A New Image Copyright Protection Algorithm Using Digital Signature Of Trading Message and Bar Code watermark

Ji-Hong Chang and Long-Wen Chang
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan, 300
lchang@cs.nthu.edu.tw

Abstract

A new digital watermarking algorithm for images with bar code and digital signature is proposed. Digital signature is used to prevent a buyer from distributing unauthorized image copies and is also used to prevent a seller from forging the sale transaction of image copies without a buyer. We encode both the buyer's and the seller's signature into a bar-code and embed the bar-code into the image for trading. Both signatures are used to verify the integrity of the trading messages. Also, a cryptography technique is used not only for choosing embedded positions but also for protecting the trading messages. The signed trading message that contains the buyer's name, the seller's name, the object name, the trading price and the trading date is embedded in four images to test our proposed copyright protection algorithm. Our simulation shows that the proposed algorithm gets satisfactory results for various attacks.

Keywords: digital watermarking, copyright protection, cryptography

1. INTRODUCTION

The cryptography and the digital watermark have begun to attract extensive attention from the industrial field along with the rapid growth of the Internet and electronic business [1-5]. Encryption and decryption [6-7] are used to provide protection for trade messages in the Internet, and digital signature is utilized for identification. Robust watermark can be seen as a mean for the declaration of copyright. Fragile watermark is used to check the integrity of the images.
In this paper, we show how digital signature protects copyrights of images traded in the Internet. Copyright protection is known to be a buyer-seller problem [8]. It must meet the three following requirements. First, when we find an unauthorized image, we should be able to detect where the original source comes from. Second, the message embedded in the image should include the buyer information. Finally, the signature should be accessed to anyone who verifies the validation with special information like a private key or an embedded position list.

Usually, a meaningful logo or a random sequence is usually chosen as a watermark and if the watermark was damaged from various attacks, it is corrected with error correcting code. A meaningful logo is a nice tool for the declaration of the copyright, but we may run into difficulty when one wants to hide a long message such as trading messages or digital keys. In our method, we encode a long message into a bar-code [9]. The maximal length of the message in our simulation is 64 bytes. It is long enough for the inclusion of the buyer's identification, the object name, the object number, the price, the timestamp, and so on. We find that the bar-code can sustain various kinds of attacks. .

In the simulation, we encrypt a trading message with 1024-bits RSA [6-7] and the encrypted trading message is verified by both buyer and seller. Then, the encrypted message is signed with the buyer's private key. The signed massage is encoded into a bar-code to be a watermark for embedding in the original image. The embedded position list is decided by SHA-1[6], a hash function, with the seller's private key. In the watermark detection, only the seller can extract the watermark because nobody else has his private key except the seller. However, after the signature is extracted everyone can verify the trading message with the buyer's public key.

2. THE BAR-CODE WATERMARKING SYSTEM

The proposed bar-code watermarking system is composed of three stages: the verifying procedure, the embedding procedure and the extraction procedure. The mechanism of the bar-code used in our simulation is similar to that of Code 39. We expand the number of lines to present each byte from nine to twelve. The twelve lines are composed of three bold black lines, three thin black lines, two bold white lines and four thin white lines. So, there are

300 combinations and they are enough to express all ASCII codes and the remaining 44 combinations are reserved. We set a black line as a start line so that it will end with a white one. The bold line must be presented in two pixels and the thin one in one pixel. That is, each byte will be presented in 17 pixels.
A bar-code records information that includes the buyer, the seller, the object for sale, the object price and so on. Because the bar-code is made to have error correcting capability, we can recover the damaged bar-code image with the following recovery rules so that the bar-code is robust against various attacks.

The five recovery rules of our bar-code mechanism are listed below:

1. According to the majority rule and the average gray level, we judge the line is black or white.
2. By the presupposition, the start line representing each byte is a black one and its ending (17TH) line is a white one.
3. When we judge the line is black or white in step 1, we record the error pixels of each line and the number of error pixels can be viewed as the severity of attack.
4. If one line and its neighboring line are of the same color, we view these two lines as a bold line. We can check if there are three bold black lines, three thin black lines, two bold white lines and four thin white lines for each byte. If not, we change the color of the line that has the most error pixels until the number of bold and thin lines for each byte is correct.
5. We group every twelve lines as a set for the bar code and map a set of 12 lines to an ASCII code. If we can not map it to any ASCII code, we mark this byte to be a '?' to indicate an error.

Suppose the buyer B wants to buy an object from the seller A. He makes a trading message M which includes the buyer's name, the seller's name, the kinds of objects, the trade price and the trading date. The message is signed with B's private key and the buyer B encrypts the signature with A's public key to avoid the message is known to the third party. Then he sends the encrypted trading message to the seller A:

B: M = $\mathrm{ID_A}$ || $\mathrm{ID_B}$ || Objects || Price || Date
B: S = $\mathrm{E_{Bs}}$(M)
(Note: $\mathrm{E_{Bs}}$ means encrypt with B's private key)
B → A: T = $\mathrm{ID_B}$ || $\mathrm{E_{Ap}}$(S)

The notation || is a concatenation operator. When A receives the message T from B, he can decrypt it obtain S= $\mathrm{E_{Bs}}$(M) with his private key and then decrypt $\mathrm{E_{Bs}}$(M) to obtain the trading message M with B's public key. So, he can confirm the content of the trade message from the buyer B. If the seller A accepts the deal, he calculates SHA-1(Bp) and concatenates it with $\mathrm{E_{Bs}}$(M) to be encoded into a bar-code I.

A: S = $\mathrm{E_{Bs}}$(M) = $\mathrm{D_{As}}$($\mathrm{E_{Ap}}$(S))
A: M = $\mathrm{D_{Bp}}$($\mathrm{E_{Bs}}$(M))
If A accepts the deal,
A: I = SHA-1(Bp) || $\mathrm{E_{Bs}}$(M); otherwise the trade is rejected.

The verifying procedure for the seller A is shown in Fig. 1. The second stage of the proposed bar-code watermarking system is the embedding procedure. The flowchart of the embedding procedure is shown in Figure 2. In the bar-code image, each byte takes 17 pixels width of bar-code image. The default bar-code height is 21 pixels. So, the bar code image is a bitmap whose size is (64*17) by 21, equal to 22848 pixels.

Since we want that the watermark embedded algorithm can be public and the watermark still can maintain its robustness. We choose the embedded positions according to the coefficients in the frequency domain and the seller's private key. Therefore, each image has different embedded point list.

Suppose we want to choose five positions to embed the watermark each block. We sort the 63 AC terms by their absolute values in a descending order first. Then, we pick out the top ten terms and encrypt them with A's private key. We sort the value after encryption in a descending order again. The five larger ones are the positions where we want to embed the watermark. Eq. (1) is the watermarking equation. C_i is the original coefficient and $C_i^{'}$ is the embedded coefficient. W_i is a sequence of gray levels of bar-code bitmap composed of 0 and 255 and α is an unfixed scaling factor varying with the DCT coefficients. The smaller the coefficients are, the larger the scaling factor is adjusted [6]. The rule of adjusting α value is shown in Eq. (2).

$$\begin{cases} C_i^{'} = C_i(1+\alpha), \text{if } \mathrm{W_i} = 255 \\ C_i^{'} = C_i(1-\alpha), \text{ if } \mathrm{W_i} = 0 \end{cases} \quad (1)$$

$$\begin{cases} if\ (\ C_i > 35\ ) & \alpha = 0.15 \\ else\ if\ (\ 35 > C_i > 20\ ) & \alpha = 0.3 \\ else\ if\ (\ 20 > C_i > 10\ ) & \alpha = 0.6 \\ else\ if\ (\ 10 > C_i > 5\ ) & \alpha = 0.8 \\ else\ if\ (\ C_i < 5\ and\ W_i = 255\ ) & \alpha = 0,\quad C_i = C_i + 12 \\ else\ if\ (\ C_i < 5\ and\ W_i = 0\ ) & \alpha = 0,\quad C_i = C_i - 12 \end{cases} \tag{2}$$

Eq. (3) is the equation to retrieve the watermark sequences. C_i is the original coefficient and $C_i^{'}$ is the watermarked image coefficient. $W_i^{'}$ is the extracted watermark sequence.

$$\begin{aligned} &T_i = C_i^{'} / C_i \\ &\text{If}\quad T_i \le 1,\ W_i^{'} = 0; \\ &\text{If}\quad T_i > 1,\ W_i^{'} = 255\ ; \end{aligned} \tag{3}$$

The information I can be clipped into two parts. The first part is SHA-1(Bp) and the second part is $E_{Bs}(M)$. We decrypt the cipher-text $E_{Bs}(M)$ with B's public key to obtain the trade message M.

I = SHA-1(Bp) || $E_{Bs}(M)$
$D_{Bp}(E_{Bs}(M)) = M$
M = ID_A || ID_B || Objects || Money || Date

3. SIMULATION RESULTS

In this experiment, four 512 * 512 test images "Jet", "Lena", "Scene" and "Babara" are used to be the original images. The trading message for the test image "jet" is:

"&ALICE &BOB &2002\5\18 &4,000,000"

The trading message hided into each original image is different. We encode these messages to the bar-code that are used as the watermark. We define the similarity value between W and W^* as

$$\text{Sim}(W, W^*) = \frac{\sum_i \overline{W_i \oplus W_i^*}}{\sum_i \overline{W_i \oplus W_i}} \tag{4}$$

Fig. 3 shows the damaged images after JPEG compression and uniform noise. Fig. 4 shows the processed images of "Jet" after several kinds of image processing. We also attack the "Jet" image with a painter algorithm. The attacked images are shown in Fig. 5. The image quality, the extracted message and the number of errors of the test image "Jet" are listed in Table 1. From Table 1, we see that the extracted trade message has no error even when the watermarked image has been attacked by JPEG compression, uniform noise corruption, image cropping, image blurring, image sharpening and painting.

4. CONCLUSION

We have successfully combined the cryptology technique and a bar code watermark. Cryptography is used not only for choosing embedded positions but also for protecting the trading messages. We used digital signature to verify the buyer's information and confirm the integrity of messages.
We use the RSA encryption algorithm to trace the buyer's information in the unauthorized copies to prevent unauthorized distribution of images.

Acknowledgement
This work was supported by National Science Council under NSC91-2213-E007-060.

5. References

[1] N. Nikolaidis and I. Pitas, "Copyright protection of images using robust digital signatures," in Proceedings, IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 4, May 1996, pp. 2168-2171.

[2] Houng-Jyh Wang and C.-C. Jay Kuo, "Image Protection via Watermarking on Perceptually Significant Wavelet Coefficients." IEEE 1998 Workshop on Multimedia Signal Processing, Redondo Beach, CA. Dec. 7-9, 1998.

[3] I. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia", IEEE Transaction on Image Processing, vol. 6, pp. 1673-1687, Dec. 1997.

[4] C.-T. Hsu and J.-L. Wu, "Hidden Digital Watermarks in Images, "IEEE Transactions on Image Processing, Vol. 8,No. 1, pp.58-68, Jan. 1999.

[5] S. Craver, N. Memon, B. L and M. M Yeung "Resolving rightful ownerships with invisible watermarking techniques: limitations, attacks, and implications," IEEE Journal on Selected Areas in Communications, Vol. 16 Issue: 4, May 1998, pp. 573 –586.

[6] W. Stallings, "Cryptography and network security: principles and practice, 2nd Edition." Prentice Hall, 1999.

[7] Ping Wath Wong, Nasir Memon, "Secret and Public Key Image Watermarking Schemes for Image Authentication and Ownership Verification", IEEE Transactions on Image Processing, Vol. 10, No. 10, pp 1593-1601, October 2001.

[8] Nasir Memon, Ping Wah Wong, "A Buyer-Seller Watermarking Protocol", IEEE Transactions on Image Processing, Vol. 10, No. 4, pp 643-649, April 2001.

[9] Hiroo Wakaumi, Takatoshi Komaoka, Eiji Hankui, "Grooved Bar-Code Recognition System with Tape-Automated-Bonding Head Detection Scanner", IEEE Transactions on Magnetics, Vol. 36, No. 1, pp 366-370, January 2000

	PSNR	**Sim**	**The Extracted Message**	**Error**
WMKed	**38.35**	**1.000**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Jpeg6	**36.53**	**0.967**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Jpeg4	**35.11**	**0.908**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Blur	**37.52**	**0.909**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Sharpen	**30.91**	**0.938**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Noise5	**29.64**	**0.881**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Noise10	**24.29**	**0.736**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Crop1/4	**9.01**	**0.769**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
CropMore	**4.45**	**0.339**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**
Painter	**7.29**	**0.624**	**&Alice &Bob &2002\5\18 &4,000,000**	**0**

Table 1: The experiment data of the test image "Jet". JPEGx denotes JPEG compression with quality level =x. Noisex denotes the corruption of x% noise and crop1/4 denotes cropping shown in Fig.5c and cropmore denotes cropping shown in Fig. 5d.

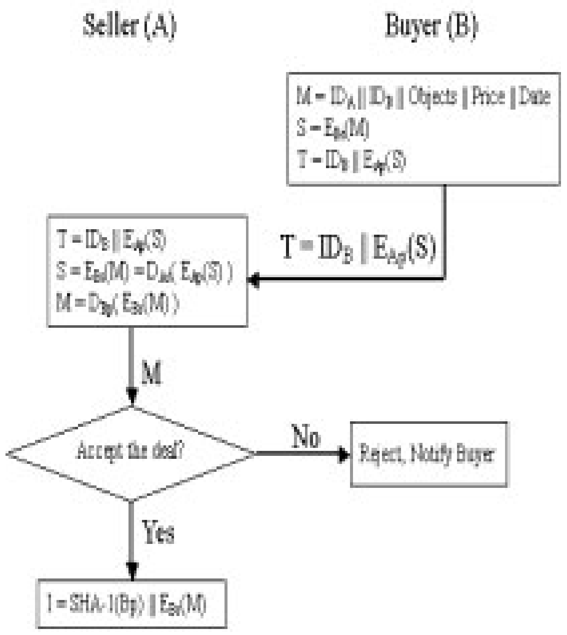

Fig. 1 The flow chart of the verifying procedure

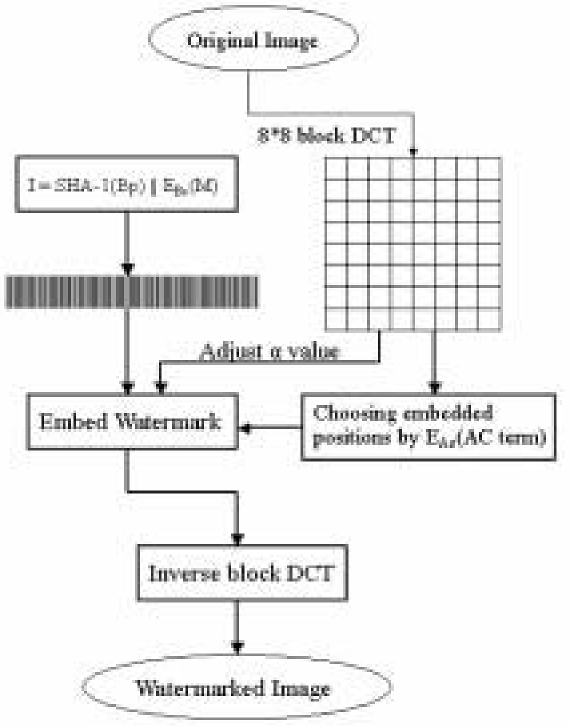

Fig. 2 The flow chart of the embedding procedure

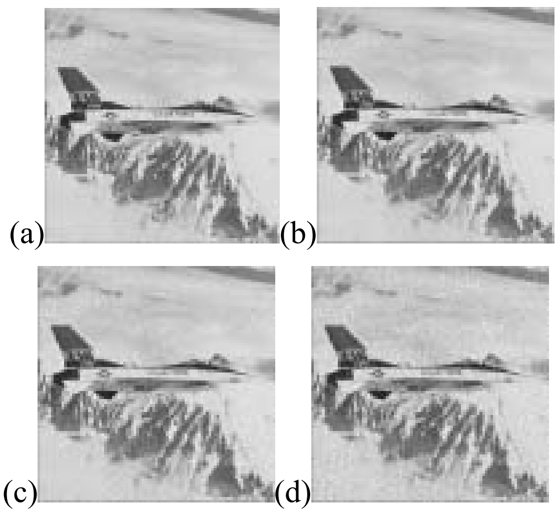

Fig. 3(a) “Jet” after Jpeg compression (quality level =6), PSNR = 36.53, Image size from 256KB to 45.5KB (b) “Jet” after Jpeg compression(quality level =4), PSNR = 35.11, Image size from 256KB to 31.6KB (c) “Jet” after adding 5% uniform noise, PSNR = 29.64 (d) “Jet” after adding 10% uniform noise, PSNR = 24.29

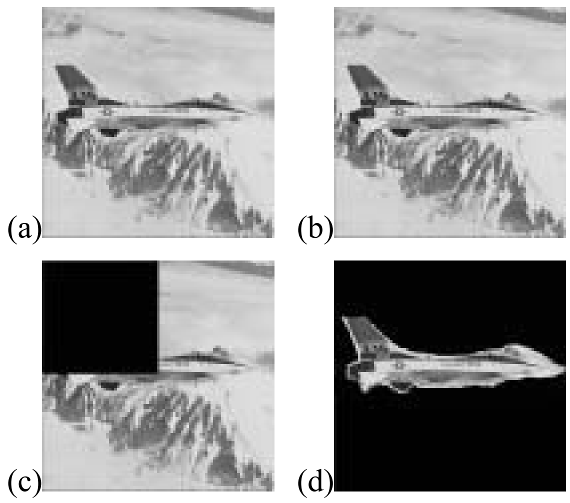

Fig.4 (a) after blurring, PSNR = 37.52 (b) after Sharpening, PSNR = 30.91 (c) After cropping the left-up corner, PSNR = 9.01 (d) After cropping all besides the jet, PSNR = 4.45

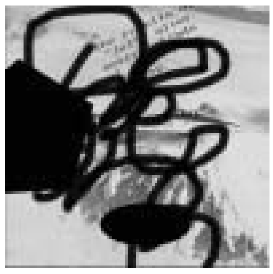

Fig. 5 “Jet” Damaged with a painter, PSNR=7.29