Anti-Collusion Fingerprinting for Multimedia

Wade Trappe, Min Wu, Zhen Wang, and K. J. Ray Liu

Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
E-mail: wxt, minwu, zhenwang, kjrliu @eng.umd.edu

Abstract

Digital fingerprinting is a technique for identifying users who might try to use multimedia content for unintended purposes, such as redistribution. These fingerprints are typically embedded into the content using watermarking techniques that are designed to be robust to a variety of attacks. A cost-effective attack against such digital fingerprints is collusion, where several differently marked copies of the same content are combined to disrupt the underlying fingerprints. In this paper, we investigate the problem of designing fingerprints that can withstand collusion and allow for the identification of colluders. We begin by introducing the collusion problem for additive embedding. We then study the effect that averaging collusion has upon orthogonal modulation. We introduce an efficient detection algorithm for identifying the fingerprints associated with K colluders that requires $\mathcal{O}(K\log(n/K))$ correlations for a group of n users. We next develop a fingerprinting scheme based upon code modulation that does not require as many basis signals as orthogonal modulation. We propose a new class of codes, called anti-collusion codes (ACC), which have the property that the composition of any subset of K or fewer codevectors is unique. Using this property, we can therefore identify groups of K or fewer colluders. We present a construction of binary-valued ACC under the logical AND operation that uses the theory of combinatorial designs and is suitable for both the on-off keying and antipodal form of binary code modulation. In order to accommodate n users, our code construction requires only $\mathcal{O}(\sqrt{n})$ orthogonal signals for a given number of colluders. We introduce four different detection strategies that can be used with our ACC for identifying a suspect set of colluders. We demonstrate the performance of our ACC for fingerprinting multimedia and identifying colluders through experiments using Gaussian signals and real images.

1 Introduction

The advancement of multimedia technologies, coupled with the development of an infrastructure of ubiquitous broadband communication networks, promises to facilitate the development of a digital marketplace where a broad range of multimedia content, such as image, video, audio and speech, will be available. However, such an advantage also poses the challenging task of insuring that content is appropriately used. Before viable businesses can be established to market content on these networks, mechanisms must be in place to ensure that content is used for its intended purpose, and by legitimate users who have purchased appropriate distribution rights.

Although access control is an essential element to ensure that content is used by its intended recipients, it is not sufficient for protecting the value of the content. The protection provided by encryption disappears when the content is no longer in the protected domain. Regardless of whether the content is stored in an unencrypted format, or decrypted prior to rendering, it is feasible for users to access cleartext representations of the content. Users can then redistribute unencrypted representations, which affects the digital rights of the original media distributors.

In order to control the redistribution of content, digital fingerprinting is used to trace the consumers who use their content for unintended purposes [1–4]. These fingerprints can be embedded in multimedia content through a variety of watermarking techniques [2,5–10]. Conventional watermarking techniques are concerned with robustness against a variety of attacks such as filtering, but do not always address robustness to attacks mounted by a coalition of users with the same content that contains different marks. These attacks, known as *collusion* attacks, can provide a cost-effective approach to removing an identifying watermark. One of the simplest approaches to performing a collusion attack on multimedia is to average multiple copies of the content together [11]. Other collusion attacks might involve forming a new content by selecting different pixels or blocks from the different colluders' content. By gathering a large enough coalition of colluders, it is possible to sufficiently attenuate each of the colluders' identifying fingerprints and produce a new version of the content with no detectable fingerprints. It is therefore important to design fingerprints that are not only able to resist collusion, but are also able to identify the colluders and thereby provide a means to discourage attempts at collusion by the users.

1.1 Prior Art

One of the first works on designing fingerprints that are resistant to collusion was presented by Boneh and Shaw [4]. This work considered the problem of fingerprinting generic data that satisfied an underlying principle referred to as the *Marking Assumption*. A mark was modeled as a position in a digital object that could be in a finite number of different states, while a fingerprint was a collection of marks. A mark is considered detectable when a coalition of users do not all have the same mark in that position. The Marking Assumption states that the undetectable marks cannot arbitrarily be changed without rendering the object useless. However, it is considered possible for the colluding set to change a detectable mark to any other state or into an unreadable state. Under their collusion framework, Boneh and Shaw show that it is not possible to design *totally c-secure codes*, which are fingerprint codes that are capable of tracing at least one colluder out of a coalition of at most c colluders. Instead, they use randomization techniques to construct codes that are able to capture at least one colluder out of a coalition of at most c colluders with arbitrarily high probability.

A similar work was presented in [3]. This work is concerned with the distribution of large amounts of content, such as through television broadcasts, where each user has a decoder that contains a set of keys needed to decrypt the broadcast content. Users might collude to create a pirate decoder that consists of keys from some of the colluders' decoders. When a pirate decoder is captured, the goal is then to be able to trace and identify at least one of the colluders involved in creating the illicit device. Thus, the goal is not to trace the leakage of the content, but rather to trace the decryption keys needed to access the content. In this case, the challenge lies in reducing the size of the ciphertext from being linear in the amount of users.

In both of these cases, the ability to trace or identify a colluder relied on the fact that the identifying information cannot be blindly altered by the coalition. In particular, the construction of the fingerprinting schemes for generic data, as presented in [4], relies on the validity of the Marking Assumption. One key difference between generic data and multimedia data is that multimedia data is sensually robust to minor perturbations in the data values. This sensual robustness of multimedia data makes it desirable and feasible to embed digital fingerprints in the multimedia. Rather than attaching fingerprints in headers, embedding will tie the fingerprint with the host multimedia signal and make the fingerprint resilient to format conversion, compression, and other moderate distortions. Watermarking techniques that embed information in multimedia invisibly, such as those in [5, 6], can be used to embed digital fingerprints. While generic data may allow long fingerprint marks to be attached to them, the number of marks that can be embedded in multimedia data and accurately extracted after distortion by hostile parties is limited [12,13]. Thus, the long fingerprint codes proposed for generic data may not even be embeddable in multimedia data. The process of fingerprinting multimedia should, therefore, jointly consider the design of appropriate fingerprints, and the efficient and effective detection of these fingerprints. Furthermore, unlike Boneh and Shaw's assumption for generic data, where adversaries can easily manipulate the detectable marks to any value, different bits of fingerprint codes that are additively embedded in multimedia may not be easily identifiable and arbitrarily manipulated by colluders. For this reason, linear collusion attacks, such as averaging several fingerprinted signals, are often more feasible for multimedia [11].

The resistance of digital watermarks to linear collusion attacks has been studied [11, 14–17]. In [14], the original document is perturbed by the marking process to produce fingerprinted documents with a bounded distortion from the original document. They propose a collusion attack that consists of adding a small amount of noise to the average of K fingerprinted documents. In their analysis, they show that $O(\sqrt{N/\log N})$ adversaries are sufficient to defeat the underlying watermarks, where N is the dimensionality of the fingerprint. This result supports the claim of [15], where the watermarks are assumed to be uncorrelated Gaussian random vectors that are added to the host vector to produce fingerprinted documents.

Further work on the resistance of digital watermarks to collusion attacks was done in [16]. They consider a more general linear attack than [14], where the colluders employ multiple-input-single-output linear shift-invariant (LSI) filtering plus additive Gaussian noise to thwart the orthogonal fingerprints. Under the assumption that the fingerprints are independent and have identical statistical characteristics, they show that the optimal LSI attack involves each user weighting their marked document equally prior to the addition of additive noise. Additionally, they investigated an alternative fingerprinting strategy by embedding c-secure codes, such as described in [4, 18], and studied the amount of samples needed in order for the marking assumption to hold while maintaining a prescribed probability of falsely identifying a colluder. Their fingerprinting capacity study suggested that independent fingerprints require shorter sequence length than fingerprints constructed from c-secure codes.

Finally, a different perspective on collusion for multimedia was presented in [19]. A watermark conveying the access and usage policy is embedded in the multimedia content. Different users' media players use different variations of the watermark to correlate with marked content in detection. Each detection key is the sum of the watermark and a strong, independent Gaussian random vector that serves as a digital fingerprint. When an attacker breaks one device, obtains the detection key inside, and subtracts the key from the watermarked content, the watermark will not be completely removed from the attacked copy and a fingerprint signal will remain in the attacked copy that indicates the attacker's identity. The paper quantitatively analyzed the collusion resistance issues and discussed related problems of segmentation and key compression.

1.2 Paper Organization

The work of [16] suggested that independent, or orthogonal, fingerprints are advantageous to fingerprints built using collusion-secure codes. However, several disadvantages for orthogonal fingerprints remain, such as the high computational complexity required in detection, and the large storage requirements needed to maintain a library of fingerprints. In this paper, we address these disadvantages by proposing an efficient detection scheme for orthogonal fingerprints, and introducing a new class of codes for constructing fingerprints that require less storage resources. Our results are suitable for both averaging-based collusion attacks, and for collusion attacks that interleave values or pixels from differently marked versions of the same content. For the convenience of discussion, we will use images as an example, while the extension to audio or video is quite straightforward.

We begin, in Section 2, by describing multimedia fingerprinting, and introduce the problem of user collusion for a class of additive watermark schemes. We then review orthogonal modulation in Section 3, and examine the effect that collusion has upon orthogonal fingerprinting. In order to overcome the linear complexity associated with traditional detection schemes for orthogonal modulation, 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. However, storage demands remain high and it is desirable that we use fewer basis signals to accommodate a given amount of users. Therefore, in Section 4, we propose the use of a class of codes, which we call anti-collusion codes (ACC), that are used in code-modulated embedding. The resulting fingerprints are appropriate for different multimedia scenarios. The purpose of ACC is not only to resist collusion, but also to trace who the colluders are. The proposed ACC have the property that the composition of any subset of K or fewer codevectors is unique, which allows us to identify groups of K or fewer colluders. We present a construction of binary-valued ACC under the logical AND operation that uses the theory of combinatorial designs. For a given number of colluders, our code construction is able to accommodate n users, while requiring only $\mathcal{O}(\sqrt{n})$ basis vectors. We study the detector, and present four different strategies that may be employed for identifying a suspect set of colluders. We evaluate the performance of these detection strategies using simulations involving an abstract model consisting of Gaussian signals. We also examine the behavior of our fingerprints using actual images. Finally, we present conclusions in Section 5, and provide proofs of various claims in the appendices.

2 Fingerprinting and Collusion

In this section, we will review additive embedding, where a watermark signal is added to a host signal. 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, i.e. $s_j(l) = \alpha(l) w_j(l)$, 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 [6]. 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

$$\mathbf{y} = \mathbf{s}_j + \mathbf{d}. \tag{1}$$

The watermarks $\{\mathbf{w}_j\}$ are often chosen to be orthogonal noise-like signals [5], or are represented using a basis of orthogonal noiselike signals $\{\mathbf{u}_i\}$ via

$$\mathbf{w}_j = \sum_{i=1}^{B} b_{ij}\mathbf{u}_i, \tag{2}$$

where $b_{ij} \in \{0, 1\}$ or $b_{ij} \in \{\pm 1\}$ [20]. We will present detailed discussions on different ways to construct watermarks for fingerprinting purposes in Section 3 and Section 4.

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 *non-blind* 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 $\mathbf{x}$. Non-blind 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 non-blind 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 non-blind 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[1].

[1]Note that there are other types of watermarking schemes that do not suffer from interference from unknown host signals

In this case, the distortion can be modeled as $\mathbf{d} = \mathbf{x} + \mathbf{z}$.

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 [5, 6], the detection performance can be studied via the following simplified antipodal model:

$$\begin{cases} H_0 : y(l) = -s(l) + d(l) & (l = 1, ..., N) \qquad \text{if } b = -1 \\ H_1 : y(l) = +s(l) + d(l) & (l = 1, ..., N) \qquad \text{if } b = +1 \end{cases} \tag{3}$$

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 [22] with a detection statistic T_N given by

$$T_N = \mathbf{y}^T \mathbf{s} / \sqrt{\sigma_d^2 \cdot \|\mathbf{s}\|^2} \tag{4}$$

where $\mathbf{y} = [y(1), ..., y(N)]^T$, $\mathbf{s} = [s(1), ..., s(N)]^T$ 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

$$E(T_N) = b \cdot \sqrt{\|\mathbf{s}\|^2 / \sigma_d^2}. \tag{5}$$

If b is equally likely to be "-1" and "+1", the optimal (Bayesian) detection rule is to compare T_N with a threshold of zero to decide H_0 against H_1, in which case, 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 $X \sim \mathcal{N}(0, 1)$. The error probability can be reduced by raising the watermark-to-noise-ratio (WNR) $\|\mathbf{s}\|^2 / (N\sigma_d^2)$, or increasing the length N 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) [6]. 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 $\hat{\sigma}_d^2 = (\|\mathbf{y}\|^2 - \|\mathbf{s}\|^2)/N$.

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 [22, 23]. That is,

$$T'_N = \sum_{l=1}^{N} \frac{y(l) \cdot s(l)}{\sigma_{d(l)}^2} \Big/ \sqrt{\sum_{l=1}^{N} \frac{s^2(l)}{\sigma_{d(l)}^2}} \tag{6}$$

[12, 21]. Their appropriateness for fingerprinting and anti-collusion capabilities are to be investigated and will be addressed in our future work.

and

$$E(T'_N) \quad = \quad b\sqrt{\sum_{l=1}^{N} \frac{s^2(l)}{\sigma^2_{d(l)}}}. \tag{7}$$

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 [2]. In reality, the interference from the host signal as well as the noise introduced by many attacks and distortions are often non-Gaussian and non-stationary. Under these scenarios, an optimal detector can be derived by using a non-Gaussian and/or non-stationary noise model in the classic detection framework [22, 24]. For example, generalized matched filters have been proposed as watermark decoders for the generalized Gaussian channel model [25,26]. Channel estimation has also been used in conjunction with the generalized matched filter against fading and geometrical distortions with unknown parameters [25].

For concept proving purposes, in this paper we consider the simple noise model of independent Gaussian noise and use the correlator with normalized noise variance as described in (6). This simplification allows us to focus on the unique issues of fingerprint encoding and colluder detection for the anti-collusion fingerprinting problem. From a layered viewpoint on data hiding systems [20], 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 [25, 26] to improve the performance in the watermark detection layer and in turn enhance the overall performance.

Another model, used often for conveying ownership information [5,6], leads to a similar hypothesis testing problem described by:

$$\begin{cases} H_0 : y(l) = d(l) & (l = 1, ..., N) \quad \text{if watermark is absent} \\ H_1 : y(l) = s(l) + d(l) & (l = 1, ..., N) \quad \text{if watermark is present} \end{cases} \tag{8}$$

This is often referred as *On-Off Keying* (OOK). The detection statistic is the same as shown in (4) for additive white Gaussian noise (AWGN) or (6) for independent Gaussian noise with non-identical variance. 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 [22]. The probability of detection errors can be obtained accordingly.

In the following sections we shall examine collusion for fingerprints constructed using orthogonal modulation as well as using binary code modulation. When two parties with the same image (but fingerprinted differently) come together, they can compare the difference between the two fingerprinted images. The collusion attack generates a new image from the two fingerprinted images so that the traces of either fingerprint in the new image is removed or attenuated. For fingerprinting through additive embedding, this can be

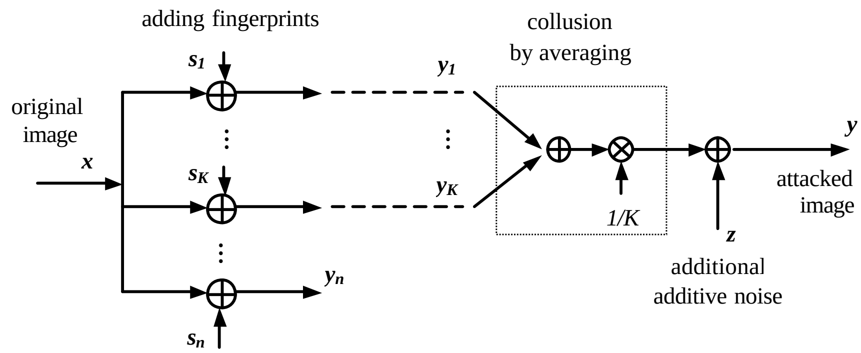


Figure 1: Model for collusion by averaging

done by averaging the two fingerprinted images $\lambda_1\mathbf{y}_1 + \lambda_2\mathbf{y}_2$ where $\lambda_1 + \lambda_2 = 1$, so that the energy of each of the fingerprints is reduced by a factor of ${\lambda_i}^2$. The requirement that $\lambda_1 + \lambda_2 = 1$ is necessary in order to retain the pixel magnitudes of the original image. As a result of this weighted average, the detection statistic with respect to the i-th fingerprint is scaled by a factor of λ_i. In a K-colluder averaging-collusion the watermarked content signals $\mathbf{y}_j$ are combined according to $\sum_{j=1}^{K} \lambda_j \mathbf{y}_j$. Alternatively, the new image can be formed by taking part of the pixels or transform coefficients from each of the two images $\mathbf{\Lambda}\mathbf{y}_1 + (\mathbf{I} - \mathbf{\Lambda})\mathbf{y}_2$ where $\mathbf{I}$ is the $N \times N$ identity matrix and $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_N)$ with $\lambda_i \in \{0, 1\}$. In terms of the effects on the energy reduction of the original fingerprints and the effect it has upon the detection performance, this alternating type of collusion is similar to the averaging type of collusion. For this reason, we shall only consider the averaging type collusion, where $\lambda_j = 1/K$ for all j in the remainder of this paper. Further, additional distortion noise $\mathbf{z}$ is added following averaging. We illustrate the model for this type of collusion in Figure 1. A similar model for collusion was used in [14,16]. We note, however, that there may exist cases in which the underlying fingerprints will not necessarily have the same energy, or be independent of each other, and that other choices for λ_j might be more appropriate. These cases are beyond the scope of the current paper.

3 Orthogonal Modulation and Anti-Collusion

In this section we will focus on the methods of orthogonal modulation [27] for embedding unique fingerprints in multiple copies of images. In orthogonal modulation, there are v orthogonal signals $\mathbf{s}_j$ that are used to convey $B = \log_2 v$ bits by inserting one of the v signals into the host signal. These B bits can be used to identify the n users by identifying a B-bit ID sequence with each user, and therefore we have $n = v$. The detector determines the B information bits by performing the correlation of the test signal with each of the

v signals, and decides the signal that has the largest correlation above a minimum threshold. Typically, v correlations are used to determine the embedded signal, and the computational complexity associated with performing v correlations is considered one of the drawbacks of orthogonal modulation. In Section 3.1, we present an improved detection strategy that cuts the computational complexity from $\mathcal{O}(v)$ to $\mathcal{O}(\log v)$.

An additional drawback for using orthogonal modulation in data embedding is the large number of orthogonal signals needed to convey B bits. In many situations, it might not be possible to find 2^B orthogonal signals in the content. In audio applications, it might be desirable to periodically repeat a watermark embedding in the content in order to fingerprint clips from the audio. In this case the number of orthogonal basis signals available is limited by the sample rate. For example, if we repeat a watermark every second in audio with a 44.1kHz sample rate, then we can allow for at most $44,100$ users to purchase the content if orthogonal modulation is used in fingerprinting. Although other media, such as images and video, might have more points per embedding period, many of these degrees of freedom will be lost since embedding should only take place in perceptually significant components [5]. In particular, some content, such as smoothly textured images and binary images, are known to have a significantly lower embedding rate than what is suggested by the amount of points in the image. Further, the necessary bookkeeping and storage of the $v = 2^B$ basis vectors, or a set of keys for generating them, is another drawback of orthogonal modulation. In Section 4, we build watermarks using code modulation that are able to accommodate more users than orthogonal modulation for the same amount of orthogonal vectors.

We can study the effect of collusion on orthogonal modulation by calculating the distance between the constellation points and averages of the constellation points. Additionally, since the goal of collusion is to create a new content whose watermarks have been sufficiently attenuated that they are undetectable, we would like to calculate the distance between the averages of the constellation points and the origin. In an additive white Gaussian noise model, the Euclidean distance between the constellation points as well as the distance between the constellation points and the origin are directly related to the probability of detection through the argument of a Q function [27]. Smaller distances lead to higher probability of detection error.

Suppose each watermark is embedded using $\mathcal{E}$ energy. If we average K watermarks, then the distance from the colluded mark to any of the watermarks used in forming it is $\sqrt{\mathcal{E}(K-1)/K}$. The distance from the colluded mark to any of the other watermarks not used in the collusion is $\sqrt{\mathcal{E}(K+1)/K}$. Further, the distance of the colluded mark from the origin is $\sqrt{\mathcal{E}/K}$. Thus, as K increases, the identifying watermarks in the colluded mark will become harder to detect.

3.1 Efficient Detection Strategy for Orthogonally Modulated Fingerprints

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 [5, 28]. In this section we present an algorithm that dramatically reduces the computation needed to detect which watermarks are present in a host signal.

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 $SUM(A) = \sum_{j\in J}\mathbf{w}_j$. We start by considering the case of detecting 1 watermark. Let us denote by $S = \{\mathbf{w}_, \cdots, \mathbf{w}_v\}$ 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 $SUM(S_0)$ then the correlation will satisfy

$$\langle \mathbf{y}, \sum_{\mathbf{w}_j\in S_0}\mathbf{w}_j\rangle = \sum_{j\in J}\langle \mathbf{y}, \mathbf{w}_j\rangle, \tag{9}$$

where $\langle \mathbf{y}, \mathbf{w}\rangle$ denotes a correlation statistic, such as is described in (4). If the one watermark we desire to detect belongs to the set S_0 then $\langle \mathbf{y}, SUM(S_0)\rangle$ 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 $\langle \mathbf{y}, SUM(S_0)\rangle$ 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 1, 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 1 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 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 1 to identify K signals from a set S of n orthogonal signals. Lemma 1 provides a bound for $C(n, K)$ in the ideal case where each correlation is truthful.

Algorithm: $EffDet(\mathbf{y}, S)$
Divide S into two sets S_0 and S_1, where $|S_0| = 2^{\lceil \log_2 |S| \rceil - 1}$, and $S_1 = S \backslash S_0$;
Calculate $\mathbf{e}_0 = SUM(S_0)$ and $\mathbf{e}_1 = SUM(S_1)$;
Calculate $\rho_0 = \langle \mathbf{y}, \mathbf{e}_0 \rangle$ and $\rho_1 = \langle \mathbf{y}, \mathbf{e}_1 \rangle$;
$\tau_0 = DetermineThreshold(|S_0|)$;
$\tau_1 = DetermineThreshold(|S_1|)$;
if $\rho_0 > \tau_0$ **then**
    if $|S_0| = 1$ **then**
        output S_0 ;
    else
        $EffDet(\mathbf{y}, S_0)$;
    end
end
if $\rho_1 > \tau_1$ **then**
    if $|S_1| = 1$ **then**
        output S_1 ;
    else
        $EffDet(\mathbf{y}, S_1)$;
    end
end
return ;

Algorithm 1: Efficient detection algorithm, $EffDet(\mathbf{y}, S)$

Lemma 1. *The number of correlations $C(n, K)$ needed in Algorithm 1 satisfies*

$$C(n, K) \leq 2\left(-1 + K\left(\log_2(2^{\lceil \log_2 n \rceil}/K) + 1\right)\right). \tag{10}$$

This lemma can be shown using standard techniques for tree-based algorithms [29–32]. In particular, the result of this lemma gives us 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 (9).

We now provide analysis for this phenomenon when there is only one colluder, i.e. $y(k) = s_1(k) + d(k)$. 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

$$\begin{cases} H_0 : \mathbf{y} = \mathbf{d} \\ H_1 : \mathbf{y} = \mathbf{d} + \mathbf{s}_1 \end{cases} \tag{11}$$

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

$$\langle \mathbf{y}, \mathbf{s} \rangle = \sum_{k=1}^{N} y(k)s(k). \tag{12}$$

Under hypothesis H_1, the calculation for ρ_0 is

$$\rho_0 = \langle \mathbf{s}_1 + \mathbf{d}, \mathbf{s}_1 + \mathbf{s}_2 + \cdots + \mathbf{s}_{n/2} \rangle = \|\mathbf{s}_1\|^2 + \sum_{j=1}^{n/2} \langle \mathbf{d}, \mathbf{s}_j \rangle. \tag{13}$$

Under hypothesis H_0, the calculation for ρ_0 is

$$\rho_0 = \langle \mathbf{s}_1 + \mathbf{z}, \mathbf{s}_1 + \mathbf{s}_2 + \cdots + \mathbf{s}_{n/2} \rangle = \sum_{j=1}^{n/2} \langle \mathbf{z}, \mathbf{s}_j \rangle. \tag{14}$$

Then $E(\rho_0; H_0) = 0$, $E(\rho_0; H_1) = \mathcal{E}$, and $Var(\rho_0; H_0) = Var(\rho_0; H_1) = (n/2)\sigma_d^2 \mathcal{E}$. 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

$$P_D = Pr(\rho_0 > \tau; H_1) = Q\left(\frac{\tau - \mathcal{E}}{\sqrt{\sigma_d^2 n\mathcal{E}/2}}\right). \tag{15}$$

The probability of false alarm is

$$P_{FA} = Pr(\rho_0 > \tau; H_0) = Q\left(\frac{\tau}{\sqrt{\sigma_d^2 n\mathcal{E}/2}}\right). \tag{16}$$

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 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, i.e. from (15) or (16). 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.

We now provide a bound for the expected amount of correlations, $E[C(n,1)]$, needed to identify a single colluder when $n = 2^r$. We denote the probability of false alarm for a node b by $P_{FA,b}$, where b corresponds to the binary representation describing the path from the root to that node.

Lemma 2. *When $n = 2^r$ and there is only one colluder, which we label as user 1, the expected number of correlations $E[C(n,1)]$ needed in Algorithm 1 satisfies*

$$E[C(n,1)] \le 2 + 2(r-1)P_D + 2\sum_{k=1}^{r-1} P_{FA,b_k}(2^{r-k} - 1), \tag{17}$$

where b_k is the binary string consisting of $k-1$ zeros followed by a single 1.

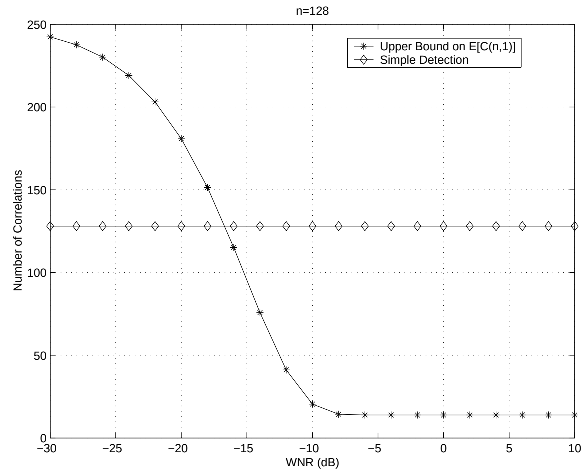
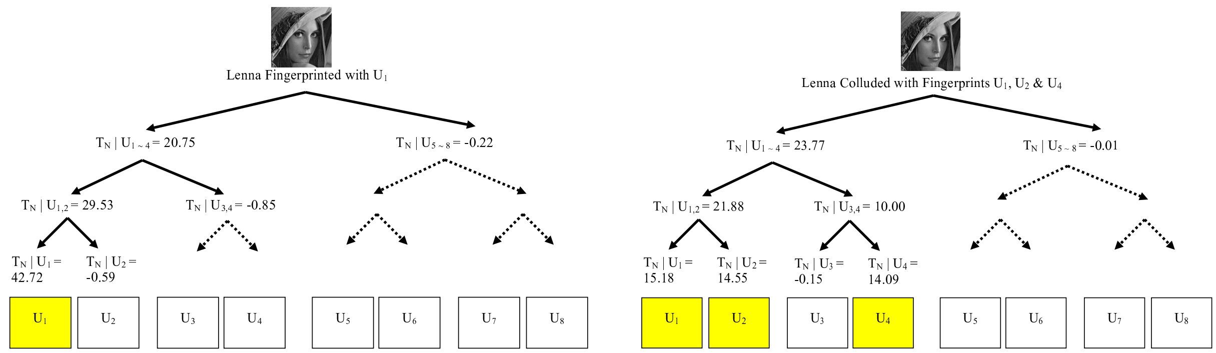


Figure 2: The bound for the expected amount of correlations needed when there is one colluder, $n = 128$ users, and $P_D = 0.99$ for each level. As a baseline, we plot the bound for $E[C(128, 1)]$ against the amount n, which is the amount of computations needed in performing simple detection.

The proof of Lemma 2 is provided in Appendix 1. The bound depends on the choice of P_D and the P_{FA,b_k} values. In Figure 2, 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 amount of correlations needed in Algorithm 1 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_{FA,b_k} small at low WNR. Thus, at low WNR the efficient detection scheme may not be advantageous over a simple detection scheme. However, at higher WNR, which corresponds to non-blind detection scenarios, the separation between the detection hypotheses increases, and it does become possible to make P_{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.

3.2 Experiments on Efficient Detection of Orthogonal Fingerprints

We desired to study the performance of the efficient detection algorithm, and the effect that collusion had on the detection statistics. In our experiments, we used an additive spread spectrum watermarking scheme similar to that in [6], 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 (6). The 512×512 Lenna was used as the host image for fingerprinting,



(a) One Colluder (b) Three Colluders

Figure 3: 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 "$T_N|U_?$" denotes the detection statistics from correlating the test image with the sum of the fingerprints $U_?$.

and the fingerprinted images had no visible artifacts with an average PSNR of 41.2dB. Figure 3 illustrates the process of identifying colluders out of 8 users using the efficient 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 psedudo-random number generator and used as an approximate orthogonal basis in orthogonal modulation.

Figure 3(a) shows the process of detecting colluders from an image with user 1's fingerprint embedded. The notation "$T_N|U_?$" 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 3(b) 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 3(a) 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 3.1. Since the detection statistics we use have their variance normalized to 1, the noisy contributions lower the values of detection statistics. We also observe in Figure 3(b) a decrease in the detection statistics in images colluded by more users.

In addition, we conducted a non-blind detection test with one colluder amongst $n = 128$ users on the Lenna image. Our test confirmed the findings of Figure 2. Only 14 correlations were needed, which is a significant reduction over the 128 correlations needed in a simple detection approach.

4 Code Modulation Embedding and Anti-Collusion Codes

In the previous section, we mentioned that a drawback of the usage of orthogonal signaling is the large amount of basis vectors needed to convey user information. In this section we will present another form of modulation, known as code modulation, that may be used to convey more fingerprint code bits for a given amount of basis vectors than orthogonal modulation. Therefore, we are able to accommodate more users than orthogonal modulation with the same amount of orthogonal signals. We will use this modulation technique, in conjunction with appropriately designed codewords, known as anti-collusion 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

$$\mathbf{w}_j = \sum_{i=1}^{v} b_{ij}\mathbf{u}_i, \tag{18}$$

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 [27]. 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 $\mathbf{u}_j$. In the following section, we shall design a code matrix $\mathbf{C}$ whose elements are either 0 or 1. By applying a suitable mapping that depends on whether the OOK or antipodal form of code modulation is used, the code matrix $\mathbf{C}$ is used to derive the matrix $\mathbf{B}$ that is used in forming the watermark 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, if antipodal code modulation is used with $\sqrt{\mathcal{E}/v}$ for each component, then 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).

4.1 Anti-Collusion Codes

In this section we design a family of codevectors $\{\mathbf{c}_j\}$ whose overlap with each other can identify groups of colluding users. A similar idea was proposed in [33], where projective geometry was used to construct such code sequences. As we will explain in this section, our proposed code construction makes more efficient usage of the basis vectors than the codes described in [33].

For this section, we describe codes using the binary symbols $\{0, 1\}$. These codevectors are mapped to *derived* codevectors by a suitable mapping depending on whether the OOK or antipodal form of binary code modulation is used for watermarking. For example, when used in the antipodal form, the binary symbols $\{0, 1\}$ are mapped to $\{-1, 1\}$ via $f(x) = 2x - 1$.

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 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 4.2, and detail approaches to improve the validity of the AND assumption.

We want to design codes such that when K or fewer users collude, we can identify the colluders. To accommodate a given number of users and trace a given number of colluders, we prefer shorter codes since, for embedded fingerprints, longer codes would require constructing and bookkeeping more basis vectors, and would have to distribute the fingerprint energy over more basis vectors. 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 anti-collusion codes. In the definition that follows, we provide a generic definition in terms of semigroups [34], and then specify the relevant case that we use in this paper.

Definition 1. *Let G be a semigroup with a binary operation $\star$. A code $\mathcal{C} = \{\mathbf{c}_1, \cdots, \mathbf{c}_n\}$ of vectors belonging to G^v is called a K-resilient $(G, \star)$ anti-collusion code, or a $(G, \star)$ ACC when any subset of K or fewer codevectors combined element-wise under $\star$ is distinct from the element-wise $\star$ of any other subset of K or fewer codevectors. When $G = \{0, 1\}$ and $\star$ is the logical AND, a K-resilient $(G, \star)$ ACC is simply called a K-resilient AND-ACC.*

We first present a 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 when $K \leq n$ of these vectors are combined under AND, that this combination 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.

It is desirable to shorten the codelength 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 now present a construction of a K-resilient AND-ACC that requires $\mathcal{O}(K\sqrt{n})$ basis vectors for n users. This construction uses balanced incomplete block designs [35]:

Definition 2. *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 $\mathcal{X}$ occur together in exactly λ blocks.*

The theory of block designs is a field of mathematics that has found application in the construction of error correcting codes and the design of statistical experiments. A (v, k, λ)-BIBD has a total of $n = \lambda(v^2 - v)/(k^2 - k)$ blocks. Corresponding to a block design is the $v \times n$ incidence matrix $\mathbf{M} = (m_{ij})$ defined by

$$m_{ij} = \begin{cases} 1 & \text{if the } i\text{th element belongs to the } j\text{th block,} \\ 0 & \text{otherwise.} \end{cases}$$

If we define the codematrix $\mathbf{C}$ as the bit-complement of $\mathbf{M}$, and assign the codevectors $\mathbf{c}_j$ as the columns of $\mathbf{C}$, then we have 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 we therefore need $v = \mathcal{O}(\sqrt{n})$ basis vectors.

Theorem 1. *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.*

The proof is provided in Appendix 2. We now present an example. The following is the bit-complement of the incidence matrix for a $(7, 3, 1)$-BIBD:

$$\mathbf{C} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{19}$$

This code requires 7 bits for 7 users and provides 2-resiliency since any two column vectors share a unique pair of 1 bits. Each column vector $\mathbf{c}$ of $\mathbf{C}$ is mapped to $\{\pm 1\}$ by $f(x) = 2x - 1$. The code modulated watermark is then $\mathbf{w} = \sum_{i=1}^{v} f(c(i))\mathbf{u}_i$. When two watermarks are averaged, the locations where the corresponding

AND-ACC agree and have a value of 1 identify the colluding users. For example, let

$$\mathbf{w}_1 = -\mathbf{u}_1 - \mathbf{u}_2 + \mathbf{u}_3 - \mathbf{u}_4 + \mathbf{u}_5 + \mathbf{u}_6 + \mathbf{u}_7 \quad (20)$$

$$\mathbf{w}_2 = -\mathbf{u}_1 + \mathbf{u}_2 - \mathbf{u}_3 + \mathbf{u}_4 + \mathbf{u}_5 - \mathbf{u}_6 + \mathbf{u}_7 \quad (21)$$

be the watermarks for the first two columns of the above $(7, 3, 1)$ code, then $(\mathbf{w}_1 + \mathbf{w}_2)/2$ has coefficient vector $(-1, 0, 0, 0, 1, 0, 1)$. The fact that a 1 occurs in the 5th and 7th location uniquely identifies user 1 and user 2 as the colluders.

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 average amount of users that can be accommodated per basis vector. AND-ACCs with a higher β are better. For codes (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 codelength v becomes larger. By Fisher's Inequality [35], 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 [33] has efficiency much less than 1, and thus requires more spreading sequences (or marking locations) to accommodate the same amount of users as our scheme. It is possible to use the collusion-secure code constructions of [4] in conjunction with code modulation for embedding. However, the construction described in [4] is limited to collusion resistance of $K \leq \log n$, and is designed to trace one colluder among K colluders. Their construction has codelength $\mathcal{O}(\log^4 n \log^2(1/\epsilon))$, where $\epsilon < 1/n$ is the decision error probability. This codelength is considerably large for small error probabilities and practical n values. For example, when $n = 2^{10}$, the codelength of [4] is on the order of 10^6, while the codelength 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.

There are systematic methods for constructing infinite families of BIBDs. For example, $(v, 3, 1)$ systems (also known as Steiner triple systems) are known to exist if and only if $v \equiv 1 \text{ or } 3 \pmod 6$; the Bose construction builds Steiner triple systems when $v \equiv 3 \pmod 6$, and the Skolem construction builds Steiner triple systems when $v \equiv 1 \pmod 6$ [36]. 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 [35,37]. Techniques for constructing these and other BIBDs can be found in [38]. 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 beyond the scope of this paper.

4.2 Detection Strategies

In this section, we discuss the problem of detecting the colluders when our AND-ACCs are used with code modulation. We present several detection algorithms that can be used to identify possible colluders. This section serves as a basis for demonstrating the performance of our ACC. Our goal here is to find efficient detection structures by taking advantages of the special characteristics of our ACC.

We assume that the total distortion $\mathbf{d}$ is an N-dimensional vector following an i.i.d. Gaussian distribution with zero-mean and variance σ_d^2. Under the colluder-absent hypothesis H_0, the observed content $\mathbf{y}$ is the distortion signal $\mathbf{d} = \mathbf{x} + \mathbf{z}$. Under the colluder-present hypothesis H_1, K colluders come together and perform an averaging attack that produces a colluded version of the content $\mathbf{y}$. Presented in a hypotheses-testing framework, we have

$$\begin{aligned} H_0: \ \mathbf{y} &= \mathbf{d} \\ H_1: \ \mathbf{y} &= \frac{1}{K}\sum_{j \in S_c} \mathbf{y}_j + \mathbf{z} = \frac{1}{K}\sum_{j \in S_c} \mathbf{s}_j + \mathbf{d} \end{aligned} \tag{22}$$

where K is the number of colluders, and S_c indicates a subset with size K. The marked content $\mathbf{y}_j$ for each user j is given as

$$\mathbf{y}_j = \mathbf{x} + \mathbf{s}_j = \mathbf{x} + \alpha \sum_{i=1}^{v} b_{ij}\mathbf{u}_i \tag{23}$$

where α is used to control the strength of the fingerprint. Clearly the precise probability law under H_1 depends on 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. Due to the discrete nature of our 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 i^{th} component expressed by

$$T_N(i) = \mathbf{y}^T \mathbf{u}_i / \sqrt{\sigma_d^2 \cdot \parallel \mathbf{u}_i \parallel^2} \tag{24}$$

for $i = 1, ..., v$. It is straightforward to show that

$$\mathbf{T}_N = \frac{\alpha_1}{K}\mathbf{B}\mathbf{\Phi} + \mathbf{n} \tag{25}$$

where the column vector $\mathbf{\Phi} \in \{0,1\}^n$, indicates colluders via the location of components whose value are 1; $\alpha_1 = \alpha\sqrt{\parallel \mathbf{u} \parallel^2 / \sigma_d^2}$ is assumed known, with $\|\mathbf{u}_i\| = \|\mathbf{u}\|$ for all i; and $\mathbf{n} = [\mathbf{u}_1, ..., \mathbf{u}_v]^T \mathbf{d} / \sqrt{\sigma_d^2 \cdot \parallel \mathbf{u} \parallel^2}$ follows

Algorithm: $\mathit{SuspectAlg}(\mathbf{\Gamma})$

$\mathbf{\Phi} = \mathbf{1}$;

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 jth row of $\mathbf{C}$;

  $\mathbf{\Phi} = \mathbf{\Phi} \cdot \mathbf{e}_j$;

end

return $\mathbf{\Phi}$;

Algorithm 2: Algorithm $\mathit{SuspectAlg}(\mathbf{\Gamma})$, which determines the vector $\mathbf{\Phi}$ that describes the suspect set.

a $N(\mathbf{0}, \mathbf{I}_v)$ distribution. Here $\mathbf{B}$ is the derived code matrix and K the number of 1's in $\mathbf{\Phi}$. Thus, the model (22) can be equivalently presented as

$$\begin{aligned} H_0 &: f(\mathbf{T}_N) = N(\mathbf{0}, \mathbf{I}_v) \\ H_1 &: f(\mathbf{T}_N|\mathbf{\Phi}) = N(\frac{\alpha_1}{K}\mathbf{B}\mathbf{\Phi}, \mathbf{I}_v) \end{aligned} \quad (26)$$

with reference to (22) and (25).

Our goal in this section is to efficiently estimate $\mathbf{\Phi}$. 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 4.1, 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 [38] and therefore the antipodal form of code modulation is preferred.

4.2.1 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 $\mathbf{\Gamma} = (\Gamma_1, \Gamma_2, \cdots, \Gamma_v)$, 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 2 starts with the entire group as the suspicious set, and uses the components of Γ that are equal to 1 to further narrow the suspicious set. We determine a vector $\mathbf{\Phi} = (\Phi_1, \Phi_2, \cdots, \Phi_n)^T \in \{0, 1\}^n$ that

```
Algorithm: AdSortAlg(T_N)
Sort elements of T_N in descending order and record corresponding index vector as J ;
Set Φ = 1; Set i = 0 and calculate the likelihood LL(i) = f(T_N|Φ) according to (26) ;
Flag = True ;
while Flag & i < v do
    Set i = i + 1 ;
    j = J(i) and Γ(j) = 1 ;
    Define e_j to be the j-th row of C ;
    Φ_up = Φ · e_j ;
    LL(i) = f(T_N|Φ_up) ;
    if LL(i) > LL(i − 1) then
        Φ = Φ_up ;
    else
        Flag = False ;
    end
end
return Φ ;
```

Algorithm 3: Algorithm $AdSortAlg(\mathbf{\Gamma})$, which uses an adaptive sorting approach to determine the vector $\mathbf{\Phi}$ that describes the suspect set.

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 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.

4.2.2 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 3, 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.

4.2.3 Sequential Algorithm

The approaches in both Section 4.2.1 and Section 4.2.2 share the same idea that the colluders can be uniquely identified by utilizing the locations of 1s in $\mathbf{\Gamma}$ due to the structural features of our AND-ACC. These two

Algorithm: $SeqAlg(\mathbf{T_N})$

Set $K = 0$;

Calculate the likelihood $LL(0) = f(\mathbf{T}_N|\mathbf{\Phi} = \mathbf{0})$;

Set $J = \emptyset$ 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 $j = 1, ..., (K-1)$ via

$$i_K = \arg\max_{i_K}\{f(\mathbf{T}_N|\mathbf{J} = \{i_1, \cdots, i_K\})\};$$

$J = \{i_1, ..., i_K\}$ and $\mathbf{\Phi}_{up}(J) = 1$;

Calculate $LL(K) = f(\mathbf{T}_N|\mathbf{\Phi}_{up})$;

if $LL(K) > LL(K-1)$ **then**

$\mathbf{\Phi} = \mathbf{\Phi}_{up}$;

else

Flag = False ;

end

end

return $\mathbf{\Phi}$;

Algorithm 4: Algorithm $SeqAlg(\mathbf{T_N})$, which uses the sequential detection approach to determine the vector $\mathbf{\Phi}$ that describes the suspect set.

procedures are simple, and easy to implement. 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 (26).

Thus, we introduce Algorithm 4, 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 4 directly estimates the colluder set, while the adaptive sorting algorithm first estimates the code bits before deciding the colluder set.

4.2.4 Alternate Maximization (AM) Approach

We observe that, since a binary variable is assigned to each user that indicates his/her presence or absence in the coalition, the collusion problem (25) is related to the estimation of superimposed signals [39]. Different algorithms have been proposed for the estimation of superimposed signals: exhaustive-search ML methods have high accuracy but are computationally expensive, while a number of other approaches have been proposed to reduce the load. Particularly, the Alternating Maximization (AM) method [40,41] gives a good tradeoff between accuracy and computational complexity. The AM algorithm approaches an M-dimensional estimation problem by solving M one-dimensional problems iteratively. We apply the AM idea in our situation in Algorithm 5. In this algorithm, we start with an initial estimate of $K = 1$ for the number of

Algorithm: $AMAlg(\mathbf{T_N})$

Set $K = 0$;

Calculate the likelihood $LL(0) = f(\mathbf{T}_N | \mathbf{\Phi} = \mathbf{0})$;

Set $J = \emptyset$ and Flag1 = True ;

while *Flag1* **do**

  Let $K = K + 1$;

  Set $t = 0$, estimate $i_K^{(0)}$, the index of the K-th user, via the ML criteria, assuming that $(K-1)$ users have indices $i_j^{(0)} = J(j)$, for $j = 1, ..., (K-1)$;

  Flag2 = True ;

  while *Flag2* **do**

    $t = t + 1$;

    for $j = 1, ..., K$ **do**

      $i_j^{(t)} = \arg\max_{i_j} \{ f(\mathbf{T}_N | J = \{ i_1^{(t)}, \cdots, i_{j-1}^{(t)}, i_j, i_{j+1}^{(t-1)}, \cdots, i_K^{(t-1)} \}) \}$;

    end

    Sort $i_j^{(t)}$;

    if $\forall j \quad i_j^{(t)} = i_j^{(t-1)}$ **then**

      Flag2= False ;

    end

    $J = \{ i_1^{(t)}, ..., i_K^{(t)} \}$ and $\mathbf{\Phi}_{up}(J) = 1$;

    Calculate $LL(K) = f(\mathbf{T}_N | \mathbf{\Phi}_{up})$;

  end

  if $LL(K) > LL(K-1)$ **then**

    $\mathbf{\Phi} = \mathbf{\Phi}_{up}$;

  else

    Flag1 = False ;

  end

end

return $\mathbf{\Phi}$;

Algorithm 5: Algorithm $AMAlg(\mathbf{T_N})$, which uses the alternate maximization approach to determine the vector $\mathbf{\Phi}$ that describes the suspect set.

colluders. For each K we apply the AM idea to estimate the index of K colluders. We increase K by one and continue this process until the likelihood decreases.

The AM algorithm converges to one of many local maxima, which, depending on the initial condition, may or may not be the global one. Therefore the initialization is a critical element in this algorithm. Based on our simulations, the initialization used in this Algorithm 5 leads to good performance and fast convergence.

We evaluate the performance of the detection schemes in Section 4.3.

4.3 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 a $N = 10000$ point vector of i.i.d. Gaussian samples. The factor α is applied equally to all components and is used to control the WNR, where $\mathrm{WNR} = 10\log_{10}\|\mathbf{s}\|^2/\|\mathbf{d}\|^2\mathrm{dB}$. We use these simulations to verify some basic issues associated with collusion and code modulation.

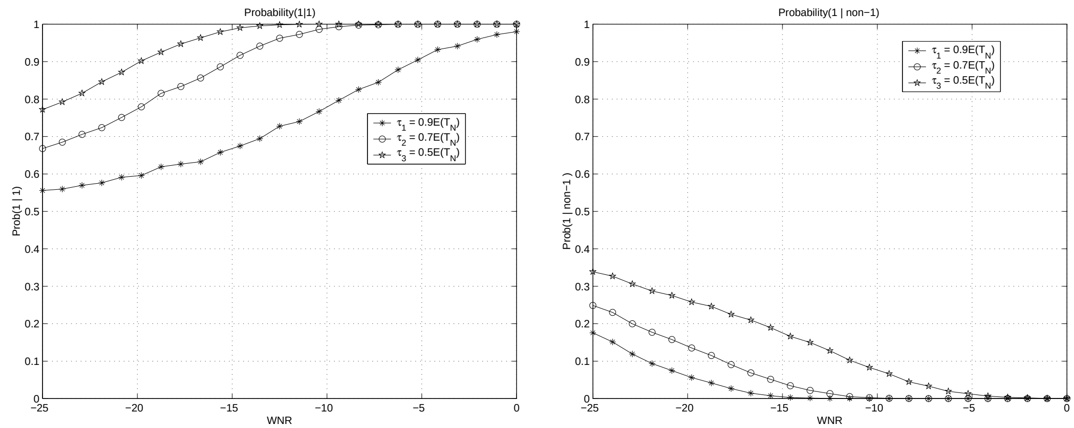
In the simulations that follow, we used a $(16, 4, 1)$-BIBD to construct our AND-ACC code. The $(v, 4, 1)$ codes exist if and only if $v \equiv 1 \text{ or } 4 \pmod{12}$. By complementing the incidence matrix, we get the following code matrix

$$\mathbf{C} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1
\end{pmatrix}. \tag{27}$$

With this code, we use 16 orthogonal basis vectors to handle 20 users, and can uniquely identify up to $K = 3$ colluders. The fingerprints for each user were assigned according to the antipodal form of code modulation, using 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 (24).

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. We examined the tradeoff between the probability $p(1|1)$ of correctly detecting a 1 when a 1 was expected from the AND logic, and the probability of $p(1|\text{non-1})$, where the detector decides a 1 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 (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 4(a), and the plot of $p(1|\text{non-1})$ is presented in Figure 4(b). We observe that for the smaller threshold of $0.5E(T_N)$ the probability $p(1|1)$ is higher, but at an 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 1, but at an expense of also decreasing the probability of correctly classifying a bit as a 1.



(a) (b)

Figure 4: (a) The probability of detection $p(1|1)$ and (b) the probability of false alarm $p(1|\text{non-1})$ for different WNR and different thresholds using hard detection.

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 4.2 and present the results in Figure 5. For each WNR we averaged over 2000 experiments. Since the AM algorithm had almost identical performance to the sequential algorithm, we only report the results of the sequential algorithm.

We observe in Figure 5 (a) and Figure 5 (b) 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

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 thresholding:	0	0	0	0	1	1	0	1	0	1	0	0	0	1	0	1

Table 1: The derived codevectors from a $(16, 4, 1)$ AND-ACC for user 1, user 4, and user 8. Also presented are the vectors from a two colluder scenario, and a three colluder scenario. The bottom row corresponds to the desired output of the detector using the AND logic for the three colluder case.

note, however, that in Figure 5(b), at low WNR between -25dB and -15dB, 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 Figure 4(a) and Figure 4(b). 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 -25dB and -15dB, the additional 1's from false detections at lower WNR eliminates 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 -20dB and -3dB, 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.

Consistent behavior is observed for the different detection schemes under the other performance measures, as depicted in Figures 5 (c)-(f). Overall, the sequential detection scheme provides the most promising balance between capturing colluders and placing innocents under suspicion.

4.4 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 [6], where the perceptually weighted watermark was added to 8×8 block DCT coefficients. The detection of the watermark is a blind detection scenario performed without the knowledge of the host image via the detection statistics as shown in (6). We used the same code matrix, detailed in

(27), for the AND-ACC as in the simulations for Gaussian signals. The 512×512 Lenna and Baboon images were used as the host signals for the fingerprints. The fingerprinted images have no visible artifacts with an average PSNR of 41.2dB for Lenna, and 33.2dB for Baboon. Figure 6 shows the original images, the fingerprinted images, and the difference with respect to the originals.

The three derived code vectors that were assigned to user 1, 4, and 8 via antipodal mapping as well as the colluded versions are presented in Table 1. Two collusion examples are illustrated in Figure 7 and the detection statistics of the two examples are shown in Figure 8. In one example we averaged the Lenna images fingerprinted with user 1 and 4's codes, and the other is for averaging user 1, 4, and 8's. The colluded images are further compressed using JPEG with quality factor (QF) 50%. Also shown in Figure 8 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 Table 1. We can see in Figure 8 and Figure 9 that non-blind 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 statistics from several collusion cases with different distortions applied to the colluded Lenna images in Figure 9. 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 9 shows the histograms of the blind and non-blind 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. 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 8. Similarly, we can use T_N with other detectors, whose performance was presented in Section 4.3.

We studied the effect of averaging collusion in the presence of no distortion, JPEG compression, and low pass filtering. We present an aggregated histogram in Figure 10 to show the separation of the one and non-one decision regions. The histogram depicts the blind detection statistics from a total of 8 collusion scenarios ranging from 1 to 3 colluders, and 4 distortion settings including no distortion, JPEG compression with quality factor of 50% and 90%, and low pass filtering.

5 Conclusion

In this paper, we investigated the problem of fingerprinting multimedia content that can resist collusion attacks and trace colluders. We studied linear collusion attacks for additive embedding of fingerprints.

We first studied the effect of collusion upon orthogonal embedding. 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 efficiently identify K colluders.

A further drawback of orthogonal modulation for embedding is that it requires as many orthogonal signals as users. 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 anti-collusion codes (ACC) that are used in conjunction with modulation to fingerprint multimedia sources. Our anti-collusion 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 four 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 four detection strategies examined. We also evaluated our fingerprints on real images, and observed that in both the blind and non-blind detection cases, the values of the detection statistics were well-separated. This behavior allows the detector to accurately determine the colluder set by estimating a fingerprint codevector that corresponds to the colluder set.

Appendix 1

We now prove the bound presented in Lemma 2 of Section 3.1. This bound is for the expected number of correlations performed by Algorithm 1. Suppose, without loss of generality, that the single colluder we wish to detect is user 1. We may place the $n = 2^r$ users at the terminal nodes of an r-level binary tree, with user 1 on the left-most branch. This binary tree represents the underlying recursive nature of the algorithm. At the initial calling of Algorithm 1, two correlations are performed to calculate ρ_0 and ρ_1. We expand the left side of the tree when the statistic ρ_0 is greater than the threshold. This occurs with probability P_D, and incurs $2P_D$ correlations to the total. On the other hand, the right hand side of the tree will be expanded with probability $P_{FA,1}$, where $P_{FA,b}$ represents the probability of false alarm for a node and the binary representation describing the path from the root to this node is b. In the worst-case, the right hand side of the tree will require $2(2^{r-1} - 1)$ correlations. Therefore, the expected amount of computations needed by the right hand side of the tree is bounded above by $2P_{FA,1}(2^{r-1} - 1)$. We may continue to expand the left branch of the tree, at each level performing an additional $2P_D$ correlations due to the left child, while adding at most $2P_{FA,b_k}(2^{r-k} - 1)$ correlations for the right child at level k, where b_k is the binary string consisting of $k - 1$ zeros followed by a single 1 and $k = 1, 2, ..., r - 1$. Collecting terms, we get

$$
\begin{aligned}
E[C(n,1)] &\leq 2 + 2P_D + 2P_{FA,1}(2^{r-1} - 1) + 2P_D + 2P_{FA,01}(2^{r-2} - 1) + \cdots + 2P_D + 2P_{FA,00\cdots01} && (28) \\
&= 2 + 2(r-1)P_D + 2\sum_{k=1}^{r-1} P_{FA,b_k}(2^{r-k} - 1). && (29)
\end{aligned}
$$

$\square$

Appendix 2

We prove Theorem 1 presented in Section 4.1 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 $\cap_{j\in J} A_j^C$ 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 $\cup_{j\in J} A_j$ for all sets J with cardinality less than or equal to $k - 1$. Suppose we have a set of $k - 1$ blocks $A_1, A_2, \cdots, A_{k-1}$, we must show that there does not exist another set of blocks whose union produces the same set. There are two cases to consider. First, assume there is another set of blocks $\{A_i\}_{i\in I}$ with $\cup_{j\in J} A_j = \cup_{i\in I} A_i$ such that $I \cap J = \emptyset$ and $|I| \leq k - 1$. Suppose 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 have k elements. Thus, there does not exist another set of blocks $\{A_i\}_{i\in I}$ with $\cup_{j\in J} A_j = \cup_{i\in I} A_i$ and $I \cap J = \emptyset$. 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, $\cup_{j \in J} A_j$ is unique. $\square$

References

[1] N. Heintze, "Scalable document fingerprinting," in *1996 USENIX Workshop on Electronic Commerce*, November 1996.

[2] I. Cox, J. Bloom, and M. Miller, *Digital Watermarking: Principles & Practice*, Morgan Kaufmann, 2001.

[3] B. Chor, A. Fiat, M. Naor, and B. Pinkas, "Tracing traitors," *IEEE Tran. on Information Theory*, vol. 46, pp. 893–910, May 2000.

[4] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," *IEEE Tran. on Information Theory*, vol. 44, pp. 1897–1905, September 1998.

[5] I. Cox, J. Kilian, F. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Tran. on Image Proc.*, vol. 6(12), pp. 1673–1687, December 1997.

[6] C. Podilchuk and W. Zeng, "Image adaptive watermarking using visual models," *IEEE Journal on Selected Areas in Communications*, vol. 16(4), pp. 525–540, May 1998.

[7] R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp, "Perceptual watermarks for digital images and video," *Proceedings of the IEEE*, vol. 87, pp. 1108–1126, July 1999.

[8] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding - a survey," *Proceedings of the IEEE*, vol. 87, pp. 1062–1078, July 1999.

[9] M.D. Swanson, M. Kobayashi, and A. H. Tewfik, "Multimedia data-embedding and watermarking technologies," *Proceedings of the IEEE*, vol. 87, pp. 1064–1087, June 1998.

[10] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings of the IEEE*, vol. 87, pp. 1079–1107, July 1999.

[11] H. S. Stone, "Analysis of attacks on image watermarks with randomized coefficients," Tech. Rep. 96-045, NEC Research Institute, 1996.

[12] B. Chen and G.W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. on Info. Theory*, vol. 47, pp. 1423–1443, May 2001.

[13] P. Moulin and J.A. O'Sullivan:, "Information-theoretic analysis of information hiding," preprint, Sept. 1999, revised Dec. 2001. Available at http:// www.ifp.uiuc.edu/∼moulin/paper.html.

[14] F. Ergun, J. Kilian, and R. Kumar, "A note on the limits of collusion-resistant watermarks," in *Eurocrypt '99*, 1999, pp. 140–149.

[15] J. Kilian, T. Leighton, L. R. Matheson, T. G. Shamoon, R. E. Tarjan, and F. Zane, "Resistance of digital watermarks to collusive attacks," Tech. Rep. TR-585-98, Department of Computer Science, Princeton University, 1998.

[16] J.K. Su, J.J. Eggers, and B. Girod, "Capacity of digital watermarks subjected to an optimal collusion attack," in *European Signal Processing Conference (EUSIPCO 2000)*, 2000.

[17] W. Trappe, M. Wu, and K.J.R. Liu, "Collusion-resistant fingerprinting for multimedia," in *IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, 2002, pp. 3309–3312.

[18] H.-J. Guth and B. Pfitzmann, "Error- and collusion-secure fingerprinting for digital data," in *Proc. 3rd Intl. Workshop on Information Hiding*, 1999.

[19] D. Kirovski, H. Malvar, and Y. Yacobi, "A dual watermarking and fingerprinting system," Tech. Rep. MSR-TR-2001-57, Microsoft Research Laboratories, 2001.

[20] M. Wu and B. Liu, "Modulation and multiplexing techniques for multimedia data hiding," in *Proc. of SPIE ITcom'01, SPIE vol 4518*, Aug. 2001.

[21] M. Wu, *Multimedia Data Hiding*, Ph.D. thesis, Princeton University, 2001.

[22] H. V. Poor, *An Introduction to Signal Detection and Estimation*, Springer Verlag, 2nd edition, 1994.

[23] M. Wu, H. Yu, and A. Gelman, "Multi-level data hiding for digital image and video," in *Proceedings of SPIE, Photonics East Conference on Multimedia Systems and Applications*, 1999, vol. 3845.

[24] S.A. Kassam, *Signal Detection in Non-Gaussian Noise*, Springer Verlag, 1987.

[25] S. Voloshynovskiy, F. Deguillaume, S. Pereira, and T. Pun, "Optimal adaptive diversity watermarking with channel state estimation," *Proc. of SPIE Photonics West, Electronic Imaging 2001, Security and Watermarking of Multimedia Contents III*, Jan. 2001.

[26] J. Hernandez, M. Amado, and F. Perez-Gonzalez, "DCT-domain watermarking techniques for still images: Detector performance analysis and a new structure," *IEEE Trans. on Image Processing*, vol. 9(1), Jan. 2000, Special Issue on Image and Video Processing for Digital Libraries.

[27] J. G. Proakis, *Digital Communications*, McGraw-Hill, 4th edition, 2000.

[28] A. Herrigel, J. Oruanaidh, H. Petersen, S. Pereira, and T. Pun, "Secure copyright protection techniques for digital images," in *Second Information Hiding Workshop (IHW), Lecture Notes in Computer Science*, vol. 1525. Springer-Verlag, 1998.

[29] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, McGraw Hill, 1989.

[30] J. Aslam and A. Dhagat, "Searching in the presence of linearly bounded errors," in *Proceedings of the 23rd ACM Symposium on Theory of Computing*, May 1991, pp. 486–493.

[31] D. Z. Du, G. L. Xue, S. Z. Sun, and S. W. Cheng, "Modifications of competitive group testing," *SIAM Journal of Computing*, vol. 23, pp. 82–96, Feb. 1994.

[32] D. Z. Du and H. Park, "On competitive group testing," *SIAM Journal of Computing*, vol. 23, pp. 1019–1025, Oct. 1994.

[33] 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, pp. 456–467, 2000.

[34] T. W. Hungerford, *Algebra*, Springer-Verlag, 1974.

[35] J. H. Dinitz and D. R. Stinson, *Contemporary Design Theory: A Collection of Surveys*, John Wiley and Sons, 1992.

[36] C.C. Lindner and C.A. Rodger, *Design Theory*, CRC Press, 1997.

[37] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge Press, 1994.

[38] C. J. Colbourn and J. H. Dinitz, *The CRC Handbook of Combinatorial Designs*, CRC Press, 1996.

[39] S. Yau and Y. Bresler, "Maximum likelihood parameter estimation of superimposed signals by dynamic programming," *IEEE Trans. on Signal Processing*, vol. 41, pp. 804–820, Feb. 1993.

[40] I. Ziskind and M. Wax, "Maximum likelihood localization of multiple source by alternate projection," *IEEE Trans. ASSP*, vol. 36, pp. 1553–1560, Oct. 1988.

[41] R. Vaccaro T. Manickam and D. Tufts, "A least-squares algorithm for multipath time-delay estimation," *IEEE Trans. on Signal Processing*, vol. 42, pp. 3229–3233, Nov. 1994.

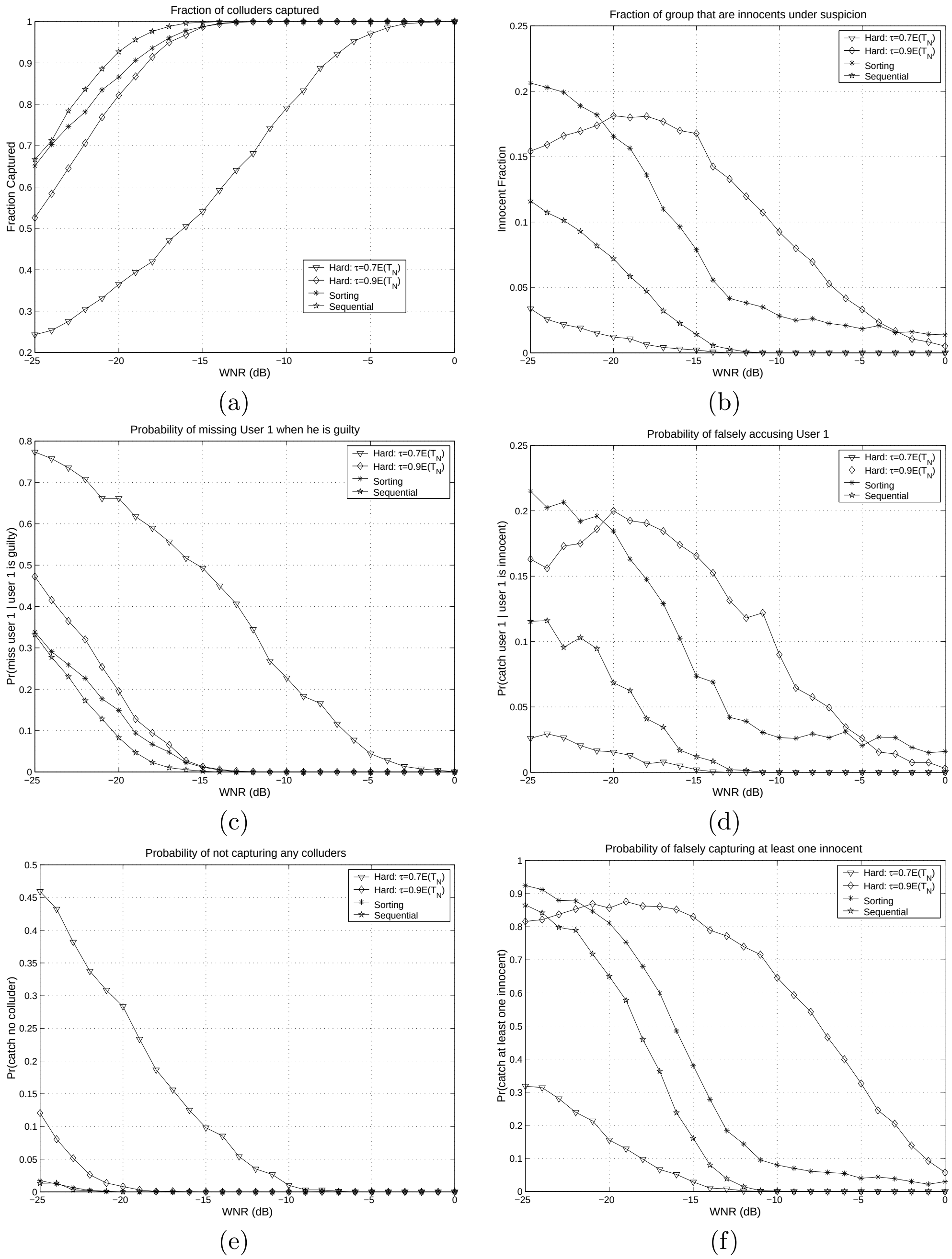


Figure 5: (a) The fraction of colluders that are successfully captured, or placed under suspicion, (b) the fraction of the total group that are innocent and falsely placed under suspicion for different WNR and different thresholds, (c) the probability of missing user 1 when he is guilty, (d) the probability 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.

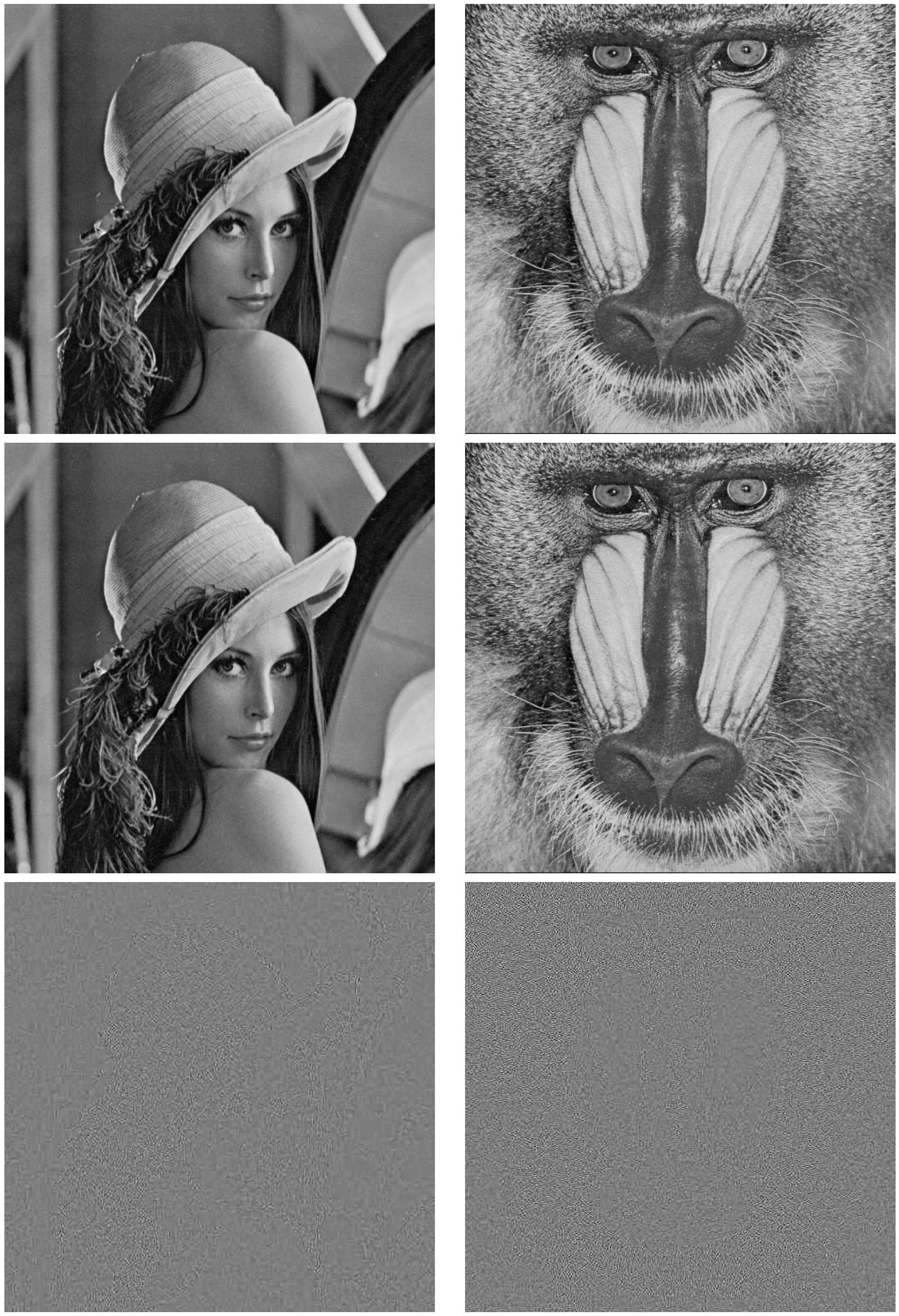


Figure 6: The original images (top), fingerprinted images (middle), and difference images (bottom) for Lenna and Baboon. In the difference images, gray color indicates zero difference between the original and the fingerprinted version, and brighter and darker indicates larger difference.

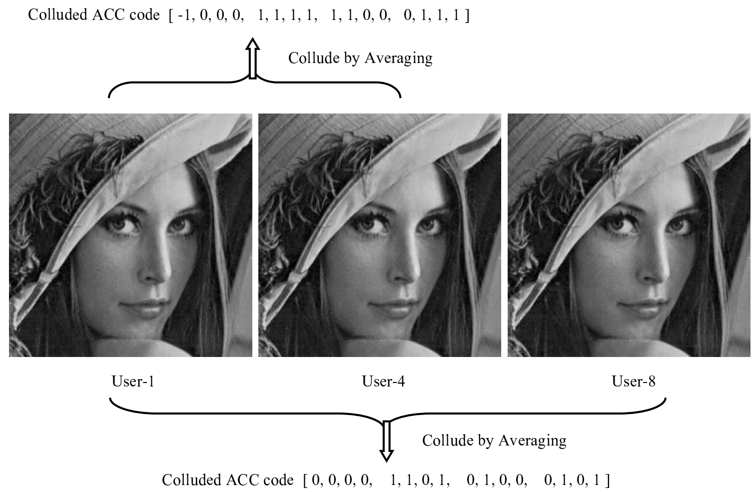


Figure 7: Illustration of collusion by averaging two and three images fingerprinted with ACC codes, respectively.

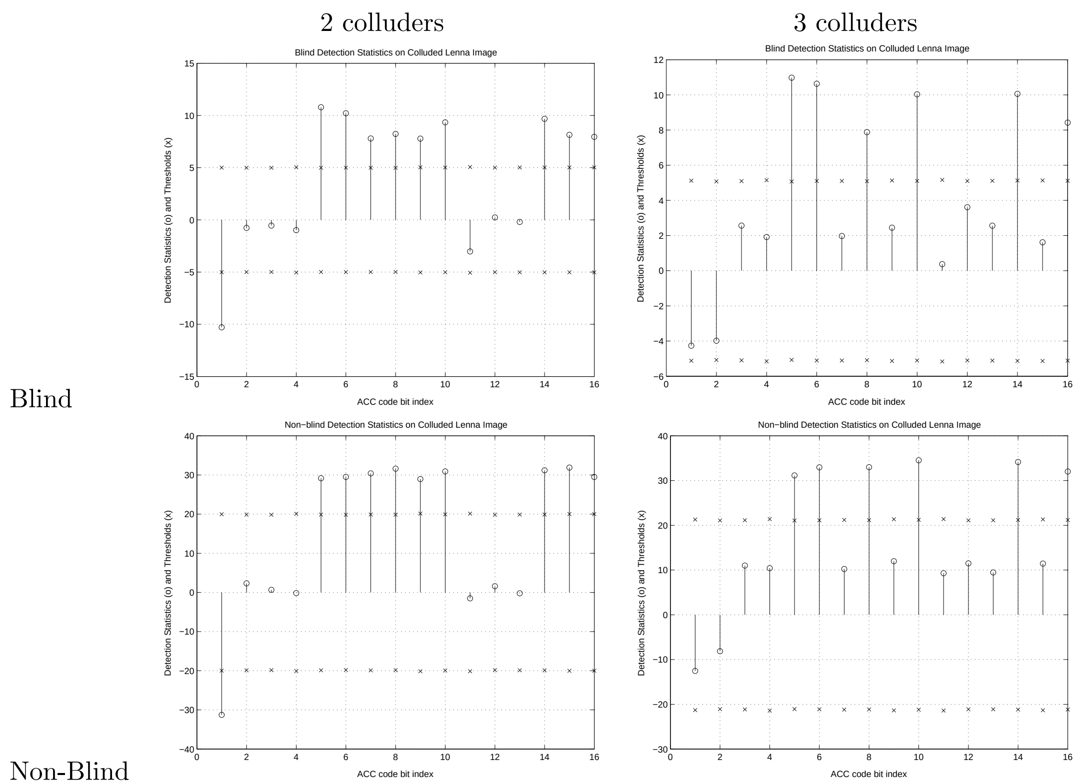


Figure 8: Example detection statistics values for 2 users' and 3 users' collusion with a $(16, 4, 1)$-BIBD AND-ACC fingerprint. (top) Blind detection scenario and (bottom) non-blind detection scenario. (left) User 1 and 4 perform averaging, resulting in the output of the detector as $(-1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1)$. (right) User 1, 4, and 8 perform averaging, resulting in the output of the detector as $(0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1)$.

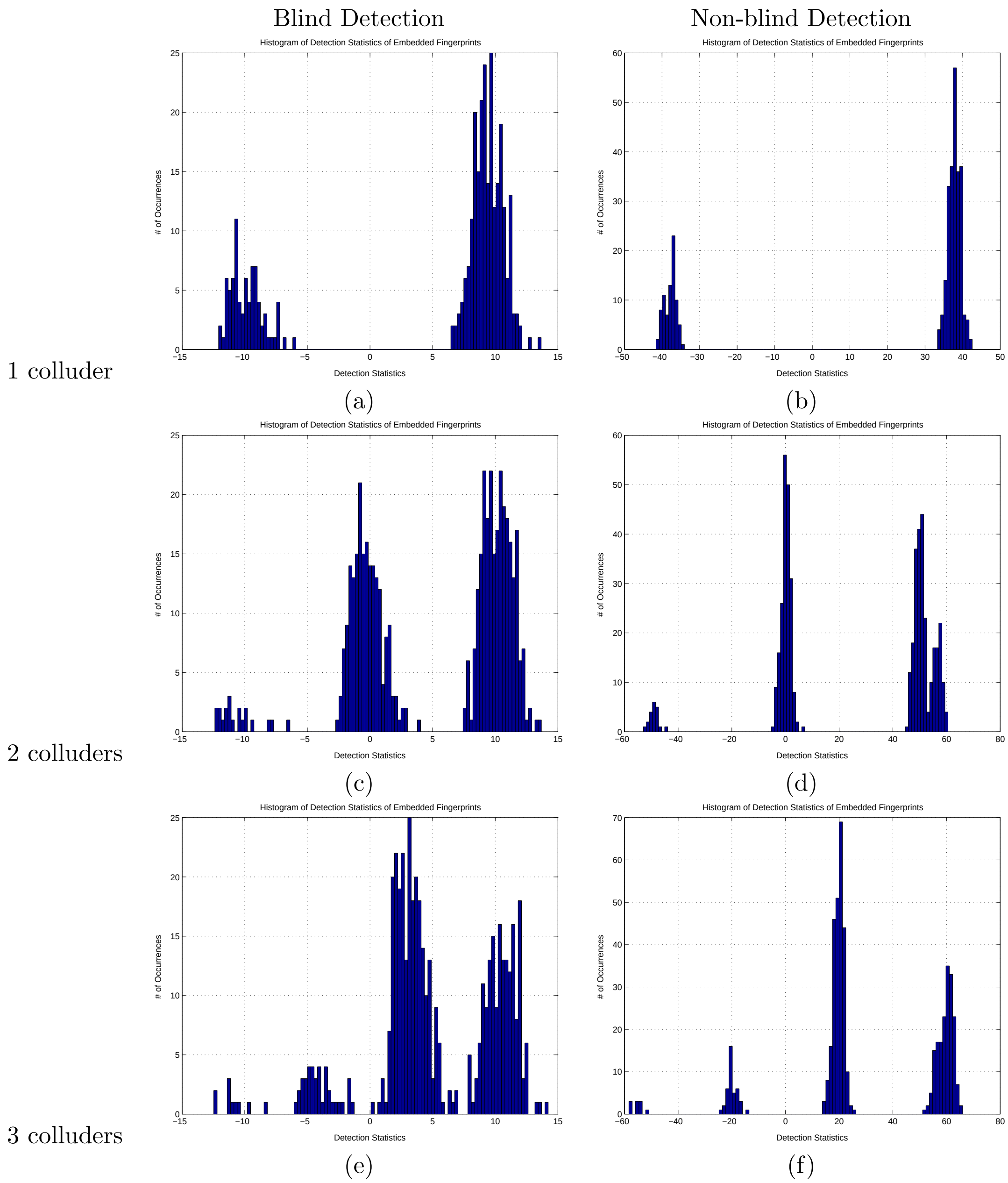


Figure 9: Histograms of detection statistics of embedded fingerprints: (top row) single fingerprint case, (middle row) 2-user collusion case, (bottom row) 3-user collusion case; (left column) blind detection, (right column) non-blind detection.

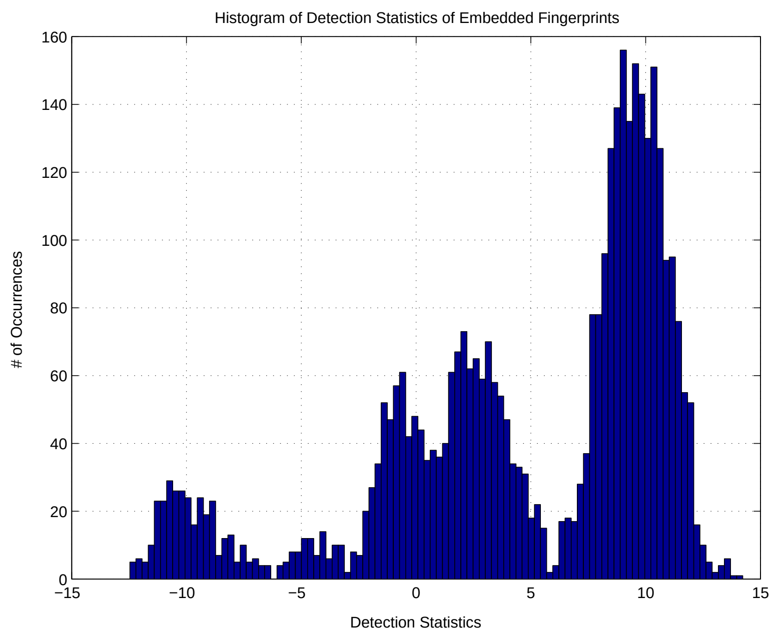


Figure 10: Aggregated histograms of blind detection statistics of embedded fingerprints covering from 1 to 3 colluders and 4 distortion settings.