A Watermarking Algorithm for Fingerprinting Intelligence Images

Yiwei Wang, John F. Doherty
Department of Electrical Engineering
The Pennsylvania State University
University Park, PA 16802
e-mail: {yxw131,jfdoherty}@psu.edu

Robert E. Van Dyck
National Institute of Standards and Technology
Gaithersburg, MD 20899
e-mail: vandyck@antd.nist.gov

Abstract — **The rapid growth of digital multimedia technologies brings tremendous attention to the field of digital watermarking. One application for digital watermarking is fingerprinting. The owner or the distributor of the multimedia data can insert unique watermarks into copies for different customers or receivers, which will be helpful to identify the source of illegal copies. Fingerprinting can be used in an intelligence agency to trace the source of the leak. In this paper, we first identify the requirements for fingerprinting algorithms of intelligence images. Then, we present an algorithm that meets the requirements.**

Keywords: watermark, fingerprint

I. Introduction

Because of the advances in technology, more and more multimedia data are now stored and transmitted in digital forms. This inspires a lot of research efforts in digital watermarking. One possible application of digital watermarking is fingerprinting [1, 2]. To fingerprint the data, unique information is inserted into each copy. This will enable the owner or the distributor to trace an unauthorized copy back to the source. Image fingerprinting can be used to trace the source of the leak of intelligence images. In this paper, we first discuss the requirements of algorithms designed for such application and then propose a wavelet-based algorithm to meet the requirements.

The rest of the paper is organized as follows: In Section II, we discuss the requirements of algorithms for fingerprinting intelligence images. In Section III, we review some necessary background on wavelets. The algorithm that meets the requirements is presented in Section IV, and experimental results are given in Section V to demonstrate its effectiveness. In Section VI, we present the conclusion.

II. Requirements for fingerprinting intelligence images

Various algorithms have been proposed to fingerprint digital data and images [3, 4, 5]. However, like many engineering problems, fingerprinting is very application-oriented. Before presenting our algorithm, we want to discuss the requirements for fingerprinting intelligence images.

The goal of our algorithm is to trace and punish the source of the leak. Most fingerprinting schemes in the literature are symmetric, which means both the user and the distributor have the access to the fingerprinted data. If an unauthorized copy of such data is found, one cannot assign responsibility to one of them with absolute certainty [6]. Although symmetric fingerprinting algorithms are very helpful for eliminating possible sources, their findings alone cannot convict anybody in the court. The intelligence images are usually very valuable. One may be willing to risk his job to sell a couple of images for a large amount of money. Hence, to prevent espionage, the criminal justice system must get involved. Because the symmetric fingerprinting algorithms cannot provide enough evidence for a conviction, we need to use an asymmetric fingerprinting algorithm.

Generally speaking, the distribution structure of intelligence images is a *tree*, as shown in Fig. 1. After an intelligence image is obtained, knowing who the end users of the image are and how many copies are needed is unlikely at that point. However, based on experience, it should be easy to decide which department should have a copy. As for the departments, it should be easy to decide the destination divisions of the image. Therefore, by following the tree structure, the image will reach the end users after several levels of distribution. At each distribution point illustrated by rectangles in Fig. 1, fingerprints should be embedded into the data to indicate the destination point at the next level. Therefore, the algorithm should have the ability of multi-level fingerprinting.

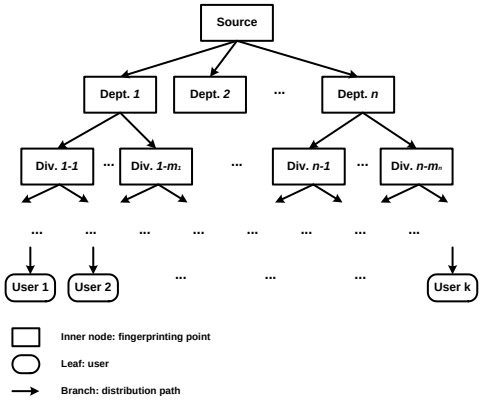


Fig. 1: distribution structure

By examining the distribution tree, we can see that only the users need to keep the images for a long period of time. As for the distribution points, after the fingerprinting process is completed, there is no need to keep the unfingerprinted image any more. Since the fingerprinting process is performed at a certain location during a short period of time, it is feasible to have the process monitored and destroy the unfingerprinted image immediately afterwards. This will make the fingerprinting process asymmetric because the distributor can no longer generate a fingerprinted image and the receivers get the only copies. If the pattern of the fingerprint is not randomly assigned by the distributor but is brought in by the receiver, it will be impossible for the distributor to frame a receiver.

Originally, there are three kinds of possible leak sources, the fingerprinting personnel illustrated by rectangles in Fig. 1, the users illustrated by rounded rectangles in Fig. 1, and the delivery personnel illustrated by arrows in Fig. 1. Since we put restrictions on the fingerprinting process, we can eliminate the fingerprinting personnel as a possible leak source. To fingerprint an image, at each level, a distributor asks each of its receivers to submit a fingerprint pattern, registers the patterns, performs the fingerprinting process, destroys the unfingerprinted copy, and distributes the fingerprinted ones. If an unauthorized copy is later found, by extracting and matching the fingerprints, we can go down the distribution tree to find the suspect. If a fingerprinting point (e.g., Dept. *1*) detects the fingerprint of a

receiver (e.g., Div. *1-1*), but the receiver does not find a fingerprint matching the ones that it has on file, then we can conclude that the delivery person between those two points must have made a detour and is the suspect. If at the fingerprinting point of each level, a matching fingerprint is found, then we will reach one of the users at the bottom of the tree. The suspect is then either that user or the person who delivers the image to him. If we require that the user retrieves his own image, there is no ambiguity any more, and finding an unauthorized copy will guarantee the conviction of the person who made it.

The above fingerprinting system is asymmetric. However, it comes at a price. Since we destroy all unfingerprinted images to make it asymmetric, we cannot use them as reference in the detection process as a lot of symmetric algorithms in the literature do [4, 5]. Moreover, the algorithm must have the ability to carry multiple fingerprints. In addition, because the receiver knows the fingerprint pattern, the algorithm should provide enough randomness so that the receiver cannot remove the fingerprint.

Furthermore, we also have to address the problems that every fingerprinting/watermarking algorithm faces. First, the false positive probability must be extremely small. In other words, if there is a probability that a fingerprint can be detected in an unfingerprinted image, it must be very small. Second, the fingerprinted image must be perceptually the same as the original image. This will limit the energy of the fingerprint. Third, the fingerprinting algorithm must be robust. The fingerprint must survive signal processing and hostile attacks. However, in our application, the intelligence images, such as satellite photos, usually require high resolution. This will limit the possible distortion introduced to the image. In addition, it is unlikely that there are a lot of leak sources. Therefore, the problem of collusion is easier to address compared to some other applications.

In summary, the algorithm should

- be able to detect the fingerprint without the original image
- be able to carry multiple fingerprints
- prevent the user from removing the fingerprint
- have extremely small false positive probability
- achieve perceptual invisibility
- achieve some robustness

III. Background on wavelets

Because our algorithm is based on wavelets, we review some necessary background on wavelets before introducing the algorithm.

The extensive study in the past two decades has made the wavelet transform an important tool in signal analysis. In digital signal and image processing, the discrete wavelet transform is closely related to filter banks. A typical 2-channel decomposition and reconstruction structure is given in Fig. 2.

As Vetterli and Kovacevic stated in [7], the requirements for perfect reconstruction (i.e., $x = \widehat{x}$ in Fig. 2) are:

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 2;$$

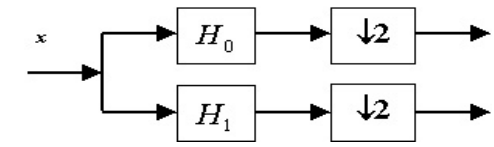
$$H_0(-z)G_0(z) + H_1(-z)G_1(z) = 0.$$

A very important class of filter banks are orthonormal filter banks. For orthonormal filter banks, the requirements become:

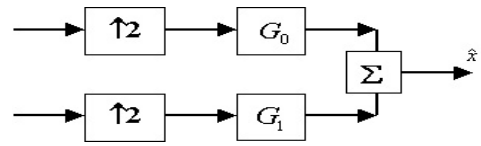
$$G_0(z)G_0(z^{-1}) + G_0(-z)G_0(-z^{-1}) = 2;$$

$$G_1(z) = -z^{-2k+1}G_0(-z^{-1}), k \in Z;$$

$$H_i(z) = G_i(z^{-1}), i \in \{0, 1\}.$$



(a) analysis part



(b) synthesis part

Fig. 2: 2-channel decomposition and reconstruction structure

Furthermore, if we define $P(z) = G_0(z)G_0(z^{-1})$, then

$$P(z) = 1 + \sum_{k\ odd} a_k z^{-k}, a_k = a_{-k}.$$

On the other hand, if we have a polynomial

$$P'(z) = 1 + \sum_{k\ odd} a'_k z^{-k}, a'_k = a'_{-k};$$

where $\sum a'_k = 1$, $a'_k \geq 0$; because $P'(z) \geq 0$, $\forall z \in \{z : |z| = 1\}$, $P'(z)$ is spectrum factorizable, i.e., $\exists\ G'_0(z)$, s.t. $P'(z) = G'_0(z)G'_0(z^{-1})$ and $G'_0(z)$ is the lowpass synthesis filter of a 2-channel orthonormal FIR real-coefficient filter bank. Therefore, we can randomly generate orthonormal filter banks by randomly generating $P'(z)$'s.

In addition, we can control the behavior of the filter bank by adding certain constraints to the a'_k's. For instance, if we pick the filter length to be six, then

$$P(z) = 1 + a_1(z + z^{-1}) + a_3(z^3 + z^{-3}) + a_5(z^5 + z^{-5}).$$

If we force $a_1 \leq 0.2$, then the filters will have large sidelobes. For example, if $P(z) = 1 + 0.1900(z + z^{-1}) + 0.0716(z^3 + z^{-3}) + 0.2383(z^5 + z^{-5})$, then the lowpass analysis filter is $H_0(z) = 0.2663 - 0.0897z^{-1} + 0.1395z^{-2} - 0.0983z^{-3} + 0.3014z^{-4} + 0.8951z^{-5}$ and the highpass analysis filter is $H_1(z) = -0.8951 + 0.3014z^{-1} + 0.0983z^{-2} + 0.1395z^{-3} + 0.0897z^{-4} + 0.2663z^{-5}$. As shown in Fig. 3, they will have large sidelobes [8].

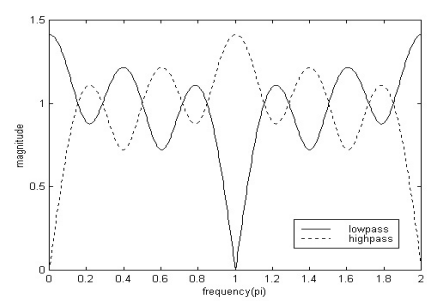


Fig. 3: an example of filters

IV. Algorithm

For the fingerprinting process, each receiver sends his fingerprinting information in the form of a watermark, which can be considered as a random vector, to his distributor. The distributor stores that watermark. Then, the distributor generates several sets of random orthonormal filter banks using the algorithm discussed in the previous

section. These filter banks are used to decompose the unfingerprinted image into different subbands and will be stored by the distributor as well. Several middle frequency subbands are selected to embed the watermark. Fig. 4 shows an example of selected subbands. A different set of filter banks is used for each decomposition. For instance, ten sets of filter banks will be used for the example illustrated by Fig. 4.

To introduce more randomness into the embedding process, we scramble the watermark using the information of the filters. If we denote the watermark by an n-tap vector $\overrightarrow{v_1} = \{v_1(1), v_1(2), \ldots, v_1(n)\}^T$ and create another m-tap vector $\overrightarrow{s} = \{s(1), s(2), \ldots, s(m)\}^T$ by putting the first digits after the decimal point of the first coefficients of each analysis filter together, then we can generate a vector $\overrightarrow{v_2}$ by doing the following. Starting from the first element in $\overrightarrow{v_1}$, we count $s(1)$ elements and assign the resulting element as the first element of $\overrightarrow{v_2}$, i.e., $v_2(1) = v_1(s(1)+1)$. Then, we remove the element that we just processed from $\overrightarrow{v_1}$ and count $s(2)$ elements from that point, then assign this element as the second element of $\overrightarrow{v_2}$. If we consider $\overrightarrow{v_1}$ and $\overrightarrow{s}$ to be circular vectors and repeat the above procedure, then after n steps, $\overrightarrow{v_2}$ will be a rearranged vector of $\overrightarrow{v_1}$. The rearranging process is reversible assuming we always have access to $\overrightarrow{s}$. When the fingerprint pattern is submitted by the receiver and the filter banks are generated by the distributor, the above rearranging process would make the embedded fingerprint contains information from both parties.

After decomposing the image and rearranging the watermark, we can embed the watermark into the image. First, the rearranged watermark is scaled by a factor to adjust the energy and then added to the wavelet coefficients of the selected subbands according to a predetermined scan order. Second, the synthesis filters are used to reconstructed the fingerprinted image from the modified coefficients. In addition, we perform rounding and clipping on the result image to make sure that it has the same gray-scale range as the unfingerprinted image. The block diagram of the embedding process is given in Fig. 5.

To determine whether a fingerprint is present in a suspect image, the following procedure is used. First, the stored filter banks are used to decompose the suspect image and the coefficients of the selected subbands are arranged to create a test vector. Then, the correlation between this test vector and the rearranged watermark is computed, which is equivalent to descrambling the test vector and calculating the correlation between the result and the original watermark. If the correlation is larger than a preset threshold, then the fingerprint is claimed to be found. The block diagram of the detection process is given in Fig. 6.

The threshold T in the detection process is an important parameter. It contributes to the false positive probability. Therefore, we need to evaluate the false positive probability as a function of T.

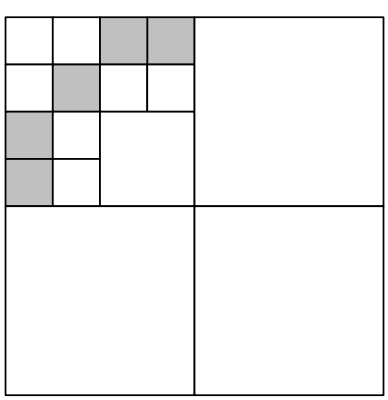


Fig. 4: an example of subbands (dark squares indicate the subbands selected to embed the watermark)

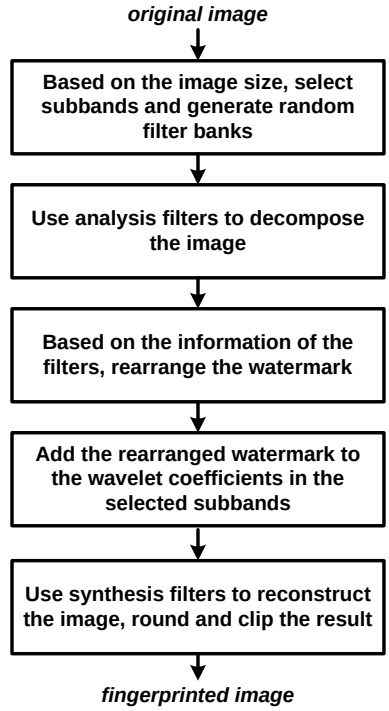


Fig. 5: block diagram of the embedding process

If we denote the wavelet coefficients by S and the embedded watermark by W, both of which are vectors of dimension n, then the correlation is $\alpha = \frac{S \cdot W}{\sqrt{(S \cdot S)(W \cdot W)}}$. $\frac{S}{\sqrt{S \cdot S}}$ and $\frac{W}{\sqrt{W \cdot W}}$ are two unit vectors of dimension n. So, the variable α is the inner product of two vectors on a unit sphere, and the original data that produce an inner product with $\frac{W}{\sqrt{W \cdot W}}$ that is larger than T will create a false positive. Without loss of generality, we can rotate $\frac{W}{\sqrt{W \cdot W}}$ to $[1, 0, 0, \ldots, 0]^T$. Assuming all possible S vectors are uniformly distributed on the sphere, the false positive probability is then a fraction between two areas, $P_f = \frac{A_1}{A}$, where A_1 contains all vectors on the sphere whose first element is greater than T and A is the area of the whole sphere. The surface area of a n-dimensional sphere of radius

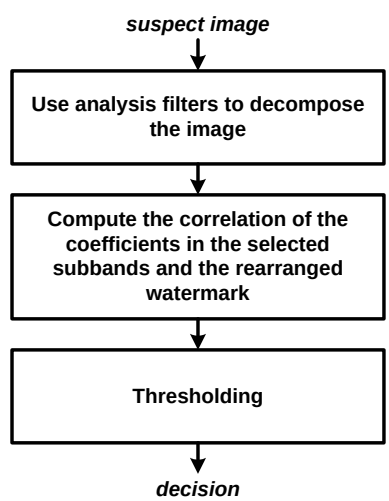


Fig. 6: block diagram of the detection process

ρ is $A_{sph} = nV_n\rho^{n-1}$, where $V_n = \frac{\pi^{n/2}}{(n/2)!}$ [9]. Therefore,

$$A_1 = \int_T^1 (n-1)V_{n-1}(\sqrt{1-x^2})^{n-2}\frac{dx}{\sqrt{1-x^2}},$$

and $A = nV_n$. So,

$$P_f = \frac{(n-1)V_{n-1}\int_T^1(\sqrt{1-x^2})^{n-3}dx}{nV_n}.$$

The exact value of the integral in the above equation is not easy to compute when the dimension n is large. However, we only interested in the order of the false positive probability. If we can develop upper and lower bounds for false positive probability as a function of T, we should have sufficient information to choose T, which is always positive and less than one. If $y = 1 - x^2$, we have

$$\int_T^1 (1-x^2)^{(m-3)/2}dx = \int_0^{1-T^2} \frac{y^{(m-3)/2}}{2\sqrt{1-y}}dy,$$

This equation can be bounded as follows

$$\begin{aligned} & \frac{\int_0^{1-T^2} y^{\frac{m-3}{2}}dy}{2T} \\ \geq & \int_0^{1-T^2} \frac{y^{\frac{m-3}{2}}}{2\sqrt{1-y}}dy \\ \geq & \frac{\int_0^{1-T^2} y^{\frac{m-3}{2}}dy}{2}, \end{aligned}$$

and it is easy to see that

$$= \frac{\int_0^{1-T^2} y^{\frac{m-3}{2}}dy}{\frac{2y^{(m-1)/2}}{m-1}\Big|_0^{1-T^2}} = \frac{2(1-T^2)^{(m-1)/2}}{m-1}.$$

So, the upper bound for the false positive probability P_f is $\frac{V_{n-1}(1-T^2)^{\frac{n-1}{2}}}{nV_nT}$ and the lower bound is $\frac{V_{n-1}(1-T^2)^{\frac{(n-1)}{2}}}{nV_n}$. Therefore, we can choose T to guarantee that the false positive probability is extremely small.

V. Experimental results

The proposed algorithm is tested on the "lion" image, which is an 8-bit gray-scale image of size 512 by 512. The depth of the distribution tree is four. In other words, four watermarks are added to the image. The subbands illustrated in Fig. 4 are selected to embed the fingerprints and the length of the watermark is $64^2 \times 5 = 20480$. The length for the orthonormal filter banks is six. Each element of the watermark is a random value uniformly distributed from -1 to 1. The threshold T in the detection process is 0.03, and the corresponding upper and lower bounds are 9.21×10^{-6} and 2.76×10^{-7}, respectively.

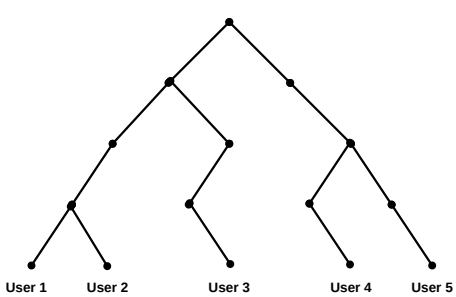
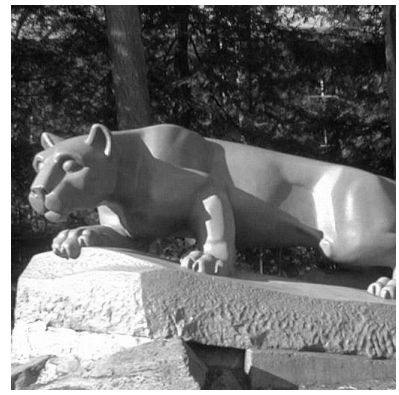


Fig. 7: an example of a six-user distribution tree

(a) the original image



(b) the fingerprinted image of user 1

Fig. 8: the original image and a fingerprinted image

We simulate a situation where there are five users. Their relationship is shown in Fig. 7. The original image and the fingerprinted image of the first user are given in Fig. 8. By comparing the two images, we can conclude that the fingerprinting algorithm achieves perceptual invisibility. The Peak-to-peak Signal-to-Noise Ratio (PSNR) of the five fingerprinted images are listed in Table 1, which is compatible with existing watermark algorithms and further indicates the invisibility. The correlation values of correct paths are listed in Table 2. We can see that all the values are larger than the threshold 0.3; also, the correlation values between mismatched detected and embedded watermarks are from -0.0105 to 0.0106, which suggests that the fingerprints are detected correctly and that the algorithm is suitable for tracing the unauthorized copy back to the source.

Tab. 1: PSNR of fingerprinted images

User No.	PSNR (dB)
1	38.87
2	38.91
3	38.90
4	38.87
5	38.89

Then, we test the robustness of our algorithm. First, we add a zero mean Gussian noise with variance 100 to the fingerprinted images, which introduces a degradation of roughly 10 dB. The correlation values are listed in Table 3 and prove that our algorithm is robust to additive Gaussian noise.

Tab. 2: The correlation values on correct tracing paths of the fingerprinted images

User No.	lvl. 1	lvl. 2	lvl. 3	lvl.4
1	0.087	0.075	0.084	0.084
2	0.087	0.075	0.084	0.076
3	0.086	0.071	0.084	0.069
4	0.087	0.081	0.076	0.071
5	0.087	0.080	0.078	0.078

Second, we test our algorithm on histogram equalization since it is a common method to improve the visual quality of the images. The results are given in Table 4 and indicate the robustness of our algorithm to histogram equalization.

Tab. 3: The correlation values on correct tracing paths of the noise corrupted fingerprinted images

User No.	lvl. 1	lvl. 2	lvl. 3	lvl.4
1	0.085	0.074	0.082	0.082
2	0.085	0.073	0.082	0.072
3	0.084	0.068	0.079	0.066
4	0.086	0.078	0.072	0.070
5	0.085	0.077	0.074	0.074

Tab. 4: The correlation values on correct tracing paths of the histogram equalized fingerprinted images

User No.	lvl. 1	lvl. 2	lvl. 3	lvl.4
1	0.106	0.095	0.103	0.103
2	0.105	0.095	0.103	0.095
3	0.106	0.086	0.098	0.084
4	0.105	0.099	0.095	0.090
5	0.105	0.098	0.099	0.094

Furthermore, we test the robustness of our algorithm to compression. JPEG baseline compression is applied on the fingerprinted images at 1.3 bpp and the results are listed in Table 5. If the bitrate increases, the correlation values will also increase. The robustness of our algorithm to compression is not as good as some algorithms in the literature. However, intelligence images in general do not go through lossy compression, and our algorithm demonstrates some robustness when the compression is performed using a relatively high bit-rate.

Besides being robust to various signal processing procedures, a good fingerprinting algorithm should also survive hostile attacks. Although the watermarks in our algorithm are provided by the users, they cannot remove them because we scramble each one using the information derived from the filters, which the users do not know. However, they still can use collusion to attack the fingerprints. Fortunately, the number of images used in a collusion attack on an intelligence fingerprinting system is usually very small. Also, since our algorithm uses a long pseudo-noise code as a watermark, similar to the algorithm in [10], the algorithm has some built-in resilience to collusion attacks. We simulate two such attacks. One is forged by users 1 and 2, and the other is forged by users 4 and 5. In each case, the average of two images is used. Table 6 shows that our algorithm can still track the fingerprints correctly.

Tab. 5: The correlation values on correct tracing paths of the compressed fingerprinted images

User No.	lvl. 1	lvl. 2	lvl. 3	lvl.4
1	0.037	0.033	0.038	0.034
2	0.037	0.032	0.038	0.030
3	0.037	0.031	0.036	0.030
4	0.038	0.036	0.031	0.032
5	0.038	0.035	0.030	0.030

Tab. 6: The resilience to collusion

Collusion by usr. 1&2			Collusion by usr. 4&5		
Lvl. No.	usr 1.	usr. 2	Lvl. No.	usr. 4	usr. 5
1	0.087	0.087	1	0.087	0.087
2	0.075	0.075	2	0.080	0.080
3	0.084	0.084	3	0.035	0.037
4	0.042	0.034	4	0.036	0.040

VI. Conclusion

In this paper, we first discuss the requirements for a fingerprinting system for intelligence images. Then, we propose an algorithm to meet these requirements. Our experimental results demonstrate the effectiveness of the algorithm.

Although the proposed algorithm possesses appealing properties such as asymmetry and multi-level fingerprinting, its application is limited to intelligence images because of the robustness issues. For other applications, such as video fingerprinting, significant modifications are needed.

References

[1] I. Cox, M. Miller, and J. Bloom, "Watermarking applications and their properties", *Proc. Intl. Conf. Inf. Tech.: Coding & Computing 2000*, pp. 6-10, Mar. 2000.

[2] G. Langelaar, I. Setyawan, and R. Lagendijk, "Watermarking digital image and video data, a state-of-the-art overview", *IEEE Sig. Proc. Mag.*, vol. 17, no. 5, pp. 20-46, Sept. 2000.

[3] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data", *IEEE Trans. Inf. Theory*, vol. 44, no. 5, pp. 1897-1905, Sept. 1998.

[4] J. Dittmann, "Combining digital watermarks and collusion secure fingerprints for customer copy monitoring", *Proc. IEE Seminar Sec. Image & Image Auth. 2000*, pp. 6/1-6/6, Apr. 2000.

[5] J. Domingo-Ferrer and J. Herrera-Joancomarti, "Simple collusion-secure fingerprinting schemes for images", *Proc. Intl. Conf. Inf. Tech.: Coding & Computing 2000*, pp. 128-132, Mar. 2000.

[6] B. Pfitzmann and M. Schunter, "Asymmetric fingerprinting", *Proc. EUROCRYPT '96, LNCS 1070*, pp. 84-95, Springer-Verlag Berlin Heidelberg, 1996.

[7] M. Vetterli and J. Kovacevic, *Wavelets And Subband Coding*, Prentice-Hall, INC., 1995.

[8] Y. Wang, J. Doherty, and R. Van Dyck, "A novel wavelet-based algorithm for watermarking digital images", *Proc. Conference on Information Sciences and Systems*, pp. TA5.7-12, Mar. 2000.

[9] J. Conway and N. Sloane, *Sphere Packings, Lattices and Groups*, Springer-Verlag New York, 1999.

[10] J. Kilian, F. Leighton, L. Matheson, T. Shamoon, R. Tarjan, and F. Zane,"Resistance of digital watermarks to collusive attacks", *Proc. Intl. Symp. Info. Theory, 1998*, pp. 271, 1998.