Optical Engineering 46(5), 057004 (May 2007)

Practical fingerprinting system for images

Yu-Tzu Lin
National Taiwan University
Communications and Multimedia Laboratory
Department of Computer Science and
Information Engineering
Taipei, Taiwan
E-mail: linyt@cmlab.csie.ntu.edu.tw

Ja-Ling Wu
National Taiwan University
Communications and Multimedia Laboratory
Department of Computer Science and
Information Engineering and
Graduate Institute of Networking
and Multimedia
Taipei, Taiwan

Abstract. This paper presents a feasible system for image fingerprinting. Two important problems of fingerprinting are the design collusion-secure codes and the watermarking strategies for the fingerprints. We address these problems by proposing hierarchical fingerprint codes and a content adaptive watermarking scheme. It is challenging to find a practical collusion-secure code for fingerprinting because of the trade-offs between the fingerprint code's length, its collusion resiliency, and the size of the customer base. Therefore, instead of pursuing a high-performance code, we construct a new fingerprint code by concatenating existing codes using a hierarchical structure. The resulting code is efficient in decoding, much shorter than existing codes for similar parameters, and thus practical for real systems. In the fingerprinting, the codes are often very long if the applicable colluder-resisting ability and large customer bases are needed. The resulting acute degradation of fingerprinted content may be unacceptable. Hence the long-length watermark must be appropriately embedded into images to preserve the image quality. We design a strength-decision algorithm, on the basis of neural networks, to adaptively embed the long-length fingerprints with suitable magnitudes into different image regions. This adaptive watermarking technique maintains the combination of robustness and imperceptibility without the effort of dealing with visual models. In addition, we analyze the disturbance due to collusion attacks on images and propose an optimization algorithm that can select better embedding positions to resist collusion attacks and preserve acceptable transparency of the watermark according to different multimedia contents. Experimental results show the high detection correctness of the traitor tracing. It implies that our fingerprinting system, constructed by applying a hierarchical fingerprint code to the content-adaptive watermarking scheme, is effective for multimedia application. [DOI: 10.1117/1.2740765]

Subject terms: fingerprinting; watermarking; traitor tracing; collusion resistance.

Paper 060671 received Aug. 31, 2006; accepted for publication Nov. 17, 2006;

1 Introduction

In applications of multimedia distribution, such as pay TV, data authorized to some privileged users are guarded against unprivileged ones. A fingerprinting system protects media data by embedding a unique key to the host data of each user. These keys can be used to identify the source of illegal copies. However, a coalition of users may collude to make an illegal copy that is different from each colluder's copy. Collusion-resistant fingerprinting is designed to solve this problem. Boneh and Shaw initiated this effort,[1] which considers how to design fingerprint codes to resist the coalition of c traitors, called frameproof codes. Reference 2 proposes a traceability (TA) scheme and describes resilient schemes and threshold tracing schemes, which inherit from the key generation in broadcast encryption schemes.[3] Stinson presented several constructions[4,5] for broadcast encryption schemes, such as orthogonal arrays, universal hash families, and t-designs. Combinatorial properties of frameproof codes and traceability codes then were derived in Ref. 6, in which a c-TA scheme is defined and combinatorial structures like t-designs and packing designs are used to construct frameproof codes and traceability schemes. Besides theoretical works from the viewpoint of key generating in broadcast encryption, some workers proposed practical fingerprinting systems for multimedia: In Ref. 7, a collusion-secure fingerprinting scheme based on finite geometries is presented. Reference 8 constructs an anticollusion code for multimedia by using a balanced incomplete block design (BIBD),[9] and applies the code-modulation technique to reduce the spreading sequences significantly. It introduced the spread-spectrum watermarking strategy for multimedia fingerprinting, in which the orthogonal noise-like fingerprints are suitable for multimedia. References 10 and 11 also construct collusion-secure fingerprints based on noiselike signals.

In addition to the combinatorial methods and orthogonal-signal-based approaches, another trend of fingerprint codes employs the concept of error-correcting codes (ECCs). The idea is that one can identify the colluders' codewords if the minimum distance of the code is large enough. In Ref. 12, Reed-Solomon codes are used to construct a traceable code, and list-decoding techniques are used to find all possible pirate coalitions efficiently. The

---

authors also described how coding-theoretic techniques may be applied to the traitor-tracing problem for linear codes.

Combinatorial design can help find structured fingerprinting codes, which provide good properties for uniquely identifying colluders. But combinatorial design is not trivial; thus the construction of this type of codes is difficult and their scalability is limited. The ECC-based fingerprinting codes treat the fingerprint-generating problem as an ECC design problem. The mathematical operations on finite fields for ECCs, such as Reed-Solomon codes, provide more regularity and efficiency in both encoding and decoding than do the combinatorial methods. However, the attempt to increase the minimum distance between codewords results in an abrupt increase of the code length. The fingerprinting systems based on orthogonal signals are suitable for multimedia data and have good performance in the literature[8,10,11] but they can be broken by only a few dozen colluders.[11] In this paper, we try to construct a new fingerprint code by means of a hierarchical structure, which has both the strong identifying ability of the combinatorial design codes and the noiselike property of the orthogonal-signal-based methods. Moreover, the hierarchical construction methods significantly shorten the fingerprint codes.

Digital fingerprinting often embeds the key to the protected data using watermarking techniques. Most of the watermarking schemes are developed to fit the following requirements: robustness, high capacity, and fidelity. However, trade-offs should be considered among these three requirements. This need is apparent in fingerprinting problems because of the long-code length mentioned. Multimedia data are not so sensitive to disturbance in their values as other digital data; this implies that multimedia data can tolerate more embedding bits. Furthermore, human visual system (HVS) models have been introduced into some watermarking schemes[13–15] to improve the quality of watermarked images by embedding watermarks with maximum strength based on visual models, so as to reach a compromise among the prescribed three watermarking requirements. However, the construction of an HVS model involves the estimation of the image quality, which is still an open problem. Therefore, some researchers apply neural networks to decide the maximum watermarking strength automatically. Reference 16 uses the backpropagation network to compute the maximal tolerable watermark strength of an image and takes the first 4096 discrete wavelet transform coefficients as the input of the network. But in the training process, the user may be baffled when assigning the watermarking strength to the whole image, and the assignments may be biased because this method ignores the variety of different regions in an image. Reference 17 feeds the discrete cosine transform (DCT) coefficients, in one 8×8 block, to the neural network and obtains the embedding strength of this block. But the choice, in the training process, of appropriate embedding values of an 8×8 block is difficult for humans. In our work, a neural network is used to find suitable watermarking strengths for various image regions, and the learning is unsupervised.

Collusion attacks are critical to the fingerprinting problem. A group of legal users may collude by combining their fingerprints to create pirate copies and spread them. These traitors are intent on destroying their unique identification and making it hard for an arbiter to trace them. Collusion-resilient fingerprinting schemes are developed to resist the colluders' malice. The fingerprint design previously mentioned tries to construct codes resistant to collusion attacks; nevertheless, poor watermarking strategies may weaken the resistance of such schemes. In Ref. 7, the position sequences for embedding watermarks are generated randomly from the user key, so as to avoid the fingerprint bits' being eliminated by collusion attacks. Reference 18 proposes a video fingerprinting architecture by combining fingerprinting and decryption based on partial encryption. There are two types of encryption keys: one serves as pointers to subsets of coefficients to encrypt, and the other is a scrambling key dictating the order in which the coefficients are permuted within a given subset. Instead of building the anticollusion scheme by randomness or encryption, we extend the idea of watermark optimization[19] and devise an algorithm to select suitable embedding positions for fingerprints in order to intensify the collusion resistance.

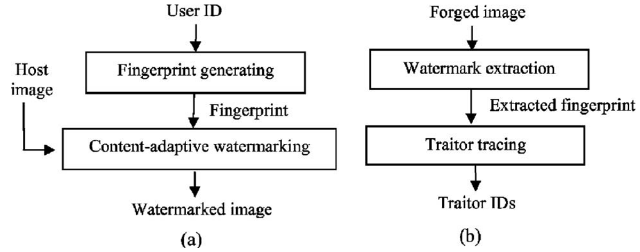


Fig. 1 System overview: (a) fingerprinting scheme, (b) traitor-tracing scheme.

Few works have considered practical parameters for actual applications. In this paper, we try to find a practical multimedia fingerprinting scheme to take both the need for a large number of fingerprints for numerous users and the characteristics of multimedia data into account. This multimedia fingerprinting scheme embeds a hierarchical fingerprint code into appropriately selected positions with the strength computed from a watermark-strength decision technique to make the code applicable to multimedia.

We first briefly describe our system in Sec. 2. In Sec. 3, we present the hierarchical fingerprint code used for traitor tracing. Details of content-adaptive watermarking strategies are provided in Sec. 4. Section 5 presents our experiments, and Sec. 6 gives the concluding remarks.

2 System Overview

Figure 1 shows our system architecture. Figure 1(a) illustrates the fingerprinting process, in which fingerprints are generated according to user IDs and embedded into the host image using an adaptive image watermarking technique, and the produced watermarked images are then assigned to users individually. In the fingerprint-generating step, a collusion-secure code is constructed from the proposed hierarchical structure. The content-adaptive watermarking scheme determines the maximum watermarking strength locally by the self-organizing map (SOM) neuron network, using Fourier transform coefficients, and picks suitable embedding positions from the candidate positions for the purpose of resisting collusion attacks (as detailed in Sec. 4).

After watermark embedding, each user acquires a particular watermarked copy with his particular fingerprint in it.

If some legal users collude to produce a pirate image, at least one of the colluders should be identified. We show the traitor-tracing scheme in Fig. 1(b). First, the pirate code is extracted from the forged image in the watermark extraction stage. The traitors can be identified according to the extracted code. Then we find at least one of the colluders by decoding the hierarchical fingerprint code (the decoding algorithm is detailed in Sec. 3).

3 Collusion-Secure Code

3.1 *Introduction*

As mentioned in Sec. 1, there are three types of collusion-secure codes: combinatorially designed codes, ECCs, and orthogonal signals, which all have their merits and shortcomings. In this section, a hierarchical construction method for fingerprint codes is detailed. For ease of explanation, several terms for collusion-secure codes are briefly described in the following.

3.1.1 *c-Secure codes*

3.1.1.1 Definition 1[1]. A code Γ is totally c-secure if there exists a tracing algorithm A satisfying the condition: If a coalition $\mathcal{C}$ of at most c users generates a word x, then $A(x) \in \mathcal{C}$.

Definition 1 says that a code is c-secure if there exists an algorithm that can identify one of the codewords of colluders. But unfortunately, when $c > 1$, totally c-secure codes do not exist. Boneh and Shaw[1] therefore use randomness to derive c-secure codes with error ε that enable the capture of one member of the illegal coalition with probability at least $1-\varepsilon$. They introduced an (l,n) code that is n-secure with error ε for any $\varepsilon > 0$. Let c_m be a column of height n in which the first m bits are 1 and the rest are 0. The code $\Gamma_0(n,d)$ consists of all columns $c_1, c_2, \ldots, c_{n-1}$, each duplicated d times. For example, the code $\Gamma_0(4,3)$ for four users u_0, u_1, u_2, u_3 is

u_0:111111111,

u_1:000111111,

u_2:000000111,

u_3:000000000.

In order to construct a logarithmic-length c-secure code with error ε, they then composed $\Gamma_0(n,d)$ and C, in which C is an error-correcting code. Although $\Gamma_0(n,d)$ is "n-secure," one does not use it directly, because the length of this code is $O(n^3 \log(n/\varepsilon))$, which is impractical in many applications.

3.1.2 *c-TA codes*

3.1.2.1 Definition 2[12]. Suppose C is an $(l,n,d)_q$ ECC and $c \geq 2$ is an integer. Let $C_i \subseteq C$, $i=1,2,\ldots,t$, be all the subsets of C such that $|C_i| \leq c$. Then C is a c-TA code if for all i and all $x \in \mathrm{desc}(C_i)$, there is at least one codeword $y \in C_i$ such that $|I(x,y)| > |I(x,z)|$ for any $z \in C \setminus C_i$ (in which $I(x,y)=\{i : x_i = y_i\}$, and $\mathrm{desc}(C_0)=\{x \in Q^N : x_i \in \{a_i : a \in C_0\},\ 1 \leq i \leq N\}$, where Q is the alphabet set).

In Sec. 3.2, we present a hierarchical scheme for the code construction, which yields a new fingerprint code based on existing simple collusion-secure codes. Section 3.3 gives two examples [one is $\Gamma_0(n,d)$, and the other is a c-TA code] to illustrate the significant improvement in code length with the hierarchical scheme. For the implementation, we use $\Gamma_0(n,d)$ to hierarchically construct the intended code, which shows the capability of the proposed code construction scheme.

3.2 *Hierarchical Scheme of Traceability Codes*

Reference 10 proposed a joint fingerprint design and distribution design using a tree-based fingerprint scheme to reduce the bandwidth requirement. Reference 10 also pointed out that the fingerprint modulation based on time-division multiple access (TDMA) improves the bandwidth efficiency but at a cost in collusion resiliency. Therefore, a fingerprint system combining TDMA and code-division multiple access (CDMA) modulation has been proposed to enhance the TDMA method. In group-oriented fingerprinting, as presented in Ref. 11, users who are likely to collude are assigned correlated fingerprints. Both of these two methods detect colluders based on the probabilistic estimation of fingerprint signals. We propose a new hierarchical scheme for traceability-code construction, which can not only shorten the code length significantly with a fast detection procedure, but also enable group-based fingerprinting. Moreover, we analyze the hierarchical code according to the complexity of both the code length and the decoding time, rather than a statistical model for fingerprint signals as in Refs. 10 and 11.

Hierarchical tree structure is also employed in many key-management systems for encrypted broadcasting.[20] Under the tree scheme, security properties can be proven with suitable cryptographic assumptions, and the flexibility is also superior to that of other schemes. Therefore, the hierarchical code construction method not only effectively shortens the traceability code and enhances the tracing ability, but also makes it easier to manage keys with the tree scheme.

3.2.1 *Segment codes*

Figure 2 shows the structure of the proposed hierarchical traceability code. The traceability code is divided into several hierarchies. Each hierarchy represents a segment of the fingerprint code. Let b be the number of branches of the tree, n be the number of users, and l be the length of the code. In order to encode the fingerprint segment for every hierarchy, the segment code needs b codewords, one for each branch. That is much less than the n codewords when encoding without tree structures. If the b-ary tree is balanced, the number of hierarchies, h, equals $\log_b n$. The fingerprint is then the concatenation of fingerprint segments of all hierarchies, i.e., $fs_1 fs_2 \ldots fs_h$. For example, if the segment code has codewords $w_0, w_1, \ldots, w_{b-1}$, then the finger-

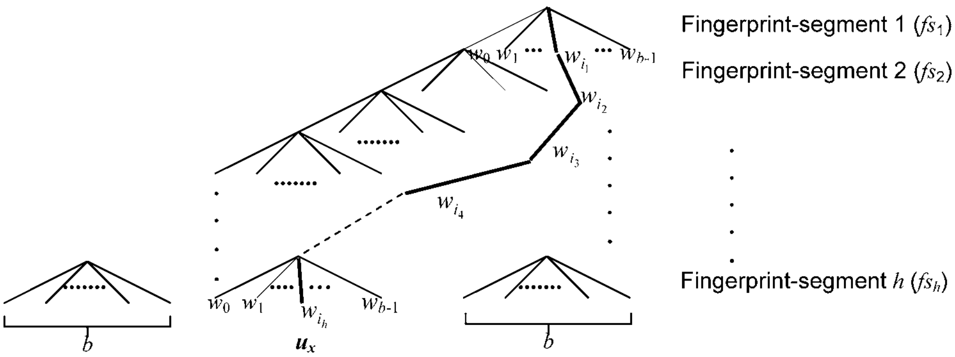

Fingerprint-segment 1 (fs_1)
Fingerprint-segment 2 (fs_2)
Fingerprint-segment h (fs_h)
w_0 w_1 ... w_{i_1} ... w_{b-1}
w_{i_2}
w_{i_3}
w_{i_4}
w_0 w_1 w_{i_h} w_{b-1}
b
u_x
b

Fig. 2 The structure of the proposed hierarchical traceability codes.

print of u_x is constructed by concatenating every segment codeword: $w_{i_1}, w_{i_2}, w_{i_3}, \ldots, w_{i_h}$, where w_{i_k} is the codeword of the k'th hierarchy for u_x and where $\forall k \in (1,h)$, $w_{i_k} \in \{w_0, w_1, \ldots, w_{b-1}\}$ (as illustrated in Fig. 2).

As already mentioned, the segment code should be capable of encoding b codewords, and the number of leaves of the tree should be large enough to represent all users. The number of branches can also be regarded as the number of groups in that subtree. That is to say, we need b codewords to discriminate b groups rather than b users. This implies that the size of customer base decreases abruptly and the collusion size considered can be reduced to no larger than b. As a result, the code length of this traceability code for each hierarchy is shortened, and so is the total code length.

The total code length of the so obtained traceability code is

$$l = hl' = (\log_b n)l', \tag{1}$$

where l' is the length of the code segment of one hierarchy. The value of l' depends on the code used in the hierarchy, the branch size b, and the collusion resiliency. But as shown in Fig. 3, in the j'th level of the tree, all b^j groups of branches are encoded by the same code, i.e., all groups have the same codewords. For example, if w_i is a codeword of the segment code, then all groups have the codeword w_i. The question is: How to discriminate one w_i from other w_i's in different groups at the same level? For example, in a four-level tree, there are three codewords $u_1 = [w_0 \quad w_1 \quad w_2 \quad w_3]$, $u_2 = [w_1 \quad w_1 \quad w_2 \quad w_3]$, and $u_3 = [w_0 \quad w_2 \quad w_1 \quad w_1]$, where u_1 and u_2 only differ in the first segment codeword. When u_2 and u_3 come together to do the collusion, they can generate the codeword of $[w_0 \quad w_1 \quad w_2 \quad w_3]$ with u_3 contributing the w_0 in the first segment and u_2 contributing the remaining segments. Thus these two colluders can successfully frame an innocent user u_1. Therefore, extra bits must obviously be paid for encoding the groups. The additional code length depends on the group code (i.e., another collusion-secure code employed for encoding the groups) used, and the group code can vary with the level. In Sec. 3.1.2, we show how to discriminate groups by orthogonal signals and construct hierarchical fingerprint codes using existing collusion-secure codes as the segment code.

3.2.2 *Group codes*

Cox et al.[21] proposed the spread-spectrum watermarking method, which embeds watermarks as noiselike signals. Some attempts construct fingerprints by orthogonal modulation, and the produced fingerprints are spread-spectrum sequences. Reference 10 employed Gaussian-distributed fingerprints by orthogonal modulation and used a

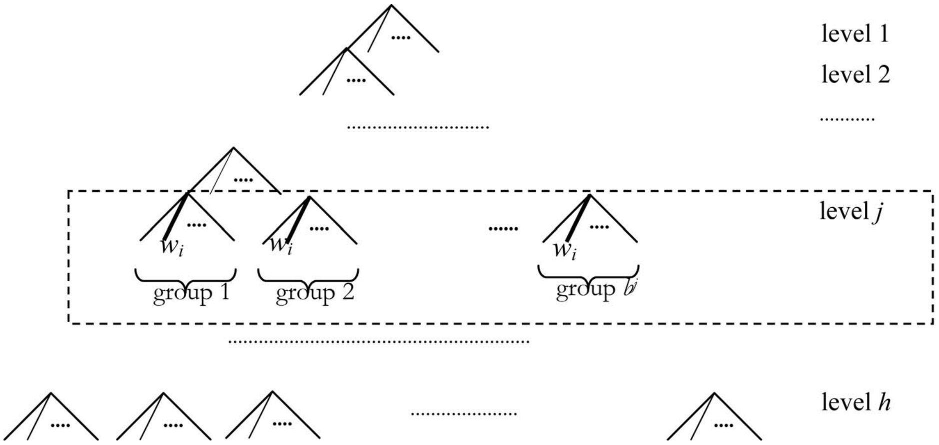


Fig. 3 The groups in one hierarchy of the tree.

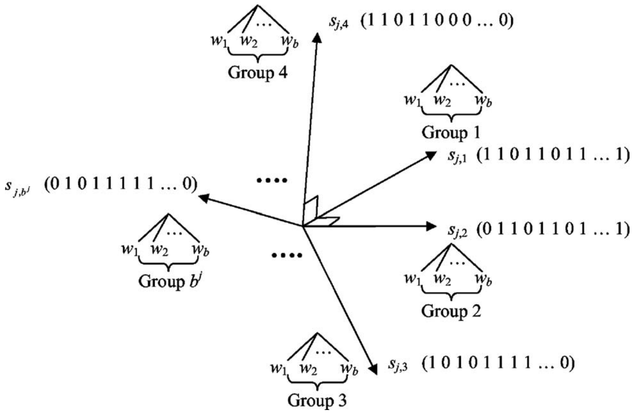

$s_{j,4}$ (1 1 0 1 1 0 0 0 ... 0)
w_1 w_2 ... w_b
Group 4
w_1 w_2 ... w_b
Group 1
$s_{j,1}$ (1 1 0 1 1 0 1 1 ... 1)
s_{j,b^j} (0 1 0 1 1 1 1 1 ... 0)
....
$s_{j,2}$ (0 1 1 0 1 1 0 1 ... 1)
w_1 w_2 ... w_b
Group b^j
....
w_1 w_2 ... w_b
Group 2
w_1 w_2 ... w_b
$s_{j,3}$ (1 0 1 0 1 1 1 1 ... 0)
Group 3

Fig. 4 The b^j orthogonal signals used to encode b^j groups in the j'th level of the tree for the proposed hierarchical fingerprinting system. These pseudorandom signals are nearly orthogonal to each other.

likelihood-based approach to estimate the number of colluders. Reference 8 modulated BIBD codes by orthogonal bases that are spread-spectrum sequences. The noiselike signals of this type of fingerprints are obviously suitable for multimedia contents, but the tracing ability is limited when the size of the collusion coalition or the customer base becomes larger.[11] However, the limit on c or n does not crash our system, because the orthogonal signals are employed in encoding the groups of branches, which have fairly small size.

Figure 4 shows how to discriminate different groups in the hierarchical tree: employing b^j nearly orthogonal signals to modulate b^j groups in the j'th level of the tree. The pseudorandom modulation is employed because it can avert extra bits and the effort of designing the group code. That is, we pseudorandomly modulate the segment-codewords in each of the b^j groups of the j'th level by a different random sequence s_{jg} according to the corresponding group key; thus different groups are orthogonal to each other. Then the pseudorandom signals become our group code. So even though each of the groups has the same segment code, the segment codeword w_i of group j_1 can be discriminated from that of group j_2 because of the different modulating sequences. The pseudorandom modulation is given by

$$\tilde{w}_{j,g,i} = w_i \oplus s_{j,g} = w_i \oplus P(K_{j,g}), \tag{2}$$

where w_i is the codeword vector for the i'th branch, $\oplus$ is the exclusive-or operation, $P(\cdot)$ is a pseudorandom-sequence generator, $K_{j,g}$ is the group key of the g'th group at the j'th level, and $\tilde{w}_{j,g,i}$ is the pseudorandomly modulated codeword for w_i in the g'th group at the j'th level. That is, groups are orthogonal to each other after the pseudorandom modulation. Therefore, the hazard illustrated in Sec. 3.2.1 will not occur, because w_i is modulated by different pseudorandom sequences for different groups: $u_1=[w_0 \oplus s_{0,0}w_1 \oplus s_{1,0}w_2 \oplus s_{2,1}w_3 \oplus s_{3,6}]$ and $u_2=[w_1 \oplus s_{0,0}w_1 \oplus s_{1,1}w_2 \oplus s_{2,5}w_3 \oplus s_{3,22}]$ cannot frame $u_3=[w_0 \oplus s_{0,0}w_2 \oplus s_{1,0}w_1 \oplus s_{2,2}w_1 \oplus s_{3,8}]$.

The demodulation simply reverses the modulation operation:

$$w_i' = \tilde{w}_{j,g,i}' \oplus s_{j,g'} = \tilde{w}_{j,g,i}' \oplus P(K_{j,g'}), \tag{3}$$

where w_i' is the demodulated segment codeword, $\tilde{w}_{j,g,i}'$ is the extracted fingerprint segment, which has been pseudorandomly modulated, and g' is the suspected group number in the j'th level. If we demodulate the fingerprint segment by the corresponding pseudorandom signal, then it will be correctly restored, else the demodulated sequence will be almost all zeros because the pseudorandom sequences are nearly orthogonal to each other, that is,

$$w_i' = \begin{cases} =\tilde{w}_{j,g,i}' \oplus s_{j,g'} = w_i & \text{if } g = g', \\ \approx \vec{0} & \text{otherwise.} \end{cases} \tag{4}$$

Of course, the extra step of the pseudorandom modulation will introduce errors if the pseudorandom signals are not "random enough" (as may result from the limited length of the signal sequences), but this kind of errors can be restricted when h is not too large. In fact, the number of levels used for discriminating groups is small, because most of the levels of the tree contribute to differentiating users in the same group. The higher levels in the tree often have fewer branches (see Fig. 5), so fewer operations of pseudorandom modulation are performed. Moreover, colluders are often in the same group. That is to say, traitors' codewords are almost pseudorandomly modulated by the same random signal for every hierarchy. Consequently, er-

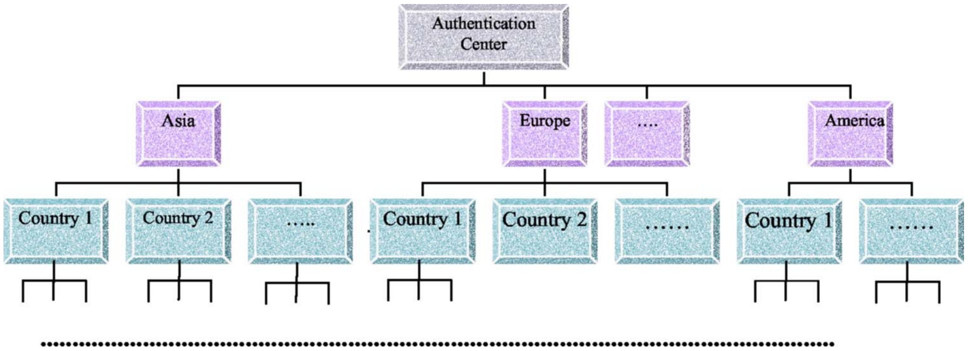


Fig. 5 The hierarchical authentication structure of the fingerprinting system.

rors introduced from the pseudorandom modulation are limited. In addition, we use the list decoding for the segment-code decoding, i.e., pick the codewords that are close to the demodulated word w_i' as the decoding results; thus it is almost impossible for the result to be affected by errors introduced by the pseudorandom modulation.

3.3 *Constructing Tree-Structured Traceability Codes from Existing Collusion-Secure Codes*

Because of the limited collusion resiliency when n or c is large for orthogonal-signal-based methods and the difficulty of finding ECC-based or combinatorial-designed codes with reasonable code length when n or c is large, we combine different types of codes to obtain a superior code. In this subsection, two existing traceability codes are used as the segment codes in the tree-structured construction strategy to show the effectiveness of the proposed method, which is detailed subsequently.

We then describe examples of the hierarchical construction.

First, we employ the c-secure code Γ_0 in the tree-structured construction. The code length of this n-secure code with error ε is

$$2(n^3 - n^2)\log(2n/\varepsilon), \tag{5}$$

where n is the number of codewords. Because we apply this code to encode b branches, n in Eq. (5) must be replaced by b. Therefore, the code length of the fingerprint segment for one hierarchy is

$$l' = 2(b^3 - b^2)\log(2b/\varepsilon). \tag{6}$$

The total length of the fingerprint is

$$l = [\log_b n][2(b^3 - b^2)\log(2b/\varepsilon) + l_{\mathrm{ps}}], \tag{7}$$

in which the last term l_{ps} is the number of required bits of pseudorandom sequences for encoding at most n/b groups at one level, which is $O(\log_b(n/b))$. The error rate for traitor tracing is

$$\varepsilon_{\mathrm{total}} = 1 - [1 - (\varepsilon + \varepsilon_p)]^h = 1 - [1 - (\varepsilon + \varepsilon_p)]^{\log_b n}, \tag{8}$$

where ε_p is the error produced by pseudorandom modulation.

Because the branch size is always small and invariant, b can be treated as a constant; then the code length is reduced from $O(n^3 \log(n/\varepsilon))$ to $O([\log n][\log(1/\varepsilon) + \log n])$.

The second example is to employ the sequential c-TA code proposed in Ref. 22 for the tree-structured construction, in which the code length $l = O(n)$. Then the length of the code segment for one hierarchy is

$$l' = O(b + \log_b n); \tag{9}$$

thus the total length of the code is

$$l = hl' = O(\log_b^2 n). \tag{10}$$

Thus the code length is reduced from $O(n)$ to $O(\log_b^2 n)$.

One of the important superiorities of the tree-structured code-construction scheme is that the new collusion resilience c of the tree-structured fingerprint code increases to approach n if the errors resulting from the pseudorandom modulation are restricted. By identifying all possible branches in every hierarchy, all the colluders can be traced correctly. This good characteristic is demonstrated in Sec. 5.3.

The following pseudocode states the decoding algorithm of the hierarchical traceability scheme, in which **Decode** $(\cdot)$ is the decoding algorithm for the segment code that can detect the collusion branches for the current node, and **Demodulate** $(\cdot, K_g)$ is a pseudorandom demodulation procedure according to the group key K_g:

```
Trace(node $n_i$)
{
    Decode(Demodulate (Modulated_Segment_Code
    ($n_i$), Group_Key($n_i$)))
    for each suspicious branch $b_j$
            add the corresponding child node $c_j$ into
            the set of the collusion children $C$
    end

    for each collusion child $c_j \in C$
            Trace($c_j$);
    end
}
```

The tree is traced from the root. The first code segment is pseudorandomly demodulated according to the root key. The demodulated sequence is then decoded to decide collusion groups in the first level. For each of the collusion groups, the tracing procedure proceeds recursively. The time complexity of this algorithm is $O((\log_b n)c_a)$, where c_a represents the actual collusion size.

In the implementation, we construct a quaternary tree (i.e., b is 4) and use the 4-secure code $\Gamma_0(4, 288)$ with error 0.001 to encode every two branches. The resulting length of the code segment is 864, and the total length of the five hierarchies for encoding 1024 codewords is 4320, which is much shorter than the codeword length (31,228,440,620) of the naive Γ_0.

4 Content-Adaptive Watermarking

To achieve high peak SNR (PSNR) while preserving robustness of the watermark and resistance to collusion attacks, a content-adaptive watermarking scheme is proposed. We build an SOM[23] neural network learning different levels of tolerance of quality distortion for different multimedia contents, to simulate the HVS. After the SOM neural net is trained, it learns to adjust the watermark strength adaptively according to the multimedia contents, so as to minimize the distortion of the content quality while preserving robustness. In this paper, we focus on image watermarking, and the proposed technology can be applied to other types of media in a similar way. Another problem of resistance to collusion attacks,—viz., where to embed—is considered and solved by an optimization procedure, which is described in Sec. 4.2.

4.1 *SOM Architecture*

An SOM neural network can imitate the mechanism of the cerebral cortex. A 2-D map is used in our architecture to represent the cortex map, involving the sensitivity of image distortion. During the training period of the network, each unit with a positive activity within the neighborhood of the winning unit updates the value it assigns to the features of certain image appearances, using

$$\varpi_i(t+1) = \begin{cases} \varpi_i(t) + \alpha(t)[x - \varpi_i(t)], & i \in N_c, \\ 0, & \text{otherwise.} \end{cases} \qquad (11)$$

where $\varpi_i(t)$ is the weight vector of the i'th neuron unit in the t'th iteration, $\alpha(t)$ is the learning rate, x is the current input vector, and N_c is the list of unit indices that make up the neighborhood.

After iterations of updating values of neurons by Kohonen's algorithm,[23] nodes in the 2-D array organize themselves in response to the input vectors. The resulting map projects the input vectors onto a 2-D map preserving the natural topology of training data, which contains the feature vectors of various types of images, including smooth, high-frequency, and regular-texture images.

Initially, a 2-D SOM with 4×4 neurons is constructed. Each node is randomly assigned to feature values of one of the training samples. The learning algorithm performs the weight-update process using an approximation to the Mexican-hat function.[24] If one node is too coarse to represent a certain category of training data, it is divided into four child nodes. Further dividing should be performed if nodes after the first division are still too rough, and so on. The architecture of our SOM network is shown in Fig. 6.

The feature vectors used in our SOM are obtained from performing the discrete Fourier transform (DFT) on the original data. Vectors derived from images with similar characteristics are trained to gather together in adjacent nodes. DFT coefficients are used as features because, in the Fourier domain, significant features are compressed into a few coefficients only. Moreover, the good properties of scaling invariance and rotation invariance in the DFT domain will make our features approximate the human's impression closely.

In the training stage, training samples of size-$m \times m$ images are first categorized manually. For each training image, we increase the watermark strength gradually until the watermark is perceptible, and then classify this watermarked image into that category in which all images can

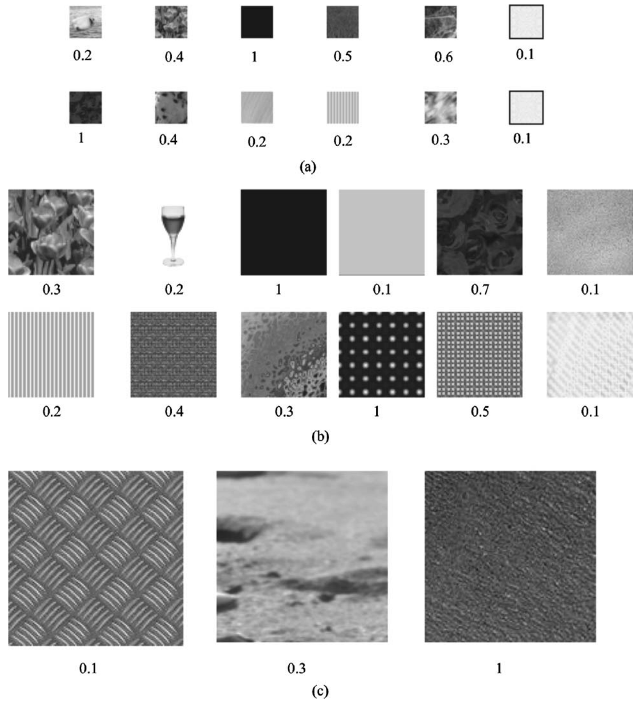


Fig. 7 Some of the training images of the SOM. Values below the images are the corresponding JNDs (i.e., the maximum strength for imperceptible watermarking). The sub-image size *m* is (a) 32, (b) 64, and (c) 128.

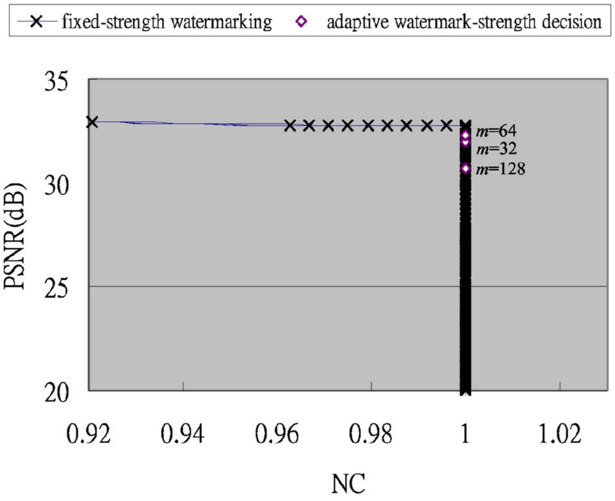


Fig. 8 The NC-PSNR curves for fixed-strength watermarking versus adaptive watermark-strength decision. The size of the test images is 256×256, the code length of the fingerprints is 24,576, and the subimage size *m* is 32, 64, or 128.

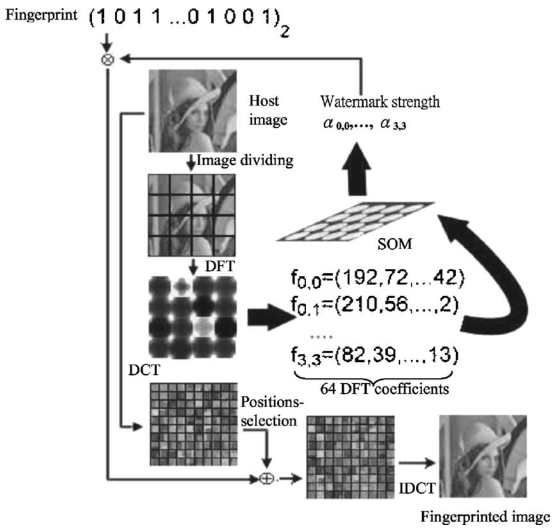
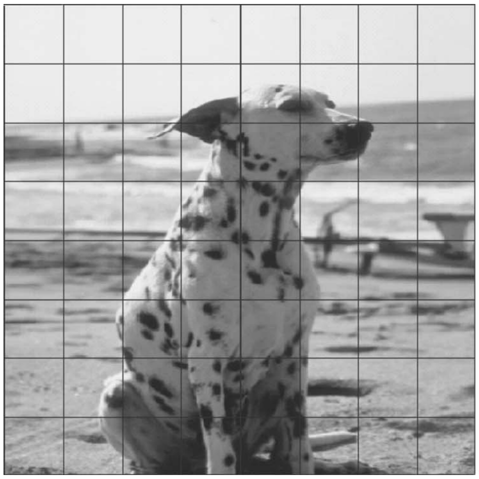


Fig. 9 The fingerprint embedding process.



0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
0.1	0.1	0.1	0.2	0.2	0.2	0.1	0.1
0.3	0.2	0.2	0.3	0.3	0.3	0.2	0.3
0.1	0.1	0.1	0.2	0.3	0.1	0.1	0.2
0.2	0.3	0.3	0.2	0.3	0.3	0.2	0.3
0.2	0.2	0.3	0.3	0.3	0.3	0.3	0.3
0.2	0.2	0.3	0.3	0.7	0.2	0.2	0.2
0.3	0.3	0.6	0.6	1.0	0.3	0.3	0.3

Fig. 10 The corresponding watermark strength for each of the subimages of the "Dog" image.

tolerate equal strength of the watermark; the corresponding strength is called the just-noticeable distortion (JND) of that category, and its value has been normalized to the closed interval [0,1]. In the network learning stage, we take the first 64 DFT coefficients of the classified training images as input vectors, which can be represented as a 64-dimensional vector. If the feature vector of the current training sample is the nearest one to the center of certain category, the coordinate of that center is moved toward the training vector. After iterations of learning, the SOM then eventually memorizes the maximum watermark strength imperceptible to humans for every category of the training

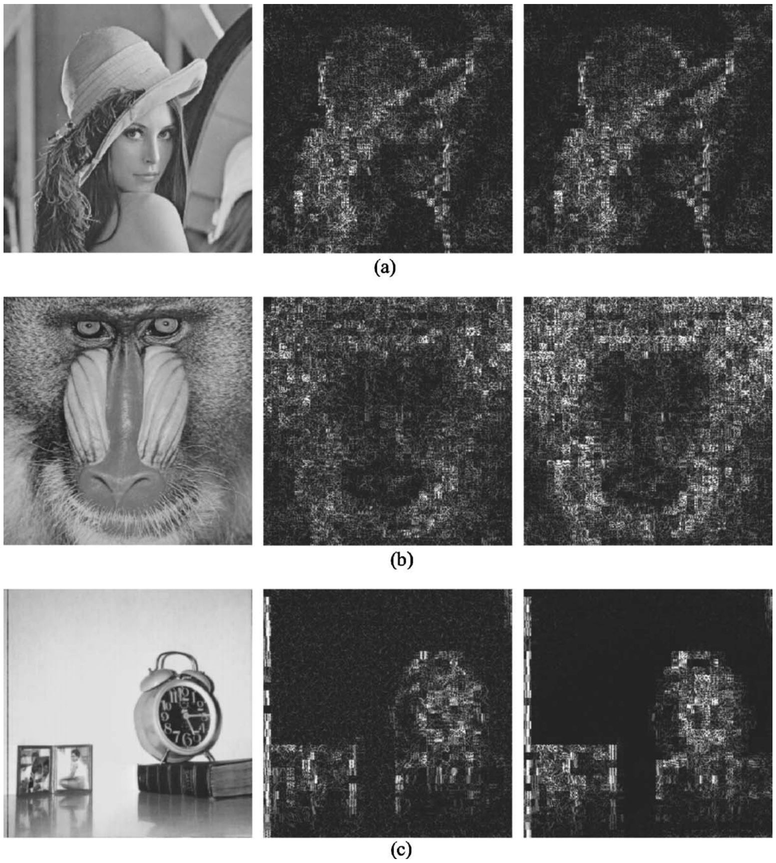


Fig. 11 The watermarks produced by fixed-strength watermarking and adaptive watermark-strength decision for some example images: (a) "Lena," (b) "Baboon," and (c) "Clock." Left: original images. Middle: difference images for fixed-strength watermarking. Right: difference images for adaptive strength-decision watermarking. Here "difference image" means the difference between the original image and the watermarked image.

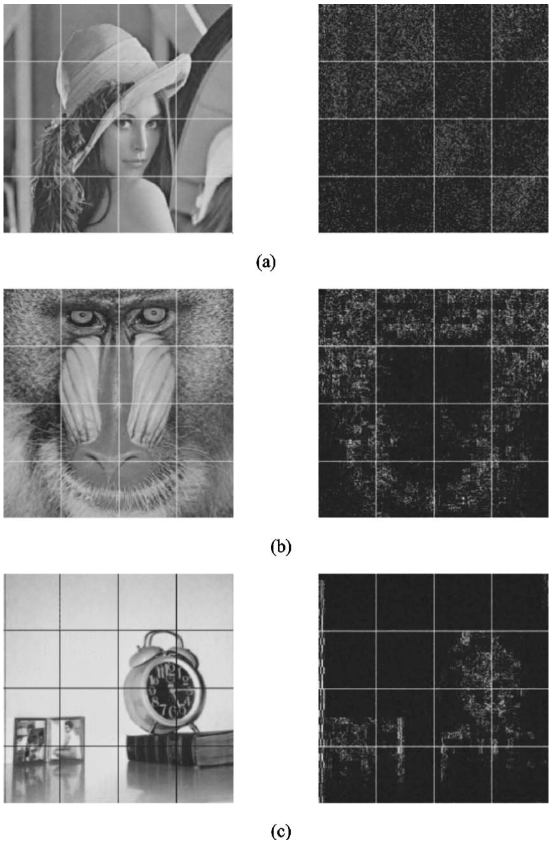


Fig. 12 Original images (left) and the difference images between watermarks of FSW and ADSW (right). The dividing lines illustrate the blocks for watermark decision (where $m=64$). (a) “Lena,” (b) “Baboon,” and (c) “Clock.”

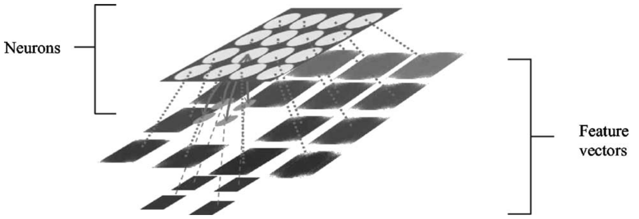


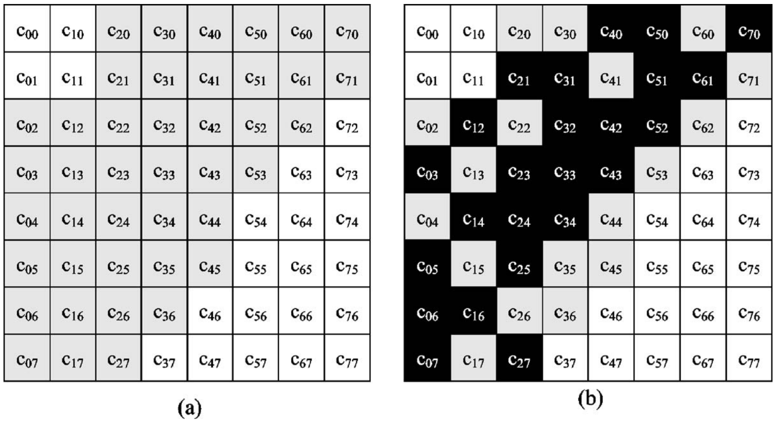
Fig. 6 The architecture of the adopted SOM network. Solid lines indicate the links between parent nodes and child nodes; dotted lines map neurons to feature vectors expressed in images.

images. In the recalling process, the input image is divided into several $m \times m$ subimages. These subimages are then fed to the neural network as inputs. If features of the current input subimage are most similar to features of a certain category, the watermark is embedded into the subimage according to the JND of that category. Some of the training samples of various sizes and their corresponding JNDs are shown in Fig. 7. It should be noted that JND equal to 1 means the watermark strength can be $\alpha_s \cdot 1$ (in which α_s is described below); in other words, the watermark strength still is bounded by α_s, and is not as large as possible. Therefore, the high JND of the third image in Fig. 7(b) can be explained in that areas with very high or low gray levels are less noticeable to humans.[13] We can also find that the sizes of images will affect a human’s perception and may cause the chosen JNDs to be distinct.

4.2 *Adaptive Watermark-Strength Decision*

After fingerprint generation, the fingerprint codewords should be embedded into the host image. First, the host image is divided into several subimages of size $m \times m$. Our experiments show that the performance is best when m is 64 (see Fig. 8, which is discussed later in this section). Second, the DFT coefficients of these subimages are fed to the SOM individually to get their watermark strengths. Finally, the fingerprint is added DCT coefficients (we select suitable coefficients to embed by using an optimization algorithm, which is detailed in Sec. 4.3) for every 8×8 block of the host image except the first one, according to the obtained strength in the previous stage. The watermark embedding procedure is illustrated in Fig. 9.

The proposed watermarking method is *oblivious*—in other words, the watermark detection does not need the original image. Suppose the fingerprint code is $(v_0, v_1, \ldots, v_{q^k-1})$, where $v_i \in \{0,1\}$. We modify the DCT coefficients block by block as follows:



c_{00}	c_{10}	c_{20}	c_{30}	c_{40}	c_{50}	c_{60}	c_{70}
c_{01}	c_{11}	c_{21}	c_{31}	c_{41}	c_{51}	c_{61}	c_{71}
c_{02}	c_{12}	c_{22}	c_{32}	c_{42}	c_{52}	c_{62}	c_{72}
c_{03}	c_{13}	c_{23}	c_{33}	c_{43}	c_{53}	c_{63}	c_{73}
c_{04}	c_{14}	c_{24}	c_{34}	c_{44}	c_{54}	c_{64}	c_{74}
c_{05}	c_{15}	c_{25}	c_{35}	c_{45}	c_{55}	c_{65}	c_{75}
c_{06}	c_{16}	c_{26}	c_{36}	c_{46}	c_{56}	c_{66}	c_{76}
c_{07}	c_{17}	c_{27}	c_{37}	c_{47}	c_{57}	c_{67}	c_{77}

(a)

c_{00}	c_{10}	c_{20}	c_{30}	c_{40}	c_{50}	c_{60}	c_{70}
c_{01}	c_{11}	c_{21}	c_{31}	c_{41}	c_{51}	c_{61}	c_{71}
c_{02}	c_{12}	c_{22}	c_{32}	c_{42}	c_{52}	c_{62}	c_{72}
c_{03}	c_{13}	c_{23}	c_{33}	c_{43}	c_{53}	c_{63}	c_{73}
c_{04}	c_{14}	c_{24}	c_{34}	c_{44}	c_{54}	c_{64}	c_{74}
c_{05}	c_{15}	c_{25}	c_{35}	c_{45}	c_{55}	c_{65}	c_{75}
c_{06}	c_{16}	c_{26}	c_{36}	c_{46}	c_{56}	c_{66}	c_{76}
c_{07}	c_{17}	c_{27}	c_{37}	c_{47}	c_{57}	c_{67}	c_{77}

(b)

Fig. 13 The embedding positions: (a) candidates for embedding DCT coefficients (shown in gray), and (b) one example of positions selected for embedding (shown in black).

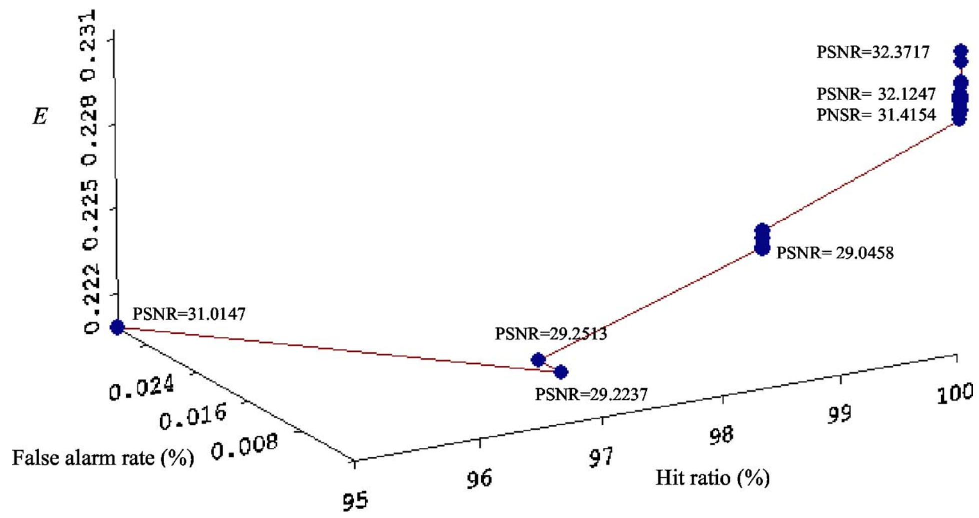


Fig. 14 The maximization of the energy function *E*. The curve shows the variations of hit ratio, false-alarm rate, and PSNR in the maximization process.

$$\tilde{x}_{k,b}=\begin{cases} x_{k,b-1}+\alpha_s\beta Q_k x_{0,b}, & \text{if } v_i=1,\\ x_{k,b-1}-\alpha_s\beta Q_k x_{0,b}, & \text{if } v_i=0,\end{cases} \tag{12}$$

where $x_{0,b}$ and $x_{k,b}$, respectively, are the dc and the k'th ac coefficient of the b'th DCT block; $k \in S$, $S \subseteq \{1,2,\ldots,63\}$, is the set of indices DCT coefficients selected for watermark embedding by our position-selection algorithm; $x_{k,b-1}$ is the k'th ac coefficient in the previous block; α_s is the watermark strength of the s'th subimage derived from the SOM; Q_k is the quantization value for the k'th ac coefficient in the default JPEG quantization table; and β is a scaling factor.

For example, suppose a 512×512 image needs to be watermarked and m is set to 64. The watermarking procedure is the following: First, we divide the original image into 4096 8×8 blocks and apply the DCT to them. Moreover, the original image is divided into 64 64×64 subimages and fed into the SOM to get 64 watermark strengths α_s, $s \in \{1,\ldots,64\}$. Figure 10 gives one example of the obtained watermark strengths. The sky area gets small strength values while the region under the legs gets large values.

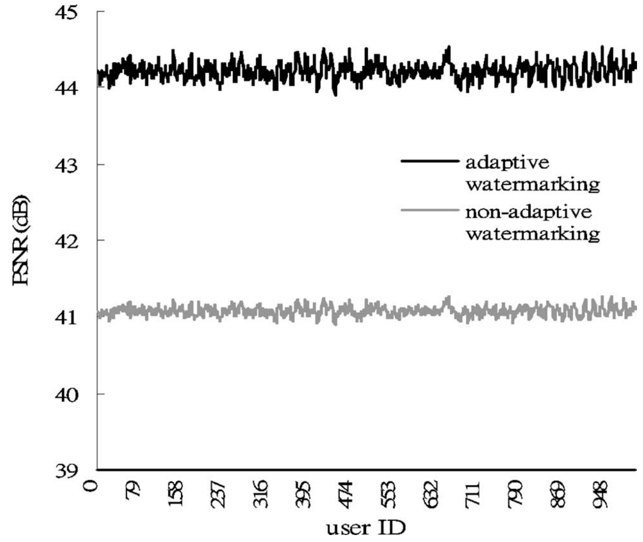


Fig. 15 Imperceptibility comparisons between the adaptive watermarking scheme and the naive watermarking scheme with fixed watermark strength.

The watermark can be extracted by simply comparing the values of $\tilde{x}_{k,b}$ and $\tilde{x}_{k,b-1}$. The watermark value is

$$\tilde{v}_i=\begin{cases} 1 & \text{if } \tilde{x}_{k,b} > \tilde{x}_{k,b-1},\\ 0 & \text{if } \tilde{x}_{k,b} < \tilde{x}_{k,b-1},\\ ? & \quad\text{otherwise.}\end{cases} \tag{13}$$

If the extracted value is ?, we decode it as an erasure. The erasures should be tolerated by the traitor detection procedure, which is introduced in the next section.

We measure the robustness of watermarking by the normalized correlation

$$\mathrm{NC}=\frac{\sum_i v_i \tilde{v}_i}{\sum_i v_i^2}. \tag{14}$$

The relationship between robustness (NC) and imperceptibility (PSNR) is shown in Fig. 8. The size of the test images is 256×256, and the fingerprint is a 16-bit code, and

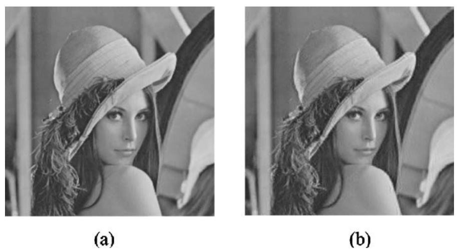


Fig. 16 Imperceptibility of the fingerprinting scheme: (a) the original image (256×256), and (b) one of the fingerprinted images.

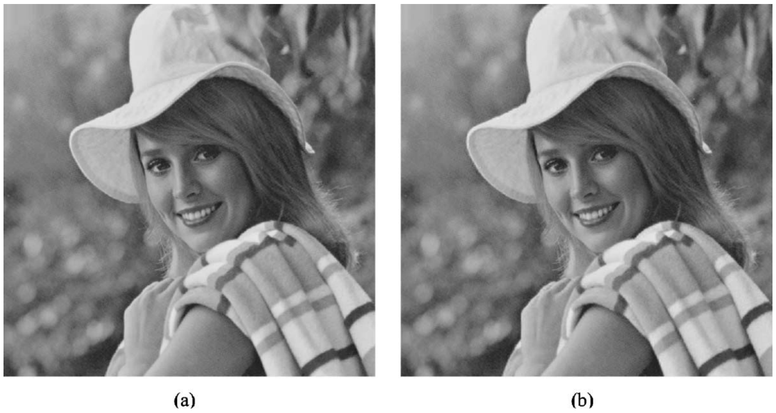

(a) (b)

Fig. 17 Imperceptibility of the fingerprinting scheme: (a) the original image (512 × 512), and (b) one of the fingerprinted images.

Table 1 Comparisons of robustness (in terms of NC values) to JPEG compression between ASDW and IA-DCT for various quality factors *Q*.

			NC					
Algorithm	Watermark length	Image	$Q=80$	60	40	20	10	5
ASDW	24,552	"Elaine"	0.80	0.79	0.76	0.66	0.54	0.82
			0.68	0.65	0.59	0.58	0.50	0.76
IA-DCT	24,936	"IM8"	0.9	0.78	0.66	0.51	0.38	0.22

Table 2 Comparisons of robustness (in terms of NC values) to JPEG compression plus cropping between ASDW and IA-DCT.

			NC				
Algorithm	Watermark length	Image	Crop	Crop; $Q=80$	Crop; $Q=60$	Crop; $Q=40$	Crop; $Q=20$
ASDW	24,552	"Elaine"	0.81	0.79	0.79	0.77	0.66
			0.70	0.68	0.62	0.57	0.52
IA-DCT	24,936	"IM8"	1	0.92	0.79	0.68	0.52

Table 3 Robustness (in terms of NC values) to contrast enhancing and noise addition.

			NC			
			Contrast enhancing		Noise addition	
Algorithm	Watermark length	Image	20%	40%	5%	20%
ASDW	24,552	"Elaine"	0.80	0.79	0.65	0.57
			0.76	0.75	0.64	0.56

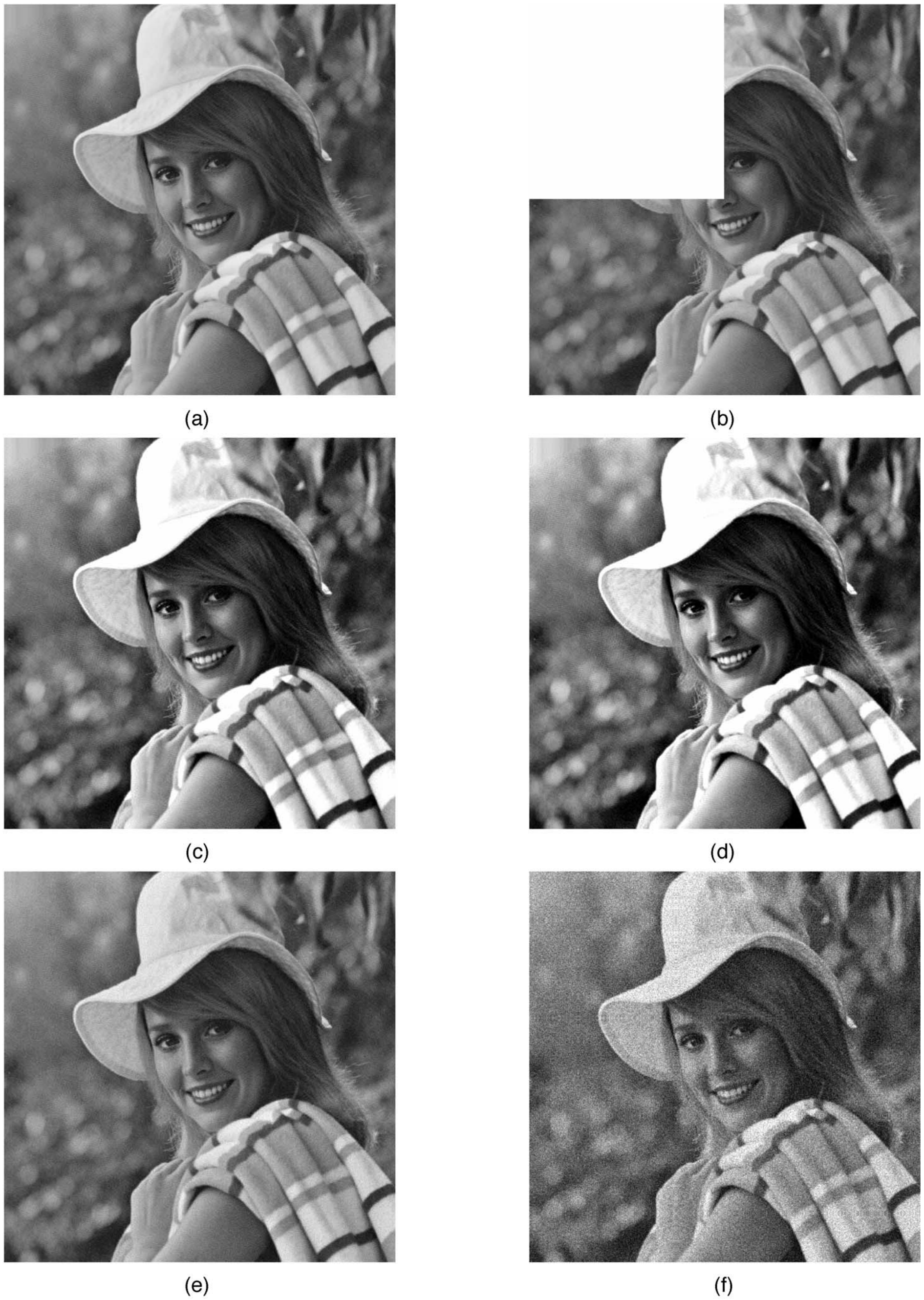


Fig. 18 The resulting watermarked images attacked by applying various image-processing operations: (a) original image (512×512), (b) cropping (1/4), (c) contrast enhancing (20%), (d) contrast enhancing (40%), (e) noise addition (5%), and (f) noise addition (20%).

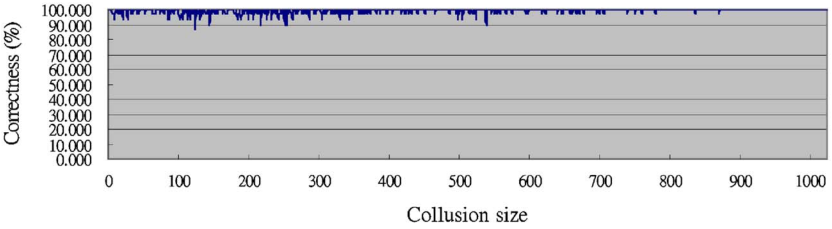


Fig. 19 The correctness of the traitor tracing for the experiments directly applied to the codewords.

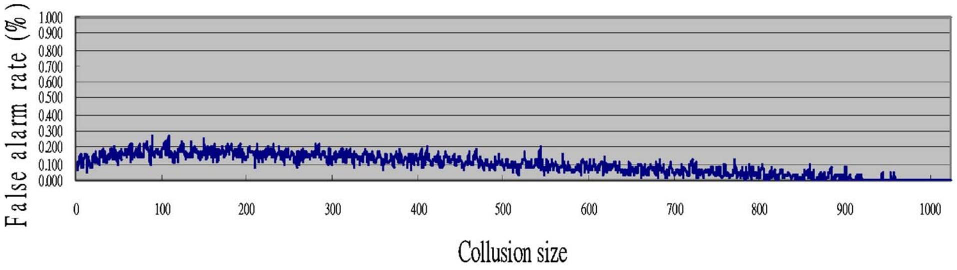


Fig. 20 The false-alarm rates of the traitor tracing for the experiments directly applied to the codewords.

which is repeatedly embedded in 24 DCT coefficients, determined by the embedding-position selection algorithm presented in Sec. 4.3, for each of the 8×8 DCT blocks except the first one (i.e., 24,552 DCT coefficients are modified in total). Without adaptive watermark-strength decision, the NC-PSNR point can be located at any point on the curve. Watermark strengths should be assigned according to the trade-off between NC and PSNR, but it is impractical to decide the strength after analyzing the curve; furthermore, global assignments of watermark strength on the whole image may ignore local characteristics of the image. The experiments show good results of our adaptive decision strategy, since the three points resulting from different sizes of image blocks all gather at the top right corner of the curve, which means the decision is close to the optimal solution. In addition, the performance is the best when the block size m is 64, which seems reasonable in that 128 is too coarse to localize the decision and 32 is too small to assign the strength precisely for humans while training.

Figure 11 illustrates the watermarks produced by fixed-strength watermarking (FSW) and adaptive strength-decision watermarking (ASDW), and Fig. 12 shows the difference between watermarks of FSW and ASDW. In Fig. 12(a), we find the difference between watermarks dispersed over the whole image, while in Fig. 12(b) and 12(c), the difference is obviously larger in the high-activity (darker) areas. This can be explained by observing the distribution of DCT coefficients. The ac coefficients are larger in the high-activity blocks than in low-activity ones; therefore, the corresponding watermark values are larger, because they are obtained from the ac coefficients multiplied by the watermark strength. In dark blocks, the dc coefficients are smaller, so slight changes in ac values may cause an obvious effect in the spatial domain.

4.3 *Selection Algorithm for Positions of Embedding*

Collusion attack is the main menace to fingerprinting schemes. Besides the fingerprint-code design, watermarking strategies are also important for collusion resistance. In this subsection, we present the algorithm to select embedding positions (namely, the DCT coefficients for embedding fingerprints) such that the fingerprinting scheme will have good performance in collusion resistance.

One of the goals of a fingerprinting scheme is to reduce the possible effect of the information loss through traitors. Such *collusion resistance* can be achieved by designing the watermarking method properly. As mentioned in Sec. 1, randomness and cryptography are two commonly used techniques for collusion resistance. Instead of applying these two techniques, we propose an optimization algorithm. Our algorithm is based on the maximization of an energy function that indicates the collusion resistance. The algorithm can be described as follows:

1. Choose applicable candidates, CP, for embedding positions, the embedding number n ($n<|\mathrm{CP}|$), and the resilience size c for fingerprint decoding and attacking simulations.
2. Initialize the embedding positions $\mathrm{EP}^0=\{\mathrm{ep}_{01},\mathrm{ep}_{02},\dots,\mathrm{ep}_{0n}\}$, $\mathrm{ep}_{0i}\in\mathrm{CP}\ \forall i\in\{1,\dots,n\}$, and the energy function $E^0=f(R^0_{\mathrm{hit}},R^0_{\mathrm{false}},Q^0)$, where R^t_{hit} is the hit ratio, R^t_{false} denotes the false-alarm rate, and Q^t is the imperceptibility of the watermark in the t'th iteration (now the set of embedding positions is EP^t). The values o R^t_{hit} *and* R^t_{false} are computed in a series of collusion-attacking simulations, while Q^t is estimated by watermarking experiments.
3. For $t=1,2,\dots,t_{\max}$:

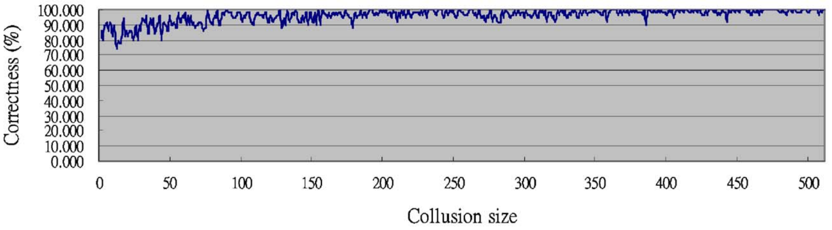


Fig. 21 The correctness of the traitor tracing for the experiments applied to the 256×256 "Lena" image.

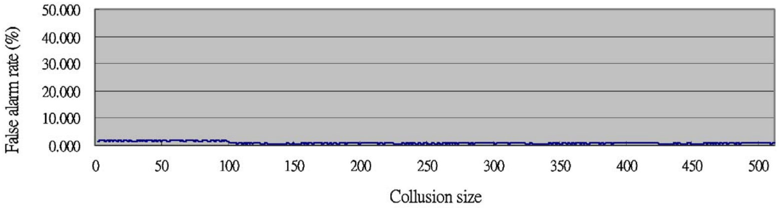


Fig. 22 The false-alarm rates of the traitor tracing for the experiments applied to the 256 × 256 "Lena" image.

```
    (1) Randomly reselect the set of embedding positions $EP^t$ from CP.
    (2) Compute $R^t_{hit}$, $R^t_{false}$ and $Q^t$.
    (3) Calculate $E^t = f(R^t_{hit}, R^t_{false}, Q^t)$.
    (4) If $E^t < E^{t-1}$ then
            $R^t_{hit} \leftarrow R^{t-1}_{hit}; R^t_{false} \leftarrow R^{t-1}_{false}; Q^t \leftarrow Q^{t-1}; EP^t \leftarrow EP^{t-1}$.
    (5) If $|E^t - E^{t-1}| \leq \varepsilon$ then
    $T = t$; go to 5;
    next $t$.
4.      $T = t_{max}$.
5. Output $EP^T$.
```

Our goal is to optimize the function value E^T, and we define the energy function as $E^t = \alpha R^t_{\text{hit}} - \beta R^t_{\text{false}} + \gamma \text{PSNR}^t$, that is, we use PSNR serve as to the estimate of watermark imperceptibility and to limit the distortion of image watermarking. In our experiments, CP is set as shown in Fig. 13(a), α is 0.2, β is 0.8, γ is 0.001, the collusion size is 2, and 50 collusion attacks (in which colluder pairs are picked randomly) are performed to test the hit ratios and false-alarm rates for traitor detection. Increasing hit rates will also cause a rise of false-alarm rates and distortion of image quality, so the trade-offs among $R^t_{\text{hit}}, R^t_{\text{false}}$, and Q^t must be considered when setting these parameters. One example of EP^t is given in Fig. 13(b). After iterations of the proposed optimization procedure, the embedding positions are selected to maximize the energy function approximately. We test the performance of the algorithm for selecting embedding positions by computing the hit ratio and the false-alarm rate. The test image is of size 256 × 256, the fingerprint code is the hierarchical code mentioned in Sec. 3, and the watermarking strategy is illustrated in Sec. 4.2. Figure 14 illustrates the optimization process of our algorithm. Obviously, with the execution of iterations, the hit ratio increases, the false-alarm rate decreases, and PSNR becomes progressively larger.

5 Experimental Results

In the experiments, 256 × 256 and 512 × 512 images are used as the test data for our fingerprinting system. The fingerprints are constructed as described in Sec. 3, and the watermarking procedure is described in Sec. 4. That is to say, a 4320-bit watermark is periodically embedded in 24 coefficients for each of the 8 × 8 DCT blocks except the first one. This embedding ratio (24/64) is much larger than (1000/256 × 256) in Ref. 21 and that in several other investigations,[25,26] but our method still provides watermark transparency without losing robustness. The imperceptibility (Sec. 5.1) and robustness (Sec. 5.2) of the watermarking methodology were examined in these experiments, although our ultimate goal is to achieve high performance in traitor tracing. In experiments for traitor detection, collusion attacks are performed on both codewords and fingerprinted images, and the proposed tracing algorithm is applied to trace as many colluders as possible.

5.1 *Imperceptibility*

The 1024 codewords for the 1024 users are embedded into the host image to test the visibility of our system. If the image size is 512 × 512, the average PSNR of embedded images is 44.21 dB, which is almost imperceptible to humans. But if the image size is 256 × 256, the average PSNR reduces to 32.49 dB, because we have to embed many more bits in an 8 × 8 DCT block. Figure 15 demonstrates the superiority of the adaptive watermark-strength decision in contrast to fixed-strength watermarking. As described in Sec. 4, watermark strengths will adapt to JNDs when ASDW is adopted; thus the transparency is really improved, as shown in Fig. 15. Figure 16 shows the original

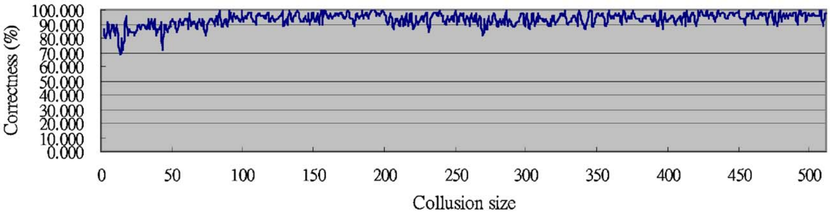


Fig. 23 The correctness of the traitor tracing for the experiments applied to the 256 × 256 "Lena" image compressed by JPEG (quality factor 80).

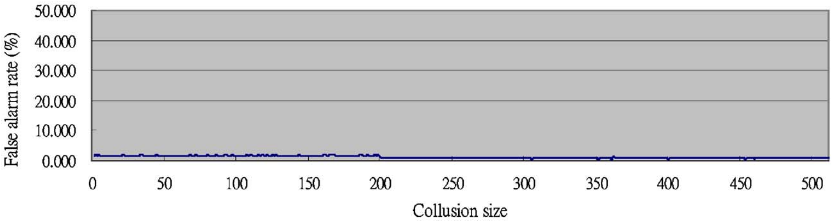


Fig. 24 The false-alarm rates of the traitor tracing for the experiments applied to the 256 × 256 "Lena" image compressed by JPEG (quality factor 80).

image and one instance of the fingerprinted images (where the image size is 256 × 256), and Fig. 17 gives another example for the case of image size 512 × 512. The watermarks in both of the two fingerprinted images are visually imperceptible.

5.2 *Robustness*

The size 512 × 512 "Elaine" image [see Fig. 17(a)] is used to test the robustness. Tables 1–3 show the NC values of extracted fingerprints after applying several different types of image processing to fingerprinted images, and Fig. 18 shows the resulting images after attacks. For comparing our algorithm with the image-adaptive watermarking in the DCT domain (IA-DCT) with visual models proposed in Ref. 14, the same compression and cropping attacks were performed.

In Table 1, robustness to JPEG compression is examined. For the ASDW there are two rows: values in the first row are the NC values when erasure bits are skipped, and in the second row are the actual NC values (so also in Table 2 and Table 3). It is observed that ASDW does not yield so good performance as IA-DCT if the quality factor Q is large, but for lower Q, ASDW performs better, especially when Q is 5. This is because ASDW uses the relation between DCT coefficients in adjacent blocks to embed watermarks, rather than embedding an absolute value in the DCT coefficients; moreover, ASDW has taken the quality factor into account in the embedding formula.

Table 2 shows the results after cropping the original image followed by compression, which yields similar results to those in Table 1. In Table 3, we find that ASDW can resist the contrast enhancement well and has a certain resistance to the noise-addition attack.

However, IA-DCT is a pure *watermarking* scheme, while our system is a *fingerprinting* scheme. As a result, the adjustments of parameters may have different purposes, and the algorithms may yield different trade-offs among speed, robustness, and imperceptibility. That is, the resistance of the proposed algorithm to these attacks is good enough because it is impossible to attack contents severely in the scenario of the practical fingerprinting system. In fact, experiments (detailed in Sec. 5.3) show that the collusion resiliency is still acceptable for images attacked by JPEG compression, which occurs more often than other attacks.

Moreover, the transparency, the collusion resilience, and the embedding time for large customer bases are also quite important. In Sec. 4.3, the maximization of the energy function E take into account both image quality and collusion resistance. In addition, as mentioned in the preceding subsection, the proposed method has good imperceptibility for long-length fingerprints. The authors of Ref. 14 did not give quantified experimental results for the transparency of their watermarking strategy, but the PSNR (44.21 dB) for 512 × 512 images of our method makes it almost imperceptible to humans.

5.3 *Correctness of Traitor Tracing*

We have two experiments for traitor detection: one is a test directly on codewords, and the other on images. Fifty coalitions are chosen randomly for each of the 511 collusion sizes (from 2 to 512); we then use averaging attacks to produce the pirate codes or images. In the experiments on images, averaging attacks are performed in the spatial domain.

Figures 19–22 illustrate the correctness and the false-alarm rates of the traitor tracing; here the correctness is defined as the probability of catching at least one of the colluders correctly, and the false-alarm rate as the probability of pronouncing an innocent user guilty. In the first experiment, rather promising results are obtained. Almost all of the correctnesses are larger than 90%, and the false-alarm rates are less than 0.3%, even if the collusion size is large. In the second experiment, even though the performance is not as good as that in the first one, we can refine it by adaptively adjusting the thresholds when estimating the Hamming distances in the list decoding for the segment code, or improving the watermarking approach. Moreover, replacing the 4-secure code (i.e., the segment code) by a b-secure or b-TA code with larger b will improve the detection correctness, because the accumulated error is reduced (but at the cost of increasing the length of the segment code). In the realistic application, the branch size can be variable and different for every hierarchy. Figures 23 and 24 show that even when the images are compressed by JPEG with quality factor 80 (i.e., the images are attacked by both collusion and JPEG compression), the collusion resiliency is still acceptable, in contrast with that of Refs. 10 and 11.

In conclusion, the proposed hierarchical fingerprinting system has good collusion resiliency even when the collusion size is near n. The most important thing is that the code length of the naive encoding method, without using the tree structure, is 31,228,440,620, which is much longer than the 4320 required in the proposed approach. And the collusion resiliency c increases from 2 to close to 1024.

5.4 *Computation Time Analysis*

Although the adaptive watermarking scheme requires a training and a position-selection procedure, it should be noticed that both of these two time-consuming steps only take place once. After determining the embedding strengths and positions by these steps, they are passed as parameters to the embedding or detection procedures. In the experiment for image fingerprinting as mentioned previously, both the watermark embedding and the extraction step take 0.010 s/image on a 1.8-GHz Pentium 4 PC. The proposed traitor-tracing algorithm for the hierarchical scheme takes 1.812 s to trace all possible colluders if the recursive function is not optimized by iterative codes, whereas the brute force search for the code tree takes 38.034 s, which is much longer than the proposed method.

6 Conclusions

We have introduced a traceable content-adaptive fingerprinting system for images. A hierarchical fingerprint code that can be constructed systematically and flexibly for multimedia is adopted. Most important of all, the produced code is much shorter than the naive code without the hierarchical construction while preserving high collusion resiliency. Combining the proposed content-adaptive watermarking scheme with this code can achieve high image quality of fingerprinted images and good performance of traitor tracing while serving large numbers of users. This fingerprinting scheme, with good performance, has been implemented with practical parameters, including applicable resilience sizes for large customer bases. Moreover, both the watermarking and the traitor-tracing steps are computationally efficient. It is our belief that the work done in this paper provides a feasible solution for multimedia fingerprinting and traitor tracing.

Acknowledgment

This work was partially supported by the National Science Council and the Ministry of Education of ROC under contract Nos. NSC92-2622-E-002-002, NSC92-2213-E-002-023, 89E-FA06-2-4-8, and NSC93-2752-E-002-006-PAE. We would also like to thank Dr. Chun-Hsiang Huang for his helpful discussions and valuable suggestions.

References

1. D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," in *Proc. CRYPTO'95*, pp. 452–465, IACR (1995).
2. B. Chor, A. Fiat, and M. Naor, "Tracing traitors", in *CRYPTO'94*, Vol. **839**, pp. 480–491, IACR (1994).
3. A. Fiat and M. Naor, "Broadcast encryption", in *CRYPTO'93*, pp. 480–491, IACR (1993).
4. D. R. Stinson and T. v. Trung, "Some new results on key distribution patterns and broadcast encryption," *Designs, Codes, Cryptogr.* **14**, 261–279 (1998).
5. D. R. Stinson, "On some methods for unconditionally secure key distribution and broadcast encryption," *Designs, Codes, Cryptogr.* **12**, 215–243 (1997).
6. D. R. Stinson and R. Wei, "Combinatorial properties and constructions of traceability schemes and frameproof codes," *SIAM J. Discrete Math.* **11**, 41–53 (1998).
7. J. Dittmann, P. Schmitt, E. Saar, and J. Ueberberg, "Combining digital watermarks and collusion secure fingerprints for digital images," *J. Electron. Imaging* **9**(4), 456–467 (2000).
8. W. Trappe, M. Wu, J. Wang, and K. J. R. Liu, "Anti-collusion fingerprinting for multimedia," *Signal Process.* **51**, 1069–1087 (2003).
9. http://www.its.caltech.edu/~ssachan/math6b/design.pdf
10. H. Zhao and K. J. Liu, "Fingerprint multicast in secure video streaming," *IEEE Trans. Image Process.* **15**(1) 12–29 (2006).
11. Z. J. Wang, M. Wu, W. Trappe, and K. J. R. Liu, "Group-oriented fingerprinting for multimedia forensics," in *Proc. 17th Int. EURASIP Conf.*, pp. 2153–2173 (2004).
12. A. Silverberg, J. Staddon, and J. L. Walker, "Applications of list decoding to tracing traitors," *IEEE Trans. Inf. Theory* **49**, 1312–1318 (2003).
13. M. S. Kankanhalli and R. K. R. Ramakrishnan, "Content based watermarking of images," in *ACM Multimedia'98* 61–70 (1998).
14. C. I. Podilchuk and W. Zeng, "Image-adaptive watermarking using visual models," *IEEE J. Sel. Areas Commun.* **16**(4), 525–539 (1998).
15. M. D. Swanson, B. Zhu, and A. H. Tewfik, "Multiresolution scene-based video watermarking using perceptual models," *IEEE J. Sel. Areas Commun.* **16**, 540–550 (1998).
16. K. J. Davis and K. Najarian, "Maximizing strength of digital watermarks using neural networks," in *IEEE Int. Conf. on. Neural Networks*, , Vol. **4**, No. 15–19, pp. 2893–2898 (2001).
17. Z. M. Zilang, R. Y. Li, and L. Wang, "Adaptive watermark scheme with RBF neural networks," in *IEEE Int. Conf. on. Neural Networks & Signal Processing*, pp. 1517–1520 (2003).
18. D. Kundur and K. Karthik, "Video fingerprinting and encryption principles for digital rights management," *Proc. IEEE* **92**(6), 918–932 (2004).
19. C. H. Huang and J. L. Wu, "A watermark optimization technique based on genetic algorithms," in *Proc. SPIE* **3971**, 516–523 (2000).
20. A. Wool, "Key management for encrypted broadcast," *ACM Trans. Inf. Syst. secur.* **3**(2), 107–134 (2002).
21. I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Trans. Image Process.* **6**, 1673–1687 (1997).
22. R. Safavi-Naini and Y. Wang, "A code for sequential traitor tracing," in *ASPCS*, , pp. 211–224 (2002).
23. T. Kohonen, "The self-organizing map," *Proc. IEEE* **78**(9), 1464–1480 (1990).
24. J. A. Freeman and D. M. Skapura, *Neural Networks: Algorithms, Applications, and Programming Techniques*, p. 267, Addison-Wesley (1992).
25. C. T. Hsu and J. L. Wu, "Hidden digital watermarks in images," *IEEE Trans. Image Process.* **8**, 58–68 (1999).
26. Q. Cheng and T. S. Huang, "An additive approach to transform-domain information hiding and optimum detection structure," *IEEE Trans. Multimedia* **3**, 273–284 (2001).

Yu-Tzu Lin received BS and MS degrees in information and computer education from National Taiwan Normal University, Taipei, Taiwan, R.O.C., in 1994 and 1997. She is currently pursuing her PhD degree from the Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. Her research interests include copyright protection for multimedia, pattern recognition, and image processing.

Ja-Ling Wu received his BS degree in electronic engineering from TamKang University, Tamshoei, Taiwan, R.O.C., in 1979, and MS and PhD degrees in electrical engineering from Tatung Institute of Technology, Taipei, Taiwan, in 1981 and 1986. Since 1987, he has been with the Department of Computer Science and Information Engineering, National Taiwan University, where he is presently a professor. He has published more than 200 journal and conference papers. His research interests include algorithm design for digital signal processing, data compression, digital watermarking, and multimedia systems. Prof. Wu was the recipient of the Excellent Research Award from NSC, Taiwan, in 1999, 2001, and 2004. He was elected a lifetime distinguished professor, National Taiwan University, in November 2006.