Multimedia Fingerprinting Forensics for Traitor Tracing

K. J. Ray Liu, Wade Trappe, Z. Jane Wang,
Min Wu, and Hong Zhao

EURASIP Book Series on Signal Processing and Communications

Multimedia Fingerprinting Forensics for Traitor Tracing

EURASIP Book Series on Signal Processing and Communications, Volume 4

Multimedia Fingerprinting Forensics for Traitor Tracing

K. J. Ray Liu, Wade Trappe, Z. Jane Wang, Min Wu, and Hong Zhao

Hindawi Publishing Corporation
http://www.hindawi.com

Hindawi Publishing Corporation
410 Park Avenue, 15th Floor, #287 pmb, New York, NY 10022, USA
Nasr City Free Zone, Cairo 11816, Egypt
Fax: +1-866-HINDAWI (USA Toll-Free)

ISBN 977-5945-18-6

Dedication

To Our Families

Contents

Preface

Multimedia is becoming an integral part of our daily life. It is a means for us to communicate important information with each other, as well as a way to express our creative sides. The information and art contained inside media have economic value, personal value, and often broader impacts on the general welfare of our society. Consequently, multimedia is a form of digital information that must be protected.

This book is about protecting the economic and sensitive nature of multimedia. Since the Internet has become increasingly widespread, and now reaches into our everyday actions, it is easy to foresee that our modern communication networks will become the means for distributing multimedia content. This distribution will take many forms, ranging from a deceptively simple download-and-play model where a single consumer is the end-target for that content to streaming modes of operation where content is being enjoyed simultaneously by many consumers. Regardless of how you look at it, the future of multimedia is closely tied to the pervasiveness of our communication infrastructure. It therefore seems natural to protect multimedia by securing its distribution across these networks, that is, by employing the methods of network security.

Although securing the network and protecting the data crossing the network from eavesdropping is certainly essential for protecting multimedia, it is nonetheless a generic problem with generic solutions. Network security methods are important to many other applications, such as electronic commerce and computer security, in addition to being important to multimedia security. However, this book, *Multimedia Fingerprinting Forensics for Traitor Tracing*, is not about securing the communication infrastructure that will deliver multimedia.

Rather, this book focuses on the issue of protecting multimedia content when it is outside the realm of cryptography and network security. It is now relatively easy for adversaries to access multimedia content after it has been decrypted. Adversaries may now alter and repackage digital content. Therefore, ensuring that media content is employed by authorized users for its intended purpose, regardless of how it was delivered, is becoming an issue of eminent importance for both governmental security and commercial applications. As such, this book is about issues that are unique to multimedia and focuses specifically on how multimedia, unlike generic data types, can be protected by using fingerprint signals that are invisibly embedded inside the multimedia to trace and deter unauthorized content redistribution. That is, this book is about the rather nascent field of multimedia forensics, where the goal is to track and identify entities involved in the illegal manipulation and unauthorized usage of multimedia content. Ultimately, a solid foundation for media forensics will deter content fraud.

This book is targeted at an audience that is familiar with the fundamentals of multimedia signal processing and will teach the reader about the tools needed to build, analyze, and deploy solutions that will protect a variety of multimedia types. It, therefore, provides foundational material intended to assist the digital rights management (DRM) engineer understand technologies that complement traditional cryptographic security methods.

In this book, we will review a few major design methodologies for collusion-resistant fingerprinting of multimedia and highlight common and unique issues of various different fingerprinting techniques. The goal is to provide a broad overview of the recent advances in fingerprinting for tracing and identifying colluders. We will first provide background on robust data embedding, upon which multimedia fingerprinting system is built. We will then introduce the basic concepts of fingerprinting and collusion and provide a discussion on the various goals associated with fingerprint design and colluder tracing. Detailed discussions are then provided on two major classes of fingerprinting strategies, namely, orthogonal fingerprinting and correlated fingerprinting, where the latter involves the design of suitable codes that are employed with code modulation to create the fingerprints. As part of our discussion, we will arrive at a unified view of fingerprint design that covers orthogonal fingerprints, coded fingerprints, and other correlated fingerprints. After concluding the discussion of fingerprint design methodologies, we will explore two applications of fingerprinting. We will explore the migration of multimedia forensic technologies to networks, whereby the fingerprinting process will be integrated in core multicast functionality to provide DRM solution suitable for streaming delivery of content. Next, we will examine the protection of a type of multimedia content that has, until recently, been left unprotected by multimedia security solutions. In particular, we will explore the design of fingerprints for digital curves and maps and exploit the unique properties of digital curves in order to devise fingerprinting solutions.

We would like to thank Ms. Hongmei Gou, a Ph.D. student in the University of Maryland, for her contribution and involvement in preparing the draft of Chapter 8. The results presented in this book have been, in part, supported by the National Science Foundation and the Air Force Research Laboratories. We would like to thank these organizations for the support to explore and develop this exciting research area.

K. J. Ray Liu
Wade Trappe
Z. Jane Wang
Min Wu
Hong Zhao

1 Introduction

The ubiquity of high-bandwidth communication technologies, in combination with well-developed multimedia standards, has led to the proliferation of multimedia content in both the government and commercial sectors. We are witnessing the integration of next-generation multimedia standards, such as MPEG-4 [1, 2, 3, 4] and MPEG-7 [5], into software and hardware. As a result of this integration, users are able to readily create, manipulate, and combine multimedia content, such as audio clips and segments of video.

Multimedia data has become the mode by which we communicate with each other. We share digital photos with childhood friends whom we have not seen in years, and we share home videos of our children with our parents. Video conferences and the sharing of recorded presentations allow both corporate and governmental sectors to increase their productivity. It is now easier for artists to create their own cinema or record the performance of their garage-operated band. The combination of the availability of multimedia software and hardware with the availability of the Internet and the Web has encouraged artists, professional and amateur alike, to share their creative expressions. Ultimately, this has led to the creation of a digital marketplace.

Whether you examine the role of multimedia to convey information between different branches of the government, or you examine the role of multimedia in the digital marketplace, the picture is the same: the promise of multimedia is great, but its successful adoption stands on a dangerous precipice right now as the very technologies which facilitate its success also threaten its success. The combination of multimedia technologies and a pervasive communication infrastructure introduces an explosion of threats to the sharing of multimedia content. The tools that allowed users to create content, also allow them to duplicate or forge content. The medium that allowed users to share their expressions also facilitates the sharing of illicit or fraudulent content.

The alteration, repackaging, and redistribution of multimedia content pose a serious threat to both governmental security and commercial markets. The ability to securely and reliably exchange multimedia information is a strategic imperative in order for governments to operate smoothly. In order to facilitate the global cooperation necessary to foster international business and to battle international threats, like terrorism, it is necessary that the respective government agencies share information. It is already common for video and imaging information associated with a particular military conflict zone to be gathered by an agency in one country and shared with agencies in an ally country. This sharing of information, though, to foreign agencies means that there is no direct mechanisms for accountability, and no definite guarantee that shared content will not be leaked. In order to prevent information from leaking out of an authorized circle, it is essential that government agencies have the forensic capability to track and identify entities involved in unauthorized redistribution of multimedia information.

In addition to the demands from a national security point of view, preventing the leakage of multimedia information is also crucial to the global economy. Let us examine the US copyright industries as an example. The US copyright industries, which include prerecorded CD/DVDs and tapes, as well as videos, motion pictures, and periodicals, account for about 5.2% of US Gross Domestic Product (GDP), or $531.1 billion, and are responsible for close to 6% of all US employment [6, 7]. The copyright industries, however, are experiencing a substantial decline in income and job positions, which is largely attributed to piracy. For example, US music sales by unit were reported to have dropped 31% from mid 2000 to 2003. The affected sectors include not only the retail stores, but also CD/DVD plant employees and workers from every aspect of the complex business of making and distributing multimedia content. The negative impact on the local and national economy is reflected by the significant reduction in the tax base of local and federal governments.

Given the popularity and economic value of media, such as music and video, it is not surprising that altered content is now distributed over the Internet for unauthorized purposes. The legal warfare that erupted around Napster, Kazaa, and DVD decryption and the recent delays in the introduction of rewritable DVDs to the consumer market serve as evidence of the important issues and powerful emotions at the core of this problem. For both the commercial and government sectors, unauthorized distribution of content may have very serious economic and political consequences. Therefore, before multimedia technologies can safely be used as a means to exchange information, or as the basis for establishing a viable digital marketplace aiming to share content with an eager consumer base, mechanisms must be in place to ensure that content is used for its intended purpose by legitimate users who have appropriate usage rights. In response, the past decade has witnessed the birth of an emerging market for digital rights management (DRM) technologies that protect the commercial rights and enforce the usage policies of content creators and distributors. The need for techniques that protect the digital rights of multimedia has become a critical issue for the multimedia community, and recent attention in international standards groups, such as MPEG-21 and SDMI [8, 9, 10], has shifted towards integrating security into the multimedia framework.

There are two fundamental and complementary approaches to address the problem of unauthorized distribution of information. First, media may be encrypted to prevent an unauthorized user from accessing the content [11, 12, 13, 14, 15, 16]. Second, one can attempt to identify who has had access to the information and whether it has been altered so that appropriate penalties can be meted out for illegal behavior. The application of access control to multimedia markets has proven problematic as the protection provided by encryption disappears when the content is no longer in the protected domain. It is feasible for users to access clear text representations prior to playing the media. Users can then redistribute unencrypted representations and subvert the digital rights of the original media distributors.

The second approach, which is the focus of this book, can be referred to generically as *multimedia forensics*. Multimedia forensics combines digital domain evidence to demonstrate that multimedia content has been altered, to indicate how these alterations were made, and to identify who participated in the alterations. Multimedia forensics opens up uncharted territory for the field of media security since introducing mechanisms that hold consumers accountable for their behavior requires interdisciplinary techniques that build upon the synergies between signal processing, watermarking, coding, security, and communication theory.

Much like the fingerprint is a key tool for performing forensics of a crime scene, the fundamental tool for media forensics is a special type of digital fingerprint suitable for multimedia content. Digital fingerprinting is a technology for enforcing digital rights policies whereby unique labels, known as *digital fingerprints*, are inserted into content prior to distribution to assist in investigating how unauthorized content was created, and what entities were involved in forming the fraudulent media. As illustrated in Figure 1.1, unique fingerprints are assigned to each intended recipient. In order for media fingerprinting to be a powerful forensic tool, it is essential that these fingerprints be difficult to remove. Fingerprints may be embedded into multimedia through robust watermarking techniques [17, 18, 19, 20, 21, 22], and a substantial amount of literature has been devoted to combating a variety of signal processing attacks mounted by an individual adversary on a single copy of a watermarked signal [23, 24, 25, 26, 27, 28].

Unfortunately, these robust watermarking techniques are not enough to provide the ability to trace an entity involved in the distribution of fraudulent content. In fact, robust embedding techniques are merely a necessary first step that protects against attacks mounted by an individual. Guaranteeing the appropriate use of multimedia content is not simply a traditional security issue with a single adversary. The global nature of the Internet has brought adversaries closer to each other, and it is now easy for a group of users with differently marked versions of the same content to work together and collectively mount attacks against fingerprints embedded inside of media data. These attacks, known as collusion attacks, provide a particularly cost-effective method for attenuating or removing each member's fingerprints—thereby making it difficult to detect any of the colluders involved. If we use an improperly designed embedding and identification scheme, then we open ourselves up to being vulnerable to small coalition of traitors working together to create a new version of the content with no detectable traces of the digital fingerprints. Thus, collusion poses a real threat to protecting media data and

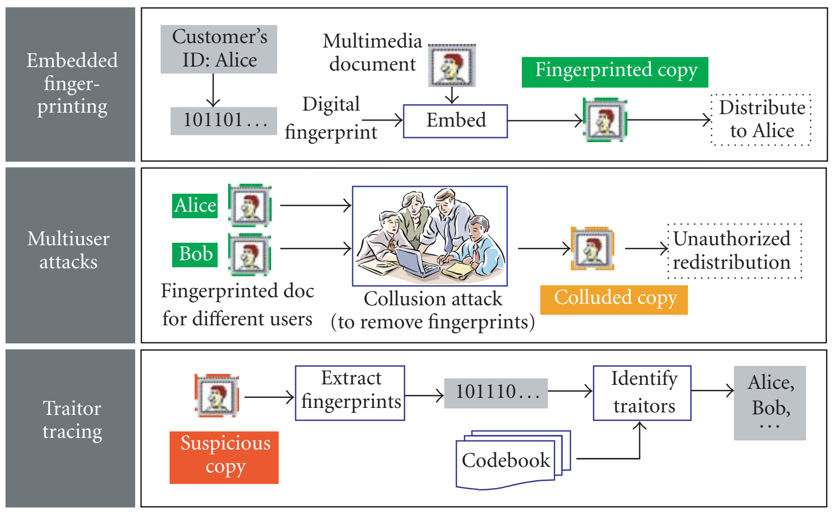


FIGURE 1.1. Using embedded fingerprinting for tracing users.

enforcing usage policies. It is desirable, therefore, to design fingerprints that resist collusion and identify the colluders.

This book is focused on exploring the task of creating and embedding multimedia fingerprints that are capable of resisting collusion, as well as on the challenge of developing efficient and effective techniques for identifying those entities involved in collusion. We will begin our study of multimedia fingerprints by first reviewing the basics of robust data embedding and watermarking techniques which may be used to insert fingerprints into multimedia. Next, we will introduce the problem of collusion and present both linear and nonlinear collusion strategies that adversaries might employ to subvert multimedia fingerprinting. After introducing collusion threats, we examine orthogonal embedding strategies, which are a very popular technique for marking multimedia content. Due to their prevalence, it is important that we understand their ability to resist collusion attacks, and we therefore present a thorough statistical analysis of the collusion resistance of orthogonal fingerprints. Following our discussion of orthogonal fingerprints, we propose group-oriented fingerprinting, which is a strategy whereby we take advantage of a priori knowledge of potential collusion patterns amongst the consumers in order to improve the design of the fingerprints. We then introduce a more general framework for building fingerprints in which we design a new family of codes, known as anticollusion codes, that are used in conjunction with code modulation to build collusion-resistant fingerprints. In addition to presenting the design of these collusion-resistant fingerprints, we present several algorithms for identifying colluders. We then address the secure and efficient distribution of fingerprint multimedia over networks, and investigate how to multicast fingerprinted video to multiple users without revealing the secrecy of the multimedia content as well as that of the embedded fingerprints. Finally, we present a new robust curve fingerprinting algorithm to trace and track topographic maps as well as writings/drawings from pen-based inputs. The protection of such documents has increasing importance to emerging digital operations in government, military, intelligence, and commerce.

2 Preliminaries on data embedding

This chapter reviews the basics of robust data embedding. After a brief overview on digital watermarking and data embedding technologies, we steer our attention to a popular class of robust embedding techniques known as the spread-spectrum embedding. The detailed formulation on the embedding and detection aspects of the spread-spectrum technique establishes a foundation to unveil our technical discussions on multimedia fingerprinting in the subsequent chapters.

2.1. Content protection via digital watermarking

Multimedia content has both commercial and personal value that must be protected before one can share his/her work, or businesses can be founded to distribute and add value to their creations. Prior to digital multimedia content being put onto the network for delivery, the data can be modified to help protect the intellectual property of the content's creators and service providers. Encryption and data embedding are two complementary techniques for protecting multimedia content that have different goals. The primary goal behind encryption is confidentiality [29, 30, 31], that is, to provide access control so that only authorized users with the correct decryption keys can access the content. The protection provided by encryption terminates after decryption. Complementing this functionality, data embedding or digital watermarking associates a set of secondary data with the host media in a seamless way [17, 18]. The term "digital watermark" comes from an analogy to its analog counterpart: as an art of paper making, paper watermarks usually indicate the origin and the ownership, and/or establish the integrity and prevent counterfeiting. Similarly, digital watermarking has been considered in several real-world applications related to multimedia content protection and security. These include copy prevention for DVD and digital music, the assertion of ownership, the fingerprinting and tracing of content recipients, and the authentication of the content. While the protection provided by watermarks is usually passive, the embedded watermarks can travel with the host media and assume their protection function even after decryption. This capability of associating additional data with the digital multimedia content in a seamless way is so unique that few conventional protection tools for generic data, such as cryptographic encryption, can offer it.

Digital watermarking technologies, or more generally, the data embedding technologies, can be applied for a number of applications, each of which has its own design requirements in terms of the imperceptibility, the robustness, and the embedding payload measured in terms of how many bits are embedded. As the embedding process more or less changes the original multimedia signal (known as the *host signal*), it is important in a number of applications to confine the changes to below the perceivable levels of human in order to preserve the commercial and aesthetic values of the host multimedia signal. In the following, we briefly review a few major applications and approaches of data embedding for multimedia.

2.1.1. Major applications and design requirements

Ownership protection. In this application, a watermark signal is secretly selected by the copyright holder to represent his/her ownership, and is imperceptibly embedded in the multimedia source. As pirates are highly motivated to remove the copyright watermark, the embedding should survive common processing and resist intentional attacks so that the owner can demonstrate the presence of this watermark in case of dispute to verify his/her ownership. The detection should have as little ambiguity and false alarm as possible. In most scenarios, the total number of bits that can be embedded and extracted reliably does not have to be high.

Authentication or tampering detection. In this application, we embed a set of secondary data in the multimedia source beforehand, and later use it to determine whether the host media is tampered or not. The robustness against removing the watermark or making it undetectable is not a major concern as this clearly signals the occurrence of tampering and there is no strong incentive to do so from an attacker's point of view. The main threat is the forging of a valid authentication watermark in an unauthorized or tampered multimedia signal, which must be prevented. In many practical applications, it is also desirable to locate the tampered regions and distinguish some noncontent changes (such as those incurred by moderate lossy compression) from other changes (such as content tampering). In general, the embedding payload should be sufficiently high to accommodate these needs. The detection must be performed without using the original unwatermarked copy, because either this original is unavailable or its integrity has not been established yet. This kind of detection is known as *noncoherent detection* or *blind detection*.

Digital fingerprinting. As we have explained in the introductory chapter, a watermark in this application serves as a fingerprint to help trace the originator or recipients of a particular copy of multimedia content. The robustness against removal and the ability to convey a nontrivial number of bits are necessary requirements. In addition, digital fingerprinting techniques should also be robust against *collusion* when users having access to the same host image embedded with different fingerprint IDs get together and try to remove the fingerprints through such operations as averaging. To assure the reliable tracing of true traitors and avoid framing innocents, we must carefully design how to construct, embed, and detect fingerprint signals. This suggests that there are more considerations than those that the robust embedding technologies offered in the existing digital watermarking literature. We will devote our attention to these issues in the rest of this book.

Copy control and access control. The embedded watermark in this application represents certain copy control or access control policy. A watermark detector is often integrated in a recording or playback system. 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 nontrivial number of bits are required. Two examples from the past standardization efforts are the DVD copy control [32] and the Secure Digital Music Initiatives (SDMI) activities [9].

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, many scenarios may prefer a certain degree of robustness against common processing such as lossy compression. More generally, data embedding is a tool to convey side information while retaining the original appearance. This property has been found useful in multimedia communications [33, 34, 35] to achieve additional functionalities, or improve security and performance.

2.1.2. Basic embedding approaches

A typical data hiding framework is illustrated in Figure 2.1. Starting with an original digital media (I_0), which is also known as the *host media* or *cover work*, the embedding module inserts in it a set of secondary data ($\underline{b}$), which is referred to as *embedded data* or *watermark*, to obtain the *marked media* (I_1). The insertion or embedding is done such that I_1 is perceptually identical to I_0. The difference between I_1 and I_0 is the distortion introduced by the embedding process. In most cases, the embedded data is a collection of bits, which may come from an encoded character string, from a pattern, or from some executable agents, depending on the application.

The embedded data $\underline{b}$ will be extracted from the marked media I_1 by a detector, often after I_1 has gone through various processing and attacks. The input to the detector is referred to as *test media* (I_2), and the *extracted data* from I_2 is denoted by $\hat{\underline{b}}$. The difference between I_2 and I_1 is called *noise*. In such applications as ownership protection, fingerprinting, and access control, accurate decoding of hidden data from distorted test media is preferred. In other applications such as authentication and annotation, robustness is not critical.

The embedding of one bit in host media is basic to every data hiding system. Almost all embedding approaches belong to one of two general types. Below we examine them in more detail.

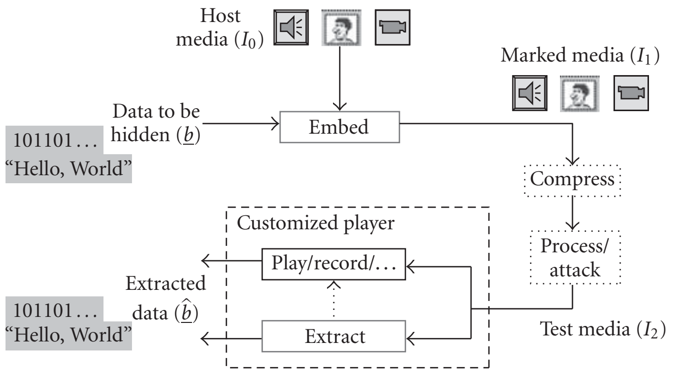


FIGURE 2.1. General framework of data hiding systems.

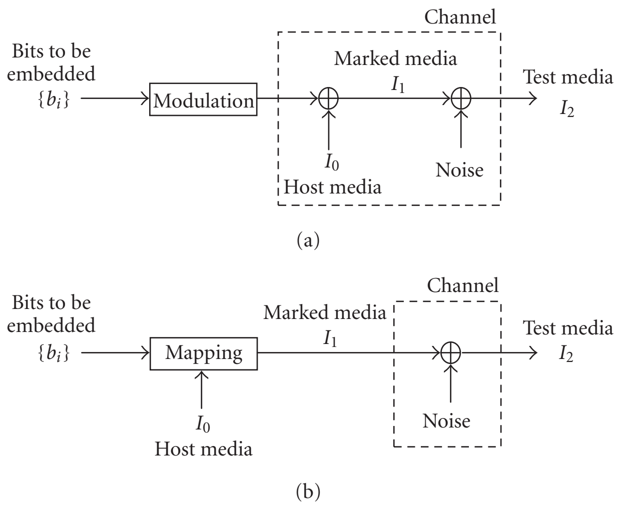


FIGURE 2.2. Channel models for (a) Type-I and (b) Type-II embedding.

In *Type-I* embedding, the secondary data, possibly encoded, modulated, and/or scaled, is added to the host signal, as illustrated in Figure 2.2a. The addition can be performed in a specific domain or on specific features. To embed one bit b, the difference between the marked signal I_1 and the original host signal I_0 is a function of b, that is, $I_1 - I_0 = f(b)$. Although it is possible to detect b directly from I_1 [36], the knowledge of I_0 helps enhance detection performance. Additive spread-spectrum watermarking [23, 24] is an example of Type-I, at which we will take a closer look in the next section.

In *Type-II* embedding, the signal space is partitioned into subsets, each of which is mapped by a function $g(\cdot)$ to the set of values taken by the secondary data, as illustrated in Figure 2.2b. The marked value I_1 is then chosen from the subset that 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. A simple example is *odd-even embedding*, whereby a closest even number is used as I_1 to embed a "0" and a closest odd number is used to embed a "1." The embedded bit is extracted simply by checking the odd-even parity, which does not require the knowledge of original I_0. There can be other constraints imposed on I_1 for robustness considerations. For example, the enforcement can be done in a quantized domain with a step size Q [18, 37]. The odd-even enforcement can also be extended to higher dimensions involving features computed from a set of host components.

Under blind detection where the host signal is not available at the detector and becomes a major source of noise for Type-I embedding, the number of bits that can be reliably embedded by Type-II is much higher than that of Type-I when the noise is not strong [18, 38, 39]. On the other hand, Type-I is commonly used for robust embedding with strong noise (especially when noise becomes much stronger than the watermark) as well as for nonblind detection. Motivated by Costa's information-theoretical result [40], distortion compensation has been proposed to be incorporated into quantization-based enforcement embedding [39, 41, 42], where the enforcement is combined linearly with the host signal to form a watermarked signal. The optimal scaling factor is a function of WNR and will increase the number of bits that can be embedded. This distortion-compensated embedding can be viewed as a combination of Type-I and Type-II embedding.

2.2. Robust additive spread-spectrum embedding

With a big picture in mind, we now zoom in to the problem of robust embedding, which serves as an important building block for embedding fingerprints into the multimedia content.

Fingerprinting multimedia requires the use of robust data embedding methods that are capable of withstanding attacks that adversaries might employ to remove the fingerprint. Although there are many techniques that have been proposed for embedding information in multimedia signals [17], in the sequel we will use the spread-spectrum additive embedding technique for illustrating the embedding of fingerprint signals into multimedia. Spread-spectrum embedding has proven robust against a number of signal processing operations (such as lossy compression and filtering) and attacks [23, 24]. With appropriately chosen features and additional alignment procedures, the spread-spectrum watermark can survive moderate geometric distortions, such as rotation, scale, shift, and cropping [43, 44]. Further, information-theoretic studies suggest that it is nearly capacity optimal when the original host signal is available in detection [38, 39]. The combination of robustness and capacity makes spread-spectrum embedding a promising technique for protecting multimedia. In addition, as we will see in later chapters, its capability of putting multiple marks in overlapped regions also limits the effective strategies mountable by colluders in fingerprinting applications.

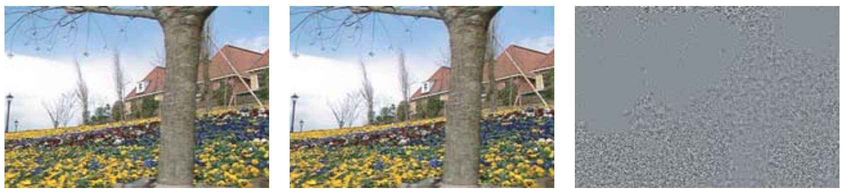


Figure 2.3. The original flower garden image, the watermarked version, and the watermark embedded, respectively. The watermark shown is the amplified difference between the original and the watermarked versions by a factor of 5, with mid-level gray denoting zero amplitude and black/white denoting large amplitude.

2.2.1. Overview of spread-spectrum embedding

Spread-spectrum embedding borrows ideas from spread-spectrum modulation [45]. The basic process of spread-spectrum embedding consists of four steps. The first step is to identify and compute features that will carry watermark signals. Depending on the application and design requirements, the features can be signal samples, transform coefficients (such as DCT and DFT coefficients), or other functions of the media content. Next, we generate a watermark signal and tune its strength to ensure imperceptibility. Typically, we construct the watermark to cover a broad spectrum as well as a large region of the content, resulting in a watermark that resembles noise. A third step is to add the watermark to the feature signal. Finally, we replace the original feature signal with the watermarked version and convert it back to the signal domain to obtain a watermarked signal. The detection process for spread-spectrum watermarks begins with extracting features from a media signal in question. Then the similarity between the features and a watermark is examined to determine the existence or absence of the watermark in the media signal. Typically, a correlation similarity measure is used, often in conjunction with preprocessing (such as whitening) and normalization [17].

An example of spread-spectrum watermarking for image is provided in Figure 2.3. Messages are mapped to a noise-like watermark pattern shown in Figure 2.3c, where the pixels with mid-level gray indicate zero amplitude in the corresponding part of the watermark, and the brighter or darker pixels indicate large positive or negative amplitude, respectively. We can see that the amplitude of the watermark signal is closely related to Figure 2.3a, the original image to be watermarked. The watermark corresponding to the smooth area of the image (such as sky) is weaker, while that to the texture area (such as the flower bed) is stronger. The watermark is added to the original image to produce the watermarked version as shown in Figure 2.3b. The small strength of the watermark coupled with the noisy nature and perceptual shaping helps the spread-spectrum watermark be imperceptible to eyes.

2.2.2. Distortion and attacks against robust embedding

Many robust embedding tasks are in a competitive environment, where an adversary has the incentive to render the embedded data undetectable. Understanding the threats and analyzing effective attacks play an important role in identifying the weaknesses and limitations of the existing watermarking schemes, as well as in suggesting directions for further improvement. More importantly, it helps us have a realistic understanding of what aspect of the problem can be solved by data embedding technologies and what should be taken care of by other means such as an appropriate business model and policy.

Common signal processing, such as lossy compression, lowpass filtering, and histogram-based enhancement, may be unavoidable during the life span of the multimedia content. These processing methods distort the data embedded in multimedia signal in such a way that is often modeled as additive noise. As we will see in the next subsection, detecting watermark under these distortions can be formulated using classical signal detection theory. As long as the distortion is not too severe and the watermark is sufficiently long, the embedded data can be detected with high probability [23].

Geometric transformations, such as rotations, scale, translation, and nonlinear warping, are notorious attacks against spread-spectrum embedding. Surviving them is particularly challenging when the original unwatermarked image is not available at the detector. This is mainly due to the misalignment introduced by the geometric attacks between the embedded watermark in an image in question and the reference watermark presented to a detector. Because spread-spectrum watermarks generally have low autocorrelation at nonzero shift and taking correlation is the popular way to detect these watermarks, the misalignment 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 to achieve. For example, Fourier magnitude domain 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.[1] On the other hand, combating rotation and scaling is more sophisticated. Common approaches to building geometric resilient watermarks include embedding an invisible registration pattern [46, 47], using salient features from a host signal as reference [48], or embedding watermark in some resilient domains. The domain in the latter category can be a canonical, normalized space based on moments [49], or feature domain exploiting Fourier properties [43].

To illustrate the effect of various distortions on watermark, we have implemented a spread-spectrum embedding similar to the one in [23] but embedded the watermark in the magnitude of discrete Fourier transform (DFT) coefficients rather than the DCT coefficients. For a 512×512 Lena image, the PSNR of the watermarked image with respect to the original is 42.88 dB. The detection results on the marked Lena image are shown in Table 2.1. We can see that minor rotation

[1]Larger shifts are likely to incur cropping, which may reduce the detection statistics.

Table 2.1. Detection statistics of spread-spectrum watermarking on a 512×512 Lena image in DFT magnitude domain under various distortions.

Test condition	*Detection statistics*	*Test condition*	*Detection statistics*
With no distortion	13.54	With 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	—	—

and scaling are powerful enough to render the watermark undetectable. On the other hand, the watermark can survive strong compression and translation, with detection statistics well above the threshold that is usually set between 3 and 6, corresponding to a false alarm probability of 10^{-3} to 10^{-10}.

Let us focus now on the possible distortions and attacks for fingerprinting applications, where the embedded data serves as identifying IDs to help trace the originator or recipients of a particular copy of multimedia content. As a traitor has incentive to remove the identifying fingerprint before leaking the content, the fingerprint should first of all be embedded in a robust way and can survive a number of distortions and attacks, such as compression and filtering. Geometric attacks, however, are not a major concern, as it is often reasonable to have the original image available to detector for fingerprinting applications [23, 44, 50]. The rationale for this nonblind detection assumption is that the fingerprint verification is usually handled by the content owner or by an authorized central server, who can have access to the original host signal and use it in detection to answer the primary question of whose fingerprint is in the suspicious document. As we will see later in this chapter, having original unmarked copy in the detection gives a high power ratio between watermark and noise, which allows for high resistance against noise and attacks. Further, using the original unmarked copy as a reference copy, the detector can register a test copy that suffers from geometric distortions, and the geometric distortion can be substantially inverted. A thorough study in [44] as well as in Chapter 8 has shown that the alignment error of inverting geometric attacks is very small and will not significantly deter the detection performance.

With many users receiving the same multimedia content but embedded with different data, new issues arise in fingerprinting applications. A group of users can work together, examine their different copies, create a new signal that may no longer be tied to any of the colluders. We will present a detailed discussion in Chapter 3 regarding this *multiuser collusion attacks*. It is worth mentioning that another class of collusion attacks, which is sometimes referred to as *intracontent collusion*, may be mounted against the embedded data by replacing each segment of the content signal with another seemingly similar segment from different spatial or temporal regions of the content. As an example, an adversary may produce an attacked signal by integrating information from consecutive frames to remove watermarks from a video sequence [51]. Such *intracontent collusion* should be taken into account in designing robust embedding. We will not further elaborate on this issue in the current book. Interested readers may refer to [51, 52, 53] for detailed discussions.

We now take a closer look at the mathematical formulation on embedding and detection.

2.2.3. Mathematical formulation

Suppose that the host signal is a vector denoted as $\mathbf{x}$ and that we have a family of watermarks $\{\mathbf{w}_j\}$ that are fingerprints associated with the different users who purchase the rights to access $\mathbf{x}$. Before the watermarks are added to the host signal, every component of each $\mathbf{w}_j$ is scaled by an appropriate factor, that is, $s_j(l) = \alpha(l)w_j(l)$, where we refer to the lth component of a vector $\mathbf{w}_j$ by $w_j(l)$. One possibility for $\alpha(l)$ is to use the *just-noticeable-difference* (JND) from human visual system models [24]. Corresponding to each user is a marked version of the content $\mathbf{y}_j = \mathbf{x} + \mathbf{s}_j$.

The content may experience additional distortion before it is tested for the presence of the watermark $\mathbf{s}_j$. This additional noise could be due to the effects of compression, or from attacks mounted by adversaries in an attempt to hinder the detection of the watermark. We represent this additional distortion by $\mathbf{z}$. There are therefore two possible sources of interference hindering the detection of the watermark: the underlying host signal $\mathbf{x}$ and the distortion $\mathbf{z}$. For simplicity of notation, we gather both of these possible distortions into a single term denoted by $\mathbf{d}$. As we will discuss later, in some detection scenarios, it is possible for $\mathbf{d}$ to only consist of $\mathbf{z}$. A test content $\mathbf{y}$ that originates from user j, thus can be modeled by

$$\mathbf{y} = \mathbf{s}_j + \mathbf{d}. \tag{2.1}$$

The watermarks $\{\mathbf{w}_j\}$ are often chosen to be orthogonal noise-like signals [23], or are represented using a basis of orthogonal noise-like signals $\{\mathbf{u}_i\}$ via

$$\mathbf{w}_j = \sum_{i=1}^{B} b_{ij}\mathbf{u}_i, \tag{2.2}$$

where $b_{ij} \in \{0, 1\}$ or $b_{ij} \in \{\pm 1\}$. We will present detailed discussions on different ways to construct watermarks for fingerprinting purposes in later chapters.

The detection of additive watermarks can be formulated as a hypothesis testing problem, where the embedded data is considered as the signal that is to be detected in the presence of noise. For the popular spread-spectrum embedding [23, 24], the detection performance can be studied via the following simplified antipodal model:

$$\begin{aligned} H_0 &: y(l) = -s(l) + d(l) \quad (l = 1, \ldots, L) \text{ if } b = -1, \\ H_1 &: y(l) = +s(l) + d(l) \quad (l = 1, \ldots, L) \text{ if } b = +1, \end{aligned} \tag{2.3}$$

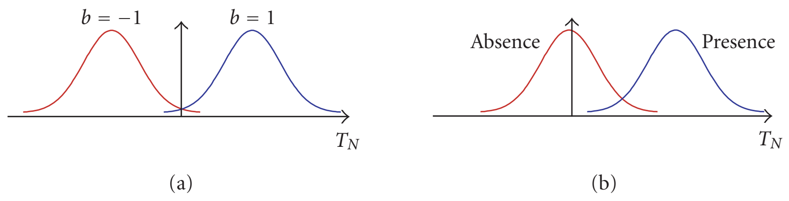


Figure 2.4. Illustration of the distribution of watermark detection statistics under i.i.d. Gaussian noise: (a) antipodal modulation with $b \in \{\pm 1\}$; (b) on-off keying with $b \in \{-1, +1\}$.

where $\{s(l)\}$ is a deterministic spreading sequence (often called the *watermark*), b is the bit to be embedded and is used to antipodally modulate $s(l)$, $d(l)$ is the total noise, and N is the number of samples/coefficients used to carry the hidden information.

If $d(l)$ is modeled as i.i.d. Gaussian $\mathcal{N}(0, \sigma_d^2)$, the optimal detector is a (normalized) correlator [54] with a detection statistic T_N given by

$$T_N = \mathbf{y}^T \mathbf{s} / \sqrt{\sigma_d^2 \cdot \|\mathbf{s}\|^2}, \tag{2.4}$$

where $\mathbf{y} = [y(1), \ldots, y(L)]^T$, $\mathbf{s} = [s(1), \ldots, s(L)]^T$, and $\|\mathbf{s}\|$ is the Euclidean norm of $\mathbf{s}$. Under the i.i.d. Gaussian assumption for $d(l)$, T_N is Gaussian distributed with unit variance and a mean value

$$E(T_N) = b \cdot \sqrt{\frac{\|\mathbf{s}\|^2}{\sigma_d^2}}. \tag{2.5}$$

If b is equally likely to be "-1" and "$+1$," it is often known as the *antipodal modulation*. In this case, the optimal (Bayesian) detection rule is to compare T_N with a threshold of zero to decide H_0 against H_1, and the probability of error is $\mathcal{Q}(E(T_N))$, where $\mathcal{Q}(x)$ is the probability $P(X > x)$ of a Gaussian random variable $X \sim \mathcal{N}(0, 1)$. This detection rule is illustrated in Figure 2.4a. The error probability can be reduced by raising the watermark-to-noise ratio (WNR) $\|\mathbf{s}\|^2/(L\sigma_d^2)$, or increasing the length L of the spreading sequence per bit. 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) [24], which we will discuss more in Section 2.2.5. Assuming that both $\{s(l)\}$ and $\{d(l)\}$ are zero mean, σ_d^2 is estimated from the power of $y(l)$ and $s(l)$, for example, via $\hat{\sigma}_d^2 = (\|\mathbf{y}\|^2 - \|\mathbf{s}\|^2)/L$.

The i.i.d. Gaussian noise assumption is critical for the optimality of a correlator-type detector, but it may not reflect the statistical characteristics of the actual noise and interference. For example, the noise and interference in different frequency bands can differ. In such a scenario, we should first normalize the observations $\{y(l)\}$ by the corresponding noise standard deviation to make the noise distribution i.i.d. before taking the correlation [55]. That is,

$$T'_N = \sum_{l=1}^{L} \frac{y(l) \cdot s(l)}{\sigma^2_{d(l)}} \Big/ \sqrt{\sum_{l=1}^{L} \frac{s^2(l)}{\sigma^2_{d(l)}}}, \tag{2.6}$$

$$E(T'_N) = b\sqrt{\sum_{l=1}^{L} \frac{s^2(l)}{\sigma^2_{d(l)}}}. \tag{2.7}$$

This can be understood as a weighted correlator with more weight given to less noisy components. Similarly, colored Gaussian noise needs to be whitened before correlation [17]. In reality, the interference from the host signal as well as the noise introduced by many attacks and distortions are often non-Gaussian and nonstationary. Under these scenarios, an optimal detector can be derived by using a non-Gaussian and/or a nonstationary noise model in the classic detection framework [54]. For example, generalized matched filters have been proposed as watermark decoders for the generalized Gaussian channel model [56, 57]. Channel estimation has also been used in conjunction with the generalized matched filter against fading and geometrical distortions with unknown parameters [56].

Another model, used often for conveying ownership information [23, 24], leads to a similar hypothesis testing problem described by

$$\begin{aligned} &H_0 : y(l) = d(l) \quad (l = 1, \dots, L) \text{ if watermark is absent,} \\ &H_1 : y(l) = s(l) + d(l) \quad (l = 1, \dots, L) \text{ if watermark is present.} \end{aligned} \tag{2.8}$$

This is often referred to as *on-off keying* (OOK). The detection statistic is the same as shown in (2.4) for additive white Gaussian noise (AWGN) or (2.6) for independent Gaussian noise with nonidentical variance, and the distribution of the detection statistic under each hypothesis is shown in Figure 2.4b. The threshold for distinguishing the two hypotheses is a classic detection problem, for which we can use a Bayesian rule or a Neyman-Pearson rule [54]. The probability of detection errors can be obtained accordingly.

2.2.4. Alternative detection statistics

As we can see from the previous subsection, watermark detection can be formulated as hypothesis testing [58, 59], which is commonly handled by evaluating the similarity between the estimated watermark sequence and each watermark in the database through a correlation-based statistic. The correlation statistics are simple to implement and optimal when the noise is additive white Gaussian. In the watermarking literature and practice, several correlation-based statistics have been employed. They share a kernel term measuring the total correlation $\langle \mathbf{y}, \mathbf{s} \rangle$, and usually differ in how they are normalized. To facilitate the evaluation of detection performance, we often normalize the detection statistic to make it have unit variance and follow approximately a Gaussian distribution under distortions and attacks. There are several ways to do so [60], for example, through normalizing by the product of the noise's standard deviation and the watermark's L_2 norm as seen in the T_N statistic in (2.4), or through a logarithm-based transformation to be introduced next.

Z **statistic for detection**

A nonlinear function of the sample correlation coefficient from the mathematical statistics literature [61] was introduced to the watermarking community by Stone and colleagues of the NEC Research Institute [62]. This is often referred to as the *Fisher's Z statistic*. Among several correlation-based statistics to be compared in Chapter 3, the Z statistic shows excellent robustness against different collusion attacks and does not require the explicit estimation of the noise's variance.

The Z statistic originated from the statistical problem of sampling a bivariate normal population [61], that is, to obtain i.i.d. samples from a pair of random variables that are jointly Gaussian distributed, with correlation coefficient ρ unknown to the observers. Let r be the sample correlation coefficient computed from L pairs of sample data. The following function of r,

$$\frac{1}{2}\log\frac{1+r}{1-r}, \tag{2.9}$$

has been shown to asymptotically follow a normal distribution when $L \to \infty$ [61], with the mean approximating $(1/2)\log((1+\rho)/(1-\rho))$ and the variance approximating $1/(L-3)$. Thus the Z statistic defined below follows a unit-variance Gaussian distribution:

$$Z \triangleq \frac{\sqrt{L-3}}{2}\log\frac{1+r}{1-r} \sim \mathcal{N}\left(\frac{\sqrt{L-3}}{2}\log\frac{1+\rho}{1-\rho}, 1\right). \tag{2.10}$$

The approximation is found excellent with as few as 10 pairs of samples.

A simplified model considers that the corresponding components of an extracted signal in question and a particular-user Alice's watermark are i.i.d. samples from a bivariate normal population. When the extracted watermark does not have Alice's contribution, the expected correlation coefficient is zero, and the Z statistic will approximately follow a Gaussian distribution with a zero mean and a unit variance. When the test signal has Alice's contribution, the Z statistic will have a large positive mean determined by the correlation coefficient ρ. To derive the expression for ρ, we define a random variable $Y \triangleq W + D$ and the extracted watermark consists of i.i.d. samples from Y. Here W is a zero-mean Gaussian random variable representing Alice's watermark, and N is a zero-mean Gaussian random variable representing noise. The correlation coefficient of this bivariate normal population (W, Y) is

$$\rho = \frac{\operatorname{cov}(W, Y)}{\sqrt{\operatorname{Var}(W) \cdot \operatorname{Var}(Y)}} \tag{2.11}$$

$$= \frac{\operatorname{Var}(W) + \operatorname{cov}(W, D)}{\sqrt{\operatorname{Var}(W)\left[\operatorname{Var}(W) + \operatorname{Var}(D) + 2\operatorname{cov}(W, D)\right]}}. \tag{2.12}$$

When W and N are uncorrelated, the correlation coefficient becomes

$$\rho = \sqrt{\frac{\operatorname{Var}(W)}{\operatorname{Var}(W) + \operatorname{Var}(D)}} = \sqrt{\frac{1}{1 + 1/\mathrm{WNR}}}, \tag{2.13}$$

where the watermark-to-noise ratio $\mathrm{WNR} \triangleq \operatorname{Var}(W)/\operatorname{Var}(D)$. In Figure 2.5a, we plot the mean of the Z statistic computed using the derived ρ for different WNR and L.

Now knowing the distribution of the Z statistic under the presence and absence of Alice's watermark, we can compute the probabilities of detection P_d and false alarm P_{fa} for different decision thresholds. A threshold of 3 gives a false alarm probability of about 10^{-3}, while a threshold of 6 corresponds to 10^{-9}. The tradeoff between P_d and P_{fa} can be visualized in terms of receiver operating characteristic (ROC) curves, as shown in Figure 2.5b for different L at a fixed WNR and Figure 2.5c for different WNR at a fixed L.

Denoting the average values of the components in $\mathbf{y}$ and $\mathbf{s}$ as $\tilde{\mu}$ and μ, respectively, we compute the sample correlation coefficient r between $\mathbf{y}$ and $\mathbf{s}$ by

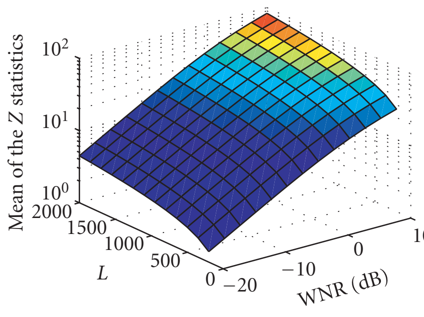
$$r = \frac{\sum_{i=1}^{L} (y_i - \tilde{\mu})(s_i - \mu)}{\sqrt{\sum_{j=1}^{L} (y_j - \tilde{\mu})^2 \sum_{k=1}^{L} (s_k - \mu)^2}}. \tag{2.14}$$

When the noise introduced by attacks does not follow a Gaussian distribution and/or the noise samples are mutually correlated, the Z statistics with the true traitors may be different from the unit-variance Gaussian distribution, but experimental studies in watermarking and fingerprinting scenarios show that the difference is small. The actual distribution of the Z statistics and the detection performance in terms of the detection probability and the false alarm probability can be measured through experiments.

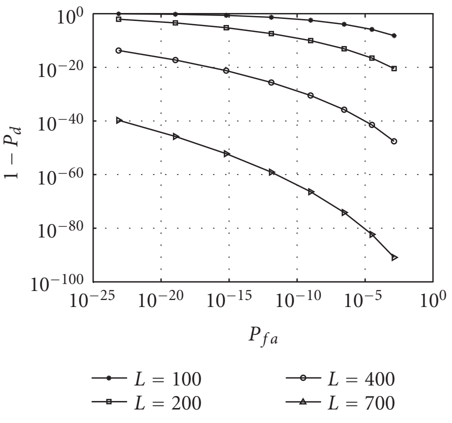
q statistic

We have seen that the correlation $\langle Y, W \rangle$ consists of the sum of the product between the corresponding components from the test signal Y_i and the watermark signal W_i. Treating the product as a random variable, we can arrive at the third way of normalizing the variance, which we will call as the *q statistic* [36]:

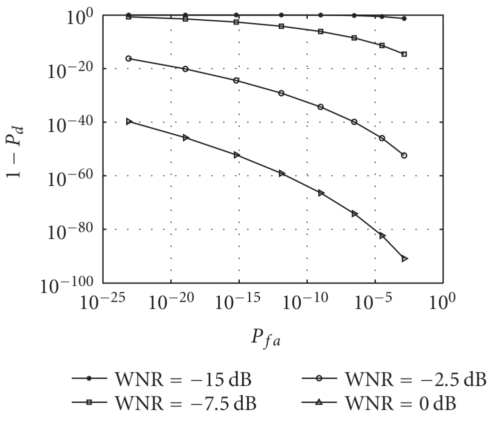
$$q^{(i)} = \frac{\sqrt{N} M_y}{\sqrt{V_y^2}}, \tag{2.15}$$



(a)



(b)



(c)

FIGURE 2.5. Z statistics and ROC curves: (a) mean of Z statistics for different WNR and L, (b) ROC curves for different L at WNR = 0 dB, and (c) ROC curves for different WNR at $L = 700$.

where

$$M_y = \sum_{j=1}^{N} \frac{Y_j W_j^{(i)}}{N}, \qquad V_y^2 = \sum_{j=1}^{N} \frac{(Y_j W_j^{(i)} - M_y)^2}{N-1}. \tag{2.16}$$

It is easy to see that M_y and V_y^2 are the sample mean and sample variance of $\{Y_j W_j^{(i)}\}$.

The three detection statistics introduced so far, namely, the T_N, Z, and q, all have the correlation between the extracted watermark $\mathbf{Y}$ and the original watermark $\mathbf{W}^{(i)}$ as the kernel term. They differ primarily in the way of normalization. In practice, the noise introduced by attacks and other processing, along with the interference from the host signal if it is not available in the detection, may substantially alter the distribution and make it deviate from zero-mean Gaussian distribution, for example, the mean can be shifted from zero and the distribution could become bimodal. As a result, the three statistics may exhibit different performance and additional preprocessing may be needed. We will elaborate more on this issue in the context of nonlinear collusion in Chapter 3.

2.2.5. Exploiting human visual properties

Properties of the human visual system (HVS) play a crucial role to simultaneously ensure the robustness and imperceptibility of the embedded data [23, 24, 63]. In a classic paper on spread-spectrum watermarking, Cox et al. [23] 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. Rules and quantitative models of HVS can be used to determine the just-noticeable-difference (JND) for each feature to be watermarked (which can be a pixel, a coefficient, or other quantities chosen by the embedding algorithm). The JND can be used to locally adjust the strength of the watermark. It can also help us determine which feature can be changed and which cannot. For example, we often consider features that already have very small values as unembeddable to avoid introducing excessive relative changes in those features.

In the implementation in the initial work by Cox et al. [23], 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. More specifically, each of n watermarked coefficients $\{v_i'\}$ is generated from the original unmarked one $\{v_i\}$ by

$$v_i' = v_i + \alpha_i \cdot w_i \tag{2.17}$$

with the watermark strength $\alpha_i = 0.1 \cdot |v_i|$. By more explicitly utilizing human visual models known as frequency-domain masking, Podilchuck and Zeng [24]

AC$_{01}$ Horizontal frequency AC$_{07}$

DC

Vertical frequency

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

AC$_{70}$ AC$_{77}$

FIGURE 2.6. Standard table in JPEG image compression consisting of quantization step size for each DCT coefficient in an 8×8 block.

and Swanson et al. [63] embed watermarks in block-DCT domain and use masking models to tune the watermark strength in each block. The masking model is based on the following observations of the human visual system: first, different frequency bands have different just-noticeable levels, and generally the 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 nonsmooth blocks obtained by this model is generally larger than that of the smooth ones. Similar masking effect in the spatial domain has also been explored [63].

It is worth mentioning that the block-DCT domain is a popular embedding domain in literature, for 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 the human visual model for JPEG compression [64, 65]). For example, a simple approach to approximate the JND for each coefficient in an 8×8 block is to use half of the size of quantization table used in moderate JPEG compression. Figure 2.6 shows the reference table suggested in the JPEG standard (corresponding to quality factor 50%) [66, 67]. We can also scale the table for more or less stringent control on perceptual distortion, or use it as a baseline and perform further adjustment according to local image characteristics. This latter aspect is another advantage for choosing block-based domain in embedding, allowing for fine-tuning watermark strength at each local region to achieve a good tradeoff between imperceptibility and robustness against distortion. A main perceptual problem with block-DCT domain embedding is the ringing artifacts introduced on edges when the high-valued mid-frequency coefficients are modified substantially during the embedding process. This problem can be mitigated through distinguishing edge and texture blocks so as to refine the JND and prevent the features in the edge blocks from being over-modified [18, 68].

2.3. Employing spread-spectrum embedding in fingerprinting

A straightforward way of applying spread-spectrum watermarking to fingerprinting is to use mutually orthogonal watermarks as fingerprints to identify each user [69, 70]. In practical implementation, the orthogonality may be approximated by using random number generators to produce independent watermark signals for different users. The orthogonality allows for distinguishing the fingerprints to the maximum extent. The simplicity of encoding and embedding orthogonal fingerprints makes them attractive to identification applications that involve a small group of users. One drawback of orthogonal fingerprints is that the amount of fingerprints is limited by the amount of orthogonal signals that can be created. Building nonorthogonal fingerprints becomes necessary in such practical scenarios as supporting more fingerprints than the dimensionality of the feature signal or limiting the number of orthogonal basis signals for reducing computation and bookkeeping costs.

A second option for using spread-spectrum watermarking is to employ code modulation. As we will discuss further in Chapter 6, code modulation allows fingerprint designers to design more fingerprints for a given fingerprint dimensionality by constructing each user's fingerprint signal as a linear combination of orthogonal noise-like basis signals. Code modulation has the potential to provide a compact way to represent fingerprints. As we will see, systematically constructing them with high tracing capability and collusion resistance for multimedia data is a challenging research issue.

One important application of fingerprinting is identifying a user who is redistributing marked content $\mathbf{y}_j$ by detecting the watermark associated with the user to whom $\mathbf{y}_j$ was sold. By identifying a user, the content owner may be able to more closely monitor future actions of that user, or gather evidence supporting that user's illicit usage of the content. There are two different detection strategies that might arise in fingerprinting applications. They are differentiated by the presence or lack of the original content in the detection process. We will refer to *nonblind* detection as the process of detecting the embedded watermarks with the assistance of the original content $\mathbf{x}$, and refer to *blind* detection as the process of detecting the embedded watermarks without the knowledge of the original content $\mathbf{x}$. Nonblind fingerprint detection requires that the entity performing detection first identify the original version corresponding to the test image from a database of unmarked original images. This database can often be very large and requires considerable storage resources. In the nonblind fingerprint detection, the distortion can be modeled as $\mathbf{d} = \mathbf{z}$. Blind detection, on the other hand, offers more flexibility in detection, such as distributed detection scenarios. It does not require vast storage resources, and does not have the computational burden associated with image registration from a large database. This is particularly attractive for enabling fingerprint detection by distributed verification engines. However, unlike the nonblind detection scenario, in the blind detection scenario, the host signal is unknown to the detector and often serves as a noise source that hinders the ability to detect the watermark. In this case, the distortion can be modeled as $\mathbf{d} = \mathbf{x} + \mathbf{z}$.

Note that as we have reviewed in the previous section, there are other types of watermarking schemes that do not suffer from interference from unknown host signals [18, 39]. Many of these schemes can be viewed as selecting between the indices of a few alternative quantizers. Unlike the additive spread-spectrum embedding, quantization-based embedding offers limited capability to simultaneously embed different signals over the same signal segment. As such, the effect of collusion would be largely influenced by the upper-level coding, which can have limited collusion resistance for multimedia applications compared with a joint coding and embedding framework built upon spread-spectrum embedding [71]. The appropriateness of quantization-based embedding for fingerprinting and its anticollusion capabilities remain to be further investigated. In this book, we will focus on multimedia fingerprinting built upon the robust spread-spectrum embedding. We will look at two paradigms, namely, a signal processing centered approach and a joint coding and embedding approach.

For concept-proving purposes, we consider the simple noise model of independent Gaussian noise and use the correlator with normalized noise variance as described in (2.6). This simplification allows us to focus on the unique issues of fingerprint encoding and colluder detection for the anticollusion fingerprinting problem. From a layered viewpoint on data hiding systems [18], the modules of fingerprint encoding and colluder detection are built on top of the modules of one-bit watermark embedding and detection. The design and optimization of the former and latter modules have some degree of independence in system development. We can replace the simplified model used here with more sophisticated single-watermark detectors considering more realistic noise such as those in [56, 57] to improve the performance in the watermark detection layer and in turn enhance the overall performance.

3 Collusion attacks

Conventional embedding and watermarking techniques are typically concerned with robustness against a variety of attacks mounted by an individual. However, protecting the sanctity of digital fingerprints is no longer a traditional security issue with a single adversary. The global nature of the Internet has not only brought media closer to the consumers, but it has also brought adversaries closer to the media. It is now easy for a group of users with differently marked versions of the same content to come together and work together to mount attacks against the fingerprints. These attacks, known as collusion attacks, provide a cost-effective method for removing an identifying fingerprint and poses a significant threat to multimedia fingerprinting. For an improperly designed fingerprint, it is possible to gather a small coalition of colluders and sufficiently attenuate each of the colluders' identifying fingerprints to produce a new version of the content with no detectable traces. Thus, to design fingerprints that can resist collusion and identify the colluders, it is important to first model and analyze collusion and understand this new challenge in multimedia fingerprinting.

There are several types of collusion attacks that may be used against multimedia fingerprints. One method is simply to synchronize the media signals and average them, which is an example of the linear collusion attack. Another collusion attack, referred to as the copy-and-paste attack, involves users cutting out portions of each of their media signals and pasting them together to form a new signal. Other attacks may employ nonlinear operations, such as taking the maximum or median of the values of corresponding components of individual copies.

To uncover the underlying complexities governing the effect of nonlinear collusion attacks, this chapter conducts both analytical and experimental studies on the behavior of nonlinear collusion attacks. This study will serve as a guideline for later chapters where we jointly consider the issue of designing fingerprints, embedding fingerprints, and devising appropriate detection schemes that have the ability to robustly resist a broader spectrum of collusion attacks. We will build upon the discussion about using orthogonal modulation for fingerprinting that was provided in the previous chapter, and will focus our analysis of nonlinear collusion exclusively on orthogonal multimedia fingerprints. The analysis that we provide can be extended to code-modulated and other correlated fingerprint schemes.

In this chapter, we first take the role of the attackers and examine different types of collusion attacks that may be employed in an attempt to remove the fingerprints and make it difficult for the information protector to trace content leakage. We then shift our role to designer/detector and analyze the performance of several commonly used detection statistics [36, 54, 62] in the literature under collusion attacks. The analysis of different detection statistics provides a guideline for the selection of the detector in a multimedia forensic system.

3.1. Introduction to collusion attacks

3.1.1. Linear collusion attacks

Linear collusion is one of the most feasible collusion attacks that may be employed against multimedia fingerprints. Given several differently marked copies of the same content, the colluders linearly combine all the copies to generate a colluded copy. In a K-colluder linear collusion attack, the fingerprinted signals $\{\mathbf{y}_i\}$ are combined according to $\sum_{i=1}^{K} \lambda_i \mathbf{y}_i$, where the weights $\{\lambda_i\}$ satisfy $\sum_{i=1}^{K} \lambda_i = 1$ to maintain the average intensity of the original multimedia signal. With orthogonal fingerprinting, for colluder i who is assigned the weight λ_i, the energy of his fingerprint is reduced by a factor of λ_i^2. When λ_i is smaller, the trace of colluder i's fingerprint is weaker and colluder i is less likely to be caught by the detector.

In multiuser collusion, since no colluder would like to take more of a risk than any other colluder, they usually agree to distribute the risk of being detected evenly among themselves and apply fair collusion. A simple way to achieve the fairness of collusion is to average all the fingerprinted signals with an equal weight for each user and let $\lambda_i = 1/K$ for all i. Figure 3.1 shows an example of collusion by averaging with three colluders and all three fingerprints are averaged with the same weight 1/3.

In [69], the collusion attack was modeled by averaging the fingerprinted signals followed by an additive noise to further hinder the detection. Their work showed that $O(N/\log N)$ adversaries are sufficient to defeat the underlying watermarks, where N is the total length of the fingerprint. Similar results were also presented in [70]. In [72], a more general linear attack was considered, where the colluders employ multiple-input single-output linear shift-invariant (LSI) filtering plus additive Gaussian noise to thwart the fingerprints. Under the assumption that all fingerprints are independent and have identical statistical characteristics, it was shown that the optimal LSI attack involves each user weighting their marked document equally prior to the addition of additive noise.

Another type of fair collusion, referred to as the cut-and-paste attack, involves users cutting out portions of each of their media signals and pasting them together to form a new signal. Figure 3.2 shows an example of the cut-and-paste attack with two colluders: Alice and Chris. The colluded copy in Figure 3.2 is generated

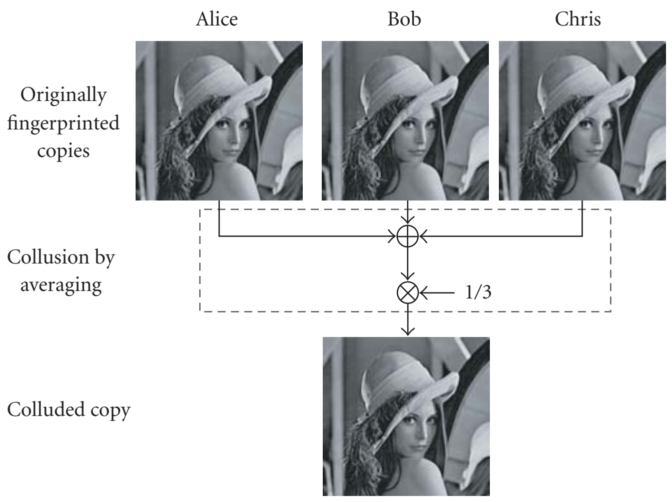


FIGURE 3.1. Collusion by averaging.

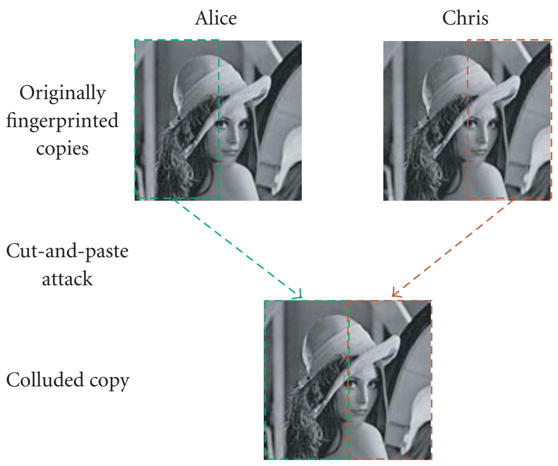


FIGURE 3.2. Collusion by cut and paste.

by copying the left half of Alice's fingerprinted signal and taking the right half of Chris' copy.

When the fingerprint is spread throughout the entire host signal by such techniques as spread-spectrum embedding and detected through some form of correlation processing, the cut-and-paste collusion attack has an effect that is similar to averaging collusion. In particular, in both cases, the energy of each contributing fingerprint is reduced by a factor corresponding to the amount of copies involved in the collusion. As an example, if Alice contributes half of her samples to a cut-and-paste collusion, the energy of Alice's fingerprint in the colluded copy is only

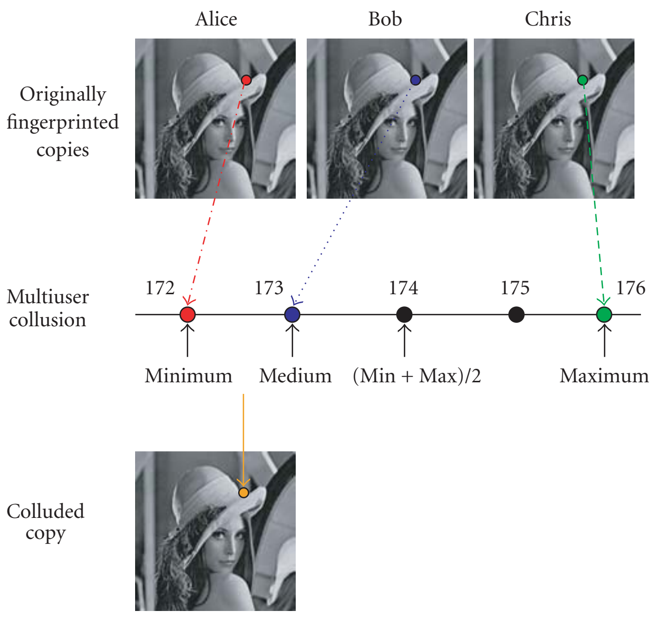


Figure 3.3. Examples of nonlinear collusion attacks.

half of her overall fingerprint energy. Therefore, in terms of the effect on the fingerprint energy reduction and the impact on the probability of being detected, we may consider cut-and-paste collusion analogous to average-based collusion when considering spread-spectrum embedding.

3.1.2. Nonlinear collusion attacks

Linear collusion by averaging is a simple and effective way for a coalition of users to attenuate embedded fingerprints. Averaging, however, is not the only form of collusion attacks available to a coalition of adversaries. In fact, for each component of the multimedia signal, the colluders can output any value between the minimum and maximum corresponding values, and have high confidence that the spurious value they get will be within the range of the just-noticeable-different since each fingerprinted copy is expected to have high perceptual quality.

An important class of nonlinear collusion attacks is based upon such operations as taking the maximum, minimum, and median of corresponding components of the colluders' fingerprinted copies. Figure 3.3 shows examples of different types of nonlinear collusion and their effects with three colluders, Alice, Bob, and Chris. For one pixel at the *n*th row and the *m*th column in the image, assume that it takes the values 172, 173, and 176 in the three copies corresponding to the three colluders. When generating the colluded copy, for the pixel at row *n* and column *m*, the colluders can take the minimum of the three values, which gives 172; and they can also use the maximum or the medium of the corresponding pixels in the three copies, which are 176 and 173, respectively. During collusion, the colluders can also combine these basic operations to generate a colluded copy. As an example, for that pixel at row *n* and column *m* in the colluded copy, the colluders can take the average of the maximum and the minimum, which is 174. The colluders repeat this process for every pixel in the image and generate the colluded copy.

A few nonlinear attacks were studied in [62]. For uniformly distributed fingerprints, nonlinear collusion attacks were shown to defeat the fingerprinting system more effectively than the averaging collusion [62]. Simulation results in [62] also showed that normally distributed fingerprints are more robust against nonlinear collusion attacks than uniform fingerprints, but an analytical study on the Gaussian fingerprints' performance was not provided. In addition to the robustness against collusion attacks, when compared with discrete watermarks and uniform watermarks, Gaussian watermarks have the advantage that they do not provide the attackers with the positions and the amplitudes of the embedded watermarks under statistical and histogram attacks [73]. Therefore, to improve the robustness of the embedded fingerprints against collusion as well as statistical and histogram attacks, Gaussian distributed fingerprints should be used in multimedia fingerprinting systems, and it is important to provide both analytic and experimental studies on the behavior of nonlinear attacks on Gaussian fingerprints.

3.2. Introduction to order statistics

Before jumping into the discussion on nonlinear collusion attacks, we will first introduce order statistics and review the results from that area that enables the statistical analysis of nonlinear collusion.

The research on order statistics focuses on properties and applications of ordered random variables and their functions. If n random variables $X_1, X_2, \dots, X_n$ are sorted in ascending order of the magnitude and labeled as follows:

$$X_{1:n} \le X_{2:n} \le \cdots \le X_{n:n}, \tag{3.1}$$

then $X_{i:n}$ is called the ith-order statistics in a sample of size n [74].

We are particularly interested in the minimum

$$X_{\min} \triangleq X_{1:n} = \min\left(X_1, X_2, \dots, X_n\right) \tag{3.2}$$

which returns the smallest value of $X_1, X_2, \dots, X_n$; the maximum

$$X_{\max} \triangleq X_{n:n} = \max\left(X_1, X_2, \dots, X_n\right) \tag{3.3}$$

which returns the largest value of the n random variables; and the median

$$X_{\text{med}} \triangleq \begin{cases} X_{(n+1)/2:n} & \text{if } n \text{ is odd}, \\ \dfrac{1}{2}\left(X_{n/2:n} + X_{(n/2)+1:n}\right) & \text{if } n \text{ is even}. \end{cases} \tag{3.4}$$

3.2.1. Distribution of order statistics

Assume that $X_1, X_2, \dots, X_n$ are i.i.d. (independent and identically distributed) random variables and they have probability density function (pdf) $f(x)$ and cumulative distribution function (cdf) $F(x)$. Let $F_{\min}(x) \triangleq P[X_{\min} \leq x]$ denote the cumulative distribution function of $X_{\min}$, and we have

$$\begin{aligned} F_{\min}(x) &= P[\min(X_1, X_2, \dots, X_n) \leq x] \\ &= 1 - P[\min(X_1, X_2, \dots, X_n) > x] \\ &= 1 - P[X_1 > x, X_2 > x, \dots, X_n > x]. \end{aligned} \tag{3.5}$$

Since $X_1, \dots, X_n$ are independently and identically distributed,

$$F_{\min}(x) = 1 - \prod_{i=1}^{n} P[X_i > x] = 1 - [1 - F(x)]^n. \tag{3.6}$$

The probability density function of $X_{\min}$ is

$$f_{\min}(x) = \frac{dF_{\min}(x)}{dx} = nf(x)[1 - F(x)]^{n-1}. \tag{3.7}$$

In general, the probability density function of the ith-order statistics is

$$f_{X_{i:n}}(x) = \frac{n!}{(i-1)!(n-i)!}[F(x)]^{i-1}[1 - F(x)]^{n-i} f(x), \tag{3.8}$$

for $i = 1, 2, \dots, n$ [74].

3.2.2. Joint distribution of two different order statistics

Let us define $F_{X_{\min}, X_{\max}}(x', x'') \triangleq P[X_{\min} \leq x',\ X_{\max} \leq x'']$ as the joint cumulative distribution function (jcdf) of $X_{\min}$ and $X_{\max}$. When $x' < x''$,

$$\begin{aligned} &P[X_{\min} \leq x',\ X_{\max} \leq x''] \\ &\quad = P[X_{\max} \leq x''] - P[X_{\min} > x',\ X_{\max} \leq x''] \\ &\quad = P[X_1 \leq x'', \dots, X_n \leq x''] - P[x' < X_1 \leq x'', \dots, x' < X_n \leq x''] \\ &\quad = [F(x'')]^n - [F(x'') - F(x')]^n. \end{aligned} \tag{3.9}$$

When $x' \geq x''$,

$$P[X_{\min} \leq x',\ X_{\max} \leq x''] = P[X_{\max} \leq x''] = [F(x'')]^n. \tag{3.10}$$

Therefore, from (3.9) and (3.10),

$$F_{X_{\min},X_{\max}}(x',x'') = \begin{cases} [F(x'')]^n - [F(x'') - F(x')]^n & \text{if } x' < x'', \\ [F(x'')]^n & \text{otherwise,} \end{cases} \tag{3.11}$$

and the joint probability density function (jpdf) of $X_{\min}$ and $X_{\max}$ is

$$\begin{aligned} f_{X_{\min},X_{\max}}(x',x'') &= \frac{\partial F_{X_{\min},X_{\max}}(x',x'')}{\partial x' \partial x''} \\ &= \begin{cases} n(n-1)f(x')f(x'')[F(x'') - F(x')]^{n-2} & \text{if } x' < x'', \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{3.12}$$

In general, for the ith-order statistics $X_{i:n}$ and the jth-order statistics $X_{j:n}$, where $1 \le i < j \le n$, their joint probability density function is

$$\begin{aligned} f_{X_{i:n},X_{j:n}}(x',x'') = {}& \frac{n!}{(i-1)!(j-i-1)!(n-j)!}[F(x')]^{i-1}[F(x'') - F(x')]^{j-i-1} \\ & \times [1 - F(x'')]^{n-j} f(x') f(x'') \end{aligned} \tag{3.13}$$

for $x' < x''$. Detailed derivation is available in [74].

3.2.3. Joint distribution of order statistics and the unordered random variables

In this section, we consider the joint probability density function of the order statistics $X_{j:n}$ and the unordered random variable X_i.

Let us define $f_{X_{\min},X_i}(x',x)$ as the joint probability density function of $X_{\min}$ and X_i. From [74], $f_{X_{\min},X_i}(x',x)$ breaks up into three nonoverlapping regions $x > x'$, $x = x'$, and $x < x'$:

$$f_{X_{\min},X_i}(x',x) = \begin{cases} (n-1)[1 - F(x')]^{n-2} f(x) f(x') & \text{if } x > x', \\ [1 - F(x')]^{n-1} f(x') & \text{if } x = x', \\ 0 & \text{otherwise.} \end{cases} \tag{3.14}$$

Similarly, the joint density function between $X_{\max}$ and X_i is

$$f_{X_{\max},X_i}(x',x) = \begin{cases} (n-1)[F(x')]^{n-2} f(x) f(x') & \text{if } x < x', \\ [F(x')]^{n-1} f(x') & \text{if } x = x', \\ 0 & \text{otherwise.} \end{cases} \tag{3.15}$$

For $X_{j:n}$ and X_i where $1 < j < n$, their joint density function is

$$f_{X_{j:n},X_i}(x',x) = \begin{cases} (n-1)\binom{n-2}{j-2}[F(x')]^{j-2}[1-F(x')]^{n-j}f(x)f(x') & \text{if } x < x', \\ \binom{n-2}{j-1}[F(x')]^{j-1}[1-F(x')]^{n-j}f(x') & \text{if } x = x', \\ (n-1)\binom{n-2}{j-1}[F(x')]^{j-1}[1-F(x')]^{n-j-1}f(x)f(x') & \text{if } x > x'. \end{cases} \tag{3.16}$$

For $X_{\min}$, $X_{\max}$, and X_i, by extending (3.14) and (3.15), their joint probability density function is

$$f_{X_{\min},X_{\max},X_i}(x',x'',x) = \begin{cases} (n-1)f(x')f(x'')[F(x'')-F(x')]^{n-2} & \text{if } x = x', \\ (n-1)f(x')f(x'')[F(x'')-F(x')]^{n-2} & \text{if } x = x'', \\ (n-1)(n-2)f(x')f(x'')f(x)[F(x')-F(x'')]^{n-3} & \text{if } x' < x < x'', \\ 0 & \text{otherwise.} \end{cases} \tag{3.17}$$

When n is odd and $l = (n-1)/2$ is an integer, the joint density function of $X_{\min}$, X_{med}, and X_i is

$$f_{X_{\min},X_{\text{med}},X_i}(x',x'',x) = \begin{cases} 2l\binom{2l-1}{l}f(x')f(x'')[F(x'')-F(x')]^{l-1}[1-F(x'')]^{l} & \text{if } x = x', \\ 2l\binom{2l-1}{l}f(x')f(x'')[F(x'')-F(x')]^{l-1}[1-F(x'')]^{l} & \text{if } x = x'', \\ 2l(2l-1)\binom{2l-2}{l}f(x')f(x'')f(x)[F(x'')-F(x')]^{l-2}[1-F(x'')]^{l} & \text{if } x' < x < x'', \\ 2l(2l-1)\binom{2l-2}{l-1}f(x')f(x'')f(x)[F(x'')-F(x')]^{l-1}[1-F(x'')]^{l-1} & \text{if } x > x'', \\ 0 & \text{otherwise.} \end{cases} \tag{3.18}$$

When n is odd and $l = (n-1)/2$ is an integer, the joint density function of X_{med}, $X_{\max}$, and X_i is

$$
f_{X_{\text{med}},X_{\max},X_i}(x',x'',x)
= \begin{cases}
(n-1)\binom{2l-1}{l} f(x')f(x'')\big[F(x'')-F(x')\big]^{l-1}F^{l}(x'') & \text{if } x = x', \\
(n-1)\binom{2l-1}{l} f(w')f(w'')\big[F(w'')-F(w')\big]^{l-1}F^{l}(w'') & \text{if } x = x'', \\
(n-1)(n-2)\binom{2l-2}{l-1} f(x')f(x'')f(x)\big[F(x'')-F(x')\big]^{l-1}F^{l-1}(x'') & \text{if } x < x', \\
(n-1)(n-2)\binom{2l-2}{l} f(x')f(x'')f(x)\big[F(x'')-F(x')\big]^{l-2}F^{l}(x'') & \text{if } x' < x < x'', \\
0 & \text{otherwise.}
\end{cases}
\tag{3.19}
$$

3.3. Multimedia fingerprinting system model

3.3.1. Fingerprinting systems and collusion attacks

If we look at both the digital fingerprinting process and the collusion attack process collectively, then the complete system may be viewed as consisting of three main parts: fingerprint embedding, collusion attacks, and fingerprint detection. We now look at each of these components individually.

Fingerprint embedding. At the content owner's side, for each user in the system, he generates a unique fingerprint of the same size as the host signal. Due to its robustness against many attacks by a single adversary, spread-spectrum embedding [23, 24] is applied to hide fingerprints in the host signal and human visual models are used to guarantee the imperceptibility of the embedded fingerprints. Finally, the content owner distributes the fingerprinted copies to the corresponding users.

Assume that there are a total of M users in the system. Given a host signal represented by a vector $\mathbf{x}$ of length N, assume that $\mathbf{s}_i$ is the fingerprint for the ith user where $i = 1, 2, \ldots, M$, and it has length N. In orthogonal fingerprint modulation, the M fingerprints are generated independently. The fingerprinted copy $\mathbf{y}_i$ that is distributed to the ith user is generated by

$$
\mathbf{y}_i(j) = \mathbf{x}(j) + \text{JND}(j) \cdot \mathbf{s}_i(j),
\tag{3.20}
$$

where $\mathbf{y}_i(j)$, $\mathbf{x}(j)$, and $\mathbf{s}_i(j)$ are the jth components of the fingerprinted copy, the original signal, and the fingerprint, respectively. JND here is the *just-noticeable-difference* from human visual models [24] to control the energy of the embedded fingerprints so as to ensure their imperceptibility.

Collusion attacks. To examine families of nonlinear collusion, the averaging-based collusion attack is used as the benchmark to measure the effectiveness of collusion. The set of typical nonlinear collusion that are considered includes

(i) *minimum/maximum/median attack*: under these attacks, the colluders create an attacked signal, in which each component is the minimum, maximum, and median, respectively, of the corresponding components of the fingerprinted signals associated with the colluders;

(ii) *minmax attack*: each component of the attacked signal is the average of the maximum and minimum of the corresponding components of the fingerprinted signals;

(iii) *modified negative attack*: each component of the attacked signal is the difference between the median and the sum of the maximum and minimum of the corresponding components of the fingerprinted signals;

(iv) *randomized negative attack*: each component of the attacked signal takes the value of the maximum of the corresponding components of the fingerprinted signals with probability p, and takes the minimum with probability $1 - p$.

In order to make it easier to acquire analytical insight, we typically assume that the nonlinear collusion attacks are performed in the same domain of features as the fingerprint embedding process. Further, we note that it is possible to evaluate the performance for these attacks when the attack domain and the embedding domain differ by performing experimental studies.

Assume that K out of M users collude and $S_C = \{i_1, i_2, \ldots, i_K\}$ is the set containing the indices of the colluders. The fingerprinted copies that are received by these K colluders are $\{\mathbf{y}_k\}_{k\in S_C}$. The colluders generate the jth component of the attacked copy $\mathbf{y}(j)$ using one of the following collusion functions:

$$\begin{aligned}
\text{average attack} : \mathbf{y}_{\text{ave}}(j) &= \sum_{k\in S_C} \frac{\mathbf{y}_k(j)}{K},\\
\text{minimum attack} : \mathbf{y}_{\min}(j) &= \min\left(\{\mathbf{y}_k(j)\}_{k\in S_C}\right),\\
\text{maximum attack} : \mathbf{y}_{\max}(j) &= \max\left(\{\mathbf{y}_k(j)\}_{k\in S_C}\right),\\
\text{median attack} : \mathbf{y}_{\text{med}}(j) &= \text{med}\left(\{\mathbf{y}_k(j)\}_{k\in S_C}\right),\\
\text{minmax attack} : \mathbf{y}_{\text{MinMax}}(j) &= \frac{\mathbf{y}_{\min}(j) + \mathbf{y}_{\max}(j)}{2},\\
\text{modified negative attack} : \mathbf{y}_{\text{ModNeg}}(j) &= \mathbf{y}_{\min}(j) + \mathbf{y}_{\max}(j) - \mathbf{y}_{\text{med}}(j),\\
\text{randomized negative attack} : \mathbf{y}_{\text{RandNeg}}(j) &= \begin{cases} \mathbf{y}_{\min}(j) & \text{with prob. } p,\\ \mathbf{y}_{\max}(j) & \text{with prob. } 1-p. \end{cases}
\end{aligned} \tag{3.21}$$

In (3.21), $\min(\{\mathbf{y}_k(j)\}_{k\in S_C})$, $\max(\{\mathbf{y}_k(j)\}_{k\in S_C})$, and med $(\{\mathbf{y}_k(j)\}_{k\in S_C})$ return the minimum, the maximum, and the median values of $\{\mathbf{y}_k(j)\}_{k\in S_C}$, respectively. The colluded copy is $\mathbf{y} = [\mathbf{y}(1), \mathbf{y}(2), \dots, \mathbf{y}(N)]$. For the fingerprint embedding and collusion attack model in this section, applying the collusion attacks to the fingerprinted copies is equivalent to applying the collusion attacks to the embedded fingerprints. For example,

$$\begin{aligned} \mathbf{y}_{\min}(j) &= \min\left(\{\mathbf{y}_k(j) + \mathrm{JND}(j)\cdot \mathbf{s}_k(j)\}_{k\in S_C}\right) \\ &= \mathbf{x}_j + \mathrm{JND}_j \cdot \min\left(\{\mathbf{s}_k(j)\}_{k\in S_C}\right). \end{aligned} \tag{3.22}$$

Fingerprint detection and colluder identification. In many fingerprinting applications, the original signal is often available at the detector, and therefore, a nonblind detection scenario is feasible. Since a nonblind detection process operates at a higher effective watermark-to-noise ratio, it is preferable for the colluder detection scheme to use nonblind detection whenever possible. The nonblind detector first removes the host signal from the test copy before colluder identification. Then, it extracts the fingerprint from the test copy, measures the similarity between the extracted fingerprint and each of the original fingerprints, compares with a threshold, and outputs the estimated identities of the colluders.

Under the nonlinear collusion attacks in (3.21), the extracted fingerprint is

$$\mathbf{w} = g\left(\{\mathbf{s}_k\}_{k\in S_C}\right), \tag{3.23}$$

where $g(\cdot)$ is a collusion function defined in (3.21). To test the presence of the original fingerprint $\mathbf{s}_i$ in the extracted fingerprint $\mathbf{w}$, we use the three detection statistics discussed in Chapter 2: T_N, z, and q. Note that all three detection statistics are correlation-based in which the correlation between the extracted fingerprint $\mathbf{w}$ and the original fingerprint $\mathbf{s}_i$ is the kernel term, and they differ primarily in the way of normalization.

3.3.2. Performance criteria

To measure the effectiveness of collusion attacks in removing multimedia fingerprints and study the performance of detection statistics under collusion, the following set of criteria are used:

(i) P_d: the probability of capturing at least one colluder,

(ii) P_{fp}: the probability of falsely accusing at least one innocent user,

(iii) γ_c: the fraction of colluders that are successfully captured,

(iv) γ_i: the fraction of innocent users that are falsely accused.

Different sets of criteria address different scenarios in multimedia fingerprinting and emphasize different requirements of various applications. We will discuss different scenarios of digital fingerprinting and the corresponding performance criteria in detail in Chapter 4.

When considering the perceptual quality of a fingerprinted copy or the colluded copy, one of the commonly used objective measurements on perceptual distortion is the mean square error (MSE) and equivalently the PSNR for image applications. A major weakness of MSE is that it ignores the unique characteristic of multimedia data: minor perturbations on the data values will not cause noticeable distortion as long as they do not exceed the *just-noticeable-difference* [24]. Furthermore, MSE only measures the average energy of the noise introduced and does not consider the local constraints on each noise component.

For a colluded copy $\mathbf{y}$, assume that $\mathbf{n} = \mathbf{y} - \mathbf{x}$ is the distortion introduced by collusion when compared with the host signal $\mathbf{x}$. Taking JND into consideration, two new measurements were defined in [60]:

(i) $F_{\text{JND}} \triangleq \sum_{j=1}^{N} I_{[|n(j)|>\text{JND}(j)]}/N$, and

(ii) the redefined mean square error $\text{MSE}_{\text{JND}} \triangleq \sum_{j=1}^{N} n(j)'^2$, where n'_j is defined as

$$n'_j = \begin{cases} n(j) + \text{JND}(j) & \text{if } n(j) < -\text{JND}(j), \\ 0 & \text{if } -\text{JND}(j) \le n_j \le \text{JND}(j), \\ n(j) - \text{JND}(j) & \text{if } n(j) > \text{JND}(j). \end{cases} \tag{3.24}$$

MSE_{JND} calculates the power of the noise components that introduce perceptual distortion and F_{JND} reflects the percentage of the noise components that exceed JND. A large MSE_{JND} or a large F_{JND} indicates large perceptual distortion introduced.

3.4. Statistical analysis of collusion attacks

This section analyzes the statistical behavior of the detection statistics under different collusion attacks.

3.4.1. Analysis of collusion attacks

As discussed in Section 3.3, when measuring the similarity between the extracted fingerprint $\mathbf{w}$ and the original fingerprint $\mathbf{s}_i$, all three statistics are correlation based, and the common kernel term is the linear correlation:

$$T'_N(i) \triangleq \frac{1}{N}\langle \mathbf{w}, \mathbf{s}_i\rangle = \frac{1}{N}\sum_{j=1}^{N} g\Big(\{\mathbf{y}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j), \tag{3.25}$$

where N is the length of the fingerprint. For different collusion attacks, $T'_N(i)$ follows different distributions. This section analyzes the statistical behavior of this correlation term under different collusion attacks.

Under the assumption that $\{\mathbf{s}_k(j), k = 1, \dots, M\}_{j=1,\dots,N}$ are i.i.d. distributed with zero mean and variance σ_W^2, $\{g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)\}_{j=1,\dots,N}$ are also i.i.d. distributed. From the central limit theorem, if $\{g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)\}_{j=1,\dots,N}$ have finite mean $\mu_{T'_N(i)}$ and finite variance $\sigma^2_{T'_N(i)}$, then $T'_N(i)$ can be approximated as follows:

$$T'_N(i) \sim \mathcal{N}\left(\mu_{T'_N(i)}, \frac{\sigma^2_{T'_N(i)}}{N}\right). \tag{3.26}$$

Therefore, the problem is reduced to finding $\mu_{T'_N(i)} = E[g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)]$ and $\sigma^2_{T'_N(i)} = \operatorname{var}[g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)]$. For a given K and a given collusion function $g(\cdot)$, due to the symmetry of $g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)$ with respect to the user index i, the terms $\{g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)\}$ for all $i \in S_c$ have the same mean and variance, and similarly, $\{g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)\}$ for all $i \notin S_C$ have the same mean and variance.

For $i \in S_C$, define

$$\begin{aligned}
\mu_{g,H_1} &\triangleq E\Big[g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big], \\
\sigma^2_{g,H_1} &\triangleq \operatorname{var}\Big[g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big] \\
&= E\Big[\Big(g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big)^2\Big] - \big(\mu_{g,H_1}\big)^2.
\end{aligned} \tag{3.27}$$

For $i \notin S_C$, because $\{\mathbf{s}_i(j)\}_{i=1}^M$ are i.i.d. distributed with zero mean and variance σ_W^2, we have

$$\begin{aligned}
\mu_{g,H_0} &\triangleq E\Big[g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big] \\
&= E\Big[g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big)\Big]E[\mathbf{s}_i(j)] = 0, \\
\sigma^2_{g,H_0} &\triangleq \operatorname{var}\Big[g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big] \\
&= E\Big[\Big(g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big) \cdot \mathbf{s}_i(j)\Big)^2\Big] \\
&= E\Big[\Big(g\Big(\{\mathbf{s}_k(j)\}_{k\in S_c}\Big)\Big)^2\Big]\sigma_W^2.
\end{aligned} \tag{3.28}$$

Therefore, $E[g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j)]$, $E[(g(\{\mathbf{s}_k(j)\}_{k\in S_c}) \cdot \mathbf{s}_i(j))^2]$ for $i \in S_C$, and $E[(g(\{\mathbf{s}_k(j)\}_{k\in S_c}))^2]$ are needed for analyzing the correlation term under each collusion attack.

Under the averaging attack, for a guilty colluder $i \in S_C$, since $\{\mathbf{s}_k(j)\}_{k\in S_C}$ are independent of each other with zero mean and variance σ_W^2,

$$
\begin{aligned}
&E\left[\left(\frac{1}{K}\sum_{k\in S_C}\mathbf{s}_k(j)\right)\cdot\mathbf{s}_i(j)\right] \\
&\quad = \frac{1}{K}E\left[\left(\mathbf{s}_i(j)\right)^2\right] + \frac{1}{K}\sum_{k\in S_C, k\neq i} E\left[\mathbf{s}_k(j)\cdot\mathbf{s}_i(j)\right] = \frac{1}{K}\sigma_W^2, \\
&E\left[\left(\frac{1}{K}\sum_{k\in S_C}\mathbf{s}_k(j)\cdot\mathbf{s}_i(j)\right)^2\right] \\
&\quad = \frac{1}{K^2}E\left[\sum_{k\in S_C}\left(\mathbf{s}_k(j)\cdot\mathbf{s}_i(j)\right)^2\right] + \frac{2}{K^2}\sum_{k\neq l, k,l\in S_C} E\left[\mathbf{s}_k(j)\cdot\mathbf{s}_l(j)\cdot\left(\mathbf{s}_i(j)\right)^2\right] \\
&\quad = \frac{1}{K^2}E\left[\left(\mathbf{s}_i(j)\right)^4\right] + \frac{1}{K^2}\sum_{k\in S_C, k\neq i} E\left[\left(\mathbf{s}_k(j)\right)^2\left(\mathbf{s}_i(j)\right)^2\right] \\
&\quad = \frac{1}{K^2}E\left[\left(\mathbf{s}_i(j)\right)^4\right] + \frac{K-1}{K^2}\sigma_W^4.
\end{aligned}
\tag{3.29}
$$

In addition, under the averaging collusion,

$$
\begin{aligned}
&E\left[\left(\frac{1}{K}\sum_{k\in S_C}\mathbf{s}_k(j)\right)^2\right] \\
&\quad = \frac{1}{K^2}E\left[\sum_{k\in S_C}\left(\mathbf{s}_k(j)\right)^2\right] + \frac{2}{K^2}\sum_{k\neq l, k,l\in S_C} E\left[\mathbf{s}_k(j)\mathbf{s}_l(j)\right] = \frac{1}{K}\sigma_W^2.
\end{aligned}
\tag{3.30}
$$

Under the minimum attack, the second moment of $\mathbf{w}_{\min}(j)$ is

$$
E\left[\left(\mathbf{w}_{\min}(j)\right)^2\right] = \int_{-\infty}^{\infty} w^2 f_{\mathbf{w}_{\min}(j)}(w)\,dw,
\tag{3.31}
$$

where $f_{\mathbf{w}_{\min}(j)}(w)$ is the probability density function of $\mathbf{w}_{\min}(j)$. It can be calculated from order statistics, and detailed derivation of $f_{\mathbf{w}_{\min}(j)}(w)$ is in Section 3.2.

For a guilty colluder $i \in S_C$, $f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w)$, the joint pdf of $\mathbf{w}_{\min}(j)$ and $\mathbf{s}_i(j)$, breaks into two nonzero regions: $w > w'$ and $w = w'$ from order statistics [74]. Therefore, given $f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w)$ as shown in (3.14) in Section 3.2, $E[W^{\min}W^{(i)}]$ is equal to

$$
E\left[\mathbf{w}_{\min}(j)\cdot\mathbf{s}_i(j)\right] = E\left[\mathbf{w}_{\min}(j)\cdot\mathbf{s}_i(j)\right]_1 + E\left[\mathbf{w}_{\min}(j)\cdot\mathbf{s}_i(j)\right]_2,
\tag{3.32}
$$

where

$$\begin{aligned} E[\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)]_1 &= \int_{-\infty}^{\infty} w'^2 f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w=w')\,dw', \\ E[\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)]_2 &= \int_{-\infty}^{\infty} w' \int_{w'}^{\infty} w f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w>w')\,dw\,dw'. \end{aligned} \tag{3.33}$$

Similarly,

$$E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big] = E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big]_1 + E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big]_2, \tag{3.34}$$

where

$$\begin{aligned} E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big]_1 &= \int_{-\infty}^{\infty} w'^4 f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w=w')\,dw', \\ E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big]_2 &= \int_{-\infty}^{\infty} w'^2 \int_{w'}^{\infty} w^2 f_{\mathbf{w}_{\min}(j),\mathbf{s}_i(j)}(w', w>w')\,dw\,dw'. \end{aligned} \tag{3.35}$$

The analysis of the maximum and median attacks follows the same approach, and the probability density functions are in Section 3.2.

Under the minmax attack $\mathbf{w}_{\text{MinMax}}(j) \triangleq (1/2)(\mathbf{w}_{\min}(j)+\mathbf{w}_{\max}(j))$, for a guilty colluder $i \in S_C$,

$$\begin{aligned} &E[\mathbf{w}_{\text{MinMax}}(j)\cdot \mathbf{s}_i(j)] \\ &\quad = E\Big[\Big(\frac{1}{2}\mathbf{w}_{\min}(j)+\frac{1}{2}\mathbf{w}_{\max}(j)\Big)W^{(i)}\Big] \\ &\quad = \frac{1}{2}E[\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)] + \frac{1}{2}E[\mathbf{w}_{\max}(j)\cdot \mathbf{s}_i(j)], \\ &E\Big[\big(\mathbf{w}_{\text{MinMax}}(j)\cdot \mathbf{s}_i(j)\big)^2\Big] \\ &\quad = E\Big[\Big(\frac{1}{2}\mathbf{w}_{\min}(j)+\frac{1}{2}\mathbf{w}_{\max}(j)\Big)^2 \big(\mathbf{s}_i(j)\big)^2\Big] \\ &\quad = \frac{1}{4}E\Big[\big(\mathbf{w}_{\min}(j)\cdot \mathbf{s}_i(j)\big)^2\Big] + \frac{1}{4}E\Big[\big(\mathbf{w}_{\max}(j)\cdot \mathbf{s}_i(j)\big)^2\Big] \\ &\qquad + \frac{1}{2}E\Big[\mathbf{w}_{\min}(j)\cdot \mathbf{w}_{\max}(j)\cdot \big(\mathbf{s}_i(j)\big)^2\Big]. \end{aligned} \tag{3.36}$$

In addition, under the minmax attack,

$$\begin{aligned} &E\Big[\big(\mathbf{w}_{\text{MinMax}}(j)\big)^2\Big] \\ &\quad = E\Big[\Big(\frac{1}{2}\mathbf{w}_{\min}(j)+\frac{1}{2}\mathbf{w}_{\max}(j)\Big)^2\Big] \\ &\quad = \frac{1}{4}E\Big[\big(\mathbf{w}_{\min}(j)\big)^2\Big] + \frac{1}{4}E\Big[\big(\mathbf{w}_{\max}(j)\big)^2\Big] + \frac{1}{2}E[\mathbf{w}_{\min}(j)\cdot \mathbf{w}_{\max}(j)]. \end{aligned} \tag{3.37}$$

The results from the previous analysis on the minimum and maximum attacks can be applied to (3.38). For a guilty colluder $i \in S_C$, $E[\mathbf{w}_{\min}(j)\cdot\mathbf{w}_{\max}(j)(\mathbf{s}_i(j))^2]$ can be calculated by

$$E\Big[\mathbf{w}_{\min}(j)\cdot\mathbf{w}_{\max}(j)\big(\mathbf{s}_i(j)\big)^2\Big] = \iiint w'w''w^2 f_{\mathbf{w}_{\min}(j),\mathbf{w}_{\max}(j),\mathbf{s}_i(j)}(w',w'',w)\,dw\,dw''\,dw', \tag{3.38}$$

where $f_{\mathbf{w}_{\min}(j),\mathbf{w}_{\max}(j),\mathbf{s}_i(j)}(w',w'',w)$ is the joint pdf of $\mathbf{w}_{\min}(j)$, $\mathbf{w}_{\max}(j)$, and $\mathbf{s}_i(j)$ as shown in (3.18) in Section 3.2. To calculate the correlation between $\mathbf{w}_{\min}(j)$ and $\mathbf{w}_{\max}(j)$,

$$E[\mathbf{w}_{\min}(j)\cdot\mathbf{w}_{\max}(j)] = \int_{-\infty}^{\infty}\int_{w'}^{\infty} w'w'' f_{\mathbf{w}_{\min}(j),\mathbf{w}_{\max}(j)}(w',w'')\,dw''\,dw', \tag{3.39}$$

where $f_{\mathbf{w}_{\min}(j),\mathbf{w}_{\max}(j)}(w',w'')$ is joint probability density function of $\mathbf{w}_{\min}(j)$ and $\mathbf{w}_{\max}(j)$ and is in (3.12) in Section 3.2.

The analysis of the modified negative (modneg) attack is similar to that of the minmax attack, and the probability density functions are available in Section 3.2.

Assume that the parameter p in the randomized negative (randneg) attack is independent of the fingerprints $\{\mathbf{s}_i(j)\}$. Therefore, under the randomized negative attack, the colluded fingerprint can be written as

$$\mathbf{w}_{\text{RandNeg}}(j) = \mathbf{w}_{\min}(j)\cdot B_p + \mathbf{w}_{\max}(j)\cdot(1-B_p), \tag{3.40}$$

where B_p is a Bernoulli random variable with parameter p and is independent of $\{\mathbf{s}_i(j)\}$. For a guilty colluder $i \in S_C$, the mth moment ($m = 1,2,\ldots$) of $\mathbf{w}_{\text{RandNeg}}(j)\cdot\mathbf{s}_i(j)$ is

$$\begin{aligned}E\big[\big(\mathbf{w}_{\text{RandNeg}}(j)\cdot\mathbf{s}_i(j)\big)^m\big] &= E\big[E\big[\big(\mathbf{w}_{\text{RandNeg}}(j)\cdot\mathbf{s}_i(j)\big)^m|B_p\big]\big]\\ &= p\cdot E\big[\big(\mathbf{w}_{\min}(j)\cdot\mathbf{s}_i(j)\big)^m\big]+(1-p)\cdot E\big[\big(\mathbf{w}_{\max}(j)\cdot\mathbf{s}_i(j)\big)^m\big],\end{aligned} \tag{3.41}$$

and the mth moment ($m = 1,2,\ldots$) of W^{RandNeg} is

$$\begin{aligned}E\Big[\big(\mathbf{w}_{\text{RandNeg}}(j)\big)^m\Big] &= E\Big[E\Big[\big(\mathbf{w}_{\text{RandNeg}}(j)\big)^m|B_p\Big]\Big]\\ &= p\cdot E\Big[\big(\mathbf{w}_{\min}(j)\big)^m\Big]+(1-p)\cdot E\Big[\big(\mathbf{w}_{\min}(j)\big)^m\Big].\end{aligned} \tag{3.42}$$

From all the above analysis, the correlation kernel term $T'_N(i)$ can be approximated by the following Gaussian distribution:

$$T'_N(i) \sim \begin{cases} \mathcal{N}\left(0, \dfrac{\sigma^2_{g,H_0}}{N}\right) & \text{if } i \notin S_C, \\ \mathcal{N}\left(\mu_{g,H_1}, \dfrac{\sigma^2_{g,H_1}}{N}\right) & \text{if } i \in S_C. \end{cases} \tag{3.43}$$

3.4.2. Analysis of detection statistics

From (3.43), the detection statistics $T_N(i)$ can be approximated by a Gaussian random variable

$$T_N(i) = \frac{NT'_N(i)}{\sqrt{||\mathbf{s}_i||^2}} \sim \begin{cases} \mathcal{N}\left(0, \dfrac{\sigma^2_{g,H_0}}{\sigma^2_W}\right) & \text{if } i \notin S_C, \\ \mathcal{N}\left(\dfrac{\sqrt{N}\mu_{g,H_1}}{\sigma_W}, \dfrac{\sigma^2_{g,H_1}}{\sigma^2_W}\right) & \text{if } i \in S_C. \end{cases} \tag{3.44}$$

The Z statistics can be approximated by a Gaussian random variable $\mathcal{N}(\mu_Z(i), 1)$ with mean

$$\mu_Z(i) = \frac{1}{2}\sqrt{N-3}\log\frac{1+E[\rho(i)]}{1-E[\rho(i)]}. \tag{3.45}$$

$E[\rho(i)]$ is the mean of $\rho(i)$ defined in (2.12) and is the estimated correlation coefficient of the extracted fingerprint $\mathbf{w}$ and the original fingerprint $\mathbf{s}_i$ [62]. So

$$Z(i) \sim \begin{cases} \mathcal{N}\left(0, 1\right) & \text{if } i \notin S_C, \\ \mathcal{N}\left(\dfrac{1}{2}\sqrt{N-3}\log\dfrac{1+E[\rho(i)]}{1-E[\rho(i)]}, 1\right) & \text{if } i \in S_C, \end{cases} \tag{3.46}$$

where for $i \in S_C$,

$$E[\rho(i)] \approx \frac{\operatorname{cov}\left[g\left(\{\mathbf{s}_k\}_{k\in S_C}\right), \mathbf{s}_i\right]}{\sqrt{\sigma^2_W\sigma^2_{g,w}}} = \frac{\mu_{g,H_1}}{\sqrt{\sigma^2_W\sigma^2_{g,w}}}. \tag{3.47}$$

$\sigma^2_{g,Y}$ is the variance of the extracted fingerprint.

The q statistics normalize the correlation term with the unbiased estimate of its variance, and therefore,

$$q(i) \sim \begin{cases} \mathcal{N}\left(0, 1\right) & \text{if } i \notin S_C, \\ \mathcal{N}\left(\dfrac{\sqrt{N}\mu_{g,H_1}}{\sqrt{\sigma^2_{g,H_1}}}, 1\right) & \text{if } i \in S_C. \end{cases} \tag{3.48}$$

3.4.3. System performance analysis

Analysis of P_d, P_{fp}, $E[F_d]$, and $E[F_{fp}]$. In the system model in Section 3.3 with a total of M users and K colluders, given a signal to be tested and given one detection statistics, K out of the M statistics $\{T_N(i)\}_{i=1,\dots,M}$ are normally distributed with a positive mean and the others are normally distributed with a zero mean, as analyzed in the previous section.

Take the T_N statistics as an example, define

$$\begin{aligned}\mu_1 &\triangleq \frac{\sqrt{N}\mu_{g,H_1}}{\sigma_W},\\ \sigma_1^2 &\triangleq \frac{\sigma_{g,H_1}^2}{\sigma_W^2},\\ \sigma_0^2 &\triangleq \frac{\sigma_{g,H_0}^2}{\sigma_W^2}.\end{aligned} \tag{3.49}$$

If $\{T_N(i)\}_{i=1,\dots,M}$ are uncorrelated with each other or the correlation is very small, then for a given threshold h, P_d and P_{fp} can be aproximated by

$$\begin{aligned}P_d &= P\Big[\max_{i\in S_C} T_N(i) > h\Big]\\ &= 1 - P\big[T_N(i_1 \in S_C) \le h, \dots, T_N(i_K \in S_C) \le h\big]\\ &\approx 1 - \left[1 - Q\left(\frac{h-\mu_1}{\sigma_1}\right)\right]^K,\\ P_{fp} &= P\Big[\max_{j\notin S_C} T_N(j) > h\Big]\\ &= 1 - P\big[T_N(j_1 \notin S_C) \le h, \dots, T_N(j_{M-K} \notin S_C) \le h\big]\\ &\approx 1 - \left[1 - Q\left(\frac{h}{\sigma_0}\right)\right]^{M-K},\end{aligned} \tag{3.50}$$

respectively, where $Q(x) = \int_x^\infty (1/\sqrt{2\pi})e^{-t^2/2}dt$ is the Gaussian tail function.

To calculate γ_c and γ_i, we define the indication function $I(i)$ as

$$I(i) = \begin{cases}1 & \text{if } T_N(i) \ge h,\\ 0 & \text{if } T_N(i) < h.\end{cases} \tag{3.51}$$

Therefore, γ_c and γ_i can be approximated by

$$\begin{aligned}\gamma_c &= E\left[\frac{\sum_{i\in S_c} I(i)}{K}\right] = \sum_{i\in S_C} \frac{P[T_N(i) > h]}{K} \approx Q\left(\frac{h-\mu_1}{\sigma_1}\right),\\ \gamma_i &= E\left[\frac{\sum_{j\notin S_c} I(j)}{M-K}\right] = \sum_{j\notin S_C} \frac{P[T_N(j) > h]}{M-K} \approx Q\left(\frac{h}{\sigma_0}\right),\end{aligned} \tag{3.52}$$

respectively.

The analysis of P_d, P_{fp}, γ_c, and γ_i for the Z and q statistics are the same.

Perceptual quality. In the system model in Section 3.3, the distortion introduced to the host signal by the colluded fingerprint is $n(j) = \text{JND}(j) \cdot g(\{\mathbf{s}_k(j)\}_{k\in S_C})$, $j = 1, 2, \ldots, N$. Given the collusion attack $g(\cdot)$ and the number of colluders K, if $\mathbf{A} \triangleq g(\{\mathbf{s}_k(j)\}_{k\in S_C})$ has the pdf $f_{g,K}(w)$, MSE_{JND} can be simplified to

$$\begin{aligned}\text{MSE}_{\text{JND}} &\approx N \times E\left[(|\mathbf{A}| - 1)^2 \mid |\mathbf{A}| > 1\right]\\ &= N\int_{-\infty}^{-1} (w+1)^2 f_{g,K}(w)dw + N\int_{1}^{\infty} (w-1)^2 f_{g,K}(w)dw,\end{aligned} \tag{3.53}$$

and $E[F_{\text{JND}}]$ is set as follows:

$$E[F_{\text{JND}}] = P[|\mathbf{A}| > 1] = \int_{-\infty}^{-1} f_{g,K}(w)dw + \int_{1}^{\infty} f_{g,K}(w)dw. \tag{3.54}$$

3.5. Collusion attacks on Gaussian-based fingerprints

It has been shown in [62] that the uniform fingerprints can be easily defeated by nonlinear collusion attacks, and the simulation results there also showed that the Gaussian fingerprints are more resistant to nonlinear collusion attacks than the uniform fingerprints. However, no analytic study was provided in the literature on the resistance of Gaussian fingerprints to nonlinear collusion attacks. This section studies the effectiveness of nonlinear collusion attacks on Gaussian-based fingerprints.

3.5.1. Unbounded Gaussian fingerprints

Statistical analysis. First, the resistance of unbounded Gaussian fingerprints to collusion attacks is studied. As before, assume that there are a total of M users and the fingerprints $\{\mathbf{s}_i(j)\}$ are i.i.d. Gaussian with zero mean and variance σ_W^2. Usually $\sigma_W^2 \approx 1/9$ is used to ensure that around 99.9% of fingerprint components are in the range of $[-1, 1]$ and are imperceptible after being scaled by a JND factor.

Under the assumption that the Bernoulli random variable B_p in the randomized negative attack is independent of the zero-mean Gaussian fingerprints,

$$E\left[(\mathbf{w}_{\text{RandNeg}}(j))^2\right] = E\left[(\mathbf{w}_{\min}(j))^2\right] = E\left[(\mathbf{w}_{\max}(j))^2\right] \tag{3.55}$$

for all possible $p \in [0, 1]$. Consequently,

$$\begin{aligned} \sigma^2_{\text{RandNeg},w} &= E\Big[\big(\mathbf{w}_{\text{RandNeg}}(j)\big)^2\Big] - \big(E[\mathbf{w}_{\text{RandNeg}}(j)]\big)^2 \\ &\leq E\Big[\big(\mathbf{w}_{\min}(j)\big)^2\Big], \end{aligned} \tag{3.56}$$

and the upper bound of the variance in (3.56) is achieved when $p = 0.5$ and $E[\mathbf{w}_{\text{RandNeg}}(j)] = 0$. From (3.50) and (3.52), the larger the variance, the more effective the attack. Consequently, $p = 0.5$ is considered which corresponds to the most effective attack.

Given the analysis in the previous section, the parameters μ_{g,H_1}, σ^2_{g,H_1}, σ^2_{g,H_0}, and $\sigma^2_{g,w}$ can be calculated for Gaussian distribution with zero mean and variance σ^2_W. Due to the existence of the $Q(\cdot)$ terms in the probability density functions, analytical expressions are not available. The recursive adaptive Simpson quadrature method [75] can be used to numerically evaluate the integrals with an absolute error tolerance of 10^{-6} and the results for $\sigma^2_W = 1/9$ are plotted in Figure 3.4.

From Figure 3.4, for a given number of colluders K, μ_{g,H_1} are the same for all collusion attacks and equal to σ^2_W/K. Different collusion attacks have different σ^2_{g,H_1}, σ^2_{g,H_0}, and $\sigma^2_{g,w}$. The relationship of σ^2_{g,H_1} and σ^2_{g,H_0} for different collusion attacks are

$$\begin{aligned} \sigma^2_{\text{RandNeg},H_1} &= \sigma^2_{\min,H_1} = \sigma^2_{\max,H_1} \\ &> \sigma^2_{\text{ModNeg},H_1} > \sigma^2_{\text{ave},H_1} \approx \sigma^2_{\text{med},H_1} \approx \sigma^2_{\text{MinMax},H_1}, \\ \sigma^2_{\text{RandNeg},H_0} &= \sigma^2_{\min,H_0} = \sigma^2_{\max,H_0} \\ &> \sigma^2_{\text{ModNeg},H_0} > \sigma^2_{\text{ave},H_0} \approx \sigma^2_{\text{med},H_0} \approx \sigma^2_{\text{MinMax},H_0}, \end{aligned} \tag{3.57}$$

and that of $\sigma^2_{g,w}$ is

$$\sigma^2_{\text{RandNeg},w} > \sigma^2_{\text{ModNeg},w} > \sigma^2_{\min,w} = \sigma^2_{\max,w} > \sigma^2_{\text{ave},w} \approx \sigma^2_{\text{med},w} \approx \sigma^2_{\text{MinMax},w}. \tag{3.58}$$

Note that the extracted fingerprint $\mathbf{w}$ under the minimum or maximum attack is not zero mean. σ^2_{g,H_0} is proportional to the second moment of $\mathbf{w}$, and is the largest under the minimum, maximum, and randomized negative attacks. However, the variance of $\mathbf{w}$ under the minimum or maximum attacks is small and comparable with $\sigma^2_{g,w}$ under the average, median, and minmax attacks.

In order to compare the effectiveness of different collusion attacks, define

(i) "attack A > attack B": attack A is more effective than attack B in defeating the system,

(ii) "attack A = attack B": attack A and attack B have the same performance in defeating the system,

(iii) "attack A ≈ attack B": attack A and attack B have similar performance in defeating the system.

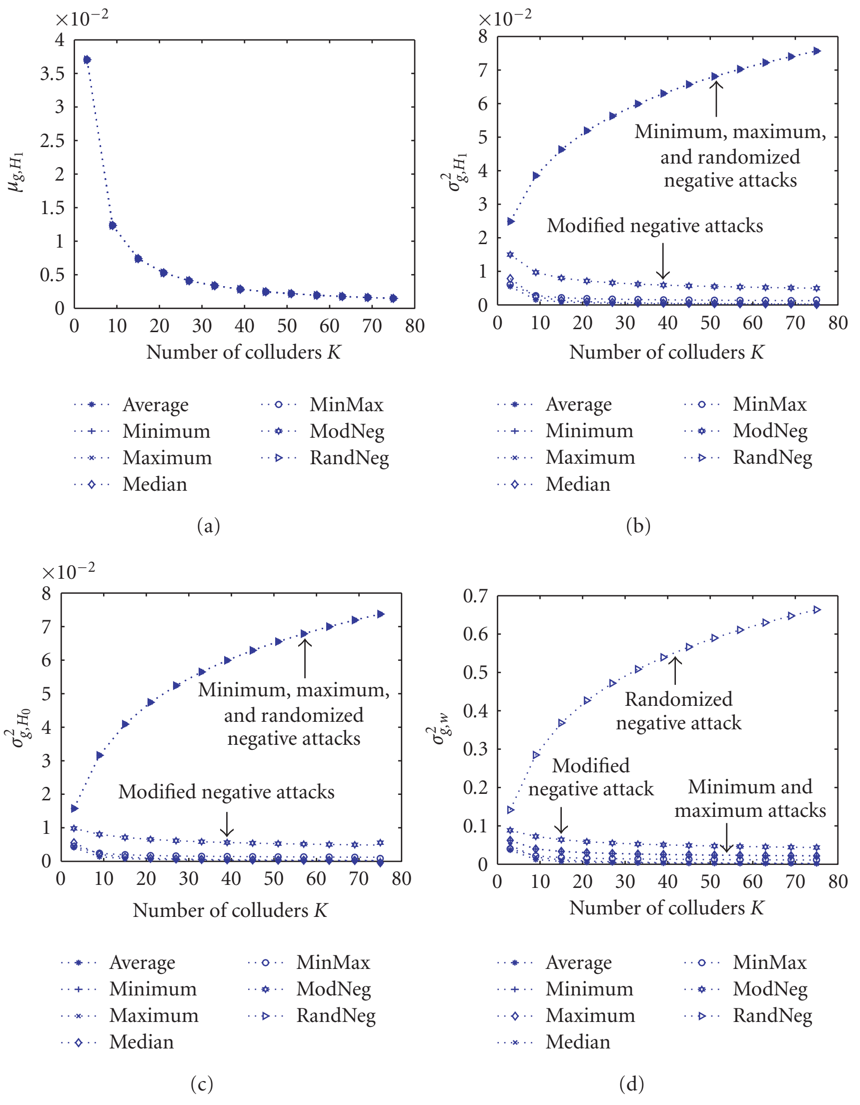


FIGURE 3.4. (a) μ_{g,H_1}, (b) σ^2_{g,H_1}, (c) σ^2_{g,H_0}, and (d) $\sigma^2_{g,w}$ of the unbounded Gaussian fingerprints with $\sigma^2_W = 1/9$.

From (3.50), (3.52), (3.57), and (3.58), with the T_N statistics or the q statistics, different collusion attacks can be sorted in the descending order of their effectiveness as follows:

$$\begin{aligned} &\text{Minimum} = \text{Maximum} = \text{RandNeg} \\ &\qquad > \text{ModNeg} > \text{Average} \approx \text{Median} \approx \text{MinMax}; \end{aligned} \tag{3.59}$$

and with the Z statistics, different attacks can be sorted in the descending order of their effectiveness as follows:

$$\text{RandNeg} > \text{ModNeg} > \text{Minimum} = \text{Maximum} > \text{Average} \approx \text{Median} \approx \text{MinMax}. \tag{3.60}$$

Therefore, the randomized negative attack is the most effective attack.

To analyze the perceptual quality of the colluded copies under different collusions, Figure 3.5 shows the MSE_{JND} and $E[F_{\text{JND}}]$ of different collusion attacks with i.i.d. $\mathcal{N}(0, 1/9)$ fingerprints. From Figure 3.5, although the minimum, maximum, and randomized negative attacks are more effective in defeating the fingerprinting system, they also introduce larger noticeable distortion that is proportional to the number of colluders.

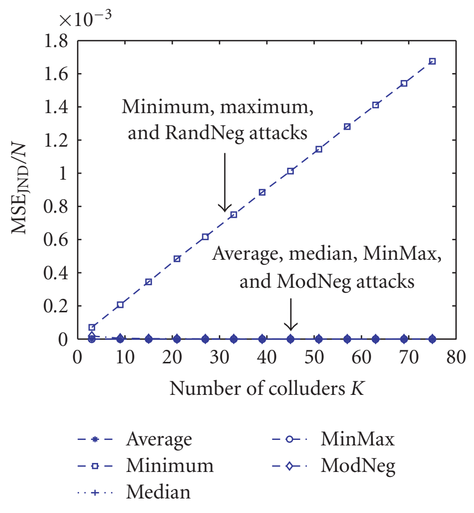
Simulation results. The simulation is set up as follows. Since the number of embeddable coefficients in 256×256 and 512×512 images is usually $O(10^4)$, the length of the fingerprints is assumed to be equal to 10 000. To accommodate a total of $M = 100$ users, 100 different fingerprints of length 10 000 are generated independently. Every fingerprint component is independent of each other and follows the $\mathcal{N}(0, 1/9)$ Gaussian distribution. The simulation results are based on a total of 2000 simulation runs.

Figure 3.6 compare the performance of the T_N statistics under different collusion attacks. The performance of the q statistics is similar to that of T_N and omitted. The simulation results agree with the statistical analysis in Section 3.5.1. From Figures 3.6a and 3.6b, with the T_N or q statistics, the minimum, maximum, and randomized negative attacks are the most effective attacks followed by the modified negative attack. The average, median, and minmax attacks are the least effective attacks.

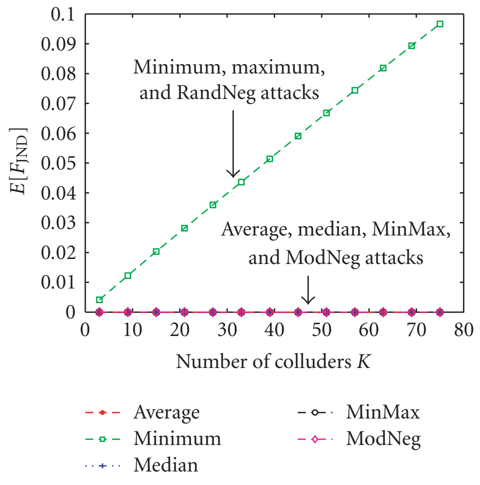
In Figure 3.7, we compare the effectiveness of different collusions with the Z statistics. From Figures 3.7a and 3.7b, with the Z statistics, the randomized negative attacks is the most effective attack followed by the modified negative attack. The average, median, and minmax attacks have similar performance and they are the least efficient attacks. The minimum and maximum attacks are the second least effective attacks. This is in agreement with the analysis in the previous section.

We compare the performance of different statistics in Figure 3.8a. We only plot the performance of the minimum and that of the modified negative attack since the maximum attack yield the same result as the minimum attack and all other attacks have a similar trend. From Figures 3.8a and 3.8b, the Z statistics are more resistant to the minimum and maximum attacks than the T_N and q statistics while the three statistics have similar performance under other collusion attacks.

To summarize, from the colluders' point of view, the best strategy for them is to choose the randomized negative attack. From the detector's point of view, the Z statistics should be used to be more robust against the minimum and maximum attacks.



(a)



(b)

FIGURE 3.5. Perceptual quality of the attacked copy under different attacks with unbounded Gaussian fingerprints. Here $\sigma_W^2 = 1/9$. (a) $\text{MSE}_{\text{JND}}/N$. (b) $E[F_{\text{JND}}]$.

Figure 3.9 shows the attacked images after the average and the minimum attacks with 75 colluders. Although the minimum, maximum, and randomized negative attacks are more effective, they also introduce much larger noticeable distortion in the host image. This is because the fingerprints are not bounded, and in fact, such unbounded fingerprints can introduce noticeable distortion in the fingerprinted copies even when without collusion.

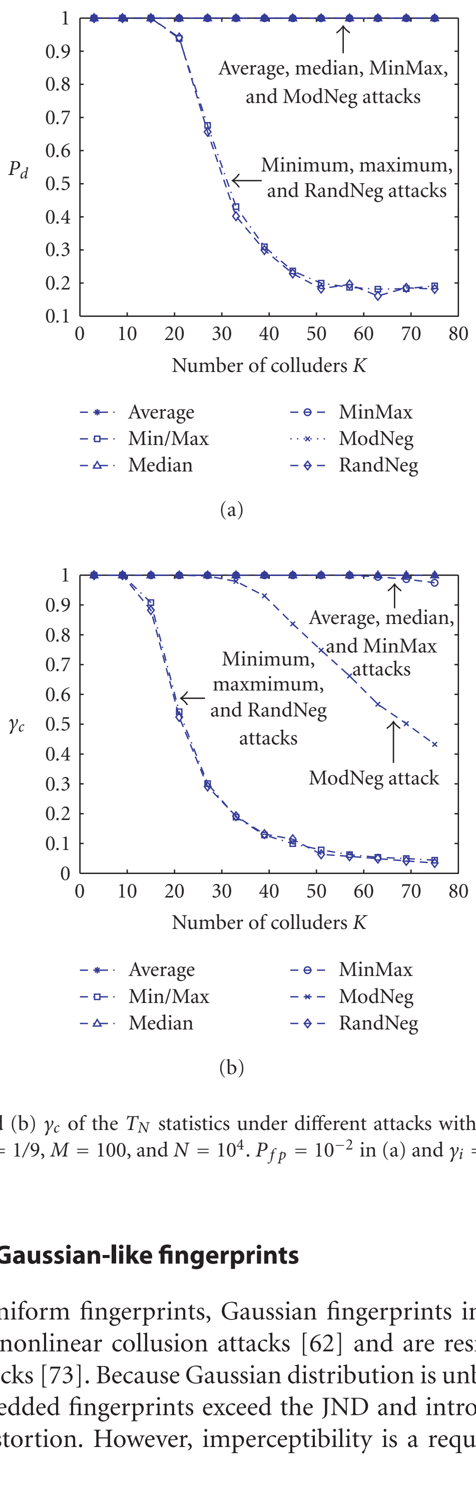


FIGURE 3.6. (a) P_d and (b) γ_c of the T_N statistics under different attacks with unbounded Gaussian fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

3.5.2. Bounded Gaussian-like fingerprints

Compared with uniform fingerprints, Gaussian fingerprints improve the detector's resistance to nonlinear collusion attacks [62] and are resilient to statistical and histogram attacks [73]. Because Gaussian distribution is unbounded, it is possible that the embedded fingerprints exceed the JND and introduce perceptually distinguishable distortion. However, imperceptibility is a requirement of digital

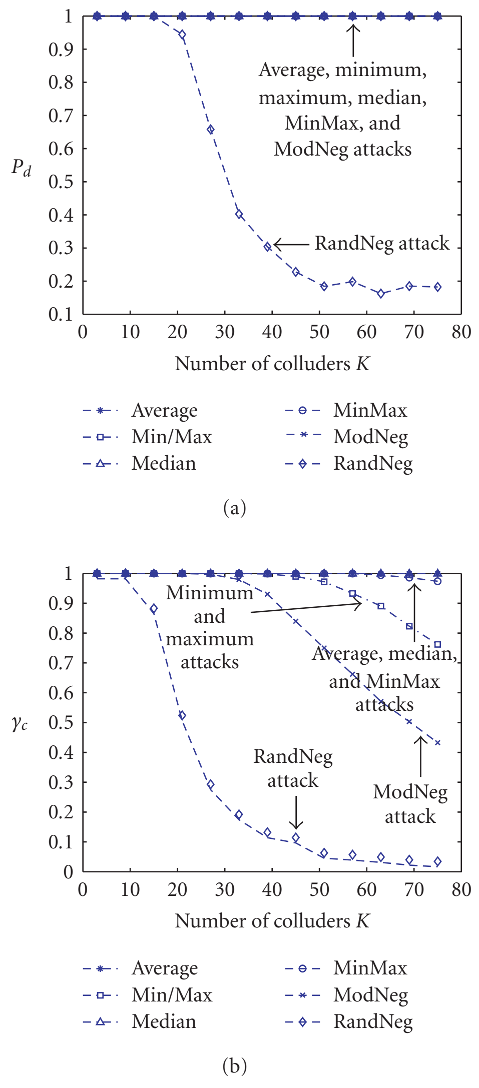
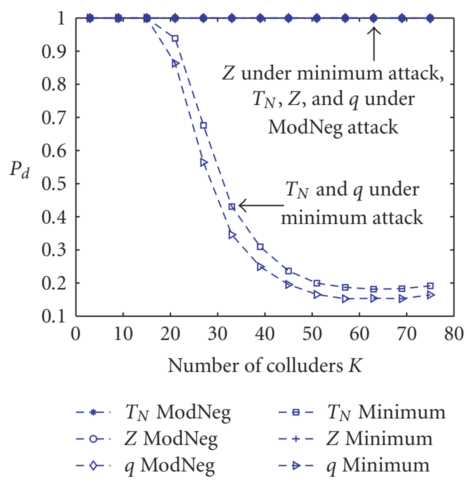


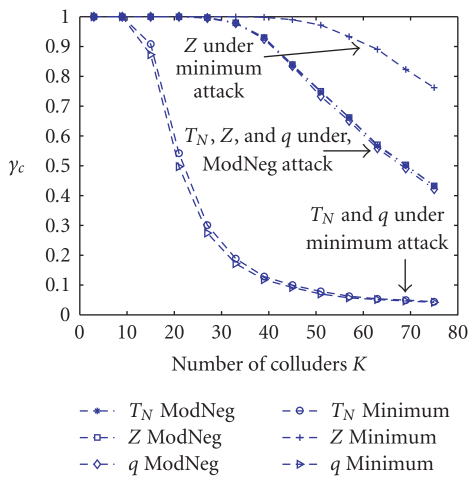
FIGURE 3.7. (a) P_d and (b) γ_c of the Z statistics under different attacks with unbounded Gaussian fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

fingerprinting and the owner has to guarantee the perceptual quality of the fingerprinted copies. In order to remove the perceptual distortion while maintaining the robustness against collusion attacks, we introduce the bounded Gaussian-like fingerprints.

Assume that $f_X(\cdot)$ and $F_X(\cdot)$ are the probability density function and cumulative distribution function of a Gaussian random variable with zero mean and variance σ_W^2, respectively. The probability density function of a bounded Gaussian-like



(a)



(b)

FIGURE 3.8. (a) P_d and (b) γ_c of different statistics under the minimum attack and modified negative attack with unbounded Gaussian fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

distribution $\widetilde{f}_X(\cdot)$ is

$$\widetilde{f}_X(x) = \begin{cases} \dfrac{f_X(x)}{F_X(1) - F_X(-1)} & \text{if } -1 \leq x \leq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.61}$$

Figure 3.10 shows an example of the probability density function of a bounded Gaussian-like distribution.

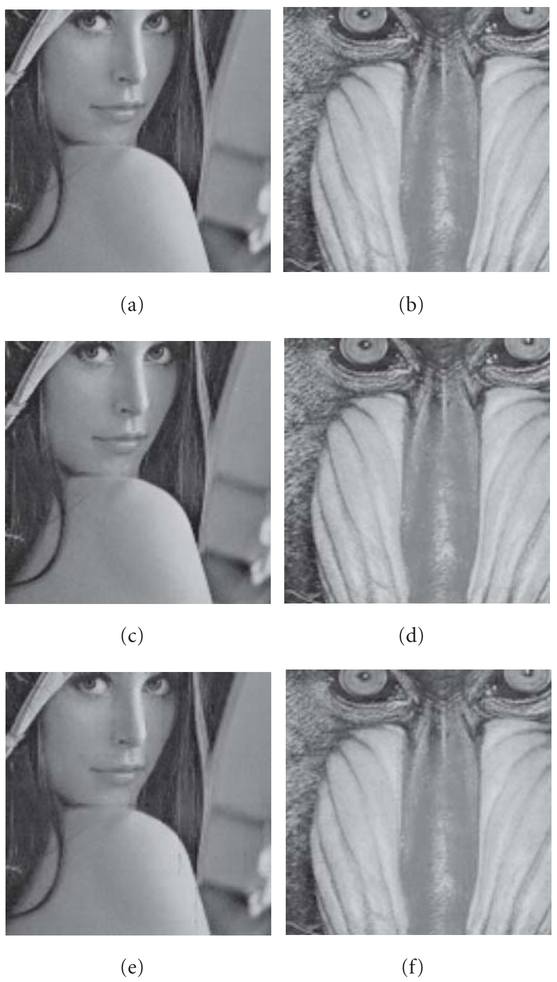


Figure 3.9. Comparison of perceptual quality of the attacked images under different attacks with 75 colluders. Fingerprints are generated from unbounded Gaussian distribution with $\sigma_W^2 = 1/9$. (Left) Lena. (Right) Baboon. (Top) The zoomed-in region of the original 256×256 images. (Middle) The colluded images under the average attack. (Bottom) The colluded images under the minimum attack.

It can be shown that the variance of fingerprints following pdf (3.61) is σ_W^2, and the embedded fingerprints introduce no perceptual distortion since $\mathrm{MSE}_{\mathrm{JND}} = 0$ and $F_{\mathrm{JND}} = 0$. By bounding the fingerprints in the range of $[-1, 1]$, the content owner maintains the energy of the embedded fingerprints while achieving the imperceptibility.

For fingerprints following distribution (3.61), the analyses of the collusion attacks and the detection statistics are similar to the unbounded case and thus omitted. If different collusion attacks are sorted according to their effectiveness, the result is the same as that of the unbounded Gaussian fingerprints.

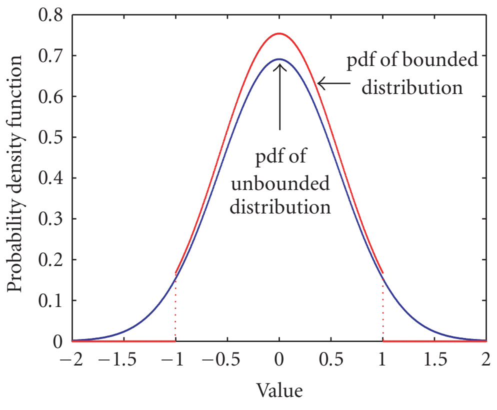


Figure 3.10. An example of the probability density function of a bounded Gaussian-like distribution.

The simulation of the bounded Gaussian-like fingerprints under collusion attacks is set up similarly to that in Section 3.5.1. Assume that there are a total of $M = 100$ users and the host signal has $N = 10^4$ embeddable coefficients. The i.i.d. fingerprints are generated from the distribution (3.61) with $\sigma_W^2 = 1/9$.

Figures 3.11 and 3.12 show the performance of the T_N and Z statistics, respectively, under different attacks. The performance of the q statistics is similar to that of T_N. Figure 3.13 compares the performance of different detection statistics under the minimum and the modified negative attack. The simulation results are similar to those in the unbounded case. From the colluders' point of view, the most efficient attack is the randomized negative attack, and from the detector's point of view, the Z statistics are more robust.

3.6. Preprocessing of the extracted fingerprints

The three detection statistics in Section 3.3 are not specifically designed for collusion scenarios, and therefore do not take into account the characteristics of the newly generated copies after the collusion attacks. Intuitively, utilizing the statistical features of the attacked copies may improve the detection performance, and one of such features is the sample mean of the extracted fingerprint under the collusion attacks. From the histogram plots of the extracted fingerprints under different attacks as shown in Figure 3.14, different patterns of the sample means of the extracted fingerprints can be observed: the extracted fingerprints have approximately zero sample mean under the average, median, minmax, and modified negative attacks; the minimum attack yields a negative sample mean, and the maximum attack yields a positive sample mean; and under the randomized negative attack, the histogram of the extracted fingerprint components have two clusters, one with a negative mean and the other with a positive mean.

Recalling from Section 3.4.1 that σ_{g,H_0}^2 is proportional to the second moment of the extracted fingerprint, subtracting the sample mean from the extracted fingerprint will reduce its second-order moment, thus help improve the detection

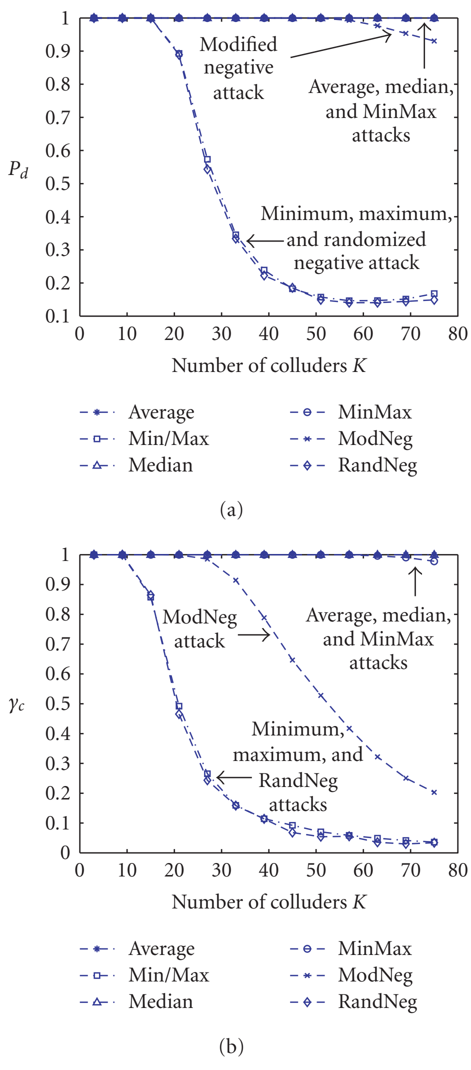


FIGURE 3.11. (a) P_d and (b) γ_c of the T_N statistics under different attacks with bounded Gaussian-like fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

performance. Similarly, the detection performance under the randomized negative attack can be improved by decreasing σ^2_{g,H_0} and $\sigma^2_{g,w}$.

Motivated by this analysis, a preprocessing stage before the detection process was proposed in [76]: given the extracted fingerprint $\{g(\{\mathbf{s}_k(j)\}_{k\in S_c})\}_{j=1,\dots,N}$, the detector first investigates its histogram. If a single nonzero sample mean is observed, the detector subtracts it from the extracted fingerprint, and then applies the detection statistics. If the fingerprint components are merged from two (or more) distributions that have distinct mean values, the detector needs to cluster

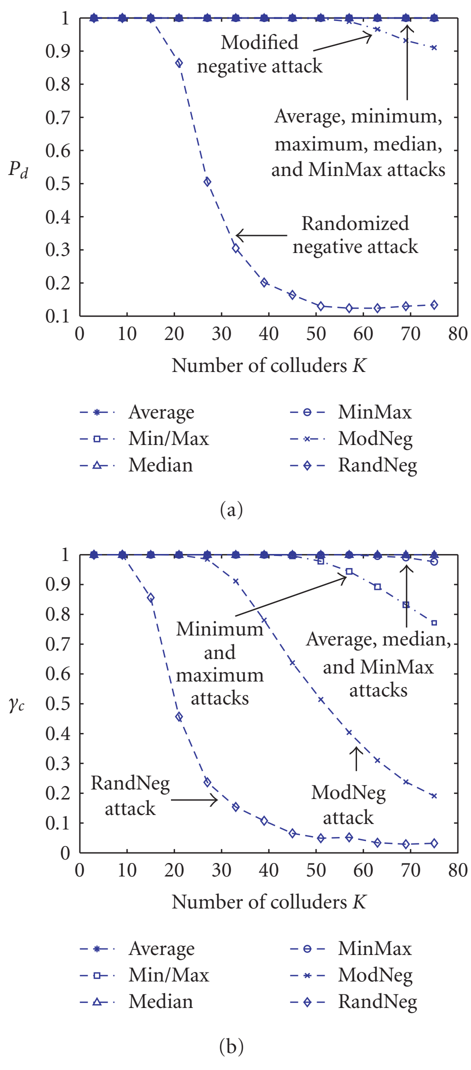
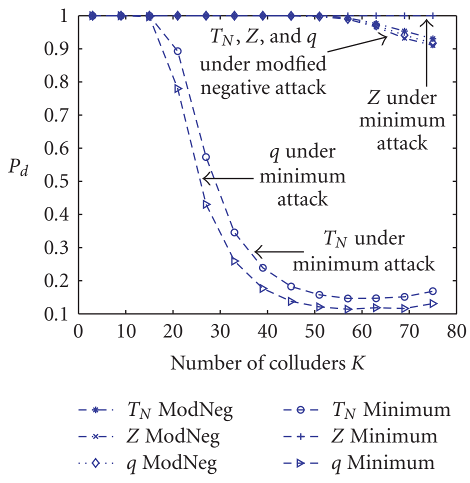
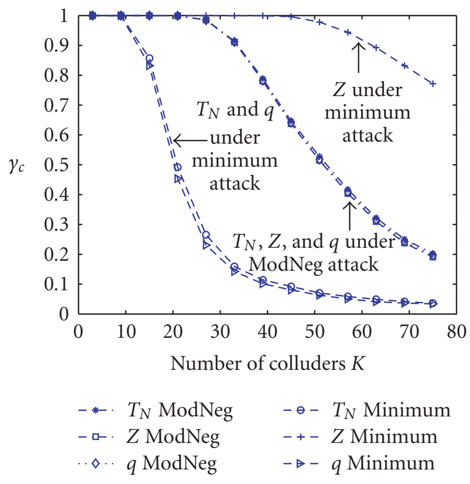


FIGURE 3.12. (a) P_d and (b) γ_c of the Z statistics under different attacks with bounded Gaussian-like fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

components and then subtract from each colluded fingerprint component the sample mean of the corresponding cluster. In the later case, the means can be estimated using a Gaussian-mixture approximation, and the clustering is based on the nearest-neighbor principle. Under the randomized negative attack, a simple solution is to first observe the bimodality in the histogram of $\{\mathbf{w}(j)\}$, and then cluster all negative components into one distribution and cluster all positive components into the other distribution. Given the extracted fingerprint $\{\mathbf{w}(j)\}_{j=1,\dots,N}$, define $\mu_{\text{neg}} \triangleq \sum_j \mathbf{w}(j) \cdot I[\mathbf{w}(j) < 0] / \sum_l I[\mathbf{w}(l) < 0]$ as the sample mean of the negative



(a)



(b)

FIGURE 3.13. (a) P_d and (b) γ_c of different statistics under the minimum attack and modified negative attack with bounded Gaussian-like fingerprints. Here $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. $P_{fp} = 10^{-2}$ in (a) and $\gamma_i = 10^{-2}$ in (b).

extracted fingerprint components where $I[\cdot]$ is the indication function, and define $\mu_{\text{pos}} \triangleq \sum_j \mathbf{w}(j) \cdot I[\mathbf{w}(j) > 0] / \sum_l I[\mathbf{w}(l) > 0]$ as the sample mean of the positive extracted fingerprint components. Then the preprocessing stage generates

$$\mathbf{w}'(j) = \begin{cases} \mathbf{w}(j) - \mu_{\text{neg}} & \text{if } \mathbf{w}(j) < 0, \\ \mathbf{w}(j) - \mu_{\text{pos}} & \text{if } \mathbf{w}(j) > 0, \end{cases} \tag{3.62}$$

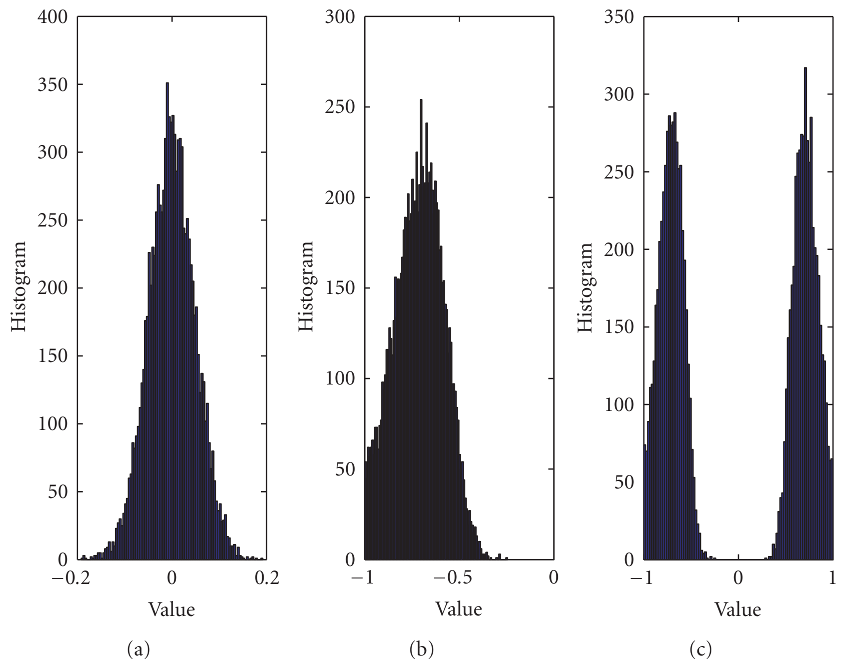
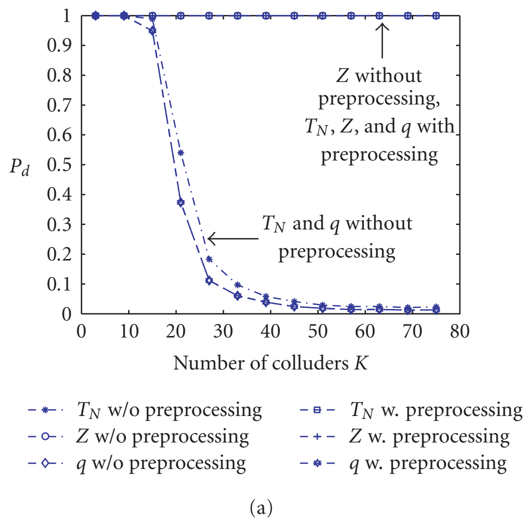


Figure 3.14. Histograms of the extracted fingerprints under the (a) average, (b) minimum, and (c) randomized negative attacks, respectively. The original fingerprints follow the distribution in (3.61) with $\sigma_W^2 = 1/9$, $N = 10^4$, and $K = 45$.

and the detector applies the detection statistics to $\{\mathbf{w}'(j)\}_{j=1}^N$. The analysis of the detection statistics with the preprocessing is the same as in Section 3.4 and is not repeated.

The simulation is set up the same as before and the fingerprint components are generated from the bounded Gaussian-like distribution (3.61) with $\sigma_W^2 = 1/9$. Figure 3.15 compares the performance of the three detection statistics with and without the preprocessing under the minimum attack. The detection performance under the maximum attack is the same as that of the minimum attack and is not shown here. In Figure 3.16, we compare the performance of the detection statistics with and without preprocessing under the randomized negative attack. From Figures 3.15 and 3.16, the preprocessing substantially improves the detection performance of the detector, and the three statistics have similar performance under the minimum, maximum, and randomized negative attacks.

Note that the estimated correlation coefficient $\rho(i)$ in the Z statistics removes the mean of the extracted fingerprint before calculating the correlation between the extracted fingerprint and the original fingerprint. This explains why the Z statistics perform better than the T_N and q statistics without preprocessing under the minimum and maximum attacks, whereby the mean of the colluded fingerprint components is substantially deviated from zero.



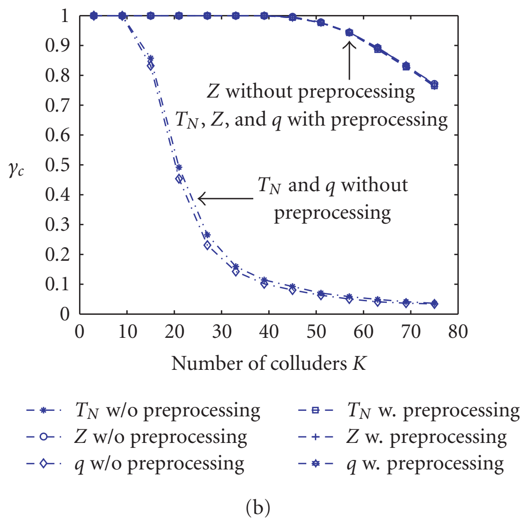
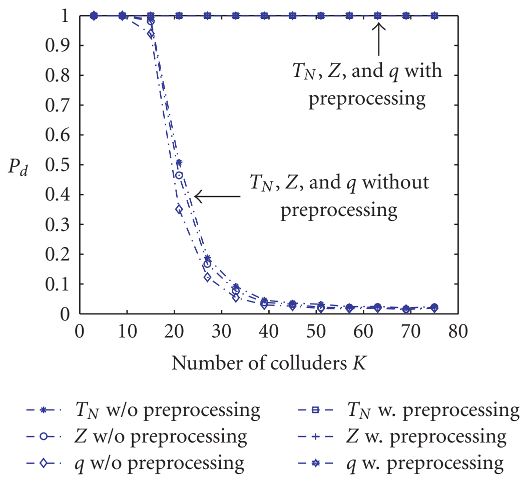


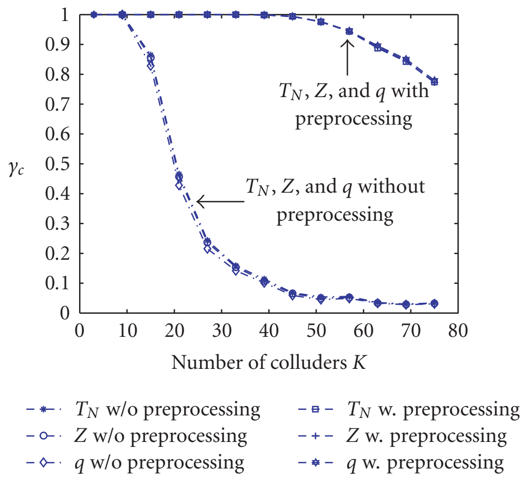
FIGURE 3.15. Performance of the detection statistics under the minimum attack with and without preprocessing. Fingerprints are generated from bounded Gaussian-like distribution (3.61) with $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. In (a), $P_{fp} = 10^{-2}$ and we compare P_d with and without preprocessing. In (b), $\gamma_i = 10^{-2}$ and we plot γ_c with and without preprocessing.

3.7. Experiments with images

This section studies the performance of Gaussian-based fingerprints under different nonlinear collusion attacks on real images. Two 256×256 host images, Lena and Baboon, are chosen and they have a variety of representative visual features such as the texture, sharp edges, and smooth areas. The human-visual-model-based spread-spectrum embedding in [24] is used and the fingerprints are



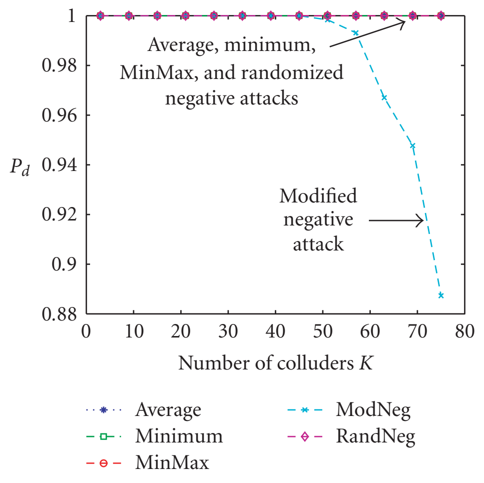
(a)



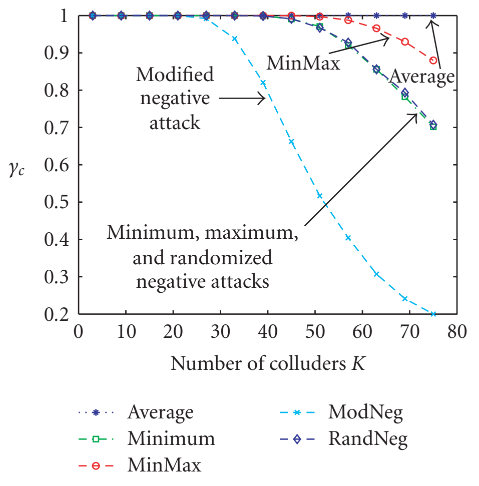
(b)

FIGURE 3.16. Performance of the detection statistics under the randomized negative attack with and without preprocessing. Fingerprints are generated from bounded Gaussian-like distribution (3.61) with $\sigma_W^2 = 1/9$, $M = 100$, and $N = 10^4$. In (a), $P_{fp} = 10^{-2}$ and we compare P_d with and without preprocessing. In (b), $\gamma_i = 10^{-2}$ and we plot γ_c with and without preprocessing.

embedded in the DCT domain. The generated fingerprints follow the bounded Gaussian-like distribution (3.61) with $\sigma_W^2 = 1/9$. Assume that the collusion attacks are also in the DCT domain. At the detector's side, a nonblind detection is performed where the host signal is first removed from the colluded copy. Then the detector applies the preprocessing to the extracted fingerprint if a nonzero sample mean is observed. Finally, the detector uses the detection statistics to identify the colluders.



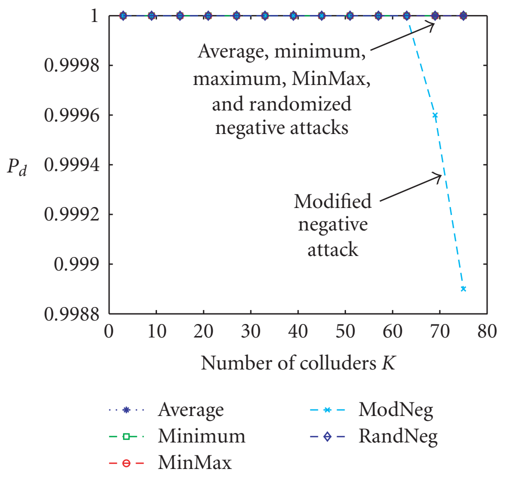
(a)



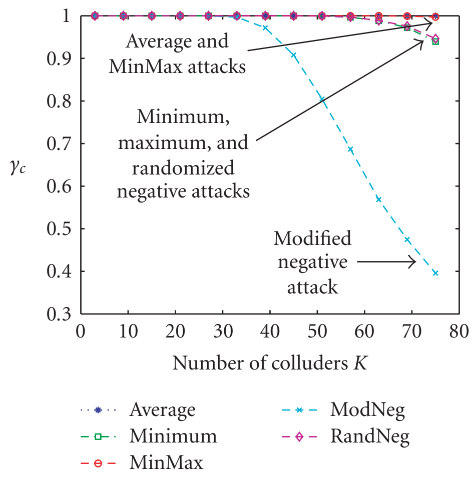
(b)

FIGURE 3.17. (a) P_d and (b) γ_c of Lena with the Z statistics under different collusion attacks. The original fingerprints follow the distribution in (3.61) with $\sigma_W^2 = 1/9$ and $M = 100$. The length of the embedded fingerprints is $N = 13\,691$. In (a), $P_{fp} = 10^{-2}$ and simulation results are based on 10 000 simulation runs. In (b), $\gamma_i = 10^{-2}$ and simulation results are based on 1 000 simulation runs.

Figures 3.17 and 3.18 show the simulation results of Lena and Baboon, respectively. We only show the performance the Z statistics under different nonlinear collusion attacks. The T_N and q statistics have similar performance and are omitted. Assume that there are a total of $M = 100$ users. The simulation results from real images agree with the analysis in Section 3.4, and are comparable to the simulation results in Sections 3.5 and 3.6. In addition, a better performance is observed



(a)



(b)

FIGURE 3.18. (a) P_d and (b) γ_c of Baboon with the Z statistics under different collusion attacks. The original fingerprints follow the distribution in (3.61) with $\sigma_W^2 = 1/9$ and $M = 100$. In (a) and (b), the length of the embedded fingerprints is $N = 13\,691$. The length of the embedded fingerprints is $N = 19\,497$. In (a), $P_{fp} = 10^{-2}$ and simulation results are based on 10 000 simulation runs. In (b), $\gamma_i = 10^{-2}$ and simulation results are based on 1000 simulation runs.

in the Baboon example than in Lena. This is because the length of the embedded fingerprints in Baboon, which is $N = 19\,497$, is larger than that in Lena, which is $N = 13\,691$. Different characteristics of the two images, for example, smooth regions and the texture, also contribute to the difference in performance.

3.8. Chapter summary

In this chapter, we have provided theoretical analysis detailing the effectiveness of different collusion attacks against orthogonal fingerprints. We studied the perceptual quality of the attacked signals under different collusion attacks. We also studied several commonly used detection statistics and compared their performance under these collusion attacks. Furthermore, we have proposed preprocessing techniques that may be used specifically for collusion scenarios to improve the detection performance.

We began by first studying the effectiveness of the average collusion attack, as well as various basic nonlinear collusion attacks, on unbounded Gaussian fingerprints. From both our analytical and simulation results, we found that, for the three detection statistics that commonly arise in the literature, the randomized negative attack is the most effective attack against the fingerprinting system. We showed that the Z statistics are more robust against the minimum and maximum attacks than the other two statistics by implicitly removing the mean of the extracted fingerprint. We also showed that all three statistics have similar performance under other collusion attacks. However, the unbounded Gaussian fingerprints may exceed JND and introduce perceptual distortion in the host signal even in the absence of collusion, and the minimum, maximum, and randomized negative attacks introduce much larger distortion in the attacked copies than others.

In order to remove the noticeable distortion introduced by the unbounded fingerprints, we introduced a family of bounded Gaussian-like fingerprints, which maintain desirable levels of robustness against the collusion attacks. With the bounded Gaussian-like fingerprints, the randomized negative attack is still the most effective attack, and the Z statistics are more robust against the minimum and maximum attacks than the other two statistics we examined. The bounding improves the perceptual quality of the fingerprinted copies and that of the attacked copies. Consequently, both the fingerprint designer and the colluders do not introduce noticeable distortion.

Observing that the extracted fingerprints under the minimum and the maximum attacks do not have a zero mean, the preprocessing stage removes the mean from the extracted fingerprints before applying the detection statistics. We also applied preprocessing to the extracted fingerprints after the randomized negative attacks, which have distinct bimodal distribution as opposed to the single modality under other collusions. We showed that these preprocessing techniques improve the detection performance, and all detection statistics give similar performance after preprocessing.

4 Orthogonal fingerprinting and collusion resistance

We are interested in collusion-resistant fingerprinting technologies for protecting multimedia data. An early milestone work was presented in [77], addressing generic data fingerprinting using an underlying principle referred to as the *marking assumption*. However, multimedia data have very different characteristics from generic data and the marking assumption may not hold when fingerprinting multimedia data. In particular, fingerprints need to be embedded into media data. These differences have a critical impact on fingerprinting design.

There have been many technologies proposed in the literature to embed and hide fingerprints (watermarks) into different media. The combination of robustness [23, 24] and capacity [38, 39] has made additive spread-spectrum embedding a promising technique for protecting multimedia, and thus it was selected for our investigations. Though most watermarking methods are easy to defeat by collusion attacks, the spread-spectrum watermarking method proposed in [23], where the watermarks have a component-wise Gaussian distribution and are statistically independent, was argued to be highly resistant to collusion attacks [23, 70]. The basic intuition of this natural strategy is that the randomness inherent in such watermarks makes the probability of accusing an innocent user very unlikely. It was shown that randomness is needed to obtain collusion-resistance [78]. There are two main approaches to using spread spectrum for fingerprint embedding: orthogonal modulation originally proposed in [23], and code modulation. As reviewed earlier, *orthogonal modulation* [79] is a popular technique for watermarking and naturally lends itself to fingerprinting applications. The orthogonality or independence allows distinguishing the fingerprints to the maximum extent. The simplicity of encoding and embedding orthogonal fingerprints makes them attractive to applications involving a small group of users.

In order to facilitate the design of multimedia forensic systems for applications with different protection requirements, one critical research direction is evaluating the resistance performance of specific fingerprinting schemes when considering different types of attacks. Thus, it is essential to provide a fundamental understanding and analysis of collusion resistance for a specific fingerprinting system, where the main purpose is to study the relationships between the resistance performance and other system parameters such as the length of the data to be marked (N); the number of users accommodated in a fingerprinting system (n); the WNR, and the number of colluding users (K). We are aware of only a few previous works that focus on analyzing the collusion resistance of digital watermarks. However, these works do not provide a precise analysis of the collusion resistance of watermarks when employed with different possible detection schemes. In this chapter we mainly address the fundamental analysis of collusion resistance for multimedia forensic systems. Since it is easy to realize, analytically tractable, and carries many of the basic features of a multimedia forensic system, we focus our collusion resistance study with a fingerprinting system that employs orthogonal Gaussian fingerprints.

Before examining the collusion-resistance performance of a specific fingerprinting system, we need to address three basic issues: one is to quantify the collusion resistance of a fingerprinting system; another is to state the specific fingerprinting system (i.e., in terms of the two major components of a fingerprinting system, the fingerprint design, and detection scheme); and the third one is to state the system requirements under different collusion-attack model. In this chapter, we present results quantifying the collusion resistance of a fingerprinting system by evaluating how many colluders are allowed before the collusion undermines the tracing capability of the system. In other words, we study the collusion resistance of a fingerprinting system in terms of its *tracing capability*, which describes how many colluders out of the total number of users are sufficient to bypass the protection provided by a particular multimedia fingerprinting system. The main goal is to analyze the relationships between the maximum allowable colluders by a fingerprinting system, $K_{\max}$, and other parameters, such as the sample length, the WNR, the total number of users, and the system performance requirements. For instance, one popular form of performance requirement is represented by the probability of a false negative (i.e., the detector fails to identify any of the colluders) and the probability of a false positive (i.e., the detector falsely indicates that an innocent user is a colluder). These relationship curves regarding tracing capability and the system parameters provide important design guidelines for a forensic system.

This chapter is organized as follows. We begin with the description of the collusion problem of interest in Section 4.1. We introduce two detection schemes, namely, the maximum detector and the thresholding detector, and also examine the theoretical collusion resistance of orthogonal fingerprinting when considering the average collusion attack. We represent the system performance by the probability of a false positive and the probability of a false negative. Since different detection goals arise under different application scenarios, two more sets of performance criteria are examined in Section 4.2. In Section 4.3, we further study other types of collusion. Since the knowledge of the number of colluders is normally not available in practice, we propose in Section 4.4 a maximum-likelihood (ML) approach to estimate the number of colluders K, and carry out simulations. Experiments using real images are demonstrated in Section 4.5. In order to overcome the linear complexity associated with traditional detection schemes for

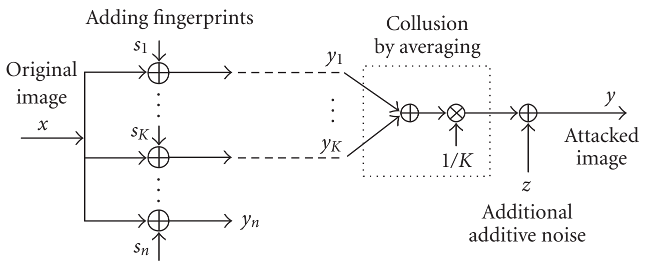


FIGURE 4.1. Model for collusion by averaging.

orthogonal modulation, in Section 4.6, we develop a tree-based detection scheme that is able to efficiently identify K colluders with an amount of correlations that is logarithmic in the number of basis vectors. Finally, we present chapter summary in Section 4.7.

4.1. Collusion resistance analysis

In this chapter, we use independent normally distributed random values as fingerprints. We first introduce the average collusion attack for these fingerprints. There are different types of collusion attacks in the literature [62]. We start with average collusion due to its popularity, its simple form, and its feasibility for analysis. We will extend our study to other attacks later in Section 4.3.

Additive embedding is a widely used watermarking scheme, where a watermark signal $\mathbf{s}_j$ is added to a host signal $\mathbf{x}$. As shown in Figure 4.1, the content owner has a family of watermarks, denoted by $\{\mathbf{s}_j\}$, which are used to mark copies of the content and facilitate colluder tracing. For the jth user, the owner computes the marked version of the content $\mathbf{y}_j$ by adding the watermark $\mathbf{s}_j$ to the host signal, $\mathbf{y}_j = \mathbf{x} + \mathbf{s}_j$. In addition to attacks operating on a single copy, collusion attacks are possible when several buyers/users having different marked copies of the same host signal come together and combine several copies to generate a new composite copy $\mathbf{y}$ such that the traces of each "original" fingerprint in the new version is removed or attenuated. We illustrate the average collusion attack in Figure 4.1, a similar model was used in [69, 72, 80]. Based on this average attack model, the observed content $\mathbf{y}$ after collusion is

$$\mathbf{y} = \frac{1}{K} \sum_{j \in S_c} \mathbf{y}_j + \mathbf{d} = \frac{1}{K} \sum_{j \in S_c} \mathbf{s}_j + \mathbf{x} + \mathbf{d}, \tag{4.1}$$

where all vectors have dimension N, K is the number of colluders, where $K \geq 1$ since each single copy is marked, and S_c indicates the colluder subset of size K, where $S_c \subseteq [1, \dots, n]$ and n is the total number of users. The fingerprints $\mathbf{s}_j$ are assumed to be orthogonal to each other, have equal energy, and be normally distributed. Due to the orthogonality of $\mathbf{s}_j$, we have $n \leq N$. We also assume the distortion $\mathbf{d}$ is an N-dimensional vector following an *iid* $\mathcal{N}(0, \sigma_d^2)$ distribution, and define the watermark-to-noise ratio as $\text{WNR} = 10\log_{10}(\|\mathbf{s}\|^2/\|\mathbf{d}\|^2)$. In this chapter, we will be concerned with detecting colluders, and we will study the collusion resistance performance of the fingerprinting system. Our detection scheme seeks to identify the colluders based on the observations $\mathbf{y}$. Since we assume a nonblind detection scenario in this chapter, the host signal $\mathbf{x}$ is always subtracted from $\mathbf{y}$. Because of the orthogonality of the basis $\{\mathbf{s}_j\}$, when performing detection it suffices to consider the correlator vector $\mathbf{T}_N$, where the jth component is given by

$$T_N(j) = \frac{(\mathbf{y}-\mathbf{x})^T\mathbf{s}_j}{\sqrt{\|\mathbf{s}_j\|^2}} \tag{4.2}$$

for $j = 1, \ldots, n$. It is straightforward to show that

$$p\big(T_N(j) \mid H_K, S_c\big) = \begin{cases} \mathcal{N}\left(\dfrac{\|\mathbf{s}\|}{K}, \sigma_d^2\right) & \text{if } j \in S_c, \\ \mathcal{N}\left(0, \sigma_d^2\right) & \text{otherwise,} \end{cases} \tag{4.3}$$

where H_K represents the hypothesis that there are K colluders, $\|\mathbf{s}\| = \|\mathbf{s}_j\|$ for all j due to the equal energy assumption, and each component $T_N(j)$ is independent of each other due to the orthogonality of $\mathbf{s}_j$.

In this section, we are interested in the theoretical collusion resistance of such fingerprinting systems. When studying the efficiency of a detection algorithm in collusion applications, appropriate criterion should be used to address the need of each specific application. The probability of a false negative and the probability of a false positive are popular criteria explored by researchers [69, 70]. From the detector (owner)'s point of view, a detection approach fails if either the detector fails to identify any of the colluders (a false negative) or the detector falsely indicates that an innocent user is a colluder (a false positive). Therefore, it is desirable to find an efficient detector that minimizes the probability of a false negative (P_{fn}), with a given probability of a false positive (P_{fp}). In general, P_{fp} should be exceptionally low, since a false positive may have severe consequences, such as serving as false testimony in a court of law. Though we consider the criteria P_{fp} and P_{fn} in this section, it is worth mentioning that other performance criteria also deserve consideration. We will present the study of two additional sets of criteria in Section 4.2.

Next we will introduce two other detection approaches and study their collusion resistance under the average attack.

4.1.1. The maximum detector

We have observed in collusion detection that the more colluders a detector aims to catch, the higher probability a false positive occurs. A detector designed to catch only one colluder should be capable of providing a smaller P_{fp}. A maximum detector is

$$T_{\max} = \max_{j=1}^{n} T_N(j), \tag{4.4}$$

where $T_N(j)$ as defined in (4.2), can be applied to catch one colluder with high confidence. This maximum detector should be compared to a threshold h chosen to yield the desired P_{fp}. Thus, we have the following test:

$$T_{\max} = \max_{j=1}^{n} T_N(j), \qquad \hat{j} = \begin{cases} \arg\max_{j=1}^{n} T_N(j) & \text{if } T_{\max} \geq h, \\ \varnothing & \text{if } T_{\max} < h, \end{cases} \tag{4.5}$$

where $\hat{j}$ indicates the index of the accused user, and $\hat{j} = \varnothing$ means that no accusation is made. In practice, it is possible that more than one j maximizes $T_N(j)$ simultaneously. In this case, the test randomly accuses one of these users. The following analysis reveals that the threshold h is determined by parameters including the length of the host signal N, the total number of users n, the number of colluders K, and the WNR.

Performance analysis. To analyze the detection performance of the maximum detector, we assume that the number of colluders K is known, and without loss of generality, we set the subset $S_c = [1, 2, \ldots, K]$, indicating that the first K users are colluders. We now have

$$\begin{aligned} P_{fp} &= P_r\{T_{\max} > h,\ \hat{j} \notin S_c\} \\ &= P_r\{T_1 < T_2,\ T_2 \geq h\} \\ &= P_r\{T_2 \geq h\}P_r\{T_1 < h\} + \int_h^{\infty} P_r\{T_2 \geq T_1\}p(T_1)\,dT_1 \end{aligned} \tag{4.6}$$

with the statistics $T_1 = \max_{j=1}^{K} T_N(j)$ and $T_2 = \max_{j=K+1}^{n} T_N(j)$. Here n is the total number of users, and $p(T_1)$ is the *pdf* of the random variable T_1. Clearly T_1 is independent of T_2 due to the independency of $T_N(j)$'s. We also define the detection probability P_d as

$$\begin{aligned} P_d &= 1 - P_{fn} \\ &= P_r\{T_{\max} > h,\ \hat{j} \in S_c\} = P_r\{T_1 > T_2,\ T_1 \geq h\} \\ &= P_r\{T_1 \geq h\}P_r\{T_2 < h\} + \int_h^{\infty} P_r\{T_1 \geq T_2\}p(T_2)\,dT_2. \end{aligned} \tag{4.7}$$

Since $p(T_N(j) \mid H_K, S_c)$ is given as in (4.3), we have

$$\begin{aligned} P_r(T_1 \leq t) &= \left(1 - Q\left(\frac{t - \|\mathbf{s}\|/K}{\sigma_d}\right)\right)^K, \\ P_r(T_2 \leq t) &= \left(1 - Q\left(\frac{t}{\sigma_d}\right)\right)^{n-K}, \end{aligned} \tag{4.8}$$

where the Q-function is defined as $Q(t) = \int_t^\infty (1/\sqrt{2\pi}) \exp(-x^2/2) dx$. The pdfs $p(T_1)$ and $p(T_2)$ can be derived correspondingly from the above cdf. Therefore, for a given small value of ϵ, we can numerically solve for h to yield $P_{fp} = \epsilon$ for different K, n, and WNR, and then numerically compute the corresponding P_d.

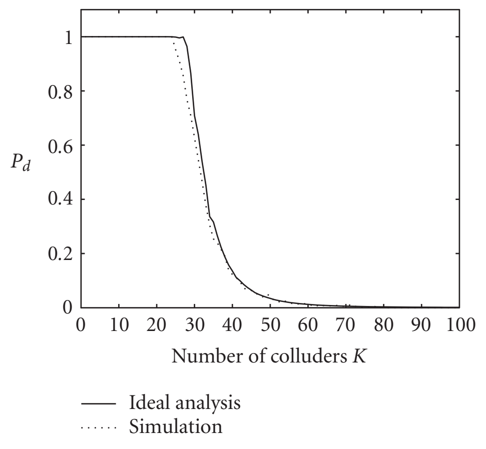
One important efficiency measure of a fingerprint detector is the maximum number of colluders that can be tolerated by a fingerprinting system with a total of n different N-point fingerprints. Specifically, with a given P_{fp}, we explore how many differently marked copies of the host signal are required for an averaging attack to generate a colluded copy from which no colluder's fingerprint can be detected with a high probability. A reasonably high P_d and a reasonably low P_{fp} are necessary to maintain the system's resistance to collusion.

We illustrate the resistance performance using an example, where WNR = 0 dB and the vector length is $N = 10^4$. Since 0 dB WNR corresponds to a *non-blind* scenario, the distortion **d** only consists of the additional additive noise. The variance σ_d^2 is assumed known and set to 1 for simplicity. In this example, the system requirements are expressed as $P_d \geq 0.8$ and $P_{fp} \leq 10^{-3}$. The symbol $K_{\max}$ represents the maximum number of colluders the fingerprinting system can successfully resist. In the examples shown in Figures 4.2a and 4.2b, when the number of users n is as high as 10^4, the fingerprinting system can resist up to 29 colluders; while, when n is set as a small number 75, the fingerprinting system can resist up to 75 colluders. It is also noted in Figure 4.2a that, if an attacker can collect 50 independent copies, the chance that the system can trace any original copy is only 4%. We note in Figure 4.2b that, as K increases, P_d first decreases slowly, then decreases quickly over the range $50 < K < 65$, and then increases. This behavior is determined by the expressions of P_{fp} and P_d in (4.6) and (4.8). We will give a similar explanation in Section 4.1.2, where a similar behavior is observed for the thresholding detector and the reason is more obvious. To have an overall understanding of the collusion resistance of this scheme, in Figure 4.3, we also plot the maximum resistible number of colluders $K_{\max}$ as a function of the total number of users n, under $N = 10^4$ and WNR = 0 dB. It is noted that the system can resist up to n colluders when the total number of users (fingerprints) n is less than 75. However, as a system accommodates more than 75 users, the collusion resistance of the system starts to decrease. For a system accommodating more than one thousand users, the maximum number of colluders that the system can handle is 30.

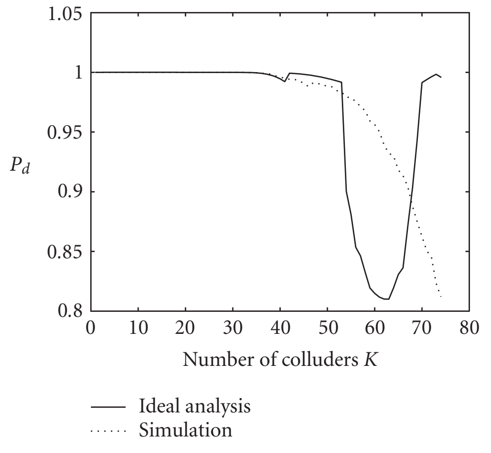
4.1.2. The thresholding detector

Although the goal of this section is to identify at least one of the colluders, from the content owner's point of view, it is beneficial to catch as many colluders as possible as long as we satisfy the P_{fp} requirement. We employ the traditional correlator $T_N(j)$ and compare it to a threshold h, and finally report that the jth fingerprint is present if $T_N(j)$ exceeds h. This simple approach is described as

$$\hat{\mathbf{j}} = \arg_{j=1,\ldots,n} \{T_N(j) \geq h\}, \tag{4.9}$$



(a)



(b)

FIGURE 4.2. Probability of detection P_d as a function of the number of colluders K when applying the maximum detector, with WNR = 0 dB, $N = 10^4$, and $P_{fp} \leq 10^{-3}$. In (a) the number of users $n = 10^4$; in (b) $n = 75$.

where the set $\hat{\mathbf{j}}$ indicates the indices of colluders, and an empty set means that no user is accused. Similar to the case of the maximum detector, the threshold h here is determined by such parameters as the document length N, the total number of users n, the number of colluders K, and the WNR.

Performance analysis. The threshold h in test (4.9) is chosen to yield $P_{fp} = \epsilon$, where ϵ is a desired small value. Same as in Section 4.1.1, to analyze the theoretical performance, we assume that the number of colluders K is known. And without

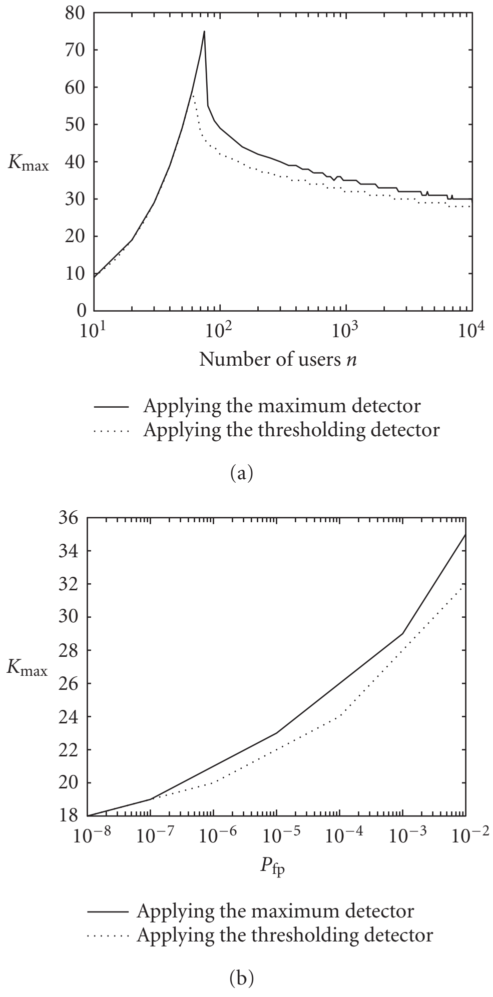


FIGURE 4.3. Collusion resistance of the orthogonal fingerprinting system to the averaging attack. Here WNR = 0 dB, $N = 10^4$, $\epsilon = 10^{-3}$, and $\beta = 0.8$. (a) $P_{fp} = 10^{-3}$ and $P_d = 0.8$. (b) $n = 10^4$ and $P_d = 0.8$.

loss of generality, we set the subset $S_c = [1, 2, \ldots, K]$. We now have

$$\begin{aligned} P_{fp} &= P_r\{\hat{\mathbf{j}} \cap \bar{S}_c \neq \varnothing\} = P_r\{T_2 \geq h\} \\ &= 1 - \left(1 - Q\left(\frac{h}{\sigma_d}\right)\right)^{n-K}, \end{aligned} \tag{4.10}$$

$$\begin{aligned} P_d &= 1 - P_{fn} = P_r\{\hat{\mathbf{j}} \cap S_c \neq \varnothing\} = P_r\{T_1 \geq h\} \\ &= 1 - \left(1 - Q\left(\frac{h - \|\mathbf{s}\|/K}{\sigma_d}\right)\right)^{K}, \end{aligned} \tag{4.11}$$

where $\bar{S}_c$ is the complement set of S_c, $T_1 = \max_{j \in S_c} T_N(j)$, $T_2 = \max_{j \in \bar{S}_c} T_N(j)$, and n the total number of users. Due to the independency among $T_N(j)$'s, T_1 is independent of T_2. The cdf's of the order statistics T_1 and T_2 are given as in (4.8). Therefore, according to (4.13), we can numerically calculate h to yield $P_{fp} = \epsilon$ with given K, n, and WNR, and then compute the corresponding P_d. Similar to the analysis in Section 4.1.1, our goal is to study the resistance of the fingerprinting system to averaging collusion when employing the thresholding detector (4.9). A sufficiently high P_d and a sufficiently low P_{fp} are required to make a fingerprinting system resistant to collusion attacks.

We illustrate the resistance performance using an example, where WNR = 0 dB, and $N = 10^4$. The variance σ_d^2 is set to 1 like before. The system requirements are defined as $P_d \geq 0.8$ and $P_{fp} \leq 10^{-3}$. As shown in Figures 4.9a and 4.9b, when the number of users n is in the order of 10^4, the fingerprinting system can resist up to 28 colluders; when n is set as a small number 75, the system can resist up to 46 colluders. Similar to Section 4.1.1, Figure 4.9 shows that P_d first decreases slowly, then decreases quickly, and then increases, as K increases. This behavior can be intuitively explained by the expressions of P_{fp} and P_d in (4.13). The sudden quick decrease is due to the exponential nature of the Q-function; when K is reasonably small, the term $\|\mathbf{s}\|/K$ in $Q(\cdot)$ function is the dominating factor in deciding P_d, this term decreases as K increases and therefore results in a decreasing P_d. On the other hand, when K is sufficiently large, the exponent term K is the dominating factor in deciding P_d, and thus P_d increases as K increases. To have an overall understanding of the collusion resistance of the orthogonal fingerprinting scheme, we plot the maximum resistible number of colluders $K_{\max}$ as a function of the total number of users n in Figure 4.3, where $N = 10^4$ and WNR = 0 dB. It is noted that the system can resist up to n colluders when the total number of users n is less than 60. However, for a system accommodating more than 60 users, its collusion resistance starts to decrease. For a system accommodating more than one thousand users, the number $K_{\max}$ is 28, meaning that the system requirements for the fingerprinting system is no longer met if the number of colluders is larger than 28.

We also compare the collusion resistance of the orthogonal fingerprinting scheme when applying both test (4.5) and test (4.9). Figure 4.3a shows $K_{\max}$ as a function of the total number of users n, with $N = 10^4$ and WNR = 0 dB. In we present $K_{\max}$ as a function of P_{fp} for a specific function of P_{fp} for a specific system with 10^4 users. We note that the maximum detector provides better performance than the thresholding detector. The intuitive explanation for this observation is that the maximum detector is designed to catch only one colluder. The overall difference is small, however, especially when the total number of users is large.

Lower and upper bounds of $K_{\max}$. We next provide analytic bounds on the maximum number of colluders $K_{\max}$ for an orthogonal fingerprinting system employing the thresholding detector. Since the above analysis is based on numerical computation, it does not provide an explicit understanding of the relationships between $K_{\max}$ and other system parameters, such as the sample length N, the WNR, the total number of users n, and the performance requirements of P_{fp} and P_d. To get more insight into the collusion-resistance of the thresholding detector, it is useful to study the analytic lower and upper bounds of $K_{\max}$. We begin by introducing two important lemmas.

Lemma 4.1. *Define the Gaussian tail integral as*

$$Q(t) = \int_t^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx. \tag{4.12}$$

$Q(t)$ is nonnegative for all t and monotonously decreases as t increases for $t > 0$. $Q(t) = 1 - Q(-t)$ by definition. This tail integral $Q(t)$ can be lower and upper bounded by

$$\begin{aligned} Q_a(t) &= \frac{1}{\sqrt{2\pi}t}\left(1 - \frac{1}{t^2}\right) \exp\left(-\frac{t^2}{2}\right) \\ &< Q(t) < \frac{1}{\sqrt{2\pi}t} \exp\left(-\frac{t^2}{2}\right) = Q_b(t) \end{aligned} \tag{4.13}$$

for $t > 0$, respectively. Please refer to [81] *for a detailed proof.*

Lemma 4.2. *Let n be a positive integer. For $0 < x < 1/n$, $(1-x)^n$ can be bounded by*

$$1 - nx < (1-x)^n < 1 - nx + \frac{n(n-1)}{2}x^2. \tag{4.14}$$

Proof. We first expand $(1-x)^n$ as

$$(1-x)^n = \sum_{i=0}^{n} \binom{n}{i} (-x)^{n-i}, \tag{4.15}$$

and utilizing the fact that $\binom{n}{i}x^i > \binom{n}{i+1}x^{i+1}$ for $0 < x < 1/n$, we derive the above inequality. □

Setting $\sigma_d^2 = 1$ for convenience, note that now $\|s\| = \sqrt{\eta N}$ with the WNR $\eta = \|s\|^2/\|d\|^2$. Recalling the expressions for P_{fp} and P_d in (4.13), we restate the system requirements as

$$\begin{aligned} P_{fp} &= 1 - \left(1 - Q(h)\right)^{n-K} \le \epsilon, \\ P_d &= 1 - \left(1 - Q\left(h - \frac{\sqrt{\eta N}}{K}\right)\right)^K \ge \beta, \end{aligned} \tag{4.16}$$

where ϵ is a small number and β is close to 1. For instance, a typical setting is $\epsilon = 10^{-3}$ and $\beta = 0.8$. A key step in determining $K_{\max}$ is to figure out the appropriate threshold h in (4.16). Though the explicit solution of h is hard to obtain, we can take advantage of the lower and upper bound of the threshold h by linking it to the lower and upper bounds of $K_{\max}$. We now provide the detailed derivation in the following, by using Lemmas 4.1 and 4.2.

Recall that N represents the sample length, n is the number of total users, and K is the number of colluders. Since we assume $\epsilon \ll 1$, meaning a false positive should be unlikely to occur, it immediately implies that the threshold h should yield $Q(h) < 1/(n-K)$ for a fingerprinting system accommodating n users. We provide an intuitive proof for this observation, defining

$$\gamma_j = \begin{cases} 1 & \text{if } j\text{th user is falsely accused,} \\ 0 & \text{otherwise,} \end{cases} \tag{4.17}$$

then the expectation of the number of innocents falsely accused is

$$E\left(\sum_{j\notin S_c} \gamma_j\right) = \sum_{j\notin S_c} E(\gamma_j) = \sum_{j\notin S_c} P_r\{\gamma_j = 1\} = (n-K)Q(h). \tag{4.18}$$

Thus if $Q(h) > 1/(n-K)$, then a false positive almost always happens, which is against our assumption. Therefore, it gives the observation $Q(h) < 1/(n-K)$. We further note that $\epsilon \ll 1$ and K is normally small compared to n, the assumption that ϵ is small implies that the choice of h can meet the condition $Q(h) \ll 1/n$. Therefore, it is fair to claim $Q(h) \ll 1/n$ in most situations. Since $Q(h) \ll 1/n$, it is safe to assume $h > 1$ due to the fact that $Q(1) \approx 1/6$ and that $Q(h)$ is a monotonously decreasing function for $h > 0$. We summarize these useful observations as

$$Q(h) < \frac{1}{n-K}; \quad Q(h) \ll \frac{1}{n}; \quad h > 1, \tag{4.19}$$

to help our derivation. By applying Lemmas 4.1 and 4.2, we have

$$\begin{aligned}
1-(n-K)Q(h) &< (1-Q(h))^{n-K} \\
&< 1-(n-K)Q(h) \\
&\quad + \frac{(n-K)(n-K-1)}{2}Q^2(h), \quad \text{by Lemma 4.2,} \\
(n-K)Q(h) - \frac{(n-K)(n-K-1)}{2}Q^2(h) &< P_{fp} = 1-(1-Q(h))^{n-K} < (n-K)Q(h), \\
(n-K)Q(h) &< (n-K)Q_b(h), \quad \text{by Lemma 4.1,} \\
(n-K)Q(h) - \frac{(n-K)(n-K-1)}{2}Q^2(h) &> (n-K)Q_a(h) - \frac{(n-K)(n-K-1)}{2}Q_b^2(h),
\end{aligned} \tag{4.20}$$

therefore, the following inequalities are observed:

$$P_{fp} < (n-K)Q_b(h),$$
$$P_{fp} > (n-K)Q_a(h) - \frac{(n-K)(n-K-1)}{2}Q_b^2(h). \tag{4.21}$$

The observations (4.19) could be used to find a lower bound for h. Since $h > 1$,

$$Q(h) < \frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{h^2}{2}\right) < \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{h^2}{2}\right). \tag{4.22}$$

Suppose we let the last term be equal to $1/n$,

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{h^2}{2}\right) = \frac{1}{n}, \quad \text{thus } h = \sqrt{\log\frac{0.5n^2}{\pi}} \triangleq h_{L1}, \tag{4.23}$$

the corresponding h_{L1} serves as a lower bound of the threshold to guarantee $Q(h) \ll 1/n$.

Recalling (4.21) and applying the lower bound h_{L1} result in

$$\begin{aligned} P_{fp} &< (n-K)Q_b(h) = (n-K)\frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{h^2}{2}\right) \\ &< (n-K)\frac{1}{\sqrt{2\pi}h_{L1}}\exp\left(-\frac{h^2}{2}\right) < n\frac{1}{\sqrt{2\pi}h_{L1}}\exp\left(-\frac{h^2}{2}\right). \end{aligned} \tag{4.24}$$

To provide $P_{fp} \le \epsilon$, we can require the last term to yield ϵ. It gives

$$n\frac{1}{\sqrt{2\pi}h_{L1}}\exp\left(-\frac{h^2}{2}\right) = \epsilon, \quad \text{thus } h = \sqrt{\log\left(\frac{n^2}{2\pi\epsilon^2\log\left(0.5n^2/\pi\right)}\right)} \triangleq h_H, \tag{4.25}$$

this h_H serves as an upper bound of the threshold h to guarantee $P_{fp} \le \epsilon$.

Since the tighter the bounds of the threshold h the better, we would like to further adjust the lower bound of h by considering (4.21) and the above upper bound h_H:

$$\begin{aligned} P_{fp} &> (n-K)Q_a(h) - \frac{(n-K)(n-K-1)}{2}Q_b^2(h) \\ &= (n-K)\frac{1}{\sqrt{2\pi}h}\exp\left(-\frac{h^2}{2}\right)\left(\left(1-\frac{1}{h^2}\right) - \frac{n-K-1}{2\sqrt{2\pi}h}\exp\left(-\frac{h^2}{2}\right)\right) \\ &> (n-K)\frac{1}{\sqrt{2\pi}h_H}\exp\left(-\frac{h^2}{2}\right)\left(\left(1-\frac{1}{h_{L1}^2}\right) - \frac{n-K-1}{2\sqrt{2\pi}h_{L1}}\exp\left(-\frac{h_{L1}^2}{2}\right)\right) \\ &= (n-K)\frac{1}{\sqrt{2\pi}h_H}\exp\left(-\frac{h^2}{2}\right)\left(1-\frac{1}{h_{L1}^2} - \frac{n-K-1}{2nh_{L1}}\right) \\ &> \frac{1}{\sqrt{2\pi}h_H}\exp\left(-\frac{h^2}{2}\right)\left(1-\frac{1}{h_{L1}^2} - \frac{1}{2h_{L1}}\right). \end{aligned} \tag{4.26}$$

By requiring the last term to be equal to ϵ, we will obtain a lower bound to satisfy $P_{fp} = \epsilon$ such that

$$h = \sqrt{2\log\left(\frac{2h_{L1}^2 - h_{L1} - 2}{2\sqrt{2\pi}\epsilon h_H h_{L1}^2}\right)} \triangleq h_{L2}. \tag{4.27}$$

By combining together the lower bounds in (4.23) and (4.27), we will determine a tighter lower bound as $h_L = \max\{h_{L1}, h_{L2}\}$. Therefore, it completes the derivation of the threshold bounds. In summary, we obtain a lower and upper bound of h as

$$\begin{aligned} h < h_H &= \sqrt{\log\left(\frac{n^2}{2\pi\epsilon^2 \log(0.5n^2/\pi)}\right)}, \\ h > h_L &= \max\{h_{L1}, h_{L2}\}, \end{aligned} \tag{4.28}$$

where the bounds are defined as $h_{L1} = \sqrt{\log(0.5n^2/\pi)}$, and

$$h_{L2} = \sqrt{2\log\left(\frac{2h_{L1}^2 - h_{L1} - 2}{2\sqrt{2\pi}\epsilon h_H h_{L1}^2}\right)}. \tag{4.29}$$

We would like to point out that the above derived $\{h_L, h_H\}$ is one, but not the only one, choice of bound pairs satisfying the inequalities in (4.16).

So far, we have obtained a lower and upper bound for the threshold h with a few reasonable assumptions. We now proceed to show that a lower and upper bound of the maximum number of colluders $K_{\max}$ can be obtained by using the bounds of h in (4.28) to evaluate the probability of accurate detection, P_d, in (4.16). The basic idea is to find a lower bound K_L of $K_{\max}$ such that the resulting pair (K_L, h_H) simultaneously satisfies the conditions that the corresponding P_d is larger than but close to the requirement β, and P_{fp} is smaller than but close to the requirement ϵ. Similarly, an upper bound K_H is chosen such that the pair (K_H, h_L) results in a P_d, which is smaller than but close to the requirement β, and a P_{fp}, which is larger than but close to the requirement ϵ. The smaller the difference between the two sets of results, the tighter the bounds represented by K_L and K_H. We now give a detailed derivation on the collusion resistance $K_{\max}$.

We repeat the formula of P_d in (4.16) as

$$P_d = 1 - \left(1 - Q\left(h - \frac{\sqrt{\eta N}}{K}\right)\right)^K \geq \beta, \tag{4.30}$$

where β is close to 1. We first show a lower bound of $K_{\max}$ tolerated by a Gaussian fingerprinting system with n users under some specific WNR η. The lower bound K_L must be chosen such that the pair (K_L, h_H) satisfies the probability requirements. Since the tail integral $Q(t)$ monotonously decreases as t increases, we observe that

$$h_H - \frac{\sqrt{\eta N}}{K} = 0, \qquad Q(0) = \frac{1}{2} \Longrightarrow \left(1 - Q\left(h_H - \frac{\sqrt{\eta N}}{K}\right)\right) = \frac{1}{2},$$
$$P_d = 1 - \left(1 - Q\left(h - \frac{\sqrt{\eta N}}{K}\right)\right)^K = 1 - \left(\frac{1}{2}\right)^K \longrightarrow 1. \tag{4.31}$$

If K is reasonably large, for instance $K = 4$, this gives $P_d = 15/16$ which is close to 1. Therefore $K = \sqrt{\eta N}/h_H$ serves as a loose lower bound

$$K_L = \frac{\sqrt{\eta N}}{h_H}. \tag{4.32}$$

We next find an upper bound K_H such that the pair (K_H, h_L) results in a smaller P_d than the requirement β, and a larger P_{fp} than the requirement ϵ. The smaller the gap, the tighter the bound. Similar as in the above observation, if the number of colluders $K \le \sqrt{\eta N}/h_L$, then the resulting $P_d \to 1$, thus the bound $K_{HL} = \sqrt{\eta N}/h_L$ actually is a lower bound of the upper bound K_H and we have $h_L - \sqrt{\eta N}/K > 0$ assumed for searching K_H. We further note that

$$P_d = 1 - \left(1 - Q\left(h_L - \frac{\sqrt{\eta N}}{K}\right)\right)^K < 1 - \left(1 - Q\left(h_L - \frac{\sqrt{\eta N}}{K}\right)\right)^n \tag{4.33}$$

since $(1 - Q(h_L - \sqrt{\eta N}/K)) \in (0,1)$ and $K \le n$ are assumed by definition. By setting the last term to be β, we obtain the solution

$$K = \frac{\sqrt{\eta N}}{h_L - Q^{-1}\left(1 - \sqrt[n]{1-\beta}\right)} \triangleq \widetilde{K}. \tag{4.34}$$

Clearly, this $\widetilde{K}$ can serve as an upper bound of the upper bound K_H. Therefore, we have

$$1 - \left(1 - Q\left(h_L - \frac{\sqrt{\eta N}}{K}\right)\right)^K < 1 - \left(1 - Q\left(h_L - \frac{\sqrt{\eta N}}{K}\right)\right)^{\widetilde{K}}, \tag{4.35}$$

and calculate the corresponding K_H via letting

$$1 - \left(1 - Q\left(h_L - \frac{\sqrt{\eta N}}{K}\right)\right)^{\widetilde{K}} = \beta, \qquad \text{thus } K = \frac{\sqrt{\eta N}}{h_L - Q^{-1}\left(1 - \sqrt[\widetilde{K}]{1-\beta}\right)} \triangleq K_H. \tag{4.36}$$

Clearly $P_d < \beta$ is met with this choice of K_H. Recall that as $K \leq n$ by definition, it is straightforward that

$$K_{\max} \geq \min\{n, K_L\}; \qquad K_{\max} \leq \min\{n, K_H\}. \tag{4.37}$$

In summary, we obtain the following collusion resistance:

$$\begin{aligned} K_{\max} &\geq \min\{n, K_L\} \quad \text{with } K_L = \frac{\sqrt{\eta N}}{h_H} = \sqrt{\frac{\eta N}{\log\left(n^2/2\pi\epsilon^2 \log\left(0.5n^2/\pi\right)\right)}}, \\ K_{\max} &\leq \min\{n, K_H\} \quad \text{with } K_H = \frac{\sqrt{\eta N}}{h_L - Q^{-1}\left(1 - \sqrt[\widetilde{K}]{1-\beta}\right)}, \end{aligned} \tag{4.38}$$

where $Q^{-1}(\cdot)$ represents the inverse Q-function, and $\widetilde{K}$ serves as an upper bound of K_H:

$$\widetilde{K} = \frac{\sqrt{\eta N}}{h_L - Q^{-1}\left(1 - \sqrt[n]{1-\beta}\right)}. \tag{4.39}$$

It is worth mentioning that, the bound K_H can be further tightened by letting $\widetilde{K} = K_H$, and then updating K_H according to (4.36) iteratively, until $\widetilde{K}$ is very close to K_H. Also, it is possible that a tighter lower and upper bound of $K_{\max}$ can be obtained by solving the one-dimensional problem $P_d = \beta$ when h_H and h_L are considered, respectively. However, this would require more computation and no explicit expressions of K_H and K_L, as would be available in (4.38), due to the complex nature of P_d. In addition, though the bounds (4.38) are derived for the thresholding detector, they are also applicable to the maximum detector since, as shown in Figure 4.3, the overall performance difference between these two schemes is small and can be neglected.

We illustrate the resistance analysis in Figure 4.4, where $\sigma_d^2 = 1$, WNR $= 0$ dB, and $N = 10^4$. Setting the requirements $P_{fp} \leq 10^{-3}$ and $P_d \geq 0.8$, we plot the lower and upper bounds of $K_{\max}$ versus the number of users n, along with the numerical result $K_{\max}$. It is noted that the lower and upper bounds are within a factor of 2 of the true value of $K_{\max}$. Given the lower and upper bounds, some interesting observations are noted from this example. From the attacker point of view, if an attacker can only collect up to 20 copies, he/she can never succeed in removing all trace of the fingerprints; however, an attacker is guaranteed success if 80 independent copies are available. From the owner (detector) point of view, if the owner has a means to ensure that a potential attacker has no way to obtain 20 or more independent copies, the fingerprinting system is essentially collusion-resistant. Further, in order to maximize the worst case of P_d, the owner should limit the number of independent distributions. For instance, if the number of independent copies is less than 60, the system is also collusion-resistant.

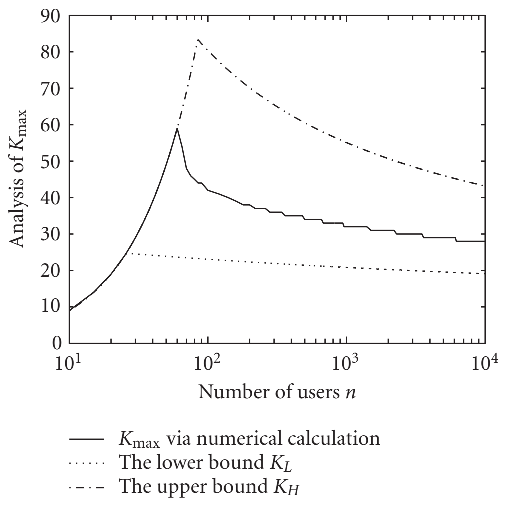


FIGURE 4.4. The lower and upper bounds of $K_{\max}$ as a function of the number of users n when applying the thresholding detector in (4.9). Here WNR = 0 dB, $N = 10^4$, $\epsilon = 10^{-3}$, and $\beta = 0.8$.

4.2. Extensions to other performance criteria

In Section 4.1, we were concerned with capturing one true colluder with high confidence. The motivating application was to provide digital evidence in the court of law. However, different goals arise under different situations, and there are other possible performance measures for colluder identification. These measures place a varying amount of emphasis on capturing colluders and placing innocents under suspicion. In fact, colluder identification might only be one component of the evidence gathering process. Since the final decision will depend upon many types of evidence, there might be different roles that collusion detection will play in protecting content value. For example, it might be desirable to use colluder identification to identify a set of suspects and then perform other types of surveillance on these suspects to gather the remaining evidence. This suggests that researchers should consider a wider spectrum of performance measures.

We consider two additional sets of performance criteria in this section and study the thresholding detector under the average attack. The analysis of the thresholding detector is easier than the maximum detector. However, the results are similar, and for that reason we will omit the analysis of the maximum detector.

Case 1 (capture more). This set of performance criteria consists of the expected fraction of colluders that are successfully captured, denoted by r_c, and the expected fraction of innocent users that are falsely placed under suspicion, denoted by r_i. Here the major concern is to catch as many colluders as possible, though potentially at a cost of accusing more innocents. The balance between capturing colluders and placing innocents under suspicion is represented by these two expected fractions. We define

$$\gamma_j = \begin{cases} 1 & \text{if } j\text{th user is accused,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.40}$$

Considering the thresholding detector and the average attack, we have

$$\begin{aligned} r_c &= \frac{E\left(\sum_{j\in S_c}\gamma_j\right)}{K} = \frac{\sum_{j\in S_c} P_r\{\gamma_j = 1\}}{K} \\ &= \frac{KQ((h-\|s\|/K)/\sigma_d)}{K} = Q\left(\frac{h-\|s\|/K}{\sigma_d}\right), \\ r_i &= \frac{E\left(\sum_{j\notin S_c}\gamma_j\right)}{n-K} = Q\left(\frac{h}{\sigma_d}\right). \end{aligned} \tag{4.41}$$

The above observation indicates that studying the behavior of the fractions r_c and r_i is equivalent to studying the probability of correctly detecting a specific colluder and the probability of falsely accusing a specific innocent user. Based on this pair $\{r_c, r_i\}$, now the system requirements are

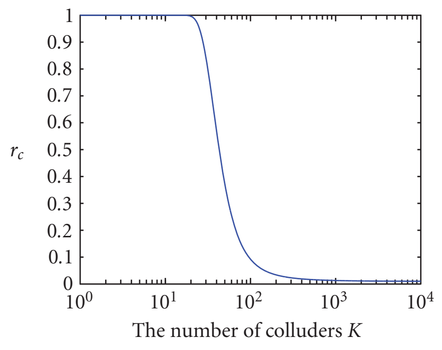
$$r_i = Q\left(\frac{h}{\sigma_d}\right) \le \alpha_i, \qquad r_c = Q\left(\frac{h-\|s\|/K}{\sigma_d}\right) \ge \alpha_c, \tag{4.42}$$

meaning a reasonably high r_c and a reasonably low r_i are required to keep the fingerprinting system safe from attacks.

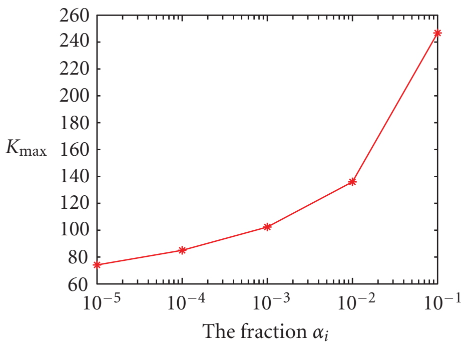
We now study the resistance performance of orthogonal fingerprints under requirements (4.42). In our analysis, $\|s\| = \sqrt{\eta N}\sigma_d$ with η being the WNR and N being the vector length. Based on (4.41) and (4.42), we can obtain the threshold h and the maximum number of colluders $K_{\max}$ as

$$\begin{aligned} h &= Q^{-1}(\alpha_i)\sigma_d, \\ K_{\max} &= \frac{\sqrt{\eta N}}{Q^{-1}(\alpha_i) - Q^{-1}(\alpha_c)}. \end{aligned} \tag{4.43}$$

It is interesting to note that the threshold h is a constant value determined by α_i, and $K_{\max}$ is not affected by the total number of users n. The collusion resistance $K_{\max}$ is proportional to the square root of the vector length N and the WNR η. To illustrate this, in Figure 4.5a, we observe that a system with the requirements $r_i \le 0.01$ and $r_c \ge 0.5$, which involves $N = 10^4$ fingerprints, can withstand 43 colluders. If we allow a larger fraction of innocents to be placed under suspicion, then the system can resist more colluders, as depicted in Figure 4.5b. Here, let us look at an example represented by the point with coordinate values $\{10^{-2}, 136\}$ in Figure 4.5b. In this example, since $N = 10^5$, $\alpha_i = 10^{-2}$, and $\alpha_c = 0.5$, the decision maker will have to identify 68 suspected colluders (calculated as $136 \times \alpha_c$) from a pool of people containing up to one thousand innocent users (calculated as $N \times \alpha_i$).



(a)



(b)

FIGURE 4.5. The resistance performance under the criteria r_c and r_i, when applying the thresholding detector in (4.9). (a) We plot the expected fraction r_c versus the number of colluders K, with $N = 10^4$, the WNR $\eta = 1$, and the expected fraction $r_i = 0.01$. (b) $K_{\max}$ under different requirements of α_i with $N = 10^{-5}$, the WNR $\eta = 1$, and $d_c = 0.5$.

Case 2 (capture all). This set of performance criteria consists of the efficiency rate R, which describes the expected number of innocents accused per colluder, and the probability of capturing all K colluders, which we denote by P_d. Here the goal is to capture all colluders with a high probability. The tradeoff between capturing colluders and placing innocents under suspicion is managed through the adjustment of the efficiency rate R. More specifically, when considering the thresholding detector and the average attack, we have

$$R = \frac{E\left(\sum_{j\notin S_c} \gamma_j\right)}{E\left(\sum_{j\in S_c} \gamma_j\right)} = \frac{(n-K)Q(h/\sigma_d)}{KQ((h-\|s\|/K)/\sigma_d)},$$
$$P_d = P_r\{S_c \subseteq \hat{\mathbf{j}}\} = P_r\left\{\min_{j\in S_c} T_N(j) \geq h\right\} = Q\left(\frac{h-\|s\|/K}{\sigma_d}\right)^K. \tag{4.44}$$

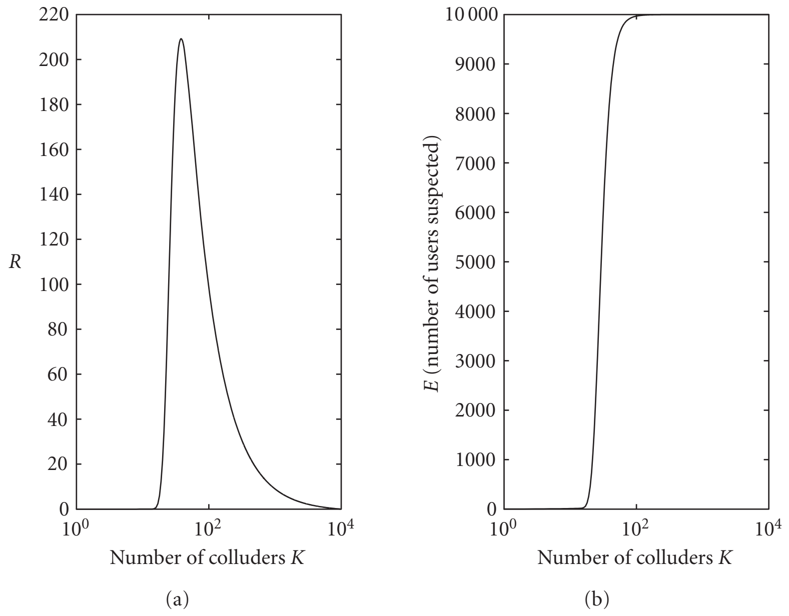


FIGURE 4.6. The behavior of the efficiency rate R and the expected number of users suspected as K increases. Here $N = 10^4$, $\eta = 1$, $n = 10^4$, and $P_d = 0.99$. (a) We plot the rate R versus the number of colluders K. (b) The expected number of users suspected is plotted against K.

Based on this pair $\{R, P_d\}$, the system requirements are expressed as

$$R \leq \alpha; \qquad P_d \geq \beta. \tag{4.45}$$

We first illustrate the resistance performance of the fingerprinting system under these requirements by examples, where $N = 10^4$ and $\eta = 1$. We set $\sigma_d^2 = 1$ for simplicity and recall that $\|s\| = \sqrt{\eta N}$. First, for a system accommodating as many as 10^4 users and requiring $P_d = 0.99$, we study the behavior of R when the number of colluders K increases as shown in Figure 4.6. For each choice of K, the threshold h is chosen to yield $P_d = 0.99$ and then the corresponding R is calculated. It is clear that almost all users will be placed under suspicion if more than 100 users come together and perform the collusion. The decision of placing all users under suspicion certainly provides no useful clues to the identity of the colluders. If the rate R is set as 0.01, the system can resist up to 13 colluders. To obtain an overall understanding of the collusion resistance of the system, we further study the performance of the system when different amounts of users are involved (as illustrated in Figure 4.7) by requiring $R \leq 0.01$ and $P_d \geq 0.99$. It is clear that the system can afford up to n colluders if the number of total users n is smaller than 21. The resistance performance degrades when more than 21 users are accommodated. In

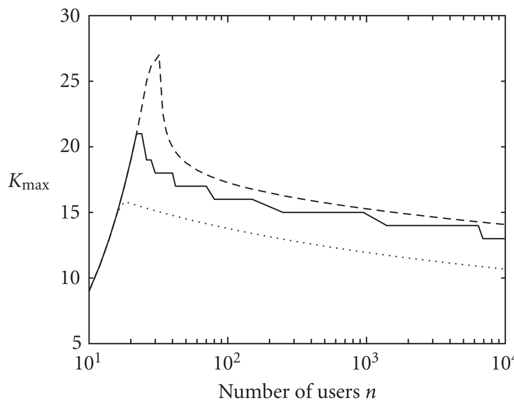


FIGURE 4.7. Resistance performance of the orthogonal fingerprinting system under the criteria R and P_d. Here $N = 10^4$, $\eta = 1$, $\alpha = 0.01$, and $P_d = 0.99$. The lower and upper bounds are also plotted.

situations where the system is required to distribute more than one thousand independently marked copies, an attacker having as few as 15 independent copies has the capability to break down the system.

Similar to Section 4.1.2, we provide a lower and upper bound of $K_{\max}$ under this set of criteria. Assume $\sigma_d^2 = 1$ for convenience. A derivation similar to that in Section 4.1.2 leads to the following bounds:

$$K_{\max} \geq \min\{n, K_L\} \quad \text{with } K_L = \frac{\sqrt{\eta N}}{Q^{-1}(2\alpha/n) - Q^{-1}\left(\sqrt[n]{\beta}\right)}, \tag{4.46}$$

$$K_{\max} \leq \min\{n, K_H\} \quad \text{with } K_H = \frac{\sqrt{\eta N}}{Q^{-1}(\alpha\widetilde{K}/(n-\widetilde{K})) - Q^{-1}\left(\sqrt[K_L]{\beta}\right)}, \tag{4.47}$$

with $\widetilde{K}$ being

$$\widetilde{K} = -\frac{\sqrt{\eta N}}{Q^{-1}\left(\sqrt[K_L]{\beta}\right)}. \tag{4.48}$$

The details of this derivation are omitted due to the limitation of space and due to its similarity to the derivations in Section 4.1.2. An example is given in Figure 4.7.

The analysis in this section reveals that the maximum number of colluders allowed by a Gaussian fingerprinting system is on the same order, under three different sets of criteria. Basically, a few dozen colluders could break down the orthogonal Gaussian fingerprinting system by generating a new composite copy such that the identification of the original fingerprints would unlikely be successful.

4.3. Extensions to other types of attacks

So far, we have studied the collusion resistance of the Gaussian fingerprinting system for the average attack. When an attacker has access to multiple independently watermarked copies of the same host signal, attacks other than the averaging attack are also possible. In this section, we consider several nonlinear attacks suggested by Stone in [62], and we evaluate the resistance of the maximum detector and the thresholding detector. We have further considered a few other collusion attacks (see Chapter 3), such as randomly copying and pasting parts of content from individual copies, or randomly choosing any value between the minimum and the maximum values. Our study has shown that this additional set of attacks can be approximated as the collusion attacks discussed in this chapter followed by additive noise. Thus the attacks studied here represent a wide range of attacks.

(i) Attacks based on the median operation: under this attack, the attacker obtains K independently marked copies of the same host signal, and computes the composite observation $\mathbf{y}$ such that the ith component of y is

$$\begin{aligned} y(i) &= \text{median}_{j\in S_c}\{x(i)+s_j(i)\}+d(i) \\ &= \text{median}_{j\in S_c}\{s_j(i)\}+x(i)+d(i) \end{aligned} \tag{4.49}$$

for $i = 1, 2, \ldots, N$, where the subset S_c indicates the colluder index and median$(\cdot)$ represents the median operation. This attack is named the *median* attack, as indicated by its definition.

(ii) Attacks based on the minimum operations: under the *minimum* attack, the attacker creates a copy $\mathbf{y}$ whose ith component is the minimum of the ith components of the observed copies plus a noise term. Similarly, we can define the *maximum* attack and the so-called *randomized negative* attack (also referred as Kilian's attack) [60]. Since our statistical analysis reveals that these three attacks share the same property in terms of collusion resistance, we study only the minimum attack here to save space.

(iii) Attacks based on the average of the minimum and maximum operations: under the *minmax* attack, the attacker creates a copy $\mathbf{y}$ whose ith component is

$$\begin{aligned} y(i) &= \frac{\min_{j\in S_c}\{x(i)+s_j(i)\}+\max_{j\in S_c}\{x(i)+s_j(i)\}}{2}+d(i) \\ &= \frac{\min_{j\in S_c}\{s_j(i)\}+\max_{j\in S_c}\{s_j(i)\}}{2}+x(i)+d(i) \end{aligned} \tag{4.50}$$

for $i = 1, 2, \ldots, N$, where min$(\cdot)$ and max$(\cdot)$ are the minimum and maximum operations, respectively.

(iv) Attacks based on the median, minimum, and maximum operations: since Kilian's attack produces unacceptable distortion, Stone suggested a modified version of Kilian's attack such that

$$\begin{aligned} y(i) &= \left(\min_{j\in S_c} \{x(i)+s_j(i)\} + \max_{j\in S_c} \{x(i)+s_j(i)\} - \text{median}_{j\in S_c}\{x(i)+s_j(i)\}\right) + d(i) \\ &= \left(\min_{j\in S_c} \{s_j(i)\} + \max_{j\in S_c} \{s_j(i)\} - \text{median}_{j\in S_c}\{s_j(i)\}\right) + x(i) + d(i) \end{aligned} \tag{4.51}$$

for $i = 1, 2, \ldots, N$. It is noted that Stone's attack produces less distortion than Kilian's.

For a specific attack, we should examine the overall distortion introduced to the host signal, and the efficiency comparison of different attacks should be carried out under the assumption that the distortion level created by different attacks is approximately equal. The purpose of this section is to show that the nonlinear attacks described above can be regarded as attacks by averaging in the sense that they yield pretty similar performance when employing the maximum and the thresholding detectors, as long as the overall MSE (mean-square-error) introduced to the host signal by different attacks is the same. More specifically, our goal is to demonstrate that the attacks

$$\begin{aligned} \mathbf{y}_g &= g(\mathbf{y}_j,\ j \in S_c) + \mathbf{d}_g, \\ \mathbf{y}_{\text{mean}} &= \frac{1}{K}\sum_{j\in S_c} \mathbf{y}_j + \mathbf{d}_{\text{mean}} \end{aligned} \tag{4.52}$$

provide close collusion resistance performance as long as

$$E\left\{||\mathbf{y}_g - \mathbf{x}||^2\right\} = E\left\{||\mathbf{y}_{\text{mean}} - \mathbf{x}||^2\right\} \triangleq \xi_0, \tag{4.53}$$

where $g(\cdot)$ represents the attack operation, and the additive noise $\mathbf{d}_g$ are $\mathcal{N}(0, \sigma_{d,g}^2)$ distributed where the variance $\sigma_{d,g}^2$ is determined by the power ξ_0. Note that the power of the composite observation indicates the level of MSE introduced to the host signal. Therefore, given the MSE level allowed by the system, we want to show that the underlying attack model does not matter from the detector point of view. In other words, we want to demonstrate that the thresholding detector is robust to different attacks. A similar argument can be made for the maximum detector.

First, we illustrate an example based on 10^4 simulation runs in Figure 4.8, where $N = 10^4$, $n = 100$, and thresholds are chosen to yield $P_{fp} = 10^{-2}$. Three types of attacks are studied: the average, minmax, and minimum attacks. The fingerprints $\mathbf{s}_j$'s are taken as $\mathcal{N}(0, \sigma_s^2)$ distributed random values with $\sigma_s^2 = 1$, and the additive noise added to the minimum attack $\mathbf{d}_{\text{min}}$ follows $\mathcal{N}(0, 1)$ distribution. Thus, the additive noises introduced by the average attack and the minmax attack are correspondingly generated to provide the same MSE level as by the minimum attack. From Figure 4.8, it is noted that the performance curves are close to each other, with the minimum attack marginally superior to the other two attacks from the detector's point of view (i.e., worse from the attacker's point of view).

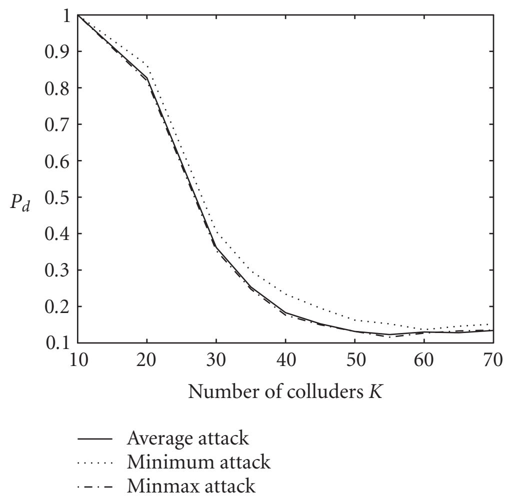


FIGURE 4.8. The probability of detection as a function of the number of colluder K under different attacks, when applying the thresholding detector and the same MSE level is introduced. Here $N = 10^4$, $n = 100$, and $P_{fp} = 10^{-2}$.

The observation noted in the above example is encouraging. We intuitively explain the reasons by referring to the statistical analysis in [60, 82]. We need to analyze the statistical behavior of the test $T_N(j)$ under different collusion attacks. Due to the *iid* Gaussian assumption of the fingerprint components and since N is generally in the order of 10^4 for 256×256 images, by applying the central limit theorem (CLT), we propose to approximate the distribution of $T_N(j)$ with a Gaussian distribution. Our results show that the correlator $T_N(j)$ still yields zero mean for $j \notin S_c$, and the mean of $T_N(j)$, for $j \in S_c$, is the same under different attacks. By calculating the corresponding mean and variance, we have that the correlator $T_N(j)$ is approximately distributed as

$$T_N(j) \mid K, S_c, g \sim \begin{cases} \mathcal{N}\left(0, \sigma_{g0}^2 + \sigma_{d,g}^2\right) = \mathcal{N}\left(0, \dfrac{E\left\{\|\mathbf{y}_g - \mathbf{x}\|^2\right\}}{N}\right) & \text{if } j \notin S_c, \\ \mathcal{N}\left(\dfrac{\sqrt{N\sigma_s^2}}{K}, \sigma_{g1}^2 + \sigma_{d,g}^2\right) & \text{if } j \in S_c, \end{cases} \tag{4.54}$$

in which for all $l \in S_c$,

$$\begin{aligned} \sigma_{g0}^2 &= E\left\{g\left(s_j(i),\ j \in S_c\right)^2\right\}, \\ \sigma_{g1}^2 &= \frac{\operatorname{Var}\left\{g\left(s_j(i),\ j \in S_c\right)s_l(i)\right\}}{\sigma_s^2}. \end{aligned} \tag{4.55}$$

Under each attack, $T_N(j)$, for $j \notin S_c$, is independent of each other. It is clear that for a given K, the behavior of $T_N(j)$, for $j \notin S_c$, is fully characterized by the overall power ξ_0, therefore, the threshold and P_{fp} are not affected by the type of attack. For approximating the distribution of $T_N(j)$ with a Gaussian distribution, we need to calculate the equivalent mean and variance. As one example, we provide the derivation of the mean and the variance σ_{g1}^2 under the minimum attack in the following.

Denote the pdf of each Gaussian fingerprint component as $f(x)$, that is, $f(x) = \mathcal{N}(0, \sigma_s^2)$, and the cdf as $F(x)$. Now under the minimum attack, the correlator $T_N(j)$ is

$$T_N(j) = \frac{1}{\|\mathbf{s}\|} \sum_{i=1}^{N} \Big(\min_{l \in S_c} \{s_l(i)\} + d_{\min}(i) \Big) s_j(i). \tag{4.56}$$

Define $s_{\min}(i) = \min_{l \in S_c}\{s_l(i)\}$, we have the pdf of $s_{\min}(i)$ as

$$f_{\min}(x) = Kf(x)[1 - F(x)]^{K-1}. \tag{4.57}$$

For $j \notin S_c$, it is easy to show that $E\{T_N(j)\} = 0$. For $j \in S_c$, we can express the joint pdf of $s_{\min}(i)$ and $s_j(i)$ as

$$\begin{aligned} &f_{\min,1}(s_{\min}(i) = x', s_j(i) = x) \\ &\quad = \begin{cases} f(x')[1 - F(x')]^{K-1} & \text{if } s_{\min}(i) = s_j(i), \\ (K-1)f(x')f(x)[1 - F(x')]^{K-2} & \text{if } s_{\min}(i) < s_j(i). \end{cases} \end{aligned} \tag{4.58}$$

By employing the rule of integration by parts, we have

$$\begin{aligned} E\{s_{\min}(i)s_j(i)\} &= \int_{-\infty}^{\infty} x'^2 f(x')[1 - F(x')]^{K-1} dx' \\ &\quad + \int_{-\infty}^{\infty} x'(K-1)f(x')[1 - F(x')]^{K-2} \Big(\int_{x'}^{\infty} x f(x) dx \Big) dx' \\ &= \sigma_s^2 \int_{-\infty}^{\infty} f(x')[1 - F(x')]^{K-1} dx' \\ &= \frac{\sigma_s^2}{K} \int_{-\infty}^{\infty} f_{\min}(x') dx' = \frac{\sigma_s^2}{K}. \end{aligned} \tag{4.59}$$

It is clear that, for $j \in S_c$, the mean of $T_N(j)$ under the minimum attack is the same as that of the average attack. We can calculate $E\{(s_{\min}(i)s_j(i))^2\}$ and $\operatorname{var}\{s_{\min}(i)s_j(i)\}$ numerically. Therefore, we can calculate the mean and variance

TABLE 4.1. The corresponding σ_{g0}^2, σ_{g1}^2, $\sigma_{d,g}^2$, and $\mathrm{var}\{T_N(j)\}$ under different attacks, where $K = 15$, $\sigma_s^2 = 1$.

Variance\attack	Average	Minimum	Median	Minmax	Stone
σ_{g0}^2	0.0667	3.3144	0.1017	0.1581	0.5757
$\sigma_{d,g}^2$	3.2477	0	3.2127	3.1563	2.7387
σ_{g1}^2	0.0711	3.7519	0.1108	0.1747	0.6480
$\mathrm{var}\{T_N(j)\}$, $j \in S_c$	3.3188	3.7519	3.3235	3.3310	3.3867

of $T_N(j)$ correspondingly as

$$\begin{aligned} E\{T_N(j)\} &= \frac{N}{\|\mathbf{s}\|}E\{s_{\min}(i)s_j(i)\} = \frac{\sqrt{N\sigma_s^2}}{K}, \\ \mathrm{var}\{T_N(j)\} &= \frac{\mathrm{var}\{s_{\min}(i)s_j(i)\}}{\sigma_s^2}. \end{aligned} \tag{4.60}$$

The analysis of other attacks can be similarly derived. We refer the interested readers to [82] for more details. It is worth mentioning that there is no closed form expression for the variance σ_{g1}^2 available under most attacks, due to the existence of $Q(\cdot)$ terms in the distributions. Therefore, in our implementation we numerically evaluate the integrals by employing the recursive adaptive Simpson quadrature method. As an example, suppose $\sigma_s^2 = 1$ and no noise is added to the minimum attack, we report the results in Table 4.1, where we can see that the variance of $T_N(j)$, for $j \in S_c$, is comparable under different attacks and thus results in comparable P_d's under different attacks. Our results also reveal that the difference of this variance among different attacks gets smaller as the number of colluders K increases.

The above fact that different attacks provide comparable performance from the detector's point of view suggests, for the same MSE distortion, that the average attack is the most efficient from the attacker point of view. This is because, from the detector point of view, there exist better detection schemes than detectors based on the correlators $T_N(j)$'s for attacks other than the average attack. For this reason, we have concentrated only on the average attack in this chapter, and we only address the collusion resistance of a fingerprinting system under the average attack.

In addition, to maintain an acceptable quality of the image, a basic requirement is that the collusion attack is unlikely to generate noticeable distortion. Therefore, three types of attacks, namely the minimum attack, the maximum attack, and Kilian's attack, should be excluded from consideration, since our analysis indicates that the energy of the composite watermark generated by these attacks is greater than that of the original watermark (e.g., large σ_{g0}^2 and σ_{g1}^2), and grows with the number of colluders K. This unfortunate feature of these attacks suggests that these attacks are likely to produce noticeable distortion which increases with K.

4.4. A practical estimator for the amount of colluders

In the above analysis we have assumed that the number of colluders K is known. However, knowledge of K is normally not available in a practical collusion scenario. Therefore, in real colluder-identification situations, we need to estimate the number of colluders K. To start, we present the problem in a multiple-hypotheses-testing framework, where the different hypotheses lead to different $\mathbf{y}$ as

$$H_K : \mathbf{y} = \frac{1}{K} \sum_{j \in S_c} \mathbf{s}_j + \mathbf{x} + \mathbf{d} \tag{4.61}$$

for $1 \leq K \leq n$. An optimal way to estimate K can be based on the Bayesian classifier

$$\hat{K} = \arg\max_K p(H_K \mid \mathbf{y}) = \arg\max_K \sum_{S_c} p(H_K, S_c \mid \mathbf{y}), \tag{4.62}$$

where $p(\cdot)$ represents likelihood functions. However, it is immediately noted that the probability summation over all subsets S_c with size K is infeasible in practice. We address this issue by obtaining the maximum-likelihood (ML) estimates of K and S_c jointly based on the observations $\mathbf{y}$:

$$(\hat{K}, \hat{S}_c) = \arg\max_{K,S_c} p(\mathbf{y} \mid H_K, S_c). \tag{4.63}$$

Because of the orthogonality of the basis $\{\mathbf{s}_j\}$, it suffices to consider the correlator vector $\mathbf{T}_N$, defined in (4.2). Now the estimator is equal to

$$(\hat{K}, \hat{S}_c) = \arg\max_{K,S_c} p(\mathbf{y} \mid H_K, S_c) = \arg\max_{K,S_c} p(\mathbf{T}_N \mid H_K, S_c). \tag{4.64}$$

By introducing an additional dummy class H_0 as $p(\mathbf{T}_N \mid H_0) = \mathcal{N}(0, \sigma_d^2 \mathbf{I}_n)$, we have

$$\begin{aligned}
&(\hat{K}, \hat{S}_c) \\
&= \arg\max_K \left\{ \max_{S_c} p(\mathbf{T}_N \mid H_K, S_c) \right\} \\
&= \arg\max_K \left\{ \max_{S_c} \frac{p(\mathbf{T}_N \mid H_K)}{p(\mathbf{T}_N \mid H_0)} \right\} \\
&= \arg\max_K \left\{ \max_{S_c} \frac{\Pi_{j \in S_c} \left(\frac{1}{\sqrt{2\pi}\sigma_d} \right) \exp\left(- \frac{(T_N(j) - \|\mathbf{s}\|/K)^2}{2\sigma_d^2} \right) \Pi_{j \notin S_c} \left(\frac{1}{\sqrt{2\pi}\sigma_d} \right) \exp\left(\frac{-T_N(j)^2}{2\sigma_d^2} \right)}{\Pi_{j=1}^n \left(\frac{1}{\sqrt{2\pi}\sigma_d} \right) \exp\left(- \frac{T_N(j)^2}{2\sigma_d^2} \right)} \right\} \\
&= \arg\max_K \left\{ \max_{S_c} \sum_{j \in S_c} \left(\frac{2\|\mathbf{s}\|}{K} T_N(j) - \frac{\|\mathbf{s}\|^2}{K^2} \right) \right\} \quad \text{by applying log operation} \\
&= \arg\max_K \left\{ \max_{S_c} \frac{2\|\mathbf{s}\|}{K} \sum_{j \in S_c} T_N(j) - \frac{\|\mathbf{s}\|^2}{K} \right\},
\end{aligned} \tag{4.65}$$

thus

$$\begin{aligned}\hat{K} &= \arg\max_K \left\{ \frac{2\|\mathbf{s}\|}{K} \sum_{j\in\hat{S}_c} T_N(j) - \frac{\|\mathbf{s}\|^2}{K} \right\}\\ &= \arg\max_K \left\{ \frac{2\|\mathbf{s}\|}{K} \sum_{j=1}^{K} T_N^{(j)} - \frac{\|\mathbf{s}\|^2}{K} \right\},\end{aligned} \tag{4.66}$$

where $T_N^{(j)}$'s are ordered as $T_N^{(1)} \ge T_N^{(2)} \ge \cdots \ge T_N^{(n)}$. The last equation is due to the ML estimate

$$\begin{aligned}\hat{S}_c &= \arg\max_{|S_c|=K} \left\{ \frac{2\|\mathbf{s}\|}{K} \sum_{j\in S_c} T_N(j) - \frac{\|\mathbf{s}\|^2}{K} \right\}\\ &= \arg\max_{|S_c|=K} \sum_{j\in S_c} T_N(j) = \text{the index of } K \text{ largest } T_N(j).\end{aligned} \tag{4.67}$$

Therefore, in summary, we have

$$\begin{aligned}\hat{K} &= \arg\max_K \left\{ \frac{2\|\mathbf{s}\|}{K} \sum_{j=1}^{K} T_N^{(j)} - \frac{\|\mathbf{s}\|^2}{K} \right\},\\ \hat{S}_c &= \text{the set of indices of } \hat{K} \text{ largest } T_N(j)\text{'s}.\end{aligned} \tag{4.68}$$

Based on $(\hat{K}, \hat{S}_c)$ obtained from the above approach, a fingerprinting system may accuse all users indicated by $\hat{S}_c$ as colluders. However, the above approach is aimed at jointly finding the ML estimates of K and the colluder set S_c. Although it might be interesting to study P_{fn} and P_{fp} of this approach in (4.68), this approach is not designed to allow one to adjust the tradeoff between P_{fn} and P_{fp}, which is a desirable functionality for colluder tracing applications. Therefore, the above approach is not appropriate to meet our specific detection goal. Thus we only use it to estimate the total number of colluders K, and examine the effects of the estimated K next.

Simulations for the maximum detector. Since K is unknown in a practical collusion scenario, we need to estimate K first before setting a suitable threshold h for the detection process. With given N, WNR, and n, the colluder identification algorithm using the maximum detector becomes as follows.

(1) Estimate the number of colluders K via (4.68).
(2) Determine the threshold h correspondingly to yield a desired P_{fp}, according to (4.6). It is clear that the threshold h is only a function of $\hat{K}$ when N, WNR, and n are given.
(3) Apply the maximum test statistic described in (4.5) and return the index $\hat{j}$.

In Figures 4.2a and 4.2b for $N = 10^4$ and WNR $= 0$ dB, the simulation results are compared to the ideal performance analysis shown in Section 4.1.2 where K is assumed known. Unlike the ideal case where K is assumed known, when K is estimated based on simulated observations, the resulting P_d always decreases with the increasing of K. Good match is observed over the nonincreasing part of the ideal case, that is, when K is small. Mismatch is noted over the increasing part of the ideal case, that is, when K is close to n, since K is underestimated in this situation due to the increasing overlap between the two Gaussian distributions $\mathcal{N}(0, \sigma_d^2)$ and $\mathcal{N}(\|\mathbf{s}\|/K, \sigma_d^2)$ as K increases. However, using an estimate of K will not alter $K_{\max}$ significantly from the results when we use the exact value of K since only the nonincreasing part (also the matched part) of the ideal case in the P_d versus K curve is evaluated to decide $K_{\max}$, the maximum number of colluders a system can afford.

Simulations for the thresholding detector. As in Section 4.4, we need to first estimate K before setting a threshold h for the detection process. We introduce the following implementation.

(1) Estimate the number of colluders K via (4.68).
(2) Determine the threshold h correspondingly to yield a desired P_{fp}, according to (4.13). It is clear that the threshold h is only a function of $\widehat{K}$ when N, WNR, and n are given.
(3) Apply the thresholding test statistic described in (4.9) and return the set $\widehat{j}$.

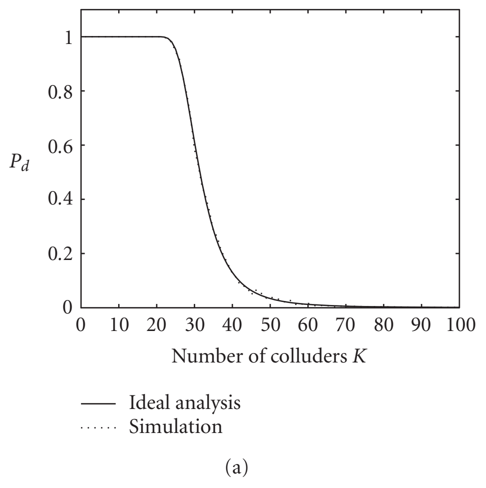
We compare the simulation results with the ideal performance analysis in Figures 4.9a and 4.9b. We can see that, with the estimated number of colluders, the observation when employing the thresholding detection is similar to that of the maximum detection.

4.5. Experiments with images

In order to demonstrate the performance of a Gaussian fingerprinting system using orthogonal modulation on real images for identifying colluders, we apply an additive spread-spectrum watermarking scheme similar to that in [24], where the original host image is divided into 8×8 blocks, and the watermark (fingerprint) is perceptually weighted and then embedded into the block DCT coefficients. The detection of the fingerprint is performed with the knowledge of the host image. To generally represent the performance, the 256×256 Lena and Baboon images with quite different natures are used as the host images for fingerprinting. The fingerprinted images have an average PSNR of 44.6 dB for Lena, and 41.9 dB for Baboon. We compare the performance of the thresholding detector under average, minimum, and minmax collusion attacks, respectively. We show in Figure 4.12 the original host images, the colluded images, and the difference images. With $K = 50$, an average PSNR of 37.3 dB for Lena and 34.6 dB for Baboon result after collusion attacks.

Denoting $\mathbf{s}_j$ as the Gaussian fingerprint, we note that the ith component of the jth fingerprint is actually embedded as

$$s_j(i)^t = \alpha(i)s_j(i), \tag{4.69}$$



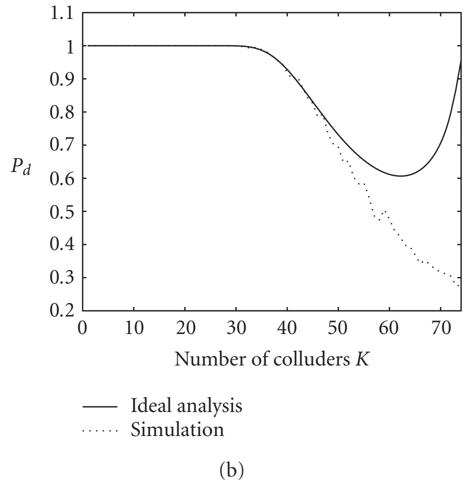


FIGURE 4.9. Probability of detection P_d as a function of the number of colluders K when applying the thresholding detector, with WNR = 0 dB, $N = 10^4$, and $P_{fp} \le 10^{-3}$. In (a) the number of users $n = 10^4$; in (b) $n = 75$.

where the superscript t means *actual*, with $\{\alpha(i)\}$'s being the just-noticeable-difference (JND) parameters from human visual model to achieve the imperceptibility of the embedded fingerprint. Therefore, the composite embedded fingerprint $\mathbf{y}^t$ after attack is represented as

$$y(i)^t = g\big(y_j(i)^t,\ j \in S_c\big) + d(i) = \alpha(i)g\big(s_j(i),\ j \in S_c\big) + x(i) + d(i), \tag{4.70}$$

where $g(\cdot)$ is the collusion function discussed in Section 4.3, and the noise d is independently distributed. Under nonblind detection, $\alpha(i)$'s are known in the detector side and thus the effects of real images can be partially compensated by

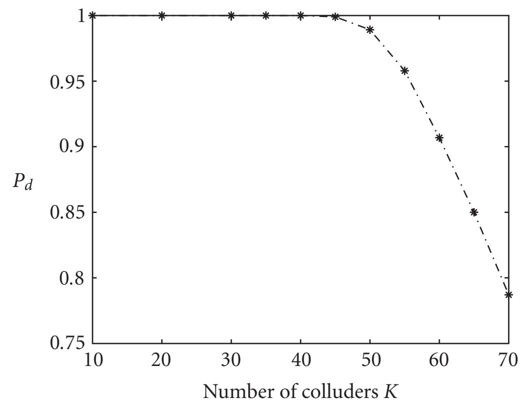


FIGURE 4.10. The detection performance of the thresholding detector on Lena images under the average attack, where equivalently $N = 13\,691$. Here $\sigma_d^2 = 1$, $n = 100$, and $P_{fp} = 10^{-3}$.

computing

$$w(i) = \frac{y(i)^t - x(i)}{\alpha(i)} = g\big(s_j(i),\ j \in S_c\big) + \frac{d(i)}{\alpha(i)} \tag{4.71}$$

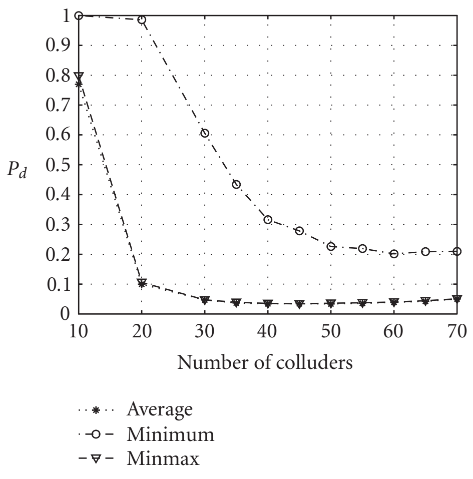
for $i = 1,\dots,N$. In practice, the variance of $d(i)$ is often proportional to the value of $\alpha(i)^2$, for example, in image compression attack. As such, $d(i)/\alpha(i)$ can be approximately modelled as *iid* $\mathcal{N}(0, \sigma_d^2)$ distributed. Therefore, the test statistic $T_N(j)$ used in the thresholding detector is now defined as

$$T_N(j) = \frac{\mathbf{w}^T \mathbf{s}_j}{\sqrt{||\mathbf{s}_j||^2}} \tag{4.72}$$

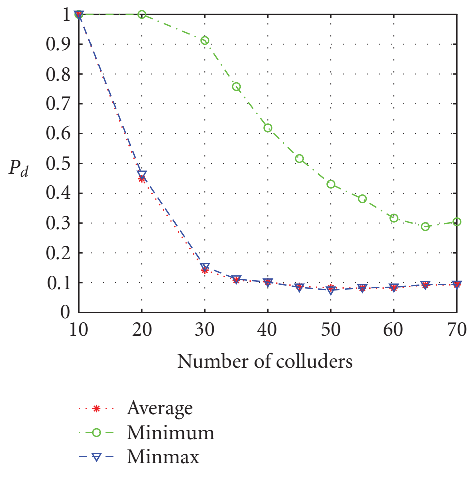
for $j = 1,\dots,n$.

We present the results in Figures 4.10 and 4.11 based on 10^5 simulations using real images. The number of total users n is set to 100. We ignored the round-off error introduced by DCT/IDCT transform in simulations. The fingerprint $\mathbf{s}_j$ is assumed to be $\mathcal{N}(0, \mathbf{I})$. To make a fair comparison between the experimental and analytical results, we first demonstrate the results for Lena image under the average attack in Figure 4.10, where the additive noise is with variance $\sigma_d^2 = 1$ and $\sigma_d^2 = 1$ and $P_{fp} = 10^{-3}$ is required. We note that the real image is comparable to that based on analysis in Section 4.1.2.

We further compare the performance of the thresholding detector under different types of attacks in Figure 4.11. The threshold for each K is chosen to satisfy $P_{fp} = 10^{-2}$ by simulation runs. σ_d^2 is set as 1 for the minimum attack case, and the corresponding σ_d^2 is properly adjusted for the cases of the average and minmax attacks to ensure the attacked images have the same MSE level (thus PSNR) with respect to the host image. The level of MSE is larger as K increases. It is noted that the detection performance is better under the minimum attack than under



(a)



(b)

FIGURE 4.11. The detection performance of the thresholding detector on real images under different kinds of attacks. Here $n = 100$ and $P_{fp} = 10^{-2}$. (a) The Lena image with equivalent $N = 13\,691$ and (b) the Baboon image with equivalent $N = 19\,497$.

the other two attacks. This suggests that the minimum attack is less efficient from the attacker point of view, an observation that matches with the analysis. It is also noticed that a better performance is observed in the Baboon example than in Lena. One possible explanation for this is that, in Lena, the efficient length of the fingerprint is $N = 13\,691$, while a longer $N = 19\,497$ is allowed in Baboon. Different characteristics such as the amount of edges and smooth regions of these two images also contribute to the difference in the performance. It is worth mentioning

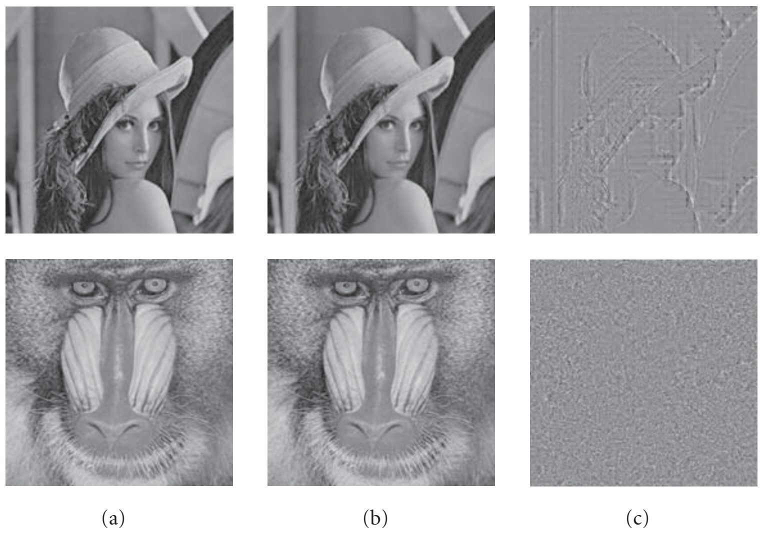


FIGURE 4.12. (a) The host images, (b) colluded images with $K = 50$, and (c) difference images for Lena and Baboon. The min attack is illustrated for Lena, and the average attack for Baboon.

that, for Gaussian watermarking, if K is large, the minimum attack is likely to produce noticeable distortion even with no additive noise added. For instance, under the minimum attack, the MSE is as large as 13.3 for Lena when $K = 70$. In order to have the same MSE under the average attack, we need to have a corresponding WNR as low as -7.5 dB. With such a low WNR, noticeable distortion is introduced to the host signal and the quality of the image may not be acceptable. Thus, the minimum attack is not favored in practice because it generates noticeable distortion.

4.6. Efficient fingerprint detection using tree structure

4.6.1. Tree-structured detection strategy

The classical method for estimating which signal was embedded in the host signal is done via v correlators, and determines the B-bit message that identifies which user's watermark was present. This has been considered a major drawback of the method of orthogonal modulation [23, 83]. In this section we present an algorithm that reduces the computation needed to detect which watermarks are present in a host signal.

Algorithm 4.3. Suppose that K colluders are involved in forming a colluded signal $\mathbf{y}_c$. We desire to identify the basis vectors of these K colluders. For a set $A = \{\mathbf{w}_j\}_{j\in J}$ where J is an indexing set, we define the sum of A by $\mathrm{SUM}(A) = \sum_{j\in J}\mathbf{w}_j$. We start by considering the case of detecting 1 watermark. Let us denote by $S = \{\mathbf{w},\dots,\mathbf{w}_v\}$ the set of orthogonal watermark signals, and suppose the test signal is $\mathbf{y}$. Suppose that we break S into two complementary subsets S_0 and S_1. If we correlate the test signal $\mathbf{y}$ with $\mathrm{SUM}(S_0)$, then the correlation will satisfy

$$\left\langle \mathbf{y}, \sum_{\mathbf{w}_j \in S_0} \mathbf{w}_j \right\rangle = \sum_{j \in J} \langle \mathbf{y}, \mathbf{w}_j \rangle, \tag{4.73}$$

where $\langle \mathbf{y}, \mathbf{w} \rangle$ denotes a correlation statistic. If the one watermark we desire to detect belongs to the set S_0, then $\langle \mathbf{y}, \mathrm{SUM}(S_0) \rangle$ will experience a large contribution from that one basis vector, and all the other terms will have small values. If this watermark is not present in S_0, then $\langle \mathbf{y}, \mathrm{SUM}(S_0) \rangle$ will consist only of small contributions due to noise. Therefore, if we test two sets S_0 and S_1 such that $S_1 = S \backslash S_0$, then we are likely to get a large value in at least one of the two correlations with the sum of the basis vectors. We can repeat this idea by further decomposing S_0 and/or S_1 if they pass a threshold test. This idea can be extended to detecting the presence of K orthogonal signals. At each stage we test two sets S_0 and S_1, and if a set passes a threshold test, then we further decompose it.

We use this idea to develop a recursive detection algorithm for detecting the presence of K orthogonal signals in a test signal $\mathbf{y}$. In Algorithm 4.3, we begin by initially splitting the set S into S_0 and S_1. There are many possible choices for dividing S into S_0 and S_1 in such an algorithm. In Algorithm 4.3 we have chosen S_0 such that $|S_0| = 2^{\lceil \log_2 |S| \rceil - 1}$, which is the largest power of 2 less than $|S|$. Another possible choice would be to take S_0 such that $|S_0| = \lceil |S|/2 \rceil$. The algorithm proceeds in a recursive manner, subdividing either S_0 or S_1 if a threshold test is passed. As we will shortly discuss, the choice of the thresholds τ_0 and τ_1 is dependent on the signal-to-noise ratio, the cardinality of either S_0 or S_1, and the desired probability of detection for that level.

We now make some observations about the performance of this algorithm. First, the algorithm can be described via a binary tree, where each internal node corresponds to two correlations. Let us assume that each correlation truthfully reveals whether there is a colluder present or not. We denote by $C(n, K)$ the number of correlations needed in Algorithm 4.3 to identify K signals from a set S of n orthogonal signals. We can derive a bound for $C(n, K)$ in the ideal case where each correlation is truthful, namely

$$C(n, K) \leq 2\left(-1 + K\left(\log_2 \frac{2^{\lceil \log_2 n \rceil}}{K} + 1 \right)\right). \tag{4.74}$$

This bound can be shown using standard techniques for tree-based algorithms [84, 85, 86, 87]. In particular, the bound on the amount of truthful correlations needed to identify K colluders is $\mathcal{O}(K \log(n/K))$. Further, we observe that if we were trying to detect a single signal, then we need to perform at most $2(\lceil \log_2 |S| \rceil - 1)$ correlations as opposed to $|S|$ in a traditional implementation. Also, as K becomes larger, the improvement in the amount of correlations performed decreases since it becomes necessary to perform correlations for multiple branches of the tree.

Realistically, however, the correlations performed at each node of the algorithm are not guaranteed to be truthful. In fact, although we have achieved an improvement in computational efficiency, this comes at a tradeoff in detector variance. When we calculate the correlation with the sums of basis vectors, we get many small, noisy contributions from correlating the test signal with signals not present in the test signal, as in (4.73).

We now provide analysis for this phenomenon when there is only one colluder, that is, $y(k) = s_1(k) + d(k)$. For simplicity, let $\mathbf{d} = N(\mathbf{0}, \sigma_d^2\mathbf{I})$. The $\mathbf{s}_j$ are known and have power $\|s_j\|^2 = \mathcal{E}$. The two possible hypotheses are

$$\begin{aligned} H_0 &: \mathbf{y} = \mathbf{d}, \\ H_1 &: \mathbf{y} = \mathbf{d} + \mathbf{s}_1. \end{aligned} \tag{4.75}$$

We break S into $S_0 = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{n/2}\}$ and $S_1 = \{\mathbf{s}_{n/2+1}, \mathbf{s}_{n/2+2}, \ldots, \mathbf{s}_n\}$. For simplicity of derivation, we use an unnormalized correlator for the detection statistics ρ_0 and ρ_1. That is,

$$\langle \mathbf{y}, \mathbf{s} \rangle = \sum_{k=1}^{N} y(k)s(k). \tag{4.76}$$

Under hypothesis H_1, the calculation for ρ_0 is

$$\rho_0 = \langle \mathbf{s}_1 + \mathbf{d}, \mathbf{s}_1 + \mathbf{s}_2 + \cdots + \mathbf{s}_{n/2} \rangle = \|\mathbf{s}_1\|^2 + \sum_{j=1}^{n/2} \langle \mathbf{d}, \mathbf{s}_j \rangle. \tag{4.77}$$

Under hypothesis H_0, the calculation for ρ_0 is

$$\rho_0 = \langle \mathbf{d}, \mathbf{s}_1 + \mathbf{s}_2 + \cdots + \mathbf{s}_{n/2} \rangle = \sum_{j=1}^{n/2} \langle \mathbf{d}, \mathbf{s}_j \rangle. \tag{4.78}$$

Then $E(\rho_0; H_0) = 0$, $E(\rho_0; H_1) = \mathcal{E}$, and $\text{Var}(\rho_0; H_0) = \text{Var}(\rho_0; H_1) = (n/2)\sigma_d^2\mathcal{E}$. Thus $\rho_0 \sim N(0, n\sigma_d^2\mathcal{E}/2)$ under H_0, and $\rho_0 \sim N(\mathcal{E}, n\sigma_d^2\mathcal{E}/2)$ under H_1. Similar results can be derived for ρ_1. The probability of detection is

$$P_D = P_r(\rho_0 > \tau; H_1) = Q\left(\frac{\tau - \mathcal{E}}{\sqrt{\sigma_d^2 n\mathcal{E}/2}}\right). \tag{4.79}$$

The probability of false alarm is

$$P_{\text{FA}} = P_r(\rho_0 > \tau; H_0) = Q\left(\frac{\tau}{\sqrt{\sigma_d^2 n\mathcal{E}/2}}\right). \tag{4.80}$$

As we iterate down the tree, the SNR will become better. For example, at the second level of the algorithm's tree, the set S_0 has $n/4$ elements, and $\rho_0 \sim N(0, n\sigma_d^2\mathcal{E}/4)$ under H_0, and $\rho_0 \sim N(\mathcal{E}, n\sigma_d^2\mathcal{E}/4)$ under H_1. At each level of the

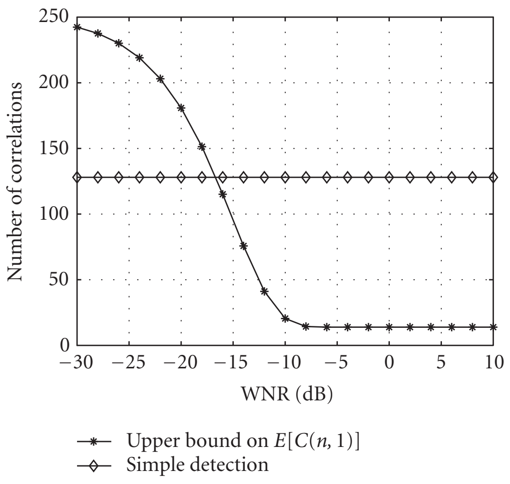


FIGURE 4.13. The bound for the expected amount of correlations needed when there is one colluder, $n = 128$ users, and $P_D = 0.99$ for each level. As a baseline, we plot the bound for $E[C(128, 1)]$ against the amount n, which is the amount of computations needed in performing simple detection.

algorithm, the decision threshold τ may be determined using either a chosen value for the probability of detection or probability of false alarm for the one colluder case, that is, from (4.79) or (4.80). If we choose τ at each level of the tree to keep P_D fixed at a sufficiently high value, then the probability of a false alarm will change at each level of the tree. This means that initially we will let through some false alarms until we proceed further down the tree, where there are higher effective SNRs.

It can be shown that a bound for the expected amount of correlations, $E[C(n, 1)]$, needed to identify a single colluder using Algorithm 4.3 when $n = 2^r$ is

$$E[C(n, 1)] \leq 2 + 2(\ln n - 1)P_D + 2\sum_{k=1}^{r-1} P_{\mathrm{FA}}^{b_k}\left(2^{r-k} - 1\right), \tag{4.81}$$

where b_k is the binary string consisting of $k-1$ zeros followed by a single 1. Here we have chosen to label the one colluder as user 1, and have denoted the probability of false alarm for a node b by P_{FA}^{b}.

The bound depends on the choice of P_D and the $P_{\mathrm{FA}}^{b_k}$ values. In Figure 4.13, we present the bound for the expected amount of correlations needed when there is one colluder, $n = 128$ users, and $P_D = 0.99$ for each level. As a baseline, we have plotted the bound for $E[C(128, 1)]$ against $n = 128$, which is the amount of computations needed in performing simple detection. Examining this figure, one observes that at low WNR, which could correspond to a blind detection scenario, the bound on the amount of correlations needed in Algorithm 4.3 is above the baseline amount of correlations needed for simply correlating with each of the fingerprint waveforms. This poor performance of the bound is due to the tradeoff between P_D and P_{FA}. Specifically, given $P_D = 0.99$, it is not possible to make the $P_{\mathrm{FA}}^{b_k}$

small at low WNR. Thus, at low WNR the tree-structured detection scheme may not be advantageous over a simple detection scheme. However, at higher WNR, which corresponds to nonblind detection scenarios, the separation between the detection hypotheses increases, and it does become possible to make $P_{\mathrm{FA}}^{b_k}$ small. In these cases, the bound guarantees that we will need less correlations than simply correlating with each waveform to identify a single colluder.

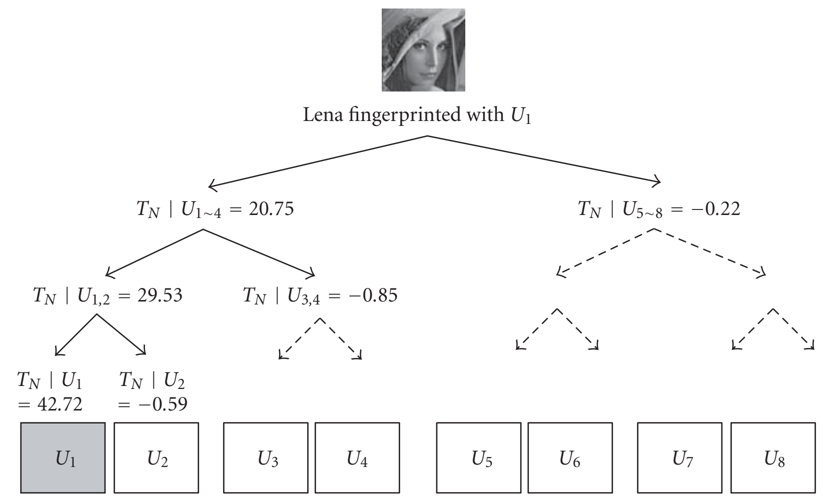
4.6.2. Experiments on tree-based detector

We desired to study the performance of the tree-structured detection algorithm, and the effect that collusion had on the detection statistics. We now explore real image experiments on tree-based detection of orthogonal fingerprints. In our experiments, we used an additive spread-spectrum watermarking scheme similar to that in [24], where a perceptually weighted watermark was added to DCT coefficients with a block size of 8×8. The detection of the watermark is performed without the knowledge of the host image via the detection statistics as shown in (2.6). The 512×512 Lena was used as the host image for fingerprinting, and the fingerprinted images had no visible artifacts with an average PSNR of 41.2 dB. Figure 4.14 illustrates the process of identifying colluders out of 8 users using the tree-structured detection algorithm (Algorithm 1). The detection statistics are averaged over 10 different sets of watermarks, and each set has 8 mutually uncorrelated spread-spectrum watermarks for 8 users. These watermarks are generated via a pseudorandom number generator and used as an approximate orthogonal basis in orthogonal modulation.

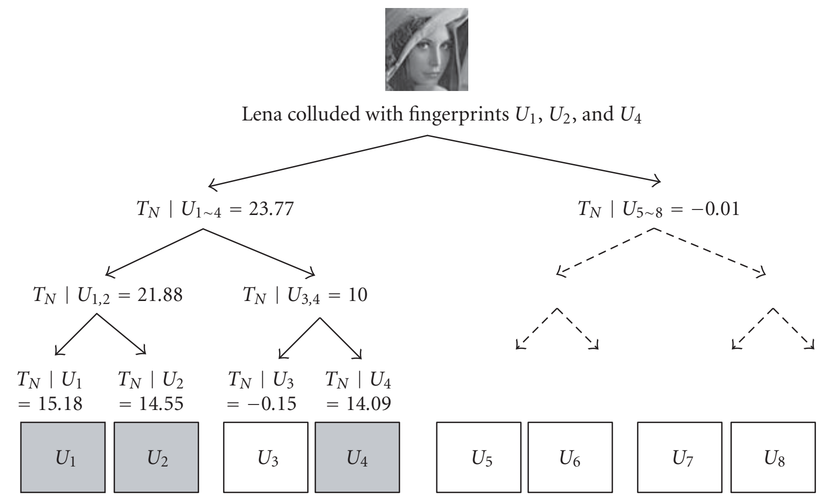
Figure 4.14a shows the process of detecting colluders from an image with user 1's fingerprint embedded. The notation "$T_N \mid U_?$" denotes the detection statistics when correlating the test image with the sum of the fingerprints $U_?$. Detection statistics close to zero indicate the unlikely contributions from the corresponding fingerprints, and the branches of the detection tree below them, indicated by dotted lines, are not explored further. The number of correlations performed is 6. Figure 4.14b shows the process of detecting colluders from an image colluded from user 1, user 2, and user 4's fingerprinted images. The number of correlations performed is 8.

We see from Figure 4.14a that the detection statistics when correlating with a sum of a larger number of basis vectors are smaller than that with a smaller amount of basis vectors. This reflects the noisy contributions from the basis vectors that are present in the sum of basis vectors but are not present in the test image. We discussed this phenomena earlier in Section 4.6. Since the detection statistics we use have their variance normalized to 1, the noisy contributions lower the detection statistics values. We also observe in Figure 4.14b a decrease in the detection statistics in images colluded by more users.

In addition, we conducted a nonblind detection test with one colluder amongst $n = 128$ users on the Lena image. Our test confirmed the findings of Figure 4.13. Only 14 correlations were needed, which is a significant reduction over the 128 correlations needed in a simple detection approach.


Lena fingerprinted with U_1
$T_N \mid U_{1\sim4} = 20.75$
$T_N \mid U_{5\sim8} = -0.22$
$T_N \mid U_{1,2} = 29.53$
$T_N \mid U_{3,4} = -0.85$
$T_N \mid U_1 = 42.72$
$T_N \mid U_2 = -0.59$
U_1 U_2 U_3 U_4 U_5 U_6 U_7 U_8

(a)


Lena colluded with fingerprints U_1, U_2, and U_4
$T_N \mid U_{1\sim4} = 23.77$
$T_N \mid U_{5\sim8} = -0.01$
$T_N \mid U_{1,2} = 21.88$
$T_N \mid U_{3,4} = 10$
$T_N \mid U_1 = 15.18$
$T_N \mid U_2 = 14.55$
$T_N \mid U_3 = -0.15$
$T_N \mid U_4 = 14.09$
U_1 U_2 U_3 U_4 U_5 U_6 U_7 U_8

(b)

FIGURE 4.14. Detection trees for identifying colluders using Algorithm 1. The images for different users are fingerprinted via orthogonal modulation. The fingerprints of colluders are indicated by shadowed boxes U_j. The notation "$T_N \mid U_?$" denotes the detection statistics from correlating the test image with the sum of the fingerprints $U_?$. (a) One colluder. (b) Three colluders.

4.7. Chapter summary

In this chapter, we have investigated the collusion resistance of a Gaussian fingerprinting system based upon orthogonal modulation. Specifically, assuming the host content is available on the detector side (nonblind scenario), we study the problem of determining how many independently marked copies of the same multimedia content are required for attackers to cause a fingerprinting system to fail. We introduced the collusion problem for additive embedding and started with the average collusion attack where an average operation is performed by weighing marked copies equally. Since knowledge of the number of colluders (different marked copies) is normally not available in practice, a likelihood-based classifier approach was proposed to estimate the number of colluders K. Simulation results show that the collusion resistance based on the estimated K matches the analysis of the ideal case.

We introduced two detection approaches, and studied the collusion resistance of a fingerprinting system to the average attack when considering the performance criteria represented by P_{fp} and P_{np}. We derived lower and upper bounds of the maximum number of colluders $K_{\max}$. It is noted that $\sqrt{\eta N}$ is an important factor, where η is the watermark-to-noise ratio. Using the upper bound, an attacker can determine how many independent copies are required to guarantee the success of a collusion attack; on the other hand, an owner (detector) will benefit from these bounds in designing a fingerprinting system. For instance, in order to achieve a collusion-free fingerprinting system, a desirable security requirement is to have it very unlikely for a potential attacker to collect more copies than the lower bound, and further to have the distribution size limited by the maximum value of $K_{\max}$.

Our work was further extended to different attacks and different sets of performance criteria. From the detector point of view, the thresholding detector is robust to different attacks, since different attacks yield very close performance as long as the levels of MSE distortion introduced by different attacks are the same. Therefore, the average attack is most efficient from the attacker side. We also evaluated the performance on real images, and noted that the average attack is the most efficient from the attacker point of view under the same MSE (thus PSNR) assumption. Different sets of performance criteria were explored to satisfy different concerns. And it seems that attacks based on a few dozen independent copies will confound a fingerprinting system accommodating as many as ten thousand users. This observation suggests that the number of independently marked copies of the same content that can be distributed should be determined by many concerns, such as the system requirements, and the cost of obtaining multiple independent copies. Furthermore, it suggests that tracing colluders via fingerprints should work in concert with other operations, for example, suspecting a user may lead the owner to more closely monitor that user and further gather additional evidence. As fingerprinting is one of the many components in decision-making, it is the confidence in the fidelity of all technical and nontechnical evidences as a whole that allows a system to identify a colluder.

In addition, the traditional detection schemes for orthogonal modulation in embedding applications require an amount of correlations that is linear in the amount of orthogonal basis signals. To address this deficiency, we presented a tree-based detection algorithm that reduces the amount of correlations from linear to logarithmic complexity, and is able to identify K colluders in a computationally efficient manner.

5 Group-oriented fingerprinting

In the previous chapter we have examined fingerprinting systems using orthogonal modulation. Despite the superior collusion resistance of orthogonal Gaussian fingerprints over other fingerprinting schemes, previous analysis revealed that attacks based on averaging a few dozen independent copies can confound a fingerprinting system using orthogonal modulation [58, 59, 69, 70]. Averaging collusion attack is proved effective on orthogonal fingerprinting system due to its effect on the energy reduction of the original fingerprints and the effect it has upon the detection performance. Therefore, by gathering a few dozen colluders, it is possible to sufficiently attenuate each colluder's identifying fingerprint and produce a new version of the content with no detectable fingerprints. Ultimately, for mass market consumption of multimedia, content will be distributed to thousands of users. In these scenarios, it is possible for a coalition of adversaries to acquire a few dozen copies of marked content, employ a simple average collusion attack, and thereby thwart the protection provided by the fingerprints. Thus, an alternative fingerprinting scheme is needed that will exploit a different aspect of the collusion problem in order to achieve improved collusion resistance.

We note that one major drawback of fingerprinting using orthogonal modulation is its severe energy reduction. For example, under the average attack, the resulting energy of the colluded copy is reduced to $1/K$ of the original fingerprint energy, with K being the number of colluders. This energy reduction significantly degrades the detection performance of each original fingerprint. As we mentioned earlier, there are two main approaches using spread spectrum for fingerprint embedding: orthogonal modulation and code modulation. The second option allows for constructing the fingerprint for each user as a linear combination of orthogonal noise-like basis signals. Along the code-modulation line, a key is to strategically introduce correlations into different fingerprints to allow accurate identification of the contributing fingerprints involved in collusion. The correlation concern also helps to decrease the energy reduction ratio observed in the case of orthogonal modulation. The resulting fingerprints can be based upon binary or real-valued code modulation. The group-oriented fingerprinting scheme studied in this chapter can be regarded as exploring real-valued code constructions. In the next chapter, we will focus on a class of collusion-resistant fingerprints exploiting a class of binary codes, which have the property that the composition of any subset of K or fewer codevectors is unique.

In this chapter, we introduce a new direction for improving collusion resistance, the group-oriented fingerprinting, by taking advantage of the prior knowledge about potential collusions. We observe that some users are more likely to collude with each other than with other users, perhaps due to underlying social or cultural factors. We propose to exploit this a priori knowledge to improve the fingerprint design. We introduce a fingerprint construction that is an alternative to the traditional independent Gaussian fingerprints. Like the traditional spread-spectrum watermarking scheme, our fingerprints are Gaussian distributed. However, we assign statistically independent fingerprints to members of different groups that are unlikely to collude with each other, while the fingerprints we assign to members within a group of potential colluders are correlated. For instance, we present a two-tier fingerprinting scheme in which the groups of potential colluders are organized into sets of users that are equally likely to collude with each other. We assume in the two-tier model that intergroup collusion is less likely than intragroup collusion. We extend our construction to more general group collusion scenarios by presenting a tree-based construction of fingerprints. The design of the fingerprint is complemented by the development and analysis of a detection scheme capable of providing the forensic ability to identify groups involved in collusion and to trace colluders within each group.

5.1. Motivation for group-based fingerprinting

In this section, we will introduce fingerprinting and collusion. Collusion-resistant fingerprinting requires the design of fingerprints that can survive collusion and identify colluders, as well as the robust embedding of the fingerprints in the multimedia host signal. We will employ spread-spectrum additive embedding of fingerprints in this chapter since this technique has proven to be robust against a number of attacks [23]. Additionally, information theory has shown that spread-spectrum additive embedding is near optimal when the original host signal is available at the detector side [38, 39], which is a reasonable assumption for collusion applications.

We begin by reviewing spread-spectrum additive embedding. Suppose that the host signal is represented by a vector $\mathbf{x}$, which might, for example, consist of the most significant DCT components of an image. The owner generates the watermark $\mathbf{s}$ and embeds each component of the watermark into the host signal by $y(l) = x(l) + s(l)$, with $y(l)$, $x(l)$, and $s(l)$ being the lth component of the watermarked copy, the host signal, and the watermark, respectively. It is worth mentioning that, in practical watermarking, before the watermark is added to the host signal, each component of the watermark $\mathbf{s}$ is scaled by an appropriate factor to achieve the imperceptibility of the embedded watermark as well as control the energy of the embedded watermark. One possibility for this factor is to use the *just-noticeable difference* (JND) from a human visual model [24].

In digital fingerprinting, the content owner has a family of watermarks, denoted by $\{\mathbf{s}_j\}$, which are fingerprints associated with different users who purchase the rights to access the host signal $\mathbf{x}$. These fingerprints are used to make copies of content that may be distributed to different users, and allow for the tracing of pirated copies to the original users. For the jth user, the owner computes the marked version of the content $\mathbf{y}_j$ by adding the watermark $\mathbf{s}_j$ to the host signal, meaning $\mathbf{y}_j = \mathbf{x}+\mathbf{s}_j$. Then this fingerprinted copy $\mathbf{y}_j$ is distributed to user j and may experience additional distortion before it is tested for the existence of the fingerprint $\mathbf{s}_j$. The fingerprints $\{\mathbf{s}_j\}$ are often chosen to be orthogonal noise-like signals [23], or are built by using a modulation scheme employing a basis of orthogonal noise-like signals [80, 88]. For this chapter, we restrict our attention to linear modulation schemes, where the fingerprint signals $\mathbf{s}_j$ are constructed using a linear combination of a total of v orthogonal basis signals $\{\mathbf{u}_i\}$ such that

$$\mathbf{s}_j = \sum_{i=1}^{v} b_{ij}\mathbf{u}_i, \tag{5.1}$$

and a sequence $\{b_{1j}, b_{2j}, \ldots, b_{vj}\}$ is assigned for each user j. It is convenient to represent $\{b_{ij}\}$ as a matrix $\mathcal{B}$, and different matrix structures correspond to different fingerprinting strategies. An identity matrix for $\mathcal{B}$ corresponds to orthogonal modulation [23, 79, 83], where $\mathbf{s}_j = \mathbf{u}_j$. Thus, each user is identified by means of an orthogonal basis signal. In practice, it is often sufficient to use independently generated random vectors $\{\mathbf{u}_j\}$ for spread-spectrum watermarking [23]. The orthogonality or independence allows for distinguishing different users' fingerprints to the maximum extent. The simple structure of orthogonal modulation for encoding and embedding makes it attractive in identification applications that involve a small group of users. Fingerprints may also be designed using code modulation [45]. In this case, the matrix $\mathcal{B}$ takes a more general form. One advantage of using code modulation is that we are able to represent more than v users when using v orthogonal basis signals. One method for constructing the matrix $\mathcal{B}$ is to use appropriately designed binary codes. As an example, we recently proposed a class of binary-valued anticollusion codes (ACC), where the shared bits within codevectors allow for the identification of up to K colluders (see Chapter 6). In more general constructions, the entries of $\mathcal{B}$ can be real numbers. The key issue of fingerprint design using code modulation is to strategically introduce correlation among different fingerprints to allow for accurate identification of the contributing fingerprints involved in collusion.

In a collusion attack on a fingerprinting system, one or more users with different marked copies of the same host signal come together and combine several copies to generate a new composite copy $\mathbf{y}$ such that the traces of each of the "original" fingerprints are removed or attenuated. In earlier chapters, we showed that different nonlinear collusion attacks had almost identical performance to linear collusion attacks based on averaging marked content signals, when the levels of MSE distortion introduced by the attacks were kept fixed. In a K-colluder averaging-collusion attack the watermarked content signals $\mathbf{y}_j$ are combined according to $\sum_{j=1}^{K} \lambda_j \mathbf{y}_j + \mathbf{d}$, where $\mathbf{d}$ is an added distortion. For the simplicity of analysis, we will focus on the averaging-type collusion for the remainder of this chapter.

One principle for enhancing the forensic capability of a multimedia fingerprinting system is to take advantage of any prior knowledge about potential collusion attacks during the design of the fingerprints. In this chapter, we investigate mechanisms that improve the ability to identify colluders by exploiting fundamental properties of the collusion scenario. In particular, we observe that fingerprinting systems using orthogonal modulation do not consider the following issues.

(1) Orthogonal fingerprinting schemes are designed for the case where all users are equally likely to collude with each other. This assumption that users collude together in a uniformly random fashion is unreasonable. It is more reasonable that users from the same social or cultural background will collude together with a higher probability than with users from a different background. For example, a teenage user from Japan is more likely to collude with another teenager from Japan than with a middle-aged user from France. In general, the factors that lead to dividing the users into groups are up to the system designer to determine. Once the users have been grouped, we may take advantage of this grouping in a natural way: divide fingerprints into different subsets and assign each subset to a specific group whose members are more likely to collude with each other than with members from other groups.

(2) Orthogonality of fingerprints helps to distinguish individual users. However, this orthogonality also puts innocent users into suspicion with equal probability. It was shown in Chapter 4 that when the number of colluders is beyond a certain value, catching one colluder successfully is very likely to require the detection system to suspect all users as guilty. This observation is obviously undesirable for any forensic system, and suggests that we introduce correlation between the fingerprints of certain users. In particular, we may introduce correlation between members of the same group, who are more likely to collude with each other. Therefore, when a specific user is involved in a collusion, users from the same group will be more likely accused than users from groups not containing colluders.

(3) The performance can be improved by applying appropriate detection strategies. The challenge is to take advantages of the previous points when designing the detection process.

By considering these issues, we can improve on the orthogonal fingerprinting system, and provide a means to enhance collusion resistance. The underlying philosophy is to introduce a well-controlled amount of correlation into user fingerprints. Our fingerprinting systems involve two main directions of development: the development of classes of fingerprints capable of withstanding collusion and the development of forensic algorithms that can accurately and efficiently identify members of a colluding coalition. Therefore, for each of our proposed systems, we will address the issues of designing collusion-resistant fingerprints and developing efficient colluder detection schemes. To validate the improvement of such group-oriented fingerprinting system, we will evaluate the performance of our proposed systems under the average attack and compare the resulting collusion resistance to that of an orthogonal fingerprinting system.

5.2. Two-tier group-oriented fingerprinting system

As an initial step for developing a group-oriented fingerprinting system, we present a two-tier scheme that consists of several groups, and within each group are users who are equally likely to collude with each other but less likely to collude with members from other groups. The two-tier group-oriented fingerprints described above can be viewed using a two-level tree. The first level consists of L nodes, with each node supporting M leaves that correspond to the fingerprints of individual users within one group. In Section 5.3, if we allow for more general tree structures, we can achieve more flexibility in capturing the collusion dynamics between different groups. As mentioned earlier, design of fingerprinting systems consist of two design components in order to fight against collusion attacks: one is the design of collusion-resistant fingerprints and the other is the detection schemes needed to identify and trace members of the adversarial coalition. We will address these two issues respectively.

5.2.1. Fingerprint design scheme

The design of our fingerprints are based upon (1) grouping and (2) code modulation.

Grouping. The overall fingerprinting system is implemented by designing L groups. For simplicity, we assume that each group can accommodate up to M users. Therefore, the total number of users is $n = M \times L$. The choice of M is affected by many factors, such as the number of potential purchasers in a region and the collusion pattern of users. We also assume that fingerprints assigned to different groups are statistically independent of each other. There are two main advantages provided by independence between groups. First, the detection process is simple to carry out, and secondly, when collusion occurs, the independence between groups limits the amount of innocent users falsely placed under suspicion within a group, since the possibility of wrongly accusing another group is negligible.

Code modulation within each group. We will apply the same code matrix to each group. For each group i, there are v orthogonal basis signals $\mathbf{U}_i = [\mathbf{u}_{i1}, \mathbf{u}_{i2}, \dots, \mathbf{u}_{iv}]$, each having Euclidean norm $\|\mathbf{u}\|$. We choose the sets of orthogonal bases for different groups to be independent. In code modulation, information is encoded into $\mathbf{s}_{ij}$, the jth fingerprint in group i, via

$$\mathbf{s}_{ij} = \sum_{l=1}^{v} c_{lj}\mathbf{u}_{il}, \tag{5.2}$$

where the symbol c_{lj} is a real value and all $\mathbf{s}$ and $\mathbf{u}$ terms are column vectors with length N and equal energy. We define the *code matrix* $\mathbf{C} = (c_{lj}) = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_M]$

as the $v \times M$ matrix whose columns are the codevectors of different users. We have $\mathbf{S}_i = [\mathbf{s}_{i1}, \mathbf{s}_{i2}, \dots, \mathbf{s}_{iM}] = \mathbf{UC}$, with the correlation matrix of $\{\mathbf{s}_{ij}\}$ as

$$\mathbf{R}_s = \|\mathbf{u}\|^2\mathbf{R}, \quad \text{with } \mathbf{R} = \mathbf{C}^T\mathbf{C}. \tag{5.3}$$

The essential task in designing the set of fingerprints for each subsystem is to design the underlying correlation matrix $\mathbf{R}_s$. With the assumption in mind that the users in the same group are equally likely to collude with each other, we create the fingerprints in one group to have equal correlation. Thus, we choose a matrix $\mathbf{R}$ such that all its diagonal elements are 1 and all the off-diagonal elements are ρ. We will refer to ρ as the *intragroup correlation*.

For the proposed fingerprint design, we need to address such issues as the size of groups and the coefficient ρ. The parameters M and ρ will be chosen to yield good system performance. In our implementation, M is chosen to be the best supportable user size for the orthogonal modulation scheme [58, 59]. In particular, when the total number of users is small, for instance $n \le 100$, there is no advantage in having many groups, and it is sufficient to use one or two groups. As we will see later in (5.15), the detection performance for the single group case is characterized by the mean difference $(1-\rho)\|\mathbf{s}\|/K$ for K colluders. A larger value of the mean difference is preferred, implying a negative ρ is favorable. On the other hand, when the fingerprinting system must accommodate a large number of users, there will be more groups and hence the primary task is to identify the groups containing colluders. In this case, a positive coefficient ρ should be employed to yield high accuracy in group detection. For the latter case, to simplify the detection process, we propose a structured design of fingerprints $\{\mathbf{s}_{ij}\}$'s, consisting of two components:

$$\mathbf{s}_{ij} = \sqrt{1-\rho}\,\mathbf{e}_{ij} + \sqrt{\rho}\,\mathbf{a}_i, \tag{5.4}$$

where $\{\mathbf{e}_{i1}, \dots, \mathbf{e}_{iM}, \mathbf{a}_i\}$ are the orthogonal basis vectors of group i with equal energy. The bases of different groups are independent. It is easy to check the fact that $\mathbf{R}_s = N\sigma_u^2\mathbf{R}$ under this design scheme.

5.2.2. Detection scheme

The design of appropriate fingerprints must be complemented by the development of mechanisms that can capture those involved in the fraudulent use of content. When collusion occurs, the content owner's goal is to identify the fingerprints associated with users who participated in generating the colluded content. In this section we discuss the problem of detecting the colluders when the above scheme is considered. In Figure 5.1 we depict a system accommodating n users, consisting of L groups with M users within each group. Suppose, when a collusion occurs involving K colluders who form a colluded content copy $\mathbf{y}$, that the number of colluders within group i is k_i and that k_i's satisfy $\sum_{i=1}^{L} k_i = K$. The observed content $\mathbf{y}$

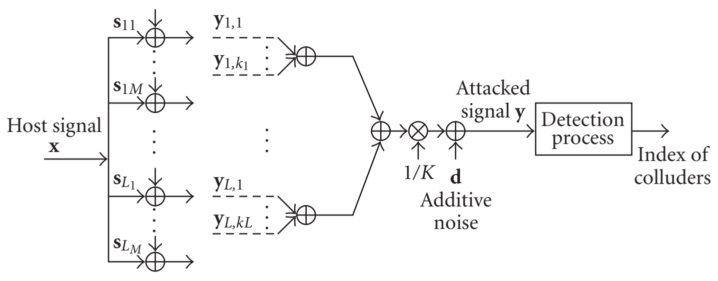


FIGURE 5.1. Model for collusion by averaging.

after the average collusion is

$$\mathbf{y} = \frac{1}{K}\sum_{i=1}^{L}\sum_{j\in S_{ci}} \mathbf{y}_{ij} + \mathbf{d} = \frac{1}{K}\sum_{i=1}^{L}\sum_{j\in S_{ci}} \mathbf{s}_{ij} + \mathbf{x} + \mathbf{d}, \tag{5.5}$$

where $S_{ci} \subseteq [1,\dots,M]$ indicates a subset of size $|S_{ci}| = k_i$ describing the members of group i that are involved in the collusion, and the $\mathbf{s}_{ij}$'s are Gaussian distributed. We also assume that the additive distortion $\mathbf{d}$ is an N-dimensional vector following an i.i.d. Gaussian distribution with zero-mean and variance σ_d^2. In this model, the number of colluders K and the subsets S_{ci}'s are unknown parameters. The nonblind scenario is assumed in our consideration, meaning that the host signal $\mathbf{x}$ is available at the detector and thus always subtracted from $\mathbf{y}$ for analysis.

The detection scheme consists of two stages. The first stage focuses on identifying groups containing colluders, and the second one involves identifying colluders within each "guilty" group.

Stage 1: group detection. Because of the independency of different groups and the assumption of i.i.d. Gaussian distortion, it suffices to consider the (normalized) correlator vector $\mathbf{T}_G$ for identifying groups possessing colluders. The ith component of $\mathbf{T}_G$ is expressed by

$$T_G(i) = \frac{(\mathbf{y}-\mathbf{x})^T(\mathbf{s}_{i1}+\mathbf{s}_{i2}+\cdots+\mathbf{s}_{iM})}{\sqrt{\|\mathbf{s}\|^2[M+(M^2-M)\rho]}} \tag{5.6}$$

for $i = 1,2,\dots,L$. Utilizing the special structure of the correlation matrix $\mathbf{R}_s$, we can show that the distribution follows

$$p\big(T_G(i) \mid K, k_i, \sigma_d^2\big) = \begin{cases} N\big(0,\sigma_d^2\big), & \text{if } k_i = 0, \\[2ex] N\left(\dfrac{k_i\|\mathbf{s}\|\sqrt{1+(M-1)\rho}}{K\sqrt{M}}, \sigma_d^2\right), & \text{otherwise,} \end{cases} \tag{5.7}$$

where $k_i = 0$ indicates that no user within group i is involved in the collusion attack. We note that based on the independence of fingerprints from different groups, the $T_G(i)$'s are independent of each other. Further, based on the distribution of $T_G(i)$, we see that if no colluder is present in group i, $T_G(i)$ will only consist of small contributions. However, as the amount of colluders belonging to group i increases, we are more likely to get a larger value of $T_G(i)$.

We employ the correlators $T_G(i)$'s for detecting the presence of colluders within each group. For each i, we compare $T_G(i)$ to a threshold h_G, and report that the ith group is *colluder-present* if $T_G(i)$ exceeds h_G. That is,

$$\widehat{\mathbf{i}} = \arg_{i=1}^{L} \{T_G(i) \geq h_G\}, \tag{5.8}$$

where the set $\widehat{\mathbf{i}}$ indicates the indices of groups including colluders. As indicated in the distribution (5.7), the threshold h_G here is determined by the pdf. Since normally the number of groups involved in the collusion is small, we can correctly classify groups with high probability under the nonblind scenario.

Stage 2: colluder detection within each group. After classifying groups into the colluder-absent class or the colluder-present class, we need to further identify colluders within each group. For each group $i \in \widehat{\mathbf{i}}$, because of the orthogonality of basis $[\mathbf{u}_{i1}, \mathbf{u}_{i2}, \ldots, \mathbf{u}_{iM}]$, it is sufficient to consider the correlators $\mathbf{T}_i$, with the jth component $T_i(j) = (\mathbf{y} - \mathbf{x})^T \mathbf{u}_{ij} / \sqrt{\|\mathbf{u}\|^2}$ for $j = 1, \ldots, M$. We can show that

$$\mathbf{T}_i = \frac{\|\mathbf{u}\|}{K} \mathbf{C}\mathbf{\Phi}_i + \mathbf{n}_i, \tag{5.9}$$

where $\mathbf{\Phi} \in \{0,1\}^M$ with $\Phi_i(j) = 1$ for $j \in S_{ci}$, indicates colluders within group i via the location of components whose values are 1; and $\mathbf{n}_i = \mathbf{U}_i \mathbf{d}^T / \sqrt{\|\mathbf{u}\|^2}$ follows an $N(0, \sigma_d^2 \mathbf{I}_M)$ distribution. Thus, we have the distribution

$$p(\mathbf{T}_i \mid K, S_{ci}, \sigma_d^2) = N\left(\frac{\|\mathbf{u}\|}{K} \mathbf{C}\mathbf{\Phi}_i, \sigma_d^2 \mathbf{I}_M\right). \tag{5.10}$$

Recall the correlation coefficients

$$\mathbf{c}_j^T \mathbf{c}_l = \begin{cases} 1, & \text{if } j = l, \\ \rho, & \text{if } j \neq l. \end{cases} \tag{5.11}$$

Now assume the parameters K and k_i are known, we can estimate the subset S_{ci} via

$$\begin{aligned}
\widehat{S}_{ci} &= \arg\max_{|S_{ci}|=k_i} \{p(\mathbf{T}_i \mid K, S_{ci}, \sigma_d^2)\} = \arg\min_{|S_{ci}|=k_i} \left\| \mathbf{T}_i - \frac{\|\mathbf{u}\|}{K}\mathbf{C}\mathbf{\Phi}_i \right\|^2 \\
&= \arg\min_{|S_{ci}|=k_i} \left\{ \mathbf{T}_i^T\mathbf{T}_i - \frac{2\|\mathbf{u}\|}{K} \sum_{j\in S_{ci}} \mathbf{T}_i^T\mathbf{c}_j + \frac{\|\mathbf{u}\|^2}{K^2} \left(\sum_{j\in S_{ci}} \mathbf{c}_j \right)^T \left(\sum_{j\in S_{ci}} \mathbf{c}_j \right) \right\} \\
&= \arg\min_{|S_{ci}|=k_i} \left\{ \mathbf{T}_i^T\mathbf{T}_i - \frac{2\|\mathbf{u}\|}{K} \sum_{j\in S_{ci}} \mathbf{T}_i^T\mathbf{c}_j + \frac{\|\mathbf{u}\|^2}{K^2} \left[k_i + (k_i^2 - k_i)\rho\right] \right\} \\
&= \arg\max_{|S_{ci}|=k_i} \left\{ \frac{2\|\mathbf{u}\|}{K} \sum_{j\in S_{ci}} \mathbf{T}_i^T\mathbf{c}_j \right\} = \text{the indices of } k_i \text{ largest } T_{si}(j)\text{'s},
\end{aligned} \tag{5.12}$$

where the vector $\mathbf{T}_{si}$ is defined as

$$\mathbf{T}_{si} = \mathbf{C}_i^T\mathbf{T}_i = \mathbf{C}^T\left[\frac{\mathbf{U}_i^T(\mathbf{y}-\mathbf{x})}{\|\mathbf{u}\|} \right] = \frac{\mathbf{S}_i^T(\mathbf{y}-\mathbf{x})}{\|\mathbf{s}\|}, \tag{5.13}$$

since $\|\mathbf{s}\| = \|\mathbf{u}\|$. We can see that $\mathbf{T}_{si}$ are the correlation statistics involving the colluded observation $\mathbf{y}$, the host signal $\mathbf{x}$, and the fingerprints $\mathbf{s}_{ij}$'s. Since $\mathbf{T}_{si} = \mathbf{C}^T\mathbf{T}_i$, $\mathbf{T}_{si}$ conditioned on K and S_{ci} is also Gaussian distributed with the mean vector and the covariance matrix decided as

$$\mu_i = \mathbf{C}^T E\{\mathbf{T}_i \mid K, S_{ci}, \sigma_d^2\} = \frac{\|\mathbf{u}\|}{K}\mathbf{R}\Phi_i, \tag{5.14}$$

$$\text{thus } \mu_i(j) = \begin{cases} \dfrac{1+(k_i-1)\rho}{K}\|\mathbf{s}\|, & \text{if } j \in S_{ci}, \\ \dfrac{k_i\rho}{K}\|\mathbf{s}\|, & \text{otherwise}, \end{cases} \tag{5.15}$$

$$R = \mathbf{C}^T \operatorname{Cov}\{\mathbf{T}_i \mid K, S_{ci}, \sigma_d^2\}\mathbf{C} = \sigma_d^2\mathbf{R},$$

according to the properties of the vector-valued Gaussian distribution [89]. In summary, suppose the parameters K and k_i are assumed known, we can estimate the subset S_{ci} via

$$\begin{aligned}\hat{S}_{ci} &= \arg\max_{|S_{ci}|=k_i} p(\mathbf{T}_i \mid K, S_{ci}, \sigma_d^2) \\ &= \text{the indices of } k_i \text{ largest } T_{si}(j)\text{'s}\end{aligned} \tag{5.16}$$

where the jth component of the correlator vector $\mathbf{T}_{si}$ is defined as

$$T_{si}(j) = \mathbf{T}_i^T \mathbf{c}_j = \frac{(\mathbf{y}-\mathbf{x})^T \mathbf{s}_{ij}}{\sqrt{\|\mathbf{s}\|^2}}. \tag{5.17}$$

However, applying (5.16) to locate colluders within group i is not preferred in our situation for two reasons. First, knowledge of K and k_i is usually not available in practice, and they must be estimated. Further, the above approach aims to minimize the joint estimation error of all colluders, and it lacks of the capability of adjusting parameters for addressing specific system design goals, such as minimizing the probability of a false positive and maximizing the probability of catching at least one colluder. Regardless of these concerns, the observation in (5.16) suggests the use of $\mathbf{T}_{si}$ for colluder detection within each group.

To overcome the limitations of the detector in (5.16), we employ a colluder identification approach within each group $i \in \hat{\mathbf{i}}$ by comparing the correlator $T_{si}(j)$ to a threshold h_i and indicating a colluder-presence whenever $T_{si}(j)$ is greater than the threshold. That is,

$$\hat{\mathbf{j}}_i = \arg_{j=1}^{M} \{\mathbf{T}_{si}(j) \geq h_i\}, \tag{5.18}$$

where the set $\hat{\mathbf{j}}_i$ indicates the indices of colluders within group i and the threshold h_i is determined by other parameters and the system requirements.

In our approach, we choose the thresholds such that false alarm probabilities satisfy

$$\begin{aligned} P_r\{T_G(i) \geq h_G \mid k_i = 0\} &= Q\left(\frac{h_G}{\sigma_d}\right) = \alpha_1, \\ P_r\{T_{si}(j) \geq h_i \mid k_i,\ j \notin S_{ci}\} &= Q\left(\frac{h_i - k_i\rho\|s\|/K}{\sigma_d}\right) = \alpha_2, \end{aligned} \tag{5.19}$$

where the Q-function is $Q(t) = \int_t^\infty (1/\sqrt{2\pi}) \exp(-x^2/2)dx$ and the values of α_1 and α_2 depend upon the system requirements.

When the fingerprint design scheme in (5.4) is applied to accommodate a large number of users, we observe the following:

$$T_{si}(j) = \frac{(\mathbf{y}-\mathbf{x})^T\mathbf{s}_{ij}}{\sqrt{\|\mathbf{s}\|^2}} = T_{ei}(j) + T_a(i), \quad \text{with } T_{ei}(j) = \frac{\sqrt{1-\rho}(\mathbf{y}-\mathbf{x})^T\mathbf{e}_{ij}}{\sqrt{\|\mathbf{s}\|^2}},$$
$$T_a(i) = \frac{\sqrt{\rho}(\mathbf{y}-\mathbf{x})^T\mathbf{a}_i}{\sqrt{\|\mathbf{s}\|^2}},$$
$$\text{thus } p(\mathbf{T}_{ei} \mid K, S_{ci}, \sigma_d^2) = N(\mu_{ei}, (1-\rho)\sigma_d^2\mathbf{I}_M),$$
$$\text{with } \mu_{ei}(j) = \begin{cases} \frac{1-\rho}{K}\|\mathbf{s}\|, & \text{if } j \in S_{ci}, \\ 0, & \text{otherwise,} \end{cases}$$
$$p(T_a(i) \mid K, S_{ci}, \sigma_d^2) = N\left(\frac{k_i\rho\|\mathbf{s}\|}{K}, \rho\sigma_d^2\right). \tag{5.20}$$

Since, for each group i, $T_a(i)$ is common for all $T_{si}(j)$'s, it is only useful in group detection and can be subtracted in detecting colluders. Therefore, the detection process (5.18) in Stage 2 now becomes

$$\hat{\mathbf{j}}_i = \arg_{j=1}^{M}\{\mathbf{T}_{ei}(j) \geq h\}. \tag{5.21}$$

Now the threshold h is chosen such that

$$P_r\{T_{ei}(j) \geq h \mid j \notin S_{ci}\} = Q\left(\frac{h}{\sigma_d\sqrt{1-\rho}}\right) = \alpha_2,$$
$$\text{thus } h = Q^{-1}(\alpha_2)\sigma_d\sqrt{1-\rho}. \tag{5.22}$$

Note that h is a common threshold for different groups. Advantages of the process (5.21) are that components of the vector $\mathbf{T}_{ei}$ are independent and that the resulting variance is smaller than σ_d^2.

5.2.3. Performance analysis

One important purpose of a multimedia fingerprinting system is to trace the individuals involved in digital content fraud and provide evidence to both the company administering the rights associated with the content and law enforcement agencies. In this section, we show the performance of the above fingerprinting system under different performance criteria. To compare with the orthogonal scheme

[58], we assume the overall MSE (mean square error) with respect to the host signal is constant. More specifically,

$$E\{\|\mathbf{y}-\mathbf{x}\|^2\} = \left(\frac{1-\rho}{K} + \frac{\rho\sum_{i=1}^{L} k_i^2}{K^2}\right)\|\mathbf{s}\|^2 + N\sigma_d^2 \triangleq \|\mathbf{s}\|^2, \tag{5.23}$$

meaning the overall MSE equals the fingerprint energy. Therefore, the variance σ_d^2 is based on $\{k_i\}$ correspondingly.

Different concerns arise in different fingerprinting applications. As discussed in Chapter 4, in studying the effectiveness of a detection algorithm in collusion applications, there are several performance criteria that may be considered. For instance, "catch one" performance criteria involves measuring the probability of a false negative (miss) and the probability of a false positive (false alarm). Such performance metrics are significant when presenting forensic evidence in a court of law, since it is important to quantify the reliability of the evidence when claiming an individual's guilt. On the other hand, if overall system security is a major concern, the goal would then be to quantify the likelihood of catching *all* colluders, since missed detection of any colluder may result in severe consequences. Further, multimedia fingerprinting may aim to provide evidence *supporting* the suspicion of a party. Tracing colluders via fingerprints should work in concert with other operations. For example, when a user is considered as a suspect based on multimedia forensic analysis, the agencies enforcing the digital rights can more closely monitor that user and gather additional evidence that can be used collectively for *proving* the user's guilt. Overall, identifying colluders through anticollusion fingerprinting is one important component of the whole forensic system, and it is the confidence in the fidelity of all evidence that allows a colluder to be finally identified and their guilt sustained in court. Similar to Chapter 4, we consider the following three sets of performance criteria. Without loss of generality, we assume $\mathbf{i} = [1, 2, \ldots, l]$, where $\mathbf{i}$ indicates the indices of groups containing colluders and l is the number of groups containing colluders.

Case 1 (catch at least one colluder). One of the most popular criteria explored by researchers are the probability of a false negative (P_{fn}) and the probability of a false positive (P_{fp}) [69, 70]. The major concern is to identify at least one colluder with high confidence without accusing innocent users. From the detector's point of view, a detection approach fails if either the detector fails to identify any of the colluders (a false negative) or the detector falsely indicates that an innocent user is a colluder (a false positive). We first define a false alarm event A_i and a correct detection event B_i for each group i,

$$\begin{aligned} A_i &= \left\{T_G(i) \geq h_G,\ \max_{j \notin S_{ci}} T_{si}(j) \geq h_i\right\}, \\ B_i &= \left\{T_G(i) \geq h_G,\ \max_{j \in S_{ci}} T_{si}(j) \geq h_i\right\}, \end{aligned} \tag{5.24}$$

for the scheme of (5.18), or

$$A_i = \left\{T_G(i) \geq h_G, \max_{j \notin S_{ci}} T_{ei}(j) \geq h\right\},$$
$$B_i = \left\{T_G(i) \geq h_G, \max_{j \in S_{ci}} T_{ei}(j) \geq h\right\}, \tag{5.25}$$

for the scheme of (5.21). Then we have

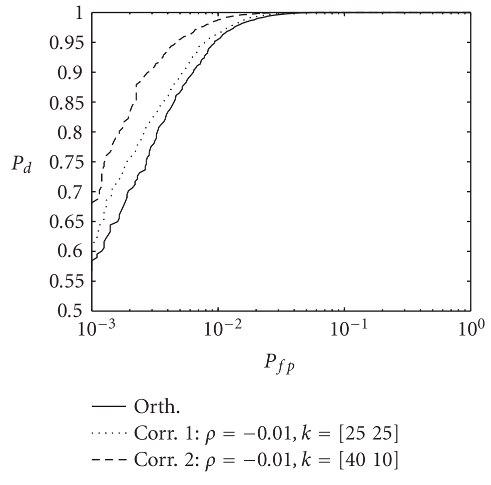
$$\begin{aligned}
P_d &= P_r\{\exists \hat{\mathbf{j}}_i \cap S_{ci} \neq \varnothing\} = P_r\left\{\bigcup_{i=1}^{l} B_i\right\}, \\
&= P_r\{B_1\} + P_r\{\bar{B}_1 \cap B_2\} + \cdots + P_r\{\bar{B}_1 \cap \bar{B}_2 \cap \cdots \cap \bar{B}_{l-1} \cap B_l\} \\
&= \sum_{i=1}^{l} q_i \prod_{j=1}^{i-1} (1 - q_j), \quad \text{with } q_i = P_r\{B_i\},
\end{aligned} \tag{5.26}$$

$$\begin{aligned}
P_{fp} &= P_r\{\exists \hat{\mathbf{j}}_i \cap \bar{S}_{ci} \neq \varnothing\} = P_r\left\{\bigcup_{i=1}^{L} A_i\right\} \\
&= P_r\left\{\bigcup_{i \notin \mathbf{i}} A_i\right\} + \left(1 - P_r\left\{\bigcup_{i \notin \mathbf{i}} A_i\right\}\right) P_r\left\{\bigcup_{i=1}^{l} A_i\right\} \\
&= \left[1 - (1 - p_{l+1})^{L-l}\right] + (1 - \alpha_1)^{L-l} P_r\left\{\bigcup_{i=1}^{l} A_i\right\} \\
&= \left[1 - (1 - p_{l+1})^{L-l}\right] + (1 - p_{l+1})^{L-l} \sum_{i=1}^{l} p_i \prod_{j=1}^{i-1} (1 - p_j), \quad \text{with } p_i = P_r\{A_i\}.
\end{aligned} \tag{5.27}$$

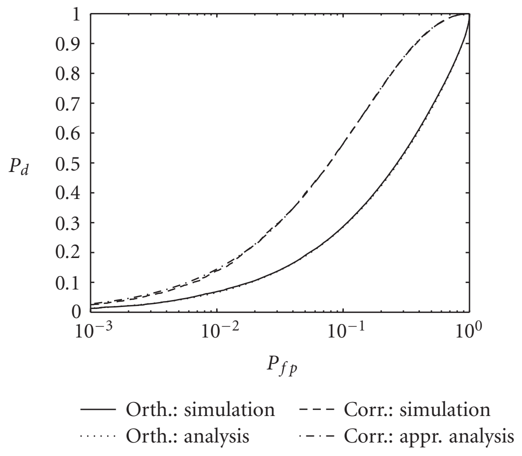
These formulas can be derived by utilizing the law of total probability in conjunction with the independency between fingerprints belonging to different groups, and the fact that $p_{l+1} = p_{l+2} = \cdots = p_L$ since there are no colluders in $\{A_{l+1}, \ldots, A_L\}$. Based on this pair of criteria, the system requirements are represented as

$$P_{fp} \leq \epsilon, \qquad P_d \geq \beta. \tag{5.28}$$

We can see that the difficulty in analyzing the collusion resistance lies in calculating joint probabilities p_i's and q_i's. When the total number of users is small such that all the users will belong to one or two groups, Stage 1 (guilty group identification) is normally unnecessary and thus ρ should be chosen to maximize the detection probability in Stage 2. We note that the detection performance is



(a)



(b)

FIGURE 5.2. ROC curves P_d versus P_{fp} of different examples, compared with the orthogonal scheme in [58]. In (a) a small number of users $n = 100$ and a negative $\rho = -0.01$ are considered. In (b) a large number of users $n = 6000$ and a positive $\rho = 0.4$ are considered.

characterized by the difference between the means of the two hypotheses in (5.15), and hence is given by $(1 - \rho)\|\mathbf{s}\|/K$. Therefore a negative ρ is preferred. Since the matrix $\mathbf{R}$ should be positive definite, $1 + (M - 1)\rho > 0$ is required. We show the performance by examples when the total number of users is small, as in Figure 5.2a where $n = 100$, $M = 50$, and a negative $\rho = -0.01$ is used. It is clear that introducing a negative ρ helps to improve the performance when n is small. It also reveals that the worst case in performance happens when each guilty group contributes equal number of colluders, meaning $k_i = K/|\mathbf{i}|$, for $i \in \mathbf{i}$.

In most applications, however, the total number of users n is large. Therefore, we focus on this situation for performance analysis. One approach to accommodate large n is to design the fingerprints according to (5.4) and use a positive value of ρ. Now after applying the detection scheme in (5.21), the events A_i's and B_i's are defined as in (5.25). We further note, referring to (5.6) and (5.20), that the correlation coefficient between $T_G(i)$ and $T_{ei}(j)$ is equal to $\sqrt{(1-\rho)/(M+(M^2-M)\rho)}$, which is a small value close to 0. For instance, with $\rho = 0.2$ and $M = 60$, this correlation coefficient is as small as 0.03. This observation suggests that $T_G(i)$ and $T_{ei}(j)$'s are approximately uncorrelated, therefore we have the following approximations

$$\begin{aligned} p_i &\approx P_r\{T_G(i) \geq h_G\} P_r\Big\{\max_{j \notin S_{ci}} T_{ei}(j) \geq h\Big\} \\ &= Q\left(\frac{h_G - k_i r_0}{\sigma_d}\right)\left[1 - \left(1 - Q\left(\frac{h}{\sigma_d\sqrt{1-\rho}}\right)\right)^{M-k_i}\right], \\ q_i &\approx P_r\{T_G(i) \geq h_G\} P_r\Big\{\max_{j \in S_{ci}} T_{ei}(j) \geq h\Big\} \\ &= Q\left(\frac{h_G - k_i r_0}{\sigma_d}\right)\left[1 - \left(1 - Q\left(\frac{h-(1-\rho)\|\mathbf{s}\|/K}{\sqrt{1-\rho}\,\sigma_d}\right)\right)^{k_i}\right] \end{aligned} \tag{5.29}$$

in calculating P_{fp} and P_d in (5.27), with $r_0 = (\|\mathbf{s}\|\sqrt{1+(M-1)\rho})/K\sqrt{M}$. Note that here we employ the theory of order statistics [90]. We show an example in Figure 5.2b where $n = 6000$, $L = 100$, and there are eight groups involved in collusion with each group having eight colluders. We note that this approximation is very accurate compared to the simulation result and that our fingerprinting scheme is superior to using orthogonal fingerprints.

To have an overall understanding of the collusion resistance of the proposed scheme, we further study the maximum resistable number of colluders $K_{\max}$ as a function of n. For a given n, M, and $\{k_i\}$'s, we choose the parameters α_1, which determines the threshold h_G, α_2, which determines the threshold h, and ρ, which determines the probability of the group detection, so that

$$\{\alpha_1, \alpha_2, \rho\} = \arg \max_{\{\alpha_1,\alpha_2,\rho\}} P_d(\alpha_1, \alpha_2, \rho) \quad \text{subject to } P_{fp}(\alpha_1, \alpha_2, \rho) \leq \epsilon. \tag{5.30}$$

In reality, the value of ρ is limited by the quantization precision of the image system and ρ should be chosen at the fingerprint design stage. Therefore, ρ is fixed in real applications. Since, in many collusion scenarios the size $|\mathbf{i}|$ would be reasonably small, our results are not as sensitive to α_1 and ρ as to α_2, and the group detection in Stage 1 often yields very high accuracy. For example, when $|\mathbf{i}| \leq 5$, the

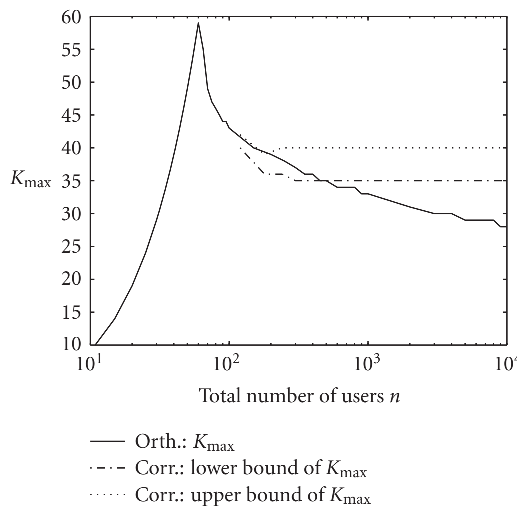


FIGURE 5.3. Comparison of collusion resistance of the orthogonal and the proposed group-based fingerprinting systems to the average attack. Here $M = 60$, $k_i = K/|i|$, $|i| = 5$, and the system requirements are represented by $\epsilon = 10^{-3}$ and $\beta = 0.8$.

threshold h_G can be chosen such that $\alpha_1 \ll \epsilon$ and $P_r(T_G(i) \geq h_G)$ is sufficiently close to 1 for at least one group $i \in \mathbf{i}$. Therefore, to simplify our searching process, we can fix the values of α_1. Also, in the design stage, we consider the performance of the worst case where $k_i = K/|\mathbf{i}|$, for $i \in \mathbf{i}$. One important efficiency measure of a fingerprinting scheme is $K_{\max}$, the maximum number of colluders that can be tolerated by a fingerprinting system such that the system requirements are still satisfied. We illustrate an example in Figure 5.3, where $M = 60$ is used since it is shown to be the best supportable user size for the orthogonal scheme [58, 59], and the number of guilty groups is up to five. It is noted that $K_{\max}$ of the proposed scheme (indicated by the dotted and the dashed-dotted lines) is larger than that of the orthogonal scheme (the solid line), when n is large. The difference between the lower bound and upper bound is due to the fact that $k_i = K/|\mathbf{i}|$ in our simulations. Overall, the group-oriented fingerprinting system provides the performance improvement by yielding better collusion resistance. It is worth mentioning that the performance is fundamentally affected by the collusion pattern. The smaller the number of guilty groups, the better chance the colluders are identified.

Case 2 (fraction of guilty captured versus fraction of innocent accused). This set of performance criteria consists of the expected fraction of colluders that are successfully captured, denoted as r_c, and the expected fraction of innocent users that are falsely placed under suspicion, denoted as r_i. Here the major concern is to catch more colluders, possibly at the cost of accusing more innocents. The balance between capturing colluders and placing innocents under suspicion is represented by these two expected fractions. Suppose the total number of users n is large and the detection scheme in (5.21) is applied. We have

$$\begin{aligned} r_i &= \frac{E\left(\sum_{i=1}^{l}\sum_{j\notin S_{ci}}\gamma_{ij}+\sum_{i=l+1}^{L}\sum_{j=1}^{M}\gamma_{ij}\right)}{n-K} \\ &= \frac{\sum_{i=1}^{l}(M-k_i)p_{0i}+M(L-l)p_{0,l+1}}{n-K}, \\ r_c &= \frac{E\left(\sum_{i=1}^{l}\sum_{j\in S_{ci}}\gamma_{ij}\right)}{K}=\frac{\sum_{i=1}^{l}k_i p_{1i}}{K}, \end{aligned} \tag{5.31}$$

where

$$\begin{aligned} p_{1i} &= P_r\{T_G(i)\ge h_G, T_{ei}(j)\ge h_i \mid j\in S_{ci}\}, \quad \text{for } i=1,\dots,l, \\ p_{0i} &= P_r\{T_G(i)\ge h_G, T_{ei}(j)\ge h_i \mid j\notin S_{ci}\}, \quad \text{for } i=1,\dots,l+1, \end{aligned} \tag{5.32}$$

and γ_{ij} is defined as

$$\gamma_{ij}=\begin{cases}1, & \text{if } j\text{th user of group } i \text{ is accused},\\ 0, & \text{otherwise.}\end{cases} \tag{5.33}$$

Based on this pair $\{r_i, r_c\}$, the system requirements are represented by

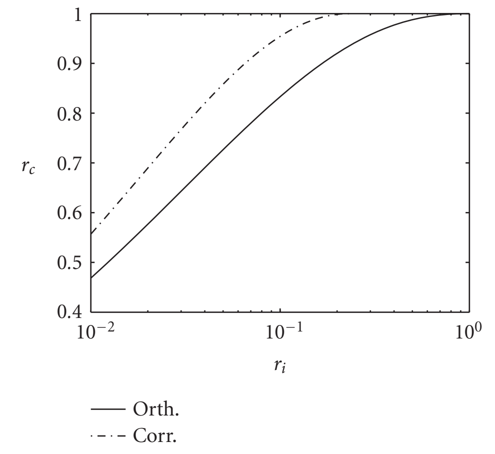
$$r_i \le \alpha_i, \qquad r_c \ge \alpha_c. \tag{5.34}$$

We further notice that $T_G(i)$ and $T_{ei}(j)$'s are approximately uncorrelated, therefore we can approximately apply $p_{1i} = P_r\{T_G(i) \ge h_G\}P_r\{T_{ei}(j) \ge h \mid j \in S_{ci}\}$, for $i = 1,\dots,l$, and $p_{0i} = P_r\{T_G(i) \ge h_G\}P_r\{T_{ei}(j) \ge h \mid j \notin S_{ci}\}$, for $i = 1,\dots,l+1$, in calculating r_i and r_c. With a given n, M, and $\{k_i\}$'s, the parameters α_1 which determines the threshold h_G, α_2 which determines the threshold h, and ρ which determines the probability of the group detection are chosen such that

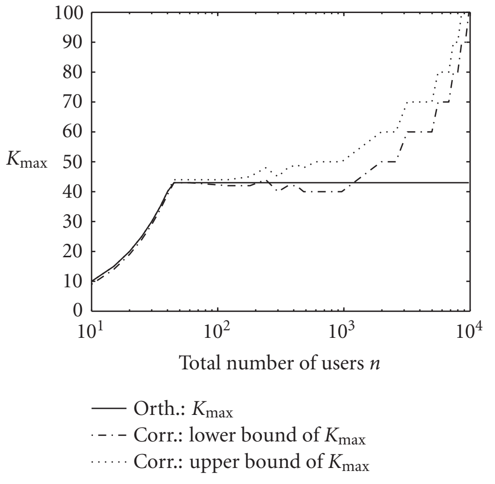
$$\max_{\{\alpha_1,\alpha_2,\rho\}} r_c(\alpha_1,\alpha_2,\rho) \quad \text{subject to } r_i(\alpha_1,\alpha_2,\rho)\le\alpha_i. \tag{5.35}$$

Similarly, finite discrete values of α_1 and ρ are considered to reduce the computational complexity.

We first illustrate the resistance performance of the system by an example, shown in Figure 5.4a, where $N = 10^4$, $\rho = 0.2$, and three groups are involved in collusion, with each group including 15 colluders. We note that the proposed scheme is superior to using orthogonal fingerprints. In particular, for the proposed scheme, all colluders are identified as long as we allow 10 percent innocents to be wrongly accused. We further examine $K_{\max}$ for the case $k_i = K/|\mathbf{i}|$ when different number of users is managed, as shown in Figure 5.4b by requiring $r \le 0.01$ and $P_d \ge 0.5$ and setting $M = 60$ and the number of guilty groups is up to ten. The $K_{\max}$ of our proposed scheme is larger than that of $K_{\max}$ for orthogonal fingerprinting when large n is considered.



(a)



(b)

FIGURE 5.4. The resistance performance of the group-oriented and the orthogonal fingerprinting system under the criteria r_i and r_c. Here $N = 10^4$. In (a) we have $M = 50$, $n = 500$, $\rho = 0.2$, and there are three groups involved in collusion, with each group having 15 colluders. $K_{\max}$ versus n is plotted in (b) where $M = 60$, the numbers of colluders within guilty groups are equal, meaning $k_i = K/|\mathbf{i}|$, the number of guilty groups is $|\mathbf{i}| = 10$, and the system requirements are represented by $\alpha = 0.01$ and $\beta = 0.5$.

Case 3 (catch all colluders). This set of performance criteria consists of the efficiency rate r, which describes the amount of expected innocents accused per colluder, and the probability of capturing all K colluders, which we denote by P_d. The goal in this scenario is to capture all colluders with a high probability. The trade-off between capturing colluders and placing innocents under suspicion is achieved through the adjustment of the efficiency rate r. More specifically, supposing n is large and the detection scheme in (5.21) is applied, we have

$$\begin{aligned} r &= \frac{E\Big(\sum_{i=1}^{l}\sum_{j\notin S_{ci}}\gamma_{ij}+\sum_{i=l+1}^{L}\sum_{j=1}^{M}\gamma_{ij}\Big)}{E\Big(\sum_{i=1}^{l}\sum_{j\in S_{ci}}\gamma_{ij}\Big)} \\ &= \frac{\sum_{i=1}^{l}(M-k_i)p_{0i}+M(L-l)p_{0,l+1}}{\sum_{i=1}^{l}k_i p_{1i}}, \\ P_d &= P_r\{\forall S_{ci}\subseteq \hat{\mathbf{j}}_i\} = \prod_{i=1}^{l}P_r\{C_i\}, \\ &\text{with } C_i = \Big\{T_G(i)\geq h_G,\ \min_{j\in S_{ci}}T_{ei}(j)\geq h\Big\}, \end{aligned} \tag{5.36}$$

in which p_{0i} and p_{1i} are defined as in (5.31). Based on this pair $\{r, P_d\}$, the system requirements are expressed as

$$r \leq \alpha, \qquad P_d \geq \beta. \tag{5.37}$$

Similar to the previous cases, we further notice that $T_G(i)$ and $T_{ei}(j)$'s are approximately uncorrelated, and we may approximately calculate p_{1i}'s and p_{0i}'s as done earlier. Using the independency, we also apply the approximation

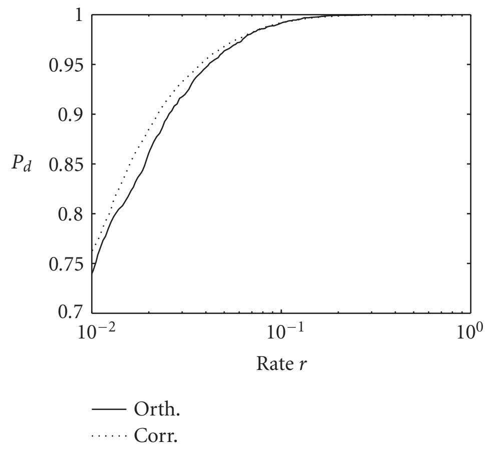
$$\begin{aligned} P_r\{C_i\} &= P_r\{T_G(i)\geq h_G\}P_r\Big\{\min_{j\in S_{ci}}T_{ei}(j)\geq h\Big\} \\ &= Q\bigg(\frac{h_G-k_i r_0}{\sigma_d}\bigg)Q\bigg(\frac{h-(1-\rho)\|\mathbf{s}\|}{\sigma_d\sqrt{1-\rho}}\bigg)^{k_i} \end{aligned} \tag{5.38}$$

in calculating P_d. With a given n, M, and $\{k_i\}$'s, the parameters α_1 which determines the threshold h_G, α_2 which determines the threshold h, and ρ which determines the probability of the group detection, are chosen such that

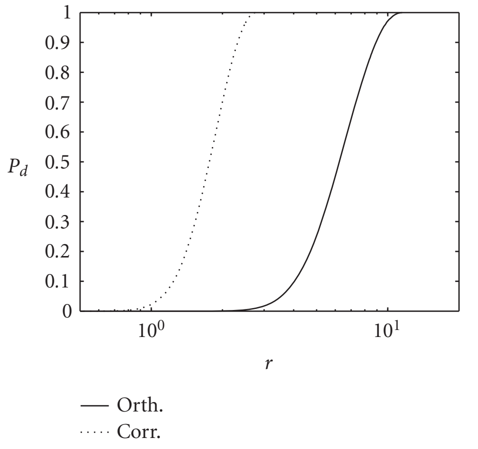
$$\max_{\{\alpha_1,\alpha_2,\rho\}} P_d(\alpha_1,\alpha_2,\rho) \quad \text{subject to } r(\alpha_1,\alpha_2,\rho)\leq\alpha. \tag{5.39}$$

Similarly, finite discrete values of α_1 and ρ are considered to reduce the computational complexity.

We illustrate the resistance performance of the proposed system by two examples shown in Figure 5.5. It is worth mentioning that the accuracy in the group



(a)



(b)

FIGURE 5.5. Performance curves P_d versus r of different examples, compared with the orthogonal scheme in [58]. In (a) $K = 23$, a small number of users $n = 40$, and a negative $\rho = -0.023$ are considered. In (b) a large number of users $n = 600$ and a positive $\rho = 0.3$ are considered. Three groups are involved in collusion, with numbers of colluders being 32, 8, and 8, respectively.

detection stage is critical for this set of criteria, since a missed detection in Stage 1 will result in a much smaller P_d. When capturing all colluders with high probability is a major concern, our proposed group-oriented scheme may not be favorable in cases where there are a moderate number of guilty groups involved in collusion or when the collusion pattern is highly asymmetric. The reason is that, under these situations, a threshold in Stage 1 should be low enough to identify all colluder-present groups; however, a low threshold also results in wrongly accusing innocent groups. Therefore, Stage 1 is not very useful in these situations.

5.3. Tree-structure-based fingerprinting system

In this section, we propose to extend our construction to represent the natural social and geographic hierarchical relationships between users by generalizing the two-tier approach to a more flexible group-oriented fingerprinting system based on a tree structure. As in the two-tier group-oriented system, to validate the improvement of such tree-based group fingerprinting, we will evaluate the performance of our proposed system under the average attack and compare the resulting collusion resistance to that of an orthogonal fingerprinting system.

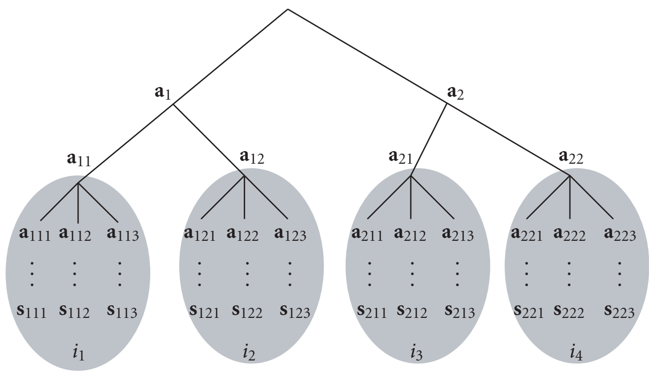
5.3.1. Fingerprint design scheme

The group-oriented system proposed earlier can be viewed as a symmetric two-level tree-structured scheme. The first level consists of L nodes, with each node supporting P leaves that correspond to the fingerprints of individual users within one group. We observe that a user is often more likely to collude with some groups than with other groups. If we allow for a more general tree structure in the fingerprint design, we can achieve more flexibility in capturing the collusion dynamics between different groups. For instance, we may consider a simple region-based collusion pattern: users within Maryland are more likely to come together in generating an attacked copy than they are likely to collude with other users from Texas, and the probability for these two groups to come together to collude is higher than they would with users from Asia. We may view this subgroup hierarchy via a tree structure, as depicted in Figure 5.6. In this diagram, we assume that (1) users in region $\mathbf{i}_1$ are equally likely to collude with each other with a probability p_1, (2) each user in region $\mathbf{i}_1$ is equally likely to colluder with users within region $\mathbf{i}_2$ with a probability p_2, and (3) with users within other regions corresponding to different subtrees with a probability p_3, where $p_1 > p_2 > p_3$. Therefore, it is desirable for us to design a fingerprint tree that matches the large-scale collusion pattern (e.g., represented by the cultural, social, and geographic relationships among users) in such a way that the fingerprints on the same branch of the tree are more correlated with each other than with those on other branches, and correspondingly, the associated users on the same branch of the tree are more likely to collude with each other.

More generally, we design a tree with M levels where each node at the $(m-1)$th level supports a total of L_m nodes. Let $[i_1, \dots, i_M]$ indicate the index vector of a user/fingerprint. Exploiting the tree structure, we propose the following design of fingerprints $\{\mathbf{s}_{i_1,\dots,i_M}\}$:

$$\mathbf{s}_{i_1,\dots,i_M} = \sqrt{\rho_1}\mathbf{a}_{i_1} + \cdots + \sqrt{\rho_{M-1}}\mathbf{a}_{i_1,\dots,i_{M-1}} + \sqrt{1 - \sum_{j=1}^{M-1} \rho_j}\mathbf{a}_{i_1,\dots,i_M}, \tag{5.40}$$

where the $\mathbf{a}$ vectors correspond to orthogonal basis vectors with equal energy $\|\mathbf{a}\| = \|\mathbf{s}\|$, each ρ_j satisfies $0 \le \rho_j \le 1$, and $\rho_M = 1 - \sum_{j=1}^{M-1} \rho_j$. In this design



$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_{11}$
$\mathbf{a}_{12}$
$\mathbf{a}_{21}$
$\mathbf{a}_{22}$
$\mathbf{a}_{111}$ $\mathbf{a}_{112}$ $\mathbf{a}_{113}$
$\mathbf{a}_{121}$ $\mathbf{a}_{122}$ $\mathbf{a}_{123}$
$\mathbf{a}_{211}$ $\mathbf{a}_{212}$ $\mathbf{a}_{213}$
$\mathbf{a}_{221}$ $\mathbf{a}_{222}$ $\mathbf{a}_{223}$
$\mathbf{s}_{111}$ $\mathbf{s}_{112}$ $\mathbf{s}_{113}$
$\mathbf{s}_{121}$ $\mathbf{s}_{122}$ $\mathbf{s}_{123}$
$\mathbf{s}_{211}$ $\mathbf{s}_{212}$ $\mathbf{s}_{213}$
$\mathbf{s}_{221}$ $\mathbf{s}_{222}$ $\mathbf{s}_{223}$
i_1
i_2
i_3
i_4

FIGURE 5.6. A tree-structure-based fingerprinting scheme.

scheme, the correlations between fingerprints are controlled by adjusting the coefficients ρ_i's, which are determined by the probabilities for users under different tree branches to carry out collusion attacks.

5.3.2. Detection scheme

We now discuss the problem of detecting the colluders when the proposed fingerprint design scheme in (5.40) is employed. For simplicity in analysis, we consider a balanced tree structure, where the system accommodates n users, and the tree involves M levels where each node at the $(m-1)$th level supports L_m nodes. The marked copy for a user with the index vector $[i_1, \dots, i_M]$ is represented as $\mathbf{y}_{i_1,\dots,i_M} = \mathbf{x} + \mathbf{s}_{i_1,\dots,i_M}$, where $\mathbf{x}$ is the host signal. When a collusion occurs, suppose that a total of K colluders are involved in forming a copy of colluded content $\mathbf{y}$, and the number of colluders within each level-m subregion represented by an index vector $[i_1, \dots, i_m]$ is $k_{i_1,\dots,i_m}$. For instance, for a tree with $M = 3$, in a subregion where users are all with indices $i_1 = 2$ and $i_2 = 1$, if $\mathbf{s}_{2,1,1}$ and $\mathbf{s}_{2,1,3}$ are colluders, then $k_{2,1}$ equals 2. We note that, for each level $m = 1, \dots, M$, we have $\sum_{i_1=1}^{L_1} \cdots \sum_{i_m=1}^{L_m} k_{i_1,\dots,i_m} = K$. The observed content $\mathbf{y}$ after the average collusion is

$$\mathbf{y} = \frac{1}{K} \sum_{\mathbf{i}_c \in \mathbf{S}_c} \mathbf{y}_{\mathbf{i}_c} + \mathbf{d} = \frac{1}{K} \sum_{\mathbf{i}_c \in \mathbf{S}_c} \mathbf{s}_{\mathbf{i}_c} + \mathbf{x} + \mathbf{d}, \tag{5.41}$$

where $\mathbf{i}_c$ indicates the index vector of length M, $\mathbf{S}_c$ indicates a vector set of size K, and each element of $\mathbf{S}_c$ is an index vector. We also assume that additional noise $\mathbf{d}$ is introduced after the average collusion and $\mathbf{d}$ is a vector following an i.i.d. Gaussian distribution with zero mean and variance σ_d^2. The number of colluders K and the set $\mathbf{S}_c$ are the parameters to be estimated. We consider the nonblind scenario, where the host signal $\mathbf{x}$ is available at the detector and thus always subtracted from $\mathbf{y}$ for analysis.

Using such a formulation, we will address the issue of detecting the colluders. The tree-structured, hierarchical nature of group-oriented fingerprints leads to a *multistage* colluder identification scheme: the first stage focuses on identifying the "guilty" regions at the first level; at the second stage, we further narrow down by specifying "guilty" subregions within each "guilty" region. We continue the process along each "guilty" branch of the tree until we detect the colluders at the leaf level. More specifically, at each level m, with $m = 1, \dots, M$, and with a previously identified region indexed by $\mathbf{i} = [i_1, \dots, i_{m-1}]$, we report that the subregion indexed by $\mathbf{i} = [i_1, \dots, i_m]$ is *colluder-present* when the correlator $T_{i_1,\dots,i_{m-1}}(i_m)$ is greater than a threshold h_m. That is, for $m = 1, \dots, (M-1)$, we define Stage m in the overall detection scheme as follows.

Stage m: subregion detection at level m of the tree structure. With a previously identified region indexed by $\mathbf{i} = [i_1, \dots, i_{m-1}]$, we need to further examine the subregions indexed by $\mathbf{i} = [i_1, \dots, i_m]$ for $i_m = 1, \dots, L_m$. Due to the orthogonality of basis $\{\mathbf{a}_{i_1,\dots,i_m}\}$, it suffices to consider the (normalized) correlator vector $\mathbf{T}_{i_1,\dots,i_{m-1}}$ for identifying subregions including colluders. The i_mth component of $\mathbf{T}_{i_1,\dots,i_{m-1}}$ is expressed by

$$T_{i_1,\dots,i_{m-1}}(i_m) = \frac{(\mathbf{y}-\mathbf{x})^T \mathbf{a}_{i_1,\dots,i_m}}{\sqrt{\|\mathbf{s}\|^2}}, \tag{5.42}$$

for $i_m = 1, \dots, L_m$. We can show that

$$p(\mathbf{T}_{i_1,\dots,i_{m-1}} \mid K, \mathbf{S}_c, \sigma_d^2) = N(\mu_{i_1,\dots,i_{m-1}}, \sigma_d^2 \mathbf{I}_{L_m}) \tag{5.43}$$

with

$$\mu_{i_1,\dots,i_{m-1}}(i_m) = \frac{k_{i_1,\dots,i_m}\sqrt{\rho_m}}{K}\|\mathbf{s}\| \tag{5.44}$$

and $k_{i_1,\dots,i_m} = 0$ indicating that no colluder is present within the subregion represented by $[i_1, \dots, i_m]$. If many colluders belong to the subregion represented by $[i_1, \dots, i_m]$, we are likely to observe a large value of $T_{i_1,\dots,i_{m-1}}(i_m)$. Therefore, the detection process in Stage m is

$$\widehat{\mathbf{j}}_m = \arg_{i_m=1}^{L_m} \{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m\}, \tag{5.45}$$

where $\widehat{\mathbf{j}}_m$ indicates the indices of subregions containing colluders within the previously identified region represented by $[i_1, \dots, i_{m-1}]$.

Finally, we note that the individual colluders are identified at level M (the leaf level). Now with previously identified region represented by $[i_1, \dots, i_{M-1}]$, we have, for $i_M = 1, \dots, L_M$,

$$\begin{gathered} T_{i_1,\dots,i_{M-1}}(i_M) = \frac{(\mathbf{y}-\mathbf{x})^T \mathbf{a}_{i_1,\dots,i_M}}{\sqrt{\|\mathbf{s}\|^2}}, \\ \text{with } p(\mathbf{T}_{i_1,\dots,i_{M-1}} \mid K, \mathbf{S}_c, \sigma_d^2) = N(\mu_{i_1,\dots,i_{M-1}}, \sigma_d^2 \mathbf{I}_{L_m}), \end{gathered} \tag{5.46}$$

where

$$\mu_{i_1,\ldots,i_{M-1}}(i_M) = \begin{cases} \dfrac{\sqrt{1-\sum_{m=1}^{M-1}\rho_m}}{K}\|\mathbf{s}\|, & \text{if } k_{i_1,\ldots,i_M} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.47}$$

Now the detection process in Stage M is

$$\hat{\mathbf{j}}_M = \arg_{i_M=1}^{L_M}\{T_{i_1,\ldots,i_{M-1}}(i_M) \geq h_M\}, \tag{5.48}$$

where $\hat{\mathbf{j}}_M$ indicates the indices of colluders within the previously identified region represented by $[i_1,\ldots,i_{M-1}]$.

In our approach, at each level m, we specify a desired false positive probability α_m and choose the threshold h_m such that

$$P_r\{T_{i_1,\ldots,i_{m-1}}(i_m) \geq h_m \mid k_{i_1,\ldots,i_m} = 0\} = Q\left(\frac{h_m}{\sigma_d}\right) = \alpha_m, \tag{5.49}$$

thus

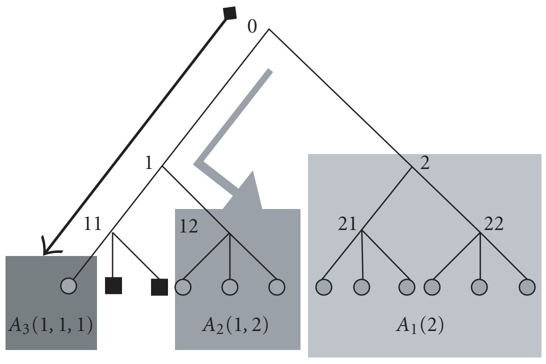
$$h_m = Q^{-1}(\alpha_m)\sigma_d. \tag{5.50}$$

In summary, the basic idea behind this multistage detection scheme is to keep narrowing down the size of the suspicious set. An advantage of this approach is its light computational burden since, when the number of colluders K is small or the number of subregions involved in collusion is small, the total amount of correlations needed can be significantly less than the total number of users.

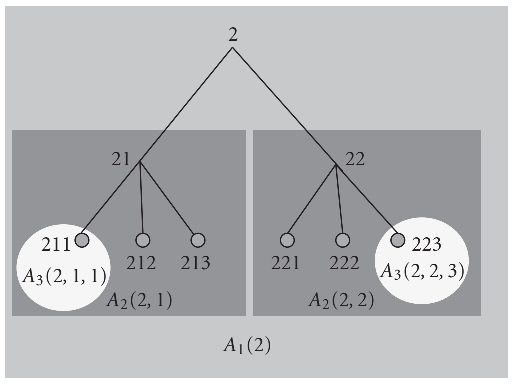
5.3.3. Parameter settings and performance analysis

In this subsection, we will address the issue of setting the parameters (e.g., how to choose the values of coefficients ρ_m's, thresholds h_m's, and the sizes L_m's) and examine the performance metrics characterized by P_{fp} and P_d. Due to the multistage nature of the proposed detection approach, calculating the overall performance P_{fp} and P_{fn} involves computing the probabilities of joint events. Furthermore, the collusion pattern will also make the analysis of P_{fp} and P_d complicated.

We first examine the types of false alarm events possible for our tree-structured scheme. A false alarm occurs when the detector claims colluders are present in a colluder-absent region. A colluder-absent region is characterized by $k_{i_1,\ldots,i_m} = 0$. As shown in Figure 5.7, where the gray rectangles represent colluder-absent



(a)



(b)

FIGURE 5.7. Demonstration of the types of false alarm events for a three-level tree structure, where at the leaf level the square-shaped nodes indicate colluders and the circle-shaped nodes indicate innocents. (a) The dark and lighter arrows represent a event in B_3 and B_2, respectively. (b) The event $A_1(2)$.

regions, we can characterize false alarm events in these regions by $A_M(\cdot), A_{M-1}(\cdot), \ldots, A_1(\cdot)$:

$$A_M(i_1,\ldots,i_M) \triangleq \{T_{i_1,\ldots,i_{M-1}}(i_M) \geq h_M \mid k_{i_1,\ldots,i_M} = 0\},$$
$$A_m(i_1,\ldots,i_m) \triangleq \left\{T_{i_1,\ldots,i_{m-1}}(i_m) \geq h_m, \bigcup_{i_{m+1}} A_{m+1}(i_1,\ldots,i_m,i_{m+1}) \mid k_{i_1,\ldots,i_m} = 0\right\}. \tag{5.51}$$

The probability of these events is given by

$$p_M = P_r\{A_M(i_1,\ldots,i_M)\} = \alpha_M,$$
$$p_m = P_r\{A_m(i_1,\ldots,i_m)\} = \alpha_m\left[1-(1-p_{m+1})^{L_{m+1}}\right] < \alpha_m L_{m+1} p_{m+1}, \tag{5.52}$$

for $m = (M-1), \dots, 1$. Denoting the index vectors for the estimated colluders as $\{\hat{\mathbf{i}}_c\}$, we now have

$$P_{fp} = P_r\{\exists \hat{\mathbf{i}}_c \in \bar{\mathbf{S}}_c\} = P_r\left\{\bigcup_m B_m\right\}, \tag{5.53}$$

with

$$\begin{aligned} B_1 &\triangleq \left\{\bigcup_{\{i_1 | k_{i_1}=0\}} A_1(i_1)\right\}, \quad \text{for } m = 2, \dots, M, \\ B_m &\triangleq \left\{\bigcup_{\mathbf{S}_m} \left(T_0(i_1) \geq h_1, \dots, T_{i_1,\dots,i_{m-2}}(i_{m-1}) \geq h_{m-1}, A_m(i_1, \dots, i_m)\right)\right\}, \end{aligned} \tag{5.54}$$

where the vector set $\mathbf{S}_m = \{\{i_1, \dots, i_m\} | \{k_{i_1} \neq 0, \dots, k_{i_1,\dots,i_{m-1}} \neq 0, k_{i_1,\dots,i_m} = 0\}$. As we can see, due to the complex nature of a collusion pattern represented in the tree structure, P_{fp} is a complicated function of the collusion pattern.

During the system design process, we normally do not have knowledge of the location of the colluders. As such, we use the upper bound of P_{fp}, which does not require detailed knowledge of the collusion pattern, to guide our selection of parameter values. Let K be the total number of colluders. Based on the analysis of probability and order statistics [90, 91], we have

$$P_{fp} \leq \sum_{m=1}^{M} P_r\{B_m\} < L_1 p_1 + K \sum_{m=2}^{M-1} L_m p_m + Kp, \tag{5.55}$$

where $p = 1 - (1 - p_M)^{L_M}$ represents the probability of a false alarm within a subregion as $[i_1, \dots, i_{M-1}]$ where all users are innocents. As we can see, the $L_m p_m$ term in the above expression is due to the type of false alarm event A_m. Intuitively, we want the probability of an event of type A_m to decrease with decreasing level m. In particular, we want the probability that a false alarm occurs in an innocent region connected directly to the root $P_r\{B_1\}$ to be negligible, thus implying that α_1 is small. This is due to the fact that our tree-structured fingerprint system can be deployed in such a way that typically only a very small number of regions at the first level are involved in collusion, thus a miss detection is rare at the first level even with a high threshold h_1. To simplify the parameter setting process, we relate the false alarm probabilities at different levels with a multiplicative factor c. That is, if at the leaf level we have the probability of a false positive represented by p, then for the $(M-1)$ level, we scale p by a factor of c and use p/c to represent the probability of the events of type B_{M-1}. We apply this scaling to upper levels in a similar way. Further, using the upper bound of p_m in (5.53), we can summarize the process as

$$\begin{aligned} L_{M-1}p_{M-1} &= L_{M-1}(\alpha_{M-1}p) = \frac{p}{c}, \\ &\longrightarrow \alpha_{M-1} = \frac{1}{L_{M-1}c}, \\ L_m p_m < L_m(\alpha_m L_{m+1}p_{m+1}) &= \frac{L_{m+1}p_{m+1}}{c}, \\ &\longrightarrow \alpha_m = \frac{1}{L_m c}, \quad \text{for } m = M-2,\dots,2. \end{aligned} \tag{5.56}$$

Based on the expression of P_{fp} in (5.55), we have $P_{fp} \le \sum_{m=1}^{M} P_r\{B_m\}$. Recall the definition of B_m's and that $\sum_{i_1=1}^{L_1}\cdots\sum_{i_m=1}^{L_m} k_{i_1,\dots,i_m} = K$. We note that the size $|\mathbf{S}_1| = |\{i_1 \mid k_{i_1} = 0\}| \le L_1$, that the size of the colluder-present regions satisfying $\{k_{i_1} \ne 0,\dots,k_{i_1,\dots,i_{m-1}} \ne 0\}$ is smaller than K, and therefore that the size of $\mathbf{S}_m$ satisfies $|\mathbf{S}_m| \le KL_m$ for $m = 2,\dots,M$. Therefore, by taking advantage of the independency of the basis vectors $\mathbf{a}$'s, we have

$$\begin{aligned} P_r\{B_1\} &\le 1-(1-p_1)^{L_1} < L_1 p_1, \quad \text{for } m = 2,\dots,M, \\ P_r\{B_m\} &\le KP_r\Big\{T_0(i_1) \ge h_1,\dots,T_{i_1,\dots,i_{m-2}}(i_{m-1}) \ge h_{m-1}, \bigcup_{i_m} A_m(i_1,\dots,i_m\}\Big\}, \\ &= K\prod_{j=1}^{m-1} P_r\{T_{i_1,\dots,i_{j-1}}(i_j) \ge h_{m-1}\}P_r\Big\{\bigcup_{i_m} A_m(i_1,\dots,i_m\}\Big\}, \\ &\le KP_r\Big\{\bigcup_{i_m} A_m(i_1,\dots,i_m\}\Big\} \le K\Big[1-(1-p_m)^{L_m}\Big] < KL_m p_m. \end{aligned} \tag{5.57}$$

By defining $p = [1-(1-p_M)^{L_M}]$ in the above, we have $P_r\{B_M\} \le Kp$. Putting all these inequations together, we have

$$P_{fp} \le \sum_{m=1}^{M} P_r\{B_m\} \le L_1 p_1 + K\sum_{m=2}^{M-1} L_m p_m + Kp. \tag{5.58}$$

By choosing $\alpha_m = 1/(L_m c)$ and using that $p_m < \alpha_m L_{m+1}p_{m+1}$ for $m = 1,\dots,(M-1)$, and choosing α_1 such that $P_r\{B_1\}$ is negligible in comparison with other terms $P_r\{B_m\}$'s, we now have

$$P_{fp} < Kp\left(o\left(\frac{1}{c^{M-1}}\right) + \frac{1}{c^{M-1}} + \cdots + \frac{1}{c^2} + \frac{1}{c} + 1\right) < \frac{Kpc}{c-1}, \tag{5.59}$$

where $o(a)$ represents a small value compared with a, and c is a positive constant larger than 1. Basically, for larger K and p, or for smaller c, we will see a larger P_{fp}.

Based on the chosen α_m's, we can set the threshold at level m as

$$h_m = Q^{-1}(\alpha_m)\sigma_d = Q^{-1}\left(\frac{1}{cL_m}\right)\sigma_d. \tag{5.60}$$

With this design scheme, we fix the thresholds at levels 1 to $(M-1)$ and only leave the threshold at the last level adjustable in our performance evaluation.

Now we proceed to study the behavior of P_d. We have

$$P_d = P_r\{\exists\, \hat{\mathbf{i}}_c \in \mathbf{S}_c\} = P_r\left\{\bigcup_{i_1\in\{k_{i_1}\neq 0\}} C_1(i_1)\right\} \tag{5.61}$$

with

$$C_1(i_1) \triangleq \left\{T_0(i_1) \geq h_1, \bigcup_{i_2\in\{k_{i_1,i_2}\neq 0\}} C_2(i_1,i_2)\right\}, \tag{5.62}$$

while for $m = 2,\dots,(M-1)$,

$$\begin{gathered} C_m(i_1,\dots,i_m) \triangleq \left\{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m, \bigcup_{i_{m+1}\in\{k_{i_1,\dots,i_{m+1}}\neq 0\}} C_{m+1}(i_1,\dots,i_{m+1})\right\}, \\ C_M(i_1,\dots,i_M) \triangleq \{T_{i_1,\dots,i_M} \geq h_M\}. \end{gathered} \tag{5.63}$$

It is worth mentioning that, due to the independency of the basis vectors $\mathbf{a}$'s in fingerprint design, all events $C_m(\cdot)$'s at the same level m are independent of each other. Without loss of generality, given a region $\{i_1,\dots,i_m\}$, we assume that $k_{i_1,\dots,i_{m+1}} \neq 0$ for the first $k_{i_1,\dots,i_{m+1}}$ indices of i_{m+1}. Therefore, we have

$$\begin{gathered} P_r\{C_m(i_1,\dots,i_m)\} = P_r\{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m\}, \\ \left[\sum_{j=1}^{k_{i_1,\dots,i_{m+1}}} q_{i_1,\dots,i_m}(j)\prod_{l=1}^{j-1}\left(1-q_{i_1,\dots,i_m}(l)\right)\right] \end{gathered} \tag{5.64}$$

with $q_{i_1,\dots,i_m}(j) = P_r\{C_{m+1}(i_1,\dots,i_m,j)\}$. Iteratively applying this relationship, we can calculate the probability of detection P_d for a given collusion pattern. Intuitively, we can see that $P_r\{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m\}$ plays an important role in P_d, thus the more tightly the colluders are concentrated in a subregion, the higher the P_d is. We want the probability $P_r\{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m\}$ to be larger at lower levels, since a miss detection in a lower level is more severe. Referring to the distribution of $T_{i_1,\dots,i_{m-1}}(i_m)$ in (5.45), we note that $P_r\{T_{i_1,\dots,i_{m-1}}(i_m) \geq h_m\}$ is characterized by the mean $\mu_{i_1,\dots,i_{m-1}}(i_m) = k_{i_1,\dots,i_m}\sqrt{\rho_m}\|\mathbf{s}\|/K$. Further, it is observed that

$$\max\{\mu_{i_1,\dots,i_{m-1}}(i_m)\} \geq \frac{1}{\min\left\{\prod_{j=1}^{m} L_j, K\right\}}\sqrt{\rho_m}\|\mathbf{s}\| = \mu_m^{\text{low}}. \tag{5.65}$$

From this, it is clear that $(1/K)\sqrt{\rho_m}\|\mathbf{s}\|$ is important in system design, since it characterizes the worst case of the detection performance due to higher levels (e.g., $\prod_{j=1}^{m} L_j \geq K$). To simplify our problem, we choose $\rho_1 = \cdots = \rho_{M-1}$ and $L_2 = \cdots = L_{M-1}$. Since the final decision is made in the last level and α_M is usually low (thus h_M is large), we want to maintain enough power at the Mth level to yield reasonable detection probability. In our case, we simply choose $1-\sum_{m=1}^{M-1}\rho_m = 0.5$, meaning half power is kept at the last level. Given the total number of users n, the WNR, and the total levels M, we choose L_1 and h_1 such that $Q((\mu_1^{\text{low}} - h_1)/\sigma_d)$ is close to 1 (e.g., 0.99) and $\alpha_1 < 1/(L_1 c)$. This strategy ensures that at least one colluder-present region will pass the detection at level 1 with very high probability. We choose L_M to maximize the number of colluders that the system can tolerate. For instance, based on the example shown in Figure 5.3, we can choose $L_M = 60$. Therefore, in addition to choosing L_1 and L_M as above, we set other parameters as follows: choose

$$\begin{aligned} \rho_1 &= \cdots = \rho_{M-1} = \frac{0.5}{M-1}, \quad \rho_M = 0.5; \\ L_2 &= L_3 = \cdots = L_{M-1} = \left(\frac{n}{L_1 L_M}\right)^{1/(M-2)}; \\ \alpha_m &= \frac{1}{L_m c}, \quad \text{for } m = 2, \ldots, (M-1). \end{aligned} \tag{5.66}$$

Now the overall performance is a function of α_M (thus h_M) and c.

We demonstrate the performance of such a fingerprinting system through examples and compare it with a fingerprinting system employing orthogonal modulation. As in the group-oriented scheme, the overall power of the colluded observation $\mathbf{y}$ is maintained as $\|\mathbf{s}\|^2$ in our simulations for a fair comparison. We consider a tree structure with four levels, where $L_1 = 8$, $L_2 = L_3 = 5$, and $L_4 = 50$, therefore it can accommodate $n = 10^4$ users. Suppose the total number of colluders $K = 40$. We first examine a scenario where the collusion pattern is balanced, that is, at each level m, all nonzero $k_{i_1,\ldots,i_m}$'s are equal. One example is illustrated in Figure 5.8, where we choose $\alpha_1 = 10^{-3}$ and $c = 10$. In this example, two regions at level 1 include colluders (e.g., $k_1 = k_2 = K/2$), and in turn two subregions at level 2 within each guilty region from level 1 are colluder-present, then one subregion at level 3 within each guilty region of level 2 is colluder-present, and finally 10 colluders are present within each guilty subregion of level 3. This example illustrates the improved collusion resistance that the tree-structured system can provide when compared to orthogonal fingerprinting.

The previous example illustrates the gain in designing fingerprints when one has precise knowledge of the collusion behavior. Sometimes, however, there might be mismatch in the assumed collusion behavior. In order to illustrate the effect of designing a group-fingerprinting system for a collusion pattern that is substantially different from the true collusion pattern, we built fingerprints using the same tree structure as in the example illustrated in Figure 5.8. We then examined two extreme scenarios, where the collusion patterns are more random. Each collusion

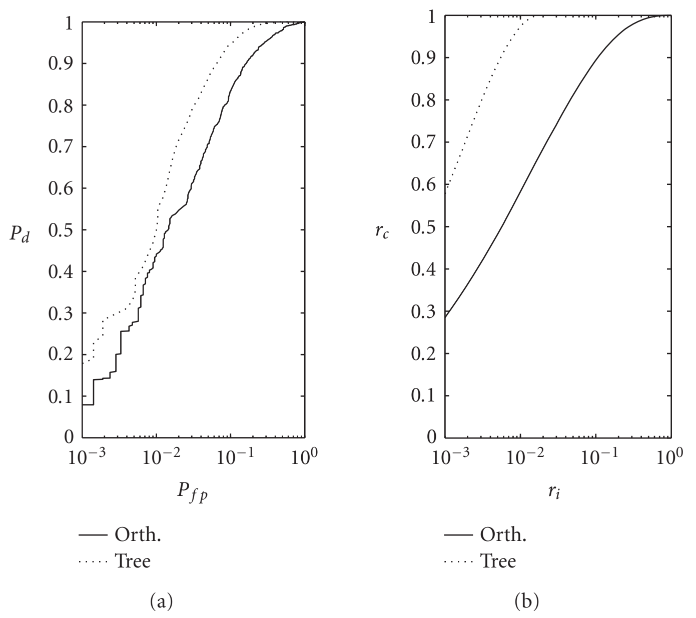


FIGURE 5.8. Performance curves of one example of the tree-structure-based fingerprinting system with a symmetric collusion pattern, compared with the orthogonal scheme in [58]. (a) The ROC curve P_d versus P_{fp}. (b) The curve of the fractions r_c versus r_i.

pattern involved 60 colluders. Random Pattern 1 involves the colluders coming together in a totally random manner, representing that all users are equally likely to collude with each other; while in Random Pattern 2, the colluders are randomly distributed in the first region at level 1. In Figure 5.9, we provide the ROC curves, P_d versus P_{fp}, for both random patterns, as well as for orthogonal fingerprints. From this figure we have two observations. First, when the collusion pattern that the fingerprints were designed for is similar to the actual collusion pattern, as in the case of Random Pattern 2 at the first level, the results show improved collusion resistance. Second, when there is no similarity between the assumed collusion pattern and the true collusion pattern, as in the case of Random Pattern 1, orthogonal fingerprints can yield higher collusion resistance than the tree-based scheme. Therefore, it is desirable for the system designer to have good knowledge of the potential collusion pattern and design the fingerprints accordingly.

One additional advantage of the tree-structure-based fingerprinting system over the orthogonal one is its computational efficiency, which is reflected by the upper bound of the expected computational burden of this approach. The upper bound is in terms of the amount of correlations required as a function of a set of parameters including the number of colluders, the threshold at each level of the tree, and the number of nodes at each level. We denote by $C(n, K)$ the number of correlations needed in our proposed detection scheme. Denoting by $E(A_m)$ the number of expected correlations needed in an event A_m, t being the number of

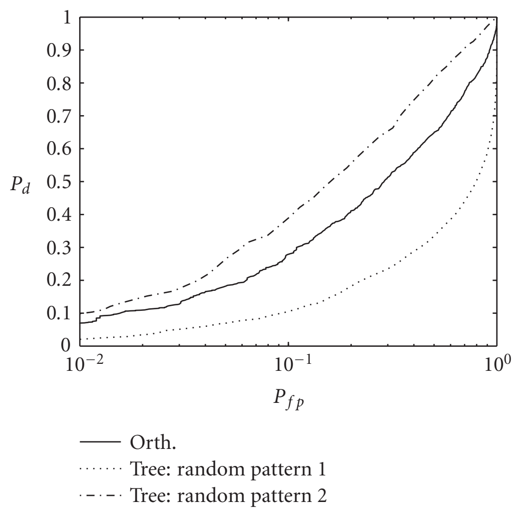


FIGURE 5.9. The ROC curve P_d versus P_{fp} of one example of the tree-structure-based fingerprinting system with random collusion patterns, compared with the orthogonal scheme.

colluder-present subregions at level $(M-1)$, and $C(\text{detection})$ and $C(\text{false alarm})$ being the number of expected correlations needed in correct detections and false alarms, respectively, we have

$$C(n, K) = C(\text{detection}) + C(\text{false alarm}). \tag{5.67}$$

Suppose all the detections for colluder-present subregions are truthful, meaning no miss detection occurs at any stage, then

$$C(\text{detection}) \le L_1 + tL_2 + \cdots + tL_m < t\sum_{m=1}^{M} L_m. \tag{5.68}$$

Recalling that the false alarms can be categorized into event types A_m's and that the number of each type of event A_m is less that tL_m, we have

$$C(\text{false alarm}) \le L_1 E(A_1) + t\sum_{m=2}^{M-1} L_m E(A_m) \tag{5.69}$$

with

$$\begin{aligned} E(A_{M-1}) &= \alpha_{M-1} L_M, \\ E(A_m) &= \alpha_m L_{m+1} E(A_{m+1}) = \cdots \\ &= \alpha_m \left(\prod_{j=m+1}^{M-1} L_j \alpha_j\right) L_M = \alpha_m \frac{1}{c^{M-(m+1)}} L_M, \end{aligned} \tag{5.70}$$

for $m = 1, \dots, (M-2)$, by referring to $\alpha_m = 1/(L_m c)$. Therefore,

$$\begin{aligned} C(\text{false alarm}) &< t \sum_{m=1}^{M-1} L_m \alpha_m \frac{1}{c^{M-(m+1)}} L_M = t \sum_{m=1}^{M-1} \frac{1}{c^{M-m}} L_M \\ &< \min\left\{M, \frac{1}{c-1}\right\} K L_M < \min\left\{M, \frac{1}{c-1}\right\} t \sum_{m=1}^{M} L_m. \end{aligned} \tag{5.71}$$

Putting C(detection) and C(false alarm) together, and assuming $c \geq 2$ usually, we have

$$C(n, K) < \left(1 + \min\left\{M, \frac{1}{c-1}\right\}\right) t \sum_{m=1}^{M} L_m < 2t \sum_{m=1}^{M} L_m < 2K \sum_{m=1}^{M} L_m. \tag{5.72}$$

In addition, the above bound is loose, since it is derived for the worst case where the number of the guilty regions at each level is set as the upper bound t. Clearly, for a small t, a situation we expect when the colluders come from the same groups, the computational cost of the tree-structure-based fingerprinting system is much smaller than the n correlations needed by fingerprinting systems using orthogonal fingerprints.

5.4. Experimental results on images

We now compare the ability of our fingerprinting scheme and a system using orthogonal fingerprints for identifying colluders when deployed in actual images. In order to demonstrate the performance of orthogonal, Gaussian fingerprints we apply an additive spread-spectrum watermarking scheme similar to that in [24], where the original host image is divided into 8×8 blocks, and the watermark (fingerprint) is perceptually weighted and then embedded into the block DCT coefficients. The detection of the fingerprint is nonblind, and is performed with the knowledge of the host image. To generally represent the performance, the 256×256 Lena and Baboon images were chosen as the host images since they have different characteristics. The fingerprinted images have an average PSNR of 44.6 dB for Lena and 41.9 dB for Baboon. We compare the performance of the thresholding detector under average collusion attack. We show in Figure 5.10 the original host images, the colluded images, and the difference images. With $K = 40$, we obtain an average PSNR of 47.8 dB for Lena and 48.0 dB for Baboon after collusion attack and the JPEG compression.

Denoting $\mathbf{s}_j$ as the ideal Gaussian fingerprint, the ith component of the fingerprint, indexed by $\mathbf{i}_c$, is actually embedded as

$$s_{\mathbf{i}_c}(i)^t = \alpha(i) s_{\mathbf{i}_c}(i) \tag{5.73}$$

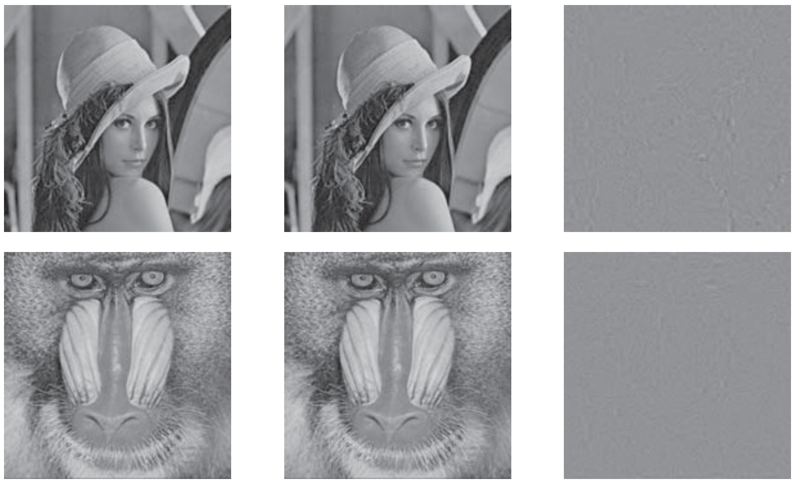


FIGURE 5.10. The host images (a), colluded images (b) with $K = 40$, and difference images (c) for Lena and Baboon under the average attack. The collusion pattern is the same as in Figure 5.11 for Lena image, and as in Figure 5.12 for the Baboon image.

with α being determined by the human visual model parameters in order to achieve imperceptibility. Therefore, the composite embedded fingerprint $\mathbf{y}^t$ after attacking is represented as

$$y(i)^t = \frac{1}{K}\alpha(i) \sum_{\mathbf{i}_c \in \mathbf{S}_c} s_{\mathbf{i}_c}(i) + x(i) + d(i), \tag{5.74}$$

where the noise d is regarded as i.i.d. $N(0, \sigma_d^2)$. Due to the nonblind assumption, α_i's are known in the detector side and thus the effects of real images can be partially compensated by correlating $(\mathbf{y}^t - \mathbf{x})$ with the α-scaled basis or fingerprints in the test statistics $T(\cdot)$'s defined in earlier sections and adjusting the norm to be $\|\mathbf{s}^t\| = \sqrt{\sum_{i=1}^N \alpha(i)^2}\|\mathbf{s}\|$. For instance, the detection scheme in (5.45) is now defined as

$$T_{i_1,\ldots,i_{m-1}}(i_m) = \frac{(\mathbf{y}^t - \mathbf{x})^T \mathbf{a}_{i_1,\ldots,i_m}{}^t}{\|\mathbf{s}^t\|} \tag{5.75}$$

with each component $\mathbf{a}_{i_1,\ldots,i_m}(i)^t = \alpha(i)\mathbf{a}_{i_1,\ldots,i_m}(i)$.

We illustrate examples where the collusion pattern is symmetric. We consider a four-level tree structure with $L_1 = 8$, $L_2 = L_3 = 5$, and $L_4 = 50$. We present the results for the Lena image in Figure 5.11 based on 10^4 simulations, where $K = 40$ and we choose $\alpha_1 = 10^{-3}$ and $c = 10$. In this example, one region at level 1 is

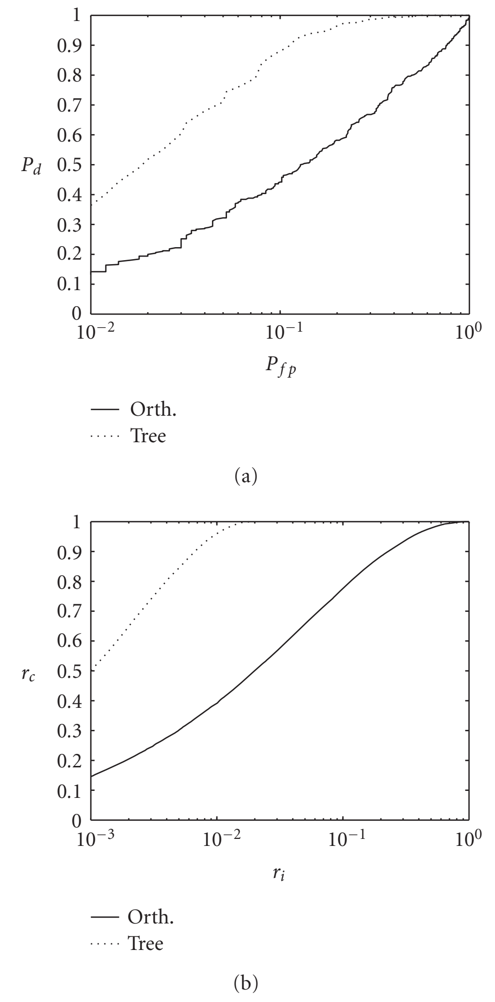


FIGURE 5.11. One example of the detection performance of the group-oriented fingerprinting system on Lena image under average attack. Here $M = 4$, $n = 10^4$, $K = 40$, and the Lena image with equivalent $N = 13691$. (a) The curve P_d versus P_{fp}. (b) The curve r_c versus r_i.

guilty, while at levels 2 and 3 we assumed that each guilty region had 2 subregions containing colluders. Finally, 10 colluders are present within each guilty subregion at the final level, that is, level 3. Additionally, we present the results for Baboon image in Figure 5.12 based on 10^4 simulations, where $K = 40$, $\alpha_1 = 10^{-3}$, and $c = 10$. In this example, two regions at level 1 are guilty, while at levels 2 and 3 we assumed that each guilty region had 2 subregions containing colluders. Finally, 5 colluders are present within each guilty subregion at level 3. We can see that the detection performance of the proposed tree-structure-based fingerprinting system is much better than that of the orthogonal system under this colluder scenario.

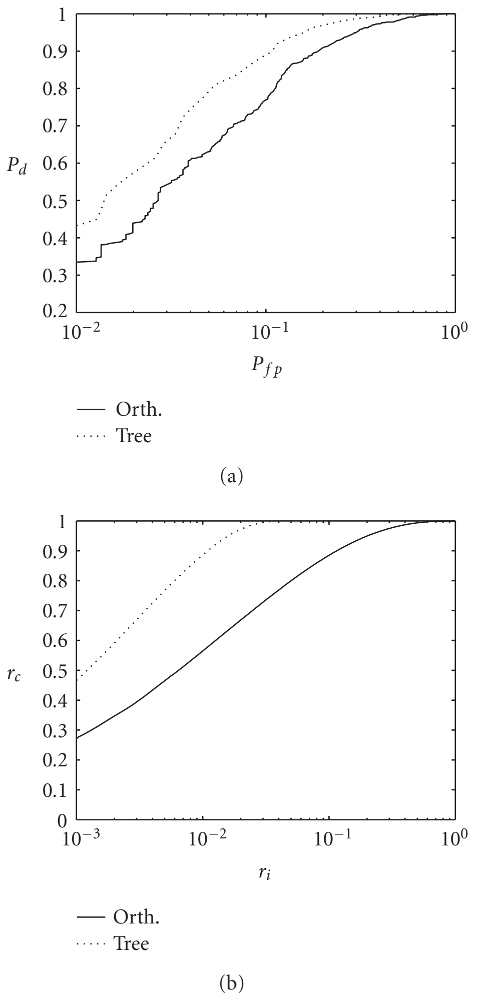


FIGURE 5.12. One example of the detection performance of the group-oriented fingerprinting system on Baboon image under average attack. Here $M = 4$, $n = 10^4$, $K = 40$, and the Baboon image with equivalent $N = 19497$. (a) The curve P_d versus P_{fp}. (b) The curve r_c versus r_i.

5.5. Chapter summary

In this chapter, we investigated a method for enhancing the collusion resistance performance of fingerprinting systems using orthogonal modulation. We proposed a group-oriented fingerprinting system by exploiting the fundamental property of the collusion scenario that adversaries are more likely to collude with some users than others due to geographic or social circumstances. With this underlying philosophy, we then introduced a well-controlled amount of correlations into user fingerprints in order to improve colluder identification.

We first developed a two-tier group-oriented fingerprinting system that involved the design of fingerprints and a two-stage detection scheme for identifying colluders. We evaluated the resistance performance of the proposed system under the average attack by examining different sets of performance criteria. It was demonstrated that the proposed fingerprinting scheme is superior to orthogonal fingerprinting system. In particular, as shown in one example, the proposed scheme can identify all colluders when we allow for up to 10 percent of the innocents to be wrongly accused. In stark contrast, a system using orthogonal fingerprints would require the detection system to suspect almost all users as guilty.

Our work was further extended to a more flexible tree-structure-based fingerprinting system in order to represent the natural hierarchical relationships between users due to social and geographic circumstances. We proposed an efficient and simple scheme for fingerprint design and proposed a multistage colluder identification scheme by exploiting the hierarchical nature of the group-oriented system where the basic idea was to successively narrow down the size of the suspicious set. Performance criteria were analyzed to guide the parameter settings during the design process. We demonstrated performance improvement of the proposed scheme over the orthogonal scheme via examples. Furthermore, we derived an upper bound on the expected computational burden of the proposed approach and showed that one additional advantage of the tree-structure-based fingerprinting system is its computational efficiency. We also evaluated the performance on real images and noted that the experimental results match the analysis. Overall, by exploiting knowledge of the dynamics between groups of colluders, our proposed scheme illustrates a promising mechanism for enhancing the collusion resistance performance of a multimedia fingerprinting system.

6 Anticollusion-coded (ACC) fingerprinting

In the previous chapters, we examined a conceptually simple strategy for fingerprinting that uses orthogonal signals as the fingerprints. We saw that the complexity of detection can be a concern for orthogonal fingerprints. Another problem with orthogonal fingerprinting arises when we examine the energy reduction in the fingerprint signals during collusion. Just looking at averaging collusion, it is easy to see that the energy reduction is roughly the same order of magnitude as the amount of colluders. This can be a significant problem for it means that once we have a few colluders, we become unlikely to identify any traitor. Further, another potential drawback with using orthogonal fingerprinting systems stems from the fact that the maximum number of users that can be supported by an orthogonal fingerprinting system is equal to the amount of orthogonal signals—that is, the dimensionality of the fingerprinting system can be a strict limit on the amount of copies of marked media that we distribute. In many commercial scenarios, the limitations imposed by using orthogonal fingerprinting is too restrictive, and it is therefore desirable to look for other fingerprinting strategies that can support a larger customer base, while also being able to resist collusion.

One natural approach to counteract the energy reduction caused by collusion is to introduce correlation between the fingerprints. When colluders combine their fingerprints, positively correlated components of the fingerprints will not experience as significant an energy reduction as would be experienced by orthogonal fingerprints. We have already seen an example of a fingerprinting strategy that uses correlated fingerprints. The group-based fingerprints that were introduced in Chapter 5 can be viewed as a special type of correlated fingerprints, where we employ a priori knowledge of the collusion pattern to guide us in introducing dependencies between fingerprints that assists in identifying collusion involving members of the same group. Further, by using an extra set of orthogonal signals to represent group information and introducing correlation, we were able to build more fingerprints than the amount of basis signals we had.

In this chapter, we will look at a more general approach for introducing dependency among the media fingerprints. We will build our fingerprints using code modulation, which is another modulation technique that is popular in digital communications [45]. The correlation between fingerprints will be introduced by carefully designing the underlying codevectors used to construct the code-modulated signals. In order to build these codevectors as well as their corresponding fingerprints, we should consider the process by which the fingerprints are embedded in the multimedia content, as well as how these fingerprints will be detected and the corresponding effect that collusion will have on the identification process. Throughout this chapter, we will highlight a recurring theme for designing collusion-resistant fingerprints—the design of fingerprints that survive collusion and can allow one to successfully identify traitors is closely dependent on the embedding and detection process. In particular, we will not only discuss the design of codes that have correlation between the codevectors, but also will discuss the design of traitor identification algorithms that facilitate the accurate identification of participants involved in a collusion attack. Our discussion will primarily focus on the case of binary codes, though nonbinary code or real-valued code constructions are also possible, and are a topic that the research community is currently investigating.

Before we jump to the discussion of code-modulated fingerprints, however, we will briefly visit the problem of designing fingerprints for generic, binary data, such as software or compressed files. In particular, we will review the seminal work of Boneh and Shaw, and their construction of c-secure fingerprint codes were built for objects that satisfy an underlying principle known as the marking assumption. Although the marking assumption is valid for some types of digital data, it is not well suited for the multimedia domain. Multimedia data have very different characteristics than generic data, and we have observed that a few fundamental aspects of the marking assumption may not always hold when fingerprinting multimedia data. In particular, fingerprints may be constructed and embedded in multimedia content in a strategic way so as to significantly limit the capability of colluders to even conduct the type of attacks suggested by the marking assumption. For example, different "marks" or fingerprint bits can be embedded in overlapped regions of an image through spread-spectrum techniques, and such "spreading" will make it challenging, if not impossible, for attackers to manipulate each individual mark at will. This confines the effect of a colluders' action to a milder form of collusion from the designer's point of view. Selectively manipulating bits in a fingerprint code is not directly possible, and instead other forms of attacks, such as an averaging collusion attack, must be used by an adversary to attempt to subvert a multimedia fingerprint. This suggests that by jointly considering the encoding, embedding, and detection processes involved with fingerprinting multimedia, we have the potential to substantially enhance the performance of multimedia fingerprinting.

This new cross-layer paradigm is fundamentally different from most existing literature on coded fingerprints, which typically considers coding issues separately from embedding issues by making assumptions, like the marking assumption. Throughout this chapter, we will consider the embedding and detection process as motivation for designing a new family of fingerprint codes, known as anticollusion codes (ACCs).

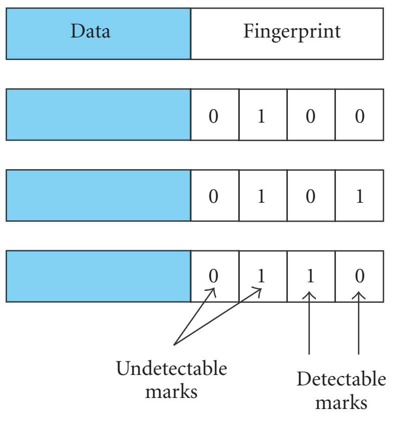


Figure 6.1. Illustration of the marking assumption.

6.1. Prior work on collusion-resistant fingerprinting for generic data

An early work on designing collusion-resistant binary fingerprint codes was presented by Boneh and Shaw in 1995 [77], which primarily considered the problem of fingerprinting generic data that satisfy an underlying principle referred to as the *marking assumption*. In this work, a fingerprint consists of a collection of marks, each of which is modeled as a position in a digital object and can take a finite number of states. A mark is considered *detectable* when a coalition of users does not have the same mark in that position, as illustrated in Figure 6.1. The marking assumption states that undetectable marks cannot be arbitrarily changed without rendering the object useless; however, it is considered possible for the colluding set to change a detectable mark to any state. Under this collusion framework, Boneh and Shaw used hierarchical design and randomization techniques to construct *c-secure codes* that are able to capture one colluder out of a coalition of up to c colluders with high probability.

The construction of a c-secure code involves two main stages: (1) the construction of a base code, and (2) the composition of the base code with an outer code to improve the efficiency when accommodating a large number of users.

In the first stage, we start with a *primitive* binary code that consists of n possible codewords of length $n - 1$. For the mth codeword, the first $(m - 1)$ bits are 0 and the rest are 1. An example of the trivial codes for $n = 4$ users A, B, C, and D is shown in Figure 6.2 (Step-I). If we assign this code to n users, we can see that everyone except user A has a "0" as the first bit, and everyone except user D has "1" as the last bit. Now, suppose that a fingerprint collusion occurs in which the first $m - 1$ users are not involved but the mth user is involved. According to the marking assumption, by inspecting the primitive code, the colluders will not be able to detect the first $m - 1$ bits, hence the first $m - 1$ bits will remain "0" after collusion. On the other hand, the colluders will detect the fact that the mth bits of their fingerprints do not agree. The colluders may then alter this bit to whatever

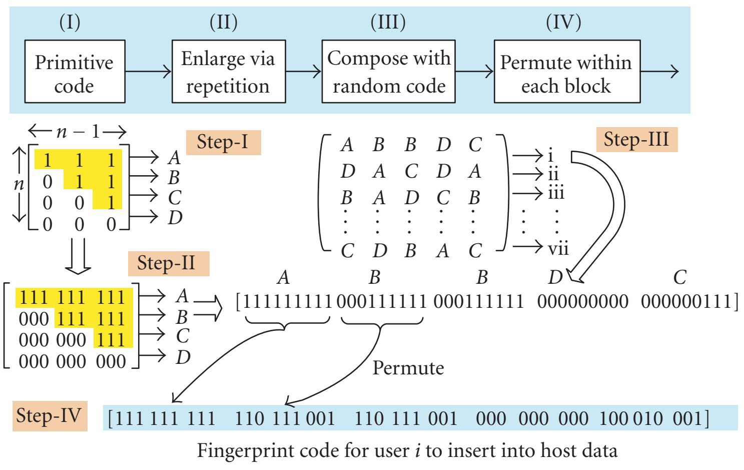


FIGURE 6.2. Construction procedure and examples of collusion-secure fingerprint codes.

they choose—either a 0 or a 1. If the detector observes that the first $m-1$ bits are 0 and the mth bit is a 1, then we can conclude that user m was involved in the collusion. We sequentially check whether this holds for $m = 1, 2, \dots, n$, and if m_0 is the first value for m passing this test, we know with high confidence that user m_0 is involved in collusion.

We note that there is no guarantee that the colluders will switch the bit to a "1," which prompts the need of some method to *encourage* a "1" showing up during collusion. This is accomplished by repetition and permutation techniques. More specifically, for each bit of the primitive code, we form a block by replicating that bit d times, arriving at a code of $(n-1)$ code blocks for a total length of $(n-1)d$. We denote this code as $\Gamma_0(n,d)$. Extending the above example, we have the $\Gamma_0(4,3)$ code shown in Figure 6.2 (Step-II), where $d = 3$. When fingerprinting digital data with a codeword, each bit is put in a location specified by a *secret* permutation table that is known only to the fingerprint creator and detector. Repetition and permutation help hide which position of the digital object encodes which fingerprint bits. In the example in Figure 6.2 (Step-II), the first six bits before permutation for A have the same value, as do C and D. Later, the bit permutation is performed as shown in Figure 6.2 (Step-IV). When colluders having A, C, and D, respectively, come together to collude, they observe six positions with different values among the three of them. But since each of them has the same value at all six positions, they would not know which three out of the six bits correspond to the first three bits before permutation, and which to the second three bits. As a result, they cannot alter the underlying $\Gamma_0(4,3)$ code at will. Based on the principle that every colluder should contribute an equal share to the colluded data, some of the six bits would be set to "1" and others to "0." A detector starts from the first block and examines each block in a block-by-block manner, which is analogous to the bit-by-bit examination of the primitive code discussed above. The number of "1"s per code block is used as an indicator of a user's involvement in collusion.

In the second stage, we use the code obtained in the first stage as a building block and combine it with a second codebook. We construct a second codebook of N codewords over an alphabet of size n, where each codeword has length L. The N codewords are chosen independently and uniformly over the n^L possibilities. We call this code $\mathcal{C}(L, N)$. For example, one random code $\mathcal{C}(5, 7)$ over an alphabet $n = 4$ is shown in Figure 6.2 (Step-III). Next, we substitute each of the n alphabets in the code $\mathcal{C}(L, N)$ by $\Gamma_0(n, d)$, and arrive at a binary code containing N possible codewords of length $L(n - 1)d$. This substitution allows us to first apply the collusion identification algorithm mentioned earlier on each of the L components using the first codebook $\Gamma_0(n, d)$, then find the best match in the second codebook to determine a likely colluder. Finally, each of the blocks of the codeword are permuted before being inserted into the data. For example, using the above code $\mathcal{C}(5, 7)$, we would be able to support 7 users, and the codeword for the first user is shown in Figure 6.2 (Step-IV). By choosing the code parameters appropriately, we can catch one colluder with high probability and keep the probability of falsely accusing innocents low. The construction that Boneh and Shaw arrived at gives a code length of $O(\log^4 N \log^2(1/\epsilon))$ for catching up to $\log N$ users out of a total of N users with error probability $\epsilon < 1/N$.

The construction strategies of Boneh-Shaw code offer insight into fingerprinting both bitstreams and other data for which each bit or unit of a fingerprint is marked in a nonoverlapped manner. An improvement was introduced in [92] to merge the low-level code with the direct sequence spread-spectrum embedding for multimedia and to extend the marking assumption to allow for random jamming. The two-level code construction also inspired the work in [78], which uses the orthogonal fingerprinting in the low level and a structured error-correction code in the upper level to improve the detection efficiency over the traditional single-level orthogonal fingerprinting.

The c-secure fingerprint codes were intended for objects that satisfy the marking assumption. We would like to emphasize that multimedia data have very different characteristics from generic data, and a few fundamental aspects of the marking assumption may not always hold when fingerprinting multimedia data. For example, different "marks" or fingerprint bits can be embedded in overlapped regions of an image through spread-spectrum techniques, and such "spreading" can make it impossible for attackers to manipulate each individual mark at will. As a result, such collusion models as linear collusion by averaging become more feasible for multimedia fingerprints, and this has a critical impact on the design of fingerprint codes. It is also desirable to capture as many colluders as possible, instead of only capturing one. Recent research in [93] explored these directions and jointly considered the encoding, embedding, and detection of fingerprints for multimedia. A new class of structured codes, known as ACCs, has been proposed that is intended to be used with spread-spectrum code modulation. Constructions of specific families of ACC have been devised using combinatorial designs, and several colluder identification algorithms for these fingerprint codes were designed and the performance tradeoffs were examined [80]. We will now shift our attention back to examining multimedia fingerprinting by taking a closer look at the design and detection of these new coded fingerprints in the sections that follow.

6.2. Code modulation with spread-spectrum embedding

Code modulation is a different form of modulation from orthogonal signaling and provides more fingerprint codes for a given amount of basis vectors than orthogonal modulation. With such a compact representation, we will be able to accommodate more users than orthogonal modulation, while using the same amount of orthogonal signals. Throughout the rest of this chapter, we will use this modulation technique, in conjunction with appropriately designed codewords, known as *anticollusion codes,* to construct a family of watermarks that have the ability to identify members of the colluding set of users.

In code modulation, there are v orthogonal basis signals $\{\mathbf{u}_j\}$, and information is encoded into a watermark signal $\mathbf{w}_j$ via

$$\mathbf{w}_j = \sum_{i=1}^{v} b_{ij}\mathbf{u}_i, \tag{6.1}$$

where $b_{ij} \in \{0, 1\}$ or $b_{ij} \in \{\pm 1\}$. The first of the two possibilities for choosing the values of b_{ij} corresponds to on-off keying (OOK) while the second choice of $\{\pm 1\}$ corresponds to an antipodal form [45]. If $b_{ij} = 0$, this is equivalent to having no contribution in the $\mathbf{u}_i$ direction. At the detector side, the determination of each b_{ij} is typically done by correlating with the $\mathbf{u}_i$, and comparing against a decision threshold.

We assign a different bit sequence $\{b_{ij}\}$ for each user j. We may view the assignment of the bits b_{ij} for different watermarks in a matrix $\mathbf{B} = (b_{ij})$, which we call the *derived* code matrix, where each column of $\mathbf{B}$ contains a *derived* codevector for a different user. This viewpoint allows us to capture the orthogonal and code modulation cases for watermarking. For example, the identity matrix describes the orthogonal signaling case since the jth user is only associated with one signal vector $\mathbf{u}_j$.

We refer to $\mathbf{B}$ as the derived code matrix since the design procedure that we will employ in the following section involves designing a code matrix $\mathbf{C}$ whose elements are either 0 or 1 that satisfies specific properties that are best examined using 0 or 1. Once we have designed a code matrix $\mathbf{C}$, we may map $\mathbf{C}$ to the derived code matrix $\mathbf{B}$ by applying a suitable mapping that depends on whether the OOK or antipodal form of code modulation is used. The columns of $\mathbf{B}$ will be used to create the watermark or fingerprint signals $\mathbf{w}_j$ via (6.1). These fingerprint signals $\mathbf{w}_j$ will then be suitably scaled in order to embed them in the host signal with a desired watermark-to-noise ratio. The resulting scaled fingerprint signals will be denoted as $\mathbf{s}_j$.

Before delving into the construction of these codes, we briefly explore the effect that collusion will have upon the fingerprint signals. In binary code modulation, if we average two watermarks $\mathbf{w}_1$ and $\mathbf{w}_2$ corresponding to bit sequences

b_{i1} and b_{i2}, then when $b_{i1} \neq b_{i2}$, the contributions attenuate or cancel depending on whether the OOK or antipodal form is used. However, when $b_{i1} = b_{i2}$, the contributions do not attenuate. For example, suppose that we use antipodal code modulation for constructing the fingerprints and desire that each fingerprint has energy $\mathcal{E}$ in order to be embedded at a target WNR. In this case, each component of the scaled fingerprint signal will have an amplitude of $\sqrt{\mathcal{E}/v}$. The result of averaging two watermark signals is that many of the components will still have $\sqrt{\mathcal{E}/v}$ amplitude, which is identical to the amplitude prior to collusion, while other components will have 0 amplitude. When we average K watermarks, those components in the bit sequences that are all the same will not experience any cancellation, and their amplitude will remain $\sqrt{\mathcal{E}/v}$, while others will experience diminishing (though not necessarily complete cancellation). This observation gives us insight into how we should strive to build our fingerprint codes. Namely, when building fingerprint codes, we should ensure that there is overlap between any subset of user codevectors so that the corresponding components of the fingerprints will survive collusion.

6.3. Combinatorial designs

The construction of anticollusion codes that we present in this chapter will be based upon combinatorial designs. Combinatorial design theory is an area of combinatorics that is devoted to studying the problem of selecting subsets of objects from a larger set of objects such that certain relationships between these subsets are satisfied. Typically, the type of relationship that we are concerned with is incidence, that is, we are concerned with whether these subsets have particular intersection properties and certain membership properties. This is a very generic definition, and there are many types of designs that arise when we consider slightly different definitions of incidence. In spite of this rather broad definition, the theory of combinatorial designs is a field of mathematics that has found application to a variety of applied fields, ranging from the construction of error-correcting codes to the design of statistical experiments. In order to better facilitate the discussion on anticollusion codes built using combinatorial designs, we present a brief survey of some of the core results from design theory. For more detail, though, we refer the reader to one of the many excellent books on design theory [94, 95].

The best way to start describing designs is to look at a simple example.

Example. We consider a set X containing seven objects:

$$X = \{1, 2, 3, 4, 5, 6, 7\}. \tag{6.2}$$

Now, if we choose a subset of three of them at a time, there are a total of $\binom{7}{3} = 21$ different ways to select these subsets. So far, we have not required any incidence relationships. Now, suppose that we impose a constraint that any pair of objects must appear only once. For instance, if the combination of $(1, 2, 3)$ has been selected, then we do not allow $(1, 2, 4)$ to be selected as the pair $(1, 2)$ has already been used. The triplet $(3, 4, 5)$, however, is still possible since it does not contain any pair of objects already appearing in $(1,2,3)$. With this constraint, the number of different valid combinations is reduced to seven. The selection is not unique, though. One possible set of combinations is

$$\{123, 145, 246, 167, 347, 257, 356\}; \tag{6.3}$$

and another possible set of combinations is

$$\{124, 136, 157, 235, 267, 347, 456\}. \tag{6.4}$$

Either of them satisfies the rule laid out above, so we consider the second one and denote it by A. What we have obtained is known as a balanced incomplete block design (BIBD), and is typically referred to as a $(7,3,1)$-BIBD. Here, the 7 refers to the total number of objects, while the 3 refers to the number of objects that we are allowed to choose at a time, and 1 indicates that each pair of objects is allowed to appear once.

We note that in the example above, we represented these objects using numbers, though they could have been anything we wanted. Nowhere in our discussion did we use the numerical properties of these objects. The numbers were just names that allowed us to easily refer to each object.

Now that we have given a specific example of a balanced incomplete block design, we look at the generic definition of a BIBD. Formally, a (v,k,λ)-BIBD is defined as follows.

Definition 6.1. A (v,k,λ) balanced incomplete block design (BIBD) is a pair $(\mathcal{X},\mathcal{A})$, where $\mathcal{A}$ is a collection of k-element subsets (blocks) of a v-element set $\mathcal{X}$, such that each pair of elements of $\mathcal{X}$ occurs together in exactly λ blocks.

There are several natural questions that should come to mind. First, if we have selected subsets of objects to form a BIBD, then how many blocks will an object belong to? Second, how many blocks will we have? We now look at both of these questions.

It turns out that these two questions are related. Each object will belong to the same amount of blocks as any other object. Therefore, we define r to be the amount of blocks that an arbitrary object belongs to. At the same time, we define n to be the amount of blocks that we will form for a (v,k,λ)-BIBD.

We may now obtain a relationship between v, k, r, and n by doing some simple counting. First, if we count repetitions, since each of the v objects belongs to r blocks, we have a total of vr objects. On the other hand, we may count this another way by observing that each of the n blocks has k objects in it, giving a total of nk objects. These two are equal, giving us the following relationship:

$$vr = nk. \tag{6.5}$$

Now, we may perform another counting to obtain another relationship between

r, k, v, and λ. We consider a particular object y and look at blocks B that contain that object and another object. We will define pairs (x, B) to correspond to blocks B containing an object $x \neq y$. Now, there are $v - 1$ ways, we can select an $x \in \mathcal{X}$ such that $x \neq y$. Each of these x appears in λ blocks, giving a total of $\lambda(v-1)$ pairs (x, B). Now, we may also calculate this another way by starting with the block B. There are r ways we may choose a block B with y contained in B. For this block, there are $k - 1$ ways to choose x different from y, giving a total of $r(k - 1)$ pairs (x, B). We may set these equal to get the relationship

$$\lambda(v - 1) = r(k - 1). \tag{6.6}$$

With these two relationships, we can solve for r to get $r = \lambda(v - 1)/(k - 1)$, and $n = \lambda(v^2 - v)/(k^2 - k)$.

These relationships give us some important information about whether or not it is possible for a particular (v, k, λ)-BIBD to exist. Since the numbers r and n must be integers, we may use the above formulas for r and n to show that certain (v, k, λ)-BIBDs do not exist by simply checking to see whether the formulas give integer values for r or n. It should be noted, however, that just because the formulas for r and n for a particular (v, k, λ) give integer values, this does not imply that a BIBD will exist. One of the challenging tasks that has yet to be solved in the field of combinatorial designs is devising a systematic way for determining precisely when a (v, k, λ)-BIBD exists. That being said, though, there are many constructions for generating infinite families of BIBDs with different (v, k, λ) values.

Balanced incomplete block designs may also be described using their incidence matrix. The incidence matrix of a (v, k, λ)-BIBD is a $v \times n$ matrix $\mathbf{A}$, where the rows index the objects of the set $\mathcal{X}$ and the columns index the blocks associated with the BIBD. The (i, j)th element of the matrix $\mathbf{A}$ is defined to be 1 if the ith object belongs in the jth block, otherwise the value is set to 0. We now give the incidence matrix for the $(7, 3, 1)$-BIBD provided in the earlier example:

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}. \tag{6.7}$$

It is clear from this definition that, since an object appears in exactly r blocks, the sum of any row of $\mathbf{A}$ should be r. Similarly, the sum of any column of $\mathbf{A}$ should be k since there are precisely k objects in each block.

The incidence matrix has rich structure that yields several interesting properties. In particular, we may look at $\mathbf{AA}^T$. Observe that the entries along the diagonal of $\mathbf{AA}^T$ are precisely r, while the off-diagonal elements are equal to λ. That is,

$$\mathbf{AA}^T = \begin{pmatrix} r & \lambda & \cdots & \lambda \\ \lambda & r & \cdots & \lambda \\ \vdots & \vdots & \ddots & \vdots \\ \lambda & \lambda & \cdots & r \end{pmatrix}. \tag{6.8}$$

Suppose that we define a matrix $\mathbf{I}_v$ to be the $v \times v$ identity matrix, and the matrix $\mathbf{J}_v$ to be the $v \times v$ matrix with all of its entries equal to 1. Then, it may be shown that

$$\mathbf{AA}^T = (r - \lambda)\mathbf{I}_v + \lambda\mathbf{J}_v. \tag{6.9}$$

There is another result from combinatorics that states that any $v \times n$ matrix $\mathbf{A}$ consisting of entries that are either 0 or 1 that satisfy $\mathbf{AA}^T = (r - \lambda)\mathbf{I}_v + \lambda\mathbf{J}_v$ for some r and λ is an incidence matrix for a corresponding (v, k, λ)-BIBD. The proof of this fact may be found in any text on combinatorial designs.

From this relationship for $\mathbf{AA}^T$, we may find that the determinant of $\mathbf{AA}^T$ is $\det(\mathbf{AA}^T) = rk(r - \lambda)^{v-1}$. To see this, one may subtract the first column of $\mathbf{AA}^T$ from the other columns, and then add the sum of all but the first row to the first row. We may calculate the determinant of the resulting matrix by performing cofactor and major expansion.

Using the fact that $k < v$, we have that $r > \lambda$, and hence the determinant $\det(\mathbf{AA}^T) \neq 0$. Thus the $v \times v$ matrix $\mathbf{AA}^T$ has rank v, and we may use the properties of matrices to deduce

$$v = \operatorname{rank}\left(\mathbf{AA}^T\right) \leq \operatorname{rank}(\mathbf{A}) \leq \min(v, n), \tag{6.10}$$

and hence $v \leq n$. This fact, which is known as Fisher's inequality, basically states that we always have at least as many blocks as we have objects in $\mathcal{X}$.

In fact, one common class of BIBDs is precisely that where $v = n$. These BIBDs are called symmetric BIBDs. The $(7, 3, 1)$-BIBD example that we have been discussing is an example of a symmetric BIBD. Later, when we construct practical anticollusion codes using BIBDs, we will want nonsymmetric BIBDs with $n > v$. There are many techniques to generate nonsymmetric BIBDs. For example, one popular method for constructing BIBDs is to use affine geometries.

We wrap up the discussion by examining one method to generate Steiner triple systems. A Steiner triple system is a $(v, 3, 1)$-BIBD. Steiner triple systems are known to only exist if and only if $v \equiv 1$ or $3 \pmod 6$. The Bose construction scheme may be used to generate Steiner triple systems when $v \equiv 3 \pmod 6$. We discuss the Bose construction in order to give the reader a systematic way to generate BIBDs, but recommend that the reader examines a reference on combinatorial designs for larger, more generic constructions of BIBDs.

We start the Bose construction by defining a building block we will need. A quasigroup $(G, \circ)$ is a set of size n with a binary relationship $\circ$ that takes two elements a and b from G and composes them to produce a third $c = a \circ b$. Further, the composition $a \circ b$ is unique in the sense that $a \circ b_1 = a \circ b_2$ implies that $b_1 = b_2$. We may look at the quasigroup as a table with elements listed along the top and side, and the table describing the composition rule. For example, the following is a quasigroup with $n = 5$ elements:

$\circ$	**1**	**2**	**3**	**4**	**5**
1	1	4	2	5	3
2	4	2	5	3	1
3	2	5	3	1	4
4	5	3	1	4	2
5	3	1	4	2	5

If we examine any row or column of the table, we will see that each number appears at most once. This is the uniqueness property that we required in the definition of the quasigroup. This example also has two additional properties that will be useful to us. First, we have that the composition is commutative, that is, $a \circ b = b \circ a$ for any pair a and b. Also, we have that the composition is idempotent, that is, $a \circ a = a$ for any a. A quasigroup $(G, \circ)$ with both of these properties is called a commutative, idempotent quasigroup. There is a close relationship between quasigroups and latin squares, and techniques used to construct latin squares may be used to construct quasigroups. Commutative, idempotent quasigroups of even order do not exist. A simple procedure for generating a commutative, idempotent quasigroup of order n is the following.

(1) Let n be odd, and start with $\{0, 1, \ldots, n-1\}$, the set of integers modulo n.

(2) To build the $n \times n$ composition table, let x and y range from 0 to $n-1$. In the (x, y)th entry of the table, place the value $((n+1)/2)(x+y) \pmod{n}$.

This procedure results quasigroup whose elements are $\{0, 1, \ldots, n-1\}$ and may be easily mapped into a quasigroup G whose elements are $\{1, 2, \ldots, n\}$ by simply adding one to each entry in the table.

The Bose construction produces Steiner triple systems of order $v = 3 \pmod{6}$. Therefore, we represent v by $v = 6n + 3$ for some appropriate value of n. Suppose that we have a commutative, idempotent quasigroup G of order $2n + 1$. We label the elements of G by $G = \{1, 2, \ldots, 2n + 1\}$. We define a new set $S = G \times \{1, 2, 3\}$. This set S has v elements. We will select subsets of S that will form the blocks of a Steiner triple system. These subsets will each have 3 elements in them. Collectively, we will gather these subsets into a new set T. There are two types of subsets that are formed in the Bose construction.

(i) Type 1. For $1 \le j \le 2n+1$, the subset $\{(j,1),(j,2),(j,3)\}$ is a triple and will belong to T.

(ii) Type 2. For $1 \le i < j \le 2n+1$, the three subsets $\{(i,1),(j,1),(i \circ j,2)\}$, $\{(i,2),(j,2),(i \circ j,3)\}$, and $\{(i,3),(j,3),(i \circ j,1)\}$ are triples and belong to T.

Now, to construct the incidence matrix, we simply form a v-dimensional vector for each subset belonging to T, and place a 1 where an element of S appears in that subset, and a 0 for elements that do not appear. Since there are $v = 6n+3$ elements, we may list them as

$$\begin{aligned} S = \{&(1,1),(2,1),\dots,(2n+1,1),\\ &(1,2),(2,2),\dots,(2n+1,2),\\ &(3,1),(3,2),\dots,(2n+1,3)\}. \end{aligned} \tag{6.11}$$

The vector corresponding to the Type 1 set $\{(j,1),(j,2),(j,3)\}$ would be represented as

$$[0,\dots,0,1,0,\dots 0,1,0,\dots,0,1,0,\dots,0]^T, \tag{6.12}$$

where the first 1 appears in the jth position. We may similarly construct vectors for Type 2 triples. Collecting all of these vectors together produces the incidence matrix $\mathbf{A}$.

For more discussion on constructing quasigroups and BIBDs, we refer the reader to one of the many textbooks on combinatorial designs.

6.4. Combinatorial-design-based anticollusion codes

In this section, we return to our discussion of fingerprinting multimedia content. We will design a family of anticollusion codevectors $\{\mathbf{c}_j\}$ whose overlap with each other can identify groups of colluding users. A similar idea was proposed in [96], where projective geometry was used to construct such code sequences. As we will explain in this section, our proposed code construction considers the relation between the code, the embedding process, and the detection process. As a consequence of this, the resulting fingerprints make more efficient usage of the basis vectors than the codes described in [96].

We begin by defining a new family of codes, called anticollusion codes (ACCs). An anticollusion code is a family of codevectors for which the bits shared between codevectors uniquely identify groups of colluding users. ACC codes have the property that the composition of any subset of K or fewer codevectors is unique. This property allows for the identification of up to K colluders. A K-resilient AND anticollusion code (AND-ACC) is such a code where the composition is an elementwise AND operation. We will show in this section that binary-valued AND-ACC can be constructed using BIBDs [93].

6.4.1. Formulation and construction of ACC codes

We want to design codes such that when K or fewer users collude, we can identify the colluders. We prefer shorter codes since for embedded fingerprints, longer codes would distribute the fingerprint energy over more basis vectors, which would lead to a higher error rate in the detection process. In order to identify colluders, we require that there are no repetitions in the different combinations of K or fewer codevectors. We will call codes that satisfy this property ACCs. In the definition that follows, we provide a definition appropriate for this paper involving binary values, but note that the definition can be easily extended to more general sets G.

Definition 6.2. Let $G = \{0, 1\}$. A code $\mathcal{C} = \{\mathbf{c}_1, \dots, \mathbf{c}_n\}$ of vectors belonging to G^v is called a K-resilient AND anticollusion code (AND-ACC) when any subset of K or fewer codevectors combined element-wise under AND is distinct from the element-wise AND of any other subset of K or fewer codevectors.

The general procedure that we will use to construct these codes will be to build them using the binary symbols $\{0, 1\}$. Once we have constructed a binary-valued code such that the codevectors satisfy the requirements for an AND-ACC, we will map these codevectors-*derived* codevectors by a suitable mapping depending on whether we will use the OOK or antipodal form of binary code modulation for embedding the fingerprint in the multimedia. For example, when used in the antipodal form, the binary symbols $\{0, 1\}$ are mapped to $\{-1, 1\}$ via $f(x) = 2x - 1$.

Looking at the above definition, one natural question that might be asked is why we used the AND logical operation. The motivation behind using AND comes from looking ahead at the collusion problem and how collusion affects the detection process. We assume, when a sequence of watermarks is averaged and detection is performed, that the detected binary sequence is the logical AND of the codevectors $\mathbf{c}_j$ used in constructing the watermarks. For example, when the watermarks corresponding to the codevectors (1110) and (1101) are averaged, we assume that the output of the detector is (1100). When we perform 2 or more averages, this assumption might not necessarily hold since the average of many 1's and a few 0's may produce a decision statistic large enough to pass through the detector as a 1. We discuss the behavior of the detector in these situations further in Section 6.5, and detail approaches to improve the validity of the AND assumption.

We now present a simple ACC, namely the n-resilient AND-ACC. Let $\mathcal{C}$ consist of all n-bit binary vectors that have only a single 0 bit. For example, when $n = 4$, $\mathcal{C} = \{1110, 1101, 1011, 0111\}$. It is easy to see that any element-wise logical AND of $K \le n$ of these vectors is unique. This code has cardinality n, and hence can produce at most n differently watermarked media. We refer to this code as the *trivial* AND-ACC for n users.

We should note that the trivial AND-ACC is very inefficient from a coding point of view. In particular, when the codevectors are mapped to fingerprints, we will require as many basis vectors as we have users. In general, it is desirable to shorten the code length to squeeze more users into fewer bits since this would require the use and maintenance of fewer orthogonal basis vectors. To do this, we need to give up some resiliency. We next present a construction of a K-resilient AND-ACC that requires $\mathcal{O}(K\sqrt{n})$ basis vectors for n users. This construction uses BIBDs [94].

Recall, from our earlier discussions, that a (v,k,λ)-BIBD has a total of $n = \lambda(v^2 - v)/(k^2 - k)$ blocks. We again denote the incidence matrix of this (v,k,λ)-BIBD by $\mathbf{M}$. The BIBD-based construction of an AND-ACC involves defining our code matrix $\mathbf{C}$ as the bit complement of $\mathbf{M}$, and assign the codevectors $\mathbf{c}_j$ as the columns of $\mathbf{C}$. We will shortly show that the resulting construction produces a $(k-1)$-resilient AND-ACC. Our codevectors are therefore v-dimensional, and we are able to accommodate $n = \lambda(v^2 - v)/(k^2 - k)$ users with these v basis vectors. Assuming that a BIBD exists, for n users and a given collusion resiliency of $(k-1)$, we will only need $v = \mathcal{O}(\sqrt{n})$ basis vectors.

We now prove the main result regarding the construction of BIBD-based ACC.

Theorem 6.3. *Let $(\mathcal{X},\mathcal{A})$ be a $(v,k,1)$-BIBD, and $\mathbf{M}$ the corresponding incidence matrix. If the codevectors are assigned as the bit complement of the columns of $\mathbf{M}$, then the resulting scheme is a $(k-1)$-resilient AND-ACC.*

Proof. We prove the theorem by working with the blocks A_j of the BIBD. The bitwise complementation of the column vectors corresponds to complementation of the sets $\{A_j\}$. We would like for $\bigcap_{j\in J} A_j^C$ to be distinct over all sets J with cardinality less than or equal to $k-1$. By De Morgan's Law, this corresponds to uniqueness of $\bigcup_{j\in J} A_j$ for all sets J with cardinality less than or equal to $k-1$. Suppose that we have a set of $k-1$ blocks $A_1, A_2, \ldots, A_{k-1}$, we must show that there does not exist another set of blocks whose union produces the same set. There are two cases to consider.

(i) First, assume that there is another set of blocks $\{A_i\}_{i\in I}$ with $\bigcup_{j\in J} A_j = \bigcup_{i\in I} A_i$ such that $I \cap J = \emptyset$ and $|I| \le k-1$. Suppose that we take a block A_{i_0} for $i_0 \in I$. Then A_{i_0} must share at most one element with each A_j, otherwise it would violate the $\lambda = 1$ assumption of the BIBD. Therefore, the cardinality of A_i is at most $k-1$, which contradicts the requirement that each block has k elements. Thus, there does not exist another set of blocks $\{A_i\}_{i\in I}$ with $\bigcup_{j\in J} A_j = \bigcup_{i\in I} A_i$ and $I \cap J = \emptyset$.

(ii) Next, consider $I \cap J \neq \emptyset$. If we choose $i_0 \in I\backslash(I\cap J)$ and look at A_{i_0}, then again we have that A_{i_0} can share at most 1 element with each A_j for $j \in J$, and thus A_{i_0} would have fewer than k elements, contradicting the fact that A_{i_0} belongs to a $(v,k,1)$-BIBD.

Thus, $\bigcup_{j\in J} A_j$ is unique. □

6.4.2. Examples of BIBD-based ACC

We have shown that we can construct AND-ACC using balanced incomplete block designs. We now look at a couple of examples of AND-ACC built using BIBDs.

The first example of an ACC code is built using the $(7,3,1)$-BIBD that we have seen in the last section:

$$A = \{124, 136, 157, 235, 267, 347, 456\}. \tag{6.13}$$

We represent each of the selection as a column vector in the following way. Starting from the first object, if the object is selected, we put a zero, and otherwise we put a one. The selection of $\{124\}$ becomes $[0,0,1,0,1,1,1]^T$. We can similarly obtain the other six column vectors. Putting all column vectors together, we will obtain the following matrix, which is simply the bit complement of the incidence matrix we presented earlier for this $(7,3,1)$-BIBD:

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{6.14}$$

This code requires 7 bits for 7 users and provides 2-resiliency since any two column vectors share a unique pair of bits. If we use the antipodal form of code modulation, each column vector $\mathbf{c}$ of $\mathbf{C}$ will be mapped to $\{\pm 1\}$ by $f(x) = 2x - 1$. The code modulated watermark is then $\mathbf{w} = \sum_{i=1}^{v} f(c(i))\mathbf{u}_i$. Thus, if we calculate the fingerprint signal corresponding to each codevector, we have the following fingerprint signals:

$$\begin{aligned} \mathbf{w}_1 &= -\mathbf{u}_1 - \mathbf{u}_2 + \mathbf{u}_3 - \mathbf{u}_4 + \mathbf{u}_5 + \mathbf{u}_6 + \mathbf{u}_7, \\ \mathbf{w}_2 &= -\mathbf{u}_1 + \mathbf{u}_2 - \mathbf{u}_3 + \mathbf{u}_4 + \mathbf{u}_5 - \mathbf{u}_6 + \mathbf{u}_7, \\ &\vdots \\ \mathbf{w}_7 &= +\mathbf{u}_1 + \mathbf{u}_2 + \mathbf{u}_3 - \mathbf{u}_4 - \mathbf{u}_5 - \mathbf{u}_6 + \mathbf{u}_7. \end{aligned} \tag{6.15}$$

When two fingerprint signals are averaged, the locations where the corresponding AND-ACC agree and have a value of 1 identify the colluding users. For example, $(\mathbf{w}_1 + \mathbf{w}_2)/2$ has coefficient vector $(-1,0,0,0,1,0,1)$. The fact that a 1 occurs in the fifth and seventh locations uniquely identifies user 1 and user 2 as the colluders.

In the second example of an AND-ACC, we present a larger code that is capable of identifying up to three colluders, and is built using a $(16,4,1)$-BIBD. The code matrix **C** is given by

$$\mathbf{C} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1
\end{pmatrix}. \tag{6.16}$$

The 20 corresponding codevectors may be used in antipodal code modulation to form 20 fingerprint signals. We have depicted the embedding of several of these fingerprints in the Lena image in Figure 6.3. This AND-ACC is an example of a 3-resilient AND-ACC, and we are thus capable of identifying up to 3 colluders. We have also presented the effect that an averaging collusion attack would have on the different components of the codevectors. Again, the set of positions of the sustained 1's is unique with respect to the colluder set, and we may therefore use these to identify colluders. For example, only users 1 and 4 can produce a set of sustained 1's at the fifth–tenth and fourteenth–sixteenth code bits; and only users 1, 4, and 8 can produce a set of sustained 1's at the fifth, sixth, eighth, tenth, fourteenth, and sixteenth code bits. Also, in this example, it should be noted that when we have three users collude, we have some components that are $\pm 1/3$. These are values where we either had two 1-values and a 0-value, or two 0-values and a 1-value in the codevectors averaging together. The AND-ACC codes that we have constructed do not use these locations to determine which users participated in the collusion process—it is only the locations where there are 1's in all colluding codevectors that identify traitors.

6.4.3. ACC coding efficiency and BIBD design methods

We have looked at some examples of ACC built using BIBDs. In order to make AND-ACC useful, we need systematic methods for constructing infinite families of BIBDs with desirable n, v, and k values.

User 1:	-1	-1	-1	-1	1	1	1	1	1	1	1	1	1	1	1	1
User 4:	-1	1	1	1	1	1	1	1	1	1	-1	-1	-1	1	1	1
User 8:	1	-1	1	1	1	1	-1	1	-1	1	1	1	1	1	-1	1
User(1,4) average:	-1	0	0	0	1	1	1	1	1	1	0	0	0	1	1	1
User(1,4,8) average:	$-\frac{1}{3}$	$-\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1	1	$\frac{1}{3}$	1	$\frac{1}{3}$	1	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	1	$\frac{1}{3}$	1
After thersholding:	0	0	0	0	1	1	0	1	0	1	0	0	0	1	0	1

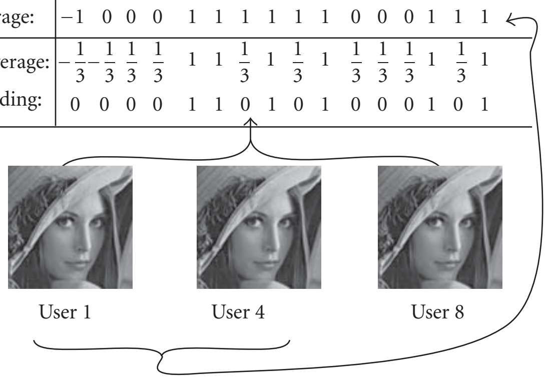


Figure 6.3. 16-bit codevectors from a $(16, 4, 1)$-ACC code for users 1, 4, and 8, and the fingerprinted 512×512 Lena images for these three users, respectively. The code can capture up to 3 colluders. Shown here is an example of two-user collusion by averaging (users 1 and 4) and an example of three-user collusion by averaging. The two codes indicated by arrows in the table uniquely identify the participating colluders.

For example, the $(7, 3, 1)$-example that we presented had no improvement in bit efficiency over the trivial AND-ACC for 7 users, and it had less collusion resilience. A useful metric for evaluating the efficiency of an AND-ACC for a given collusion resistance is $\beta = n/v$, which describes the amount of users that can be accommodated per basis vector. AND-ACCs with a higher β are better. For (v, k, λ)-BIBD AND-ACC, their efficiency is $\beta = \lambda(v-1)/(k^2-k)$. Therefore, the efficiency of an AND-ACC built from BIBDs improves as the code length v becomes larger. By Fisher's inequality [94], we also know that $n \geq v$ for a (v, k, λ)-BIBD, and thus $\beta \geq 1$ using the BIBD construction.

In contrast, the K-resilient construction in [96] has efficiency much less than 1, and thus requires more spreading sequences (or basis vectors) to accommodate the same amount of users as ACC. It is possible to use the collusion-secure code constructions of [77] in conjunction with code modulation for embedding. However, the construction described in [77] is limited to a collusion resistance of $K \leq \log n$, and is designed to trace one colluder among K colluders. Their construction has code length $\mathcal{O}(\log^4 n \log^2(1/\epsilon))$, where $\epsilon < 1/n$ is the decision error probability. This code length is considerably large for small error probabilities and practical n values. For example, when $n = 2^{10}$, the code length of [77] is on the order of 10^6, while the code length for our proposed AND-ACC is on the order of 10^2. Additionally, for the same amount of users, the use of code modulation watermarking with an AND-ACC constructed using a $(v, k, 1)$-BIBD requires less spreading sequences than orthogonal modulation. A code modulation scheme would need v orthogonal sequences for $n = (v^2 - v)/(k^2 - k)$ users, while orthogonal signaling would require n sequences.

This quadratic relationship between n and v is desirable, and luckily there are many techniques to generate BIBDs that we may use to construct efficient AND-ACC. For example, $(v, 3, 1)$-systems, which are also known as Steiner triple systems, are known to exist if and only if $v \equiv 1$ or $3 \pmod 6$. The Bose construction, which we presented earlier, builds Steiner triple systems when $v \equiv 3 \pmod 6$. An alternative construction, known as the Skolem construction, allows us to build Steiner triple systems when $v \equiv 1 \pmod 6$ [97]. Steiner triple systems build ACC with 2-resilience, and can support up to $n = (v^2 - v)/6$ users.

Another approach to constructing BIBDs is to use d-dimensional projective and affine geometry over Z_p, where p is of prime power. Projective and affine geometries yield $((p^{d+1} - 1)/(p - 1), p + 1, 1)$ and $(p^d, p, 1)$-BIBDs [94, 98]. Techniques for constructing these and other BIBDs can be found in [95]. Another construction are unitals, which yield $(p^3 + 1, p + 1, 1)$ BIBDs. When the projective and affine geometries have $d > 2$, that is they are nonplanar geometries, the constructions yield nonsymmetric BIBDs with $n > v$. Additionally, the unitals also have desirable coding efficiency.

Finally, we mention that other combinatorial objects, such as packing designs and pairwise balanced designs, have very similar properties to BIBD, and may be used to construct AND-ACC where the codevectors do not all have the same weight. The construction and use of AND-ACC built from other combinatorial objects is a natural extension of the techniques presented in this book and we recommend interested readers to refer to the references listed above for further discussion on other combinatorial objects.

6.5. Detection strategies and performance tradeoffs

Constructing fingerprints is only half of the battle in battling illicit content manipulation and redistribution. It is also essential to devise tools that will allow content distributors to effectively identify participants involved in creating fraudulent content. Therefore, in this section, we switch our focus to discuss the problem of detecting the colluders when AND-ACCs are used with code modulation to construct media fingerprints. We present several detection algorithms that can be used to identify possible colluders. Our goal here is to present efficient and powerful colluder tracing algorithms that take advantage of the special characteristics of ACC. In the following section, we will use these algorithms to demonstrate the performance of AND-ACC in combating collusion.

In the discussion that follows, we will use the notation that was introduced in Chapter 2. In particular, we will assume that an observed content signal $\mathbf{y}$ may be viewed as consisting of two components: the contributions from the fingerprints, and a total distortion vector $\mathbf{d}$. We assume that this total distortion $\mathbf{d}$ is an N-dimensional vector following an i.i.d. Gaussian distribution with zero mean and variance σ_d^2. This total distortion vector may involve the host signal $\mathbf{x}$ or may involve any noise contributed by compression $\mathbf{d}$. For example, if we consider a blind detection scenario, the total distortion is $\mathbf{d} = \mathbf{x} + \mathbf{z}$, and if there are no colluders present (meaning the observed content does not contain any of the fingerprint signals $\mathbf{s}_j$), then the observed content $\mathbf{y}$ is merely the distortion signal $\mathbf{d} = \mathbf{x} + \mathbf{z}$. We will define two hypotheses for our discussion. H_0 is the colluder-absent hypothesis, while H_1 is the colluder-present hypothesis. Under the colluder-present hypothesis, H_1, K colluders come together and perform an averaging attack that produces a colluded version of the content $\mathbf{y}$. We emphasize here that the case in which there is only a single fingerprint contained in $\mathbf{y}$, that is, only one entity is involved in the redistribution of content, is considered an example of the H_1 hypothesis with $K = 1$.

We may thus gather all of these scenarios together and present the representations for $\mathbf{y}$ in a hypotheses-testing framework, where we have

$$\begin{aligned} H_0 &: \mathbf{y} = \mathbf{d}, \\ H_1 &: \mathbf{y} = \frac{1}{K}\sum_{j\in S_c}\mathbf{y}_j + \mathbf{z} = \frac{1}{K}\sum_{j\in S_c}\mathbf{s}_j + \mathbf{d}, \end{aligned} \tag{6.17}$$

where K is the number of colluders, and S_c indicates the colluder subset, which has size K. The marked content $\mathbf{y}_j$ for each user j is given as

$$\mathbf{y}_j = \mathbf{x} + \mathbf{s}_j = \mathbf{x} + \alpha\sum_{i=1}^{v} b_{ij}\mathbf{u}_i, \tag{6.18}$$

where α is used to control the strength of the fingerprint. We note, here, that the fingerprint $\mathbf{s}_j$ is simply a scaled version of the fingerprint signals $\mathbf{w}_j$ that were presented in Section 6.4. Clearly, the precise probability law under H_1 depends on the fingerprint signals of the colluders and, since the collusion behavior represented by K and S_c is unknown, the hypotheses to be tested are composite. Further, due to the discrete nature of this model, the optimal maximum-likelihood (ML) approach usually involves the enumeration of all possible parameter values, and hence the computational cost can be prohibitively high.

Due to the orthogonality of the basis $\{\mathbf{u}_i\}$, for the purpose of detecting colluders, it suffices to consider the correlator vector $\mathbf{T}_N$, with ith component expressed by

$$T_N(i) = \frac{\mathbf{y}^T\mathbf{u}_i}{\sqrt{\sigma_d^2 \cdot \|\mathbf{u}_i\|^2}} \tag{6.19}$$

for $i = 1, \dots, v$. It is straightforward to show that

$$\mathbf{T}_N = \frac{\alpha_1}{K}\mathbf{B}\boldsymbol{\Phi} + \mathbf{n}, \tag{6.20}$$

where the column vector $\boldsymbol{\Phi} \in \{0,1\}^n$ indicates colluders via the location of the components whose value is 1. The parameter $\alpha_1 = \alpha\sqrt{\|\mathbf{u}\|^2/\sigma_d^2}$ depends on the embedded watermark-to-noise ratio, and is assumed known, with $\|\mathbf{u}_i\| = \|\mathbf{u}\|$ for all i; and $\mathbf{n} = [\mathbf{u}_1, \dots, \mathbf{u}_v]^T\mathbf{d}/\sqrt{\sigma_d^2 \cdot \|\mathbf{u}\|^2}$ follows an $N(\mathbf{0}, \mathbf{I}_v)$ distribution. Here $\mathbf{B}$

is the derived code matrix and K is the number of 1's in $\mathbf{\Phi}$. Thus, the model (6.17) can be equivalently presented as

$$\begin{aligned} H_0 &: f(\mathbf{T}_N \mid \mathbf{\Phi} = \mathbf{0}) = N(\mathbf{0}, \mathbf{I}_v), \\ H_1 &: f(\mathbf{T}_N \mid \mathbf{\Phi}) = N\left(\frac{\alpha_1}{K}\mathbf{B\Phi}, \mathbf{I}_v\right), \end{aligned} \tag{6.21}$$

where we refer the reader back to (6.17) and (6.20) to arrive at this result.

Our goal in this section is to efficiently estimate $\mathbf{\Phi}$. We will present three different strategies for estimating the colluder vector $\mathbf{\Phi}$: first, we will look at a hard detection strategy; second, we will look at a soft detection strategy that will effectively avoid hard thresholding when trying to determine code bits; and, finally, a sequential detection algorithm that bypasses estimating code bits and attempts to directly estimate the colluder set.

However, before we examine the candidate detectors, we discuss the choice of using either the OOK or antipodal form of code modulation. Suppose that a codevector $\mathbf{c}_j$ has weight $\omega = wt(\mathbf{c}_j)$. In the OOK case, the remaining $v - \omega$ positions would be zeros, while in the antipodal case, the remaining $v - \omega$ positions would be mapped to -1. If we allocate $\mathcal{E}$ energy to this codevector, then the OOK case would use $\mathcal{E}/\omega$ energy to represent each 1, while the antipodal case would use $\mathcal{E}/v$ energy to represent each ± 1. The amplitude separation between the constellation points for the 0 and 1 in OOK is $\sqrt{\mathcal{E}/\omega}$, while the separation between -1 and 1 in antipodal is $2\sqrt{\mathcal{E}/v}$. Therefore, since it is desirable to have the separation between the constellation points as large as possible, we should choose OOK only when $\omega < v/4$. In the AND-ACCs presented in Section 6.4, the weight of each codevector is $\omega = v - k$. OOK is advantageous when $k > (3/4)v$, and antipodal modulation is preferable otherwise. Typically, in BIBDs with $\lambda = 1$, the block size k is much smaller than v [95], and therefore the antipodal form of code modulation is preferred.

6.5.1. Hard detection

We first introduce a simple detection scheme based upon hard thresholding. Upon applying hard thresholding to the detection statistics $T_N(i)$, we obtain a vector $\mathbf{\Gamma} = (\Gamma_1, \Gamma_2, \dots, \Gamma_v)$, where $\Gamma_i = 1$ if $T_N(i) > \tau$ and $\Gamma_i = 0$ otherwise. Given the vector $\mathbf{\Gamma}$, we must determine who the colluders are.

Algorithm 6.1 starts with the entire group as the suspicious set, and uses the components of $\mathbf{\Gamma}$ that are equal to 1 to further narrow the suspicious set. We determine a vector $\mathbf{\Phi} = (\Phi_1, \Phi_2, \dots, \Phi_n)^T \in \{0,1\}^n$ that describes the suspicious set via the location of components of $\mathbf{\Gamma}$ whose value are 1. Thus, if $\Phi_j = 1$, then the jth user is suspected of colluding. In the algorithm, we denote the jth row vector of $\mathbf{C}$ by $\mathbf{e}_j$, and use the fact that the element-wise multiplication "$\cdot$" of the binary vectors corresponds to the logical AND operation. We start with $\mathbf{\Gamma}$ and $\mathbf{\Phi} = \mathbf{1}$, where $\mathbf{1}$ is the n-dimensional vector consisting of all ones. The algorithm then uses the

```
Algorithm: HardDetAlg($\mathbf{\Gamma}$)
$\mathbf{\Phi} = \mathbf{1}$;
Define $J$ to be the set of indices where $\Gamma_i = 1$;
for $t = 1$ to $|J|$ do
    $j = J(t)$;
    Define $\mathbf{e}_j$ to be the $j$th row of $\mathbf{C}$;
    $\mathbf{\Phi} = \mathbf{\Phi} \cdot \mathbf{e}_j$;
end
return $\mathbf{\Phi}$;
```

ALGORITHM 6.1. Algorithm HardDetAlg($\mathbf{\Gamma}$), which determines the vector $\mathbf{\Phi}$ that describes the suspect set.

indices where $\mathbf{\Gamma}$ is equal to 1, and updates $\mathbf{\Phi}$ by performing the AND of $\mathbf{\Phi}$ with the rows of the code matrix $\mathbf{C}$ corresponding to indices where $\mathbf{\Gamma}$ is 1.

6.5.2. Adaptive sorting approach

One drawback of the hard detection approach above is that the threshold τ is fixed at the beginning. This choice of τ is applied to every detection scenario, regardless of the observations. To overcome this disadvantage, it is desirable to avoid the hard-thresholding process. Consequently, in Algorithm 6.2, we present a soft-thresholding detection scheme where $\mathbf{\Phi}$ is updated iteratively via the likelihood of $\mathbf{T}_N$. We start with the highest detection statistic $T_N(j)$ to narrow down the suspicious set. At each iteration, we check whether the next largest statistic $T_N(j)$ increases the likelihood. If the likelihood increases, then we use this to further trim the suspicious set. The iteration stops when the likelihood decreases.

6.5.3. Sequential algorithm

The approaches in both Sections 6.5.1 and 6.5.2 share the same idea that the colluders can be uniquely identified by utilizing the locations of 1's in $\mathbf{\Gamma}$ due to the structural features of our AND-ACC. One key disadvantage of these schemes is that, in practice, the noise causes the thresholding decision to have errors, which in turn results in incorrect indications of colluders. Therefore, it is desirable to estimate $\mathbf{\Phi}$ directly from the pdf behavior of $\mathbf{T}_N$, as suggested by model (6.21).

Thus, we introduce Algorithm 6.3, which we refer to as the sequential algorithm, for estimating $\mathbf{\Phi}$ from the pdf of $\mathbf{T}_N$. This algorithm is similar to the adaptive sorting scheme in its sequential nature. The difference is that Algorithm 6.3 directly estimates the colluder set, while the adaptive sorting algorithm first estimates the code bits before deciding the colluder set.

Finally, we note that since a binary variable is assigned to each user that indicates his/her presence or absence in the coalition, the collusion problem (6.20) is related to the estimation of superimposed signals [99]. One may also apply the alternating maximization (AM) method [100, 101] to the problem of identifying the

```
Algorithm: AdSortAlg($\mathbf{T}_N$)
Sort elements of $\mathbf{T}_N$ in descending order and record
corresponding index vector as $\mathbf{J}$;
Set $\mathbf{\Phi} = \mathbf{1}$; Set $i = 0$ and calculate the likelihood
$LL(i) = f(\mathbf{T}_N|\mathbf{\Phi})$ according to (6.21);
Flag = True;
while Flag & $i < v$ do
    Set $i = i + 1$;
    $j = J(i)$ and $\Gamma(j) = 1$;
    Define $\mathbf{e}_j$ to be the $j$th row of $\mathbf{C}$;
    $\mathbf{\Phi}_{\text{up}} = \mathbf{\Phi} \cdot \mathbf{e}_j$;
    $LL(i) = f(\mathbf{T}_N|\mathbf{\Phi}_{\text{up}})$;
    if $LL(i) > LL(i-1)$ then
        $\mathbf{\Phi} = \mathbf{\Phi}_{\text{up}}$;
    else
        Flag = False;
    end
end
return $\mathbf{\Phi}$;
```

Algorithm 6.2. Algorithm AdSortAlg($\mathbf{\Gamma}$) which uses an adaptive sorting approach to determine the vector $\mathbf{\Phi}$ that describes the suspect set.

colluders. In our experience, we found that there was no significant performance difference between the AM approach and our sequential algorithm, though the computational complexity of the AM algorithm was noticeably higher.

6.6. Experimental results for ACC fingerprinting

Now that we have presented both techniques for creating fingerprints, and techniques for detecting fingerprints, we now turn our attention to evaluating their performance. We will study the performance of fingerprints built using AND-ACC by first conducting a study using synthetic content signals represented by signals constructed from randomly generated Gaussian samples. Then, we will examine the performance of code modulated fingerprints built from our AND-ACC by embedding our fingerprints inside of images using a popular additive spread-spectrum watermarking scheme. For both sets of experiments, we will highlight important observations about the fingerprinting, collusion, and detection process.

6.6.1. ACC simulations with Gaussian signals

In this section, we study the behavior of our AND-ACC when used with code modulation in an abstract model. The distortion signal $\mathbf{d}$ and the orthogonal basis signals $\mathbf{u}_i$ are assumed to be independent and each of them is an $N = 10000$ point vector of i.i.d. Gaussian samples. The factor α is applied equally to all components

```
Algorithm: SeqAlg($\mathbf{T}_N$)
Set $K = 0$;
Calculate the likelihood $LL(0) = f(\mathbf{T}_N | \mathbf{\Phi} = \mathbf{0})$ according to (6.21);
Set $J = \varnothing$ and Flag = True;
while Flag do
    Let $K = K + 1$;
    Estimate $i_K$, assuming that $(K - 1)$ users
    have indices $i_j = J(j)$,
    for $j = 1, \ldots, (K - 1)$ via

        $i_K = \arg\max_{i_K} \{ f(\mathbf{T}_N | \mathbf{J} = \{i_1, \ldots, i_K\}) \}$;

    $J = \{i_1, \ldots, i_K\}$ and $\mathbf{\Phi}_{\text{up}}(J) = 1$;
    Calculate $LL(K) = f(\mathbf{T}_N | \mathbf{\Phi}_{\text{up}})$;
    If $LL(K) > LL(K - 1)$ then
        $\mathbf{\Phi} = \mathbf{\Phi}_{\text{up}}$;
    else
        Flag = False;
    end
end
return $\mathbf{\Phi}$;
```

Algorithm 6.3. Algorithm SeqAlg($\mathbf{T}_N$) which is a sequential algorithm to determine the vector $\mathbf{\Phi}$ that describes the suspect set.

and is used to control the WNR, where WNR $= 10\log_{10} \|\mathbf{s}\|^2/\|\mathbf{d}\|^2$ dB. We use these simulations to verify some basic issues associated with collusion and code modulation.

In the simulations that follow, we used the $(16,4,1)$-BIBD that was presented earlier to construct our AND-ACC code. The $(v,4,1)$-codes are a broad family of AND-ACC that may be constructed from BIBDs since $(v,4,1)$-BIBDs are known to exist if and only if $v \equiv 1$ or $4 \pmod{12}$ and there are systematic methods for generating these BIBDs. With the $(16,4,1)$-code, we use 16 orthogonal basis vectors to handle 20 users, and can uniquely identify up to $K = 3$ colluders. Throughout the experiments that follow, the fingerprints for each user will be assigned according to the antipodal form of code modulation, where we use the columns of $\mathbf{C}$ as the codevectors.

We first wanted to study the behavior of the detector and the legitimacy of the AND logic for the detector under the collusion scenario. We randomly selected 3 users as colluders and averaged their marked content signals to produce $\mathbf{y}$. The colluded content signal was used in calculating T_N, as described in (6.19).

For three colluders using antipodal modulation, there are four possible cases for the average of their bits, namely $-1, -1/3, 1/3$, and 1. We refer to the cases $-1, -1/3$, and $1/3$ as the non-1 hypothesis since under the AND logic

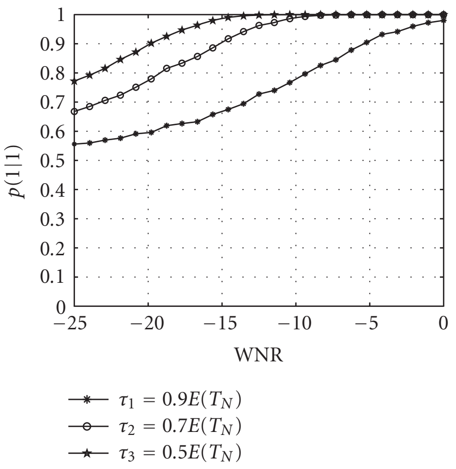


FIGURE 6.4. The probability of detection $p(1|1)$ and for different WNR and different thresholds using hard detection.

assumption of our proposed AND-ACC, they would be mapped to 0. We examined the tradeoff between the probability $p(1|1)$ of correctly detecting a one when a one was expected from the AND logic, and the probability of $p(1|\text{non-1})$, where the detector decides a one when the correct hypothesis was a non-1. We calculated $p(1|1)$ and $p(1|\text{non-1})$ as a function of WNR when using hard detection with different thresholds. The thresholds used were $\tau_1 = 0.9E(T_N)$, $\tau_2 = 0.7E(T_N)$, and $\tau_3 = 0.5E(T_N)$. In order to calculate $E(T_N)$, we used (2.5) and assumed that the detector knows the WNR, and hence the power of the distortion. The plot of $p(1|1)$ for different thresholds is presented in Figure 6.4, and the plot of $p(1|\text{non-1})$ is presented in Figure 6.5. We observe that for the smaller threshold of $0.5E(T_N)$, the probability $p(1|1)$ is higher, but at the expense of a higher probability of false classification $p(1|\text{non-1})$. Increasing the threshold allows us to decrease the probability of falsely classifying a bit as a one, but at the expense of also decreasing the probability of correctly classifying a bit as a one.

We next examined the performance of the different detection strategies for identifying the colluders. The following six measures present different, yet related aspects of the performance for capturing colluders:

(a) the fraction of colluders that are successfully captured;
(b) the fraction of innocent users that are falsely placed under suspicion;
(c) the probability of missing a specific user when that user is guilty;
(d) the probability of falsely accusing a specific user when that user is innocent;
(e) the probability of not capturing any colluders;
(f) and the probability that we falsely accuse at least one user.

We calculated these six different performance measures for each of the detection strategies described in Section 6.5 and present the results in Figure 6.6. For each WNR, we averaged over 2000 experiments.

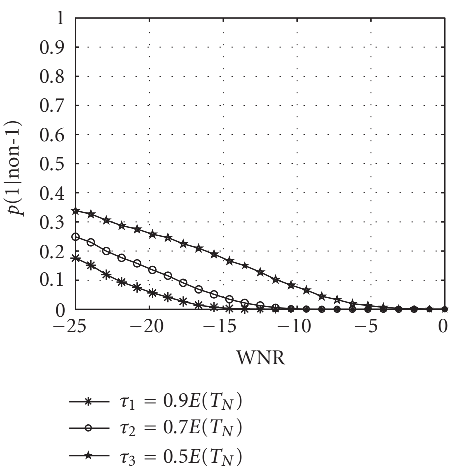


FIGURE 6.5. The probability of false alarm $p(1|\text{non-1})$ for different WNR and different thresholds using hard detection.

We observe in Figure 6.6a and Figure 6.6b that for all WNRs, the use of a higher threshold in the hard detection scheme is able to capture more of the colluders, but also places more innocent users falsely under suspicion. As WNR increases, the hard detector has lower $p(1|\text{non-1})$, and therefore does not incorrectly eliminate colluders from suspicion. Similarly, at higher WNR, the hard detector has a higher $p(1|1)$, thereby correctly identifying more 1's, which allows for us to eliminate more innocents from suspicion. Therefore, at higher WNR, we can capture more colluders as well as place less innocent users under suspicion. We note, however, that in Figure 6.6b, at low WNR between -25 dB and -15 dB, the fraction of innocents under suspicion using threshold $\tau = 0.9E(T_N)$ is lower than at slightly higher WNR. This behavior can be explained by examining Figures 6.4 and 6.5. We observe that at low WNR, the $p(1|\text{non-1})$ is higher than slightly higher WNR, particularly for the threshold $\tau = 0.9E(T_N)$. However, for this threshold, the $p(1|1)$ at these WNR is relatively flat. These two observations combined indicate that at lower WNR, we falsely decide 1 more often than at slightly higher WNR, while we do not experience much difference in the amount of correctly identified 1's. As more 1's pass through the detector, we remove more users from suspicion. Therefore, since the amount of correctly detected 1's increases slowly for WNRs between -25 dB and -15 dB, the additional 1's from false detections at lower WNR eliminate more innocent users (as well as colluders) from suspicion.

Compared to the hard detection scheme with $\tau = 0.9E(T_N)$, the adaptive sorting scheme captures a larger fraction of the colluders at all WNR, while for a large range of WNRs between -20 dB and -3 dB, the adaptive sorting scheme places fewer innocents under suspicion. However, examining the curves for the sequential algorithm, we find that we are able to capture more colluders than any other detection schemes at all WNRs. Further, the amount of innocents placed under suspicion is less than the adaptive sorting algorithm.

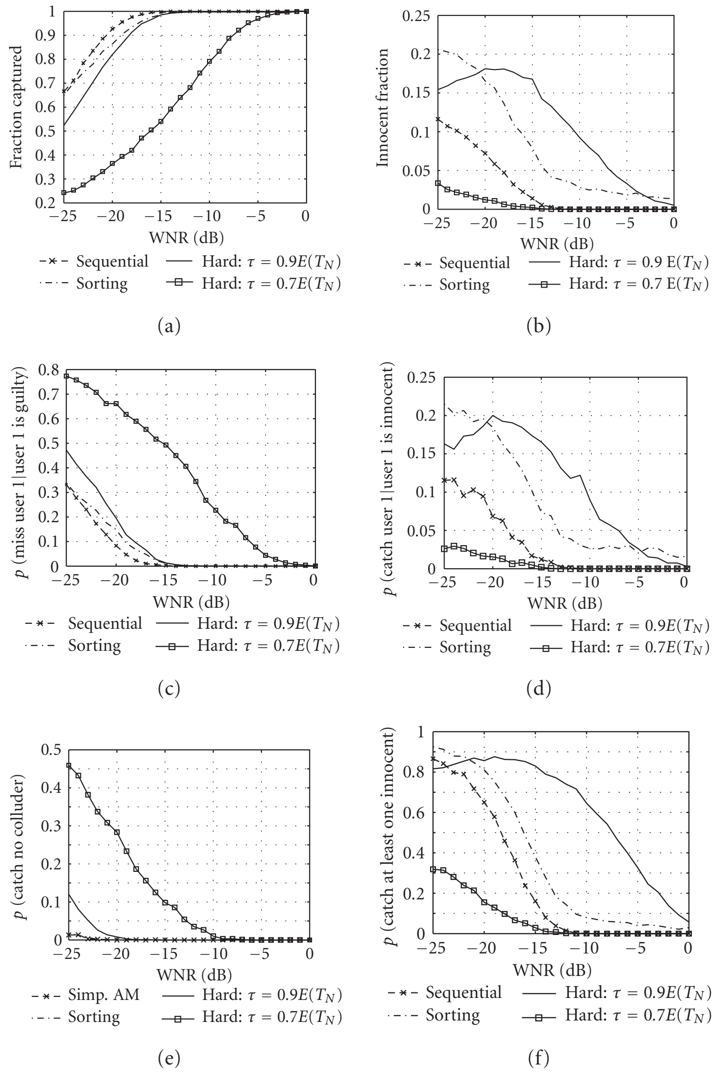


FIGURE 6.6. (a) The fraction of colluders that is successfully captured, or placed under suspicion, (b) the fraction of the total group that is innocent and falsely placed under suspicion for different WNR and different thresholds, (c) the probability of missing user 1 when he is guilty, (d) the probability of falsely accusing user 1 when he is innocent, (e) the probability of not capturing any colluder, and (f) the probability of putting at least one innocent under suspicion. In each plot, there were 3 colluders.

Consistent behavior is observed for the different detection schemes under the other performance measures, as depicted in Figures 6.6c, 6.6d, 6.6e, and 6.6f. Overall, the sequential detection scheme provides the most promising balance between capturing colluders and placing innocents under suspicion.

6.6.2. ACC experiments with images

In order to demonstrate the performance of our AND-ACC with code modulation fingerprinting on real images for fingerprinting users and detecting colluders, we used an additive spread-spectrum watermarking scheme similar to that in [24], where the perceptually weighted watermark was added to 8×8 block DCT coefficients. We focused on detecting the colluders in a blind detection scenario, where there was no knowledge of the host image at the detector, as well as a nonblind detection scenario, where the original image was subtracted off from the compressed and colluded copy. We used the detection statistics that were presented in (2.6). Just as we did for the simulations using Gaussian signals, we used the code matrix, detailed in (6.16), as our AND-ACC. This code is able to accommodate 20 users and is designed to capture up to 3 colluders. The 512×512 Lena and Baboon images were used as the host signals for the fingerprints. The fingerprinted images have no visible artifacts with an average PSNR of 41.2 dB for Lena, and 33.2 dB for Baboon. Figure 6.7 shows the original images, the fingerprinted images, and the difference with respect to the originals.

The three derived codevectors that were assigned to users 1, 4, and 8 via antipodal mapping as well as the colluded versions are presented in Figure 6.3. Two collusion examples are illustrated in Figure 6.8 and the detection statistics of the two examples are shown in Figure 6.9. In one example, we averaged the Lena images fingerprinted with users 1 and 4's codes, and the other is for averaging users 1, 4, and 8's. The colluded images are further compressed using JPEG with a quality factor (QF) of 50%. Also shown in Figure 6.9 are the thresholds determined from the estimated mean of the detection statistics $E(T_N)$. We then estimate the fingerprint codes by thresholding the detection statistics using a hard threshold of τ. The estimated fingerprint codes are identical to the expected ones shown in Figure 6.3. We can see in Figures 6.9 and 6.10 that nonblind detection increases the separation between the values of the detection statistics that are mapped to $\{-1, 0, +1\}$.

We present histograms of the $T_N(i)$ statistics from several collusion cases with different distortions applied to the colluded Lena images in Figure 6.10. For each collusion and/or distortion scenario, we used 10 independent sets of basis vectors to generate the fingerprints. Each set consists of 16 basis vectors for representing 16 ACC code bits. Figure 6.10 shows the histograms of the blind and nonblind detection scenarios, as well as the single user, two colluders, and three colluders cases. We see that there is a clear distinction between the three decision regions corresponding to $\{-1, 0, +1\}$, which is desirable for identifying colluders. This implies that the average magnitude of T_N, when the bit values agree, is much larger than the average magnitude for where the bit values disagree, therefore facilitating the accurate determination of the AND-ACC codes from colluded images. The statistics T_N can be used with hard detection to determine the colluders, as depicted in Figure 6.9. Similarly, we can use T_N with other detectors, whose performance was presented in Section 6.6.1. We have also studied the effect of averaging collusion in the presence of no distortion, JPEG compression, and lowpass filtering. We found

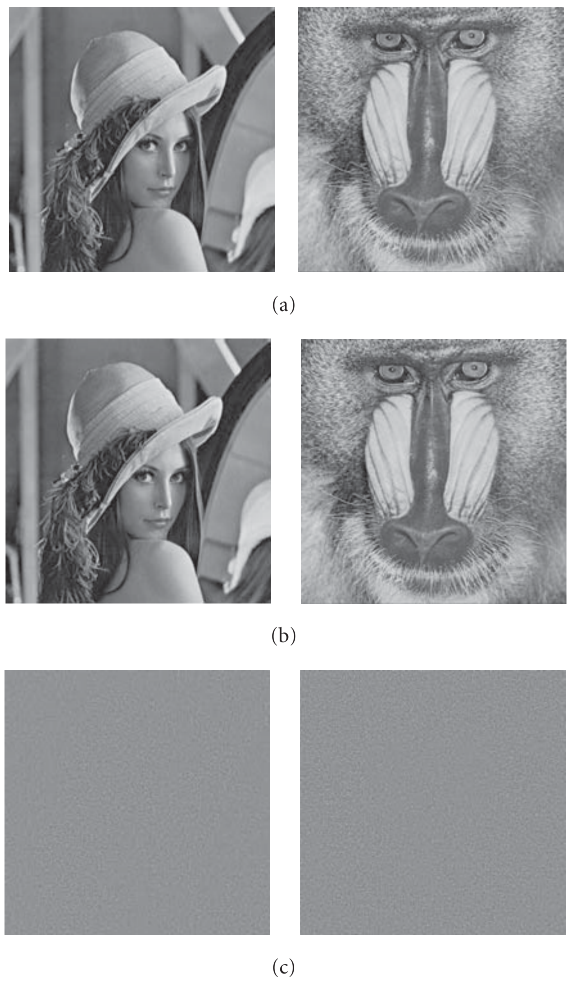


Figure 6.7. The original images (a), fingerprinted images (b), and difference images (c) for Lena and Baboon. In the difference images, gray color indicates zero difference between the original and the fingerprinted version, and brighter and darker indicate larger difference.

that the one and non-one decision regions were well separated, which can lead to reliable identification of colluders.

6.7. A unified formulation on fingerprinting strategies

We now revisit the formulation of fingerprint coding and modulation. We noted in Section 6.2 that the general form for code modulation allows us to also capture the case of orthogonal modulation by simply considering the identity matrix as the derived code matrix. The representation for fingerprints presented in (6.1) allows us to arrive at a unified framework that covers a broad spectrum of fingerprint

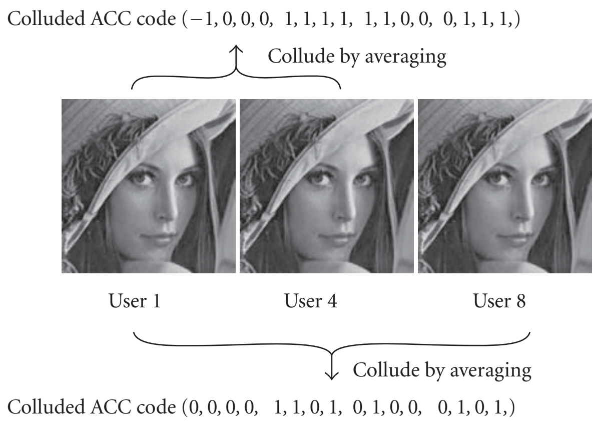

Colluded ACC code (−1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1,)
Collude by averaging
User 1
User 4
User 8
Collude by averaging
Colluded ACC code (0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1,)

FIGURE 6.8. Illustration of collusion by averaging two and three images fingerprinted with ACC codes, respectively.

designs, ranging from orthogonal fingerprints to group-oriented fingerprints to ACC-based fingerprints. Additionally, this unified framework provides a simplified framework for formulating the colluder identification problem.

Under this unified formulation, a different sequence $\{b_{1j}, b_{2j}, \ldots, b_{vj}\}$ is assigned for each user j. The matrix representation, $\mathbf{B} = \{b_{ij}\}$, will have different structure for different fingerprint strategies. Generally, we choose some noise-like signals $\mathbf{u}_i$ that will serve as basis signals for building our fingerprint signals. These basis signals span the watermark space, allowing us to represent each fingerprint signal via the matrix-vector form

$$\mathbf{w}_j = \sum_{i=1}^{v} b_{ij}\mathbf{u}_i = \mathbf{B}\mathbf{b}_j, \tag{6.22}$$

where each column of $\mathbf{U}$ is an orthogonal basis vector, and $\mathbf{b}_j = [b_{1j}, b_{2j}, \ldots, b_{vj}]^T$ is the jth column of the derived code matrix $\mathbf{B}$.

An identity matrix for $\mathbf{B}$ represents orthogonal fingerprinting $\mathbf{w}_j = \mathbf{u}_j$, where each user is identified with an orthogonal basis signal. The simple structure for encoding and embedding orthogonal fingerprints makes it attractive in identification applications that involve a small group of users. As noted earlier, when we have a large group of users, however, the linearly increasing number of basis signals has a significant impact on the complexity of detection and bookkeeping, and the energy reduction of the fingerprint signal under averaging collusion is high.

To use v orthogonal basis signals to represent more than v users, correlations between different users' fingerprints must be introduced. One way to construct a corresponding $\mathbf{B}$ matrix is to use binary codes. The c-secure code and the BIBD-ACC code discussed earlier in this chapter are two examples. In more general constructions, entries of $\mathbf{B}$ can be real numbers, as was described in the group-oriented fingerprinting strategies [102], for example. More recently, new families of anticollusion codes have been proposed that are constructed using the

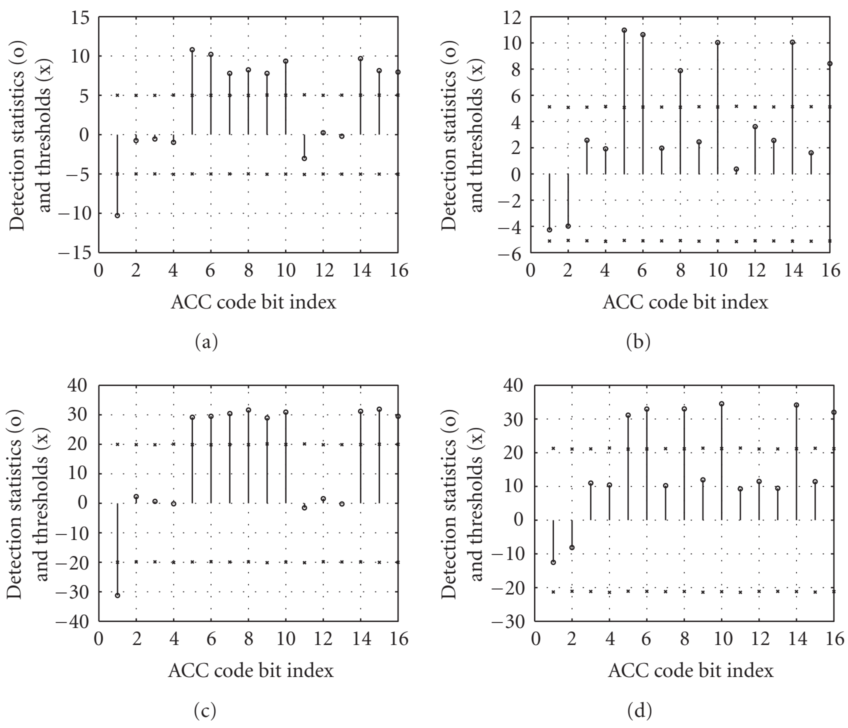


FIGURE 6.9. Example detection statistics values for 2 users' and 3 users' collusion with a (16, 4, 1)-BIBD AND-ACC fingerprint. (Top) Blind detection scenario on colluded Lena image, two and three colluders and (bottom) nonblind detection scenario on colluded Lena image. (Left) Users 1 and 4 perform averaging, resulting in the output of the detector as $(-1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1)$. (Right) Users 1, 4, and 8 average, resulting in the output of the detector as $(0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1)$.

relationships between the problem of collusion in multimedia fingerprinting and code-division multiple access (CDMA) in multiuser communication [103]. The key issue in designing a good **B** is to strategically introduce correlation among different fingerprints to allow for accurate identification of any single fingerprint as well as the contributing fingerprints involved in forming a colluded fingerprint signal.

During the collusion problem, several colluders linearly combine their copies attempting to remove the underlying fingerprints. Typically, we assume that no user wants to take higher risk than any other, and hence assume that users are simply averaging when conducting an attack. If we assume that an averaging collusion attack is being used, and also assuming there are a total of K colluders represented by the set S_c, then the colluded signal is simply

$$\mathbf{y} = \mathbf{x} + \frac{\alpha}{K} \sum_{j \in S_c} \mathbf{w}_j + \mathbf{d}, \tag{6.23}$$

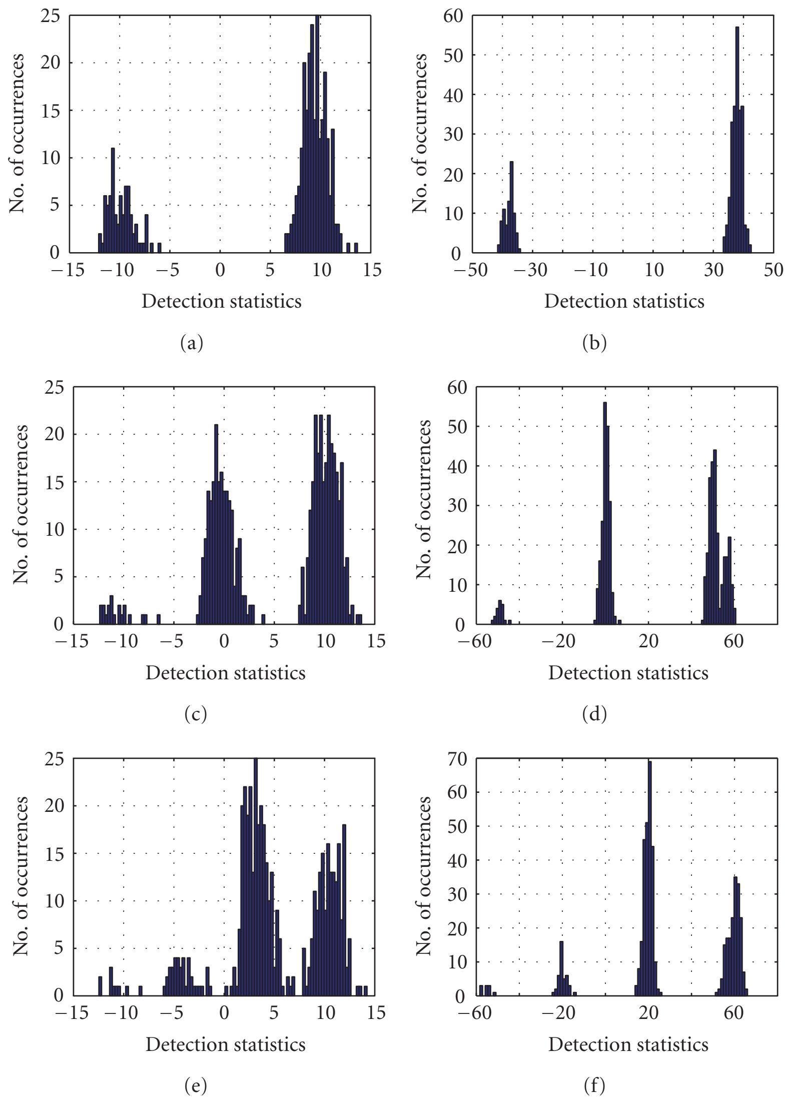


Figure 6.10. Histograms of detection statistics $\{T_N(i)\}$ of embedded fingerprints: (top row) single-fingerprint case, (middle row) two-user collusion case involving users 1 and 4 as colluders, (bottom row) three-user collusion case involving users 1, 4, and 8 as colluders; (left column) blind detection, (right column) nonblind detection.

where $\mathbf{d}$ is the random noise vector introduced by signal compression, transformation, and distortion. We assume this distortion to be additive. In the nonblind detection scenario, $\mathbf{x}$ is known and can be subtracted from the received signal. In the blind scenario, we treat $\mathbf{x}$ as part of the noise, and incorporate it into the overall distortion vector $\mathbf{d}$. The variable α is a scaling factor that is introduced to control the energy of the embedded watermark **w** and used to control the embedding watermark-to-noise ratio. It is often set using the *just-noticeable difference* from a human visual model.

At the detection stage, since we are only interested in detecting the colluders, we can project the received signal **y** onto the space spanned by **U**. The resulting $v \times 1$ coefficient vector, denoted as **T**, is the sufficient statistics for detection. Similarly, we denote the projection of **d** (for simplicity, we assume that **x** has been removed or incorporated into **d**) onto **U** as **n**, the noise vector in the watermark space. It is straightforward to show that

$$\mathbf{T} = \frac{\alpha}{K}\mathbf{B}\Phi + \mathbf{n}, \tag{6.24}$$

where the column vector $\boldsymbol{\Phi} \in \{0,1\}^n$ indicates colluders via the location of its components whose values are 1.

The goal behind the detection phase is to estimate collusion vector $\boldsymbol{\Phi}$ from the noisy observation **T**, while the code matrix **B** and the scaling factor α are known. Assuming the noise is white Gaussian, the maximum-likelihood solution of (6.24) is

$$\hat{\boldsymbol{\Phi}} = \arg\min_{\boldsymbol{\Phi}} \left\| \mathbf{T} - \frac{\alpha}{K}\mathbf{B}\boldsymbol{\Phi} \right\|, \tag{6.25}$$

that is, the distance between **T** and $(\alpha/K)\mathbf{B}\boldsymbol{\Phi}$ is minimum. If we assume that the number of colluders K is known, which is admittedly a strong assumption, then (6.25) is a typical integer least-squares problem. If the fingerprints are orthogonal, the simple matched filtering solution is optimal. However, if the fingerprints are not orthogonal, the problem generally involves searching every possible input, which is NP-hard [104]. The detection algorithms presented in Section 5.2.2 may be viewed as efficient, suboptimal alternatives to the maximum-likelihood detector. New techniques, involving the use of sphere decoding [103, 105], are currently being investigated and applied by the community to solve the colluder identification problem.

6.8. Chapter summary

In this chapter, we investigated the problem of applying coded fingerprinting for multimedia content that can resist collusion attacks and trace colluders. We developed a fingerprinting scheme based upon code modulation that does not require as many basis signals as orthogonal modulation in order to accommodate n users. We proposed anticollusion codes (ACCs) that are used in conjunction with modulation to fingerprint multimedia sources. Our anticollusion codes have the property that the composition of any subset of K or fewer codevectors is unique, which allows for the identification of subgroups of K or fewer colluders. We constructed binary-valued ACC under the logical AND operation using combinatorial designs. Our construction is suitable for both the on-off keying and antipodal form of binary code modulation. Further, our codes are efficient in that, for a given amount of colluders, they require only $\mathcal{O}(\sqrt{n})$ orthogonal signals to accommodate n users. For practical values of n, this is an improvement over prior work on fingerprinting generic digital data.

We introduced three different detection strategies that can be used with our ACC for identifying a suspect set of colluders. We performed experiments to evaluate the proposed ACC-based fingerprints. We first used a Gaussian signal model to examine the ability of the ACC to identify the colluders, as well as reveal the amount of innocent users that would be falsely placed under suspicion. We observed a close connection between the ability to capture colluders and the side effect of placing innocent users under suspicion. From our simulations, we observed that the proposed sequential detection scheme provides the most promising balance between capturing colluders and placing innocents under suspicion out of the three detection strategies examined. We also evaluated our fingerprints on real images, and observed that the values of the detection statistics can be well separated. This behavior allows the detector to accurately determine the colluder set by estimating a fingerprint codevector that corresponds to the colluder set.

7 Secure fingerprint multicast for video streaming

The popular streaming technology enables the customers to enjoy multimedia on the fly and starts playing multimedia while parts of the data are still being transmitted. In video streaming applications, a huge amount of data has to be transmitted to a large number of users using limited bandwidth available under stringent latency constraints. To maximize their profit, video streaming service providers aim to reduce the communication cost in transmitting each copy, and therefore, to accommodate as many users as possible. Prior art in the literature usually utilizes the multicast technology that provides a bandwidth advantage for content and network providers when distributing the same data to multiple users [106, 152]. It reduces the overall communication cost by duplicating packages only when routing paths to multiple receivers diverge [106, 107].

For streaming applications that require traitor tracing capability, the uniqueness of each copy poses new challenges to the secure and efficient distribution of differently marked copies. Multicast cannot be directly applied to fingerprinting applications where different users receive slightly different copies. A simple solution of unicasting each fingerprinted copy is obviously inefficient since the bandwidth requirement grows linearly as the number of users increases while the difference between different copies is small. This calls for fingerprint multicast schemes that reduce the communication cost of distributing fingerprinted media without revealing the secrecy of the video content as well as that of the embedded fingerprints.

This chapter addresses the secure and efficient transmission of multimedia for video streaming with traitor tracing requirement. We first analyze the security requirement in video streaming and then investigate the fingerprint multicast techniques to efficiently distribute fingerprinted media to multiple users. To examine the performance of fingerprint multicast schemes, we use the pure unicast scheme as the benchmark in which each fingerprinted copy is unicasted to the corresponding user. For the fingerprint multicast schemes, we evaluate their bandwidth efficiency, the collusion resistance of the embedded fingerprints, and the perceptual quality of the reconstructed sequence at the decoder's side, and investigate the tradeoff between the communication cost and computation complexity.

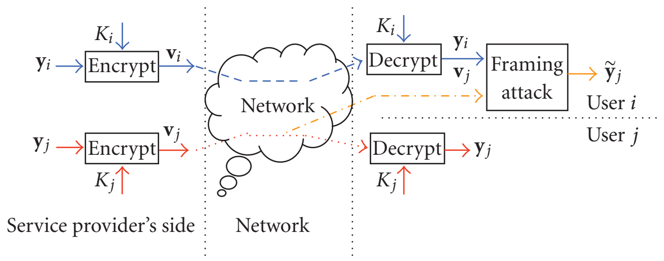


FIGURE 7.1. An example of framing attack on fingerprinting systems.

7.1. Secure video streaming

In video streaming applications, to protect the welfare and interests of the content owner, it is critical to ensure the proper distribution and authorized usage of multimedia content. To be specific, the desired security requirements in video streaming applications are [108] as follows.[1]

(1) *Secrecy of the video content*. Only legitimate users who have registered with the content owner/service provider can have access to the video content. Proper encryption should be applied to prevent outsiders who do not subscribe to the service from estimating the video's content.

(2) *Traitor tracing*. After the data are distributed to the legitimate users, the content owner has to protect multimedia from unauthorized manipulation and redistribution. Digital fingerprinting is one possible solution to traitor tracing and can be used to identify the source of the illicit copies.

(3) *Robustness of the embedded fingerprints*. If digital fingerprinting is used for traitor tracing, it is required that the embedded fingerprints can survive common signal processing (e.g., compression), attacks on a single copy [110, 111], as well as multiuser collusion attacks [23, 62].

(4) *Antiframing*. The clear text of a fingerprinted copy is known only by the corresponding legitimate user whose fingerprint is embedded in that copy, and no other users of the service can access that copy in clear text and frame an innocent user.

We will explain the antiframing requirement in detail. In digital fingerprinting applications, different fingerprinted copies do not differ significantly from each other. If the content owner or the service provider does not protect the transmitted bit streams appropriately, it is very easy for an attacker, who subscribes to the video streaming service, to impersonate an innocent user of the service.

Figure 7.1 shows an example of the framing attack. Assume that K_i and K_j are the secret keys of the ith user and the jth user, respectively; $\mathbf{y}_i$ and $\mathbf{y}_j$ are the clear

[1]Depending on the applications, there might be other security requirements except these listed in this chapter, for example, sender authentication and data integrity verification [109]. It is out of the scope of this book and we assume that the distribution systems have already included the corresponding security modules if required.

text versions of two fingerprinted copies for the ith and the jth users, respectively; and $\mathbf{v}_i$ and $\mathbf{v}_j$ are the ciphertext versions of $\mathbf{y}_i$ and $\mathbf{y}_j$ encrypted with K_i and K_j, respectively. User i first decrypts $\mathbf{v}_i$ that is transmitted to him and reconstructs $\mathbf{y}_i$. Assume that he also intercepts $\mathbf{v}_j$ that is transmitted to the ith user. Without appropriate protection by the content owner or the service provider, user i can compare $\mathbf{v}_j$ with $\mathbf{y}_i$, estimate $\mathbf{y}_j$ without knowledge of K_j, and generate $\tilde{\mathbf{y}}_j$ of good quality, which is an estimated version of $\mathbf{y}_j$. User i can then redistribute $\tilde{\mathbf{y}}_j$ or use $\tilde{\mathbf{y}}_j$ during collusion. This framing puts innocent user j under suspicion and disables the content owner from capturing attacker i. The content owner must prohibit such framing attacks.

To summarize, before transmission, the content owner should embed unique and robust fingerprints in each distributed copy, and apply proper encryption to the bit streams to protect both the content of the video and each fingerprinted coefficient in all fingerprinted copies.

7.2. Prior art in secure fingerprint multicast

Given the security requirement listed in the previous section, the most straightforward way to securely distribute the fingerprinted copies is the pure unicast scheme, in which each fingerprinted copy is encoded independently, encrypted with the corresponding user's secret key, and unicasted to him. It is simple and has limited requirement on the receivers' computation capability. However, from the bandwidth's point of view, it is inefficient because the required bandwidth is proportional to the number of users while the difference between different copies is small. In this chapter, we use the pure unicast distribution scheme as the benchmark for the purpose of performance comparison.

To improve the bandwidth efficiency, for applications that wish to survive collusion by a few attackers (e.g., ten traitors), one possible solution to enabling multicast of fingerprinted media is to adjust the fingerprint design to suit an existing communication framework (e.g., multicast). In fact, most prior work on fingerprint multicast followed this philosophy of design [112, 113, 114, 115], and their fingerprint code design was similar to that of the Boneh-Shaw code [77].

In [112], a two-layer fingerprint design was used where the inner layer of spread-spectrum embedding [23] was combined with the outer fingerprint code of [77]. Two uniquely fingerprinted copies were generated, encrypted, and multicasted, where each frame in the two copies was encrypted with a unique key. Each user was given a unique set of keys for decryption and reconstructed a unique sequence. Their fingerprinting system was vulnerable to collusion attacks. From their reported results, for a two-hour video distributed to 10,000 users, only when no more than three users colluded could their system detect at least one colluder correctly with probability 0.9. Similar work was presented in [115, 116, 117].

In [114], the sender generated and multicasted several uniquely fingerprinted copies, and trusted routers in the multicast tree differently forwarded fingerprinted packets to different users. In [118], a hierarchy of trusted intermediaries was introduced into the network. All intermediaries embedded their unique IDs as fingerprints into the content as they forwarded the packets through the network, and a user was identified by all the IDs of the intermediaries that were embedded in his received copy.

In [119], fingerprints were embedded in the DC coefficients of the luminance component in I frames using spread-spectrum embedding. For each fingerprinted copy, a small portion of the MPEG stream, including the fingerprinted DC coefficients, was encrypted and unicasted to the corresponding user, and the rest was multicasted to all users to achieve the bandwidth efficiency. The embedded fingerprints in [119] have limited collusion resistance since they are only embedded in a small number of coefficients.

A joint fingerprint and decryption scheme was proposed in [113]. In their work, the content owner encrypted the extracted features from the host signal with a secret key K_S known to the content owner only, multicasted the encrypted content to all users, and transmitted to each user i a unique decryption key $K_i \neq K_S$. At the receiver's side, each user partially decrypted the received bit stream, and reconstructed a unique version of the original host signal due to the uniqueness of the decryption key. In [113], the fingerprint information is essentially the asymmetric key pair (K_S, K_i), and the unique signature from the partial decryption was used to identify the attacker/colluders.

7.3. General fingerprint multicast distribution scheme

Most prior work considered applications where the goal of the fingerprinting system is to resist collusion attacks by a few colluders (e.g., seven or ten traitors), and designed the efficient distribution schemes accordingly. In many video applications, the number of users is in the order of thousands or tens of thousands, and therefore, the potential number of colluders is in the order of dozens or hundreds. We focus on applications that aim to withstand dozens of colluders instead of just a few and apply the fingerprint design with strong traitor tracing capability [80, 120]. For such applications, we utilize the existing multicast communication technology to fit the fingerprint design and reduce the bandwidth requirement without sacrificing the robustness of the embedded fingerprints.

In this section, utilizing the perceptual constraints on fingerprint embedding, we develop a general fingerprint multicast distribution scheme that can be used with most multimedia fingerprinting systems where the spread-spectrum embedding is adopted [121]. We consider a video distribution system that uses MPEG-2 encoding standard. For simplicity, we assume that all the distributed copies are encoded at the same bit rate and have approximately the same perceptual quality. To reduce the computation cost at the sender's side, fingerprints are embedded in the DCT domain. The block-based human visual models [24] are used to guarantee the imperceptibility and control the energy of the embedded fingerprints.

From human visual models [24], not all DCT coefficients are embeddable due to the imperceptibility constraints on the embedded fingerprints, and a nonembeddable coefficient has the same value in all copies. To reduce the bandwidth in transmitting the nonembeddable coefficients, we develop a general fingerprint multicast scheme: the nonembeddable coefficients are multicasted to all users, and the rest of the coefficients are embedded with unique fingerprints and unicasted to the corresponding user.[2]

In the general fingerprint multicast scheme, to guarantee that no outsiders can access the video content, a key that is shared by all users is used to encrypt the multicasted bit stream and the generalized index mapping [13, 122] is used to encrypt portions of the compressed bit streams that carry the most important information of the video content: the DC coefficients in the intrablocks and the motion vectors in the interblocks. To protect the fingerprinted coefficients, each unicasted bit stream is encrypted with the corresponding user's secret key. The generalized index mapping can be applied to the fingerprinted AC coefficients to prevent the attackers from framing an innocent user at the cost of introducing significant bit-rate overhead.[3] To protect the fingerprinted coefficients without significant bit-rate overhead, similar to that in [123], the stream cipher [30] from traditional cryptography is applied to the compressed bit streams of the AC coefficients.[4] It has no impact on the compression efficiency. In addition, the bit-stuffing scheme [122] is used to prevent the encrypted data from duplicating MPEG headers/markers.

Figure 7.2 shows the MPEG-2-based general fingerprint multicast scheme for video-on-demand applications where the video is stored in compressed format. Assume that K_{multi} is a key that is shared by all users, and K_i is the user's secret key. The key steps in the fingerprint embedding and distribution at the server's side are as follows.

(1) A unique fingerprint is generated for each user.

(2) The compressed bit stream is split into two parts: the first one includes motion vectors, quantization factors, and other side information and is not altered, and the second one contains the coded DCT coefficients and is variable-length decoded.

(3) Motion vectors, quantization factors, and other side information are left intact, and only the values of the DCT coefficients are changed. For each DCT coefficient, if it is not embeddable, it is variable-length coded with other nonembeddable coefficients. Otherwise, first, it is inversely quantized. Then for each user, the corresponding fingerprint component is embedded using spread-spectrum embedding, and the resulting fingerprinted coefficient is quantized and variable-length coded with other fingerprinted coefficients.

(4) The nonembeddable DCT coefficients are encrypted with K_{multi} and multicasted to all users, together with the positions of the embeddable coefficients in the 8×8 DCT blocks, motion vectors, and other shared information; the

[2]We assume that each receiver has moderate computation capability and can listen to at least 2 channels simultaneously to reconstruct one video sequence. We also assume that the receivers have large enough buffers to smooth out the jittering of delays among different channels.

[3]From [122], the bit rate is increased by more than 5.9% if two nonzero AC coefficients in each intrablock are encrypted.

[4]Only the content-carrying fields are encrypted and the headers/markers are transmitted in clear text.

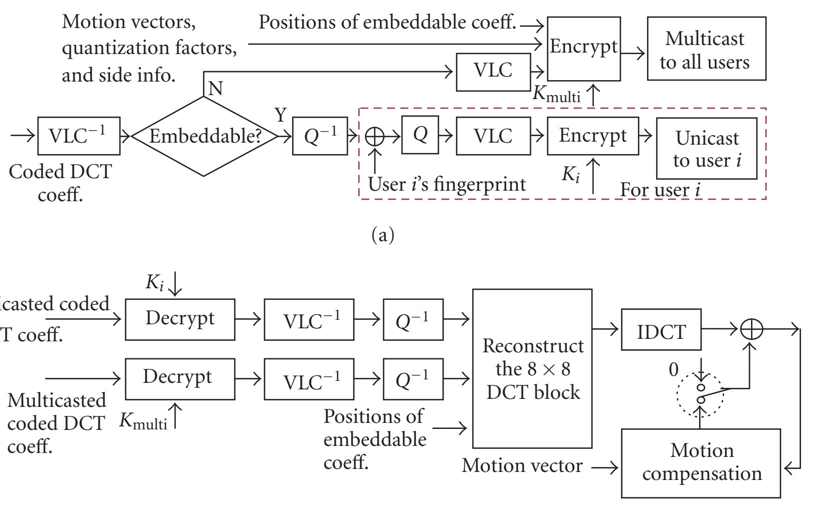


FIGURE 7.2. The MPEG-2-based general fingerprint multicast scheme for video-on-demand applications: (a) the fingerprint embedding and distribution process at the server's side; (b) the decoding process at the user's side.

fingerprinted DCT coefficients are encrypted with each user's secret key and unicasted to them.

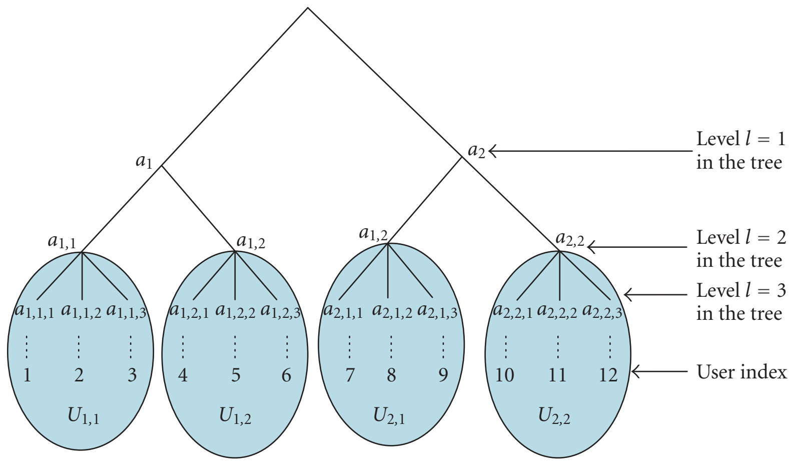
For live applications where the video is compressed and transmitted at the same time, the fingerprint embedding and distribution process is similar to that for video-on-demand applications.

The decoder at the *i*th user's side is the same for both types of applications and is similar to a standard MPEG-2 decoder. After decrypting, variable-length decoding and inversely quantizing both the unicasted bit stream to user *i* and the multicasted bit stream to all users, the decoder puts each reconstructed DCT coefficient in its original position in the 8×8 DCT block. Then, it applies inverse DCT and motion compensation to reconstruct each frame.

7.4. Joint fingerprint design and distribution scheme

The general fingerprint multicast scheme is designed for the general fingerprinting applications that use spread-spectrum embedding. To further improve the bandwidth efficiency, we utilize the special structure of the embedded fingerprints and introduce a joint fingerprint design and distribution scheme [124].

In this section, we first compare two fingerprint modulation schemes commonly used in the literature, the CDMA-based and the TDMA-based fingerprint modulations, including the bandwidth efficiency and the collusion resistance. Then in Section 7.4.2, we develop a joint fingerprint design and distribution scheme that achieves both the robustness against collusion attacks and the bandwidth efficiency of the distribution scheme. In Section 7.4.3, we take the



a_1
a_2
Level $l = 1$ in the tree
$a_{1,1}$
$a_{1,2}$
$a_{1,2}$
$a_{2,2}$
Level $l = 2$ in the tree
Level $l = 3$ in the tree
$a_{1,1,1}$ $a_{1,1,2}$ $a_{1,1,3}$
$a_{1,2,1}$ $a_{1,2,2}$ $a_{1,2,3}$
$a_{2,1,1}$ $a_{2,1,2}$ $a_{2,1,3}$
$a_{2,2,1}$ $a_{2,2,2}$ $a_{2,2,3}$
1 2 3 4 5 6 7 8 9 10 11 12
User index
$U_{1,1}$
$U_{1,2}$
$U_{2,1}$
$U_{2,2}$

FIGURE 7.3. A tree-structure-based fingerprinting scheme with $L = 3$, $D_1 = D_2 = 2$, and $D_3 = 3$.

computation constraints into consideration, and adjust the joint fingerprint design and distribution scheme to minimize the communication cost under the computation constraints.

7.4.1. Comparison of fingerprint modulation schemes

Group-oriented fingerprint design in Chapter 5 uses the tree structure to explore the hierarchical relationship among users. Figure 7.3 shows an example of the fingerprint tree with three levels. We consider a symmetric tree structure where each node at level $l-1$ has the same number of children nodes D_l for $l = 1, \dots, L-1$. Given the tree structure as in Figure 7.3, a unique basis fingerprint $\mathbf{a}_{i_1,\dots,i_l}$ following Gaussian distribution $\mathcal{N}(0, \sigma_W^2)$ is generated for each node $[i_1, \dots, i_l]$ in the tree, and the basis fingerprints $\{\mathbf{a}\}$ are independent of each other. For the ith user whose index is $i = [i_1, \dots, i_L]$, a total of L fingerprints $\mathbf{a}_{i_1}, \mathbf{a}_{i_1,i_2}, \dots, \mathbf{a}_{i_1,\dots,i_L}$ are embedded in the fingerprinted copy $\mathbf{y}_i$ that is distributed to him. Assume that the host signal $\mathbf{x}$ has a total of N embeddable coefficients. There are two different methods to embed the L fingerprints into the host signal $\mathbf{x}$: the CDMA-based and the TDMA-based fingerprint modulations.

The CDMA-based fingerprint modulation. In the CDMA-based fingerprint modulation, the basis fingerprints $\{\mathbf{a}\}$ are of the same length N and equal energy. The ith user's fingerprint $\mathbf{s}_i$ is generated by $\mathbf{s}_i = \sqrt{\rho_1}\mathbf{a}_{i_1} + \sqrt{\rho_2}\mathbf{a}_{i_1,i_2} + \cdots + \sqrt{\rho_L}\mathbf{a}_{i_1,i_2,\dots,i_L}$, the same as in Chapter 5. The fingerprinted copy distributed to the ith user is $\mathbf{y}_i = \mathbf{x} + \mathbf{s}_i$, where $\mathbf{x}$ is the host signal. $\{\rho_l\}$ are determined by the probabilities of users under different tree branches to collude with each other, $0 \le \rho_1, \dots, \rho_L \le 1$, and $\sum_{j=1}^{L} \rho_j = 1$. They are used to control the energy of the embedded fingerprints at each level and adjust the correlation between fingerprints assigned to different users.

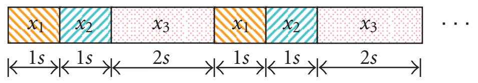


Figure 7.4. An example of the partitioning of the host signal for a tree with $L = 3$ and $[\rho_1, \rho_2, \rho_3] = [1/4, 1/4, 1/2]$.

The TDMA-based fingerprint modulation. In the TDMA-based fingerprint modulation, the host signal $\mathbf{x}$ is divided into L nonoverlapping parts $\mathbf{x}^1, \dots, \mathbf{x}^L$ such that the number of embeddable coefficients in $\mathbf{x}^l$ is $N_l = \rho_l N$ with $\sum_{l=1}^{L} N_l = N$. An example of the partitioning of the host signal is shown in Figure 7.4 for a tree with $L = 3$, $[\rho_1, \rho_2, \rho_3] = [1/4, 1/4, 1/2]$ and $[N_1, N_2, N_3] = N[1/4, 1/4, 1/2]$. For every 4 seconds, all the frames in the first second belong to $\mathbf{x}^1$, all the frames in the second second are in $\mathbf{x}^2$ and all the frames in the last two seconds are in $\mathbf{x}^3$. If the video sequence is long enough, the number of embeddable coefficients in $\mathbf{x}^l$ is approximately N_l.

In the TDMA-based fingerprint modulation, the basis fingerprints $\{\mathbf{a}_{i_1,\dots,i_l}\}$ at level l are of length N_l. In the fingerprinted copy $\mathbf{y}_i$ that is distributed to the ith user, the basis fingerprint $\mathbf{a}_{i_1,\dots,i_l}$ at level l is embedded in the lth part of the host signal $\mathbf{x}^l$, and the lth part of the fingerprinted copy $\mathbf{y}_i$ is $\mathbf{y}_i^l = \mathbf{x}^l + \mathbf{a}_{i_1,\dots,i_l}$.

Performance comparison of the CDMA-based and the TDMA-based fingerprint modulations. To compare the CDMA-based and the TDMA-based fingerprint modulation schemes in the tree-based fingerprinting systems, we measure the energy of the fingerprints that are embedded in different parts of the fingerprinted copies. Assume that the host signal $\mathbf{x}$ is partitioned into L nonoverlapping parts $\{\mathbf{x}^l\}_{l=1,\dots,L}$ where there are N_l embeddable coefficients in $\mathbf{x}^l$, the same as in the TDMA-based modulation. We also assume that for the ith user, $\mathbf{s}_i^l$ is the fingerprint that is embedded in $\mathbf{x}^l$, and $\mathbf{y}_i^l = \mathbf{x}^l + \mathbf{s}_i^l$ is the lth part of the fingerprinted copy that is distributed to the ith user. Define $E_{k,l}$ as the energy of the basis fingerprint $\mathbf{a}_{i_1,\dots,i_k}$ at level k that is embedded in $\mathbf{y}_i^l$, and $E_l \triangleq \sum_{k=1}^{L} E_{k,l}$ is the overall energy of $\mathbf{s}_i^l$. We further define a matrix $\mathbf{P}$ whose element at row k and column l is $p_{k,l} \triangleq E_{k,l}/E_l$, and it is the ratio of the energy of the kth level fingerprint $\mathbf{a}_{i_1,\dots,i_k}$ embedded in $\mathbf{y}_i^l$ over the overall energy of $\mathbf{s}_i^l$. The $\mathbf{P}$ matrices for the CDMA-based and the TDMA-based fingerprint modulation schemes are

$$\mathbf{P}^{\text{CDMA}} = \begin{pmatrix} \rho_1 & \rho_1 & \cdots & \rho_1 \\ \rho_2 & \rho_2 & \cdots & \rho_2 \\ \vdots & \vdots & \ddots & \vdots \\ \rho_L & \rho_L & \cdots & \rho_L \end{pmatrix}_{L\times L}, \qquad \mathbf{P}^{\text{TDMA}} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}_{L\times L}, \tag{7.1}$$

respectively. In addition, in the TDMA-based fingerprint modulation scheme,

$$\mathbf{P}^{\text{TDMA}} [N_1 \quad N_2 \quad \cdots \quad N_L]^T = N[\rho_1 \quad \rho_2 \quad \cdots \quad \rho_L]^T \tag{7.2}$$

and $\sum_{l=1}^{L} N_l = N$, where N is the total number of embeddable coefficients in the host signal.

Comparison of bandwidth efficiency. First, in the TDMA-based modulation scheme, $p_{k,l} = 0$ for $k > l$, and therefore, the lth part of the fingerprinted copy $\mathbf{y}_i^l$ is only embedded with the basis fingerprints at level $k \le l$ in the tree. Note that the basis fingerprints $\{\mathbf{a}_{i_1,\dots,i_k}\}_{k\le l}$ are shared by users in the subgroup $\mathbf{U}_{i_1,\dots,i_l} \triangleq \{j = [j_1,\dots,j_l,\dots,j_L] : j_1 = i_1,\dots,j_l = i_l\}$, so is $\mathbf{y}_i^l$. Consequently, in the TDMA-based fingerprint modulation, the distribution system cannot only multicast the nonembeddable coefficients to all users, but it can also multicast part of the fingerprinted coefficients that are shared by a subgroup of users to them. In the CDMA-based fingerprint modulation, $p_{k,l} > 0$ for $k > l$ and the distribution system can only multicast the nonembeddable coefficients. Therefore, from the bandwidth efficiency's point of view, the TDMA-based modulation is more efficient than the CDMA-based fingerprint modulation.

Comparison of collusion resistance. Second, in the TDMA-based modulation scheme, $p_{k,l} = 0$ for $k \ne l$ and the basis fingerprints $\mathbf{a}_{i_1,\dots,i_l}$ at level l are only embedded in the lth part of the fingerprinted copy $\mathbf{y}_i^l$. With the TDMA-based modulation scheme, by comparing all the fingerprinted copies that they have, the colluders can distinguish different parts of the fingerprinted copies that are embedded with fingerprints at different levels in the tree. They can also figure out the structure of the fingerprint tree and the positions of all colluders in the tree. Based on the information they collect, they can apply a specific attack against the TDMA-based fingerprint modulation, *the interleaving-based collusion attack*.

Assume that S_C is the set containing the indices of all colluders, and $\{\mathbf{y}_k\}_{k\in S_C}$ are the fingerprinted copies that they received. In the interleaving-based collusion attacks, the colluders divide themselves into L subgroups $\{S_C(l) \subseteq S_C\}_{l=1,\dots,L}$, and there exists at least one $1 \le l < L$ such that the lth subgroup $S_C(l)$ and the $(l+1)$th subgroup $S_C(l+1)$ are under different branches in the tree and are nonoverlapping, that is, $S_C(l) \cap S_C(l+1) = \emptyset$. The colluded copy $\mathbf{y}$ contains L nonoverlapping parts $\{\mathbf{y}^l\}_{l=1,\dots,L}$, and the colluders in the subgroup $S_C(l)$ generate the lth part of the colluded copy by $\mathbf{y}^l = g(\{\mathbf{y}_k^l\}_{k\in SC_l})$, where $g(\cdot)$ is the collusion function. Figure 7.5 shows an example of the interleaving-based collusion attack on the tree-based fingerprint design of Figure 7.3. Assume that $S_C = \{1 = [1,1,1],\ 2 = [1,1,2],\ 4 = [1,2,1],\ 7 = [2,1,1]\}$ is the set containing the indices of the colluders. The colluders choose $S_C(1) = \{7\}$, $S_C(2) = \{4\}$ and $S_C(3) = \{1,2\}$, and generate the colluded copy $\mathbf{y}$, where

$$
\begin{aligned}
\mathbf{y}^1 &= \mathbf{y}_7^1 = \mathbf{x}^1 + \mathbf{a}_2,\\
\mathbf{y}^2 &= \mathbf{y}_4^2 = \mathbf{x}^2 + \mathbf{a}_{1,2},\\
\mathbf{y}^3 &= \frac{(\mathbf{y}_1^3 + \mathbf{y}_2^3)}{2} = \mathbf{x}^3 + \frac{(\mathbf{a}_{1,1,1} + \mathbf{a}_{1,1,2})}{2}.
\end{aligned}
\tag{7.3}
$$

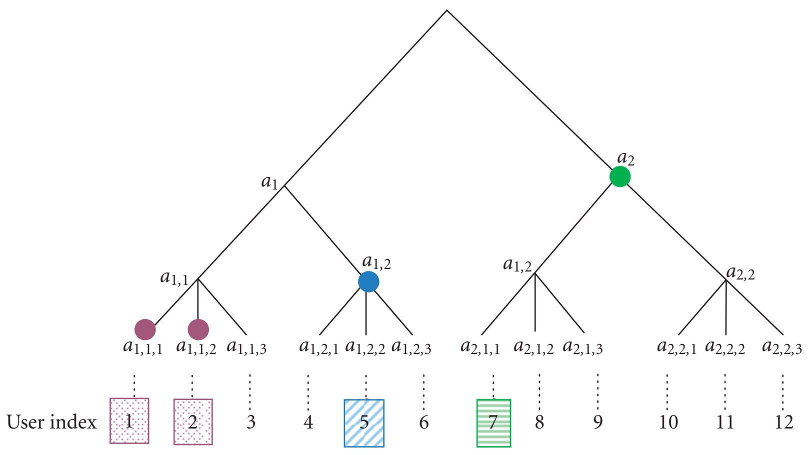

a_1
a_2
$a_{1,1}$
$a_{1,2}$
$a_{1,2}$
$a_{2,2}$
$a_{1,1,1}$ $a_{1,1,2}$ $a_{1,1,3}$ $a_{1,2,1}$ $a_{1,2,2}$ $a_{1,2,3}$ $a_{2,1,1}$ $a_{2,1,2}$ $a_{2,1,3}$ $a_{2,2,1}$ $a_{2,2,2}$ $a_{2,2,3}$
User index
1 2 3 4 5 6 7 8 9 10 11 12

Figure 7.5. An example of the interleaving-based collusion attack on the tree-based fingerprinting system shown in Figure 7.3 with the TDMA-based fingerprint modulation.

In the detection process, at the first level in the tree, although both $\mathbf{a}_1$ and $\mathbf{a}_2$ are guilty, the detector can only detect the existence of $\mathbf{a}_2$ because $\mathbf{a}_1$ is not in any part of the colluded copy $\mathbf{y}$. Following the multistage detection process in Chapter 5, the detector outputs the estimated guilty region $[2]$ at the first level of the tree. At the second level, the detector tries to detect whether $[2,1]$ and $[2,2]$ are the guilty subregions, and finds out that neither of these two are guilty since $\mathbf{a}_{2,1}$ and $\mathbf{a}_{2,2}$ are not in $\mathbf{y}$. To continue the detection process, the detectors have to check the existence of each of the four fingerprints $\{\mathbf{a}_{i_1,i_2}\}$ in $\mathbf{y}$. The performance of the detection process in the TDMA-based fingerprint modulation is worse than that of the CDMA-based fingerprint modulation [120], and it is due to the special structure of the fingerprint design and the unique "multistage" detection process in the group-oriented fingerprinting systems.

To summarize, in the group-oriented fingerprinting systems, the TDMA-based fingerprint modulation improves the bandwidth efficiency of the distribution system at the cost of the robustness against collusion attacks.

7.4.2. Joint fingerprint design and distribution

In the joint fingerprint design and distribution scheme, the content owner first applies the group-oriented fingerprint design in [120] and generates the fingerprint tree. Then, he embeds the fingerprints using the joint TDMA and CDMA fingerprint modulation scheme introduced in Section 7.4.1, which improves the bandwidth efficiency without sacrificing the robustness. Finally, the content owner distributes the fingerprinted copies to users using the distribution scheme introduced in Section 7.4.2.

Design of the joint TDMA and CDMA fingerprint modulation. To achieve both the robustness against collusion attacks and the bandwidth efficiency of the distribution scheme, we develop a *joint TDMA and CDMA fingerprint modulation scheme*, whose $\mathbf{P}$ matrix is an upper triangular matrix. In $\mathbf{P}^{\text{Joint}}$, we let $p_{k,l} = 0$ for $k > l$ to achieve the bandwidth efficiency. For $k \le l$, we choose $0 < p_{k,l} \le 1$ to achieve the robustness. Take the interleaving-based collusion attack shown in Figure 7.5 as an example, in the joint TDMA and CDMA fingerprint modulation, although $\mathbf{a}_1$ is not in $\mathbf{y}^1$, it can still be detected from $\mathbf{y}^2$ and $\mathbf{y}^3$. Consequently, the detector can apply the "multistage" detection and narrow down the guilty-region step by step, the same as in the CDMA-based fingerprint modulation.

At level 1, $p_{1,1} = 1$. At level $2 \le l \le L$, given $p_{l,l}$, we seek $\{p_{k,l}\}_{k<l}$ to satisfy $E_{1,l} : E_{2,l} : \cdots : E_{l-1,l} = \rho_1 : \rho_2 : \cdots : \rho_{l-1}$. We can show that $p_{k,l} = \rho_k(1 - p_{l,l})/(\rho_1 + \cdots + \rho_{l-1})$ for $k < l$, and

$$\mathbf{P}^{\text{Joint}} = \begin{pmatrix} 1 & 1 - p_{2,2} & \cdots & (1 - p_{L,L})\dfrac{\rho_1}{1 - \rho_L} \\ 0 & p_{2,2} & \cdots & (1 - p_{L,L})\dfrac{\rho_2}{1 - \rho_L} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_{L,L} \end{pmatrix}_{L\times L}. \tag{7.4}$$

Given $\{p_{l,l}\}_{l=1,\ldots,L}$ and $\mathbf{P}^{\text{Joint}}$ as in (7.4), we seek $N_1, N_2, \ldots, N_L$ to satisfy

$$\begin{gathered} \mathbf{P}^{\text{Joint}}[N_1 \quad N_2 \quad \cdots \quad N_L]^T = N[\rho_1 \quad \rho_2 \quad \cdots \quad \rho_L]^T \\ \text{s.t. } \sum_{l=1}^{L} N_l = N, \quad 0 \le N_l \le N. \end{gathered} \tag{7.5}$$

From (7.4), when $p_{L,L} = \rho_L$, it is the CDMA-based fingerprint modulation. Therefore, we only consider the case where $p_{L,L} > \rho_L$. Define

$$\begin{gathered} \mathbf{A} = \begin{pmatrix} 1 & 1 - p_{2,2} & \cdots & \dfrac{(1 - p_{L-1,L-1})\rho_1}{\sum_{k=1}^{L-2}\rho_k} \\ 0 & p_{2,2} & \cdots & \dfrac{(1 - p_{L-1,L-1})\rho_2}{\sum_{k=1}^{L-2}\rho_k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_{L-1,L-1} \end{pmatrix}, \\ \mathbf{B} = \frac{1 - p_{L,L}}{1 - \rho_L}\begin{pmatrix} \rho_1 & \cdots & \rho_1 \\ \rho_2 & \cdots & \rho_2 \\ \vdots & \ddots & \vdots \\ \rho_{L-1} & \cdots & \rho_{L-1} \end{pmatrix}, \end{gathered} \tag{7.6}$$

where A and B are of rank $(L-1)\times(L-1)$. We can show that (7.5) can be rewritten as

$$\left(\begin{array}{ccc} & \mathbf{A}-\mathbf{B} & \\ \hline p_{L,L} & \cdots & p_{L,L} \end{array}\right)\begin{bmatrix} N_1 \\ \vdots \\ N_{L-1} \end{bmatrix} = (p_{L,L}-\rho_L)N\begin{bmatrix} \dfrac{\rho_1}{1-\rho_L} \\ \vdots \\ \dfrac{\rho_{L-1}}{1-\rho_L} \\ 1 \end{bmatrix}, \tag{7.7}$$
$$N_L = N - \sum_{l=1}^{L-1} N_l.$$

Define

$$\mathbf{Q} \triangleq \left(\begin{array}{ccc} & \mathbf{A}-\mathbf{B} & \\ \hline p_{L,L} & \cdots & p_{L,L} \end{array}\right),$$
$$\underline{c} \triangleq \frac{(p_{L,L}-\rho_L)N}{1-\rho_L}\begin{bmatrix}\rho_1 & \cdots & \rho_{L-1} & 1-\rho_L\end{bmatrix}^T. \tag{7.8}$$

Given $\{p_{l,l}\}$, if $\mathbf{Q}$ is of full rank, then the least square solution to (7.7) is

$$\begin{bmatrix} N_1 & N_2 & \cdots & N_{L-1}\end{bmatrix}^T = \mathbf{Q}^\dagger \underline{c}, \qquad N_L = N - \sum_{l=1}^{L-1} N_l, \tag{7.9}$$

where $\mathbf{Q}^\dagger = (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}$ is the pseudoinverse of $\mathbf{Q}$. Finally, we need to verify the feasibility of the solution (7.9), that is, if $0 \le N_l \le N$ for all $1 \le l \le L$. If not, another set of $\{p_{l,l}\}_{l=1,\dots,L}$ has to be used.

Fingerprint embedding and detection in the joint TDMA and CDMA modulation. In the joint TDMA and CDMA fingerprint modulation scheme, given $\mathbf{P}^{\text{Joint}}$ as in (7.4) and $\{N_l\}_{l=1,\dots,L}$ as in (7.9), for each basis fingerprint $\mathbf{a}_{i_1,\dots,i_l}$ at level $1 \le l \le L$ in the tree, $\mathbf{a}_{i_1,\dots,i_l} = \mathbf{a}^l_{i_1,\dots,i_l} \Cup \mathbf{a}^{l+1}_{i_1,\dots,i_l} \Cup \cdots \Cup \mathbf{a}^L_{i_1,\dots,i_l}$, where $\{\mathbf{a}^k_{i_1,\dots,i_l}\}_{k=l,\dots,L}$ follow Gaussian distribution $\mathcal{N}(0,\sigma_W^2)$ and are independent of each other. $\mathbf{a}^k_{i_1,\dots,i_l}$ for $k \ge l$ is of length N_k, and is embedded in $\mathbf{x}^k$. "$\Cup$" is the concatenation operator. For the ith user whose index is $i = [i_1,\dots,i_L]$, the lth part of the fingerprinted copy that he receives is $\mathbf{y}^l_{i_1,\dots,i_l} = \mathbf{x}^l + \mathbf{s}^l_{i_1,\dots,i_l}$, where

$$\mathbf{s}^l_{i_1,\dots,i_l} = \sqrt{p_{1,l}}\mathbf{a}^l_{i_1} + \sqrt{p_{2,l}}\mathbf{a}^l_{i_1,i_2} + \cdots + \sqrt{p_{l,l}}\mathbf{a}^l_{i_1,\dots,i_l}. \tag{7.10}$$

During collusion, assume that there are a total of K colluders and S_C is the set containing their indices. The colluders divide them into L subgroups $\{S_C(l) \subseteq S_C\}_{l=1,\dots,L}$. For each $1 \leq l \leq L$, given the K copies $\{\mathbf{y}_k^l\}_{k \in S_C}$, the colluders in $S_C(l)$ generate the lth part of the colluded copy by $\mathbf{y}^l = g(\{\mathbf{y}_k^l\}_{k \in S_C(l)})$, where $g(\cdot)$ is the collusion function. Assume that $\mathbf{y} = \mathbf{y}^1 \uplus \cdots \uplus \mathbf{y}^L$ is the colluded copy that is redistributed by the colluders.

At the detector's side, given the colluded copy $\mathbf{y}$, for each $1 \leq l \leq L$, the detector first extracts the fingerprint $\mathbf{w}^l$ from $\mathbf{y}^l$, and the detection process is similar to that in Chapter 5.

Detection at the first level of the tree. The detector correlates the extracted fingerprint $\{\mathbf{w}^l\}_{l=1,\dots,L}$ with each of the D_1 fingerprints $\{\mathbf{a}_{i_1}\}_{i_1=1,\dots,D_1}$ at level 1 and calculates the detection statistics

$$T_1(i_1) = \frac{\sum_{k=1}^{L} \langle \mathbf{w}^k, \mathbf{a}_{i_1}^k \rangle}{\sqrt{\sum_{k=1}^{L} \|\mathbf{a}_{i_1}^k\|^2}} \tag{7.11}$$

for $i_1 = 1, \dots, D_1$. The estimated guilty regions at level 1 are $\hat{\mathbf{j}}_1 = \{[i_1] : T_1(i_1) > h_1\}$, where h_1 is a predetermined threshold for fingerprint detection at the first level in the tree.

Detection at level $2 \leq l \leq L$ *in the tree.* Given the previously estimated guilty regions $\hat{\mathbf{j}}_{l-1}$, for each $[i_1, i_2, \dots, i_{l-1}] \in \hat{\mathbf{j}}_{l-1}$, the detector calculates the detection statistics

$$T_l(i_1, \dots, i_{l-1}, i_l) = \frac{\sum_{k=l}^{L} \langle \mathbf{w}^k, \mathbf{a}_{i_1,\dots,i_{l-1},i_l}^k \rangle}{\sqrt{\sum_{k=l}^{L} \|\mathbf{a}_{i_1,\dots,i_{l-1},i_l}^k\|^2}} \tag{7.12}$$

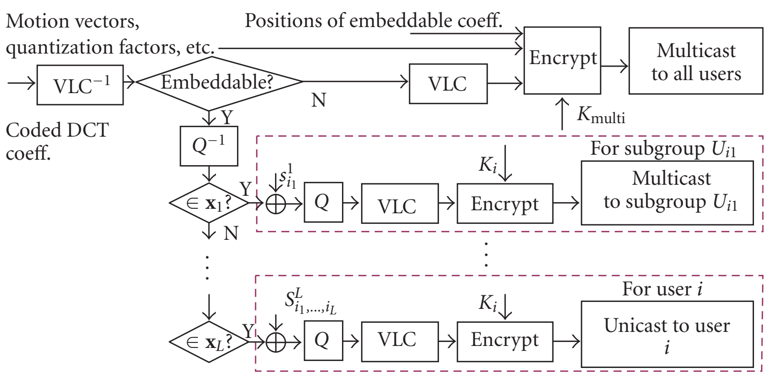
for $i_l = 1, \dots, D_l$, and narrows down the guilty regions to

$$\hat{\mathbf{j}}_l = \{[i_1, \dots, i_l] : [i_1, \dots, i_{l-1}] \in \hat{\mathbf{j}}_{l-1}, T_l(i_1, \dots, i_l) \geq h_l\}, \tag{7.13}$$

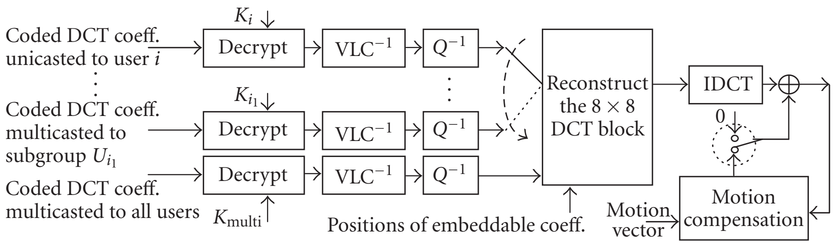
where h_l is a predetermined threshold for fingerprint detection at level l in the tree. Finally, the detector outputs the estimated colluder set $\hat{S}_C = \{i : i = [i_1, \dots, i_L] \in \hat{\mathbf{j}}_L\}$.

Fingerprint distribution in the joint fingerprint design and distribution scheme. In the joint fingerprint design and distribution scheme, the MPEG-2-based fingerprint distribution scheme for video on demand applications is shown in Figure 7.6. Assume that K_{multi} is a key that is shared by all users, $K_{i_1,\dots,i_l}$ is a key shared by a subgroup of users $\mathbf{U}_{i_1,\dots,i_l}$, and K_i is the ith user's secret key. The encryption method in the joint fingerprint design and distribution scheme is the same as that in the general fingerprint multicast. The key steps in the fingerprint embedding and distribution process at the server's side are as follows.

(i) For each user i, the fingerprint $\mathbf{s}_i$ is generated as in (7.10).



(a)


K_i
Coded DCT coeff. unicasted to user i
Decrypt
VLC^{-1}
Q^{-1}
K_{i_1}
Coded DCT coeff. multicasted to subgroup U_{i_1}
Decrypt
VLC^{-1}
Q^{-1}
Coded DCT coeff. multicasted to all users
Decrypt
VLC^{-1}
Q^{-1}
K_{multi}
Reconstruct the 8 × 8 DCT block
Positions of embeddable coeff.
IDCT
0
Motion vector
Motion compensation

(b)

FIGURE 7.6. The MPEG-2-based joint fingerprint design and distribution scheme for video-on-demand applications: (a) the fingerprint embedding and distribution process at the server's side; (b) the decoding process at the user's side.

(ii) The compressed bit stream is split into two parts: the first one includes motion vectors, quantization factors, and other side information and is not altered, and the second one contains the coded DCT coefficients and is variable-length decoded.

(iii) Only the values of the DCT coefficients are modified, and the first part of the compressed bit stream is intact. For each DCT coefficient, if it is not embeddable, it is variable-length coded with other nonembeddable DCT coefficients. If it is embeddable, first, it is inversely quantized. If it belongs to $\mathbf{x}^l$, for each subgroup $\mathbf{U}_{i_1,\ldots,i_l} = \{[j_1,\ldots,j_L] : j_1 = i_1,\ldots, j_l = i_l\}$, the corresponding fingerprint component in $\mathbf{s}^l_{i_1,\ldots,i_l}$ is embedded using spread-spectrum embedding, and the resulting fingerprinted coefficients are quantized and variable-length coded with other fingerprinted coefficients in $\mathbf{y}^l_{i_1,\ldots,i_l}$.

(iv) The nonembeddable DCT coefficients are encrypted with key K_{multi} and multicasted to all users, together with the positions of the embeddable coefficients in the 8 × 8 DCT blocks, motion vectors, and other shared information. For $1 \le l < L$, the fingerprinted coefficients in $\mathbf{y}^l_{i_1,\ldots,i_l}$ are encrypted with key $K_{i_1,\ldots,i_l}$ and multicasted to the users in the subgroup $\mathbf{U}_{i_1,\ldots,i_l}$. The fingerprinted coefficients in $\mathbf{y}^L_i$ are encrypted with the ith user's secret key and unicasted to him.

The decoder at the ith user's side is similar to that in the general fingerprint multicast scheme. The difference is that the decoder has to listen to $L+1$ bit streams in the joint fingerprint design and distribution scheme instead of 2 in the general fingerprint multicast scheme.

7.4.3. Addressing the computation constraints

Compared with the general fingerprint multicast scheme, the joint fingerprint design and distribution scheme further reduces the communication cost by multicasting some of the fingerprinted coefficients that are shared by a subgroup of users to them. However, it increases the total number of multicast groups that the sender needs to manage and the number of channels that each receiver downloads data from.

In the general fingerprint multicast scheme shown in Figure 7.2, the sender sets up and manages 1 multicast group, and each user listens to 2 bit streams simultaneously to reconstruct the fingerprinted video sequence. In the joint fingerprint design and distribution scheme, the sender has to set up a multicast group for every subgroup of users represented by a node in the upper $L-1$ levels in the tree. For a tree with $L = 4$ and $[D_1, D_2, D_3, D_4] = [4, 5, 5, 100]$, the total number of multicast groups needed is 125. Also, each user has to listen to $L = 4$ different multicast groups and 1 unicast channel. In practice, the underlying network might not be able to support so many multicast groups simultaneously, and it could be beyond the sender's capability to manage this huge number of multicast groups at one time. It is also possible that the receivers can only listen to a small number of channels simultaneously due to computation and buffer constraints.

To address these computation constraints, we adjust the joint fingerprint design and distribution scheme to minimize the overall communication cost under the computation constraints.

For a fingerprint tree of level L and degrees $[D_1, \dots, D_L]$, if the sender sets up a multicast group for each subgroup of users represented by a node in the upper l levels in the tree, then the total number of multicast groups is $\mathrm{MG}(l) \triangleq 1 + D_1 + \cdots + \prod_{m=1}^{l} D_m$. Also, each user listens to $\mathrm{RB}(l) \triangleq l + 2$ channels. Assume that $\overline{\mathrm{MG}}$ is the maximum number of multicast groups that the network can support and the sender can manage at once, and each receiver can only listen to no more than $\overline{\mathrm{RB}}$ channels. We define $L' \triangleq \max\{l : \mathrm{MG}(l) \le \overline{\mathrm{MG}}, \mathrm{RB}(l) \le \overline{\mathrm{RB}}\}$.

To minimize the communication cost under the computation constraints, we adjust the fingerprint distribution scheme in Section 7.4.2 as follows. Steps 1, 2, and 3 are not changed, and Step 4 is modified to the following.

(i) The coded nonembeddable DCT coefficients are encrypted with key K_{multi} and multicasted to all users, together with the positions of the embeddable coefficients in the 8×8 DCT blocks, motion vectors, and other shared information.

(ii) For each subgroup of users $\mathbf{U}_{i_1,\dots,i_l}$ corresponding to a node $[i_1, \dots, i_l]$ at level $l \le L'$ in the tree, a multicast group is set up and the fingerprinted coefficients in $\mathbf{y}^{l}_{i_1,\dots,i_l}$ are encrypted with key $K_{i_1,\dots,i_l}$ and multicasted to users in $\mathbf{U}_{i_1,\dots,i_l}$.

(iii) For each subgroup of users $\mathbf{U}_{i_1,\dots,i_{L'},\dots,i_m}$, where $L' < m \le L - 1$, there are two possible methods to distribute the fingerprinted coefficients in $\mathbf{y}^m_{i_1,\dots,i_{L'},\dots,i_m}$ to them and the one that has a smaller communication cost is chosen.

(a) First, after encrypting the encoded fingerprinted coefficients in $\mathbf{y}^m_{i_1,\dots,i_{L'},\dots,i_m}$ with key $K_{i_1,\dots,i_m}$, the encrypted bit stream can be multicasted to the users in the subgroup $\mathbf{U}_{i_1,\dots,i_{L'}}$. Since $K_{i_1,\dots,i_m}$ is known only to the users in the subgroup $\mathbf{U}_{i_1,\dots,i_m}$, only they can decrypt the bit stream and reconstruct $\mathbf{y}^m_{i_1,\dots,i_{L'},\dots,i_m}$.

(b) The fingerprinted coefficients in $\mathbf{y}^m_{i_1,\dots,i_{L'},\dots,i_m}$ can also be unicasted to each user in the subgroup $\mathbf{U}_{i_1,\dots,i_m}$ after encryption, the same as in the general fingerprint multicast scheme.

(iv) The fingerprinted coefficients in $\mathbf{y}^L_{i_1,\dots,i_L}$ are encrypted with the ith user's secret key $K^{(i)}$ and unicasted to him.

7.5. Analysis of bandwidth efficiency

To analyze the bandwidth efficiency of the secure fingerprint multicast schemes, we compare their communication costs with that of the pure unicast scheme. In this section, we assume that the fingerprinted copies in all schemes are encoded at the same targeted bit rate.

To be consistent with general Internet routing where hop count is the widely used metric for route-cost calculation [125], we use the hop-based link usage to measure the communication cost and set the cost of all edges to be the same. To transmit a package of length Len^{unit} to a multicast group of size M, it was shown in [107, 125] that the normalized multicast communication cost can be approximated by $C^{\text{unit}}_{\text{multi}}(M)/C^{\text{unit}}_{\text{uni}}(M) = M^{\text{EoS}}$, where $C^{\text{unit}}_{\text{multi}}(M)$ is the communication cost using multicast, $C^{\text{unit}}_{\text{uni}}(M)$ is the average communication cost per user using unicast, and EoS is the economies-of-scale factor. It was shown in [107] that EoS is between 0.66 and 0.7 for realistic networks. In this chapter, we choose $\text{EoS} \approx 0.7$.

7.5.1. "Multicast only" scenario

For the purpose of performance comparison, we consider another special scenario where the video streaming applications require the service provider to prevent outsiders from estimating the video's content, but do not require the traitor tracing capability. In this scenario, we apply the general index mapping to encrypt the DC coefficients in the intrablocks and the motion vectors in interblock; and the AC coefficients are left unchanged and transmitted in clear text. Since the copies that are distributed to different users are the same, the service provider can use a single multicast channel for the distribution of the encrypted bit stream to all users. We call this particular scenario, which does not require the traitor tracing capability and uses multicast channels only the "*multicast only*"; and we compare the communication cost of the "multicast only" with that of the secure fingerprint multicast schemes to illustrate the extra communication overhead introduced by the traitor tracing requirement.

For a given video sequence and a targeted bit rate R, we assume that in the pure unicast scheme, the average size of the compressed bit streams that are unicasted to different users is $\mathrm{Len}^{\mathrm{pu}}$. Define $\mathrm{Len}^{\mathrm{mo}}$ as the length of the bit stream that is multicasted to all users in the "multicast only" scenario. In the pure unicast scheme, the streaming cipher that we applied to the AC coefficients in each fingerprinted copy does not increase the bit rate and keep the compression efficiency unchanged. Consequently, we have $\mathrm{Len}^{\mathrm{mo}} \approx \mathrm{Len}^{\mathrm{pu}}$.

For a multicast group of size M, we further assume that the communication cost of the pure unicast scheme is C^{pu}, and C^{mo} is the communication cost in the "multicast only." We have $C^{\mathrm{pu}}(M) = M \times C_{\mathrm{uni}}^{\mathrm{unit}}(M) \times \mathrm{Len}^{\mathrm{pu}} / \mathrm{Len}^{\mathrm{unit}}$, and $C^{\mathrm{mo}}(M) = C_{\mathrm{multi}}^{\mathrm{unit}}(M) \times \mathrm{Len}^{\mathrm{mo}}/\mathrm{Len}^{\mathrm{unit}}$. We define the communication cost ratio of the "multicast only" as

$$\zeta^{\mathrm{mo}}(M) \triangleq \frac{C^{\mathrm{mo}}(M)}{C^{\mathrm{pu}}(M)} \approx M^{-0.3}, \tag{7.14}$$

and it depends only on the total number of users M.

7.5.2. General fingerprint multicast scheme

For a given video sequence and a targeted bit rate R, we assume that in the general fingerprint multicast scheme, the bit stream that is multicasted to all users is of length $\mathrm{Len}_{\mathrm{multi}}^{fm}$, and the average size of different bit streams that are unicasted to different users is $\mathrm{Len}_{\mathrm{uni}}^{fm}$. For a multicast group of size M, we further assume that the communication cost of the general fingerprint multicast scheme is C^{fm}. We have $C^{fm}(M) = C_{\mathrm{multi}}^{\mathrm{unit}}(M) \times \mathrm{Len}_{\mathrm{multi}}^{fm} / \mathrm{Len}^{\mathrm{unit}} + M \times C_{\mathrm{uni}}^{\mathrm{unit}}(M) \times \mathrm{Len}_{\mathrm{uni}}^{fm} / \mathrm{Len}^{\mathrm{unit}}$. We define the *coding parameter* as $\mathrm{CP} \triangleq (\mathrm{Len}_{\mathrm{multi}}^{fm} + \mathrm{Len}_{\mathrm{uni}}^{fm}) / \mathrm{Len}^{\mathrm{pu}}$, and the *unicast ratio* as $\mathrm{UR} \triangleq \mathrm{Len}_{\mathrm{uni}}^{fm} / (\mathrm{Len}_{\mathrm{multi}}^{fm} + \mathrm{Len}_{\mathrm{uni}}^{fm})$. Then the communication cost ratio of the general fingerprint multicast scheme is

$$\zeta^{fm}(M) \triangleq \frac{C^{fm}(M)}{C^{\mathrm{pu}}(M)} \approx \mathrm{CP}\left\{\mathrm{UR} + (1 - \mathrm{UR})M^{-0.3}\right\}. \tag{7.15}$$

The smaller the communication cost ratio ζ^{fm}, the more efficient the general fingerprint multicast scheme. Given the multicast group size M, the efficiency of the general fingerprint multicast scheme is determined by the coding parameter and the unicast ratio.

Coding parameters. Four factors affect the coding parameters.

(i) For each fingerprinted copy, two different sets of motion vectors and quantization factors are used: the general fingerprint multicast scheme uses those calculated from the original unfingerprinted copy, while the pure unicast scheme uses those calculated from the fingerprinted copy itself. Since the original unfingerprinted copy and the fingerprinted copy are similar to each other, so are both sets of parameters. Therefore, the difference between these two sets of motion vectors and quantization factors has negligible effect on the coding parameters.

(ii) In the general fingerprint multicast scheme, headers and side information have to be inserted in each unicasted bit stream for synchronization. We follow the MPEG-2 standard and observe that this extra overhead consumes no more than 0.014 bit per pixel (bpp) per copy and is much smaller than the targeted bit rate R. Therefore, its effect on the coding parameters can be ignored.

(iii) In the variable-length coding stage, the embeddable and the nonembeddable coefficients are coded together in the pure unicast scheme while they are coded separately in the general fingerprint multicast scheme. Figure 7.7 shows the histograms of the (run length, value) pairs of the "carphone" sequence at $R = 1$ Mbps (1.3 bpp) in both schemes. From Figure 7.7, the (run length, value) pairs generated by the two schemes have approximately the same distribution. Thus, encoding the embeddable and the nonembeddable coefficients together or separately does not affect the coding parameters. The same conclusion can be drawn for other sequences and for other bit rates.

(iv) In the general fingerprint multicast scheme, the positions of the embeddable coefficients have to be encoded and transmitted to the decoders. The encoding procedure is as follows.

(a) For each 8×8 DCT block, first, an 8×8 mask is generated where a bit "0" is assigned to each nonembeddable coefficient and a bit "1" is assigned to each embeddable coefficient. Since DC coefficients are not embedded with fingerprints [24], the mask bit at the DC coefficient's position is skipped and only the 63 mask bits at the AC coefficients' positions are encoded.

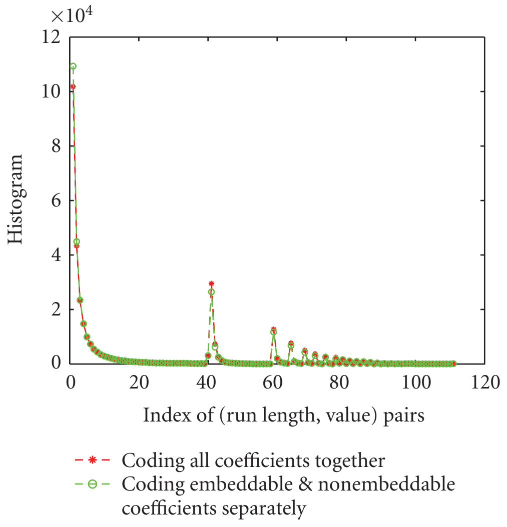
(b) Observing that most of the embeddable coefficients are in the low frequencies, the 63 mask bits are zigzag scanned in the same way as in the JPEG baseline compression.

(c) Run-length coding is applied to the zigzag scanned mask bits followed by Huffman coding.

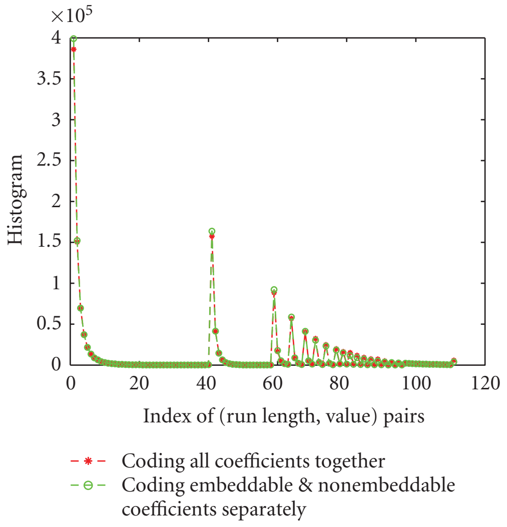
(d) An "end of block" (EoB) marker is inserted after encoding the last mask bit whose value is 1 in the block.

Communication cost ratio. We choose three representative sequences: "Miss America" with large smooth regions, "carphone" that is moderately complicated and "flower" that has large high-frequency coefficients. Figure 7.8a shows the communication cost ratios of the three sequences at $R = 1.3$ bpp.

For M in the range between 1000 and 10000, compared with the pure unicast scheme, the general fingerprint multicast scheme reduces the communication cost by 48% to 84%, depending on the values of M and the characteristics of sequences. Given a sequence and a targeted bit rate R, the performance of the general fingerprint multicast scheme improves as the multicast group size M increases. For example, for the "carphone" sequence at $R = 1.3$ bpp, $\zeta^{fm} = 0.41$ when there are a total of $M = 1000$ users, and it drops to 0.34 when M is increased to 10000. Also, given M, the performance of the general fingerprint multicast scheme depends on the characteristics of video sequences. For sequences with large smooth regions, the embedded fingerprints are shorter. Therefore, fewer bits are needed to encode



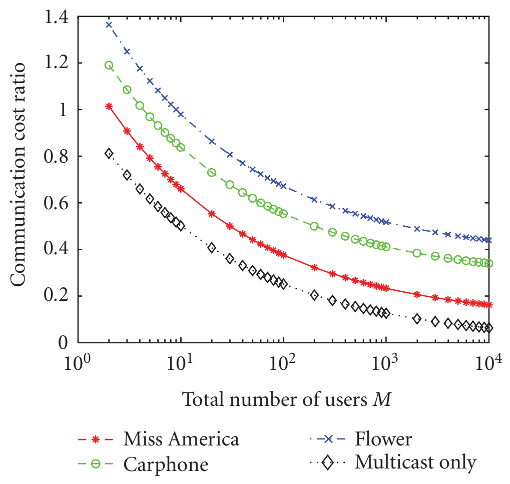
(a)



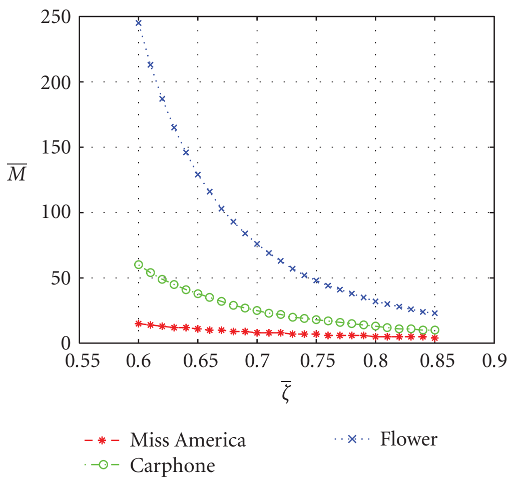
(b)

FIGURE 7.7. Histograms of the (run length, value) pairs of the "carphone" sequence that are variable-length coded in the two schemes. $R = 1$ Mbps. The indices of the (run length, value) pairs are sorted first in the ascending order of the run length, and then in the ascending order of the value: (a) in the intracoded blocks; (b) in the intercoded blocks.

the positions of the embeddable coefficients, and fewer DCT coefficients are transmitted through unicast channels. So the general fingerprint multicast scheme is more efficient. On the contrary, for sequences where the high-frequency band has



(a)



(b)

FIGURE 7.8. Bandwidth efficiency of the general fingerprint multicast scheme at $R = 1.3$ bpp: (a) $\zeta^{fm}(M)$ and $\zeta^{\text{mo}}(M)$ versus M; (b) $\overline{M}$ versus $\overline{\zeta}$.

large energy, more DCT coefficients are embeddable and have to be unicasted. Thus, the general fingerprint multicast scheme is less efficient. When there are a total of $M = 5000$ users, ζ^{fm} is 0.18 for sequence "Miss America" and is 0.46 for sequence "flower."

If we compare the communication cost of the general fingerprint multicast with that of the "multicast only" scenario, enabling traitor tracing in video streaming applications introduces an extra communication overhead of 10% to 40%, depending on the characteristics of video sequences. For sequences with fewer embeddable coefficients, for example, "Miss America," the length of the embedded fingerprints is shorter, and applying digital fingerprinting increases the communication cost by a smaller percentage (around 10%). For sequences that have much more embeddable coefficients, for example, "flower," more DCT coefficients are embedded with unique fingerprints and have to be transmitted through unicast channels, and it increases the communication cost by a larger percentage (approximately 40%).

In addition, the general fingerprint multicast scheme performs worse than the pure unicast scheme when M is small. Therefore, given the coding parameter and the unicast ratio, the pure unicast scheme is preferred when the communication cost ratio ζ is larger than a threshold $\overline{\zeta}$, that is, when M is smaller than $\overline{M}$, where

$$\overline{M} = \left\lceil \left(\frac{1 - \mathrm{UR}}{\overline{\zeta}/\,\mathrm{CP} - \mathrm{UR}} \right)^{10/3} \right\rceil. \tag{7.16}$$

The ceil function $\lceil x \rceil$ returns the minimum integer that is not smaller than x. $\overline{M}$ of different sequences for different $\overline{\zeta}$ are shown in Figure 7.8b. For example, for $\overline{\zeta} = 0.8$ and $R = 1.3$ bpp, $\overline{M}$ is 5 for sequence "Miss America," 13 for "carphone" and 32 for "flower."

7.5.3. Joint fingerprint design and distribution scheme

For a given video sequence and a targeted bit rate R, we assume that in the joint fingerprint design and distribution scheme, the bit stream that is multicasted to all users is of length $\mathrm{Len}_{\mathrm{multi}}^{\mathrm{joint}}$, where $\mathrm{Len}_{\mathrm{multi}}^{\mathrm{joint}} = \mathrm{Len}_{\mathrm{multi}}^{fm}$. For any two nodes $[i_1, \dots, i_l] \neq [j_1, \dots, j_l]$ at level l in the tree, we further assume that the bit streams that are transmitted to the users in the subgroups $\mathbf{U}_{i_1,\dots,i_l}$ and $\mathbf{U}_{j_1,\dots,j_l}$ are approximately of the same length $\mathrm{Len}_l^{\mathrm{joint}}$.

In the joint fingerprint design and distribution scheme, all the fingerprinted coefficients inside one frame are variable-length coded together. Therefore, the histograms of the (run length, value) pairs in the joint fingerprint design and distribution scheme are the same as that in the general fingerprint multicast scheme. If we ignore the impact of the headers/markers that are inserted in each bit stream, we have

$$\begin{gathered} \mathrm{Len}_1^{\mathrm{joint}} + \cdots + \mathrm{Len}_L^{\mathrm{joint}} \approx \mathrm{Len}_{\mathrm{uni}}^{fm}, \\ \frac{\mathrm{Len}_{\mathrm{multi}}^{\mathrm{joint}} + \sum_{l=1}^{L} \mathrm{Len}_l^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{pu}}} \approx \mathrm{CP}. \end{gathered} \tag{7.17}$$

Furthermore, fingerprints at different levels are embedded into the host signal periodically. In the simple example shown in Figure 7.4, the period is 4 seconds. If this period is small compared with the overall length of the video sequence, we can have the approximation that

$$\begin{aligned}\mathrm{Len}_1^{\mathrm{joint}} : \cdots : \mathrm{Len}_L^{\mathrm{joint}} &\approx N_1 : \cdots : N_L,\\ \mathrm{Len}_l^{\mathrm{joint}} &\approx \frac{N_l}{N} \cdot \mathrm{Len}_{\mathrm{uni}}^{fm}.\end{aligned} \tag{7.18}$$

In the joint fingerprint design and distribution scheme, to multicast the nonembeddable DCT coefficients and other shared side information to all users, the communication cost is

$$C_{\mathrm{multi}}^{\mathrm{joint}} = C_{\mathrm{multi}}^{\mathrm{unit}}(M) \times \frac{\mathrm{Len}_{\mathrm{multi}}^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{unit}}}, \tag{7.19}$$

where M is the total number of users. For $l \le L'$, to multicast the fingerprinted coefficients in $\mathbf{y}_{i_1,\ldots,i_l}^l$ to the users in $\mathbf{U}_{i_1,\ldots,i_l}$, the communication cost is

$$C_l^{\mathrm{joint}} = C_{\mathrm{multi}}^{\mathrm{unit}}(M_l) \times \frac{\mathrm{Len}_l^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{unit}}}, \tag{7.20}$$

where $M_l \triangleq \prod_{m=l+1}^{L} D_m$, and there are M/M_l such subgroups. For $L' < l \le L-1$, to distribute the fingerprinted coefficients in $\mathbf{y}_{i_1,\ldots,i_{L'},\ldots,i_l}^l$ to users in $\mathbf{U}_{i_1,\ldots,i_{L'},\ldots,i_l}$, the communication cost is

$$C_l^{\mathrm{joint}} = \min\left\{C_{\mathrm{multi}}^{\mathrm{unit}}(M_{L'}) \times \frac{\mathrm{Len}_l^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{unit}}},\ M_l \cdot C_{\mathrm{uni}}^{\mathrm{unit}}(M_l) \times \frac{\mathrm{Len}_l^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{unit}}}\right\}, \tag{7.21}$$

where the first term is the communication cost if they are multicasted to users in the subgroup $\mathbf{U}_{i_1,\ldots,i_{L'}}$, and the second term is the communication cost if they are unicasted to each user in the subgroup $\mathbf{U}_{i_1,\ldots,i_{L'},\ldots,i_l}$. Finally, the communication cost of distributing the fingerprinted coefficients in $\mathbf{y}_{i_1,\ldots,i_L}^L$ to the ith user is

$$C_L^{\mathrm{joint}} = M \cdot C_{\mathrm{uni}}^{\mathrm{unit}}(M) \times \frac{\mathrm{Len}_L^{\mathrm{joint}}}{\mathrm{Len}^{\mathrm{unit}}}. \tag{7.22}$$

The overall communication cost of the joint fingerprint design and distribution scheme is $C^{\mathrm{joint}} = C_{\mathrm{multi}}^{\mathrm{joint}} + \sum_{l=1}^{L}(M/M_l) \cdot C_l^{\mathrm{joint}}$, and the communication cost ratio $\zeta^{\mathrm{joint}} \triangleq C^{\mathrm{joint}}/C^{\mathrm{pu}}$ is

$$\begin{aligned}\zeta^{\mathrm{joint}} \approx \mathrm{CP}\Bigg\{&(1-\mathrm{UR}) \cdot M^{-0.3} + \mathrm{UR} \cdot \sum_{l=1}^{L'} \frac{N_l}{N} \cdot M_l^{-0.3}\\ &+ \mathrm{UR} \cdot \sum_{l=L'+1}^{L-1} \frac{N_l}{N} \cdot \min\left(\frac{M_{L'}^{0.7}}{M_l}, 1\right) + \mathrm{UR} \cdot \frac{N_L}{N}\Bigg\}.\end{aligned} \tag{7.23}$$

Listed in Table 7.1 are the communication cost ratios of the joint fingerprint design and distribution scheme under different L' for sequence "Miss America,"

TABLE 7.1. The communication cost ratios of the joint fingerprint design and distribution scheme. $L' = 0$ is the general fingerprint multicast scheme. $R = 1.3$ bpp, $p = 0.95$.

	L'	MG	RB	Miss America	Carphone	Flower	Multicast only
$M = 1000, L = 3,$ $\underline{D} = [2, 5, 100],$ $\underline{\rho} = [1/4, 1/4, 1/2]$	0	1	2	0.23	0.41	0.52	0.13
	1	3	3	0.22	0.34	0.43	
	2	13	4	0.20	0.31	0.39	
$M = 5000, L = 4,$ $\underline{D} = [2, 5, 5, 100],$ $\underline{\rho} = [1/6, 1/6, 1/6, 1/2]$	0	1	2	0.18	0.35	0.46	0.08
	1	3	3	0.16	0.30	0.39	
	2	13	4	0.15	0.27	0.35	
	3	65	5	0.14	0.25	0.32	
$M = 10000, L = 4,$ $\underline{D} = [4, 5, 5, 100],$ $\underline{\rho} = [1/6, 1/6, 1/6, 1/2]$	0	1	2	0.16	0.34	0.43	0.06
	1	5	3	0.14	0.28	0.37	
	2	25	4	0.13	0.26	0.33	
	3	125	5	0.13	0.23	0.30	

"carphone," and "flower." $L' = 0$ corresponds to the general fingerprint multicast scheme. We consider three scenarios where the numbers of users are 1000, 5000, and 10000, respectively. The tree structures of the three scenarios are listed in Table 7.1. In the three cases considered, compared with the pure unicast scheme, the joint fingerprint design and distribution scheme reduces the communication cost by 57% to 87%, depending on the total number of users, network and computation constraints, and the characteristics of video sequences.

Given a sequence, the larger the L', that is, the larger the $\overline{\text{MG}}$ and $\overline{\text{RB}}$, the more efficient the joint fingerprint design and distribution scheme. This is because more fingerprinted coefficients can be multicasted. Take the "carphone" sequence with $M = 1000$ users as an example, in the general fingerprint multicast scheme, $\gamma^{fm} = 0.41$. If $L' = 1$, the joint fingerprint design and distribution scheme reduces the communication cost ratio to 0.34, and it is further dropped to 0.31 if $\overline{\text{MG}} \geq 13$ and $\overline{\text{RB}} \geq 4$.

Also, compared with the general fingerprint multicast scheme, the extra communication cost saved by the joint fingerprint design and distribution scheme varies from sequence to sequence. For sequences that have more embeddable coefficients, the joint fingerprint design and distribution improves the bandwidth efficiency by a much larger percentage. For example, for $M = 5000$ and $L' = 2$, compared with the general fingerprint multicast scheme, the joint fingerprint design and distribution scheme further reduces the communication cost by 10% for sequence "flower," while it only further improves the bandwidth efficiency by 3% for sequence "Miss America." However, for sequence "Miss America" with $M = 5000$ users, the general fingerprint multicast scheme has already reduced the communication cost by 82%. Therefore, for sequences with fewer embeddable coefficients, the general fingerprint multicast scheme is recommended to reduce the bandwidth requirement at a low computation cost. The joint fingerprint design and distribution scheme is preferred on sequences with much more embeddable coefficients to achieve higher bandwidth efficiency under network and computation constraints.

Compared with the “multicast only” scenario, the joint fingerprint design and distribution scheme enables the traitor tracing capability by increasing the communication cost by 6% to 30%, depending on the characteristics of the video sequence as well as on the network and computation constraints. Compared with the “multicast only,” for sequences with fewer embeddable coefficients, the joint fingerprint design and distribution scheme increases the communication cost by a smaller percentage (around 6% to 10% for sequence “Miss America”); while for sequences with much more embeddable coefficients, the extra communication overhead introduced is larger (around 24% to 30% for sequence “flower”).

7.6. Robustness of the embedded fingerprints

In this section, we take the group-oriented fingerprint design as an example, and compare the robustness of the embedded fingerprints in different schemes. In the pure unicast scheme and the general fingerprint multicast scheme, we use the CDMA-based fingerprint modulation in order to resist interleaving-based collusion attacks; and in the joint fingerprint design and distribution scheme, the joint TDMA and CDMA fingerprint modulation scheme introduced in Section 7.4.2 is used. In this section, we compare the collusion resistance of the fingerprints embedded using the joint TDMA and CDMA fingerprint modulation scheme with that of the fingerprints embedded using the CDMA-based fingerprint modulation.

7.6.1. Digital fingerprinting system model

At the content owner’s side, spread-spectrum embedding is used to embed fingerprints into the host signal, and the block-based human visual models in [24] are used to control the energy and the imperceptibility of the embedded fingerprints.

During collusion, assume that there are a total of K colluders and S_C is the set containing their indices. In the joint TDMA and CDMA fingerprint modulation, the colluders can apply the interleaving-based collusion attacks, where they divide themselves into L subgroups and $\{S_C(l) \subseteq S_C\}_{l=1,\dots,L}$ contain the indices of the colluders in the L subgroups, respectively. The colluders in subgroup $S_C(l)$ generate the lth part of the colluded copy by $\mathbf{y}^l = g(\{\mathbf{y}_i^l\}_{i\in S_C(l)}) + \mathbf{d}^l$, where $g(\cdot)$ is the collusion function and $\mathbf{d}^l$ is an additive noise to further hinder the detection. In the CDMA-based fingerprint modulation, the colluders cannot distinguish fingerprints at different levels in the tree and cannot apply interleaving-based collusion. Consequently, $S_C(1) = \cdots = S_C(L) = S_C$ for collusion attacks on the CDMA-based fingerprint modulation. Following the discussion in Chapter 4, we consider the averaging collusion only, since the nonlinear collusion can be modeled as the averaging collusion followed by an additive noise.

In the interleaving-based collusion attacks on the joint TDMA and CDMA fingerprint modulation, we consider two types of collusion. In Type I interleaving-based collusion, colluders in subgroup $S_C(L-1)$ and colluders in subgroup $S_C(L)$ are under different branches of the tree and $S_C(L-1) \cap S_C(L) = \emptyset$. The example shown in Figure 7.5 belongs to this type of interleaving-based collusion attacks.

In the Type II interleaving-based collusion, $S_C(L) = S_C$ but $S_C(l) \subset S_C$ for some $l < L$. Take the fingerprint tree in Figure 7.3 as an example, if users 1, 2, 4, and 7 are the colluders, and if the colluders choose $S_C(1) = \{7\}$, $S_C(2) = \{4\}$, and $S_C(3) = \{1, 2, 4, 7\}$, then this is a Type II interleaving-based collusion attack.

At the detector's side, we consider a nonblind detection scenario, where the host signal $\mathbf{x}$ is available to the detector and is first removed from the colluded copy $\mathbf{y}$ before fingerprint detection and colluder identification. Given the extracted fingerprint $\mathbf{w}$, the detector applies the detection process as in Section 7.4.2.

7.6.2. Performance criteria

To measure the robustness of the joint TDMA and CDMA fingerprint modulation scheme against collusion attacks, we adopt the commonly used criteria in the literature: the probability of capturing at least one colluder (P_d), and the probability of accusing at least one innocent user (P_{fp}).

We assume that the colluders apply fair collusion, that is, all colluders share the same risk and are equally likely to be detected. Assume that A and B are two nonoverlapping subgroups of colluders, and $S_C(A)$ and $S_C(B)$ are the sets containing the indices of the colluders in A and B, respectively. $S_C(A) \cap S_C(B) = \emptyset$, and we define the fairness parameter $\mathrm{FP}(S_C(A), S_C(B))$ as

$$\mathrm{FP}\left(S_C(A), S_C(B)\right) \triangleq \frac{F_d\left(S_C(A)\right)}{F_d\left(S_C(B)\right)}, \tag{7.24}$$

where

$$\begin{aligned} F_d\left(S_C(A)\right) &= \frac{\sum_{i \in S_C(A)} I\left[i \in \hat{S}_C\right]}{\left|S_C(A)\right|}, \\ F_d\left(S_C(B)\right) &= \frac{\sum_{i \in S_C(B)} I\left[i \in \hat{S}_C\right]}{\left|S_C(B)\right|}. \end{aligned} \tag{7.25}$$

In the above equations, $I[\cdot]$ is the indication function, $|S_C(A)|$ and $|S_C(B)|$ are the numbers of colluders in $S_C(A)$ and $S_C(B)$, respectively, and $\hat{S}_C$ is the estimated colluder set output by the detector. If $\mathrm{FP}(S_C(A), S_C(B)) \approx 1$ for any $S_C(A) \cap S_C(B) = \emptyset$, then the collusion attack is fair and all colluders are equally likely to be detected. If $\mathrm{FP}(S_C(A), S_C(B)) \gg 1$ or $\mathrm{FP}(S_C(A), S_C(B)) \ll 1$ for some pair of $(S_C(A), S_C(B))$, some colluders are more likely to be detected than others and the collusion attack is not fair.

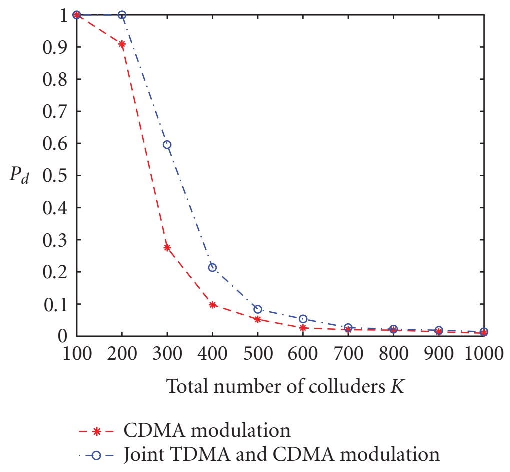
7.6.3. Comparison of collusion resistance

Resistance-to-interleaving-based collusion attacks. Our simulation is set up as follows. For the tested video sequences, the number of embeddable coefficients is in the order of 10^6 per second. So we choose $N = 10^6$ and assume that there are a total of $M = 10^4$ users. Following the group-oriented fingerprint design in [120], we consider a symmetric tree structure with $L = 4$ levels, $[D_1, D_2, D_3, D_4] = [4, 5, 5, 100]$, and $[\rho_1, \rho_2, \rho_3, \rho_4] = [1/6, 1/6, 1/6, 1/2]$. In our simulations, the basis fingerprints $\{\mathbf{a}\}$ in the fingerprint tree follow Gaussian distribution $\mathcal{N}(0, \sigma_W^2)$ with $\sigma_W^2 = 1/9$. In the joint TDMA and CDMA fingerprint modulation, for simplicity, we let $p_{2,2} = \cdots = p_{L,L} = p$ for the matrix $\mathbf{P}^{\text{Joint}}$ in (7.4) and choose $p = 0.95$ for the above fingerprint tree structure. A smaller value of p should be used if L is larger or the total number of nodes at the upper $L - 1$ levels in the tree is larger.

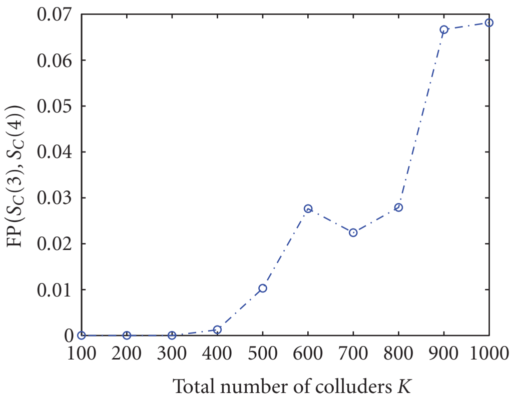
At the attackers' side, we consider the most effective collusion pattern on the group-oriented fingerprint design, where colluders are from all the 100 subgroups at level 3. We assume that each of the 100 subgroups has the same number of colluders. As an example of the interleaving-based collusion attacks, we choose different subgroups of colluders as $SC_1 = \{i = [i_1, i_2, i_3, i_4] \in SC : i_1 = 1\}$, $SC_2 = \{i = [i_1, i_2, i_3, i_4] \in SC : i_1 = 2\}$, and $SC_3 = \{i = [i_1, i_2, i_3, i_4] \in SC : i_1 = 3\}$. In the Type I interleaving-based collusion attacks, we choose $SC_4 = SC \setminus SC_3$.[5] In the Type II interleaving-based collusion attacks, $SC_4 = SC$. In the CDMA-based fingerprint modulation scheme, similarly, we assume that colluders are from all the 100 subgroups at level 3 in the tree, and each subgroup at level 3 in the tree has equal number of colluders. In the CDMA-based fingerprint modulation, the colluders cannot distinguish fingerprints at different levels, and they apply the *pure averaging collusion attack*, where $SC_1 = \cdots = SC_L = SC$. In addition to the multiuser collusion, we assume that the colluders also add an additive noise $\mathbf{d}$ to further hinder the detection. For simplicity, we assume that the additive noise $\mathbf{d}$ is i.i.d. and follows distribution $\mathcal{N}(0, \sigma_d^2)$. In our simulations, we let $\sigma_d^2 = 2\sigma_W^2$, where σ_W^2 is the variance of the embedded fingerprints, and other values of σ_n^2 give the same trend and are not shown here.

Figure 7.9 shows the simulation results of the Type I interleaving-based collusion. In Figure 7.9a, given the total number of colluders K, we compare P_d of the joint TDMA and CDMA fingerprint modulation under the interleaving-based collusion attacks with that of the CDMA-based fingerprint modulation scheme under the pure averaging collusion attacks. From Figure 7.9a, the performance of the joint TDMA and CDMA fingerprint modulation under the Type I interleaving-based collusion is even better than the CDMA-based fingerprint modulation under the pure averaging collusion. This is because, in the pure averaging collusion, all colluders contribute their leaf-node fingerprints that uniquely identify each attacker; while in the Type I interleaving-based collusion, the number of colluders who contribute their leaf-node fingerprints is $|S_C(4)|$ and it is smaller than $|S_C|$. Therefore, compared with the pure averaging collusion, the energy of each leaf-node fingerprint in the interleaving-based collusion is reduced by a smaller ratio, and therefore, it gives the detector more information about the colluders' identities. Figure 7.9b plots the fairness parameter in the Type I interleaving-based collusion in the joint TDMA and CDMA fingerprint modulation, and we observe that $\text{FP}(S_C(3), S_C(4)) \ll 1$. Consequently, the colluders in the subgroup $S_C(4)$ have a

[5]For two sets A and B, where $B \subseteq A$, $A \setminus B \triangleq \{i : i \in A, i \notin B\}$.



(a)

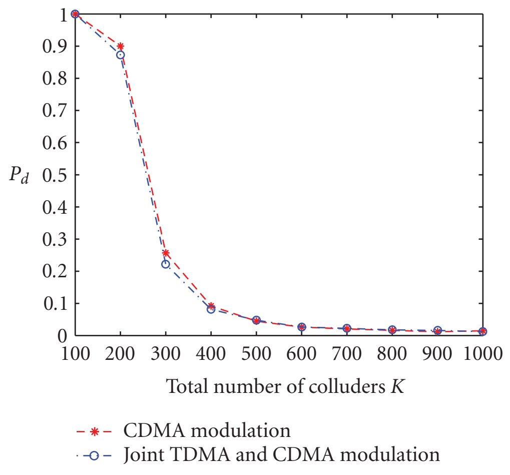


(b)

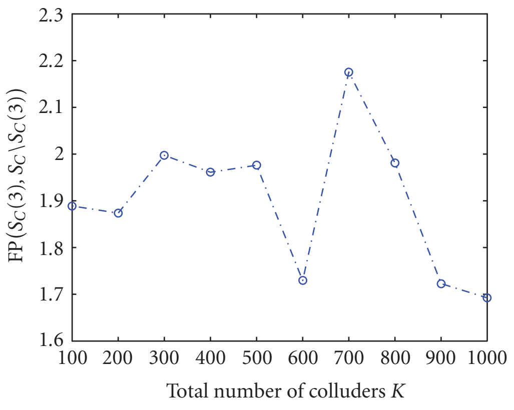
FIGURE 7.9. (a) P_d and (b) $\mathrm{FP}(S_C(L-1), S_C(L))$ the joint TDMA and CDMA fingerprint modulation scheme against the Type I interleaving-based collusion attacks. $L = 4$, $[D_1, D_2, D_3, D_4] = [4, 5, 5, 100]$, and $[\rho_1, \rho_2, \rho_3, \rho_4] = [1/6, 1/6, 1/6, 1/2]$. $N = 10^6$, $\sigma_d^2 = 2\sigma_W^2$, and $P_{fp} = 10^{-2}$. $p = 0.95$. In this type of interleaving-based collusion, $S_C(1) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 1\}$, $S_C(2) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 2\}$, $S_C(3) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 3\}$, and $S_C(4) = S_C \setminus S_C(3)$.

much larger probability to be detected than colluders in the subgroup $S_C(3)$, and this type of interleaving collusion is not fair.

Figure 7.10 shows the simulation results of the Type II interleaving-based collusion. Figure 7.10a compares P_d of the joint TDMA and CDMA fingerprint modulation under the Type II interleaving-based collusion attacks with that of the CDMA-based fingerprint modulation scheme under the pure averaging collusion attacks. From Figure 7.10a, these two have similar performance. Figure 7.10b shows the fairness parameters of the Type II interleaving-based collusion, and we



(a)



(b)

FIGURE 7.10. (a) P_d and (b) $\mathrm{FP}(S_C(L-1),\ S_C \setminus S_C(L-1))$ of the joint TDMA and CDMA fingerprint modulation scheme against interleaving-based collusion attacks. $L = 4$, $[D_1, D_2, D_3, D_4] = [4, 5, 5, 100]$, and $[\rho_1, \rho_2, \rho_3, \rho_4] = [1/6, 1/6, 1/6, 1/2]$. $N = 10^6$, $\sigma_d^2 = 2\sigma_W^2$, and $P_{fp} = 10^{-2}$. $p = 0.95$. In this type of interleaving-based collusion, $S_C(1) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 1\}$, $S_C(2) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 2\}$, $S_C(3) = \{[i_1, i_2, i_3, i_4] \in S_C : i_1 = 3\}$, and $S_C(4) = S_C$.

find that $\mathrm{FP}(S_C(3), S_C \setminus S_C(3)) \approx 1.9$. Consequently, for colluders in the subgroup $S_C(3)$, their probability of being detected is approximately twice of the other colluders' risk of being caught, and therefore, this Type II interleaving collusion is not a fair collusion either.

To summarize, the performance of the joint TDMA and CDMA fingerprint modulation scheme under the interleaving-based collusion attacks is approximately the same as, and may be even better than, that of the CDMA fingerprint modulation scheme under the pure averaging collusion attacks. Furthermore, we have shown that neither of the two types of interleaving-based collusion attacks are fair in the joint TDMA and CDMA fingerprint modulation scheme, and some colluders are more likely to be captured than others. Consequently, to guarantee the absolute fairness of the collusion attacks, the colluders cannot use the interleaving-based collusion attacks in the joint TDMA and CDMA fingerprint modulation.

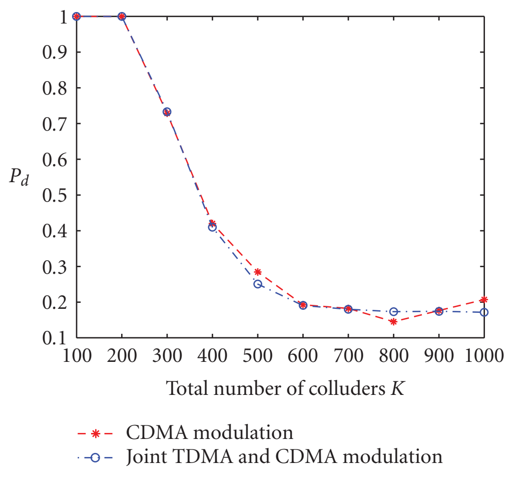
Resistance to the pure averaging collusion attacks. In this section, we study the detection performance of the joint TDMA and CDMA fingerprint modulation under the pure averaging collusion attacks, where $S_C(1) = S_C(2) = \cdots = S_C(L) = S_C$. We compare the detection performance of the joint TDMA and CDMA fingerprint modulation with that of the CDMA fingerprint modulation. In both fingerprint modulation schemes, all colluders have equal probability of detection under this type of collusion, and the pure averaging attacks are fair collusion attacks. The simulation setup is the same as in the previous section and Figure 7.11 shows the simulation results. We consider two possible collusion patterns. In the first one, we assume that one region at level 1 is guilty and it has two guilty subregions at level 2. For each of the two guilty regions at level 2, we assume that all its five children at level 3 are guilty and colluders are present in 10 out of 100 subgroups at level 3. This collusion pattern corresponds to the case where the fingerprint tree matches the hierarchical relationship among users. In the second one, we assume that all the 100 subgroups at level 3 are guilty, and this collusion pattern happens when the fingerprint tree does not reflect the real hierarchical relationship among users. We assume that each guilty subgroup at level 3 has the same number of colluders in both collusion patterns.

From Figure 7.11, the two fingerprint modulation schemes have approximately the same performance under the pure averaging collusion attacks, and both perform better when the fingerprint tree design matches the collusion patterns and the colluders are present in fewer subgroups in the tree.

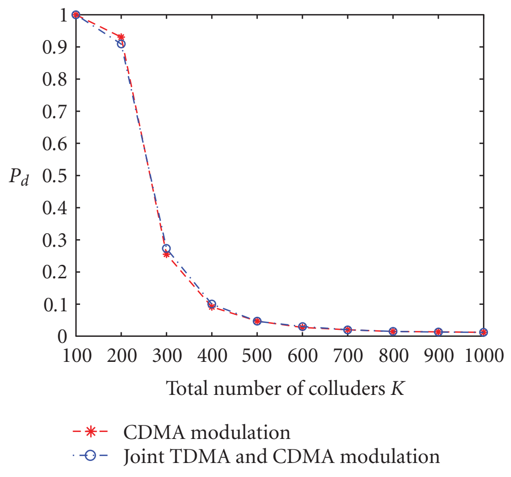
To summarize, under the constraint that all colluders share the same risk and have equal probability of detection, the joint TDMA and CDMA fingerprint modulation has approximately identical performance as the CDMA-based fingerprint modulation, and the embedded fingerprints in the three secure fingerprint distribution schemes have the same collusion resistance.

7.7. Fingerprint drift compensation

In both the general fingerprint multicast scheme and the joint fingerprint design and distribution scheme, the video encoder and the decoder use the reconstructed *unfingerprinted* and *fingerprinted* copies, respectively, as references for motion compensation. The difference, which is the embedded fingerprint, will propagate to the next frame. Fingerprints from different frames will accumulate and cause the quality degradation of the reconstructed frames at the decoder's side. A drift compensation signal, which is the embedded fingerprint in the reference frame(s) with motion, has to be transmitted to each user. It contains confidential information of the embedded fingerprint in the reference frame(s) and is unique to each user. Therefore, it has to be transmitted seamlessly with the host signal to



(a)



(b)

FIGURE 7.11. P_d of the joint TDMA and CDMA fingerprint modulation scheme under the pure averaging collusion. $L = 4$, $[D_1, D_2, D_3, D_4] = [4, 5, 5, 100]$, and $[\rho_1, \rho_2, \rho_3, \rho_4] = [1/6, 1/6, 1/6, 1/2]$. $N = 10^6$, $\sigma_d^2 = 2\sigma_W^2$, and $P_{fp} = 10^{-2}$. $p = 0.95$. (a) Colluders are from 10 subgroups at level 3 in the tree. (b) Colluders are from all the 100 subgroups at level 3 in the tree.

the decoder through unicast channels. Since the embedded fingerprint propagates to both the embeddable coefficients and the nonembeddable ones, fully compensating the drifted fingerprint will significantly increase the communication cost.

To reduce the communication overhead introduced by full-drift compensation, we compensate the drifted fingerprint that propagates to the embeddable

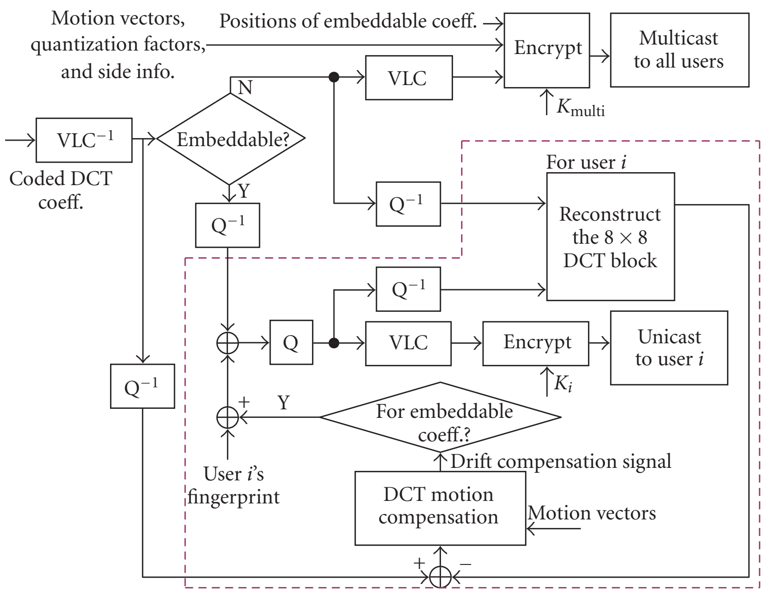


Figure 7.12. The fingerprint drift compensation scheme in the general fingerprint multicast for VoD applications.

coefficients only and ignore the rest. Shown in Figure 7.12 is the fingerprint drift compensation scheme in the general fingerprint multicast scheme for video-on-demand applications. The one in the joint fingerprint design and distribution scheme is similar and omitted. The calculation of the drift compensation signal is similar to that in [126]. Step 3 in the fingerprint embedding and distribution process is modified as follows. For each DCT coefficient, if it is not embeddable, it is variable-length coded with other nonembeddable coefficients. Otherwise, first, it is inversely quantized. *Then for each user, the corresponding fingerprint component is embedded, the corresponding drift compensation component is added, and the resulting fingerprinted and compensated coefficient is quantized and variable-length coded with other fingerprinted and compensated coefficients.*

In Table 7.2, we compare the quality of the reconstructed sequences at the decoder's side in three scenarios: PSNR_f is the average PSNR of the reconstructed frames with full drift compensation; PSNR_n is the average PSNR of the reconstructed frames without drift compensation; and PSNR_p is the average PSNR of the reconstructed frames in the proposed drift compensation scheme. Compared with the reconstructed frames with full-drift compensation, the reconstructed frames without drift compensation have an average of $1.5 \sim 2$ dB loss in PSNR, and those using the proposed drift compensation have an average of 0.5 dB loss in PSNR. Therefore, the proposed drift compensation scheme improves the quality of the reconstructed frames at the decoder's side without extra communication overhead.

Table 7.2. Perceptual quality of the reconstructed frames at the decoder's side at bit rate $R = 1.3$ bpp.

Sequence	PSNR_f (dB)	PSNR_n (dB)	PSNR_p (dB)
Miss America	44.89	42.73	44.31
Carphone	40.45	38.05	39.88
Flower	31.53	30.01	30.92

7.8. Chapter summary

In this chapter, we have investigated secure fingerprint multicast for video streaming applications that require strong traitor tracing capability, and have developed two schemes: the general fingerprint multicast scheme and the group-oriented joint fingerprint design and distribution scheme. We have analyzed their performance, including the communication cost and the collusion resistance, and studied the tradeoff between bandwidth efficiency and computation complexity. We have also introduced a fingerprint drift compensation scheme to improve the perceptual quality of the reconstructed sequences at the decoder's side without extra communication cost.

We first developed the general fingerprint multicast scheme that can be used with most spread-spectrum-embedding-based fingerprinting systems. Compared with the pure unicast scheme, it reduces the communication cost by 48% to 84%, depending on the total number of users and the characteristics of sequences. To further reduce the bandwidth requirement, we utilized the tree structure of the fingerprint design and presented the group-oriented joint fingerprint design and distribution scheme. Compared with the pure unicast scheme, it reduces the bandwidth requirement by 57% to 87%, depending on the number of users, the characteristics of sequences, and network and computation constraints. We have also shown that under the constraints that all colluders have equal probability of detection, the embedded fingerprints in these two schemes have approximately the same robustness against collusion attacks.

If we compare the three distribution schemes; the pure unicast scheme, the general fingerprint multicast scheme, and the joint fingerprint design and distribution scheme, the pure unicast scheme is preferred when there are only a few users in the system (e.g., around ten or twenty users); and the other two should be used when there are a large number of users (e.g., thousands of users). Compared with the general fingerprint multicast scheme, the joint fingerprint design and distribution scheme further improves the bandwidth efficiency by increasing the computation complexity of the systems. Therefore, for sequences that have fewer embeddable coefficients, for example, "Miss America," the general fingerprint multicast scheme is preferred to achieve the bandwidth efficiency at a low computational cost. For sequences with much more embeddable coefficients, for example, "flower," the joint fingerprint design and distribution scheme is recommended to minimize the communication cost under network and computation constraints.

Finally, we studied the perceptual quality of the reconstructed sequences at the receiver's side. We have shown that the proposed fingerprint drift compensation scheme improves PSNR of the reconstructed frames by an average of $1 \sim 1.5$ dB without increasing the communication cost.

8 Fingerprinting curves

This chapter presents a new data hiding method for curves. The proposed algorithm parameterizes a curve using the B-spline model and adds a spread-spectrum sequence to the coordinates of the B-spline control points. In order to achieve robust fingerprint detection, an iterative alignment-minimization algorithm is proposed to perform curve registration and to deal with the nonuniqueness of B-spline control points. We demonstrate through experiments the robustness of the proposed data hiding algorithm against various attacks such as collusion, cropping, geometric transformations, vector/raster-raster/vector conversions, printing and scanning, and some of their combinations. We also show the feasibility of our method for fingerprinting topographic maps as well as writings and drawings.

8.1. Introduction

Maps represent geospatial information ubiquitous in government, military, intelligence, and commercial operations. The traditional way of protecting a map from unauthorized copying and distribution is to place deliberate errors in the map, such as spelling "Nelson Road" as "Nelsen Road," bending a road in a wrong way, and/or placing a nonexisting pond. If an unauthorized user has a map containing basically the same set of errors, this is a strong piece of evidence on piracy that can be presented in court. One of the classic lawsuits is the Rockford Map Pub. versus Dir. Service Co. of Colorado, 768 F.2d 145, 147 (7th Cir., 1985), where phony middle initials of names in a map spelled out "Rockford Map Inc." when read from the top of the map to the bottom and thus copyright infringement was found. However, the traditional protection methods alter the geospatial meanings conveyed by a map, which can cause serious problems in critical government, military, intelligence, and commercial operations that require high-fidelity geospatial information. Furthermore, in the situations where distinct errors serve as fingerprints to trace individual copies, such deliberately placed errors can be easily identified and removed by computer programs after multiple copies of a map are brought to the digital domain. All these limitations of the traditional methods prompt for a modern way of map protection that can be more effective and less intrusive.

Curves are one of the major components appearing in maps as well as drawings and signatures. A huge amount of curve-based documents are being brought to the digital domain owing to the popularity of scanning devices and pen-based devices (such as TabletPC). Digital maps and drawings are also generated directly by various computer programs such as map-making software and CAD systems. Having the capability of hiding digital watermarks or other secondary data in curves can facilitate digital rights management of important documents in government, military, and commercial operations. For example, trace-and-track capabilities can be provided through invisibly embedding a unique ID, referred to as a *digital fingerprint*, to each copy of a document before distributing it to a user [17, 50]. In this chapter, we present a new, robust data hiding technique for curves and investigate its feasibility for fingerprinting maps.

As a forensic mechanism to deter information leakage and to trace traitors, digital fingerprints must be difficult to remove. For maps and other visual documents, the fingerprint has to be embedded in a robust way against common processing and malicious attacks. Some examples include collusion, where several users combine information from several copies, which have the same content but different fingerprints, to generate a new copy in which the original fingerprints are removed or attenuated [50]; various geometric transformations, such as rotation, scaling, and translation (RST); and D/A-A/D conversions such as printing and scanning. On the other hand, the fingerprint must be embedded in a visually nonintrusive way without changing the geographical and/or visual meanings conveyed by the document. This is because the intrusive changes may have serious consequences in critical military and commercial operations, for example, when inaccurate data are given to troops or fed into navigation systems.

There are a very limited amount of existing works on watermarking maps [127], and few exploit curve features or address fingerprinting issues. A text-based geometric normalization method was proposed in [128], whereby text labels are first used to normalize the orientation and scale of the map image, and conventional robust watermarking algorithms for grayscale images are then applied. As maps can be represented as a set of vectors, two related works on watermarking vector graphics perturb vertices through Fourier descriptors of polygonal lines [129] or spectral analysis of mesh models [130] to embed copyright marks. The embedding in [129] introduces visible distortions, as shown by the experimental results in the paper. The watermarking approach in [130] has high complexity resulting from the mesh spectral analysis, and cannot be easily applied to maps beyond urban areas, where curves become essential components in mapping a vast amount of land and underwater terrains. Since curve-based documents can also be represented as binary bitmap images (known as the raster representation), we expand the literature survey to data embedding works for general binary images. The data hiding algorithm in [131] enforces the ratio of black versus white pixels in a block to be larger or smaller than 1, and flippable pixels are defined and used in [132, 133] to enforce specific block-based relationship to embed data. The fragility of these embedding techniques and the dependence on precise sampling of pixels for correct decoding pose challenges in surviving geometric transformations, printing and scanning, and malicious removal in fingerprinting applications. A few other works embed information in dithered images by manipulating the dithering patterns, in fax images by manipulating the run-length [134], and in textual images by changing the line spacing and character spacing [135]. These works cannot be easily extended to robustly mark curve-based documents.

Several watermarking algorithms on graphic data explore compact representations of curves or surfaces for data embedding, such as through the nonuniform rational B-spline (NURBS) model. The work in [136] concerns how to embed data in NURBS curves and surfaces without changing the shape or increasing the number of B-spline parameters. The approach demonstrated in the paper relies on reparameterizing a curve or surface using a rational linear function that has an offset determined by the bits to be embedded. The embedded data are fragile and can be removed by perturbing the NURBS parameters or another round of reparameterization. The work in [137, 138] focuses on 3D surfaces and extracts NURBS features from a 3D surface to form a few 2D arrays. Through DCT-domain embedding in these virtual images, a watermark is embedded into the 3D NURBS surfaces. Registration techniques for 3D NURBS surfaces such as [139] may be employed to facilitate the alignment of the test surfaces with the original reference surface prior to watermark detection. These prior works provide enlightening analogy for watermarking 2D curves in the B-spline feature domain. As most existing exploration either has limited robustness or targets mainly at 3D surfaces, there are few discussions on robust fingerprinting of curves. To our best knowledge, no existing watermarking work has demonstrated the robustness under curve format conversion and D/A-A/D conversion, or addressed collusion resistance and traitor tracing issues for curves.

In this chapter, we propose a robust curve watermarking method and apply it to fingerprinting maps without interfering with the geospatial meanings conveyed by the map [140, 141, 142]. We select B-spline control points of curves as the feature domain and add mutually independent, noise-like sequences as digital fingerprints to the coordinates of the control points. A proper set of B-spline control points forms a compact collection of salient features representing the shape of the curve, which is analogous to the perceptually significant components in the continuous-tone images [23]. The shape of curves is also invariant to such challenging attacks as printing and scanning and the vector/raster-raster/vector conversions. The additive spread-spectrum embedding and the corresponding correlation-based detection generally provide a good tradeoff between imperceptibility and robustness [23], especially when the original host signal is available to the detector as in most of the fingerprinting applications [50]. To determine which fingerprint sequence(s) is(are) present in a test curve, registration with the original unmarked curve is an indispensable preprocessing step. B-splines have invariance to affine transformations in that the affine transformation of a curve is equivalent to the affine transformation of its control points. This affine invariance property of B-splines can facilitate automatic curve registration. Meanwhile, as a curve can be approximated by different sets of B-spline control points, we propose an iterative alignment-minimization (IAM) algorithm to simultaneously align the curves and identify the corresponding control points with high precision. Through the B-spline-based data hiding plus the IAM algorithm for robust fingerprint detection, our curve watermarking technique can sustain a number of challenging attacks, such as collusion, cropping, geometric transformations, vector/raster-raster/vector conversions, and printing and scanning, and is therefore capable of building collusion-resistant fingerprinting for maps and other curve-based documents.

The chapter is organized as follows. Section 8.2 discusses the feature domain in which data hiding is performed and presents the basic embedding and detection algorithms with experimental results on marking simple curves. Section 8.3 details the proposed iterative alignment-minimization algorithm for the fingerprint detection and analyzes its robustness. Experimental results on fingerprinting topographic maps are presented in Section 8.4 to demonstrate the robustness of our method against a number of distortions and attacks. Finally, chapter summary is given in Section 8.5.

8.2. Basic embedding and detection

Our proposed algorithm employs B-spline control points of curves as the feature domain and adopts spread-spectrum embedding [23] for robustly watermarking the coordinates of the control points. The fingerprints for different users are approximately orthogonal and generated by pseudorandom number generators with different keys, and the detection is based on correlation statistics. In the following subsections, we explain the main steps of the basic embedding and detection method in detail.

8.2.1. Feature extraction

A number of approaches have been proposed for curve modeling, including using chain codes, Fourier descriptors, autoregressive models, and B-splines [143]. Among them, B-splines are particularly attractive and have been extensively used in computer-aided design and computer graphics. This is mainly because the B-spline model provides a bounded and continuous approximation of a curve with excellent local shape control and is invariant to affine transformations [144]. These advantages also lead to our choosing B-splines as the feature domain for embedding data in curves.

B-splines are piecewise polynomial functions that provide local approximations of curves using a small number of parameters known as the *control points* [143]. Let $\{\mathbf{p}(t)\}$ denote a curve, where $\mathbf{p}(t) = (p_x(t), p_y(t))$ and t is a continuous indexing parameter. Its B-spline approximation $\{\mathbf{p}^{[B]}(t)\}$ can be written as

$$\mathbf{p}^{[B]}(t) = \sum_{i=0}^{n} \mathbf{c}_i B_{i,k}(t), \tag{8.1}$$

where t ranges from 0 to $n-1$, $\mathbf{c}_i = (c_{x_i}, c_{y_i})$ is the ith control point ($i = 0, 1, \dots, n$), and $B_{i,k}(t)$ is the weight of the ith control point for the point $\mathbf{p}^{[B]}(t)$ and is known as the kth order B-spline blending function. $B_{i,k}(t)$ is recursively defined as

$$\begin{aligned} B_{i,1}(t) &= \begin{cases} 1, & t_i \le t < t_{i+1}, \\ 0, & \text{otherwise}, \end{cases} \\ B_{i,k}(t) &= \frac{(t - t_i)B_{i,k-1}(t)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - t)B_{i+1,k-1}(t)}{t_{i+k} - t_{i+1}}, \quad k = 2, 3, \dots, \end{aligned} \tag{8.2}$$

where $\{t_i\}$ are parameters known as *knots* and represent locations where the B-spline functions are tied together [143]. The placement of knots controls the form of B-spline functions and in turn the control points.

As a compact representation, the number of B-spline control points necessary to represent a curve at a desired precision can be much smaller than the number of points that can be sampled from the curve. Thus, given a set of samples on the curve, finding a smaller set of control points for its B-spline approximation that minimizes the approximation error to the original curve can be formulated as a least-squares problem. Coordinates of the $m + 1$ samples on the curve can be represented as an $(m + 1) \times 2$ matrix

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_m \end{bmatrix} = \begin{bmatrix} p_{x_0} & p_{y_0} \\ p_{x_1} & p_{y_1} \\ \vdots & \vdots \\ p_{x_m} & p_{y_m} \end{bmatrix} \triangleq (\mathbf{p}_x, \mathbf{p}_y). \tag{8.3}$$

The indexing values of the B-spline blending functions corresponding to these $m + 1$ samples are $t = s_0, s_1, s_2, \dots, s_m$, where $s_0 < s_1 < s_2 < \cdots < s_m$. Further, let $\mathbf{C}$ represent a set of $n + 1$ control points

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_n \end{bmatrix} = \begin{bmatrix} c_{x_0} & c_{y_0} \\ c_{x_1} & c_{y_1} \\ \vdots & \vdots \\ c_{x_n} & c_{y_n} \end{bmatrix} \triangleq (\mathbf{c}_x, \mathbf{c}_y). \tag{8.4}$$

Then we can write the least-squares problem with its solution as

$$\min_{\mathbf{C}} \|\mathbf{BC} - \mathbf{P}\|^2 \Longrightarrow \mathbf{C} = (\mathbf{B}^{\mathrm{T}}\mathbf{B})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{P} = \mathbf{B}^{\dagger}\mathbf{P}, \tag{8.5}$$

where $\{\mathbf{B}\}_{ji}$ is the value of the kth-order B-spline blending function $B_{i,k}(t)$ in (8.2)

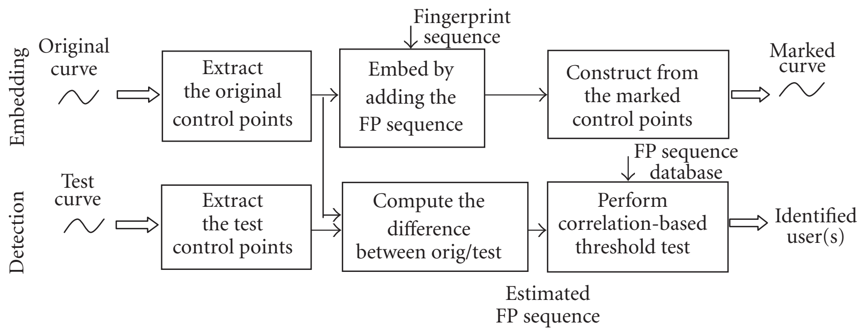

Embedding
Original curve
Extract the original control points
Fingerprint sequence
Embed by adding the FP sequence
Construct from the marked control points
Marked curve
FP sequence database
Detection
Test curve
Extract the test control points
Compute the difference between orig/test
Perform correlation-based threshold test
Identified user(s)
Estimated FP sequence

FIGURE 8.1. The basic embedding and detection process of data hiding in curves.

evaluated at $t = s_j$ for the ith control point and † denotes the pseudoinverse of a matrix. Due to the natural decoupling of the x and y coordinates in the B-spline representation, we can solve the problem separately along each of the two coordinates as

$$\begin{aligned} \min_{\mathbf{c}_x} \|\mathbf{B}\mathbf{c}_x - \mathbf{p}_x\|^2 & \qquad \mathbf{c}_x = \mathbf{B}^\dagger \mathbf{p}_x \\ & \Longrightarrow \\ \min_{\mathbf{c}_y} \|\mathbf{B}\mathbf{c}_y - \mathbf{p}_y\|^2 & \qquad \mathbf{c}_y = \mathbf{B}^\dagger \mathbf{p}_y. \end{aligned} \tag{8.6}$$

8.2.2. Fingerprinting in the control-point domain

The control points of a curve are analogous to the perceptually significant components of a continuous-tone image [23] in that they form a compact set of salient features for curves. In such a feature domain, we apply spread-spectrum embedding and correlation-based detection, as shown in Figure 8.1.

In the embedding, we use mutually independent, noise-like sequences as digital fingerprints to represent different users/IDs for trace-and-track purposes. As each of the $n + 1$ control points has two coordinate values x and y, the overall length of the fingerprint sequence is $2(n + 1)$. To apply spread-spectrum embedding on a curve, we add a scaled version of the fingerprint sequence $(\mathbf{w}_x, \mathbf{w}_y)$ to the coordinates of a set of control points obtained from the previous subsection. This results in a set of watermarked control points $(\mathbf{c}'_x, \mathbf{c}'_y)$ with

$$\begin{aligned} \mathbf{c}'_x &= \mathbf{c}_x + \alpha \mathbf{w}_x, \\ \mathbf{c}'_y &= \mathbf{c}_y + \alpha \mathbf{w}_y, \end{aligned} \tag{8.7}$$

where α is a scaling factor adjusting the fingerprint strength. A watermarked curve can then be constructed by the B-spline synthesis equation (8.1) using these watermarked control points.

To determine which fingerprint sequence(s) is(are) present in a test curve, we first need to perform registration using the original unmarked curve that is commonly available to a detector in fingerprinting applications. After registration, control points $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$ are extracted from the test curve. The accurate registration and correct extraction of control points are crucial to the detection of fingerprints, which will be detailed in Section 8.3. Assuming we have the set of sample points given by $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y) = (\mathbf{B}(\mathbf{c}_x+\alpha\mathbf{w}_x), \mathbf{B}(\mathbf{c}_y+\alpha\mathbf{w}_y))$, we can extract the test control points $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$ from $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ using (8.6). After getting $(\tilde{\mathbf{c}}_x, \tilde{\mathbf{c}}_y)$, we compute the difference between the coordinates of the test and the original control points to arrive at an estimated fingerprint sequence

$$\begin{aligned} \tilde{\mathbf{w}}_x &= \frac{\tilde{\mathbf{c}}_x - \mathbf{c}_x}{\alpha}, \\ \tilde{\mathbf{w}}_y &= \frac{\tilde{\mathbf{c}}_y - \mathbf{c}_y}{\alpha}. \end{aligned} \tag{8.8}$$

The estimated fingerprint sequence consists of one or several users' contribution as well as some noise coming from distortions or attacks. The problem of finding out which user(s) has (have) contributed to the estimated fingerprint can be formulated as hypothesis testing [58, 59], which is commonly handled by evaluating the similarity between the estimated fingerprint sequence and each fingerprint sequence in the database through a correlation-based statistic. Various correlation-based statistics share a kernel term measuring the total correlation $\langle \tilde{\mathbf{w}}, \mathbf{w} \rangle$ (where $\tilde{\mathbf{w}} \triangleq [\tilde{\mathbf{w}}_x, \tilde{\mathbf{w}}_y]$ and $\mathbf{w} \triangleq [\mathbf{w}_x, \mathbf{w}_y]$) and differ in how they are normalized. To facilitate the evaluation of detection performance, we often normalize the detection statistic to make it have a unit variance and follow approximately a Gaussian distribution under distortions and attacks. There are several ways to do so [60], for example, to normalize using the product of the noise's standard deviation and the watermark's L_2-norm, or through a logarithm-based transformation to be introduced later. As we discussed in Chapters 2 and 3, among several correlation-based statistics analyzed and compared in the literature, the Z statistic shows excellent robustness against different collusion attacks and does not require the explicit estimation of the noise variance. These advantages make it attractive to handle our problem.

In our fingerprint detection problem from the control point domain, the number of samples in the statistic test is $L = 2(n+1)$. Denoting the average values of the components in $\tilde{\mathbf{w}}$ and $\mathbf{w}$ as $\tilde{\mu}$ and μ, respectively, we compute the sample correlation coefficient r between $\tilde{\mathbf{w}}$ and $\mathbf{w}$ by

$$r = \frac{\sum_{i=1}^{L} (\tilde{w}_i - \tilde{\mu})(w_i - \mu)}{\sqrt{\sum_{j=1}^{L} (\tilde{w}_j - \tilde{\mu})^2 \sum_{k=1}^{L} (w_k - \mu)^2}}. \tag{8.9}$$

Using the above r, the Z statistic is defined as below:

$$Z \triangleq \frac{\sqrt{L-3}}{2} \log \frac{1+r}{1-r}. \tag{8.10}$$

It was shown that it follows a unit-variance Gaussian distribution when $(\mathbf{w}, \tilde{\mathbf{w}})$ forms a bivariate normal population:

$$Z \sim \mathcal{N}\left(\frac{\sqrt{L-3}}{2} \log \frac{1+\rho}{1-\rho}, 1\right), \tag{8.11}$$

where ρ is the correlation coefficient of this bivariate normal population (W, Y), as discussed in Chapter 2. It is worth mentioning that, when the noise introduced by attacks does not follow a Gaussian distribution and/or the noise samples are mutually correlated, the Z statistics with the true traitors may be somewhat different from the unit-variance Gaussian distribution. The actual distribution of the Z statistics and the detection performance in terms of the detection probability and the false alarm probability will be presented in our experimental results.

8.2.3. Fidelity and robustness considerations

Fingerprint construction

The collusion resistance requirement makes the fingerprinting problem more challenging than robustly embedding a meaningful ID label, as the simple encoding of IDs can be vulnerable to collusion (e.g., different users average their copies of the same content to remove the IDs). Designing a collusion-resistant code is one of the possible approaches and has been studied in [80]. Such coded approach requires that each code symbol be reliably embedded, which consumes a nontrivial amount of markable features per embedded bit. The markable feature for our curve watermarking problem is the coordinates of control points. As the number of control points is limited and the changes have to be small, orthogonal modulation that uses (approximately) orthogonal signals to represent different users is more attractive than the coded modulation.The general collusion resistance of orthogonal fingerprinting has been studied in [58, 59], which shows that the maximum number of colluders the system can resist is a function of the watermark-to-noise ratio, the number of markable features, the total number of users, as well as the false positive and negative requirements.

While the orthogonal design of fingerprints is conceptually simple and easy to analyze, the practical implementation often employs pseudorandom number generators to produce a sequence of independent random numbers as a fingerprint and uses different seeds for different users [60, 80, 145]. In this way, the actual fingerprints would be statistically uncorrelated, but they can have nonzero correlation. This correlation can accommodate a larger number of fingerprint vectors than the vector's dimension, but it also affects the detection performance to some degree. Since the correlation is very low between the independent fingerprints, the impact is small. This can be seen from our experimental results of the detection performance in Sections 8.2.4 and 8.4.

Curvature-based sampling

The overall distortion introduced by the embedding process on a curve consists of two parts: one is from the watermark signals added to the control point coordinates, and the other is from the B-spline modeling. To make the B-spline synthesized curve as close to the original curve as possible and thus keep the modeling error low, the knots connecting adjacent segments of B-splines should be wisely placed and the sample points properly chosen to feed into the least-squares estimator for the control points. Uniform sampling can be used when there are no abrupt changes in a curve segment, while nonuniform sampling is desirable for curve segments that exhibit substantial variations in curvature.

Inspired by [146, 147], we employ a curvature-based method to select sample points from raster curves. Formally, the curvature [148] of a point $\mathbf{p}(t) = (p_x(t), p_y(t))$ on a curve $\{\mathbf{p}(t)\}$ is defined as

$$k(t) \triangleq \frac{p'_x p''_y - p'_y p''_x}{\left({p'_x}^2 + {p'_y}^2\right)^{3/2}}, \tag{8.12}$$

where $p'_x = dp_x/dt$, $p'_x = d^2p_x/dt^2$, $p'_y = dp_y/dt$, and $p'_y = d^2p_y/dt^2$. In practical implementations, we approximate the curvature of each point on the curve by measuring the angular changes in the tangent line at its location. Specifically, we perform a first-order polynomial curve fitting on an l-pixel interval before and after the curve point $\mathbf{p}(t)$ to get two slopes k_1 and k_2. The approximate curvature $\widehat{k}(t)$ is computed by $\widehat{k}(t) = |\arctan(k_1) - \arctan(k_2)|$. Based on $\widehat{k}(t)$, we select more sample points from higher-curvature segments and fewer from lower-curvature segments.

After selecting the $m+1$ sample points $(\mathbf{p}_x, \mathbf{p}_y)$ as defined in (8.3), we need to determine their indexing values $t = s_0, s_1, s_2, \ldots, s_m$ for evaluating their B-spline blending function values $B_{i,k}(t)$. In our tests, we employ the uniform nonperiodic B-spline blending function of order $k = 3$ and the knot parameters $\{t_i\}$ are determined as $[t_0, t_1, \ldots, t_{n+3}] = [0, 0, 0, 1, 2, 3, \ldots, n-2, n-1, n-1, n-1]$. One way of the indexing value assignment is known as the chord length method [149], which increases t values of the sample points in proportion to the chord length

$$s_j = s_{j-1} + \|\mathbf{p}_j - \mathbf{p}_{j-1}\| \frac{n-1}{\sum_{i=1}^{m} \|\mathbf{p}_i - \mathbf{p}_{i-1}\|}, \quad j = 1, 2, \ldots, m, \tag{8.13}$$

where $s_0 = 0$ and $\|\mathbf{p}_j - \mathbf{p}_{j-1}\|$ denotes the chord length between points $\mathbf{p}_j$ and $\mathbf{p}_{j-1}$.

In the vector curve case, we are generally given a set of discrete, nonuniformly spaced points for each curve. Since a vector curve can be rendered as a raster curve by interpolation, we can determine its sample points with their indexing values by rendering it to be a raster curve and then performing curvature-based sampling and the chord length method as described above. A simpler alternative is to directly use the given discrete points as sample points but assign their indexing values according to a curvature-based rule. Specifically, we approximate the curvature of each sample point $(p_x(s_j), p_y(s_j))$ by

$$\hat{k}(s_j) = \left| \arctan \frac{p_y(s_{j+1}) - p_y(s_j)}{p_x(s_{j+1}) - p_x(s_j)} - \arctan \frac{p_y(s_j) - p_y(s_{j-1})}{p_x(s_j) - p_x(s_{j-1})} \right| \tag{8.14}$$

and then increase t values of the sample points in inverse proportion to their curvature. The higher the curvature, the smaller the increase in the t value. This is equivalent to having more control points for higher-curvature segments. With more "resources" in terms of B-spline control points assigned to segments with more details (i.e., higher-curvature segments), it allows for a better B-spline approximation.

Determining the fingerprint length and strength

The number of control points is an important parameter for tuning. Depending on the shape of the curve, using too few control points could cause the details of the curve to be lost, while using too many control points may lead to overfitting and bring artifacts even before data embedding. One simple method of determining the number of control points is to compute the approximate curvature of each sample point as in (8.14) and assign higher weights to points with higher curvature. We then determine the number of control points according to the total weights of all sample points on the curve. The number of control points not only affects the distortion introduced by the embedding, but also determines the fingerprint robustness against noise and attacks. The more the control points, the longer the fingerprint sequence, and in turn the more robust the fingerprint against noise and attacks. Too many control points, however, may lead to overfitting and incur visible distortions even before data embedding. In our tests, the number of control points is about 5%–8% of the total number of curve pixels.

The scaling factor α also affects the invisibility and robustness of the fingerprint. The larger the scaling factor, the more robust the fingerprint, but the larger the distortion resulted in. For cartographic applications, industrial standards provide guidelines on the maximum allowable changes [130]. Perturbation of 2 to 3 pixels is usually considered acceptable. We use random number sequences with a unit variance as fingerprints and set α to 0.5 in our tests. The difference between two curves can be quantified using such metric as the Hausdorff distance [150] in a max-min sense. More specifically, let $d(a, b)$ be the distance between two points a and b, which are on two curves A and B, respectively. We further define the distance from point a to curve B as $d(a, B) \triangleq \inf_{b \in B} d(a, b)$, and the distance from curve A to curve B as $d_B(A) \triangleq \sup_{a \in A} d(a, B)$. Thus, the Hausdorff distance between curves A and B is

$$h(A, B) \triangleq d_B(A) + d_A(B). \tag{8.15}$$

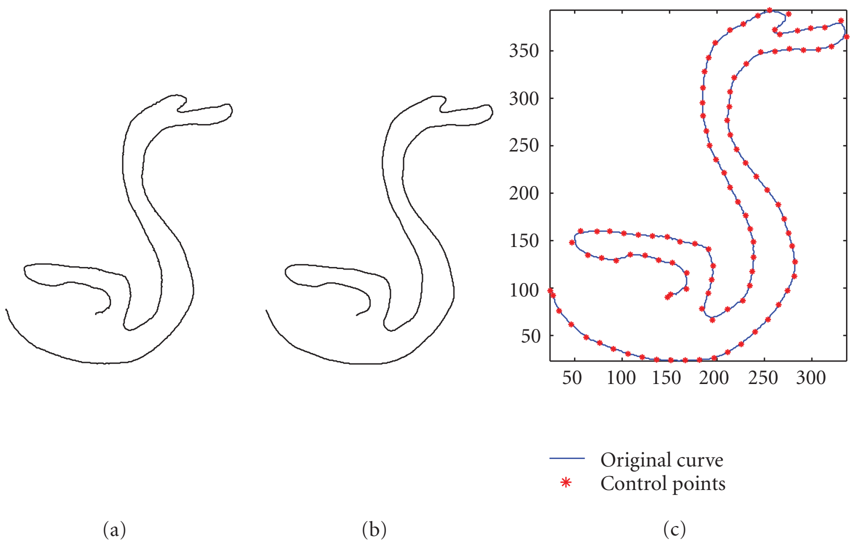


Figure 8.2. Fingerprinting a hand-drawn "*Swan*" curve: (a) original curve, (b) fingerprinted curve, (c) control points overlaid on the original curve.

Nonblind detection

The basic fingerprint detection presented earlier makes use of the original unmarked copy and is known as *nonblind detection*. While blind detection is preferred for a few major data hiding applications (such as ownership verification, authentication, and annotation), nonblind detection is considered as a reasonable assumption for many fingerprinting applications [17, 44, 80], which is also the focus of this chapter. The rationale for nonblind detection is that the fingerprint verification is usually handled by the content owner or by an authorized central server, who can have access to the original host signal and use it in detection to answer the primary question of whose fingerprint is in the suspicious document. The availability of the original unmarked copy in the detection gives a high equivalent watermark-to-noise ratio, thus allowing for high resistance against noise and attacks. Additionally, using the original unmarked copy as a reference copy, the detector can register a test copy that suffers from geometric distortions, which enables the resilience to various geometric transformations as to be demonstrated later in this chapter.

8.2.4. Experiments with simple curves

To demonstrate our basic embedding and detection algorithms, we first present the fingerprinting results on two simple curves, the "Swan" curve in Figure 8.2a and the "W" curve in Figure 8.3a. These two curves were hand-drawn on a TabletPC and stored as binary images of size 329×392 and 521×288, respectively. We use the contour following algorithm in [147] to traverse the curve and

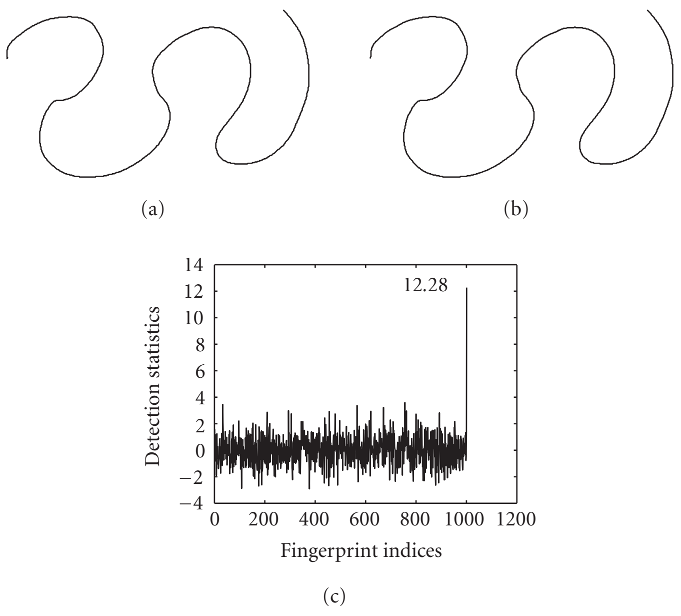


FIGURE 8.3. Fingerprinting a hand-drawn "*W*" curve: (a) original curve, (b) fingerprinted curve, (c) detection statistics.

obtain a set of ordered curve points. When fingerprinting these two curves, we perform uniform sampling on the curve points and determine the indexing values of the sample points using the chord length method. We highlight the control points of the "Swan" curve in Figure 8.2c. The fingerprinted curves are shown in Figures 8.2b and 8.3b, where we have marked 101 control points for each curve. With the marked control points, we construct a fingerprinted curve with the same number of points as the original curve by evaluating the B-spline synthesis formula (8.1) at indexing values uniformly sampled between $t = 0$ and $t = n - 1$. As for the fidelity of the fingerprinting, the Hausdorff distance between the original and marked curves is 5.0 for the "Swan" curve, and 3.4 for the "W" curve. The differences are hardly visible to human eyes.

In the detection, uniform sampling on the fingerprinted curves constructed before can give us an approximation of the set of sample points $(\widetilde{\mathbf{p}}_x, \widetilde{\mathbf{p}}_y) = (\mathbf{B}(\mathbf{c}_x + \alpha\mathbf{w}_x), \mathbf{B}(\mathbf{c}_y + \alpha\mathbf{w}_y))$ assumed in Section 8.2.2. We then estimate the test control points and perform correlation-based detection. The detection results on the fingerprinted "W" curve are shown in Figure 8.3c, which illustrates a high Z value for the correct positive detection with the 1000th sequence corresponding to the true user and very small Z statistics for the correct negative detection with other sequences of the innocent users. To quantify the detection performance, we generate 1000 different sets of fingerprint sequences and each set consists of 1000 independent fingerprint sequences that are approximately orthogonal to each other. For each set, we embed each of the 1000 fingerprint sequences to form a fingerprinted

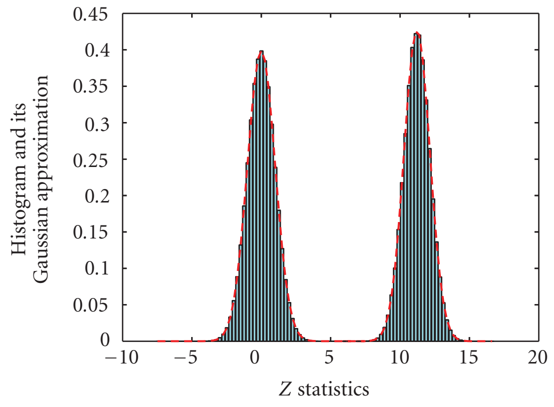


Figure 8.4. Histogram and Gaussian approximation of the Z statistics for the "W" curve.

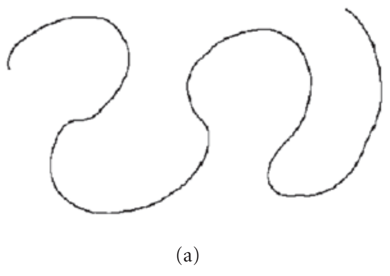
curve, and then estimate a fingerprint sequence from the fingerprinted curve and compute the Z values with the 1000 fingerprint sequences. In this way, we collect a total of $1000 \times 1000 = 1 \times 10^6$ values for the fingerprint presence case and $1000 \times 1000 \times 999 \approx 1 \times 10^9$ values for the fingerprint absence case. Using these data, we plot in Figure 8.4 the histogram of the Z values for both fingerprint presence and absence cases. For each of these two sets of data, we calculate its mean and variance, and plot the Gaussian distributions with these means and variances. As shown by the dashed curves in Figure 8.4, the two Gaussian approximations $\mathcal{N}(11.22, 0.87)$ and $\mathcal{N}(-0.0009, 1.00)$ fit the observed Z values very well. We can see that we indeed get an approximate Gaussian distribution with a large positive mean under the presence of a fingerprint and a zero mean under the absence.

We further examine the measured and the Gaussian approximated probabilities of detection and false alarm for different thresholds on the Z statistic. We see from the results in Table 8.1 that the Gaussian approximation works well in regions close to the mean, while slightly deviates from Gaussian in regions farther from the mean. In the meantime, we note that in the tail region where the Gaussian approximation becomes loose, the approximated order of magnitude matches very well with the measured distribution from our experiments. The probability of miss or false alarm concerned there is already very small (below 10^{-5}). Therefore, for many practical applications, either the probability may be deemed as zero or an approximation on the order of magnitude would be sufficient. The table also shows that a threshold of 6 on the detection statistics gives a false alarm probability of 10^{-9}, which is sufficiently low for most applications. Thus we choose 6 as the detection threshold in our tests. The detection result for the "Swan" curve is similar and will not be repeated here.

To examine the survivability of our proposed basic fingerprinting mechanism that employs the coordinates of the B-spline control points as the embedding domain and approximately orthogonal spread-spectrum signals as fingerprints, we perform a printing-and-scanning test with manual registration between the

Table 8.1. Measured detection performance and its Gaussian approximation for the "W" curve.

Threshold on Z		3	4.5	6	7.5	9
$1 - P_d$	Measured	0	0	0	4.1×10^{-5}	8.7×10^{-3}
	Gaussian approx.	6.6×10^{-19}	3.1×10^{-13}	1.1×10^{-8}	3.4×10^{-5}	8.7×10^{-3}
P_{fa}	Measured	1.4×10^{-3}	4.1×10^{-6}	2.0×10^{-9}	0	0
	Gaussian approx.	1.3×10^{-3}	3.4×10^{-6}	0.97×10^{-9}	3.1×10^{-14}	1.1×10^{-19}



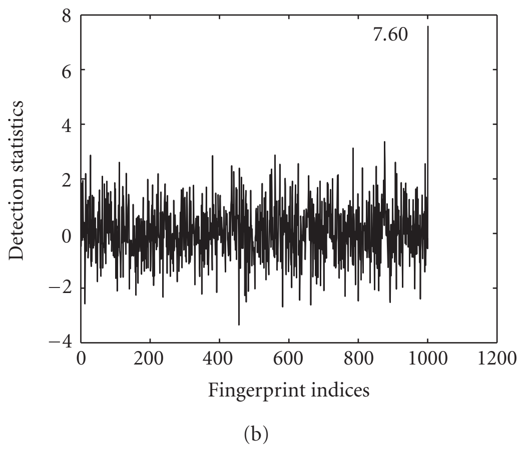


Figure 8.5. Printing-and-scanning test for the "W" curve: (a) fingerprinted curve after printing and scanning, (b) detection statistics.

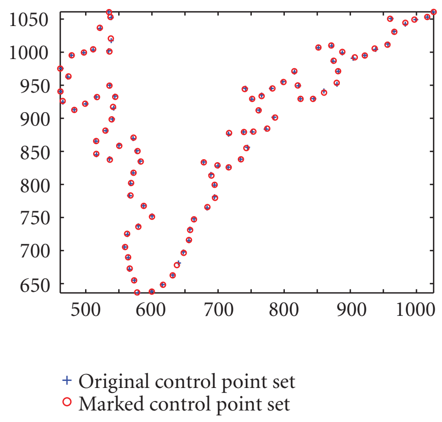
scanned fingerprinted curve and the original unmarked curve. We print out the fingerprinted "W" curve using an HP laser printer and scan it back as a 527×288 binary image as shown in Figure 8.5a. In addition to manual registration, a thinning operation is performed to extract a one-pixel-wide skeleton from the scanned curve that is usually several pixels wide after high resolution scanning. As we can see from the detection results in Figure 8.5b, despite that the curve is simple and the number of control points is relatively small, the fingerprint survives the printing-and-scanning process and gives a detection statistic higher than the detection threshold. The issue of automating the registration process will be addressed in the next section.

8.3. Iterative alignment-minimization algorithm for robust fingerprint detection

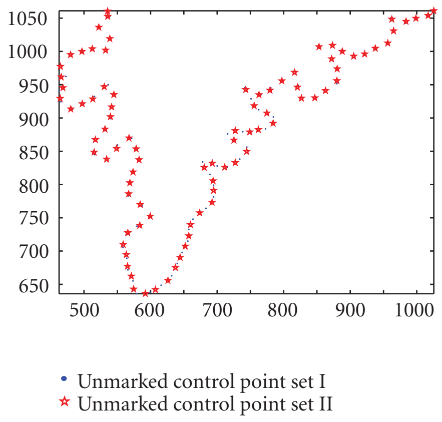
The set of test sample points $(\tilde{\mathbf{p}}_x, \tilde{\mathbf{p}}_y)$ assumed in Section 8.2.2 is not always available to a detector, especially when a test curve undergoes vector-raster conversion, undergoes geometric transformations (such as rotation, translation, and scaling), and/or is scanned from a printed hard copy. A preprocessing step preceding the basic fingerprint detection module is needed to align the test curve with the original one. While manual registration between the test curve and the original unmarked curve shown in Section 8.2.4 is a possible way to overcome simple geometric distortions, automated registration is more desirable to improve the accuracy and efficiency of this indispensable preprocessing step. It should also be noted that the test curve should be registered with the original unmarked curve, and any “clean/undistorted” fingerprinted copies known to the detector should not be used as a reference for registering the test curve. This is not only because which fingerprints are present in the test curve still remains to be determined, but also because using a fingerprinted copy as a reference for registration may increase the false alarm in determining the presence or absence of the corresponding fingerprint.

With the affine invariance property, B-splines have been used in a few existing curve alignment works. In the moment-based approach of [146], two affine related curves are fitted by two separate B-splines, and the transform parameters are estimated by using weighted B-spline curve moments. The method requires taking integration as well as the second-order curve derivatives to obtain the moments. In a recent method employing a *super curve* [151], two affine related curves are superimposed with each other in a single frame and then this combined *super curve* is fitted by a single B-spline. Through minimizing the B-spline fitting error, both transform parameters and control points of the fitting B-spline can be estimated simultaneously. Since neither integration nor differentiation is needed, this method is robust to noise and will serve as a building block in our work.

Another problem related to the test sample points assumed before is the inherent nonuniqueness of B-spline control points, which refers to the fact that a curve can be well approximated by different sets of B-spline control points. We have seen from Section 8.2.1 that B-spline control points are estimated from a set of sample points from the curve. With a different set of sample points or a different indexing value assignment, we may induce a quite different set of control points that can still accurately describe the same curve. It is possible for the differences between two sets of unmarked control points to be much larger than the embedded fingerprint sequence, as demonstrated by the example in Figure 8.6. Therefore, if we cannot find from a test curve a set of control points corresponding to the one used in the embedding, we may not be able to detect the fingerprint sequence. Considering the one-to-one relationship between sample points (including their indexing values $\{s_j\}$) and control points, we try to find the set of sample points from a test curve that corresponds to the set of sample points used in the embedding. We



(a)



(b)

Figure 8.6. Nonuniqueness of B-spline control points: (a) a set of control points for the original unmarked curve and its fingerprinted version, (b) two different sets of control points for modeling the same unmarked curve.

will refer to this problem as the *point correspondence problem*. As we will see, the nonuniqueness issue of B-spline control points can be addressed through finding the point correspondence.

In the following subsections, we first formulate the curve registration and point correspondence problem in the context of fingerprint detection. We then take the curve alignment method introduced in [151] as a building block and propose an iterative alignment-minimization (IAM) algorithm that can perform curve registration and solve the point correspondence problem simultaneously. Finally, we present a detection example for a single curve using the IAM algorithm and discuss the robustness issues.

8.3.1. Problem formulation

We use "View-I" to refer to the geometric setup of the original unmarked curve and "View-II" the setup of the test curve. Thus we can register the two curves by transforming the test curve from "View-II" to "View-I," or transforming the original curve from "View-I" to "View-II." We focus on registration under affine transformations, which can represent combinations of scaling, rotation, translation, reflection, and shearing. These are common geometric transformations and can well model common scenarios in printing and scanning.

We call two curves *affine related* if each point (x, y) on one curve is related to its corresponding point $(\tilde{x}, \tilde{y})$ on another curve via

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_{13} \\ a_{23} \end{bmatrix}, \tag{8.16}$$

where $\{a_{ij}\}$ are parameters representing the collective effect of scaling, rotation, translation, reflection, and shearing. The transform parameters can also be represented in a homogeneous coordinate by two column vectors $\mathbf{a}_x$ and $\mathbf{a}_y$, or by a single matrix $\mathbf{A}$:

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} & \mathbf{a}_x^T & \\ & \mathbf{a}_y^T & \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{8.17}$$

Similarly, the inverse transform can be represented by

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{A}^{-1} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix} \triangleq \begin{bmatrix} & \mathbf{g}_x^T & \\ & \mathbf{g}_y^T & \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{bmatrix}. \tag{8.18}$$

The original curve available to the detector in fingerprinting applications can be a raster curve or a vector curve. The detector also knows the original set of sample points $(\mathbf{p}_x, \mathbf{p}_y) \approx (\mathbf{B}\mathbf{c}_x, \mathbf{B}\mathbf{c}_y)$ that is used for estimating the set of control points upon which spread-spectrum embedding is applied. The test curve can be a vector curve with sampled curve points $(\tilde{\mathbf{v}}_x, \tilde{\mathbf{v}}_y)$ or a raster curve with pixel coordinates $(\tilde{\mathbf{r}}_x, \tilde{\mathbf{r}}_y)$. A relatively simple case is that the set of discrete points in a test vector curve corresponds to the set of sample points used in the embedding except with possible affine transformations and noise addition. In this case, the test vector points $(\tilde{\mathbf{v}}_x, \tilde{\mathbf{v}}_y)$ and the original set of control points $(\mathbf{c}_x, \mathbf{c}_y)$ are related by

$$\begin{aligned} \tilde{\mathbf{v}}_x &= [\mathbf{B}(\mathbf{c}_x + \alpha\mathbf{w}_x) \quad \mathbf{B}(\mathbf{c}_y + \alpha\mathbf{w}_y) \quad 1]\mathbf{a}_x + \mathbf{n}_x, \\ \tilde{\mathbf{v}}_y &= [\mathbf{B}(\mathbf{c}_x + \alpha\mathbf{w}_x) \quad \mathbf{B}(\mathbf{c}_y + \alpha\mathbf{w}_y) \quad 1]\mathbf{a}_y + \mathbf{n}_y, \end{aligned} \tag{8.19}$$

where $(\mathbf{n}_x, \mathbf{n}_y)$ represents additional noise applied to the transformed fingerprinted vector points, and $\mathbf{1}$ is a column vector with all 1's. With the point correspondence information, the only issue is curve alignment and it can be solved by directly applying the curve alignment method in [151]. However, in addition to possible affine transformations between the original and the test curve, the correct point correspondence information may not always be available. This is especially the case after a fingerprinted curve goes through vector-raster conversions and/or printing and scanning. Under this situation, not only transform parameters for the curve alignment but also the point correspondence must be estimated in order to locate the fingerprinted control points successfully. We consider that both the original and the test curves are represented in raster format as a vector curve can be rendered as a raster curve by interpolation, and that the sample points used in fingerprinting the original curve are known to the detector. The problem can be formulated as follows.

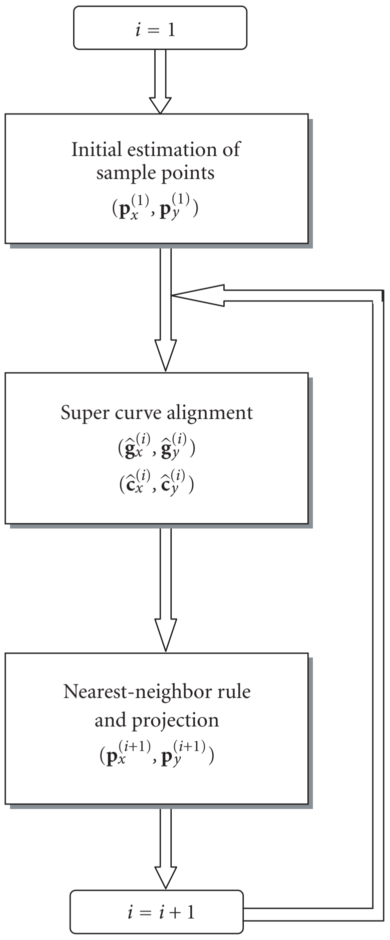
Problem 8.1. Given an original raster curve with a set of sample points $(\mathbf{p}_x, \mathbf{p}_y)$ and a test raster curve $(\widetilde{\mathbf{r}}_x, \widetilde{\mathbf{r}}_y)$, we register the test curve with the original curve and extract the control points of the test curve. Both transform parameters $(\mathbf{a}_x, \mathbf{a}_y)$ (or equivalently $(\mathbf{g}_x, \mathbf{g}_y)$) and a set of sample points $(\widetilde{\mathbf{p}}_x, \widetilde{\mathbf{p}}_y)$ corresponding to the one used in the fingerprint embedding must be found from the test curve.

8.3.2. Iterative alignment-minimization algorithm

To align the test curves with the original curves and in the meantime identify the point correspondence of the sample points, we develop an iterative alignment-minimization (IAM) algorithm. As shown in Figure 8.7, the IAM algorithm consists of three main steps and the last two steps will be executed iteratively. We first obtain an initial estimation of the test sample points. With the estimated point correspondence, we then perform "super" curve alignment to estimate both the transform parameters and the control points of the test curve. With the estimated transform parameters, we refine the estimation of point correspondence through a nearest-neighbor rule. A detailed block diagram of the proposed IAM-based fingerprint detection is shown in Figure 8.8.

Step 1. Initial estimation of sample points on the test curve.

We initialize the sample points $(\widetilde{\mathbf{p}}_x^{(1)}, \widetilde{\mathbf{p}}_y^{(1)})$ on the test curve using the following simple estimator. Let M and $\widetilde{M}$ be the number of points on the original and the test raster curve, respectively. From the known indices $\mathbf{J} = [j_0, j_1, j_2, \dots, j_m]$ of the original curve's $m+1$ sample points, where $j_0 < j_1 < j_2 < \cdots < j_m$ are integers ranging from 0 to $M-1$, we estimate the indices of the test curve's $m+1$ sample points by $\widetilde{\mathbf{J}} = \operatorname{round}(((\widetilde{M}-1)/(M-1)) \cdot \mathbf{J})$. Using this estimated index vector $\widetilde{\mathbf{J}}$, we can identify the corresponding sample points from the test curve and take them as the initial estimate.



$i = 1$
Initial estimation of sample points $(\mathbf{p}_x^{(1)}, \mathbf{p}_y^{(1)})$
Super curve alignment $(\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)})$ $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$
Nearest-neighbor rule and projection $(\mathbf{p}_x^{(i+1)}, \mathbf{p}_y^{(i+1)})$
$i = i + 1$

FIGURE 8.7. Basic flow and main modules of the proposed iterative alignment minimization algorithm.

Step 2. Curve alignment with the estimated sample points.

Given the estimated point correspondence with sample points $(\tilde{\mathbf{p}}_x^{(i)}, \tilde{\mathbf{p}}_y^{(i)})$ for the test curve in the *i*th iteration, we apply the curve alignment method in [151] to estimate the transform parameters and the control points of the test curve. More specifically, let the transform parameters from View-I (the original curve) to View-II (the test curve) be $(\mathbf{a}_x^{(i)}, \mathbf{a}_y^{(i)})$. The sample points on the test curve can be transformed back to View-I by $(\mathbf{g}_x^{(i)}, \mathbf{g}_y^{(i)})$. We then fit these transformed test sample points as well as the original sample points with a single B-spline curve (referred to as a "super curve" in [151]) and search for both the transform parameters $(\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)})$ and the B-spline control points $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$ to minimize the fitting

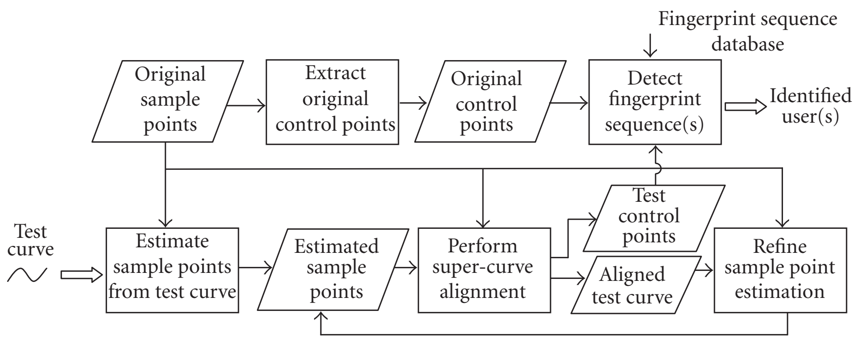

Fingerprint sequence database
Original sample points
Extract original control points
Original control points
Detect fingerprint sequence(s)
Identified user(s)
Test curve
Estimate sample points from test curve
Estimated sample points
Perform super-curve alignment
Test control points
Aligned test curve
Refine sample point estimation

FIGURE 8.8. Block diagram of curve registration and fingerprint detection using the proposed IAM algorithm.

error

$$
\begin{aligned}
&f\left(\hat{\mathbf{c}}_x^{(i)},\hat{\mathbf{c}}_y^{(i)},\hat{\mathbf{g}}_x^{(i)},\hat{\mathbf{g}}_y^{(i)}\right)\\
&\quad=\left\|\begin{bmatrix}\mathbf{B}\\ \mathbf{B}\end{bmatrix}\hat{\mathbf{c}}_x^{(i)}-\begin{bmatrix}\mathbf{p}_x\\ \widetilde{\mathbf{P}}^{(i)}\hat{\mathbf{g}}_x^{(i)}\end{bmatrix}\right\|^2+\left\|\begin{bmatrix}\mathbf{B}\\ \mathbf{B}\end{bmatrix}\hat{\mathbf{c}}_y^{(i)}-\begin{bmatrix}\mathbf{p}_y\\ \widetilde{\mathbf{P}}^{(i)}\hat{\mathbf{g}}_y^{(i)}\end{bmatrix}\right\|^2,
\end{aligned}
\tag{8.20}
$$

where $\widetilde{\mathbf{P}}^{(i)} \triangleq [\widetilde{\mathbf{p}}_x^{(i)} \quad \widetilde{\mathbf{p}}_y^{(i)} \quad \mathbf{1}]$ and $\mathbf{1}$ is a column vector with all 1's. The partial derivatives of the fitting error function with respect to $\hat{\mathbf{g}}_x^{(i)}$, $\hat{\mathbf{g}}_y^{(i)}$, $\hat{\mathbf{c}}_x^{(i)}$, and $\hat{\mathbf{c}}_y^{(i)}$ being zero is the necessary condition for the solution to this optimization problem. Thus we obtain an estimate of the transform parameters and the B-spline control points as

$$
\begin{aligned}
\hat{\mathbf{g}}_x^{(i)} &= \mathbf{C}^{(i)}\mathbf{D}^{(i)}\mathbf{p}_x, \qquad & \hat{\mathbf{g}}_y^{(i)} &= \mathbf{C}^{(i)}\mathbf{D}^{(i)}\mathbf{p}_y,\\
\hat{\mathbf{c}}_x^{(i)} &= \mathbf{D}^{(i)}\mathbf{p}_x, \qquad & \hat{\mathbf{c}}_y^{(i)} &= \mathbf{D}^{(i)}\mathbf{p}_y,
\end{aligned}
\tag{8.21}
$$

where

$$
\begin{aligned}
\mathbf{C}^{(i)} &\triangleq \left(\widetilde{\mathbf{P}}^{(i)T}\widetilde{\mathbf{P}}^{(i)}\right)^{\dagger}\widetilde{\mathbf{P}}^{(i)T}\mathbf{B},\\
\mathbf{D}^{(i)} &\triangleq \left(2\mathbf{B}^T\mathbf{B}-\mathbf{B}^T\widetilde{\mathbf{P}}^{(i)}\mathbf{C}^{(i)}\right)^{\dagger}\mathbf{B}^T.
\end{aligned}
\tag{8.22}
$$

The estimated control points $(\hat{\mathbf{c}}_x^{(i)},\hat{\mathbf{c}}_y^{(i)})$ can then be used to estimate the embedded fingerprint sequence and further compute the detection statistic $Z^{(i)}$, as described in Section 8.2.

Step 3. Refinement of sample point estimation on the test curve.

Given the estimated transform parameters $(\hat{\mathbf{g}}_x^{(i)}, \hat{\mathbf{g}}_y^{(i)})$, we align the test raster curve $(\tilde{\mathbf{r}}_x, \tilde{\mathbf{r}}_y)$ with the original curve by transforming it to View-I:

$$\begin{aligned} \tilde{\mathbf{r}}_{x,I}^{(i)} &= [\tilde{\mathbf{r}}_x \quad \tilde{\mathbf{r}}_y \quad \mathbf{1}]\hat{\mathbf{g}}_x^{(i)}, \\ \tilde{\mathbf{r}}_{y,I}^{(i)} &= [\tilde{\mathbf{r}}_x \quad \tilde{\mathbf{r}}_y \quad \mathbf{1}]\hat{\mathbf{g}}_y^{(i)}. \end{aligned} \tag{8.23}$$

As the fingerprinted sample points $(\mathbf{B}(\mathbf{c}_x + \alpha\mathbf{w}_x), \mathbf{B}(\mathbf{c}_y + \alpha\mathbf{w}_y))$ are located at the neighborhood of their corresponding unmarked version $(\mathbf{B}\mathbf{c}_x, \mathbf{B}\mathbf{c}_y)$, we apply a *nearest-neighbor* rule to get a refined estimation of the test curve's sample points. More specifically, for each point of $(\mathbf{B}\mathbf{c}_x, \mathbf{B}\mathbf{c}_y)$, we find its closest point from the aligned test raster curve $(\tilde{\mathbf{r}}_{x,I}^{(i)}, \tilde{\mathbf{r}}_{y,I}^{(i)})$ and then denote the collection of these closest points as $(\tilde{\mathbf{p}}_{x,I}^{(i+1)}, \tilde{\mathbf{p}}_{y,I}^{(i+1)})$. These nearest neighbors form refined estimates of the test sample points in View-I and are then transformed with parameters $(\hat{\mathbf{a}}_x^{(i)}, \hat{\mathbf{a}}_y^{(i)})$ back to View-II as new estimates of the test sample points:

$$\begin{aligned} \tilde{\mathbf{p}}_x^{(i+1)} &= \left[\tilde{\mathbf{p}}_{x,I}^{(i+1)} \quad \tilde{\mathbf{p}}_{y,I}^{(i+1)} \quad \mathbf{1}\right]\hat{\mathbf{a}}_x^{(i)}, \\ \tilde{\mathbf{p}}_y^{(i+1)} &= \left[\tilde{\mathbf{p}}_{x,I}^{(i+1)} \quad \tilde{\mathbf{p}}_{y,I}^{(i+1)} \quad \mathbf{1}\right]\hat{\mathbf{a}}_y^{(i)}. \end{aligned} \tag{8.24}$$

After this update, we increase i and go back to Step 2. The iteration will continue until convergence or for an empirically determined number of times. A total of 15 rounds of iterations are used in our experiments.

8.3.3. Detection example and discussion

We present a detection example employing the proposed IAM algorithm on a curve taken from a topographic map. Shown in Figure 8.9a are the original curve and a fingerprinted curve having undergone vector-raster conversion and some geometric transformations. The original curve consists of 3796 raster pixels and 367 of them are used to estimate a set of 200 control points for data embedding. Then, a fingerprinted curve with 367 vector points is generated, rendered, and transformed to be a raster curve with 3587 raster pixels. After the vector-raster conversion, the point correspondence is no longer directly available from the raster curve representation. We apply the IAM algorithm to align the test curve with the original one and estimate the correspondence between sample points. The estimated sample points for the test curve after one iteration and 15 iterations are shown in Figures 8.9b and 8.9c, respectively. We can see that the initial estimates deviate from the true values by a nontrivial amount, while after 15 iterations the estimated values converge to the true values. We plot the six estimated transform parameters for each iteration in Figure 8.10a, which shows an accurate registration by the proposed IAM algorithm. Upon convergence, we use the estimated control points $(\hat{\mathbf{c}}_x^{(i)}, \hat{\mathbf{c}}_y^{(i)})$ to perform detection with the fingerprint involved. The high fingerprint detection statistic value shown in Figure 8.10b suggests the positive identification of the correct fingerprint by using the proposed IAM algorithm.

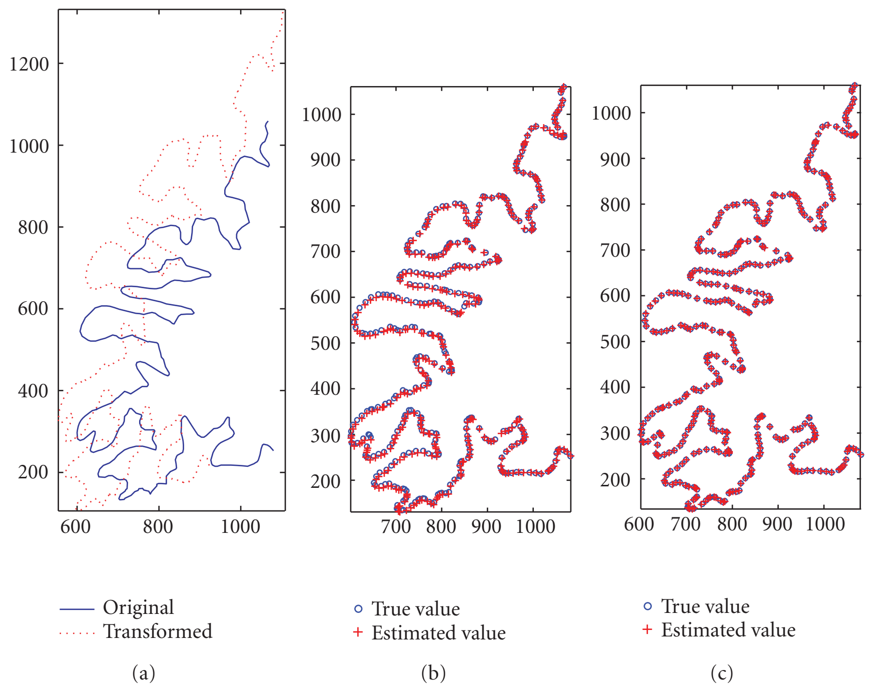
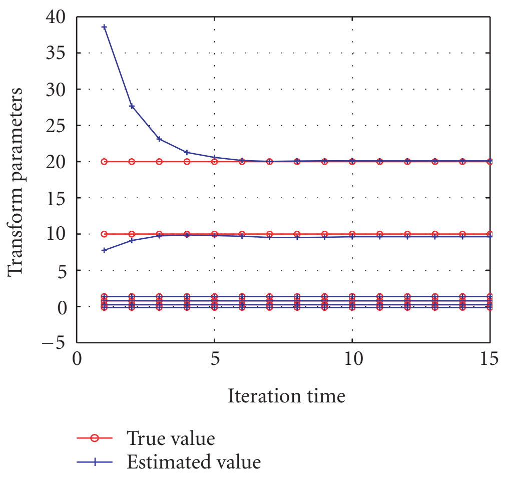


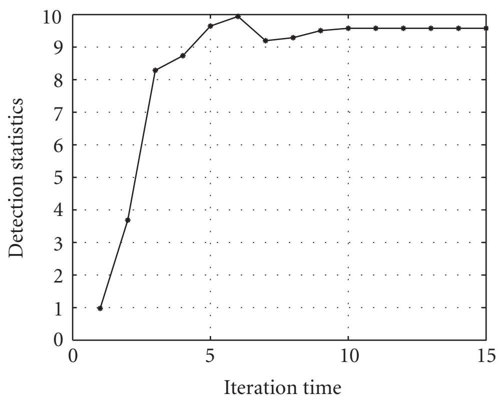
FIGURE 8.9. Registering a curve using the proposed IAM algorithm: (a) the original curve and a fingerprinted curve undergone vector-raster conversion and affine transformation, (b) estimated sample points after one iteration, (c) estimated sample points after 15 iterations.

The computation time for this experiment is as follows. The IAM algorithm is implemented in Matlab 6.5 and tested on a Pentium-4 2.0 GHz PC with 512 M RAM. Each iteration of the algorithm takes about 0.5 second, and the total 15 iterations plus the initialization take 7.61 seconds. Together with the 0.56 second required for computing Z statistics with 1000 fingerprint sequences, the total duration of the detection process is 8.17 seconds.

The above example shows that through the IAM algorithm, we can register the test curve with the original unmarked curve and extract the fingerprinted control points with high accuracy. With good estimation of affine transform parameters, our data embedding method for curves is resilient to combinations of scaling, rotation, translation, and shearing. The explicit estimation of point correspondence also provides resilience against the vector-raster conversion and vector-raster-vector conversion. In the vector-raster conversion case, a fingerprinted curve stored in vector format is rendered as a raster curve, and thus the point correspondence is no longer directly available from the raster curve representation. In the case of vector-raster-vector conversion, the intermediate raster curve is converted to a vector curve with a new set of vector points that are likely to be different from the initial vector points prior to the conversion, even though there is little visual difference between these two vector curves. Again the point correspondence



(a)



(b)

Figure 8.10. Convergence results on estimated transform parameters and detection statistics for the example curve in Figure 8.9 using the proposed IAM algorithm: (a) the estimated transform parameters after each iteration, (b) the fingerprint detection statistic after each iteration.

is likely to get corrupted by this conversion, and accurate estimation of point correspondence is a necessary step to the successful detection of the fingerprint. With the robustness resulting from spread-spectrum embedding in B-spline control points and the IAM algorithm, our curve fingerprinting approach can resist a number of challenging attacks and distortions. For example, the distortion from printing and scanning involves both vector-raster rendering and a certain amount of rotation, scaling, and translation; a fingerprinted curve in vector format may be rendered as a raster image and then affine transformed before reaching the detector; in the collusion scenario, colluders may construct a colluded copy, print it out, and then distribute it out of the allowed circle. In the next section, we use our curve-based data hiding approach to fingerprint topographic maps and demonstrate the robustness of our approach against various attacks and distortions.

8.4. Experiments with maps

We now present experimental results of the proposed curve fingerprinting algorithm in the context of tracing and tracking topographic maps. A topographic map provides a two-dimensional representation of the earth's three-dimensional surface. Vertical elevation is shown with contour lines (also known as level lines) to represent the earth's surfaces that are of equal altitude. Contour lines in topographic maps often exhibit a considerable amount of variations and irregularities, prompting the need of nonuniform sampling of curve points in the parametric modeling of the contours. We will first examine the fidelity of the fingerprinted map, and then evaluate the robustness of the fingerprints against collusion, cropping, geometric transformation, format conversion, curve smoothing, point deletion, printing and scanning, and some combinations of these distortions.

Fingerprinted topographic maps

A 1100×1100 topographic vector map obtained from http://www.ablesw.com is used in our experiment. Starting with the original map shown in Figure 8.11a, we mark nine curves that are sufficiently long. For each of these nine curves, a set of nonuniformly spaced vector points is defined. We directly use these given points as sample points and determine their indexing values according to the curvature-based rule presented in Section 8.2.3. A total of 1331 control points are used to carry the fingerprint. We overlay in Figure 8.11b these nine original and marked curves using solid lines and dotted lines, respectively. To help illustrate the fidelity of our method, we enlarge a portion of the overlaid image in Figure 8.11c. We can see that the fingerprinted map preserves the geospatial information in the original map with high precision. The perturbation can be adapted to be compliant with cartographic industry standards and/or the need of specific applications.

Resilience to collusion

To demonstrate the resistance of the proposed method against collusion, we present in Figure 8.12 the detection statistics under two different types of collusion attacks. Figure 8.12a shows the collusion results under a random interleaving attack, where the control points for each curve are equiprobably taken from two differently fingerprinted maps. The collusion attack for Figures 8.12b and 8.12c is known as averaging, where the coordinates of the corresponding control points from two and five differently fingerprinted maps are averaged, respectively. We assume the correct point correspondence is available in this test and the cases with unknown point correspondence will be addressed in the later subsections. As we can see from the detection statistics, the embedded fingerprints from all contributing users survive the collusion attacks and are identified with high confidence. For

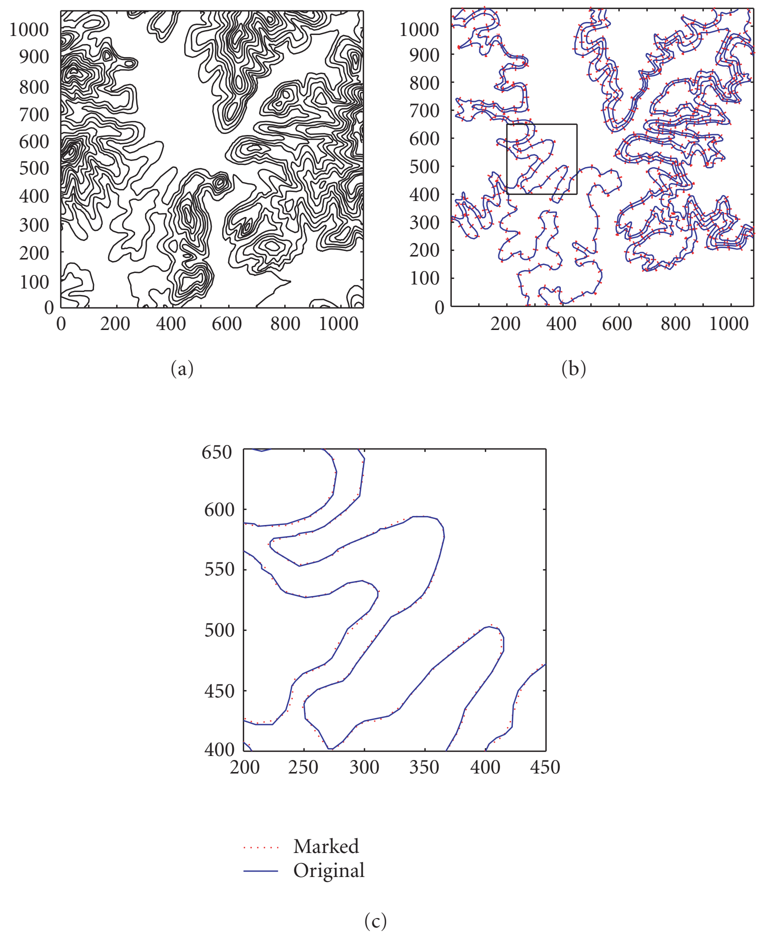


FIGURE 8.11. Fingerprinting topographic maps: (a) original map, (b) original and fingerprinted curves overlaid with each other, (c) a zoomed-in view of (b).

the cases of 2-user random interleaving collusion and 5-user averaging collusion, we use 20 different sets of fingerprint sequences to evaluate the detection performance using a similar experiment setup to the one discussed in Section 8.2.4. The histograms and the Gaussian approximations in Figure 8.13 show a very good separation between the Z values for the presence and absence of true fingerprints. Compared with 2-user averaging collusion, the 2-user random interleaving leads to more substantial coordinate changes in control points when a detector performs B-spline parametrization, and such coordinate changes may follow a distribution different from i.i.d. Gaussian. This is reflected by a reduced mean and a nonunit variance for the fingerprint presence case.

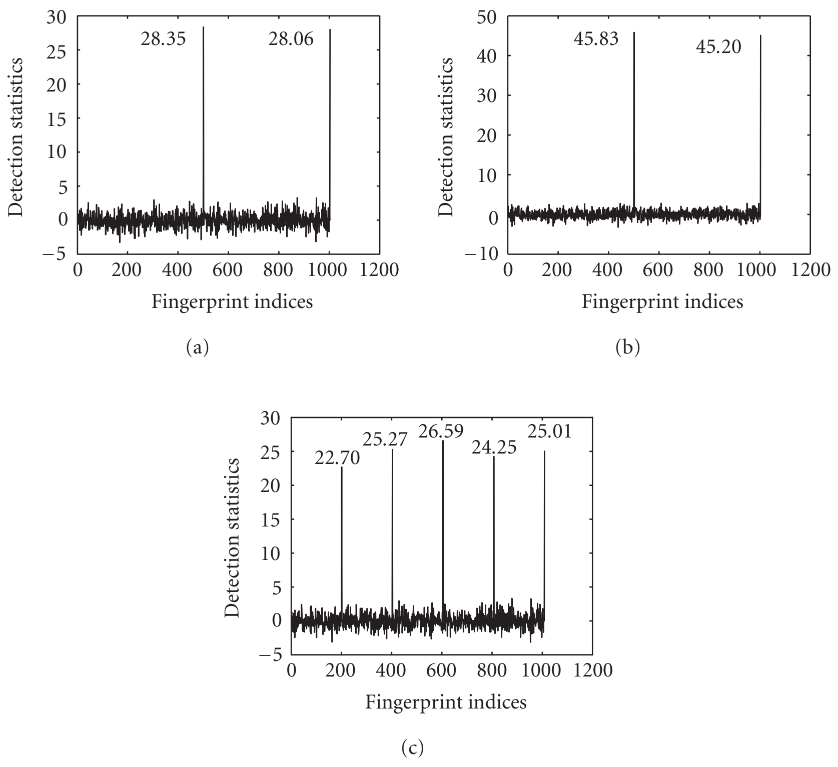


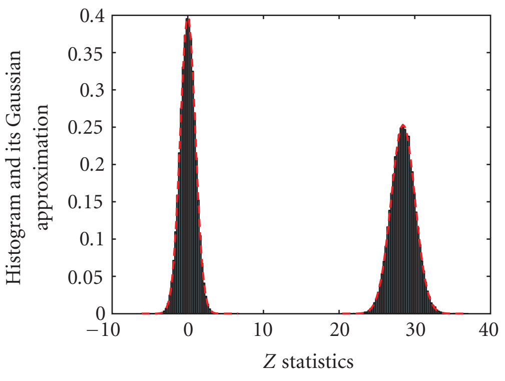
Figure 8.12. Collusion test on fingerprinted vector maps: (a) 2-user random interleaving collusion, (b) 2-user averaging collusion, (c) 5-user averaging collusion.

Resilience to cropping

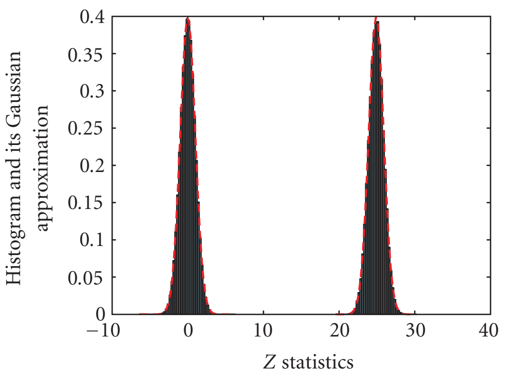
As shown in Figures 8.14a and 8.14b, we crop an area of a fingerprinted vector map and use it as the test map. Among the nine curves used for carrying the fingerprint, only two curves are retained with sufficiently large size. We perform detection on these two retained segments and obtain the detection result shown in Figure 8.14c. As we can see, the detection statistic with the correct fingerprint is still high enough so that its corresponding user can be identified with high confidence.

Resilience to affine transformations on vector maps

To demonstrate the resilience of our approach to a substantial amount of affine transformations, we take a fingerprinted vector map and apply a combination of rotation, scaling, and translation. More specifically, we rotate it by -30 degrees, then scale it by 120% or 80% in X and Y directions, respectively, followed by 100- and 200-pixel translation in X and Y directions, respectively. The resulting vector



(a)



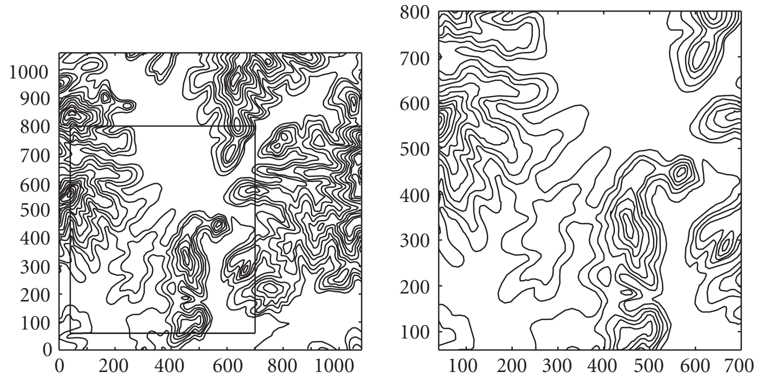
(b)

FIGURE 8.13. Histogram and the Gaussian approximation of the Z detection statistics for collusion attacks: (a) 2-user random interleaving collusion with Gaussian approximations $\mathcal{N}(0, 1.00)$ and $\mathcal{N}(28.45, 2.48)$, (b) 5-user averaging collusion with Gaussian approximations $\mathcal{N}(0, 1.00)$ and $\mathcal{N}(24.90, 1.00)$.

map is rendered in Figure 8.15a. In this test, we assume correct point correspondence is available. Thus, the super curve alignment method can be directly applied to register the original and the test curves and to extract the control points. Shown in Figure 8.15b is the registered vector map and Figure 8.15c are the detection statistics. We can see that the embedded fingerprint can survive affine transformations and the detection statistic with the correct fingerprint is high after the registration.

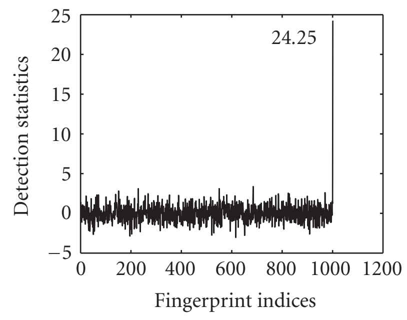
Resilience to vector-raster conversion

We now examine the resilience to vector-to-raster conversion coupled with possible affine transformations. Shown in Figure 8.16a is a fingerprinted vector map



(a)

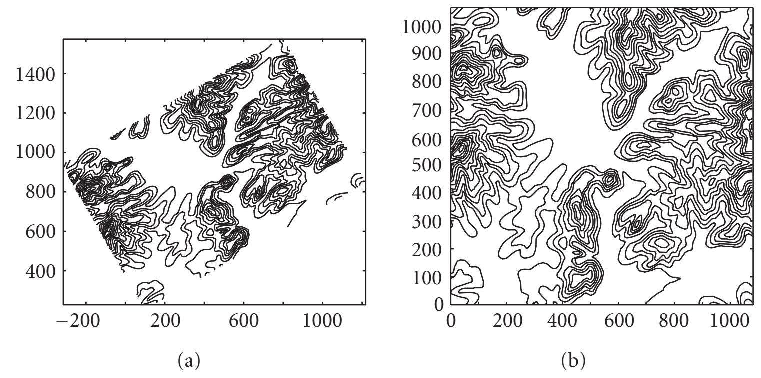
(b)



(c)

Figure 8.14. Cropping test on fingerprinted vector maps: (a) fingerprinted map with the cropping area marked by a rectangular box, (b) cropped map for fingerprint detection, (c) Z statistics.

after raster rendering as a 1100×1100 image and affine transformations. The affine transformation consists of 10-degree rotation, 80% and 140% scaling in X and Y directions, and 10- and 20-pixel translation in X and Y directions, respectively. As the point correspondence is no longer directly available after the vector-raster conversion, we apply the proposed IAM algorithm to estimate the transform parameters and locate the sample points on test curves corresponding to those used in the embedding. After 15 iterations, we get the registered raster map as shown in Figure 8.16b and the detection statistics as shown in Figure 8.16c. The detection statistic results suggest that the embedded fingerprint is identified with high confidence. Using the same settings on the computing system as in Section 8.3.3, we measure the detection time required for this transformed raster map. The total duration of the detection process is 42.43 seconds, including 40.05 seconds for curve registration and fingerprint extraction from nine curves, and 2.38 seconds for evaluating Z statistics with 1000 fingerprint sequences.



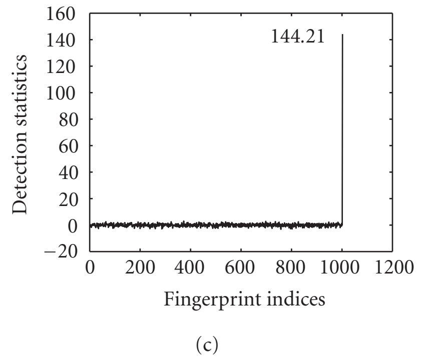


FIGURE 8.15. Affine transformation test on fingerprinted vector map: (a) the test map, (b) the aligned map, (c) Z statistics.

Similar to the collusion test, we plot in Figure 8.17 the histogram and the Gaussian approximation of the Z statistics for this vector-raster conversion test combined with geometric transformations. The transform parameters are randomly selected from the following ranges with a uniform distribution: $-20 \sim +20$ degrees of rotation, $60\% \sim 140\%$ scaling in X and Y directions, and $20 \sim 40$ pixels' translation in X and Y directions. As the impact from registration and resampling errors are not always i.i.d. Gaussian distributed, we observe a variance larger than 1 for the Z values in the fingerprint presence case.

Resilience to curve smoothing

Similar to lowpass filtering for images, curve smoothing can be applied to topographic maps as an attempt to remove the embedded fingerprint. In order to demonstrate the resilience of the proposed method to curve smoothing, we first render a fingerprinted vector map as a raster map and then apply a moving average filter to each marked curve. A curve point with coordinate (r_{x_i}, r_{y_i}) will be

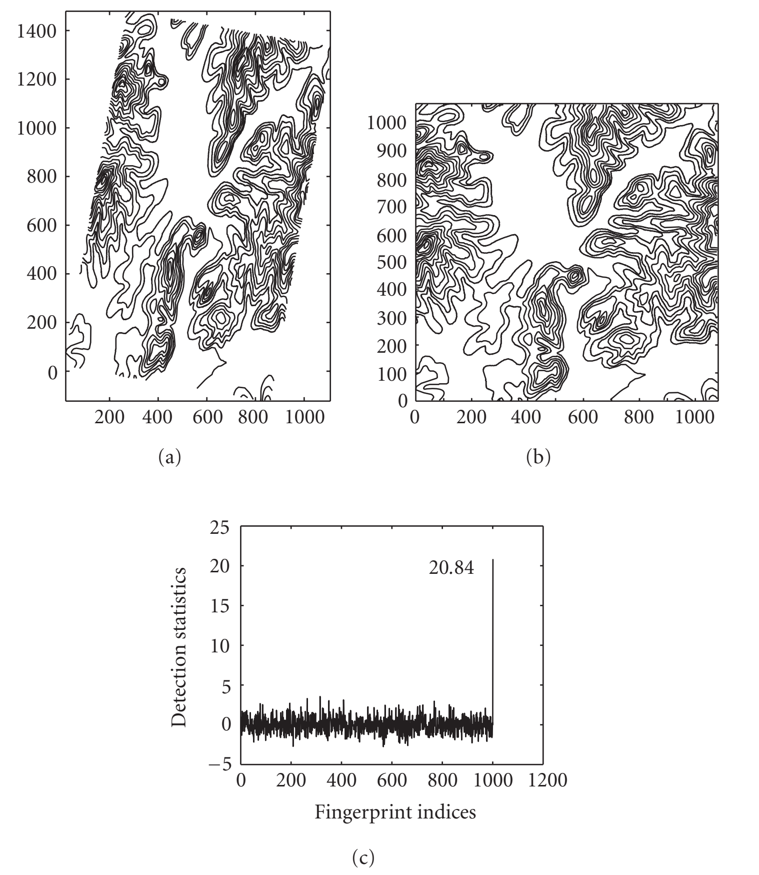


Figure 8.16. Affine transformation test on fingerprinted raster map: (a) the test map, (b) the aligned map, (c) Z statistics.

replaced by a new point, whose coordinate $(r_{x_i}^{(s)}, r_{y_i}^{(s)})$ is obtained by

$$\begin{aligned} r_{x_i}^{(s)} &= \frac{1}{2S+1}\sum_{j=-S}^{S} r_{x_{i+j}}, \\ r_{y_i}^{(s)} &= \frac{1}{2S+1}\sum_{j=-S}^{S} r_{y_{i+j}}, \end{aligned} \tag{8.25}$$

where $2S + 1$ is the filter length. Finally, we apply the proposed IAM algorithm to these smoothed curves and compute the detection statistics. Two different filter lengths, 5 and 21, are used in our experiments. As shown in Figure 8.18, the detection statistic with the correct fingerprint under 5-point averaging is 24.61 and under 21-point averaging is 9.45. Indeed, the fingerprint is weakened by the smoothing operation, but the detection statistics are still well above the threshold

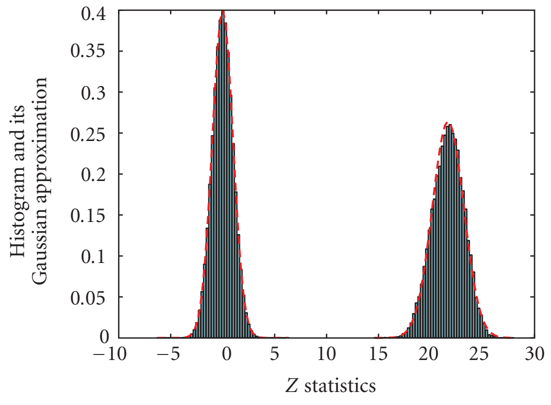


FIGURE 8.17. Histogram and the Gaussian approximation $\mathcal{N}(0, 1.00)$ and $\mathcal{N}(21.69, 2.28)$ of the Z detection statistics for the combined vector-raster conversion and geometric transformation.

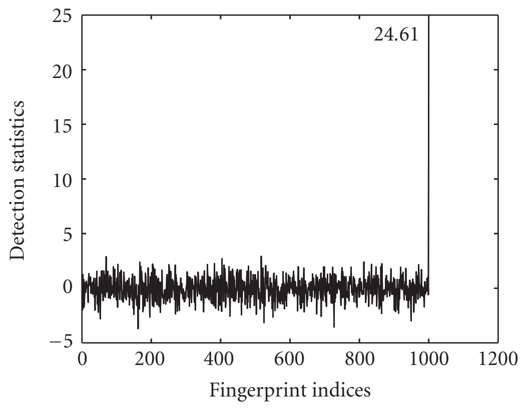
for a correct positive detection. In the latter case when a long filter is used in the smoothing attack, some visual details have been lost from the curves. This study shows that the proposed method is robust against curve smoothing, provided that the smoothing does not severely change the shape of the curves and the fingerprint sequence is sufficiently long to help the detector collect information for a positive detection.

Resilience to vector-raster-vector conversion

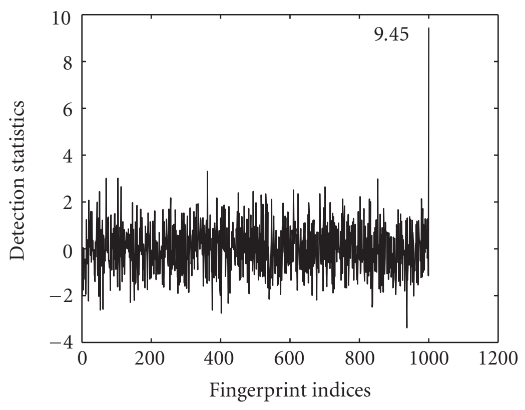
To demonstrate the resilience of our method to vector-raster-vector conversion, we first render a fingerprinted vector map as a raster map, and then uniformly sample it to be a new vector map with the same number of vector points as prior to the conversion but at different sampling locations. In the detection process, we first render this “new” vector map as a raster map by linear interpolation and then apply our IAM algorithm. From the high detection statistic shown in Figure 8.19a, we can see that our approach is robust against the vector-raster-vector attack.

Resilience to point deletion in vector and raster maps

As we have seen throughout the chapter, traitor tracing applications usually involve adversaries who have strong incentives to remove fingerprints. Attackers may delete a certain number of points from a fingerprinted vector/raster map while keeping similar shapes of its contour lines. Shown in Figures 8.19b and 8.19c are detection statistics after point deletion in a fingerprinted vector and raster map, respectively. For the vector map, 20% of points are randomly chosen and removed from each fingerprinted curve, while in the raster map 70% of the black pixels on the curve are randomly chosen and removed. We can see that the embedded fingerprints can survive point deletion applied to both vector maps and raster maps.



(a)



(b)

FIGURE 8.18. Curve smoothing test on fingerprinted raster maps: (a) Z statistics for window size 5, (b) Z statistics for window size 21.

Similar to the last test, linear interpolation and the IAM algorithm are used in the fingerprint detection.

Resilience to printing-and-scanning

To show the robustness of our approach against the printing-and-scanning attack, we render a fingerprinted vector map by taking a screen shot of its display in Matlab, print out the image using an HP laser printer, and then scan it back as a binary image by a Canon scanner with 360 dpi resolution. Preprocessing before detection includes a thinning operation to extract one-pixel wide skeletons from the scanned curves that are usually several-pixel wide after high resolution scanning. As we can

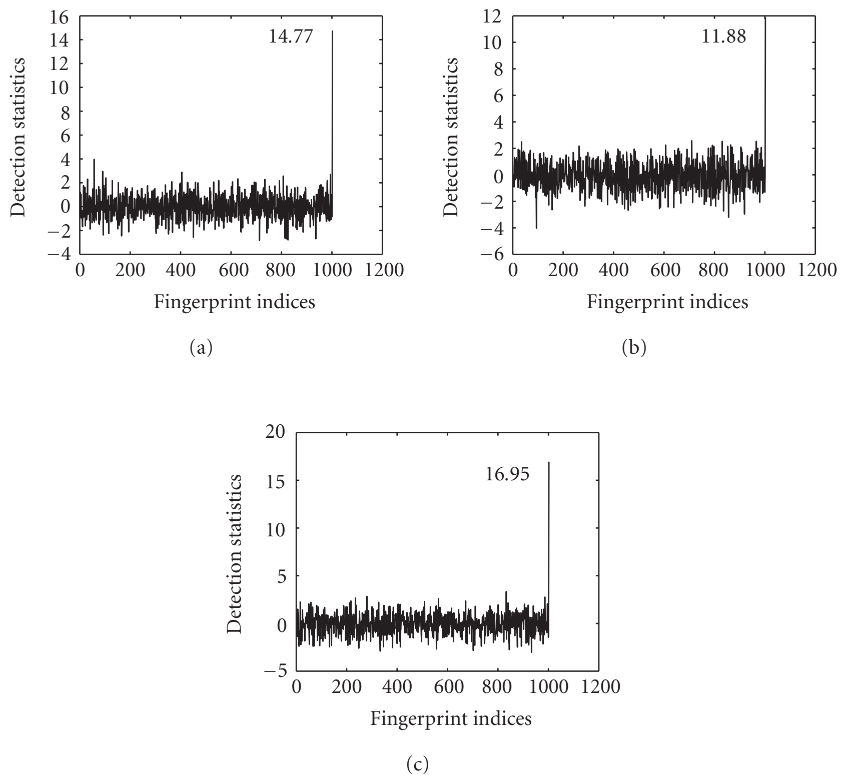


Figure 8.19. Detection results after various attacks: (a) Z statistics for vector-raster-vector conversion, (b) Z statistics for point deletion in vector maps, (c) Z statistics for point deletion in raster maps.

see from the detection results in Figure 8.20a, the fingerprint survives the printing-and-scanning test and gives a reliable positive detection with the detection statistic much higher than the detection threshold.

To further examine our approach under the combined attack of collusion and printing and scanning, we first generate a colluded map by averaging coordinates of the control points from four users' fingerprinted maps, then render and print it out, and scan it back as a binary image. From the detection statistics in Figure 8.20b, we can see that the embedded fingerprints from all the four colluders can be correctly identified after this combined attack involving collusion, vector-raster conversion, filtering, and affine transformations.

8.5. Chapter summary

In this chapter, we have presented a new data hiding algorithm for curves by parameterizing a curve using the B-spline model and adding spread-spectrum

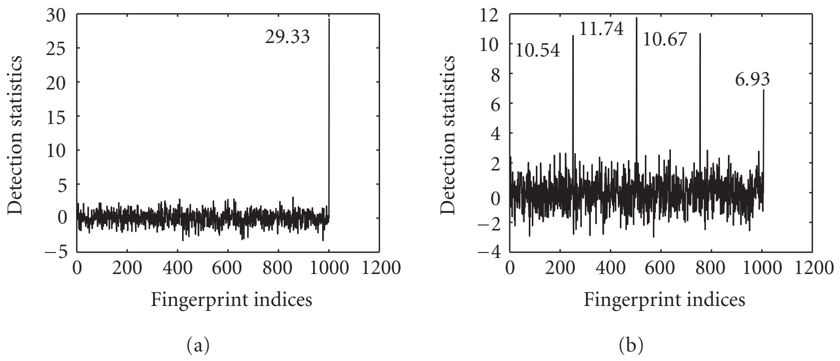


FIGURE 8.20. Printing-and-scanning test: (a) Z statistics for printing and scanning, (b) Z statistics for 4-user averaging collusion combined with printing and scanning.

sequences to curve parameters. In conjunction with the basic embedding and detection techniques, we have proposed an iterative alignment-minimization algorithm to allow for robust fingerprint detection under unknown geometric transformations and in absence of explicit point correspondence. We have demonstrated the fidelity of our method as well as its robustness against collusion, cropping, affine transformations, vector-raster and vector-raster-vector conversions, curve smoothing, point deletion, printing and scanning, and their combinations. Our work has shown the feasibility of the proposed algorithm in fingerprinting applications for tracing and tracking topographic maps as well as writings/drawings from pen-based inputs. The protection of such documents has increasing importance to emerging digital operations in government, military, intelligence, and commerce.

Bibliography

[1] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall, *MPEG Video Compression Standard*, Chapman & Hall, New York, NY, USA, 1997.

[2] J. Chen, U. Koc, and K. J. R. Liu, *Design of Digital Video Coding Systems*, Marcel Dekker, New York, NY, USA, 2002.

[3] K. Ngan, C. Yap, and K. Tan, *Video Coding for Wireless Communication Systems*, Marcel Dekker, New York, NY, USA, 2001.

[4] A. Puri and T. Chen, Eds., *Multimedia Systems, Standards, and Networks*, Marcel Dekker, New York, NY, USA, 2000.

[5] MPEG committee, "MPEG-7 overview," ISO/IEC JTC1/SC29/WG11/N5525.

[6] S. Siwek, "Copyright industries in the U.S. economy, the 2002 report," Tech. Rep., International Intellectual Property Alliance (IIPA), Washington, DC, USA, 2002.

[7] The International Intellectual Property Alliance (IIPA).

[8] MPEG committee, "MPEG-21 overview," ISO/IEC JTC1/SC29/WG11/N4801.

[9] Secure Digital Music Initiative (SDMI), 2000.

[10] MPEG4 IPMP FPDAM, ISO/IEC 14 496-1: 2001/AMD3, ISO/IEC JTC 1/SC 29/WG11 N4701, March 2002.

[11] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *Proc. 4th ACM International Conference on Multimedia (MULTIMEDIA '96)*, pp. 219–229, Boston, Mass, USA, November 1996.

[12] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *IEEE Trans. Multimedia*, vol. 5, no. 1, pp. 118–129, 2003.

[13] J. Wen, M. Severa, W. Zeng, M. H. Luttrell, and W. Jin, "A format-compliant configurable encryption framework for access control of video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 545–557, 2002.

[14] J. Song, R. Poovendran, W. Trappe, and K. J. R. Liu, "Dynamic key distribution scheme using data embedding for secure multimedia multicast," in *Security and Watermarking of Multimedia Contents III*, vol. 4314 of *Proceedings of SPIE*, pp. 618–628, Santa Clara, Calif, USA, January 2001.

[15] W. Trappe, J. Song, R. Poovendran, and K. J. R. Liu, "Key distribution for secure multimedia multicasts via data embedding," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, vol. 3, pp. 1449–1452, Salt Lake City, Utah, USA, May 2001.

[16] W. Trappe, J. Song, R. Poovendran, and K. J. R. Liu, "Key management and distribution for secure multimedia multicast," *IEEE Trans. Multimedia*, vol. 5, no. 4, pp. 544–557, 2003.

[17] I. J. Cox, J. A. Bloom, and M. L. Miller, *Digital Watermarking: Principles and Practice*, Morgan Kaufmann, San Francisco, Calif, USA, 2001.

[18] M. Wu and B. Liu, *Multimedia Data Hiding*, Springer, New York, NY, USA, 2003.

[19] M. D. Swanson, M. Kobayashi, and A. H. Tewfik, "Multimedia data-embedding and watermarking technologies," *Proc. IEEE*, vol. 86, no. 6, pp. 1064–1087, 1998.

[20] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—a survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, 1999.

[21] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proc. IEEE*, vol. 87, no. 7, pp. 1079–1107, 1999.

[22] M. D. Swanson, B. Zhu, B. Chau, and A. H. Tewfik, "Object-based transparent video watermarking," in *Proc. IEEE 1st Workshop on Multimedia Signal Processing (MMSP '97)*, pp. 369–374, Princeton, NJ, USA, June 1997.

[23] I. J. Cox, J. Kilian, F. T. Leighton, and T. G. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Processing*, vol. 6, no. 12, pp. 1673–1687, 1997.

[24] C. I. Podilchuk and W. Zeng, "Image-adaptive watermarking using visual models," *IEEE J. Select. Areas Commun.*, vol. 16, no. 4, pp. 525–539, 1998.

[25] X. G. Xia, C. G. Boncelet, and G. R. Arce, "Wavelet transform based watermark for digital images," *Optics Express*, vol. 3, no. 12, pp. 497–511, 1998.

[26] W. Zhu, Z. Xiong, and Y.-Q. Zhang, "Multiresolution watermarking for images and video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 4, pp. 545–550, 1999.

[27] D. Mukherjee, J. J. Chae, and S. K. Mitra, "A source and channel-coding framework for vector-based data hiding in video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 4, pp. 630–645, 2000.

[28] R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp, "Perceptual watermarks for digital images and video," *Proc. IEEE*, vol. 87, no. 7, pp. 1108–1126, 1999.

[29] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1996.

[30] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, Boca Raton, Fla, USA, 1996.

[31] W. Trappe and L. C. Washington, *Introduction to Cryptography with Coding Theory*, Prentice Hall, New York, NY, USA, 2001.

[32] J. A. Bloom, I. J. Cox, T. Kalker, J.-P. M. G. Linnartz, M. L. Miller, and C. B. S. Traw, "Copy protection for DVD video," *Proc. IEEE*, vol. 87, no. 7, pp. 1267–1276, 1999.

[33] J. Song, R. Poovendran, W. Trappe, and K. J. R. Liu, "Dynamic key distribution scheme using data embedding for secure multimedia multicast," in *Security and Watermarking of Multimedia Contents III*, vol. 4314 of *Proceedings of SPIE*, pp. 618–628, San Jose, Calif, USA, January 2001.

[34] P. Yin, B. Liu, and H. H. Yu, "Error concealment using data hiding," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, vol. 3, pp. 1453–1456, Salt Lake City, Utah, USA, May 2001.

[35] P. Yin, M. Wu, and B. Liu, "Robust error-resilient approach for MPEG video transmission over internet," in *Visual Communications and Image Processing*, vol. 4671 of *Proceedings of SPIE*, pp. 103–111, San Jose, Calif, USA, January 2002.

[36] W. Zeng and 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, 1999.

[37] M. Wu, H. Yu, and A. Gelman, "Multi-level data hiding for digital image and video," in *Multimedia Systems and Applications II*, vol. 3845 of *Proceedings of SPIE*, pp. 10–21, Boston, Mass, USA, September 1999.

[38] P. Moulin and J. A. O'Sullivan, "Information-theoretic analysis of information hiding," *IEEE Trans. Inform. Theory*, vol. 49, no. 3, pp. 563–593, 2003.

[39] B. Chen and G. W. Wornell, "Quantization index modulation: a class of provably good methods for digital watermarking and information embedding," *IEEE Trans. Inform. Theory*, vol. 47, no. 4, pp. 1423–1443, 2001.

[40] M. H. M Costa, "Writing on dirty paper," *IEEE Trans. Inform. Theory*, vol. 29, no. 3, pp. 439–441, 1983.

[41] M. Kesal, M. K. Mihcak, R. Koetter, and P. Moulin, "Iteratively decodable codes for watermarking applications," in *Proc. 2nd International Symposium on Turbo Codes and Related Topics (ISTC '00)*, Brest, France, September 2000.

[42] J. J. Eggers, R. Bauml, R. Tzschoppe, and B. Girod, "Scalar Costa scheme for information embedding," *IEEE Trans. Signal Processing*, vol. 51, no. 4, pp. 1003–1019, 2003.

[43] C.-Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y. M. Lui, "Rotation, scale, and translation resilient watermarking for images," *IEEE Trans. Image Processing*, vol. 10, no. 5, pp. 767–782, 2001.

[44] J. Lubin, J. A. Bloom, and H. Cheng, "Robust content-dependent high-fidelity watermark for tracking in digital cinema," in *Security and Watermarking of Multimedia Contents V*, vol. 5020 of *Proceedings of SPIE*, pp. 536–545, Santa Clara, Calif, USA, June 2003.

[45] J. G. Proakis, *Digital Communications*, McGraw-Hill, New York, NY, USA, 4th edition, 2000.

[46] G. Csurka, F. Deguillaume, J. J. K. ÓRuanaidh, and T. Pun, "A Bayesian approach to affine transformation resistant image and video watermarking," in *Proc. 3rd Information Hiding Workshop (IHW '99)*, Lecture Notes in Computer Science, pp. 315–330, Hotel Elbflorenz, Dresden, Germany, September–October 1999.

[47] S. Pereira and T. Pun, "Fast robust template matching for affine resistant image watermarks," in *Proc. 3rd Information Hiding Workshop (IHW '99)*, vol. 1768 of *Lecture Notes in Computer Science*, pp. 207–218, Hotel Elbflorenz, Dresden, Germany, September–October 1999.

[48] N. F. Johnson, Z. Duric, and S. Jajodia, "Recovery of watermarks from distorted images," in *Proc. 3rd Information Hiding Workshop (IHW '99)*, pp. 361–375, Hotel Elbflorenz, Dresden, Germany, September–October 1999.

[49] M. Alghoniemy and A. H. Tewfik, "Geometric distortion correction through image normalization," in *Proc. IEEE International Conference on Multimedia and Expo (ICME '00)*, vol. 3, pp. 1291–1294, New York, NY, USA, July–August 2000.

[50] M. Wu, W. Trappe, Z. J. Wang, and K. J. R. Liu, "Collusion-resistant fingerprinting for multimedia," *IEEE Signal Processing Mag.*, vol. 21, no. 2, pp. 15–27, 2004.

[51] K. Su, D. Kundur, and D. Hatzinakos, "A content dependent spatially localized video watermark for resistance to collusion and interpolation attacks," in *Proc. IEEE International Conference on Image Processing (ICIP '01)*, vol. 1, pp. 818–821, Thessaloniki, Greece, October 2001.

[52] M. D. Swanson, B. Zhu, and A. H. Tewfik, "Multiresolution scene-based video watermarking using perceptual models," *IEEE J. Select. Areas Commun.*, vol. 16, no. 4, pp. 540–550, 1998.

[53] D. Kirovski and F. A. P. Petitcolas, "Blind pattern matching attack on watermarking systems," *IEEE Trans. Signal Processing*, vol. 51, no. 4, pp. 1045–1053, 2003.

[54] H. V. Poor, *An Introduction to Signal Detection and Estimation*, Springer, New York, NY, USA, 2nd edition, 1999.

[55] M. Wu, H. Yu, and B. Liu, "Data hiding in image and video. II. Designs and applications," *IEEE Trans. Image Processing*, vol. 12, no. 6, pp. 696–705, 2003.

[56] S. V. Voloshynovskiy, F. Deguillaume, S. Pereira, and T. Pun, "Optimal adaptive diversity watermarking with channel state estimation," in *Security and Watermarking of Multimedia Contents III*, vol. 4314 of *Proceedings of SPIE*, pp. 673–685, San Jose, Calif, USA, January 2001.

[57] J. R. Hernandez, M. Amado, and F. Perez-Gonzalez, "DCT-domain watermarking techniques for still images: detector performance analysis and a new structure," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 55–68, 2000, *Special Issue on Image and Video Processing for Digital Libraries*.

[58] Z. J. Wang, M. Wu, H. Zhao, W. Trappe, and K. J. R. Liu, "Anti-collusion forensics of multimedia fingerprinting using orthogonal modulation," *IEEE Trans. Image Processing*, vol. 14, no. 6, pp. 804–821, 2005.

[59] Z. J. Wang, M. Wu, H. Zhao, K. J. R. Liu, and W. Trappe, "Resistance of orthogonal Gaussian fingerprints to collusion attacks," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, vol. 4, pp. 724–727, Hong Kong, China, April 2003.

[60] H. Zhao, M. Wu, Z. J. Wang, and K. J. R. Liu, "Nonlinear collusion attacks on independent fingerprints for multimedia," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03)*, vol. 5, pp. 664–667, Hong Kong, China, April 2003.

[61] H. D. Brunk, *An Introduction to Mathematical Statistics*, Ginn and Company, Boston, Mass, USA, 1960.

[62] H. Stone, "Analysis of attacks on image watermarks with randomized coefficients," Tech. Rep. 96-045, NEC Research Institute, Princeton, NJ, USA, 1996.

[63] M. D. Swanson, B. Zhu, and A. H. Tewfik, "Transparent robust image watermarking," in *Proc. International Conference on Image Processing (ICIP '96)*, vol. 3, pp. 211–214, Lausanne, Switzerland, September 1996.

[64] H. A. Peterson, A. J. Ahumada Jr., and A. B. Watson, "An improved detection model for DCT coefficient quantization," in *Human Vision, Visual Processing, and Digital Display IV*, vol. 1913 of *Proceedings of SPIE*, pp. 191–201, Bellingham, Wash, USA, February 1993.

[65] A. B. Watson, "DCT quantization matrices visually optimized for individual images," in *Human Vision, Visual Processing, and Digital Display IV*, vol. 1913 of *Proceedings of SPIE*, pp. 202–216, San Jose, Calif, USA, February 1993.

[66] Joint Photographic Experts Group (JPEG).

[67] G. K. Wallace, "The JPEG still picture compression standard," *IEEE Trans. Consumer Electron.*, vol. 38, no. 1, pp. 18–34, 1992.

[68] B. Tao and B. Dickinson, "Adaptive watermarking in the DCT domain," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)*, vol. 4, pp. 2985–2988, Munich, Germany, April 1997.

[69] F. Ergun, J. Kilian, and R. Kumar, "A note on the limits of collusion-resistant watermarks," in *Advances in Cryptology (Eurocrypt '99)*, vol. 1592 of *Lecture Notes in Computer Science*, pp. 140–149, Prague, Czech Republic, May 1999.

[70] J. Kilian, F. T. Leighton, L. R. Matheson, T. G. Shamoon, R. Tajan, and F. Zane, "Resistance of digital watermarks to collusive attacks," Tech. Rep. TR-585-98, Department of Computer Science, Princeton University, Princeton, NJ, USA, 1998.

[71] S. He and M. Wu, "Improving collusion resistance of error correcting code based multimedia fingerprinting," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, vol. 2, pp. 1029–1032, Philadelphia, Pa, USA, March 2005.

[72] J. Su, J. Eggers, and B. Girod, “Capacity of digital watermarks subjected to an optimal collusion attack,” in *Proc. 10th European Signal Processing Conference (EUSIPCO ’00)*, Tampere, Finland, September 2000.

[73] S. Craver, B. Liu, and W. Wolf, “Histo-cepstral analysis for reverse-engineering watermarks,” in *Proc. 38th Conference on Information Sciences and Systems (CISS ’04)*, pp. 824–826, Princeton, NJ, USA, March 2004.

[74] H. A. David, *Order Statistics*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1981.

[75] W. Gander and W. Gautschi, “Adaptive quadrature—revisited,” *BIT Numerical Mathematics*, vol. 40, no. 1, pp. 84–101, 2000.

[76] H. Zhao, M. Wu, Z. J. Wang, and K. J. R. Liu, “Performance of detection statistics under collusion attacks on independent multimedia fingerprints,” in *Proc. IEEE International Conference on Multimedia and Expo (ICME ’03)*, vol. 1, pp. 205–208, Baltimore, Md, USA, July 2003.

[77] D. Boneh and J. Shaw, “Collusion-secure fingerprinting for digital data,” *IEEE Trans. Inform. Theory*, vol. 44, no. 5, pp. 1897–1905, 1998.

[78] F. Zane, “Efficient watermark detection and collusion security,” in *Proc. Financial Cryptography (FC ’00)*, vol. 1962 of *Lecture Notes in Computer Science*, pp. 21–32, Anguilla, British West Indies, February 2000.

[79] M. Wu and B. Liu, “Data hiding in image and video. I. Fundamental issues and solutions,” *IEEE Trans. Image Processing*, vol. 12, no. 6, pp. 685–695, 2003.

[80] W. Trappe, M. Wu, Z. J. Wang, and K. J. R. Liu, “Anti-collusion fingerprinting for multimedia,” *IEEE Trans. Signal Processing*, vol. 51, no. 4, pp. 1069–1087, 2003, Special Issue on Signal Processing for Data Hiding in Digital Media.

[81] M. K. Simon, S. M. Hinedi, and W. C. Lindsey, “Appendix 3b: the Gaussian integral $q(x)$,” in *Digital Communication Techniques: Signal Design and Detection*, Prentice Hall, Englewood Cliffs, NJ, USA, 1995.

[82] H. Zhao, M. Wu, Z. J. Wang, and K. J. R. Liu, “Forensic analysis of nonlinear collusion attacks for multimedia fingerprinting,” *IEEE Trans. Image Processing*, vol. 14, no. 5, pp. 646–661, 2005.

[83] A. Herrigel, J. J. K. ÓRuanaidh, H. Petersen, S. Pereira, and T. Pun, “Secure copyright protection techniques for digital images,” in *Proc. 2nd Information Hiding Workshop (IHW ’98)*, vol. 1525 of *Lecture Notes in Computer Science*, pp. 169–190, Portland, Ore, USA, April 1998.

[84] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill, New York, NY, USA, 1989.

[85] J. A. Aslam and A. Dhagat, “Searching in the presence of linearly bounded errors,” in *Proc. 23rd Annual ACM Symposium on Theory of Computing (STOC ’91)*, pp. 486–493, ACM Press, New Orleans, La, USA, May 1991.

[86] D.-Z. Du, G.-L. Xue, S.-Z. Sun, and S.-W. Cheng, "Modifications of competitive group testing," *SIAM Journal on Computing*, vol. 23, no. 1, pp. 82–96, 1994.

[87] D.-Z. Du and H. Park, "On competitive group testing," *SIAM Journal on Computing*, vol. 23, no. 5, pp. 1019–1025, 1994.

[88] M. Wu and B. Liu, "Modulation and multiplexing techniques for multimedia data hiding," in *Multimedia Systems and Applications IV*, vol. 4518 of *Proceedings of SPIE*, pp. 228–238, Denver, Colo, USA, August 2001.

[89] E. Lehmann, *Adaptive Filter Theory*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.

[90] N. Balakrishnan and C. Rao, *Order Statistics: Theory and Methods*, Elsevier Science, Amsterdam, the Netherlands, 1998.

[91] H. Stark and J. Woods, *Probability and Random Processes with Applications to Signal Processing*, Prentice Hall, New York, NY, USA, 3rd edition, 2002.

[92] Y. Yacobi, "Improved Boneh-Shaw content fingerprinting," in *Topics in Cryptology—CT-RSA 2001, The Cryptographer's Track at RSA Conference (CT-RSA '01)*, vol. 2020 of *Lecture Notes in Computer Science*, pp. 378–391, San Francisco, Calif, USA, April 2001.

[93] W. Trappe, M. Wu, and K. J. R. Liu, "Collusion-resistant fingerprinting for multimedia," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, vol. 4, pp. 3309–3312, Orlando, Fla, USA, May 2002.

[94] J. H. Dinitz and D. R. Stinson, *Contemporary Design Theory: A Collection of Surveys*, John Wiley & Sons, New York, NY, USA, 1992.

[95] C. J. Colbourn and J. H. Dinitz, *The CRC Handbook of Combinatorial Designs*, CRC Press, Boca Raton, Fla, USA, 1996.

[96] J. Dittmann, P. Schmitt, E. Saar, J. Schwenk, and J. Ueberberg, "Combining digital watermarks and collusion secure fingerprints for digital images," *SPIE Journal of Electronic Imaging*, vol. 9, no. 4, pp. 456–467, 2000.

[97] C. C. Lindner and C. A. Rodger, *Design Theory*, CRC Press, Boca Raton, Fla, USA, 1997.

[98] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge University Press, Cambridge, UK, 1994.

[99] S. F. Yau and Y. Bresler, "Maximum likelihood parameter estimation of superimposed signals by dynamic programming," *IEEE Trans. Signal Processing*, vol. 41, no. 2, pp. 804–820, 1993.

[100] I. Ziskind and M. Wax, "Maximum likelihood localization of multiple sources by alternating projection," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, no. 10, pp. 1553–1560, 1988.

[101] T. G. Manickam, R. J. Vaccaro, and D. W. Tufts, "A least-squares algorithm for multipath time-delay estimation," *IEEE Trans. Signal Processing*, vol. 42, no. 11, pp. 3229–3233, 1994.

[102] Z. J. Wang, M. Wu, W. Trappe, and K. J. R. Liu, "Anti-collusion of group-oriented fingerprinting," in *Proc. IEEE International Conference on Multimedia and Expo (ICME '03)*, vol. 2, pp. 217–220, Baltimore, Md, USA, July 2003.

[103] Z. Li and W. Trappe, "Collusion-resistant fingerprints from WBE sequence sets," to appear in *Proc. IEEE International Conference on Communications (ICC '05)*, Seoul, Korea, May 2005.

[104] M. Ajtai, "The shortest vector problem in L_2 is NP-Hard for randomized reductions," in *Proc. 30th Annual ACM Symposium on Theory of Computing (STOC '98)*, pp. 10–19, Dallas, Tex, USA, May 1998.

[105] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Mathematics of Computation*, vol. 44, no. 4, pp. 463–471, 1985.

[106] S. Paul, *Multicast on the Internet and Its Application*, Kluwer Academic, Boston, Mass, USA, 1998.

[107] R. C. Chalmers and K. C. Almeroth, "Modeling the branching characteristics and efficiency gains in global multicast trees," in *Proc. 20th IEEE Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '01)*, vol. 1, pp. 449–458, Anchorage, Alaska, USA, April 2001.

[108] H. Zhao and K. J. R. Liu, "Fingerprint multicast in secure video streaming," to appear in *IEEE Trans. Image Processing*, Fall 2005.

[109] C. Pfleeger, *Security in Computing*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1996.

[110] I. J. Cox and J.-P. M. G. Linnartz, "Some general methods for tampering with watermarks," *IEEE J. Select. Areas Commun.*, vol. 16, no. 4, pp. 587–593, 1998.

[111] F. Hartung, J. K. Su, and B. Girod, "Spread spectrum watermarking: Malicious attacks and counterattacks," in *Security and Watermarking of Multimedia Contents, Electronic Imaging*, vol. 3657 of *Proceedings of SPIE*, pp. 147–158, San Jose, Calif, USA, April 1999.

[112] H.-H. Chu, L. Qiao, and K. Nahrstedt, "A secure multicast protocol with copyright protection," *ACM SIGCOMM Computer Communications Review*, vol. 32, no. 2, pp. 42–60, 2002.

[113] D. Kundur and K. Karthik, "Video fingerprinting and encryption principles for digital rights management," *Proc. IEEE*, vol. 92, no. 6, pp. 918–932, 2004.

[114] I. Brown, C. Perkins, and J. Crowcroft, "Watercasting: Distributed watermarking of multicast media," in *Proc. 1st International Workshop on Networked Group Communication (NGC '99)*, pp. 286–300, Pisa, Italy, November 1999.

[115] G. Caronni and C. Schuba, "Enabling hierarchical and bulk-distribution for watermarked content," in *Proc. 17th Annual Computer Security Applications Conference (ACSAC '01)*, pp. 277–285, New Orleans, La, USA, December 2001.

[116] D. Konstantas and D. Thanos, “Commercial dissemination of video over open networks: issues and approaches,” Tech. Rep., Object Systems Group, Center Universitaire d’Informatique of University of Geneva, Geneva, Switzerland, 2000.

[117] R. Parviainen and R. Parnes, “Enabling hierarchical and bulk-distribution for watermarked content,” in *Proc. IFIP TC6/TC11 International Conference on Communications and Multimedia Security Issues*, vol. 192, Darmstadt, Germany, May 2001.

[118] P. Judge and M. Ammar, “WHIM: Watermarking multicast video with a hierarchy of intermediaries,” in *Proc. 10th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV ’00)*, Chapel Hill, NC, USA, June 2000.

[119] T. Wu and S. F. Wu, “Selective encryption and watermarking of MPEG video,” in *Proc. International Conference on Imaging Science, Systems, and Technology (CISST ’97)*, Las Vegas, Nev, USA, June–July 1997.

[120] Z. J. Wang, M. Wu, W. Trappe, and K. J. R. Liu, “Group-oriented fingerprinting for multimedia forensics,” *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 14, pp. 2153–2173, 2004.

[121] H. Zhao and K. J. R. Liu, “Bandwidth efficient fingerprint multicast for video streaming,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’04)*, vol. 5, pp. 849–852, Montreal, Quebec, Canada, May 2004.

[122] M. Wu and Y. Mao, “Communication-friendly encryption of multimedia,” in *IEEE Workshop on Multimedia Signal Processing (MMSP ’02)*, pp. 292–295, St. Thomas, Virgin Islands, USA, December 2002.

[123] L. Qiao and K. Nahrstedt, “A new algorithm for MPEG video encryption,” in *Proc. International Conference on Imaging Science, Systems and Technology (CISST ’97)*, pp. 21–29, Las Vegas, Nev, USA, June 1997.

[124] H. Zhao and K. J. R. Liu, “A secure multicast scheme for anti-collusion fingerprinted video,” in *Proc. IEEE Global Telecommunications Conference (GLOBECOM ’04)*, vol. 2, pp. 571–575, Dallas, Tex, USA, November–December 2004.

[125] J. C.-I. Chuang and M. A. Sirbu, “Pricing multicast communication: A cost-based approach,” *Telecommunication Systems*, vol. 17, no. 3, pp. 281–297, 2001.

[126] F. Hartung and B. Girod, “Watermarking of uncompressed and compressed video,” *Signal Processing*, vol. 66, no. 3, pp. 283–301, 1998.

[127] H.-H. Chang, T. Chen, and K.-S. Kan, “Watermarking 2D/3D graphics for copyright protection,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP ’03)*, vol. 4, pp. 720–723, Hong Kong, China, April 2003.

[128] M. Barni, F. Bartolini, A. Piva, and F. Salucco, "Robust watermarking of cartographic images," *EURASIP Journal on Applied Signal Processing*, vol. 2002, no. 2, pp. 197–208, 2002.

[129] V. Solachidis and I. Pitas, "Watermarking polygonal lines using Fourier descriptors," *IEEE Comput. Graph. Appl.*, vol. 24, no. 3, pp. 44–51, 2004.

[130] R. Ohbuchi, H. Ueda, and S. Endoh, "Watermarking 2D vector maps in the mesh-spectral domain," in *Proc. Shape Modeling International (SMI '03)*, pp. 216–228, Seoul, Korea, May 2003.

[131] J. Zhao and E. Koch, "Embedding robust labels into images for copyright protection," in *Proc. International Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Technologies*, pp. 242–251, Vienna, Austria, August 1995.

[132] M. Wu, E. Tang, and B. Lin, "Data hiding in digital binary image," in *Proc. IEEE International Conference on Multimedia and Expo (ICME '00)*, vol. 1, pp. 393–396, New York, NY, USA, July–August 2000.

[133] M. Wu and B. Liu, "Data hiding in binary image for authentication and annotation," *IEEE Trans. Multimedia*, vol. 6, no. 4, pp. 528–538, 2004.

[134] K. Matsui and K. Tanaka, "Video-steganography: how to secretly embed a signature in a picture," *IMA Intellectual Property Project Proceedings*, vol. 1, no. 1, pp. 187–205, 1994.

[135] N. F. Maxemchuk and S. Low, "Marking text documents," in *Proc. IEEE International Conference on Image Processing (ICIP '97)*, vol. 3, pp. 13–13, Santa Barbara, Calif, USA, October 1997.

[136] R. Ohbuchi, H. Masuda, and M. Aono, "A shape-preserving data embedding algorithm for NURBS curves and surfaces," in *Proc. Computer Graphics International (CGI '99)*, pp. 180–188, Canmore, Canada, June 1999.

[137] J. J. Lee, N. I. Cho, and J. W. Kim, "Watermarking for 3D NURBS graphic data," in *Proc. IEEE Workshop on Multimedia Signal Processing (MMSP '02)*, pp. 304–307, St. Thomas, Virgin Islands, USA, December 2002.

[138] J. J. Lee, N. I. Cho, and S. U. Lee, "Watermarking algorithms for 3D nurbs graphic data," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 14, pp. 2142–2152, 2004.

[139] K. H. Ko, T. Maekawa, N. M. Patrikalakis, H. Masuda, and F.-E. Wolter, "Shape intrinsic fingerprints for free-form object matching," in *Proc. 8th ACM Symposium on Solid Modeling and Applications*, pp. 196–207, Seattle, Wash, USA, June 2003.

[140] H. Gou and M. Wu, "Data hiding in curves with applications to map fingerprinting," *to appear in IEEE Trans. on Image Processing*, Special Issue on Secure Media, October 2005.

[141] H. Gou and M. Wu, "Data hiding in curves for collusion-resistant digital fingerprinting," in *Proc. IEEE International Conference on Image Processing (ICIP '04)*, vol. 1, pp. 51–54, Singapore, October 2004.

[142] H. Gou and M. Wu, "Fingerprinting curves," in *Proc. IEEE International Workshop on Digital Watermarking (IWDW '04)*, pp. 13–28, Seoul, Korea, October–November 2004.

[143] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, USA, 1989.

[144] G. E. Farin, *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide*, Academic Press, New York, NY, USA, 4th edition, 1997.

[145] D. Kirovski, H. S. Malvar, and Y. Yacobi, "Multimedia content screening using a dual watermarking and fingerprinting system," in *Proc. ACM Multimedia*, pp. 372–381, Juan Les Pins, France, December 2002.

[146] Z. Huang and F. S. Cohen, "Affine-invariant B-spline moments for curve matching," *IEEE Trans. Image Processing*, vol. 5, no. 10, pp. 1473–1480, 1996.

[147] C. A. Cabrelli and U. M. Molter, "Automatic representation of binary images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 12, pp. 1190–1196, 1990.

[148] H. S. M. Coxeter, *Introduction to Geometry*, John Wiley & Sons, New York, NY, USA, 2nd edition, 1969.

[149] F. S. Cohen and J.-Y. Wang, "Part I: Modeling image curves using invariant 3-D object curve models-a path to 3-D recognition and shape estimation from image contours," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, no. 1, pp. 1–12, 1994.

[150] E. Belogay, C. A. Cabrelli, U. M. Molter, and R. Shonkwiler, "Calculating the Hausdorff distance between curves," *Information Processing Letters*, vol. 64, no. 1, pp. 17–22, 1997.

[151] M. Xia and B. Liu, "Image registration by "Super-curves"," *IEEE Trans. Image Processing*, vol. 13, no. 5, pp. 720–732, 2004.

[152] U. Varshney, "Multicast over wireless networks," *Communications of the ACM*, vol. 45, no. 12, pp. 31–37, 2002.

Index