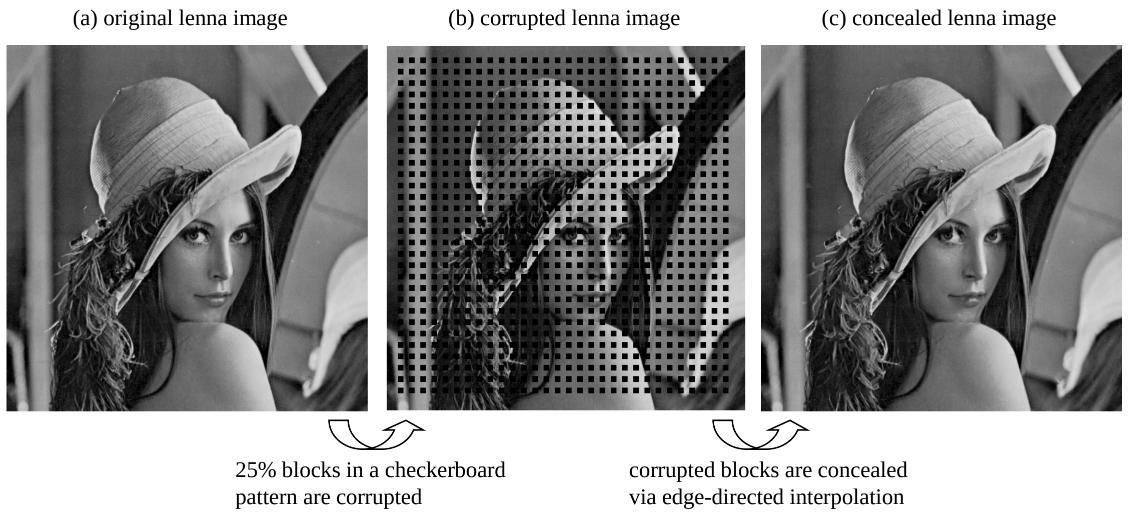

(a) original lenna image
(b) corrupted lenna image
(c) concealed lenna image
25% blocks in a checkerboard pattern are corrupted
corrupted blocks are concealed via edge-directed interpolation

Figure 8.5: An example of edge directed interpolation for concealing lost blocks using the techniques in [145].

image/video to a standard compliant bitstream. The score indicates the effectiveness that the associated block can be concealed using the surrounding information. When real-time transcoding (or *dynamic rate shaping*) to a lower bit rate is needed, the blocks with high embedded scores are dropped with high priority. Smooth blocks and blocks with simple edges that are inferable from surrounding blocks are assigned high priority because they can easily be concealed from surrounding blocks at the receiver side. The score can be embedded in DCT coefficients using simple Type-II approaches like the odd-even enforcement, as there is no much requirement on robustness. While the priority could be analyzed on the fly, embedding into the image/video the resulting scores from a pre-analysis saves computation in real-time transcoding and maintains the decodability of the image/video by a standard decoder.

The idea of balancing computation became the motivation in a recent error concealment work [130]. As can be seen from the earlier discussion, a potential hurdle against adopting sophisticated error concealment scheme in practical systems is their computational complexity since the computation power and the computation time available at a decoder are usually quite limited in many applications. It is desirable to shift some computation from the receiver side to the transmitter side. In [130], the edge information of an image/video block is proposed to be embedded in a companion block before transmitting the image over a lossy channel. The embedded edge information can be extracted on the receiver side and be used to recover corrupted blocks without the need of estimating the edges from the neighbors. This significantly reduces a decoder's burden in inferring the edge information from neighboring blocks for the purpose of concealment. Furthermore, the embedded edge is more accurate than the estimation by a concealment module. This is because the embedded edge is generated by the sender who has the knowledge of all blocks, while the receiver has to estimate the edge information of a lost block using the surrounding blocks only.

8.2.2 Proposed Techniques

Two novel uses of data hiding associated with error concealment are discussed in this thesis. One is for studying the robustness of data hiding algorithms via attack, namely, to use block concealment as a tool to remove the embedded watermark. For consistency in organization, we postpone the discussion to Part III, where the attacks and the countermeasures are addressed in detail. Another use of data hiding is to embed parity bits to recover a small number of bit errors in motion vectors. We summarize below the basic idea of this motion vector protection, which is one of the modules in an error concealment system for transmitting video over Internet. The detailed design and experimental results of the system can be found in [172].

As we know, motion estimation and compensation are used in most video coding standards to reduce the temporal redundancy between video frames. A key to this redundancy reduction is to estimate and to encode motion vectors, which indicate the displacement between blocks in the current frame and their best matching blocks in a reference frame. During video transmission, the accuracy of motion vectors is critical in ensuring the high quality of the received pictures [160]. Song *et al.* proposed to insert parity bits across Group of Macroblocks (GOBs) in one frame [42, 128], which does not provide sufficient protection for channels with packet loss. Considering channels that are subject to bursty errors, we propose to generate frame-wise parity bits from motion vectors of P-frames and to embed them in the successive I frames for the purpose of recovering some errors in the received motion vectors. More specifically, we take MPEG-1 as an example, in which the encoded motion vectors are differentially Huffman coded [19]. Because a video is transmitted in packets over networks, the lost of or the errors in motion vectors can be identified by lower layer protocols and we shall focus on correcting as many errors as possible. The error correction is achieved via parity bits in our work. To compute the parity bits, we first arrange the coded bits of the motion vectors of each P frame row by row, and pad zeros to each row to make them equally long. For the convenience of discussion, we denote the i-th bit of the motion vector in the j-th row of the k-th P-frame within a group-of-picture (GOP) as $V_j^{(k)}(i)$. The parity bits of the motion vectors in a GOP are computed as:

$$p_j(i) = V_j^{(1)}(i) \oplus V_j^{(2)}(i) \oplus ... \oplus V_j^{(N_P)}(i) \tag{8.1}$$

where $\oplus$ is the modulo-2 sum, and $p_j(i)$ is the i-th parity bit of the j-th row for a total of N_P P-frames. The modulo-2 sum enables us to correct a single bit error among N_P bits after locating the error. We choose to convey the parity bits to decoder by embedding them in the DCT coefficients of the successive I-frame. As have been discussed in Section 8.1.3, using data hiding to convey the parity bits avoids the increase of video bit rate (hence the increase of the network load) and avoids computationally expensive rate control.

Due to the congestion and other dynamic change of channel conditions during the transmission of video over the Internet and the wireless channels, the packet loss tends to occur in burst. In addition to protecting motion vectors, we also make use of interleaving during packetization to reduce the occurrence that adjacent blocks get corrupted simultaneously. More specifically, we take N-block by M-block as a unit, and pack these blocks into N packets in such a way that the blocks in consecutive packets are not side by side with each others. The packetization satisfying this condition enables us isolating lost blocks when several consecutive packets get corrupted or dropped, therefore allows a better recovery via interpolation using the information from surrounding blocks. The edge-directed interpolation reviewed at the beginning of this section is used in our concealment system [172].

8.3 Chapter Summary

In this chapter, we discussed the use of data hiding beyond traditional applications. We use two specific examples, namely, transcoding via downsizing and error concealment to demonstrate the idea of using data hiding to send side information. The side information helps to achieve better performance by a customized decoder and in the mean time retains the decodability with reasonably good quality by a standard-compliant decoder.

Quite a few business models can be supported by the customized decoding framework. For example, the embedded data may consist of both access control policies and electronic coupons. The embedded coupons aim at encouraging and rewarding customers to follow the access control policies. Data hiding can also be used, possibly with encryption or scrambling like those in [161, 162], for various access control purposes. This is a promising direction to be explored for the "Digital Rights Management (DRM)" of multimedia contents.

Acknowledgement

The transcoding and error concealment works were jointly done with Peng Yin at Princeton University.

Part III

Attacks and Countermeasures

Chapter 9

Attacks and Countermeasures on Known Data Hiding Algorithms

An attack on a watermarking system is to obliterate the watermark so that the original goal of embedding watermarks cannot be achieved. In general, the attacks test the robustness and security of the entire data hiding system, from the embedding mechanism to the system architecture. For robust watermarking, this means the detector is unable to detect the existence of watermark or there is ambiguity in making a definite decision. An effective attack does not have to remove the watermark. One simple example is to cause miss synchronization via jitter [137].

It is important to understand that the attacks are meaningful mainly for applications in rivalry environment where there are incentives to obliterate the watermarks. This includes: (1) ownership protection where an adversary is a pirate, (2) tampering detection where an attacker would like an authentication system to regard an unauthorized multimedia source or a tampered one as authentic, and (3) copy/access control where an adversary would like to copy or access a protected multimedia source in such a way that is not compliant with the specified policies. Applications in a non-rivalry environment such as annotation and enhancing communication performance

(Chapter 8) usually are not subject to attacks, although there may be specific robustness requirements such as surviving certain lossy compression. For applications in rivalry environment, finding effective attacks and analyzing them play an important role in identifying the weaknesses and limitations of watermarking schemes, as well as in suggesting directions for further improvement. More importantly, it helps us to reach a realistic understanding regarding what data hiding can do and cannot do for the purposes of ownership protection, tampering detection, and copy/access control.

A number of attacks as well as some countermeasures have been reported in the literature. Most of the previous attacks and the attacks presented in this chapter target at specific types of watermarking schemes, for which analysts have full knowledge of the watermarking algorithms. The analysts are able to perform experiments with many non-watermarked, watermarked, and attacked samples, and to observe the results in real time. In the next chapter, we will discuss attacks under an emulated rivalry environment in which analysts have no knowledge of the watermarking algorithms.

Three attacks and countermeasures are discussed in this chapter. The block replacement attack in Section 9.1 targets on removing robust watermarks embedded locally in an image. Geometric attacks on robust image watermarks have been considered as a big challenge in literature. In Section 9.2, we present a countermeasure by embedding and detecting watermarks in a domain that is resilient to rotation, scale, and translation. The double capturing attack in Section 9.3 outlines a general attack for forging fragile watermarks.

9.1 Block Replacement Attack on Robust Watermark

Claiming to “provide legal protection that the international copyright community deemed critical to the safe and efficient exploitation of works on digital networks”, the *Digital Millennium Copyright Act* (DMCA) [25], signed into law on October 28, 1998, prohibits “circumvention of technological measures used to protect copyrighted works”, and prohibits “tampering with the integrity of copyright management information” [1]. While the technologies emphasized on by the DMCA Act are cryptographic approaches such as encryption, deliberately designing tools to circumventing data hiding based copy/access control mechanisms is also a potentially illegal behavior. However, legitimate tools may be exploited by adversaries to build powerful attacks. This section will discuss such an attack via block concealment techniques originally designed for error resilient image/video transmission. Before we discuss the details of the attack, we shall give a brief review of the attacks on robust watermarks proposed in literature.

9.1.1 Existing Attacks on Robust Watermarks

Several attacks as well as some countermeasures have been reported in the literature. Forging a fake “original” image for multiple ownership claims [132] can be thwarted by imposing invertibility requirement on watermarks [132] and/or exploiting more than one detection scheme including blind detection (i.e., without the use of original unmarked image) [52]. Collusion attack involves the averaging of multiple copies of the same original but having different (independent) markings [135]. When the

[1] The DMCA act states a number of exceptions including good faith research that is “necessary to identify and analyze flaws and vulnerabilities” of the encryption technologies.

detector is in the public domain, it is possible to systematically learn about the watermarks from the input-output relationship of a detector using many manipulated versions of a watermarked image [136]. Watermarks can also be attacked by geometric distortion, including rotation, scale, translation, warping, line dropping/adding, or in conjunction with moderate lowpass filtering and interpolation [137, 138], but may not be always effective when the original image is available to perform registration.

9.1.2 Attack via Block Replacement

We proposed a new method that can remove robust invisible watermarks that are locally inserted. Unlike the existing work in literature, our attack does not require multiple copies of marked images or other restrictions in detection. The idea is to replace image blocks by an interpolated version using neighbors. As we have seen in Chapter 8, the replacement algorithm was originally proposed for error concealment (i.e., to recover lost or corrupted blocks) [144, 145] and for low bit rate coding [36]. Blocks surrounding the lost block are used to infer edge information and then to give an edge-directed interpolation of the missing block, as illustrated in Fig. 9.1. In the proposed attack, we keep all blocks at the border of an image untouched, but replace all other blocks by their interpolated version using the method in [145]. The replacement is performed on block basis. As illustrated in Fig. 9.2, we first divide an image into 4×4 or 8×8 blocks. For each block except the one on the image border, an interpolated version is obtained from the neighboring blocks. The original block is then replaced by its interpolated version. The replacement can be done selectively, for example, the original block will be retained if the interpolated version has too many artifacts. We can see that such block-by-block replacement essentially removes the watermarks originally embedded into each block.

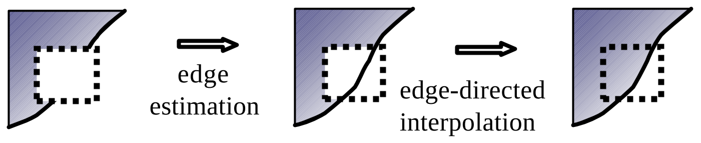


Figure 9.1: Illustration of edge directed interpolation for concealing lost blocks

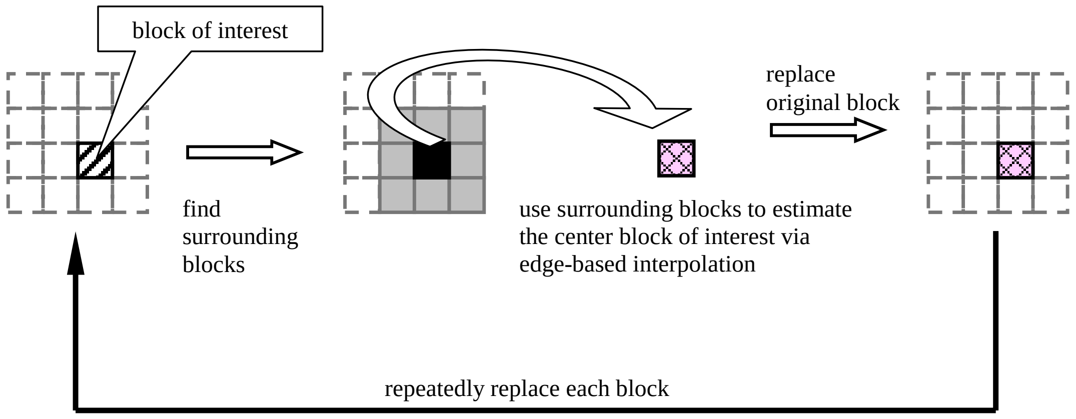

block of interest
find
surrounding
blocks
use surrounding blocks to estimate
the center block of interest via
edge-based interpolation
replace
original block
repeatedly replace each block

Figure 9.2: Block diagram of the proposed block replacement attack

Shown in Fig. 9.3(a) is a watermarked Lenna image using the block-DCT domain additive spread-spectrum method of [47] resulting in a PSNR of 42.1 dB with reference to the unmarked original. The detection statistics is 138.5 if the original is used, and is 19.3 if the original is not used. The detection threshold is generally set between 3 and 6 for a maximal detection probability with a false alarm probability of 10^{-3} to 10^{-10}. The marked image Fig. 9.3(a) is then compressed using JPEG with quality factor 10%. The result, shown in Fig. 9.3(b), has a PSNR of 29.40 dB. The two detection statistics are reduced to 34.96 and 12.40 respectively, both still well above the threshold in spite of the severe visual distortion. This confirms the claim that the spread spectrum watermark algorithm survives well under severe JPEG compression [47]. The proposed attack is then applied to the marked image, and the result, shown in Fig. 9.3(c), has a PSNR of 29.58 dB. Aside from the softer appearance, few artifacts are observed and the image is much more pleasing than Fig. 9.3(b). However, the two detection statistics have been reduced to 6.30 and 4.52, much below those from Fig. 9.3(b) and comparable with the threshold. The results are summarized in Table 9.1.

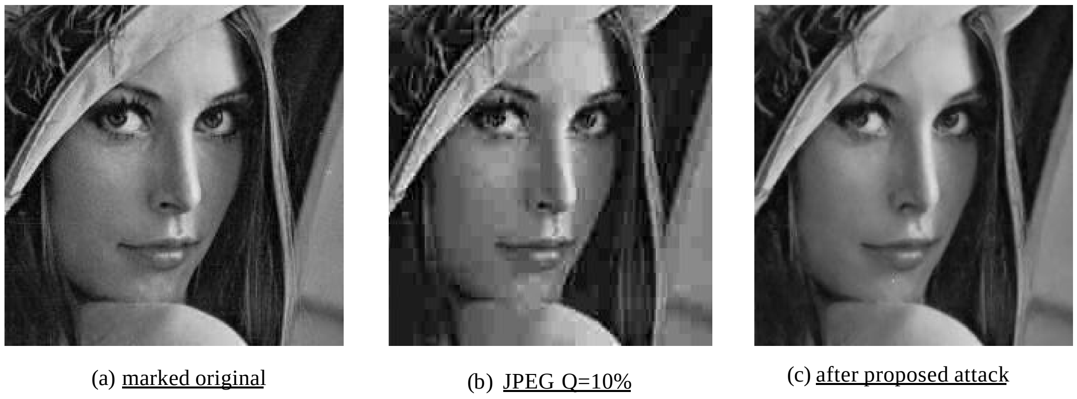


Figure 9.3: Watermarked images and distorted versions by JPEG and proposed block replacement attack: (a) watermarked Lenna image by Podilchuk-Zeng scheme; (b) watermarked Lenna after JPEG $Q = 10\%$ compression, image is extremely blocky and with poor quality; (c) watermarked Lenna after proposed attack (with block size 4×4), image is slightly blurred but with acceptable quality.

9.1.3 Analysis and Countermeasures

The proposed attack can be viewed as a non-linear low pass filtering. It makes use of the fact that images, as with other perceptual sources, is highly correlated in pixel domain and can tolerate distortions within the just-noticeable-difference. This makes it possible to make good inference from surrounding samples when small parts of

Table 9.1: Experimental results of block-replacement attack on block-DCT based spread-spectrum watermarking. Normal detection threshold for presence of a watermark is $3 \sim 6$.

	Fig. 9.3(a) w/o distort.	Fig. 9.3(b) JPEG 10%	Fig. 9.3(c) attacked
PSNR w.r.t. unmarked	42.12 dB	29.40 dB	29.58 dB
detection statistics with orig. avail.	138.51	34.96	6.30
detection statistics without orig. avail.	19.32	12.40	4.52

an image are lost. The information embedded in a local region will be lost with the removal of pixels in that region even if the embedding is not done in the pixel domain (e.g., block DCT domain embedding). The proposed attack can be extended to multi-resolution based watermarking schemes.

There are several ways to reduce the effectiveness of the proposed attack. Our block replacement based attack assumes that local information can be inferred from neighbors. This is true for smooth regions and for regions with edges extending across several blocks. However, small features that fall in a single block cannot be so inferred. This suggests the use of larger blocks for local embedding to combat the proposed attack. Embedding watermark in 16×16 block DCT coefficients, for example, would be much more difficult to attack than embedding in 8×8 block DCT coefficients. Also, the proposed attacks are not very effective for low-resolution images in which many key features are small. In this connection, it should be noted that the block replacement algorithms in [144, 145] is more effective for edge blocks than texture ones. Algorithms such as [146] may be used for blocks that are highly textured.

Our experiments have shown that the proposed attack is not effective for spread

Table 9.2: Effectiveness comparison of block replacement attack on global and local spread-spectrum watermarking.

embedding domain	block-DCT	whole-DCT
PSNR wrt unmarked	29.58 dB	29.36 dB
detection statistics with orig. avail.	6.30	24.74

spectrum embedding in the DCT transform domain of the entire image such as the one proposed by Cox *et al.* in [44]. To distinguish from the block-DCT based embedding, we shall call this *whole-DCT embedding.* Although the detection statistics of an image marked by whole-DCT approach decreases after the proposed attack, it is still well above threshold, as shown in Table 9.2. We can interpret the ineffectiveness of the proposed attack against whole-DCT embedding in a couple of ways. From pixel domain point of view, the watermark embedded in whole DCT domain has been spreaded throughout the image in pixel domain, implying that the watermark information in each block is correlated with that in other blocks. Therefore, a large part of the watermark information can still be recovered from the neighborhood during interpolation. From frequency domain point of view, while both block-DCT and whole-DCT embedding hides watermark in the low frequency DCT coefficients, the equivalent frequency range are different. A 1-D illustration of this issue is shown in Fig. 9.4. The DC coefficients after block-DCT transform correspond to not only the DC but also the low frequency bands of the entire image. The frequency bands represented by the lowest AC coefficients in each block are at higher frequency than those by the block DCs, implying that the "low-band" after block-DCT transform is not very low. Cox *et al.* have pointed out the importance of embedding watermark in perceptually significant spectrum (mostly low and middle frequency bands) to achieve robustness against distortion [44]. Embedding watermark in higher frequency, such as the block-DCT embedding, reduces the robustness against low-pass filtering. In fact, the poorer performance of the block-DCT embedding against linear low-pass filtering than the whole-DCT has been reported in [36], and Liang *et al.* recently proposed to watermark the DC image formed by the DC coefficient of each block [51]. The experiments associated with our proposed attack indicate that the block-DCT embedding is also much more vulnerable to non-linear low-pass filtering than the whole-DCT embedding because of the higher embedding spectrum. In summary, while the transform domain embedding with small block size such as 8×8 has been considered superior to the whole image transform domain due to the ease of applying human visual system to finely tune watermark locally (see Section 6.3.1), our first countermeasure suggests the opposite, namely, the block size should not be too small.

Motivated by the robustness of whole-DCT embedding, another countermeasure can be devised by adding redundancy to the watermarks. For example, the same watermark can be embedded in a group of four blocks. Thus, the lost watermark could be partially recovered from neighbors, producing a higher detection statistic. However, the watermark or hidden data embedded via Type-II approach, namely, via deterministic relationship enforcement, may still not be recoverable.

More Effective Attacks

Various improvements can be done from attacker point of view. For example, an attacker can choose not to replace a block with the concealed version if the block involves key perceptual features such as eyes and/or the concealed version significantly differs from the original block. In addition, better interpolation methods for textured regions would also further improve the perceptual quality of the attack.

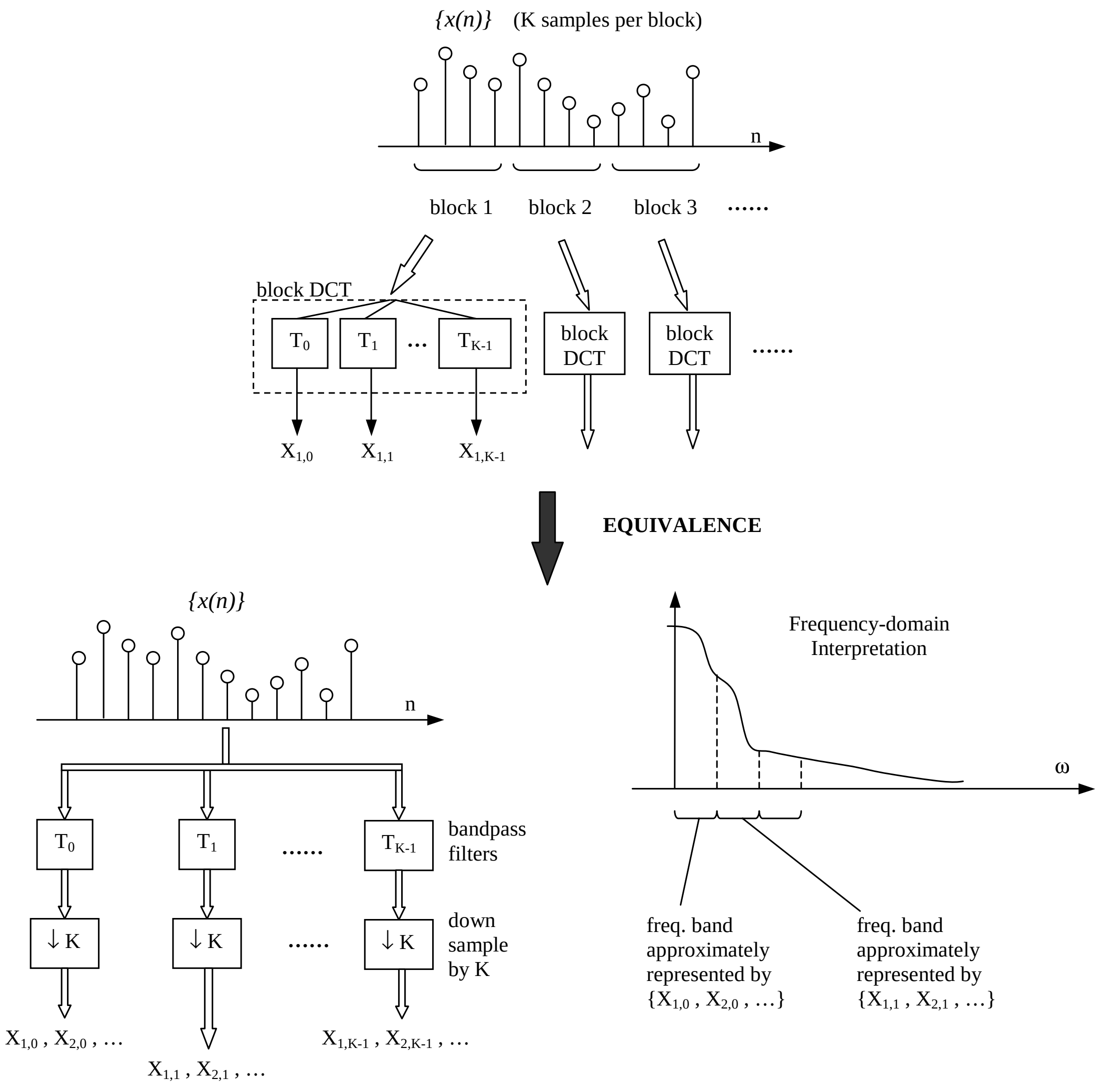

{x(n)} (K samples per block)
n
block 1
block 2
block 3
......
block DCT
T_0
T_1
...
T_{K-1}
block DCT
block DCT
......
$X_{1,0}$
$X_{1,1}$
$X_{1,K-1}$
EQUIVALENCE
{x(n)}
n
T_0
T_1
......
T_{K-1}
bandpass filters
↓K
↓K
......
↓K
down sample by K
$X_{1,0}$, $X_{2,0}$, …
$X_{1,1}$, $X_{2,1}$, …
$X_{1,K-1}$, $X_{2,K-1}$, …
Frequency-domain Interpretation
ω
freq. band approximately represented by {$X_{1,0}$, $X_{2,0}$, …}
freq. band approximately represented by {$X_{1,1}$, $X_{2,1}$, …}

Figure 9.4: Spectrum analysis of block based transform: the block DCT transform is equivalent to bandpass filtering followed by downsampling.

9.2 Countermeasures Against Geometric Attacks via RST Resilient Watermarking

In non-coherent detection where the original unwatermarked image is not available, the spread-spectrum robust image watermarks generally suffer from geometric distortion such as rotation, scaling, and translation (RST). A main reason for this is that some mis-alignment between the embedded watermark and the reference watermark presented to a detector is introduced by the RST attacks. Because spread spectrum watermarks generally have low autocorrelation at non-zero shift and taking correlation is the popular way to detect these watermarks, the mis-alignment will be likely to render low detection statistics from the popular correlator-type detectors. Among the three basic geometric distortions, namely, rotation, scaling, and translation, the resilience against translation is the easiest. Fourier magnitude is known to be invariant with respect to the shift in time or spatial domain, so embedding in this domain will be resilient against small shifts [2]. On the other hand, combating rotation and scaling is less straightforward. To illustrate the idea, we have implemented a spread spectrum embedding similar to the one in [44] but embedded the watermark in the magnitude of Discrete Fourier Transform (DFT) coefficients rather than the DCT coefficients. The results on 512×512 Lenna image are shown in Table 9.3, with the PSNR of watermarked image with respect to the original unwatermarked one being 42.88 dB. We can see that minor rotation and scaling are powerful enough to render the watermark undetectable, even though the watermark can survive strong compression and translation with detection statistics well above the threshold [3].

[2]Larger shifts are likely to incur cropping, which may reduce the detection statistics. We will discuss this in the experimental result section.

[3]The threshold is usually set between 3 and 6, corresponding to a false alarm probability of 10^{-3} to 10^{-10}.

Table 9.3: Detection statistics of spread spectrum watermarking on 512×512 Lenna image in DFT magnitude domain under various distortions.

Test condition	*Detection statistics*	*Test condition*	*Detection statistics*
w/ no distortion	13.54	w/ no wmk	1.31
right shift 5 pixels	13.23	rotate 1-degree	1.09
JPEG Q=70%	12.35	scale down by 5%	0.58
JPEG Q=30%	8.30		

Restricting ourselves to non-coherent detection in which the original unwatermarked image is not available to a watermark detector to perform registration with the test image (possibly rotated, scaled, and/or translated), it is difficult to estimate the parameters of the RST distortion and to undo the distortion. One way to combat RST attacks is to embed an invisible registration pattern [58, 59] which is known to detectors and helps to estimate the RST parameters. The weakness of this solution is that in a rivalry environment, an adversary may estimate this pattern by averaging multiple watermarked images that have the same registration pattern embedded in. He/She can then remove this pattern and apply geometric distortion.

We proposed a new alternative by embedding and detecting spread spectrum watermarks in a RST resilient domain. This resilient domain is motivated by special properties of Fourier transform and is closely related to Fourier-Mellin transform.

9.2.1 Basic Idea of RST Resilient Watermarking

We consider an image $i(x, y)$ and its RST version $i'(x, y)$ with

$$i'(x, y) = i(\sigma(x \cos\alpha + y \sin\alpha) - x_0, \sigma(-x \sin\alpha + y \cos\alpha) - y_0) \qquad (9.1)$$

where the rotation, scaling, and translation parameters are α, σ, and (x_0, y_0), respectively. The magnitudes of the Fourier transform of these two images, $|I(f_x, f_y)|$ and

$|I'(f_x, f_y)|$, have the following relations

$$|I'(f_x, f_y)| = \sigma^{-2}|I(\sigma^{-1}(f_x \cos\alpha + f_y \sin\alpha), \sigma^{-1}(-f_x \sin\alpha + f_y \cos\alpha))| \quad (9.2)$$

This tells us that (1) the Fourier magnitude is invariant to translation, (2) the Fourier transform of a rotated image is the rotation of Fourier transform of the image by the same angle, (3) the scaling in spatial domain gives the inverse scaling in Fourier domain [14]. If we rewrite Eq. 9.2 using log-polar coordinates, the image scaling results in a translational shift along the log radius axis and the image rotation results in a cyclical shift along the angle axis. That is,

$$|I'(f_x, f_y)| = \sigma^{-2}|I(\sigma^{-1}e^{\rho}\cos(\theta - \alpha), \sigma^{-1}e^{\rho}\sin(\theta - \alpha))| \quad (9.3)$$

or

$$|I'(\rho, \theta)| = \sigma^{-2}|I(\rho - \log\sigma, \theta - \alpha)| \quad (9.4)$$

where the coordinate transform is

$$\begin{cases} f_x = e^{\rho}\cos\theta \\ f_y = e^{\rho}\sin\theta \end{cases} \quad (9.5)$$

We define $g(\theta)$ to be a 1-D projection of $|I(\rho, \theta)|$ such that

$$g(\theta) = \sum_{\rho} \log(|I(\rho, \theta)|) \quad (9.6)$$

where a summation is used instead of an integration due to the discrete nature of ρ for the DFT of an digital image. We find it beneficial to add the two halves of $g(\theta)$ together, obtaining

$$g_1(\theta') = g(\theta') + g(\theta' + 90^o) \quad (9.7)$$

where $\theta' \in [0^o, ..., 90^o)$. The reason will be explained in Section 9.2.3.

Clearly, $g_1(\theta)$ is invariant to translation and to scaling except a multiplicative factor which does not affect correlator-type detection. Rotation results in a circular shift, which can be handled by an exhaustive search if the angle θ is quantized to finite number of degrees.

9.2.2 Embedding and Detection Algorithms

The basic idea presented above outlines a RST resilient watermark detector. The detector first extracts the 1-D signal $g_1(\theta)$ from a test image, then performs correlation-type detection between this 1-D signal and an input watermark [4]. The basic algorithm for watermark detection is summarized as follows:

1. Compute a discrete log-polar Fourier transform of the input image. This can be thought of as an array of K rows by N columns, in which each row corresponds to a value of ρ, and each column corresponds to a value of θ.

2. Sum up the logs of all the values in each column, and add the result of the column j to the result of the column $j + N/2$ $(j = 0, \cdots, (\frac{N}{2} - 1))$ to obtain an invariant descriptor $\underline{v}$, in which

$$v_j = g_1(\theta_j) \tag{9.8}$$

   where θ_j is the angle that corresponds to the column j in the discrete log-polar Fourier transform matrix.

[4] More sophisticated detection corresponding to a more realistic modeling of noise can be adopted to achieve higher detection performance. Here for proof of concept, we use correlation detector in our experiments.

3. Compute the correlation coefficient D between $\underline{v}$ and the input watermark vector $\underline{w}$, as

$$D = \frac{\underline{w} \cdot \underline{v}}{\sqrt{(\underline{w} \cdot \underline{w})(\underline{v} \cdot \underline{v})}} \tag{9.9}$$

4. If D is greater than a threshold T, it indicates that the watermark is present. Otherwise, the watermark is absent.

Corresponding to the watermark detection method, we can construct a watermark embedding algorithm according to the methodology described in [54]. In that paper, watermarking is cast as a case of communications with side information at the transmitter, which is a configuration studied by Shannon [98]. The side information in watermarking is the original unwatermarked image that is known to the embedder. In our problem, the embedder need to manipulate the Fourier coefficients in Cartesian coordinate to embed a selected watermark into the 1-D signal $g_1(\theta)$. In particular, we change the image coefficients in an iterative way such that the 1-D signal of the watermarked image will have high correlation with the watermark to be embedded. The detailed embedding follows the three steps in [54]:

1. Apply the same signal-extraction process to the unwatermarked image as will be applied by the detector, thus obtaining an extracted vector, $\underline{v}$. In our case, this means computing $g_1(\theta)$.

2. Use a mixing function, $s = f(\underline{v}, \underline{w})$, to obtain a mixture between $\underline{v}$ and the desired watermark vector, $\underline{w}$. At present, our mixing function simply computes a weighted average of $\underline{w}$ and $\underline{v}$, which is a convenient but sub-optimal approach. More sophisticated mixing methods, for example, those examined in [55], may be used.

3. Modify the original image so that, when the signal-extraction process is applied to it, the result will be $\underline{s}$ instead of $\underline{v}$. This process is implemented as follows:
   (a) Modify all the values in column j of the log-polar Fourier transform so that their logs sum to s_j instead of v_j. This could be done, for example, by adding $(s_j - v_j)/K$ to each of the K values in column j. Care must be taken to preserve the symmetry of DFT coefficients.
   (b) Invert the log-polar resampling of the Fourier magnitudes, thus obtaining a modified, Cartesian Fourier magnitude.
   (c) The complex terms of the original Fourier transform are scaled to have the new magnitudes found in the modified Fourier transform.
   (d) The inverse Fourier transform is applied to obtain the watermarked image.

Unfortunately, there is inherent instability in inverting the log-polar resampling of the Fourier magnitude (Step 3b). We therefore approximate this step with an iterative method in which a local inversion of the interpolation function is used for the resampling [169].

9.2.3 Implementation Issues

A number problems arise during the implementation of the watermarking algorithms proposed in last section. We summarize the handling of a few issues below. More detailed discussion can be found in [169].

DFT, Rotation, and Interpolation

In Section 9.2.1, we present the basic ideas of RST resilient watermarking in terms of continuous Fourier transform. In practice, we have to deal with discrete samples both in spatial domain and in frequency domain. We would also like to take advantage of fast algorithms for computing the DFT. Conceptually, the DFT of an image is obtained by taking a 2-D Discrete Time Fourier Transform (DTFT) of a tiled version of the image, as shown in Fig. 9.5(a). Stone *et al.* [149] have noted that the tiling has an inherent problem for any algorithm that relies on the rotational properties of the Fourier transform. This is because when the image is rotated, the rectilinear tiling grid is not rotated along with it. Thus, the DFT of a rotated image is not the rotated DFT of the image. The problem is illustrated in Fig. 9.5(b) and (c). While more sophisticated approaches are possible, such as directly computing the log-polar Fourier transform without using Cartesian DFT as an intermediate step, the computational complexity is generally high. In our work, we approximate the Fourier transform of an image in log-polar coordinate by resampling the image DFT in Cartesian coordinate using a log-polar grid.

In general, interpolation has to be performed on Fourier coefficients during both embedding and detection. The interpolation is needed not only when obtaining log-polar sampling points from the Cartesian sampling points, but also when obtaining Cartesian sampling points from the log-polar sampling points in the embedding step 3b. To obtain dense sampling grids that would allow inexpensive interpolation such as bilinear interpolation, we pad the image with zeros before performing DFT. Zero-padding also adds larger separation between the implicit tiles in the spatial domain, alleviating the distortions shown in Fig. 9.5.

Cross Artifacts in Spectrum

The rectangular boundary of an image is known to have *cross artifacts* in the image's Fourier spectrum (Fig. 9.6). The coefficients along the cross have much larger magnitude than the others. This is partly due to the dominant horizontal and vertical

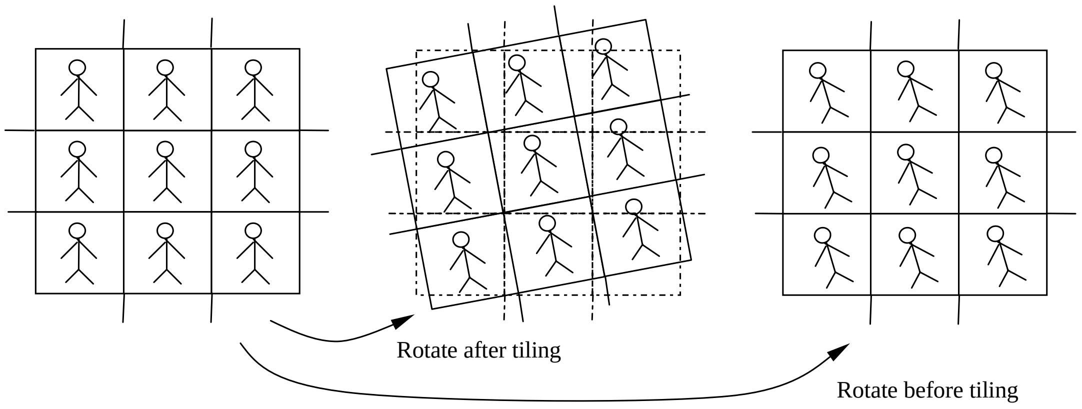


Figure 9.5: Rectilinear tiling and image rotation

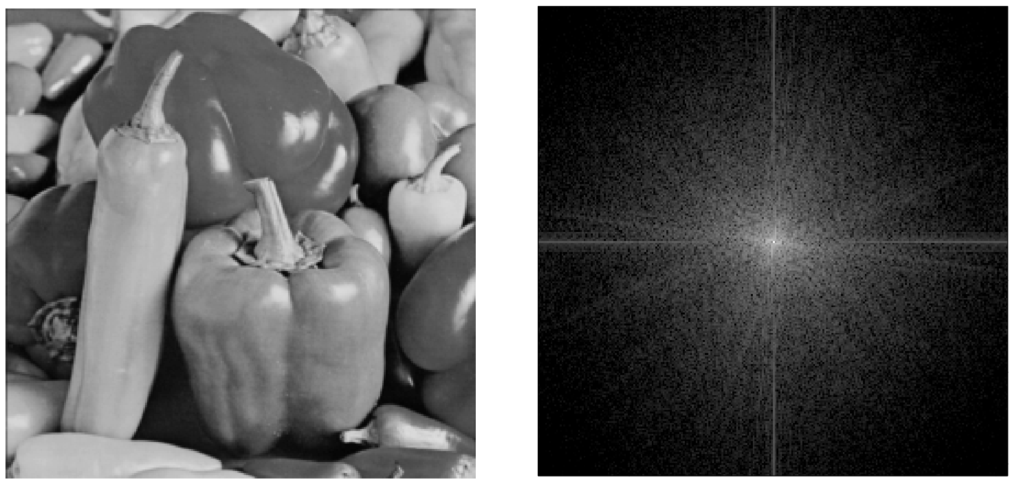


Figure 9.6: An image and the magnitude of its 2-D Discrete Fourier Transform. Coefficients near the DC and low frequency band along horizontal and vertical directions exhibit large magnitude, which is refered as “cross artifacts”.

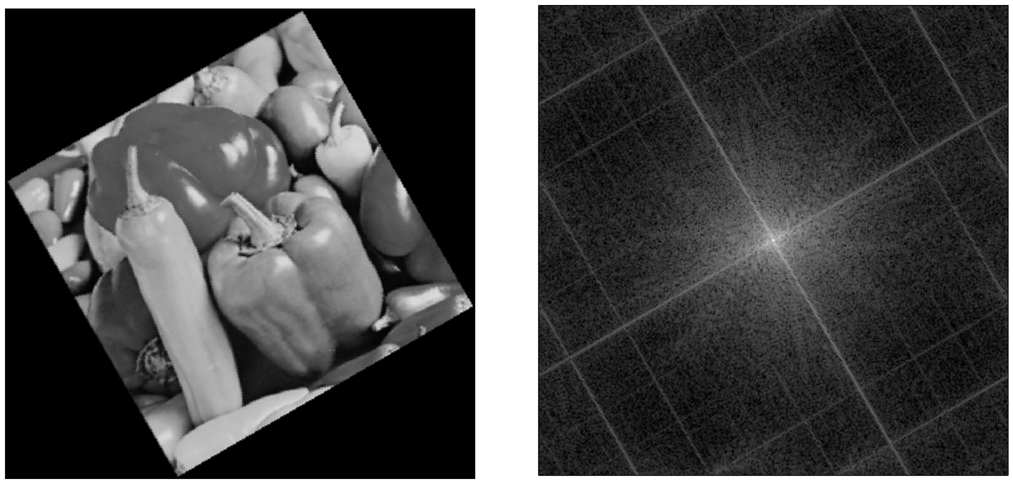


Figure 9.7: A rotated image with zero padding and the magnitude of its 2-D DFT. The “cross artifacts” also rotates by the same degree in this example.

image features, as well as to the discontinuity at horizontal and vertical edges of an image by the implicit tiling. When zero padding is used for obtaining finely sampling grids and/or for image rotation without cropping, the cross artifacts, possibly rotated, are more significant (Fig. 9.7). The artifacts may also be asymmetric if an image has much larger energy in some directions than in other directions. For example, images with significant vertical structures like trees and buildings yields more energy in the horizontal frequency, while images with strong horizontal structures like seascape yields more energy in vertical frequencies (Fig. 9.8). Our current solution to this problem is to simply ignore the bumps in the extracted 1-D signal by dropping a neighborhood around each of the two highest-valued elements. We also divide the extracted signal $g(\theta)$ into two halves and add then together (i.e., to use $g_1(\theta)$ in Eq. 9.7 instead of $g(\theta)$ in Eq. 9.6) to deal with the asymmetry. Alternative solutions include blurring of the image edges [150] or more generally, applying windowing operation. These solutions require modification to the watermark embedder to include both forward and inverse operations, and have been left for future work.

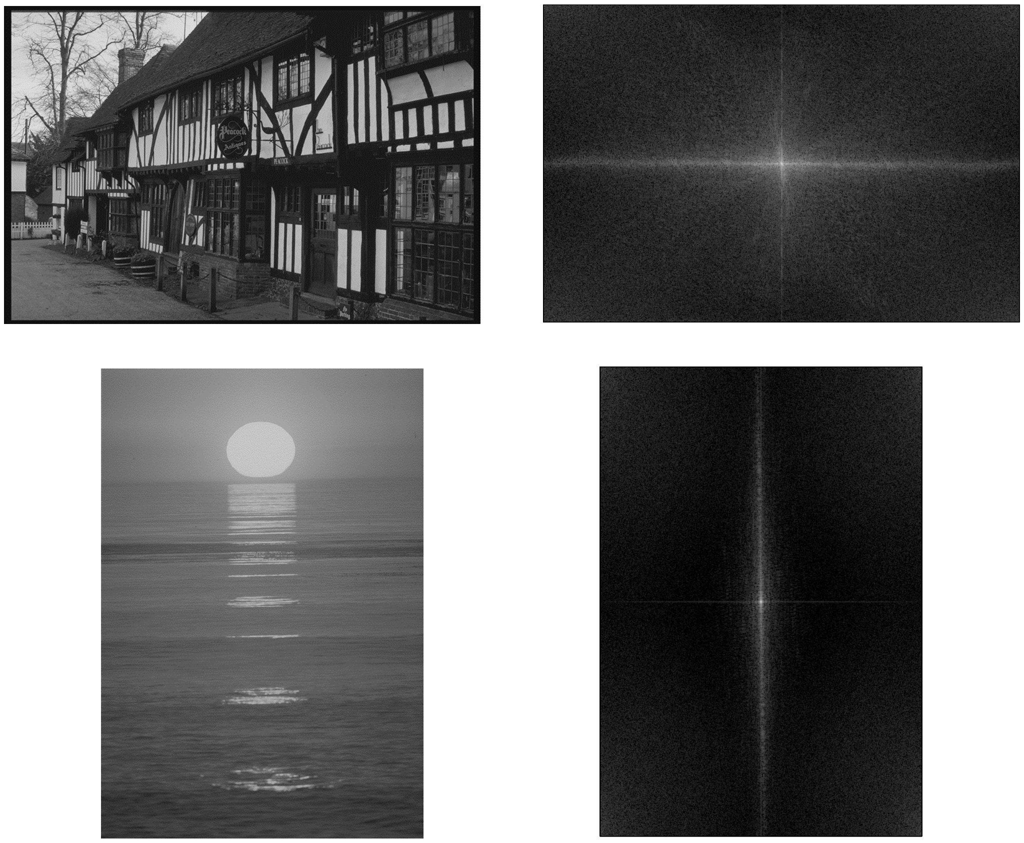


Figure 9.8: $\frac{a|b}{c|d}$ Images with dominant structures and their 2-D DFTs. An image with dominant vertical structure (a) yields strong magnitude along horizontal frequencies (b). An image with dominant horizontal structure (c) yields strong magnitude along vertical frequencies (d).

Coefficient Dynamic Range and Extreme Frequencies

Because of the large dynamic range of the magnitude of the DFT coefficients, the low frequency coefficients can be overwhelming. Furthermore, the lowest frequencies and highest frequencies are usually not reliable for watermarking because of the strong host interference for low frequencies and the vulnerability under distortion and attacks for high frequencies. Our current solution is to neglect the extreme frequency bands and not to embed watermark there. A better solution is to embed watermark in all frequencies and weigh different bands differently according to the reliability, as discussed in Section 6.2.1. We also use the log of magnitude rather than the magnitude to obtain the 1-D function $g(\theta)$ (Eq. 9.6).

Whitening Before Detection

For natural images, $g_1(\theta)$ is likely to vary smoothly as a function of θ. This indicates that the noise term in blind watermark detection is highly correlated and a simple correlation detection is not optimal for such noise. Assuming that the noise is colored gaussian, the optimal detection according to the detection theory should whiten the noise first, then perform correlation detection. This idea has been discussed in [56], showing improvement in the watermark detection. Thus, in the detection stage of our work, we apply a whitening filter to both the extracted signal and the watermark being tested before computing the correlation coefficients. The whitening filter was designed to decorrelate the elements of the 1-D signals extracted from natural images and was derived from 10,000 images in [151]. These training images were not used in the subsequent experiments.

9.2.4 Experimental Results

The following results were obtained by extracting a vector $g_1(\theta)$ of length 90 from an image and neglecting the 16 samples surrounding the peak that corresponds to the DFT cross artifacts. This leaves a 1-D descriptor of 74 samples long. The detection process involves comparing the watermark with all 90 cyclic shifts of the extracted descriptor. In this section we examine the fidelity, the false alarm (also called false positive) behavior, and the robustness against RST distortions and JPEG compression.

Fidelity

The tradeoff between fidelity and robustness is controlled by adjusting the relative weighting used in the mixing of the watermark signal and the signal extracted from the original image (see Section 9.2.2). As the relative weight assigned to the watermark signal is increased, the strength of the embedded watermark is increased at the expense of lower fidelity. We have chosen the weights empirically, yielding an average signal-to-noise ratio of about 40dB [5]. For simplicity, the same weights are used for different images in our experiments. Fig. 9.9 shows a histogram of the signal-to-noise ratio obtained from watermarking 2000 images. Fig. 9.10 shows an original image, the watermarked version, and their difference.

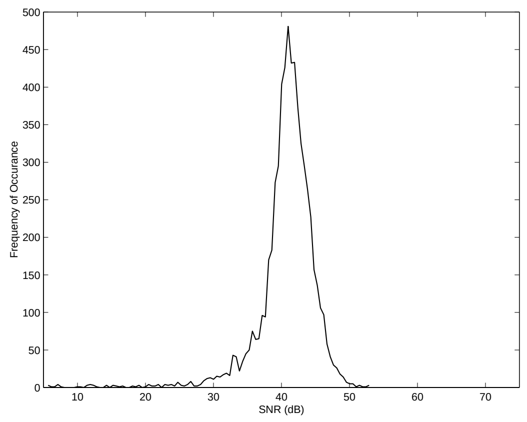


Figure 9.9: SNR histogram of watermarked images

We notice that due to the embedding in a whole image transform domain, the

[5]Here the "signal" is the image, and the "noise" is the watermark.

watermark strength cannot be tuned for each local region. Thus, if the image contains homogeneous texture, the watermark can be well hidden. However, if an image is highly non-homogeneous containing widely varying textures and smooth regions, the mark could become visible in some places unless the weighting is significantly attenuated. Improving image quality in non-homogeneous images requires modifying the algorithm to shape the watermark according to local textures and in the mean time preserving the RST resilience. This has been left for future work.

False Alarm

We performed a study of false alarm probabilities under different thresholds using 10,000 images. A false alarm or false positive occurs when the detector incorrectly concludes that an unwatermarked image contains a given watermark. The probability of false alarm is defined as

$$P_{fp} = P\left\{D_{max} > T\right\} = P\left\{(D_0 > T) \;\; or \;\; (D_1 > T) \;\; or \ldots (D_{89} > T)\right\} \qquad (9.10)$$

where T is the detection threshold, D_{max} is the maximum detection value from all 90 cyclic shifts examined $(D_0, ..., D_{89})$ when running the detector on a randomly selected, unwatermarked image. This probability is first estimated experimentally by applying the detector to 10,000 unwatermarked images from [151], each image being tested by 10 different spread spectrum watermarks. The experimental false alarm probability is plotted in solid lines in Fig. 9.11 with each trace corresponding to one of the 10 watermarks. We also apply a theoretical model in [57] to estimate the false alarm probability, especially for the threshold T greater than 0.55 because we obtain no detection values above 0.55 in the experiment. This theoretical estimate is indicated by dotted lines in Fig. 9.11.

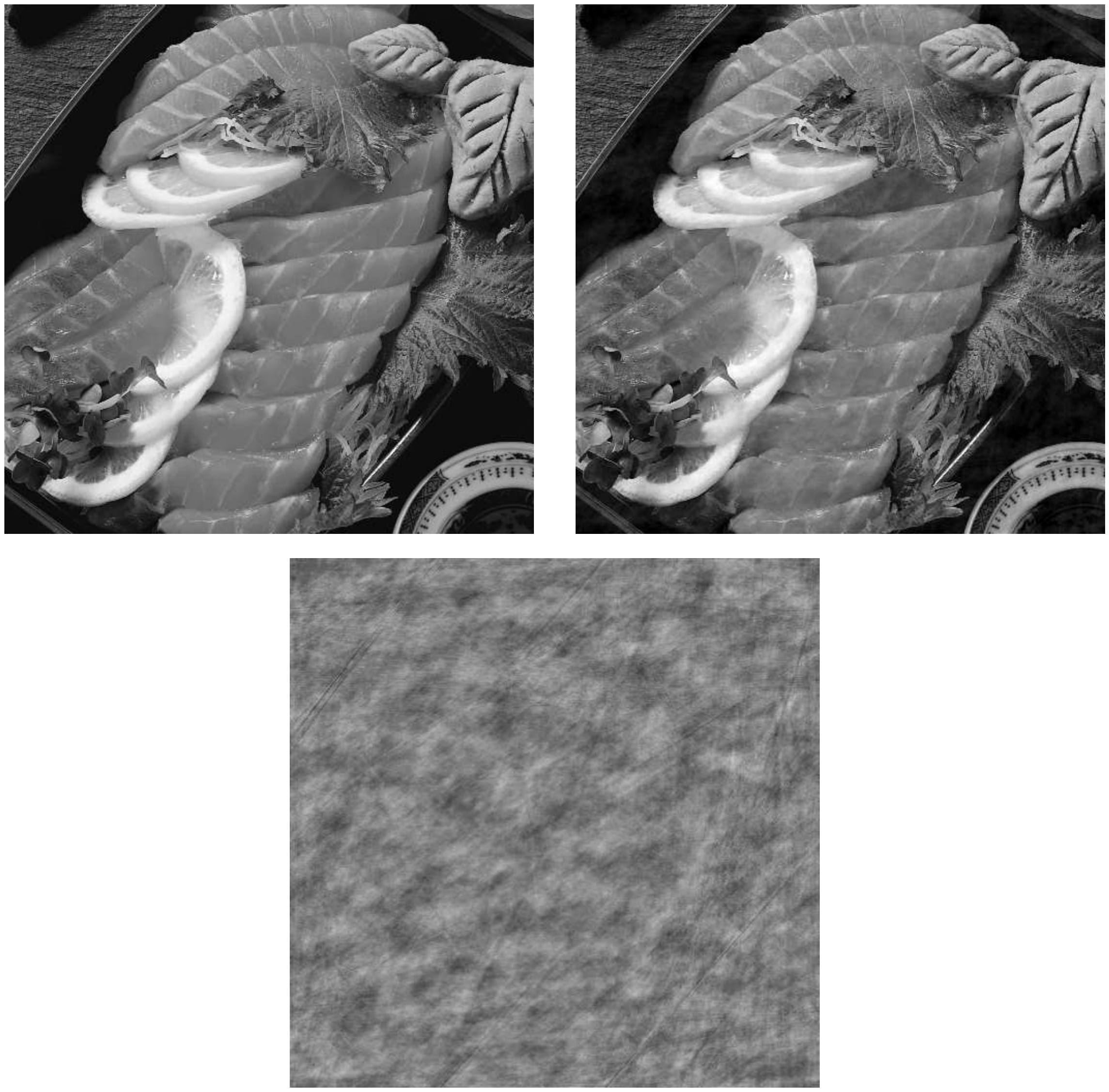


Figure 9.10: Image watermarked by the proposed RST resilient watermarking algorithm: (a) original image, (b) watermarked image, (c) the enlarged difference between original and watermarked, with gray indicating no difference and white/black indicating large difference.

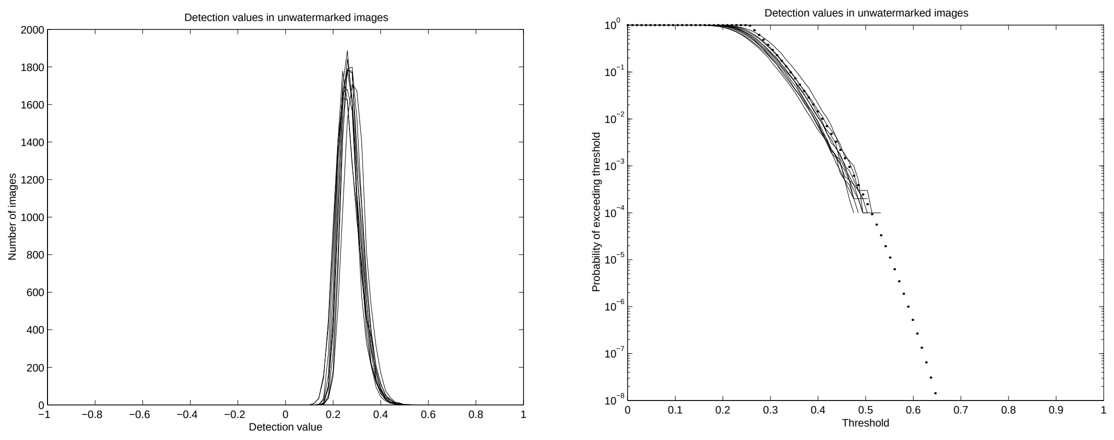


Figure 9.11: $[a|b]$ Detection results for 10 watermarks in 10,000 unwatermarked images: (a) distribution of detection statistics D_{max}; and (b) false alarm probability. Each solid trace corresponds to the result of one of the 10 watermarks, and the dotted line in (b) represents theoretical estimates.

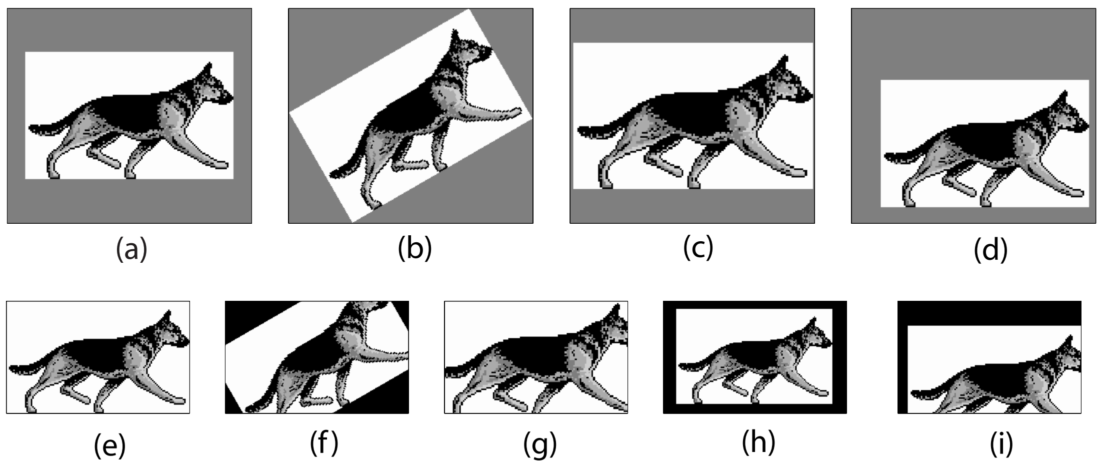


Figure 9.12: Geometric attacks tested in our experiments: (e) and (a) are the original and padded original respectively; (b)-(d) rotation, upscale, and translation without cropping, and (f)-(i) rotation, upscale, downscale, and translation with cropping.

Robustness

Robustness tests against geometric distortions as well as JPEG compression have been performed on 2000 images. Seven geometric distortions illustrated in Fig. 9.12 are examined: rotation with and without cropping (f and b), scaling up with and without cropping (g and c), scaling down (h), and translation with and without cropping (i and d). In order to isolate the RST distortion from cropping, the images have been padded with gray (shown in Fig. 9.12(a)) so that none of the RST testing shown in Fig. 9.12(b)-(d) causes the image data to be cropped. The embedder has been applied to these expanded images and then the padding in the watermarked image is replaced with unwatermarked gray padding prior to the distortion and detection.

For each of the seven geometric attacks, we plot a set of histograms of the detection statistics and ROC curves [6] for several distortion parameters in Fig. 9.13–9.16. The detection performance prior to attack (i.e., no distortion) is shown in dashed lines for comparison. These results demonstrate that when no cropping is involved, the proposed watermarking algorithm exhibits very good resilience to rotation, scale, and translation with only small decrease of detection statistics compared with the case without distortion (see Fig. 9.13(a)(b), Fig. 9.14(a)(b), Fig. 9.15, and Fig. 9.16(a)(b)); when RST distortion is accompanied with cropping, the detection performance degrades: the larger the cropping is, the more negative impact on performance there is (see Fig. 9.13(c)(d), Fig. 9.14(c)(d), and Fig. 9.16(c)(d)). This is expected because our algorithm has not been explicitly designed to withstand cropping. More detailed description of the experiment setup can be found in [169].

We also test the robustness of the proposed algorithm against JPEG compression,

[6] A receiver operating characteristic (ROC) curve describes the probability of correct detection versus the probability of false alarm [2].

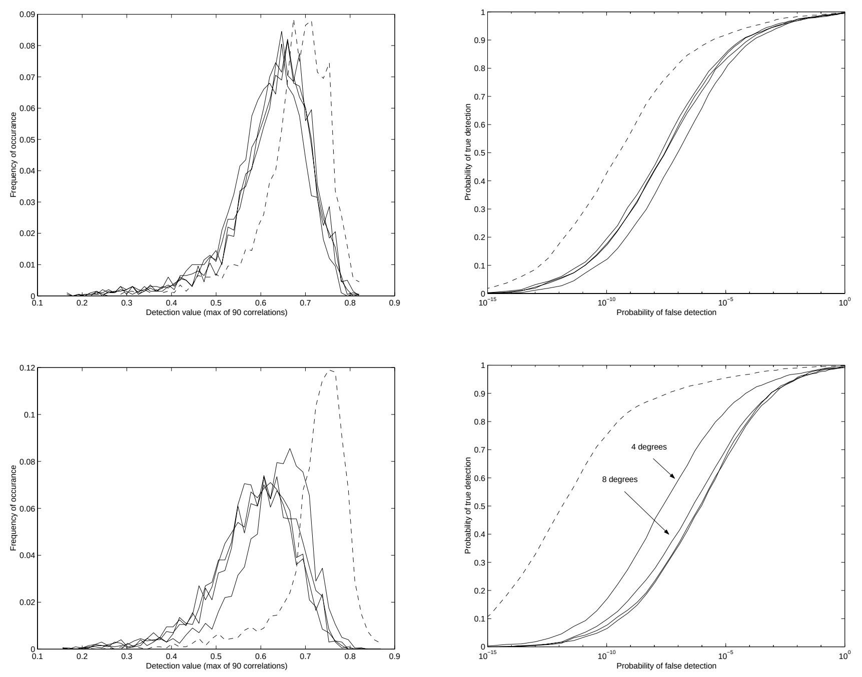


Figure 9.13: $\frac{a|b}{c|d}$ Detection results after rotation 4°, 8°, 30°, and 45°. Showing here are the histogram of detection statistics (a) and detection ROC (b) for rotation without cropping, and the histogram (c) and the ROC (d) for rotation with cropping. Dashed lines represent the detection prior to attack.

as surviving common image processing besides geometric distortions is important in practical applications. We tested JPEG compression at quality factor (QF) of 90, 80, and 70, and the results are shown in Fig. 9.17. We can see that the likelihood of detection decreases with the amount of compression noise introduced and that the amount of this decrease is dependent on the false alarm probability P_{fp}. For relatively high $P_{fp} = 10^{-3}$, the current method is extremely robust to JPEG compression at the qualities tested. At more restrictive false alarm probabilities, for example, $P_{fp} = 10^{-8}$, the moderate JPEG compression at quality factor 70 still yields an acceptable detection probability of 88%.

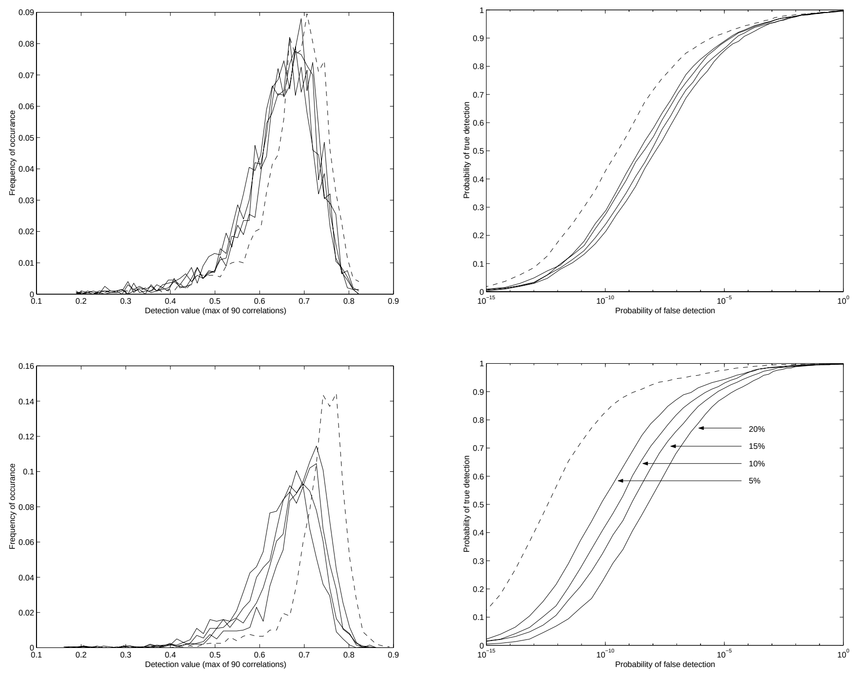


Figure 9.14: $\frac{a|b}{c|d}$ Detection results after upscaling 5%, 10%, 15%, and 20%. Showing here are the histogram of detection statistics (a) and detection ROC (b) for upscaling without cropping, and the histogram (c) and the ROC (d) for upscaling with cropping. Dashed lines represent the detection prior to attack.

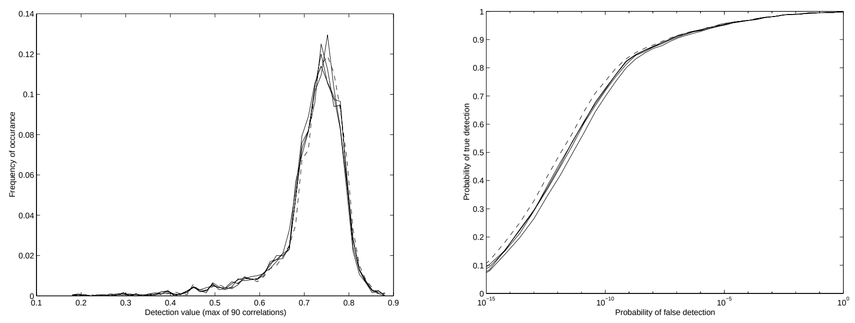


Figure 9.15: $[a|b]$ Detection results after down scaling 5%, 10%, 15%, and 20%. Showing here are the histogram of detection statistics (a) and detection ROC (b). Dashed lines represent the detection prior to attack.

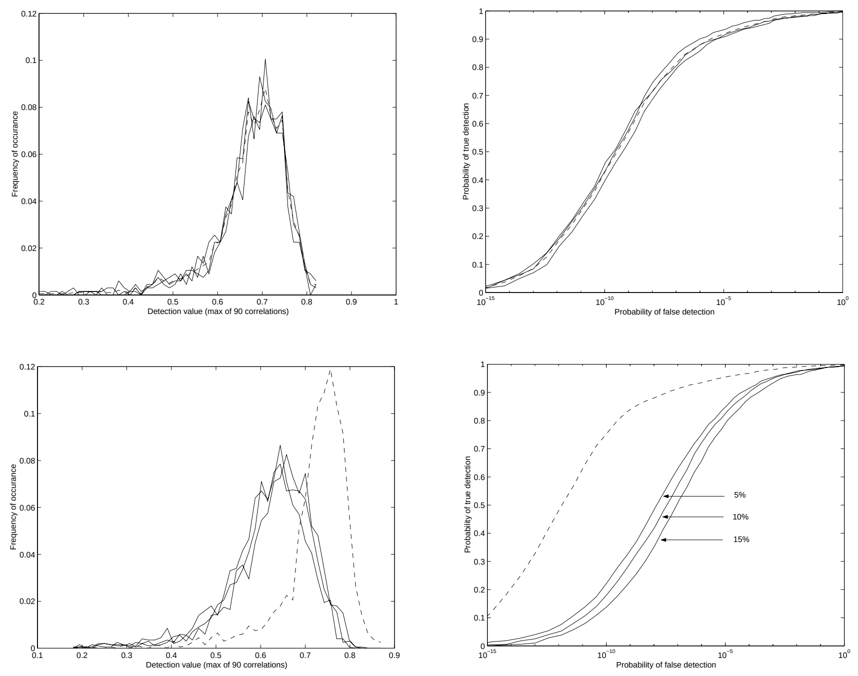


Figure 9.16: $\frac{a|b}{c|d}$ Detection results after translation of 5%, 10%, and 15% of the image size. Showing here are the histogram of detection statistics (a) and detection ROC (b) for translation without cropping, and the histogram (c) and the ROC (d) for translation with cropping. Dashed lines represent the detection prior to attack.

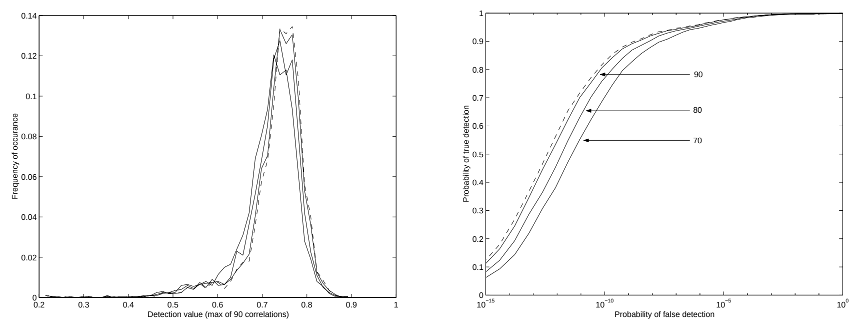


Figure 9.17: $[a|b]$ Detection results after JPEG compression with quality factor 90, 80, and 70. Showing here are the histogram of detection statistics (a) and detection ROC (b). Dashed lines represent the detection prior to attack.

In summary, we proposed a solution to the common robustness problem against rotation, scale, and translation. While the solution is related to earlier proposals in the pattern recognition literature regarding the invariants of the Fourier-Mellin transform, we do not explicitly create a truly RST invariant signal, which has many implementation difficulties [61]. Instead, we create a 1-D signal that changes in a trivial manner under rotation, scale, and translation, and handle a number of important implementation issues. Our experiments have shown that the proposed watermarking algorithm is robust to rotation, scale, and translation. It also survives moderate JPEG compression, although the resilience is lower than non-RST resilient watermarking that embeds a watermark directly in Fourier or DCT coefficients. This is partly because the human visual models in Fourier, DCT, and blockwise transform domain are much better studied than our RST resilient domain, which enables hiding stronger watermark in those domain without introducing artifacts. Quite a few simplifications have been made in our design, and they can be improved in future research.

9.3 Double Capturing Attack on Authentication Watermark

In Chapter 7, we discussed using watermark, usually fragile or semi-fragile against distortion, to detect tampering. These authentication watermarks are inserted either in the pixel domain [78], or in the transform domain [66, 82, 163]. An effective attack on authentication watermark attempts to build an altered or unauthorized image for which the detector will still regard it as an untampered or authorized image. These attacks are quite different from those for robust watermarks. Holliman *et al.*

studied these attacks in [139]. The main idea of the attacks proposed there is to replace each media component (such as a block, a pixel, or a coefficient) with one of the candidates collected from many watermarked images so that the replacement contains a valid watermark indicating authenticity and in the mean time is able to make meaningful semantic changes. These attacks are effective if the watermark is independent of the image content (e.g., the same watermark is embedded in each image) and/or independent of the locations in which it is inserted (e.g., the same watermark is embedded in each block). Carefully designing what data to embed and introducing dependency with cover image are effective countermeasures against this attack, as discussed in Section 7.5. In this section, we shall propose a new attack that does not require the use of multiple marked images.

9.3.1 Proposed Attack

Consider the scenario that an image containing an authentication watermark has been tampered. The modified marked image is then captured either by scanning or by a digital camera, and a new fragile watermark inserted by the watermark module in the capturing device. Since the capturing process can destroy almost all original fragile watermarks, the new image will only bear the new fragile watermark, hence will be regarded as the authentic. The general approach is therefore " *fragile-watermarking* $\rightarrow$ *editing* $\rightarrow$ *fragile-watermarking ...* " (Fig. 9.18), which in some sense is analogous to the multiple ownership claims in robust watermarking [132]. We shall call this type of attacks *double capturing attack.*

Double capturing attack touches a fundamental aspect of image authentication, i.e., the authenticity is always relative with respect to a reference. More specifically, fragile watermarking can only detect alteration after the embedding, but can tell

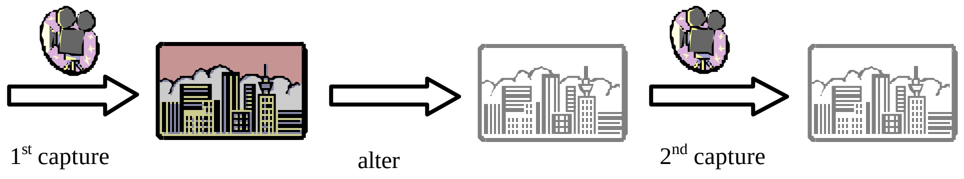


Figure 9.18: Attack on authentication watermarking via double capturing

nothing about the authenticity before embedding.

9.3.2 Countermeasures Against Proposed Attack

For specific application such as the digital camera case, we may use additional features, such as the distance between the lens and the center target object, as part of the embedded data to combat the proposed attack since the focus length when capturing a real scene are different from those when capturing a tampered image from a printed copy or computer monitors. The focus length may be readily obtained from a camera's focusing mechanism. However, with the progress of visualization tools including the development of high quality digital projectors and huge display wall [147], such a solution will eventually have a limitation.

Another countermeasure is to insert both fragile and robust watermarks in an image, as shown in Fig. 9.19. This double watermarking can not only protect both the integrity and ownership of an image [163], but also partially solve the above problem since the double captured image contains two robust watermarks while the single captured image contains only one. In practice, if every watermarking system for authentication purpose (such as those in digital cameras) also embeds one robust watermark which is randomly selected from M orthogonal candidates in each captured image, we can determine whether multiple capturing occurs by checking how

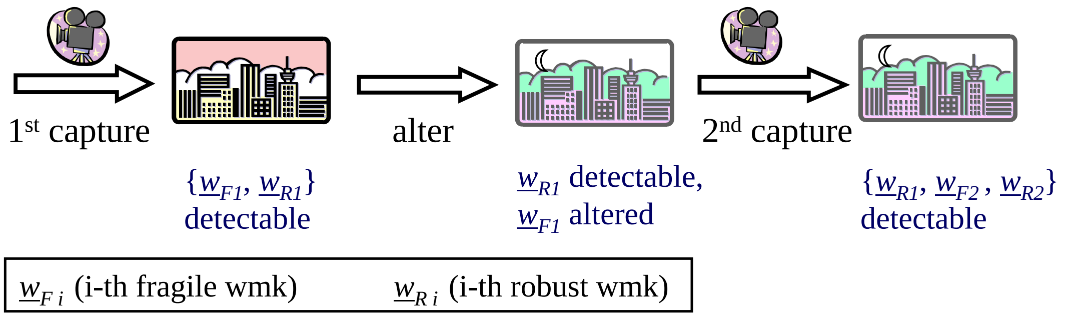


Figure 9.19: Countermeasure against the proposed double capture attack by embedding both robust and authentication (fragile) watermarks.

many robust watermarks are in the test image. The probability of not being able to find multiple robust watermarks after double capturing (i.e., the two robust watermarks inserted by two captures are identical) is $1/M$, i.e., inversely proportional to the number of candidate watermarks, hence it is small for large M. This approach have been implemented as double watermarking in [163] [7], which combines the transform-domain authentication scheme in Chapter 7 with a robust spread spectrum watermarking scheme by embedding the robust watermark first, followed by the authentication watermark. The countermeasure can also be implemented via the multi-level data hiding in Chapter 6.

Acknowledgement

Dr. W. Zeng provided the source code of the error concealment algorithm in [145]. The work of RST watermarking was jointly performed with Drs. I. Cox, J. Bloom, M. Miller, C-Y. Lin, and Y-M. Lui while the author was a summer research assistant at NEC Research Institute and Signafy, Inc..

[7]The original use of double watermarking in [163] is to protect both the ownership and the integrity of an image, but the implementation is directly applicable as a countermeasure against the proposed attack.

Chapter 10

Attacks on Unknown Data Hiding Algorithms

In last section, we discussed watermark attacks with the embedding and detection algorithms known to analysts, which is the case for most attacks studied in literature. The recent public challenge organized by the Secure Digital Music Initiative (SDMI) provided a research opportunity to study attacks under an emulated rivalry environment. We have proposed successful attacks on all four watermarking schemes currently under SDMI's consideration, pointing out the weaknesses and proposing some directions of improvements. We have also found a few general approaches that would be used by an attacker in a real rivalry environment and demonstrated a framework for studying the robustness and security of data hiding systems.

10.1 Introduction

Secure Digital Music Initiative (SDMI) is an international consortium that is developing open technology specifications aiming at protecting the playing, storing, and distributing of digital music [140]. Imperceptible digital watermarking has been proposed to be key elements in the SDMI systems. Upon detection, the watermarks may direct certain actions to be taken, for example, to permit or to deny recording. An SDMI system may incorporate a combination of robust and fragile watermarks. Robust watermarks can survive common signal processing and attacks and are crucial for ensuring the proper functioning of the entire system. The fragile watermarks may be used to indicate whether the audio has experienced certain processing such as lossy compression [141]. The SDMI watermarks are considered as *public watermarks*, meaning that (1) the detection does not use the original unwatermarked copy (i.e., blind detection), and (2) a single or a set of secret keys for detecting the watermarks are usually encapsulated in all publicly available detection devices. In early September 2000, SDMI announced a three-week public challenge for its Phase-II screening, inviting the public to evaluate the attack resistance for four watermark technologies (A, B, C, F) and two other technologies (D, E). In the following, we summarize the attacks and analysis on four watermark technologies.

10.1.1 SDMI Attack Setup

In this challenge, the watermark embedding and detection algorithms are not known to the public. Limited information is available only through the oracle submission. After each submission, the detection is performed by the SDMI staff and the result is sent back with a response time of about $4 \sim 12$ hours. For each of the four challenges, SDMI provided three audio samples, as illustrated in Fig. 10.1. They are:

- samp1?.wav (original audio with no watermark)
- samp2?.wav (samp1?.wav watermarked by Technology-?)
- samp3?.wav (a different audio watermarked by Technology-?)

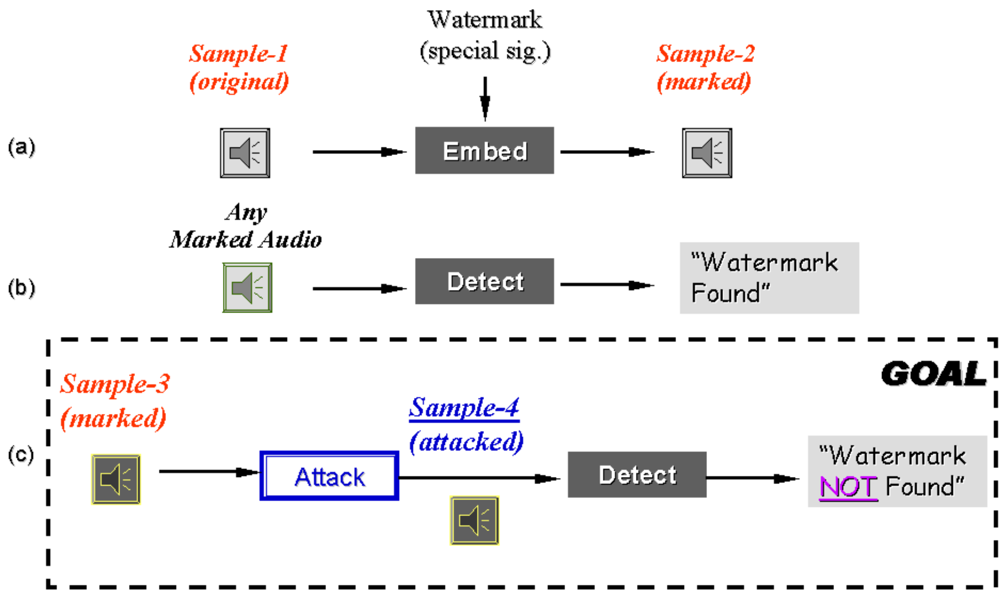


Figure 10.1: Illustration of SDMI attack problem. For each of the four watermark challenges, Samples 1 $\sim$ 3 are provided by SDMI. Sample 4 is generated by participants in the challenge and submitted to SDMI oracle for testing.

where the substitution symbol "?" stands for one of the four challenges: "a", "b", "c", or "f". All audio samples are 2-minute long, sampled at 44.1 kHz with 16-bit precision. The audio contents are mostly popular music. Sample-1 for all four technologies are identical, while sample-3 are all different.

A participant of this challenge generates an attacked audio file *sample-4* from *sample-3*, then uploads it to SDMI's oracle for testing. The detection response is binary, i.e., either "possibly successful" or "unsuccessful". According to SDMI's emails, a "possibly successful" attack must render the detector unable to find the watermark, while retaining the auditory quality comparable to the original one (*sample-3*). This indicates that a successful attack should sits in the region IV of Fig. 10.2. Interestingly, in the unsuccessful case, there is no indication whether the detector can still find watermark (region II of Fig. 10.2) or the detector can no longer find watermark but the auditory quality is considered unsatisfactory (region III of Fig. 10.2). For convenience, we shall denote the four pieces of audio as S_1, S_2, S_3, and S_4.

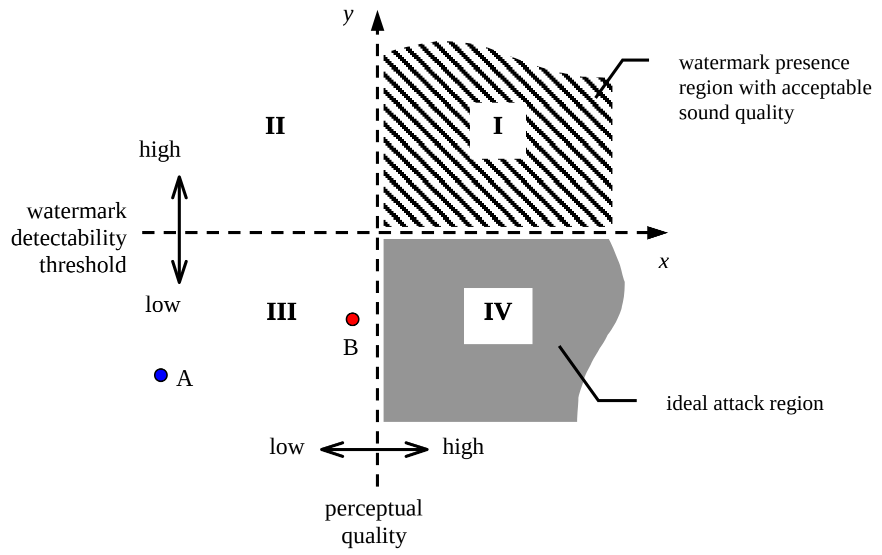

y
x
I
II
III
IV
watermark presence region with acceptable sound quality
high
low
watermark detectability threshold
A
B
ideal attack region
low
high
perceptual quality

Figure 10.2: Illustration of watermark detectability and perceptual quality .

10.1.2 Comments on Attack Setup

The SDMI public challenge presents an emulated rivalry environment, providing attackers with a limited amount of information and restricted access to watermark detectors in a very short time frame. The task is more difficult than the one in real world. First, in real world, a watermark detector encapsulated in a compliant device will be available to an attacker for unlimited uses, and the detector's response time will be instantaneous rather than hours. Second, a user of the real system will be able to distinguish whether or not a detector is able to find watermarks, regardless of the audio quality. These two aspects would enable an attacker polling the detector with different input and obtaining the corresponding output, which in turn provides a large amount of useful information for attacks. Furthermore, the SDMI business model allows a user to pass a piece of non-SDMI music that does not have watermark embedded through an SDMI admission device to make it SDMI-compliant thus has watermark embedded in. This implies that a non-trivial number of original-watermarked audio pairs rather than a single pair are likely to be available to an attacker in real world. As can be seen in the next section, these pairs provide valuable information regarding how watermarks are embedded and the information can be exploited in attacks. One should also note that the perceptual quality imposed on embedding and on attacks are different in reality. The quality criterion for embedding is much higher because part of the commercial value of a piece of audio is determined by the sound quality and in many situations it has to meet the most critical demand among a highly diversified audience (from easy listening by the general public to the professional listening by the experts). On the other hand, the sound quality criterion for attacks only need to satisfy a less demanding audience who are willing to tolerate slightly poor quality if for no-fee listening.

The setup also suggests that the SDMI challenge emphasized on evaluating the effectiveness of robust watermark in each technology and did not take much consideration on the fragile watermark. Referring to SDMI's business model, to enforce a copy control policy that allows no MP3 compression on a piece of music prior to the admission to an SDMI compliant device, the robust watermark embedded in the music would convey to the device this policy while the fragile watermark will be used to detect whether the music experiences compression or not. If the bits in the fragile watermark are designed to be a pre-determined secret pattern and are independent of the host audio, an attacker may obliterate the above policy by restoring a fragile watermark after performing MP3 compression. This attack is likely to introduce less perceptual distortion than removing a robust watermark, therefore, should be given sufficient consideration. The fragile watermarking can be formulated as an authentication problem, for which the attacks and counter-attacks can be studied similar to the material in Chapter 7 and Chapter 9. In the following, we first report our attacks and analysis on the robust watermark in SDMI challenge, then briefly discuss issues related to the fragile watermark.

10.2 Proposed Attacks and Analysis on SDMI Robust Watermarks

In this section, we first explain a general framework for tackling the attack problem. We then take two different successful attacks on Watermark **C** as examples to demonstrate our attack strategies, to describe the specific implementation, and to present analysis in detail. For completeness, the attacks for the other three watermark techniques **A**, **B**, and **F** are also briefly explained.

10.2.1 General Approaches to Attacks

An attacker may take one of three general approaches to tackle the problem: **(Type-1)** exploiting the design weakness via blind attack, **(Type-2)** exploring the embedding mechanism from $\{S_1, S_2\}$, the known original-watermarked pairs, or from the watermarked signal $\{S_3\}$ alone, **(Type-3)** a combination of the two.

Type-1 attacks are said to be *blind* in the sense that they do not rely on any understanding of embedding mechanism or the special properties held by watermarked signals. This approach includes commonly used robustness tests, such as compression, time-domain jittering, pitch change, resampling at different rate, D/A-A/D conversion, and noise addition [142]. The counter-attack strategy for such blind attacks is to find as many weaknesses as possible and to correct them. A good design, therefore, should at least have covered most of the typical robustness tests and their combinations. One of our attacks for Watermark-C and our attack for Watermark-F are blind attacks.

Type-2 attacks are designed using the knowledge about the embedding mechanism. Such knowledge, even if not available at the start, can be obtained by studying the input-output response of the embedding system. For example, if we find the difference between S_1 and S_2 is a small signal around certain frequency, we may design an attack to distort S_3 over the corresponding frequency range. Quite a few of our attacks belong to this category. This type of attack is analogous to the plaintext attack or ciphertext attack in cryptanalysis [1] [11]. The differences are: (1) signal processing analysis replaces the cryptanalytic tools in creating watermark attacks, and (2) the goal of watermark attacks is to render detector unable to detect the watermarks, instead of "cracking codes". The useful signal processing tools include the time-domain and frequency-domain differences, the frequency response, the auto- and cross-correlation, and the cepstrum analysis [12]. We also note that the original and watermarked signals are not easily available simultaneously to the public in some watermarking or data hiding applications, e.g., watermarked-based authentication or DVD video watermarking system. Hence, Type-2 attacks may not be a major concern in those cases. But in SDMI applications where an unwatermarked music

[1] *Plaintext attack* refers to deducing the encryption key or decrypting new cipher texts encrypted with the same key, based on the cipher text of several messages and their corresponding plaintext. *Ciphertext attack* only uses the knowledge of the cipher text of several messages.

may be “admitted” into SDMI domain by embedding a watermark, any successful watermarking design has to take Type-2 attacks into consideration. One possible counter-attack strategy is to intentionally wipe off the otherwise distinct “signature” of a particular embedding. Some obscuring processes may reduce the robustness against blind attacks if the obscuring distorts the embedded watermarks, showing a tradeoff among robustness against various attacks.

Because it is not always possible to find clear clues about embedding from a limited number of original-watermarked pairs, especially when the “wipe-off” is applied, attacks can be designed by combining the above two.

10.2.2 Attacks on Watermark-C

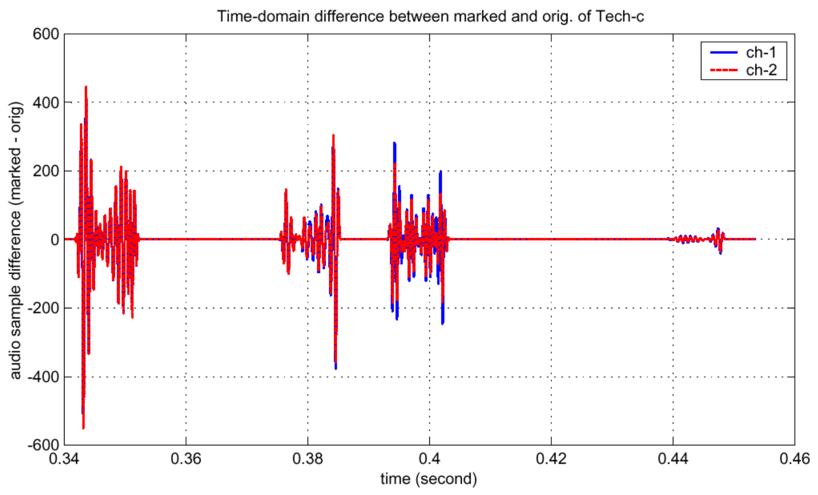
We have proposed two different attacks on Watermark-C. *Attack-C1* explores the weakness of Watermark-C under pitch change. *Attack-C2* is based on observing the difference between original and watermarked signal $\{S_1, S_2\}$. Both attacks were confirmed as successful by SDMI oracle.

Observations from Samples of Watermark-C

By taking the difference of samp1c.wav and samp2c.wav, bursts of narrow band signal are observed, as shown in Fig. 10.3. These bursts appear to be around 1350 Hz.

Attack-C1

Attack-C1 accelerates audio samples by a small amount, which in turn changes the pitch. Blind attacks of 3% pitch increase have been applied to all four watermark proposals, and SDMI detectors indicated that they are effective to Watermark C. The relations between the input and output time index of this speed-up is illustrated



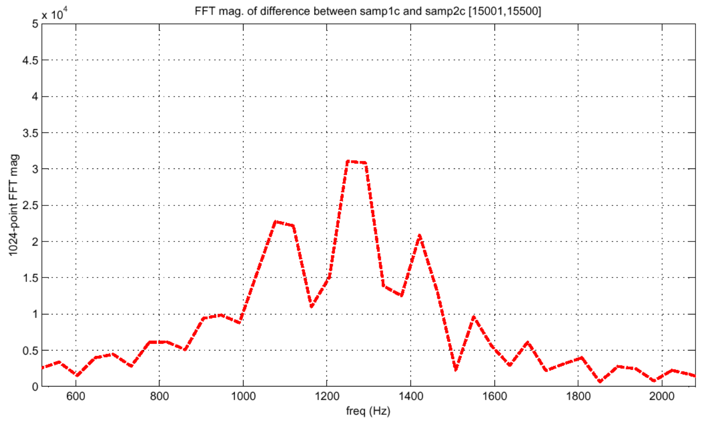


Figure 10.3: Technology-C: (a) the waveform of the difference between sample-1c and sample-2c exhibits tone bursts, and (b) the short-time DFT of one tone burst. The samples observed here occur around 0.34-th second.

in Fig. 10.4, along with several other time-domain jittering/warping that we have encountered during the challenge.

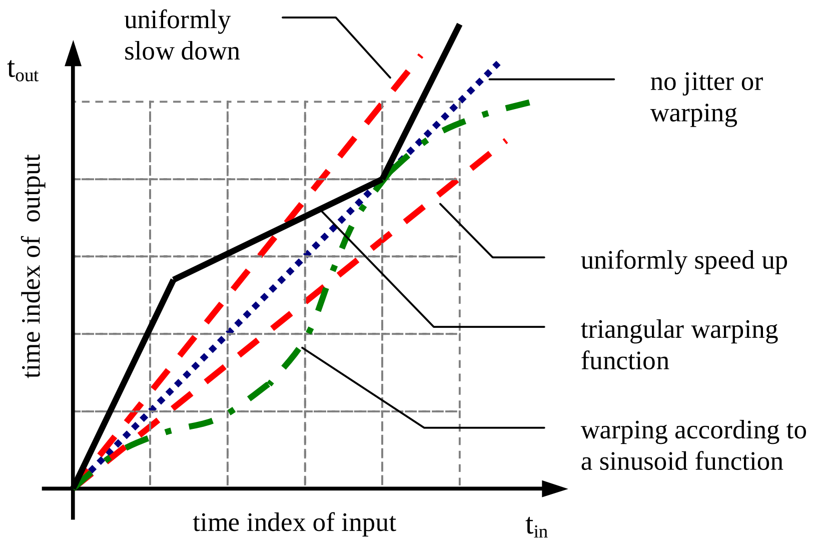


Figure 10.4: Relations between the input and output time index of a few time-domain jittering/warping, including uniform speed-up, uniform slow-down, sinusoid warping, and triangular warping.

One implementation we used is to upsample the audio by M times followed by lowpass filtering and downsampling by N times, giving an overall resampling rate of M/N. The original sampling frequency of F_s is changed to $M/N \cdot F_s$. The resampled audio is then played or stored with the same sampling rate as before, i.e., F_s. The entire process changes the pitch by a fraction of $(N - M)/M$. A precise spectrum interpretation of this can be obtained based on multi-rate signal processing theory [13]. For sampling rate conversion with $M/N > 1$, the spectrum is squeezed along frequency axis by a factor of N/M, leaving the frequency band of $(\frac{N}{M}\pi, \pi]$ with zero; for the case of $0 < M/N < 1$, the frequency band $[0, \frac{M}{N}\pi]$ of the original spectrum is stretched to cover the whole new spectrum, dropping the high frequency band

$(\frac{M}{N}\pi, \pi]$ of the original spectrum. At the end of this rate conversion, the magnitude of the new spectrum is scaled by M/N, the sampling frequency 2π radian per sample corresponds to $\frac{M}{N}F_s$ Hz, and the pitch has not changed. When the signal is played at F_s samples per second, the spectrum with frequency unit of radian per sample is unchanged, but the frequency of 2π radian per sample is now mapped to F_s Hz, effectively changing the pitch by a fraction of $(N - M)/M$. Attack-C1 can also be implemented using commercial audio editing software. For example, the *Effects* $\rightarrow$ *Pitch* menu of *GoldWave v4.19* [148] were used as an alternative way to perform pitch shift attacks (Fig. 10.5).

The ability to detect pitch change varies from individual to individual and depends on whether a reference is available. While most people can discriminate pitch difference as low as 0.6% [102], it is nevertheless rather difficult for a person to identify small pitch changes if he/she has never heard the original before. The standard pitch itself also changed significantly in music history [103, 104]. The pitch of piano's A major, for example, changed steadily from as low as 420Hz in the early 18th century to as high as 457Hz in late 19th century before settling down at the current international standard of 440Hz. Our attack with 3% pitch increase (about a quarter tone) has passed SDMI's strict 2nd round quality testing performed by "golden ears" after the challenge.

As described previously, we observed that the embedding mechanism adds a narrow band signal to the audio at around 1350Hz. Pitch change can be an effective attack because it stretches or squeezes the spectrum, causing misalignment, which in turn reduces the detector response from the popular matched-filter-type detection. One way to enhance the robustness against Attack-C1 is to estimate and undo the stretching, which is likely to be computationally expensive. Another way is to embed

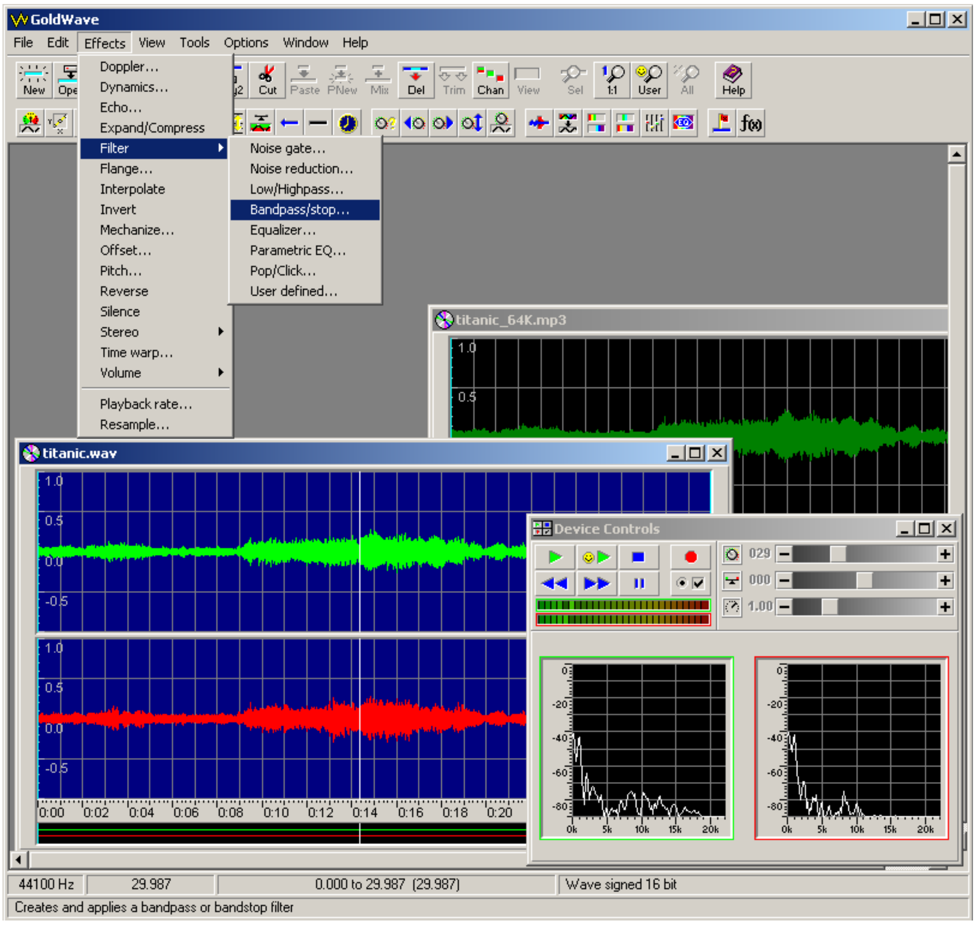


Figure 10.5: Graphics user interface of GoldWave audio editing shareware.

and/or detect watermark in a domain that is resilient to stretching/squeezing.

Attack-C2 Our second attack belongs to Type-2, attempting to jam the frequency band around 1350Hz where it was observed that a narrow band signal had been added by the embedding mechanism. This narrow band watermark signal has some randomness, making jamming difficult. The anti-jamming capability has been seen with the spread spectrum watermark. This commonly used noise-like watermark has good statistical property so that the power of uncorrelated additive noise has to be large enough to effectively jam the watermark [44]. However, to preserve auditory quality, the noise power has to be kept low. Our successful attack is to apply notch filtering to the audio signal at selected frequencies. The filtering introduces significant changes in magnitude and phase around the notch (shown in Fig. 10.6) [12], effectively damaging the embedded watermark. Specifically, we used the *Effects* $\rightarrow$ *Filters* $\rightarrow$ *Bandpass/stop* menu of the audio editor *GoldWave* to perform notch filtering, with a stop band of 1250-1450Hz and steepness of 5 (i.e., 10th order).

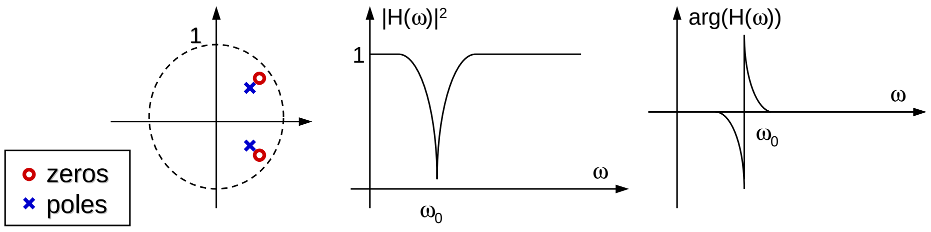


Figure 10.6: A 2nd order notch filter: (from left to right) zero-pole plot and frequency response (magnitude and phase).

Attack-C2 has passed SDMI's 2nd round quality testing performed by "golden ears". For signals with sufficiently rich spectrum, the magnitude and phase changes caused by notch filtering may not be detectable by a person because of frequency masking and other human auditory phenomena. In the next section, we will see that the embedding process of Watermark-B has a step of notch filtering, suggesting that Watermark-B is a potential attack on Watermark-C. It also suggests that the distortion on audio signal imposed by our Attack-C2 is comparable with that by the embedding process of Watermark-B.

10.2.3 Attacks on Watermark A, B & F

Watermark A Our attack on Watermark-A, referred as *copier attack*, is a Type-2 attack. By analyzing the short-time FFT of the samples, we observed regular patterns of phase difference. The observation leads to a time varying model describing the phase difference between sample-1a and sample-2a. Based on the model, our attack "copies" the phase change between sample-1a and sample-2a to sample-3a, aiming at recovering the phase modification done by embedding process. We also introduced some randomness in middle frequency bands during phase manipulation. A variation of this attack incorporating magnitude manipulation was also submitted. Both were confirmed by SDMI oracle as successful.

Watermark B Our attack on Watermark-B is also a Type-2 attack. A spectrum notch is observed around 2800Hz for some parts of the audio and around 3500Hz for some other parts. In addition, the phase difference between original and watermarked audio signals exhibits unique butterfly shape, indicating that notch filtering is involved in embedding. Our attack fills in those notches with random but bounded coefficient values. We also submitted a variation of this attack involving different parameters for notch description. Both were confirmed by SDMI oracle as successful. Interestingly, an embedding technique similar to our observations from Technology-B was found in US Patent $4,876,617$ "Signal Identification" [112] after the challenge. This once again indicates that relying on the secrecy of the embedding algorithm is not a long-term solution to protecting public watermark system.

Watermark F Our attack on Watermark-F explores the weakness of this watermarking approach under time varying warping in time domain, thus is a Type-1 attack. In particular, we warped the time axis by inserting a periodically varying delay.

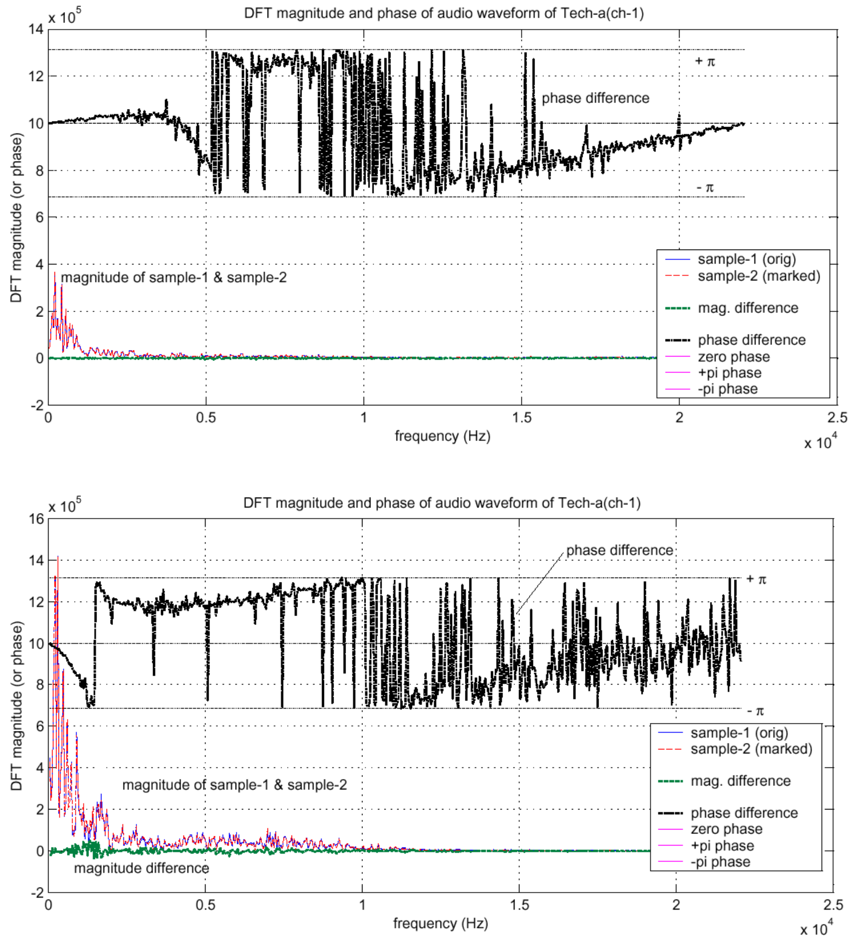


Figure 10.7: Technology-A: FFT magnitude of original and watermarked signals, and phase difference between the two signals for a 1000-sample segment. The two figures are for signals around 3.22-th second and 4.33-th second, respectively.

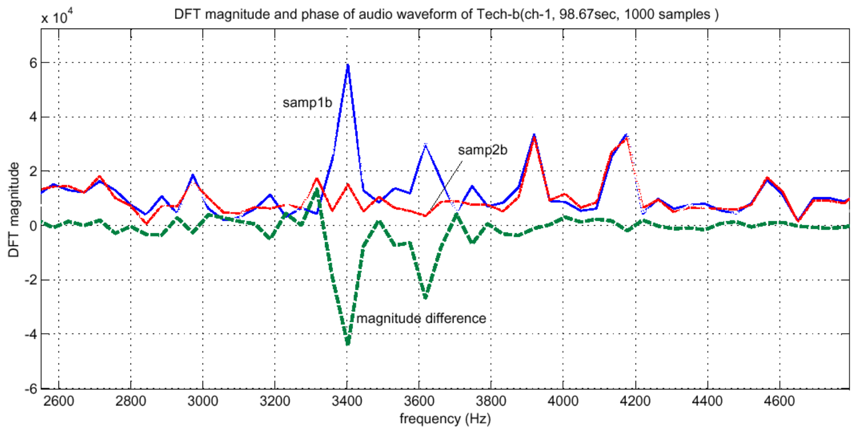


Figure 10.8: Technology-B: FFT magnitudes of sample-1b and sample-2b and their difference for 1000 samples at 98.67 sec.

The delay function comes from our study on Watermark-A, therefore the perceptual quality of our attacked audio is expected to be better than or comparable to that of the audio watermarked by Technology-A. We also submitted variations of this attack involving different warping parameters and different delay function. The warping we performed follows a sinusoid or a triangle function, as illustrated in Fig. 10.4. The attacks were confirmed by SDMI oracle as successful.

Recently, Boeufl and Stern presented their analysis and successful attack on Watermark F in [143]. An autocorrelation analysis was applied to the difference signal between the original and watermarked audio, and a periodicity of 1470 samples ($\frac{1}{30}$ second) was observed. Further study in their report suggested that the difference is a periodic spread spectrum signal with a period of 1470 samples, and the watermark is scaled with different scaling factor for every 147 samples. The scaling factor appears to be a function of the average power of the host audio signal in a local window. The watermark can be detected non-coherently (without using the original audio) by taking a correlation over a long window to suppress the strong interference from the host signal. This detection strategy has been analyzed in Chapter 3 of this thesis. Having found that the same spread spectrum signal is embedded in both sample-1 and sample-3, Boeufl *et al.* designed an algorithm to first search for the initial offset from which the watermark starts to be put into sample-3 and to successfully remove the watermark by subtraction. Their work provides a foundation to explain the effectiveness of our blind attack. Without purposely performing registration/synchronization or without embedding in a resilient domain, spread spectrum embedding is vulnerable against jittering [2].

10.2.4 Summary and Remarks

We presented a general framework for analyzing the robustness and security of audio watermark systems. The framework was demonstrated by our successful attacks in the SDMI public challenge. We pointed out that (1) the weaknesses in the watermarking design are very likely to be explored by an adversary as effective attacks, prompting the need of thorough testing by watermark designers; (2) a large amount of information regarding the embedding mechanism, derived from pairs of original and watermarked signals, can be used to build powerful attacks, prompting the need of obscuring distinct traces between original and watermarked signals. The second point, though not having received much attention in the literature, is crucial for SDMI applications and has a tradeoff with respect to the robustness against other attacks.

Due to various limitations of the challenge including the very short time frame, we adopted practical strategies to increase our chance in finding successful attack(s)

[2]The counterpart of audio (1-D) warping/jittering attacks in image (2-D) is the geometric distortions. We have discussed the attacks and countermeasures for image watermarks under rotation, scale, and translation in Chapter 9.

and in understanding all four watermark technologies. For example, we did not incorporate sophisticated human auditory system (HAS) models that can further improve the perceptual quality. Instead, we focused on finding attacks that render miss detection by a watermark detector without significantly degrading perceptual quality. As illustrated in Fig. 10.2, instead of starting from highly noisy audio around the point A, we look for attacks (such as those around the point B) that is as close to high perceptual quality region as possible and in the mean time as far away from detectability threshold as possible. These are crucial start points from which many optimizations, improvement, and fine-tuning can be feasibly made to proceed to the ideal attack region (region IV in Fig. 10.2).

10.3 Proposed Attacks and Analysis on SDMI Fragile Watermarks

We have mentioned earlier that an SDMI system may use both robust and fragile watermarks. In addition to rendering the robust watermarks undetectable, an adversary may forge a fragile watermark to obliterate the access/copy control mechanism. In the example that a policy does not allow lossy compression on audio files, adversaries may first compress an audio file. The lossy compression, which allows the easy exchange of audio files over network, is likely to destroy the fragile watermark but still retain the robust watermark. Before admitting the audio to an SDMI-compliant device, an adversary decompresses the file and forges a fragile watermark. Examining the existence and the content of the robust and fragile watermarks in an audio file, a device draws a false conclusion that the audio has not been compressed and that the user has not violated the access control policy.

More abstractly, the fragile watermark in an SDMI system serves the purpose of tampering detection, which is a major application of fragile watermarks. Issues regarding the designs, the attacks, and the countermeasures of watermark-based authentication have been discussed in Chapter 7, where the basic idea is to keep a reference and to compare with it later. It is desirable to keep the data volume of the reference small so that the overhead in storage or transmission of the entire data is small. The location of the reference does not have to be secret, but the reference must (1) be unambiguous in the sense that two sets of meaningful data are unlikely to have the same signature, and (2) be difficult to tamper without trace. For perceptual source like digital audio, the reference can be combined with the perceptual source in a more seamless way via watermarking. For example, one can embed a prescribed data pattern or some features of the host audio signal into the audio, and later when the authenticity of an audio is in question, one can verify the integrity of these embedded data to decide on the authenticity of the audio signal. The watermark-based authentication relies on either (1) the embedded data, or (2) the fragility and secrecy of the embedding mechanism, or (3) both. Compared with non-embedding approaches that only make use of the first element (e.g., attaching a cryptographic digital signature to the audio), the watermark-based approach may be able to offer additional security if designed properly. A poorly designed watermark algorithm, however, may leave holes for adversaries to forge a valid authentication watermark.

One potential flaw regarding fragile watermark in SDMI-like application is to rely too much on the secrecy of embedding mechanism. In [175], Technology A is taken as an example to demonstrate how the embedding mechanism of a fragile watermark can be explored. Weak echoes have been observed in high frequency bands around 8-16K Hz. The polarity and delay of echoes vary about every 1/50 second, and they are very likely to be used to encode some authentication information. The data embedded in such high frequency bands are likely to be distorted by lossy compression (such as MP3) and low-pass filtering. If the authentication data (i.e., the data to be embedded) is not wisely chosen, an adversary can explore the inner workings of embedding mechanism and use this knowledge to recover the authentication data after performing unallowed processing/distortion on the audio signal. A trivial choice, for example, to embed the same pattern fragilely for different audio files, could leave holes for adversaries who may repair the authentication data by using the knowledge of the embedding mechanism. Holliman *et al.* discussed a few cases of counterfeiting watermarks in images [139] and pointed out the weaknesses of embedding data that are independent of the host media. If the fragile watermark in an SDMI system were designed to be independent of the host media, it would be vulnerable to forgery attack, implying the perceptual quality of attacked signal could be very good. This is because an attacker does not need to destroy the robust watermark (which could introduce some perceptual distortion, depending on the design and the attack); what he/she needs to do is just to recover the fragile watermark that generally has lower energy and is perceptually transparent.

A countermeasure against forging fragile watermark is to introduce dependency, which has been discussed in Section 9.3. That is, we embed some data, or called "features", that are derived from the host audio signal. Denoting the features derived from an audio signal S_1 as $d_1 = f(S_1)$, and those derived from an altered signal S_2 as $d_2 = f(S_2)$ (e.g., S_2 could be an MP3 compressed version of S_1), we would like to choose a function $f(\cdot)$ such that d_1 and d_2 are sufficiently different. Encryption and/or cryptographic digest may be used in designing $f(\cdot)$ and the keys associated with $f(\cdot)$ should be kept secret. Readers may notice a potential problem that the features derived from an audio signal could be different from those derived from its watermarked version, i.e., $f(S_1) \neq f(E(S_1, d_1))$ where $E(\cdot, \cdot)$ is an embedding function. This problem can be easily fixed by embedding the data derived from the i^{th} segment of an watermarked audio in the $(i+1)^{th}$ segment of the unwatermarked audio to obtain $(i+1)^{th}$ watermarked segment, and so on so forth.

In summary, the fragile watermarks in SDMI-like system should be carefully designed to eliminate weaknesses against counterfeiting attacks and other security holes.

Acknowledgement

The works on attacking four SDMI robust watermark technologies were jointly done with Scott Craver at Princeton University. In particular, the sinusoid warping attack on Technology F was proposed by Craver.

Chapter 11

Conclusions and Perspectives

This thesis presents multiple aspects of data hiding with both analytic study and experimental results. We have shown that multimedia data hiding can be used for various applications, including ownership protection, alteration detection, access/copy control, annotation, and conveying other side information. In addition to the design issues, we discussed attacks on watermarking algorithms with a goal of identifying weaknesses and limitations of existing design/framework as well as proposing improvements.

While we have discussed many advantages of data hiding and enumerated a number of possible applications, it is necessary in practice to justify case by case the need of data hiding versus the alternatives such as putting side information in the user data field. We feel it important to understand that in spite of the interesting intellectual challenge and the current popularity of data hiding in the research community, engineering practice would always favor simplicity, efficiency, and effectiveness than simply following a new fashion. On the other hand, currently identified challenges, weaknesses, as well as limitations are not yet sufficient in drawing conclusion on the usefulness of digital watermarking and multimedia data hiding. The field is still young and involves various disciplines such as signal processing, computer security, psychology and economics/business. Paradigms and underlying theories are either just being set or to be set. Therefore, objective and multi-disciplinary approaches would continue to be necessities for studying various aspects of multimedia data hiding.

Despite of the differences, data hiding (stegnography) and cryptography are tightly connected. Many ideas of cryptography have found to be very useful in such existing data hiding works as tampering detection. As for the future research, it would be fruitful to undertake a more general investigation regarding what new value can be offered by combining stegnography and cryptography, and how to make use of this combination to complement the weaknesses or limitations of each one individually. This study would lead to a basis for designing practical media security systems, and to solutions of the *Digital Rights Management* (DRM) for digital multimedia data.

In addition, regarding the gap between the highly simplified channel models and the real-world scenarios in today's data hiding research, a rigorous analysis of the capacity versus robustness of data hiding in a realistic setting and incorporating perceptual models (rather than using a simplified assumption such as Gaussian distribution) is worthwhile to pursue. Without neglecting the intellectual contribution by capacity study toward the understanding of data hiding, it is expected that any fundamental study regarding embedding capacity could ultimately throw light to designing or improving practical data hiding systems for a variety of applications.

Besides the classic use in ownership protection and copy/access control, we have demonstrated that data hiding can be a useful tool to send side information in video communication. This direction is rather new and can be further explored for applications other than those discussed in this thesis. Along with this pursuit, research need to be performed toward the integration of error resilience, transcoding, network condition measurement, dynamic resource allocation, and admission control in a multimedia communication system, aiming at studying the relations and the interplay of various modules that were generally addressed individually. This interdisciplinary study with theories and practice in network communication, signal processing, neural network, optimization and control systems can lead to better understanding and deployment of multimedia communication.

Bibliography

[**General Ref. on Probability, Statistics, Algorithm, & Communication**]

[1] W. Feller: *An Introduction to Probability Theory and Its Applications*, vol.1, 3rd Ed. revised print, John Wiley & Sons, 1970.

[2] H.V. Poor: *Introduction to Detection and Estimation*, 2nd Ed., Springer-Verlag, 1994.

[3] V.F.Kolchin, B.A.Sevastyanov, V.P. Chistyakov: *Random Allocation*, V. H.Winston & Sons, 1978.

[4] R. Sedgewick: *Algorithms in C*, Addison-Wesley, 1990.

[5] J.G. Proakis: *Digital Communications*, 3rd Ed., McGraw-Hill, 1995.

[6] S. Verdu: *Multiuser Detection*, Cambridge University Press, 1998.

[7] T.M. Cover, J.A. Thomas: *Elements of Information Theory*, 2nd Ed., John-Wiley & Sons, 1991.

[8] R. E. Blahut: *Theory and Practice of Data Transmission Codes*, 2nd Edition (draft), 1997.

[9] A.S. Tanenbaum: *Computer Networks*, 3rd Edition, Prentice Hall, 1996.

[10] J. Walrand: *Communication Networks: A First Course*, 2nd Edition, McGraw-Hill, 1998.

[11] B. Schneier: *Applied Cryptography: protocol, algorithms, and source code in C*, 2nd Ed., John Wiley & Sons, 1996.

[**General Ref. on Multimedia Signal Processing & Coding**]

[12] S.J. Orfanidis: *Introduction to Signal Processing*, Prentice Hall, 1996.

[13] J.S. Lim, A.V. Oppenheim (Eds.): *Advanced Topics in Signal Processing*, Prentice Hall, 1988.

[14] R.N. Bracewell: *The Fourier Transform and Its Applications*, McGraw-Hill, 1986.

[15] N.S. Jayant, P. Noll: *Digital Coding of Waveforms*, Prentice-Hall, 1984.

[16] A.K. Jain: *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[17] K.R. Castleman: *Digital Image Processing*, Prentice Hall, 1996.

[18] G. K. Wallace: "The JPEG Still Picture Compression Standard",*IEEE Trans. on Consumer Electronics*, vol.38, no.1, pp18-34, 1992.

[19] J.L. Mitchell, W.B. Pennebaker, C.E. Fogg (Eds): *MPEG Video: Compression Standard*, Digital Multimedia Standards Series, Chapman & Hall, 1996.

[20] JBIG Committee: *ISO Final Committee Draft (FCD) 14492 for JBIG2 Standard*, July 1999.

[21] Joint Photographic Experts Group (JPEG), http:// www.jpeg.org .

[22] Moving Picture Experts Group (MPEG), http:// www.cselt.it/mpeg/ .

[23] MPEG Points and Resources, http:// www.mpeg.org .

[24] Mathworks, Inc: Documentation of Matlab 5.3, http:// www.mathworks.com .

[**Copyright & Legal Issues**]

[25] U.S. Copyright Office: "The Digital Millennium Copyright Act of 1998" (DMCA), Summary and Public Law, 1998.

[**Tutorial, Survey, & Special Issue on Data Hiding & Security**]

[26] F. Mintzer, G. W. Braudaway, M. M. Yeung: “Effective and Ineffective Digital Watermarks”, *ICIP*, 1997.

[27] M.D. Swanson, M. Kobayashi, A.H. Tewfik: “Multimedia data-embedding and watermarking technologies”, *Proceedings of IEEE*, vol.86, pp.1064-1087, June, 1998.

[28] R.J. Anderson, F.A.P. Petitcolas: “Information Hiding: An Annotated Bibliography”, http://www.cl.cam.ac.uk/ ∼fapp2/ steganography/ bibliography/, August 1999.

[29] F.A.P. Petitcolas, R.J. Anderson, M.G. Kuhn: “Information Hiding - A Survey”, *Proc. of IEEE*, pp.1062-1078, July, 1999.

[30] F. Hartung, M. Kutter: “Multimedia Watermarking Techniques”, *Proc. of IEEE*, pp.1079-1107, July, 1999.

[31] Special Issue on Watermarking, *Signal Processing*, vol.66, no.3, Elsevier Science, May, 1998.

[32] Special Issue on Watermarking, *Communications of the ACM*, vol.41, no.7, July, 1998.

[33] Copyright and Privacy Protection, Special Issue, *IEEE Journal on Selected Areas in Communication (JSAC)*, v.16, n.4, May 1998.

[34] Special Issue on Identification and Protection of Multimedia Information, *Proceedings of IEEE*, July, 1999.

[35] E.T. Lin, E.J. Delp: “A Review of Fragile Image Watermarks”, *Proc. of the Multimedia and Security Workshop (ACM Multimedia ’99)*, pp25-29, 1999.

[**Thesis on Data Hiding & Watermarking**]

[36] W. Zeng: *Resilient Video Transmission and Multimedia Database Application*, Ph.D. Thesis, Princeton University, 1997.

[37] L. Qiao: *Multimedia Security and Copyright Protection*, Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1998.

[38] D. Kundur: *Multiresolution Digital Watermarking: Algorithms and Implications for Multimedia Signals*, Ph.D. Thesis, University of Toronto, 1999.

[39] M. Ramkumar: *Data Hiding in Multimedia – Theory and Applications*, Ph.D. Thesis, 2000.

[40] B. Chen: *Design and Analysis of Digital Watermarking, Information Embedding, and Data Hiding Systems*, Ph.D. Thesis, MIT, 2000.

[41] C-Y. Lin: *Watermarking and Digital Signature Tecniques for Multimedia Authentication and Copyright Protection*, Ph.D. Thesis, Columbia University, 2000.

[42] J. Song: *Optimal Rate Allocation and Security Schemes for Image and Video Transmission over Wireless Channels*, Ph.D. Thesis, University of Maryland, College Park, 2000.

[**Spread Spectrum Watermarking**]

[43] W. Bender, D. Gruhl, N. Morimote, "Techniques for Data Hiding", *Proc. of SPIE*, vol.2420, pp40, 1995.

[44] I. Cox, J. Kilian, T. Leighton, T. Shamoon: "Secure Spread Spectrum Watermarking for Multimedia", *IEEE Transaction on Image Processing*, vol.6, no.12, pp.1673-1687, 1997.

[45] I.J. Cox: "Spread Spectrum Watermark for Embedded Signaling", Patent US 5,848,155, 1998.

[46] C. Podilchuk, W. Zeng: "Image Adaptive Watermarking Using Visual Models", *IEEE Journal Selected Areas of Communications (JSAC)*, vol.16, no.4, May, 1998.

[47] C. Podilchuk, W. Zeng: "Perceptual Watermarking of Still Images", *IEEE First Workshop of Multimedia Signal Processing*, 1997.

[48] M. D. Swanson, B. Zhu, A. H. Tewfik, "Transparent Robust Image Watermarking", *ICIP*, 1996.

[49] J.J.K. ÓRuanaidh, W.J. Dowling, F.M. Boland, "Phase Watermarking of Digital Images", *ICIP*, vol.3, pp 239-241, 1996.

[50] B. Tao, B. Dickinson: "Adaptive Watermarking in the DCT Domain", *ICASSP*, 1997.

[51] J. Liang, P. Xu, T.D. Tran, "A universal robust low frequency watermarking scheme," submitted to *IEEE Trans. on Image Processing*, May 2000.

[52] W. Zeng, B. Liu: "On Resolving Rightful Ownerships of Digital Images by Invisible Watermarks", *ICIP*, 1997.

[53] W. Zeng, B. Liu, "A statistical watermark detection technique without using original images for resolving rightful ownerships of digital images," *IEEE Trans. Image Processing*, vol. 8, no. 11, pp. 1534-1548, Nov. 1999.

[54] I.J. Cox, M.L. Miller, A. McKellips: "Watermarking as Communications With Side Information", *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1127-1141, 1999.

[55] M.L. Miller, J.A. Bloom, I.J. Cox: "Exploiting Detector and Image Information in Watermark Embedding", *IEEE Int. Conf. on Image Processing*, Sept. 2000.

[56] G. Depovere, T. Kalker, J-P. Linnartz: "Improved Watermark Detection Using Filtering Before Correlation", *IEEE Int. Conf. on Image Processing*, vol. 1, pp. 430-434, Chicago, Oct. 1998.

[57] M.L. Miller, J.A. Bloom: "Computing the probablity of false watermark detection", *Proceedings of the Third International Workshop on Information Hiding*, 1999.

[58] S. Pereira, T. Pun: "Fast Robust Template Matching for Affine Resistant Image Watermarks", *Proc. of the 3rd Information Hiding Workshop (IHW)*, Lecture Notes in Computer Science, Springer-Verlag, pp207-218, 1999.

[59] G. Csurka, F. Deguillaume, J.J.K. ÓRuanaidh, T. Pun: "A Bayesian approach to Affine Transformation Resistant Image and Video Watermarking", *Proc. of the 3rd Information Hiding Workshop (IHW)*, Lecture Notes in Computer Science, pp315-330, Springer-Verlag, 1999.

[60] N.F. Johnson, Z. Duric, S. Jajodia: "Recovery of Watermarks from Distorted Images", *Proc. of the 3rd Int. Information Hiding Workshop*, pp.361-375, 1999.

[61] J.J.K. ÓRuanaidh, T. Pun, "Rotation, Translation and Scale Invariant Spread Spectrum Digital Image Watermarking", *Signal Processing*, vol.66, no.3, 1998.

[62] A. Herrigel, J. Oruanaidh, H. Petersen, S. Pereira, T. Pun: "Secure Copyright Protection Techniques for Digital Images", *Second Information Hiding Workshop (IHW)*, Lecture Notes in Computer Science, Springer-Verlag, vol. 1525, 1998.

[**Robust Data Hiding Using Type-II Relational Embedding**]

[63] E. Koch, J. Zhao: "Towards Robust and Hidden Image Copyright Labeling", *IEEE Workshop on Nonlinear Signal and Image Processing*, 1995.

[64] C-T. Hsu, J-L. Wu: "Hidden Signatures in Image", *ICIP*, vol. 3, 1996.

[65] M. Ramkumar, A.N. Akansu: "A Robust Scheme for Oblivious Detection of Watermarks / Data Hiding in Still Images", *Symposium on Voice, Video, and Data Communication*, SPIE, 1998.

[66] M.D. Swanson, B. Zhu, A.H. Tewfik: "Robust Data Hiding for Images", *IEEE DSP Workshop*, 1996.

[67] M.D. Swanson, B. Zhu, A.H. Tewfik: "Data Hiding for Video-in-Video", *ICIP*, 1997.

[68] M. Alghoniemy, A.H. Tewfik: "Self-synchronizing Watermarking Techniques", *Symposium on Content Security and Data Hiding in Digital Media*, NJ Center for Multimedia Research and IEEE, 1999.

[69] D. Mukherjee, J-J. Chae, S.K. Mitra, B.S.Manjunath: "A Source and Channel Coding Framework for Vector-Based Data Hiding in Video", *IEEE Trans. on Circuits and Systems for Video Technology*, v.10, n.4, pp630-645, June, 2000.

[70] B. Chen, G.W. Wornell: "Digital Watermarking and Information Embedding Using Dither Modulation", *IEEE Workshop on Multimedia Signal Processing*, 1998.

[71] B. Chen, G.W. Wornell: "Dither Modulation; A New Approach to Digital Watermarking and Information Embedding", *Proc. of SPIE, Security and Watermarking of Multimedia Contents*, vol. 3657, Jan., 1999.

[72] B. Chen, G.W. Wornell: "Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding", to appear in *IEEE Trans. on Info. Theory*, Sept., 2000.

[**Fragile/Semi-Fragile Watermarks and Multimedia Authentication**]

[73] F.A.P. Petitcolas, Examples of Least-significant-bit Embedding, http://www.cl.cam.ac.uk/ ~fapp2/steganography/ image_ downgrading/, 1998.

[74] G. L. Friedman: "The Trustworthy Digital Camera: Restoring Credibility to the Photographic Image", *IEEE Trans. on Consumer Electronics*, Nov. 1993.

[75] M. Schneider, S-F. Chang: "A Robust Content Based Dig. Signature for Image Authentication", *ICIP*, 1996.

[76] C.Y. Lin, S.F. Chang: "A Robust Image Authentication Method Surviving JPEG Lossy Compression", *SPIE Storage and Retrieval of Image/Video Databases*, Jan. 1998.

[77] S. Walton: " Image Authentication for a Slippery New Age", *Dr. Dobb's Journal*, pp18-26, April, 1995.

[78] M. M. Yeung, F. Mintzer: "An Invisible Watermarking Technique for Image Verification", *ICIP*, 1997.

[79] D. Storck: "A New Approach to Integrity of Digital Images", *IFIP Conf. on Mobile Communication*, 1996.

[80] P. W. Wong: "A Watermark for Image Integrity and Ownership Verification", *IS&T PIC Conf. Proc.*, 1998.

[81] Epson America, Inc.: http://www.epson.com/ cam_scan/cam_extras/ ias/, Image Authentication System, 1999.

[82] D. Kundur, D. Hatzinakos: "Towards a Telltale Watermarking Technique for Tamper-Proofing", *ICIP*, 1998.

[83] F. Fridrich: "Image Watermarking for Tamper Detection", *ICIP*, 1998.

[84] L. Xie, G. R. Arce: "Joint Wavelet Compression and Authentication Watermarking", *ICIP*, 1998.

[85] C-Y. Lin and S-F. Chang: "Semi-Fragile Watermarking for Authenticating JPEG Visual Content", *SPIE International Conf. on Security and Watermarking of Multimedia Contents II (EI'00)*, vol. 3971, 2000.

[86] F. Mintzer, G. Braudaway: "If one watermark is good, are more better?," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Phoenix, Arizona, March 1999.

[87] C-S. Lu, H-Y.M. Liao: "Multipurpose Watermarking for Image Authentication and Protection", Technical Report, Institute of Information Science, Academia Sinica, Taiwan, 2000.

[88] J. Fridrich, M. Goljan: "Protection of Digital Images using Self Embedding", *Symposium on Content Security and Data Hiding in Digital Media*, Newark, NJ, May 1999.

[**Data Hiding Capacity**]

[89] J.R. Smith, B.O. Comiskey: "Modulation and Information Hiding in Images", *1st Information Hiding Workshop*, 1996.

[90] S. D. Servetto, C. I. Podilchuk, K. Ramchandran: "Capacity Issues in Digital Image Watermarking", *ICIP*, 1998.

[91] M. Ramkumar, A.N. Akansu: "Information Theoretic Bounds for Data Hiding in Compressed Images", *IEEE 2nd Multimedia Signal Processing Workshop*, 1998.

[92] L.M. Marvel, C.G. Boncelet: "Capacity of the Steganographic Channel", submitted to *IEEE Trans. on Signal Processing*, 1999.

[93] P. Moulin, J.A. O'Sullivan: "Information-Theoretic Analysis of Information Hiding", preprint, Sept. 1999.

[94] P. Moulin and J.A. O'Sullivan: "Information-Theoretic Analysis of Watermarking", *ICASSP'00*, 2000.

[95] P. Moulin, M.K. Mihcak, G-I. Lin: "An Information-theoretic Model for Image Watermarking and Data Hiding", *ICIP'00*, 2000.

[96] J. Chou, S.S. Pradhan, L.E. Ghaoui, K. Ramchandran: "Watermarking Based on Duality With Distributed Source Coding and Robust Optimization Principles", *ICIP'00*, 2000.

[97] M.H.M. Costa: "Writing on Dirty Paper", *IEEE Trans. on Info. Theory*, vol. IT-29, no. 3, May 1983.

[98] C.E. Shannon: "Channels With Side Information at the Transmitter", *IBM Journal of Research and Development*, pp. 289-293, 1958.

[99] C.E. Shannon: "The Zero-error Capacity of a Noisy Channel", *IRE Trans. on Info. Theory*, IT-2, pp8-19, 1956.

[**Perceptual Models**]

[100] H.A. Peterson, A.J. Ahumada, A.B. Watson: "Improved Detection Model for DCT Coefficient Quantization", *Proc. SPIE Conf. Human Vision, Visual Processing, and Digital Display IV*, vol. 1913, pp.191-201, Feb. 1993.

[101] A.B. Watson: "DCT Quantization Matrices Visually Optimized for Individual Images", *Proc. SPIE Conf. Human Vision, Visual Processing, and Digital Display IV*, vol. 1913, pp.202-216, Feb. 1993.

[102] Doug Coulter: *Digital Audio Processing*, R&D Books, 2000.

[103] E.E. Swenson: "The History of Musical Pitch in Tuning the Pianoforte", http://www. mozartpiano.com/ pitch.html, 2000.

[104] Association of Blind Piano Tuners: "History of Pitch", http://www. uk-piano.org/ history/pitch.html, 2000.

[105] P.R. Cook (eds.): *Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics*, The MIT Press, 1999.

[**Specialized Embedding: Binary Images**]

[106] K. Matsui, K. Tanaka: "Video-steganography: How to Secretly Embed a Signature in a Picture", *Proc. of IMA Intellectual Property Project*, vol. 1, no. 1, 1994.

[107] E. Koch, J. Zhao: "Embedding Robust Labels Into Images for Copyright Protection", *Proceedings of the International Congress on Intellectual Property Rights for Specialized Information, Knowledge & New Technologies*, 1995.

[108] N.F. Maxemchuk, S. Low: "Marking Text Documents", *ICIP*, 1997.

[109] Y. Liu, J. Mant, E. Wong, S.H. Low: "Marking and Detection of Text Documents Using Transform-domain Techniques", *Proceedings of SPIE*, vol. 3657, Electronic Imaging (EI'99) Conference on Security and Watermarking of Multimedia Contents, San Jose, CA, 1999.

[110] A.K. Bhattacharjya, H. Ancin: "Data Embedding in Text For a Copier System", *ICIP*, 1999.

[111] A. Finkelstein: personal communication, 1998.

[**Specialized Embedding: Audio**]

[112] S.J. Best, R.A. Willard: "Signal Identification", US Patent 4,876,617, Thorn EMI Plc, October 1989.

[113] S.J. Best, N. Johnson, A.M. Sandford: "Signal Identification System", US Patent 5,113,437, Thorn EMI Plc, May 1992.

[114] R. Petrovic, J.M. Winograd, K. Jemili, E. Metois: "Apparatus and Method for Encoding and Decoding Information in Analog Signals", US Patent 5,940,135, Aris Technologies, Inc., August 1999.

[115] J. Wolosewicz: "Apparatus and Method for Encoding and Decoding Information in Audio Signals", US Patent 6,005,501, Aris Technologies, Inc., December 1999.

[116] L. Boney, A.H. Tewfik, K.N. Hamdy: "Digital Watermarking for Audio Signals", *IEEE ICMCS'96*, pp.473-480, 1996.

[117] X. Li, H. Yu: "Transparent and Robust Audio Data Hiding in Cepstrum Domain", *IEEE International Conference on Multimedia & Expo (ICME'00)*, New York City, NY, 2000.

[**Specialized Embedding: Video**]

[118] F. Hartung, B. Girod: "Watermarking of Uncompressed and Compressed Video", *Signal Processing*, vol.66, 1998.

[119] M.D. Swanson, B. Zhu, A.H. Tewfik: "Multiresolution Scene-Based Video Watermarking Using Perceptual Models", *IEEE Journal on Selected Areas in Communication (JSAC)*, v.16, n.4, pp540-550, May 1998.

[120] W. Zhu, Z. Xiong, Y-Q. Zhang: "Multiresolution Wavelet-Based Watermarking of Images and Video", *IEEE Trans. Circuits and Systems for Video Tech*, vol. 9, pp. 545-550, June 1999.

[121] M.L. Miller, I.J. Cox, J.A. Bloom: "Watermarking in the Real World: An Application to DVD", *Proc. of Watermark Workshop at ACM Multimedia'98*, 1998.

[**Specialized Embedding: Miscellenous**]

[122] G.W. Braudaway, K.A. Magerlein, F. Mintzer: "Protecting Publicly-Available Images With A Visible Image Watermark", *SPIE Conf. on Optical Security and Counterfeit Deterrence Techniques*, vol. 2659, pp. 126-133, Feb. 1996.

[123] J. Meng, S-F. Chang: "Embedding Visible Video Watermarks in the Compressed Domain", *International Conf. on Image Processing (ICIP)*, 1998.

[124] E. Praun, H. Hoppe, A. Finkelstein: "Robust Mesh Watermarking", *ACM SIGGRAPH*, 1999.

[125] J. Stern, G. Hachez, F. Koeun, J-J. Quisquater: "Robust Object Watermarking: Application to Code", *3rd Info. Hiding Workshop (IHW'99)*, 1999.

[**Miscellaneous Applications of Data Hiding**]

[126] D.A. Silverstein, S.A. Klein: "Precomputing and Encoding Compressed Image Enhancement Instructions", in review for IEEE transactions on image processing, http://www.best.com/∼amnon/ Homepage/Research/Papers/ EncodingEnhance/EncodingEnhance.html .

[127] D.A. Silverstein, S.A. Klein: "Precomputing and Encoding Compressed Image Enhancement Instructions", United States Patent 5,822,458, Oct. 1998.

[128] J. Song, K.J.R. Liu: "A Data Embedding Scheme for H.263 Compatible Video Coding", *ISCAS'99*, vol.4, 1999.

[129] J. Song, R. Poovendran, W. Trappe, K.J.R. Liu: "A Dynamic Key Distribution Scheme Using Data Embedding for Secure Multimedia Multicast", *SPIE Electronic Imaging*, 2001.

[130] P. Yin, B. Liu, H. Yu: "Error Concealment Using Information Hiding", to appear in *ICASSP'01*, 2001.

[**Watermark Attacks and Security**]

[131] F. Hartung, J.K. Su, B. Girod: "Spread Spectrum Watermarking: Malicious Attacks and Counterattacks", *Proc. of SPIE, Security and Watermarking of Multimedia Contents*, vol. 3657, Jan., 1999.

[132] S. Craver, N. Memon, B-L. Yeo, M. M. Yeung: "Can Invisible Watermarks Resolve Rightful Ownerships?", *IBM Research Report*, 1996.

[133] S. Craver, N. Memon, B-L. Yeo, M. M. Yeung: "On the Invertibility of Invisible Watermarking Techniques", *ICIP*, 1997.

[134] S. Craver, N.Memon, B-L. Yeo, M.M. Yeung: "Rosolving Rightful Ownerships with Invisible Watermarking Techniques: Limitations, Attacks, and Implications", *IEEE Journal on Selected Areas in Communication (JSAC)*, v.16, n.4, pp573-586, May 1998.

[135] H. Stone, "Analysis of Attacks on Image Watermarks with Randomized Coefficients", Technical Report, NEC Research Institute, 1996.

[136] I. Cox, J-P. Linnartz: "Some General Methods for Tampering with Watermarks", *IEEE Journal Selected Areas of Communications (JSAC)*, vol.16, no.4, May, 1998.

[137] F. Petitcolas, R. Anderson, M. Kuhn, "Attacks on Copyright Marking Systems", *2nd Workshop on Info. Hiding*, 1998.

[138] *UnZign*, http://www.altern.com/watermark/, A Watermark Robustness Testing Software, 1997.

[139] M. Holliman, N. Memon, "Counterfeiting Attacks on Oblivious Blockwise Independent Invisible Watermarking Schemes", *IEEE Trans. on Image Processing*, vol.9, no.3, March 2000.

[140] Secure Digital Music Initiative (SDMI): http://www. sdmi.org

[141] Secure Digital Music Initiative (SDMI): "SDMI Portable Device Specification", Part 1, ver 1.0, 1999.

[142] Test result of International Evaluation Project for Digital Watermark Technology for Music: http://www.nri.co.jp/english/ news/ 2000/001006.html , 2000.

[143] J. Boeufl, J.P. Stern: "An Analysis of One of the SDMI Candidates", Technical Report, http://www.julienstern.org/ sdmi/files/sdmiF /sdmiF.html, 2001.

[**"Innocent" Tools for Building Attacks**]

[144] K. Jung, J. Chang, C. Lee, "Error Concealment Technique Using Projection Data for Block-based Image Coding", *Proc. of SPIE Conf. on Visual Communication and Image Processing*, vol.2308, pp1466-1476, 1994.

[145] W. Zeng, B. Liu, "Geometric-structure-based Directional Filtering for Error Concealment in Image/Video Transmission", *SPIE Photonics East'95*, vol.2601, pp.145-156, Oct. 1995.

[146] H. Igehy, L. Pereira, "Image Replacement Through Texture Synthesis", *ICIP*, 1997.

[147] Scalable Display Wall, http://www.cs. princeton. edu/ omnimedia , 1999.

[148] GoldWave: http://www.goldwave.com (audio editing software), 2000.

[**Topics on Image Processing and Analysis**]

[149] H.S. Stone, B. Tao, M. McGuire: "Analysis of Image Registration Noise Due to Rotationally Dependent Aliasing", Technical Report 99-057R, NEC Research Institute, 1999.

[150] M. McGuire: "An Image Registration Technique for Recovering Rotation, Scale and Translation Parameters", Technical Report 98-018, NEC Research Institute, 1998.

[151] Corel Stock Photo Library, Corel Corporation, Canada.

[152] B. Liu, T. Chang, H. Gaggioni: "On the Accuracy of Transformation Between Color Components in Standard and High Definition Television", *HDTV'92 Workshop*, pp57, 1992.

[**References on Multimedia Communication**]

[153] B. Shen, I.K. Sethi, V. Bhaskaran: "Adaptive Motion Vector Resampling for Compressed Video Down-scaling," *ICIP'97*, 1997.

[154] N. Yeadon, F. Garcia, D. Hutchison, D. Shepherd: "Continuous Media Filters for Heterogeneous Internetworking," *Proceedings of SPIE - Multimedia Computing and Networking (MMCN'96)*, 1996.

[155] N. Merhav, V. Bhaskaran: "A Transform Domain Approach to Spatial Domain Image Scaling," *ICASSP'96*, 1996.

[156] M.T. Orchard, G.J. Sullivan: "Overlapped Block Motion Compensation: An Estimation-theoretic Approach," *IEEE Transaction on Image Processing*, vol. 3, NO. 5, 1994.

[157] B. Liu, K-W. Chow, A. Zaccarin: "Simple Method to Segment Motion Field for Video Coding," *Proceeding of SPIE*, vol. 1818, 1992.

[158] N. Merhav, V. Bhaskaran: "A Fast Algorithm for DCT-domain Inverse Motion Compensation," *ICASSP'96*, 1996.

[159] Y. Wang, Q. Zhu: "Error Control and Concealment for Video Communication: A Review", *Proc. of IEEE*, v.86, pp.974-997, May, 1998.

[160] Y. Wang, S. Wenger, J. Wen, A. Katasggelos: "Error Resilient Video Coding Techniques", *IEEE Signal Processing Magazine*, July, 2000.

[161] W. Zeng, S. Lei: "Efficient Frequency Domain Video Scrambling for Content Access Control", *Proc. of ACM Multimedia*, Nov. 1999.

[162] J. Wen, M. Muttrell, M. Severa: "Access Control of Standard Video Bitstreams", to appear in *Inter. Conf. on Media Future*, May 2001.

[**Min Wu's publications**]

[163] M. Wu, B. Liu: "Watermarking for Image Authentication", *IEEE International Conference on Image Processing (ICIP'98)*, Chicago, IL, 1998.

[164] M. Wu, B. Liu: "Digital Watermarking Using Shuffling", *IEEE International Conference on Image Processing (ICIP'99)*, Kobe, Japan, 1999.

[165] M. Wu, E. Tang, B. Liu: "Data Hiding in Digital Binary Image", *IEEE International Conference on Multimedia & Expo (ICME'00)*, New York City, NY, 2000.

[166] M. Wu, B. Liu: "On Modulation and Multiplexing Techniques For Multimedia Data Hiding", invited paper, to appear in *SPIE ITCOM 2001 - Multimedia Systems and Applications IV*, Aug. 2001.

[167] M. Wu, H. Yu, A. Gelman: "Multi-level Data Hiding for Digital Image and Video", *Proceedings of SPIE*, vol. 3845, *Photonics East Conference on Multimedia Systems and Applications*, Boston, MA, 1999.

[168] M. Wu, H. Yu: "Video Access Control via Multi-level Data Hiding", *IEEE International Conference on Multimedia & Expo (ICME'00)*, New York City, NY, 2000.

[169] C-Y. Lin, M. Wu, Y-M. Lui, J.A. Bloom, M.L. Miller, I.J. Cox: "Rotation, Scale, and Translation Resilient Public Watermarking for Images", *IEEE Transactions on Image Processing*, vol.10, no.5, May 2001.

[170] C-Y. Lin, M. Wu, J.A. Bloom, M.L. Miller, I.J. Cox, Y-M. Lui: "Rotation, Scale, and Translation Resilient Public Watermarking for Images", *Proceedings of SPIE*, vol. 3971, Electronic Imaging (EI'00) Conference on Security and Watermarking of Multimedia Contents, San Jose, CA, 2000.

[171] P. Yin, M. Wu, B. Liu: "Video Transcoding by Reducing Spatial Resolution", *IEEE International Conference on Image Processing (ICIP'00)*, Vancouver, Canada, 2000.

[172] P. Yin, M. Wu, B. Liu: "Error Concealment for MPEG Video Over Internet", submitted to *IEEE International Conference on Image Processing (ICIP'01)*, 2001.

[173] M. Wu, B. Liu: "Attacks on Digital Watermarks", *33th Asilomar Conference on Signals, Systems, and Computers*, 1999.

[174] M. Wu, S.A. Craver, E.W. Felten, B. Liu: "Analysis of Attacks on SDMI Audio Watermarks", *IEEE International Conference on Acoustic, Speech, and Signal Processing (ICASSP'01)*, 2001.

[175] S.A. Craver, P. McGregor, M. Wu, B. Liu, A. Stubblefield, B. Swartzlander, D.S. Wallach, D. Dean, E.W. Felten: "Reading Between the Lines: Lessons from the SDMI Challenge", *Proc. of 4th Info. Hiding Workshop*, 2001.

[176] M. Wu, B. Liu: "Data Hiding in Images and Videos: Part-I –Fundamental Issues and Solutions", submitted to *IEEE Trans. on Circuits and Systems for Video Technology*, 2001.

[177] M. Wu, H. Yu, B. Liu: "Data Hiding in Images and Videos: Part-II – Designs and Applications", submitted to *IEEE Trans. on Circuits and Systems for Video Technology*, 2001.

[178] M. Wu, B. Liu: "Data Hiding in Binary Images", submitted to *IEEE Trans. on Multimedia*, 2001.

[179] M. Wu, B. Liu: "Data Hiding for Image and Video Authentication", submitted to *IEEE Trans. on Image Processing*, 2001.

[180] M. Wu, R. Joyce, H-S. Wong, L. Guan, S-Y. Kung: "Dynamic Resource Allocation Via Video Content and Short-term Traffic Statistics", *IEEE Trans. on Multimedia, Special Issue on Multimedia over IP*, June 2001.

[181] M. Wu, R. Joyce, S-Y. Kung: "Dynamic Resource Allocation Via Video Content and Short-term Traffic Statistics", *IEEE International Conference on Image Processing (ICIP'00)*, invited paper, 2000.

[182] H-S. Wong, M. Wu, R. Joyce, L. Guan, S-Y. Kung: "A Neural Network Approach For Predicting Network Resource Requirement in Video Transmission", *IEEE Pacific Rim Conference on Multimedia (PCM'00)*, Sydney, Australia, 2000.