Short Paper

An Embedding Algorithm for Multiple Watermarks

CHIN-SHIUH SHIEH, HSIANG-CHEH HUANG*, FENG-HSING WANG
AND JENG-SHYANG PAN
Department of Electronic Engineering
National Kaohsiung University of Applied Sciences
Kaohsiung, 807 Taiwan
**Department of Electronics Engineering*
National Chiao Tung University
Hsinchu, 300 Taiwan

A new algorithm for embedding several watermarks into the same original source is proposed in this paper. Instead of conventional watermark embedding algorithms that embed one watermark in one specific domain, we propose algorithms for embedding multiple watermarks into different compression domains to protect the ownership of the original images. The proposed schemes for embedding and extraction in different domains vary from the existing methods reported in the literature. Simulation results show their robustness for surviving intentional attacks and, hence, their effectiveness in protecting copyrights. They are also easy to implement.

Keywords: digital watermark, vector quantization (VQ), discrete cosine transform (DCT), joint photographic experts group (JPEG), secret key, blind detection, Stirmark

1. INTRODUCTION

With the rapid development of the computer industry and the swift growth of the Internet, there is now widespread use of digital multimedia products in daily life. Digital media like images, audio, and video have the advantages of easy editing, unlimited and perfect copying, and fast distribution over the Internet. Unfortunately, these advantages have led to the unauthorized distributing and copying of digital media, a serious problem for copyright protection that might undermine the publishing and entertainment industries. One solution for protecting ownership is to put *watermarks* into digital media. Digital watermarking has attracted a great deal of attention recently from researchers focusing on academic investigations and practical applications. Techniques for processing watermarks in text, images, video and audio have become increasingly popular in recent years [1, 2]. If we suspect that some images, audio, or video are copies of pirated versions, we may examine the media in question by extracting the watermarks therein and then check the ownership of such media [3].

Received February 1, 2001; revised December 18, 2001; accepted March 8, 2002.
Communicated by Shiuh-Pyng Shieh.

From the algorithm design viewpoint, the three most critical requirements are: (a) transparency or imperceptibility, (b) robustness to attacks, and (c) watermark capacity [4]. Although these three requirements are all very important, they conflict with each other. Invisible or imperceptible signals are generally small in magnitude; thus, they are vulnerable to attacks [5]. Spread spectrum techniques [6] can be applied to reliably hide low rate information inside a high rate original signal. However, higher reliability generally demands a lower information rate [7].

To survive intentional attacks, embedding multiple watermarks with different compression domains into the same source might be a good idea. If one of the watermarks can survive attacks, we can claim ownership of the source. Therefore, we provide several ways to embed watermarks with several secret and public keys. Our algorithm can successfully survive a variety of attacks, including the watermark benchmark *Stirmark* [8].

In this paper, we propose an algorithm for embedding multiple watermarks into the vector quantization (VQ) domain, as well as for hiding the secret keys associated with the watermarks in the transform domain. The motivation for and concept behind our algorithm differ from those of other algorithms, such as those in [9]. We have also tested several attacking methods, some of which include Stirmark, on the watermarked images to show the robustness of the proposed algorithm. To extract the watermark, we do not need the original image, which can be referred to as 'blind detection.' Simulation results illustrate the effectiveness of our multiple-watermarking system.

This paper is organized as follows. Section 2 demonstrates the algorithm for embedding multiple watermarks. Section 3 depicts the watermark extraction algorithm. Section 4 presents the simulation results obtained when no attack was applied. Section 5 shows the robustness of the algorithm to intentional attacks and the Stirmark test. Section 6 briefly discusses the proposed algorithm and the simulation results. We conclude this paper in section 7.

2. THE ALGORITHM FOR EMBEDDING MULTIPLE WATERMARKS

Let the input image be $\boldsymbol{X}$ with size $M \times N$. Our goals are to embed multiple watermarks using different algorithms into $\boldsymbol{X}$ and to output a watermarked reconstruction with a secret key.

2.1 Embedding the Watermark Into the VQ Domain

Assume that the first binary-valued watermark to be embedded is $\boldsymbol{W}_1$ of size $M_{W1} \times N_{W1}$. We perform the VQ operation first [10] and get codewords using the nearest search algorithm. Afterwards, we are able to embed the watermark with the characteristics of the indices in the VQ domain. In order to ensure survival against the picture-cropping attacks described in section 5.3, a pseudo-random number traversing method is applied to disperse the spatial relationship of the watermark. With a secret key, key_1, in the pseudo-random number generating system [11], we have

$$\mathbf{W}_{P1} = \text{permute}\,(\mathbf{W}_1, \text{key}_1), \tag{1}$$

and we can embed the permuted watermark $\boldsymbol{W}_{P1}$ into the VQ indices.

In the VQ encoding phase, $\boldsymbol{X}$ is divided into vectors $\boldsymbol{x}$ of size $\frac{M}{M_{W1}} \times \frac{N}{N_{W1}}$; then, each $\boldsymbol{x}$ finds its nearest codeword c_i in the codebook $\boldsymbol{C}$, and the index i is assigned to $\boldsymbol{x}$. During decoding with the VQ indices, the decoder merely performs a table look-up process on the received index i to obtain $\hat{c}_i$ and then gets the reconstruction image $\hat{\boldsymbol{X}}$.

In our algorithm, we perform VQ with codebook size L. The codebook, $\boldsymbol{C}$, and the codewords, c_i, $i \in [0, L-1]$, can be represented as

$$\boldsymbol{C} = \{c_0, c_1, \ldots, c_{L-1}\}. \tag{2}$$

After that, we collect every two codewords in Eq. (2) to form one embedding group. To ensure minimal distortion of the sum of squared errors in the reconstructed image, we need to use some optimization algorithms to arrange the order of the codewords. Here, we may apply the tabu search approach [12, 13] or the genetic algorithm (GA) for optimization [14].

In this paper, we complete the training of the codewords with GA and then gather every two nearest codewords together to form one embedding group:

$$\begin{aligned} \boldsymbol{C} &= \bigcup_{i=0}^{\frac{L}{2}-1} \{c_{2i}, c_{2i+1}\} \\ &= \bigcup_{i=0}^{\frac{L}{2}-1} \{g_i\} \end{aligned}. \tag{3}$$

We rearrange the order of the codewords in $\boldsymbol{C}$ as follows.

Step 1 Randomly select two codewords, c_m and c_n, into one pair until all the L codewords are chosen, and form ten individuals in the GA training process.

Step 2 Calculate the resulting squared error (SE) between the reconstructed image and the original image in Step 1, and obtain $\mathrm{SE}_{\mathrm{old},i}$, i: the i^{th} individual, $1 \leq i \leq 10$.

Step 3 Perform the crossover, mutation, and selection operations in GA on the codeword pairs in Step 1, and get the new codeword pairs in the ten new individuals of the next training iteration.

Step 4 Calculate the resulting SE between the reconstructed image and the original image in Step 3, and obtain $\mathrm{SE}_{\mathrm{new},i}$, $1 \leq i \leq 10$.

Step 5 If $\mathrm{SE}_{\mathrm{new},i} < \mathrm{SE}_{\mathrm{old},i}$, replace the codeword pairs in Step 1 with those in Step 3. In addition, take the replacement operation $\mathrm{SE}_{\mathrm{old},i} = \mathrm{SE}_{\mathrm{new},i}$, $1 \leq i \leq 10$, for every individual in the current training iteration.
If not, keep the codeword pairs in Step 1 unchanged.

Step 6 Iterate Step 1 through Step 5 for a pre-determined number of times, and output the combinations of codeword pairs $\{g_i\} = \{c_{2i}, c_{2i+1}\}$ of the final iteration.

In Eq. (3), if the watermark bit to be embedded is zero, transmit the even-indexed codeword c_{2i}; if not, transmit the odd-indexed codeword c_{2i+1} in the g_i. After finishing the embedding operation, we perform the inverse VQ operation on the codewords and produce the first watermarked image $\boldsymbol{X}'$.

2.2 Embedding the Watermark Into the Secret Key of the DCT Domain

To proceed with the embedding process with another watermark, we perform the 8×8 discrete cosine transform (DCT) [15] and get

$$\boldsymbol{Y} = \mathrm{DCT}\,(\boldsymbol{X}'). \tag{4}$$

As we continue with embedding of the second binary-valued watermark, $\boldsymbol{W}_2$, into the secret key, key_2, we need to modify the DCT coefficients $\boldsymbol{Y}$. Given that $\boldsymbol{W}_2$ has size $M_{W2} \times N_{W2}$, we also perform the pixel permutation process in Eq. (1) to get

$$\boldsymbol{W}_{P2} = \mathrm{permute}\,(\boldsymbol{W}_2, \mathrm{key}_2). \tag{5}$$

Assume that the 64 DCT coefficients in the 8×8 block (m, n) of $\boldsymbol{X}'$ can be represented by

$$\mathbf{Y}_{(m,n)} = \bigcup_{k=0}^{63} \{Y_{(m,n)}(k)\}, 1 \le m \le \tfrac{M}{8}, 1 \le n \le \tfrac{N}{8}. \tag{6}$$

Here, we propose use of the ‘F-Table Generation Method’ to help embed $\boldsymbol{W}_{P2}$ into the secret key, key_3.

To meet the three critical requirements noted in section 1, *i.e.*, imperceptibility, robustness, and watermark capacity, we ought to hide the watermark information in the middle-frequency set of the transform domain for robustness and imperceptibility [16]. The middle-frequency set, $\boldsymbol{MF}$, will be chosen randomly to embed $\boldsymbol{W}_{P2}$. The coefficients are chosen from the middle-frequency bands in order to influence the watermarked image quality as little as possible and to ensure survival the following loss data compression. For each 8×8 image block, only $\left(64 \times \frac{M_{W2} \cdot N_{W2}}{M \cdot N}\right)$ coefficients will be included in the middle-frequency set and can be represented by

$$\boldsymbol{MF} = \bigcup_{m=1}^{M/8} \bigcup_{n=1}^{N/8} \{MF_{(m,n)}(i) = Y_{(m,n)}(k)\}, \tag{7}$$

where $k = 1, 2, \ldots, 63$, and $i = 0, 1, \cdots, \left(\frac{64 \cdot M_{W2} \cdot N_{W2}}{M \cdot N} - 1\right)$ are the $\left(\frac{64 \cdot M_{W2} \cdot N_{W2}}{M \cdot N}\right)$ bands selected from the 63 DCT coefficients (except for the DC coefficient) in each block. Once the bands are selected, we designate the mapping between i and k as the secret key, key_0. After that, we generate a reference table, $\boldsymbol{F}$, with the DCT coefficients $\boldsymbol{Y}$:

$$\boldsymbol{F} = \{F(0), F(1), \ldots, F(63)\}, \tag{8}$$

where

$$F(i)=\frac{1}{\frac{M}{8}\cdot\frac{N}{8}}\cdot\sum_{m=1}^{M/8}\sum_{n=1}^{N/8}Y_{(m,n)}(i),\quad i=0,1,\cdots,63 \tag{9}$$

indicates the average of the corresponding DCT coefficients within the whole image. Using the F-table generation method, we prepare to embed $\boldsymbol{W}_{P2}$ into the secret key, key_3.

It is well known that the magnitudes of the 64 DCT coefficients differ a lot. The idea behind the F-table generation method is to magnify the magnitudes of the AC coefficients by a factor $F(i)$ to make them comparable with their counterparts among the DC coefficients. In addition, the DC value of each block plays the most important role in the reconstruction process. Hence, we use the DC value of every block, *e.g.*, $Y_{(m,n)}(0)$ in $\boldsymbol{Y}_{(m,n)}$ of block (m, n), as a reference value and produce the relationships among the DC value of one block, the current AC coefficients, and the F-table for further operations using the binary-valued watermark. Consequently, we can calculate the *polarities*, $\boldsymbol{P}$, of the DCT coefficients in the middle-frequency set by means of

$$\boldsymbol{P}=\bigcup_{m=1}^{M/8}\bigcup_{n=1}^{N/8}\bigcup_{k\in\boldsymbol{MF}}\{P_{(m,n)}(k)\} \tag{10}$$

and

$$P_{(m,n)}(k)=\begin{cases}1, & \text{if } (Y_{(m,n)}(k)\cdot F_{(m,n)}(k)\geq F_{(m,n)}(0), k\in\boldsymbol{MF},\\ 0, & \text{otherwise.}\end{cases} \tag{11}$$

With $\boldsymbol{W}_{P2}$ and the accompanying polarities, we can generate the secret key, key_3, with the exclusive-or operation:

$$\text{key}_3=\bigcup_{m=1}^{M/8}\bigcup_{n=1}^{N/8}\bigcup_{k\in\boldsymbol{MF}}(P_{(m,n)}(k)\oplus W_{P2}(k)). \tag{12}$$

When the embedding operations in both the VQ domain and the secret key in the DCT domain are finished, the DCT coefficients, $\boldsymbol{Y}$, will be inverse-DCT processed, and we will have the watermarked image $\boldsymbol{X}''$. The third secret key, key_3, cooperates with $\boldsymbol{X}''$ to protect the ownership of the original $\boldsymbol{X}$.

The building blocks for embedding multiple watermarks are shown in Fig. 1.

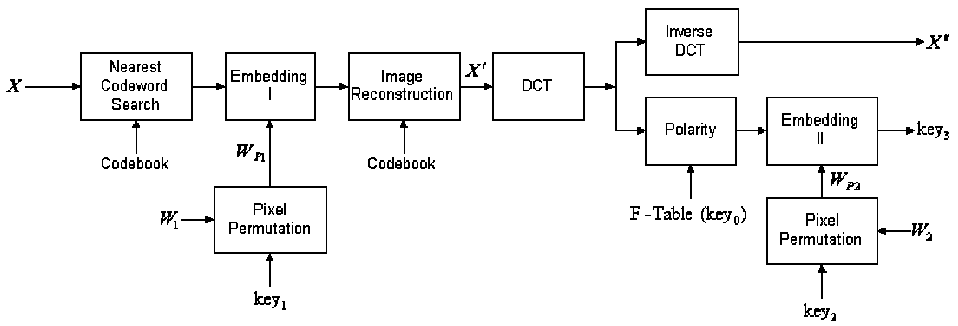


Fig. 1. The flowchart for embedding multiple watermarks.

2.3 Dividing the Embedding Algorithm Into Modules

The watermark embedding techniques described in sections 2.1 and 2.2 can be divided into independent modules to ease the implementation process. Other methods for embedding watermarks can replace the proposed technique or can be inserted into our embedding system for protecting the copyright of the original source.

3. WATERMARK EXTRACTION METHODS TO ACCOMPANY EMBEDDING ALGORITHMS

To extract watermarks, we can process the watermarked image in parallel to shorten the processing time. In our algorithm, we can divide the extraction process into two branches as shown in Fig. 2. This extraction procedure can be categorized as 'blind detection' because the original image is not needed in the watermark extractor. This is also one advantage for practical implementation with our algorithm.

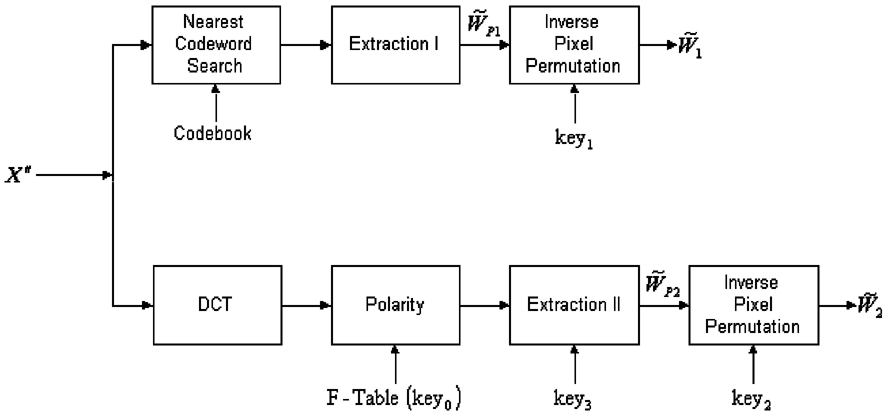


Fig. 2. The flowchart for extracting watermarks.

First, we wish to extract the watermark $\boldsymbol{W}_1$ in the VQ domain. Making use of the nearest codeword search algorithm to produce the VQ codewords from the watermarked output $\boldsymbol{X}''$, we are able to determine whether it is an odd- or even-indexed codeword in the corresponding group $\{g_i\}$ in Eq. (3). Consequently, we can extract the estimate of the permuted watermark, $\tilde{\boldsymbol{W}}_{P1}$ and perform the inverse permutation to get $\tilde{\boldsymbol{W}}_1$.

Second, we perform DCT on $\boldsymbol{X}''$ and use the DCT coefficients computed from the watermarked image to build a new F-table $\tilde{\boldsymbol{F}}$. Following the criteria in the embedding operations described in section 2.2, we are able to calculate the polarity estimates $\tilde{\boldsymbol{P}}$ and obtain the permuted watermark $\tilde{\boldsymbol{W}}_{P2}$ with key$_3$. Consequently, we have the reconstructed watermark $\tilde{\boldsymbol{W}}_2$ with the secret key, key$_2$.

The complete flowchart for extracting watermarks is illustrated in Fig. 2.

4. SIMULATION RESULTS

Here we used the test image Lena of size 512 × 512, shown in Fig. 3, and the two binary watermark images, word and rose, both of size 128 × 128 as shown in Fig. 4 (a) and (b), respectively. We checked the peak signal-to-noise ratio (PSNR) of the watermarked images and the normalized cross-correlation (NC) between the original and the extracted watermarks.

Fig. 3. The original Lena used in our simulation, size 512 × 512.

(a)

(b)

Fig. 4. The binary watermarks to be embedded, both of size 128 × 128, (a) word, (b) rose.

The mean squared error (MSE) between the original and watermarked images is defined by

$$\mathrm{MSE} = \frac{1}{M \cdot N} \sum_{i=1}^{M} \sum_{j=1}^{N} (x(i, j) - x'(i, j))^2, \tag{13}$$

where $x(i, j)$ and $x'(i, j)$ denote the pixel value in position (i, j) of the original and the watermarked images, respectively. Consequently, we can represent the PSNR by

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right) \text{ (dB)}. \tag{14}$$

Another factor for evaluation is the normalized cross-correlation (NC) between the embedded watermark, $W(i, j)$, and the extracted watermark, $\tilde{W}(i, j)$:

$$\mathrm{NC} = \frac{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}[W(i, j)\cdot \tilde{W}(i, j)]}{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}[W(i, j)]^2}. \tag{15}$$

As can be observed from the definition, NC $\in$ [0, 1]. The higher the NC value, the greater the similarity between the embedded watermark and the extracted one.

We will present the results for the VQ and DCT compression algorithms in section 4.1, and the details of embedding watermarks in section 4.2.

4.1 The Conventional VQ and DCT Compression Algorithms

We compressed the original image using both VQ and DCT. In our simulation, we set the codebook size L to 512 and let the block size be 4×4 in the VQ domain. After that we applied the LBG algorithm (with threshold = 10^{-4}) for compression [10]. We also used the conventional 8×8 DCT and the luminance quantization table in the JPEG standard [17], shown in Fig. 5, to embed the watermark in the DCT domain. The resulting PSNR values can be regarded as reference values for comparison between the watermarked image and the original one. The results are shown in Fig. 6.

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

Fig. 5. The luminance quantization table in the JPEG standard.

4.2 Embedding With Two Watermarks in the VQ Domain and the Secret Key

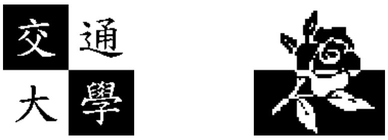
Following the procedure described in section 2, we obtained the watermarked output shown in Fig. 7. Objectively speaking, the PSNR difference between the watermarked output and the VQ compressed image is only 1.96 dB. In addition, the subjective image quality is good in Fig. 7, and the hidden watermarks are not perceptible.

(a) (b)

Fig. 6. Compression of the original image Lena. (a) With VQ, PSNR = 31.53 dB, (b) with DCT, PSNR = 35.75 dB.

Fig. 7. The watermarked image embedded with the two watermarks shown in Fig. 4. PSNR = 29.57 dB.

Using the watermark extraction algorithm described in section 3, we obtained the extracted watermarks shown in Fig. 8. Both NC_1 and NC_2 were 1.0. The words in Fig. 8(a) and the patterns in Fig. 8(b) are identical to their counterparts in Fig. 4(a) and Fig. 4(b) because $NC_1 = NC_2 = 1.0$. Hence, the watermarks could be perfectly extracted using our algorithm. This also shows the effectiveness of our algorithm for embedding watermarks in the secret key.



(a) (b)

Fig. 8. The extracted binary watermarks when no attacks was applied, (a) $NC_1 = 1.0$, (b) $NC_2 = 1.0$.

5. ATTACKING METHODS

We tried several attacking methods to evaluate the robustness of the proposed algorithm. In addition to the VQ compression attack, we also incorporated the watermark benchmark "Stirmark" [8] to perform other attacks, including JPEG compression, image cropping, and several spatial filtering methods. The results of the attacked images are shown in Fig. 9. These extracted watermarks and their corresponding NC values are illustrated in Fig. 10.

(a) VQ.

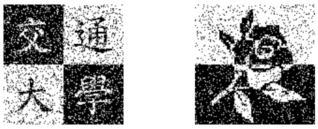
(b) JPEG, QF = 70%.

(c) Cropping.

(d) Low-pass filtering.

(e) Median filtering.

Fig. 9. The watermarked images attached using the methods described in section 5.

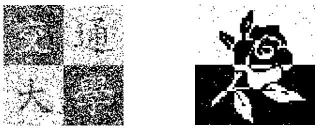


$NC_1 = 0.8611$ $NC_2 = 0.9033$

(a) VQ.

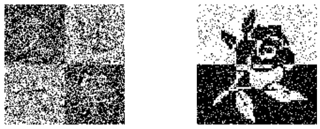
$NC_1 = 0.8447$ $NC_2 = 1.0$

(b) JPEG, QF = 70%.



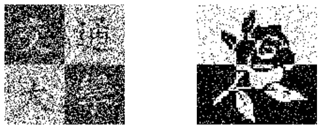
$NC_1 = 0.7684$ $NC_2 = 0.9949$

(a) Cropping.



$NC_1 = 0.6692$ $NC_2 = 0.9135$

(b) Low-pass filtering.



$NC_1 = 0.7307$ $NC_2 = 0.9380$

(e) Median filtering.

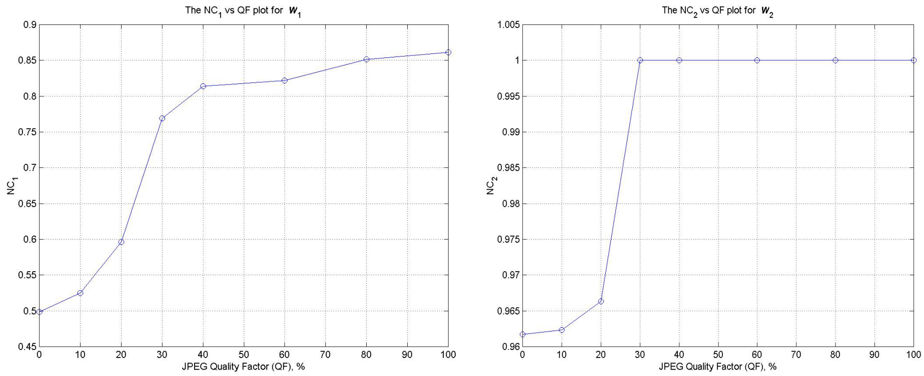
Fig. 10. The watermarks extracted using different attacking methods.

5.1 VQ Compression

When attackers wish to remove hidden watermarks, they may compress the watermarked images using some other VQ codebooks and decode the VQ indices to get the reconstruction. When we took the attacked image in Fig. 9(a) and applied our watermark extraction algorithm, we got the attacked watermarks shown in Fig. 10(a). Both watermarks could survive VQ compression attacks.

5.2 JPEG Compression

Attackers can modify watermarked images by varying the quality factor (QF) in the JPEG compression system. If we choose larger QF, we get better image quality. In addition, with different QF values, we can see whether embedded watermarks can survive JPEG compression or not. We conduct experiments with QF from 0% to 100% as shown in Fig. 11. Fig. 9(b) shows the attacked image with QF = 70%, and Fig. 10(b) shows their corresponding watermarks. Both watermarks could survive JPEG compression attacks when QF ≥ 40%. When we lowered the QF further, $\tilde{\boldsymbol{W}}_2$ could still survive. Therefore, we are able to claim ownership of the original source because of the survival of one of the watermarks.



(a) (b)

Fig. 11. The NC vs. JPEG QF plots. (a) The results of $\tilde{\boldsymbol{W}}_1$. The lower the QF value, the less chance that $\tilde{\boldsymbol{W}}_1$ can survive. (b) The results of $\tilde{\boldsymbol{W}}_2$. Varying the QF values has almost no effect on $\tilde{\boldsymbol{W}}_2$.

5.3 Image Cropping

Attackers can alter watermarked images by cropping out some parts of the images. In our simulation, we cropped out the upper-left quarter of the watermarked image, and replaced it with an all-black image, as shown in Fig. 9(c), in the hope of removing the watermarks. Fig. 10(c) depicts the watermarks extracted from the attacked image. Although the NC in $\tilde{\boldsymbol{W}}_1$ was not very large, the information conveyed therein was still recognizable. Furthermore, image cropping had almost no effect on $\tilde{\boldsymbol{W}}_2$. Hence, we were able to demonstrate the effectiveness of our algorithm.

5.4 Spatial Filtering

The application of spatial masks in image processing is usually called spatial filtering, and the masks themselves are called spatial filters [18]. The basic approach to spatial filtering is to sum the products between the mask coefficients and the luminance of the pixels under the mask at a specific location in the image. 3×3 spatial masks were applied as described in sections 5.4.1 and 5.4.2 in attacks with the Stirmark benchmark. We checked the effects of spatial filtering on the watermarked images and evaluated the robustness of our algorithm.

5.4.1 Low-pass filtering

The mask for low-pass filtering, also referred to as neighborhood averaging, can be shown by

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}.$$

Fig. 9(d) shows the low-pass filtered image, which is somewhat blurred compared with the watermarked output shown in Fig. 7. Fig. 10(d) displays the resulting extracted watermarks. Although $\tilde{\boldsymbol{W}}_1$ is not easily recognizable and its NC value is not high, $\tilde{\boldsymbol{W}}_2$ still survived the low-pass filtering attack.

5.4.2 Median filtering

In median filtering, the luminance of each pixel is replaced by the median of the gray levels in the range of the spatial mask of that pixel. Thus, the main function of median filtering is to force points with distinct intensities to be more like the luminance of their neighboring pixels.

Fig. 9(e) shows the image after median filtering, and Fig. 10(e) shows the watermarks after extraction. Information conveyed in $\tilde{\boldsymbol{W}}_1$ is perceptible, and it can be seen that the median filtering attack had little effect on $\tilde{\boldsymbol{W}}_2$.

6. DISCUSSION

We summarize our findings in the following:

A. In the embedding and extraction systems in our proposed algorithm, the secret keys, key_0, key_1, and key_2, can remain unchanged, while key_3 is dependent from the second embedded watermark, $\boldsymbol{W}_2$. From another point of view, key_i, $i = 0, 1, 2$, can be treated as the public keys. Hence, we do not have to transmit them every time.

B. The VQ watermarked image, $\boldsymbol{X}'$, is quite similar to the output watermarked image, $\boldsymbol{X}''$, because there are only small quantization errors between the two. Hence, embedding with this method will cause the output image quality to deteriorate somewhat.

C. From the simulation results obtained after applying several attacking techniques, we can see that at least one of the watermarks could survive. Therefore, we are able to assert that the proposed algorithm is effective.

D. We can extend our watermarking system by inserting other embedding/extraction modules into some other domain to archive more flexible protection of multimedia copyrights.

7. CONCLUSIONS

In this paper, we have proposed an algorithm for embedding multiple watermarks to protect the ownership of an image source. Our algorithm meets the imperceptibility, robust, and capacity requirements for a practical watermarking system. The overall system can be divided into several modules, thus making it easy to implement. The watermark embedding techniques differ from one module to another in the hope that at least one of the watermarks can survive attacks. Experimental results show the robustness of the proposed algorithm against intentional attacks of different types. We accomplished the work for embedding two watermarks into one single source successfully in this study. Further studies will concentrate on embedding more watermarks using a wide variety of techniques to protect the ownership of digital media.

REFERENCES

1. F. Hartung and M. Kutter, "Multimedia watermarking techniques," in *Proceedings of the IEEE*, Vol. 87, 1999, pp. 1079-1107.
2. C. I. Podilchuk and E. J. Delp, "Digital watermarking: algorithms and applications," *IEEE Signal Processing Magazine*, Vol. 16, 1998, pp. 525-539.
3. G. Voyatzis and I. Pitas, "The use of watermarks in the protection of digital multimedia products," in *Proceedings of the IEEE*, Vol. 87, 1999, pp. 1197-1207.
4. R. B. Wolfgong, C. I. Podilchuk, and E. J. Delp, "Perceptual watermarks for digital images and video," in *Proceedings of the IEEE*, Vol. 87, 1999, pp. 1108-1126.
5. M. Kutter and F. Petitcolas, "Fair benchmarking for image watermarking systems," in *Proceedings of the SPIE 3657, Security and Watermarking of Multimedia Contents*, 1999, pp. 226-239.
6. I. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, Vol. 6, 1995, pp. 1673-1687.
7. S. Katzenbeisser and F. Petitcolas, *Information Hiding – Techniques for Steganography and Digital Watermarking*, Artech House, Norwood, 2000.
8. F. Petitcolas, "Image watermarking – Stirmark," http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark, 2000.
9. C. -S. Lu and H. -Y. M. Liao, "Multipurpose watermarking for image authentication and protection," *IEEE Transaction on Image Processing*, Vol. 10, 2001, pp. 1579-1592.
10. Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, Vol. COM-28, 1980, pp. 84-95.
11. J. G. Proakis, *Digital Communications*, 3rd ed., McGraw-Hill, New York, 1995.
12. Z. M. Lu and S. H. Sun, "Digital image watermarking technique based on vector quantization," *Electronics Letters*, Vol. 36, 2000, pp. 303-305.
13. Z. M. Lu, J. S. Pan, and S. H. Sun, "VQ-based digital image watermarking method," *Electronics Letters*, Vol. 36, 2000, pp. 1201-1202.
14. D. E. Goldberg, *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Massachusetts, 1989.
15. N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transforms," *IEEE Transactions on Computer?Computing?*, Vol. C-23, 1974, pp. 90-93.
16. C. T. Hsu and J. L. Wu, "Hidden digital watermarks in images," *IEEE Transactions on Image Processing*, Vol. 8, 1999, pp. 58-68.
17. W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*, Van Nostrand Reinhold, 1993.
18. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, Reading, Massachusetts, 1992.

Chin-Shiuh Shieh (謝欽旭) received the B.S. degree in Electronic Engineering from National Taiwan Institute of Technology, Taiwan, in 1989, and the M.S. degree in Electrical Engineering from National Taiwan University, Taiwan, in 1991. He has been working in the areas of classifier systems, neural networks, fuzzy systems, reinforcement learning, evolutionary computation, and vector quantization. His current research interests are in self-learning fuzzy systems using evolutionary techniques, computer networking, and various issues in vector quantization.

Hsiang-Cheh Huang (黃祥哲) received the B.S., M.S. and Ph.D. degrees in Electronics Engineering from National Chiao Tung University, Taiwan, R.O.C. Currently, he is a post-doctoral researcher in the Department of Electronics Engineering, National Chiao Tung University, Taiwan, R.O.C. His current research interests include pattern recognition and image processing.

Feng-Hsing Wang (王楓興) received the B.S. degree in Electronic Engineering from the National Kaohsiung University of Applied Sciences, Taiwan, R.O.C. Currently, he is pursuing his Ph.D. degree at the University of South Australia, Adelaide, Australia. His current research interests include computer architecture and image processing.

Jeng-Shyang Pan (潘正祥) received the B.S. degree in Electronic Engineering from the National Taiwan Institute of Technology, Taiwan, in 1986, the M. S. degree in Communication Engineering from National Chiao Tung University, Taiwan, R.O.C. in 1988, and the Ph.D. degree in Electrical Engineering from the University of Edinburgh, U. K., in 1996. Currently, he is a professor in the Department of Electronic Engineering, the National Kaohsiung University of Applied Sciences, Taiwan, R.O.C. His current research interests include pattern recognition, speech coding and image processing.