VQ-Based Watermarking Techniques

Hsiang-Cheh Huang [1,*], Shu-Chuan Chu [2], and Yu-Hsiu Huang [3],

[1] Department of Microelectronic Engineering, National Kaohsiung Marine University,
Kaohsiung City 811, Taiwan
huang.hc@gmail.com

[2] Department of Information Management, Cheng-Shiu University,
Kaohsiung County 833, Taiwan
scchu@csu.edu.tw

[3] Department of Electronics Engineering, Cheng-Shiu University,
Kaohsiung County 833, Taiwan
yhhuang@csu.edu.tw

Received 30 April 2006; Revised 15 May 2006 ; Accepted 20 May 2006

Abstract. New methods for digital image watermarking based on the characteristics of vector quantization (VQ) are proposed. In contrast with the conventional watermark embedding algorithms to embed only one watermark at a time into the original source, we present several algorithms, including embedding one binary watermark, three binary watermarks, and the grey-level watermark, for copyright protection. The embedding and extraction processes are efficient for implementing with the conventional VQ techniques, and they can be accomplished in parallel to shorten the processing time. After embedding, the embedder would output one watermarked reconstruction image and secret keys associated with the embedded watermarks. These secret keys are then registered to the third party to preserve the ownership of the original source in order to prevent the attackers from inserting counterfeit watermarks. Simulation results show that under no attacks, the embedded watermarks could be perfectly extracted. If there are some intentional attacks such as VQ or JPEG compression, image cropping, spatial filtering, or geometric attacks in our simulation, all the watermarks could survive to protect the copyrights. Therefore, we are able to claim the robustness, usefulness, and ease of implementation of our algorithm.

Keywords: watermarking, vector quantization, attacking scheme

1 Introduction

With the explosive prevalence of the Internet access and usage during the last decade, digital media, including the images, audio, and video sequences, are easily acquired in our daily life [1]. Owing to the digital nature of such media, the unlimited copying and easy distribution have made copyright protection an important topic. Recently, digital watermarking is becoming increasingly popular because of its potential significance in preserving the copyrights or ownerships of the original source.

Digital watermarking is the technique to embed one or more digital signals, called *watermarks*, into the multimedia content and/or into the secret keys for copyright protection. The embedded watermarks can later be extracted or detected from the watermarked multimedia and/or the secret keys for authentication or identification. In this paper, we concentrate our attention on watermarking into the digital images. Spatial-domain based methods [2], transform-domain watermarking techniques based on discrete cosine transform (DCT) [3] or discrete Fourier transform (DFT) [4], and vector quantization (VQ) based watermarking schemes [5] have been explored in literature. In our watermarking system, we employ vector quantization techniques, and we make use of the characteristics of natural images and the efficient VQ compression technique for embedding watermarks into the original image.

This paper is organized as follows. Sec. 2 demonstrates the fundamentals of vector quantization. Sec. 3 through Sec. 5 depict the different watermarking algorithms based on vector quantization. And we conclude this paper in Sec. 6.

[*] Correspondence author

2 Preliminary of Vector Quantization and Notations in This Paper

We briefly describe the basic concept of vector quantization (VQ), and present the corresponding notations to be used in the rest of this paper.

To reduce the space requirement for storage and the bandwidth requirement for communication, a wide variety of compression techniques had been developed [6]. For multimedia applications, less significant information can be sacrificed for higher compression rate, since human sensory system is less sensitive to detailed information. In this kind of applications, vector quantization [7] had received considerable attention for its high compression rate and its essential role in various compression applications. As an extension to scalar quantization, vector quantization works on vectors of raw data. A vector can be fixed numbers of consecutive samples of audio data or a small block of image and video data, for example, the grey-level values of a 4×4 pixel image block forms a 16-dimentional vector. Figure 1 gives a block diagram illustration of the operation of vector quantization compression.

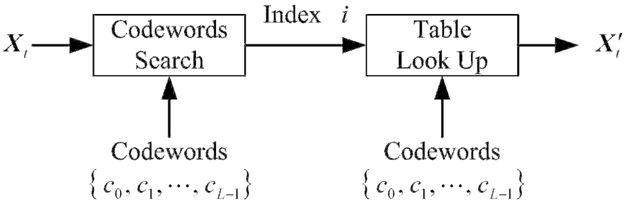


Fig. 1. A block diagram for vector quantization.

The original image $\mathbf{X}$ is composed of the combination of all the input vectors, $\mathbf{X}_t$, $\forall\ t$. In the sender end, the codeword search process looks for a “nearest” codeword, c_i, from the codebook for the given input vector $\mathbf{X}_t$. Euclidean distance is employed in the search process to measure the distance between the two vectors, the codeword c_i and the input vector $\mathbf{X}_t$, as indicated in Eq. (1),

$$i = \arg\min_j D(X_t, c_j), j = 1, 2, ..., L-1 \tag{1}$$

where $D(\bullet, \ \bullet)$ denotes the Euclidean distance.

The index of selected codeword is then transmitted to the receiver end. With the same codebook, the decompression process can easily reconstruct vector $\mathbf{X}'_t$ by simple table look-up, as depicted in Fig. 1. There will be distortion introduced by the compression-decompression process, since $\mathbf{X}'_t$ is only an approximated version of the original $\mathbf{X}_t$. If we work on 8-bit grey-level image, using a block size of 4×4 pixel and a codebook of 256 codewords, then the compression ratio is up to $\frac{4 \times 4 \times 8}{\log_2 256} = 16$. All the reconstructed vectors, $\mathbf{X}'_t$, $\forall\ t$, make up the reconstructed image, $\mathbf{X}'$.

The codebook plays an essential role in vector quantization. The codebook size, or the number of codewords in a codebook, is a trade-off between the reconstructed image quality and the compression rate. The codewords in the codebook decide the resultant compression distortion. A dedicated procedure is required for the generation of appropriate codebook. Among other alternatives, LBG algorithm [8] is widely used in various applications.

3. Watermarking with VQ Block Means

We present the algorithm for VQ-based binary watermark embedding with the means of blocks.

3.1 The algorithm

Let the input image be $\mathbf{X}$ with size $M \times N$. Our goal is to embed a robust watermark with VQ into $\mathbf{X}$, and to output a watermarked reconstruction with a secret key.

Assume that the binary-valued watermark to be embedded is $\mathbf{W}$, having size $M_W \times N_W$. We perform the VQ operation first [8] and get the codewords with 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 survive the picture-cropping attacks, a pseudo-random number traversing method [3] is applied to permute the watermark to disperse its spatial relationships. With a pre-determined key, key_1, we have the permuted watermark $\mathbf{W}_P$ for embedding into the VQ indices.

In the VQ encoding procedure, $\mathbf{X}$ is divided into vectors $\mathbf{x}$ with size $\frac{M}{M_W} \times \frac{N}{N_W}$, then each $\mathbf{x}$ finds its nearest codeword c_i in the codebook $\mathbf{C}$, and the index i is assigned to $\mathbf{x}$. While decoding with the VQ indices, the decoder merely performs a table look-up process on the received index i' to obtain c_i' and then get the reconstruction image $\mathbf{X}'$.

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

$$\mathbf{C} = \left\{c_0, c_1, \cdots, c_{L-1}\right\}. \tag{2}$$

Assume that the block at position (m,n) of the original source $\mathbf{X}$ is $\mathbf{x}(m,n)$. After performing VQ, the indices $\boldsymbol{Y}$ and $\boldsymbol{y}(m,n)$ can be represented by

$$\mathbf{Y} = \text{VQ}(\mathbf{X}), \tag{3}$$

$$\mathbf{y}(m,n) = \text{VQ}\left(\mathbf{x}(m,n)\right) \in \mathbf{C}. \tag{4}$$

To embed the binary watermark into the original source, we need to calculate the *polarities*, $\mathbf{P}$, of the VQ indices. For natural images, the VQ indices among the neighboring blocks tend to be very similar, and we can make use of this property to generate $\mathbf{P}$.

Calculating the variance of $\mathbf{y}(m,n)$ and the indices of its surrounding blocks with

$$\sigma^2(m,n) = \left(\frac{1}{9}\sum_{i=m-1}^{m+1}\sum_{j=n-1}^{n+1}\mathbf{y}^2(i,j)\right) - \left(\frac{1}{9}\sum_{i=m-1}^{m+1}\sum_{j=n-1}^{n+1}\mathbf{y}(i,j)\right)^2. \tag{5}$$

The polarities based on the variances can be decided with a pre-determined threshold value by

$$\mathbf{P} = \bigcup_{m=0}^{\frac{M}{M_W}-1}\bigcup_{n=0}^{\frac{N}{N_W}-1}\left\{P(m,n)\right\}, \tag{6}$$

where

$$P(m,n) = \begin{cases} 1, & \text{if } \sigma^2(m,n) \geq \text{ threshold;} \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

We set the threshold to be half of the codebook size for convenience. Then, we are able to generate the secret key with the exclusive-or operation

$$\text{key}_2 = \mathbf{W}_P \oplus \mathbf{P}. \tag{8}$$

After the inverse-VQ operation, both the reconstructed image, $\mathbf{X}'$, and the secret key, key_2, work together to protect the ownership of the original image.

From another point of view, the proposed algorithm is efficient for implementing with the conventional VQ compression algorithms. Once the codeword of each block is determined, we have the polarity of each block; consequently, we get the secret key. Both $\mathbf{X}'$ and key_2 are transmitted to the receiver.

In extracting the watermarks, we calculate the estimated polarities $\mathbf{P}'$ from $\mathbf{X}'$ first, and then have the exclusive-or operation with key_2 to get an estimate of the permuted watermark

$$\mathbf{W}_P' = \mathbf{P}' \oplus \text{key}_2. \tag{9}$$

Finally, we can perform the inverse operation of the permuted watermark to acquire the extracted one, $\mathbf{W}'$.

3.2 Simulation results

In our simulation, we take the test image, Lena, with size 512×512, as the original source, shown in Fig. 2(a). We have the embedded watermark, rose, with size 128×128. The original source is divided into 4×4 block for VQ compression, which also meets the number of bits to be embedded in the watermark. The watermarked reconstruction of the VQ compressed Lena image with LBG algorithm [8] is 31.53 dB with the codebook size 512, and it is shown in Fig. 2(b). We employ the normalized cross-correlation, NC, for evaluating the effectiveness of our algorithm [3]. The NC between the embedded watermark, $W(i,j)$, and the extracted one, $\tilde{W}(i,j)$, is

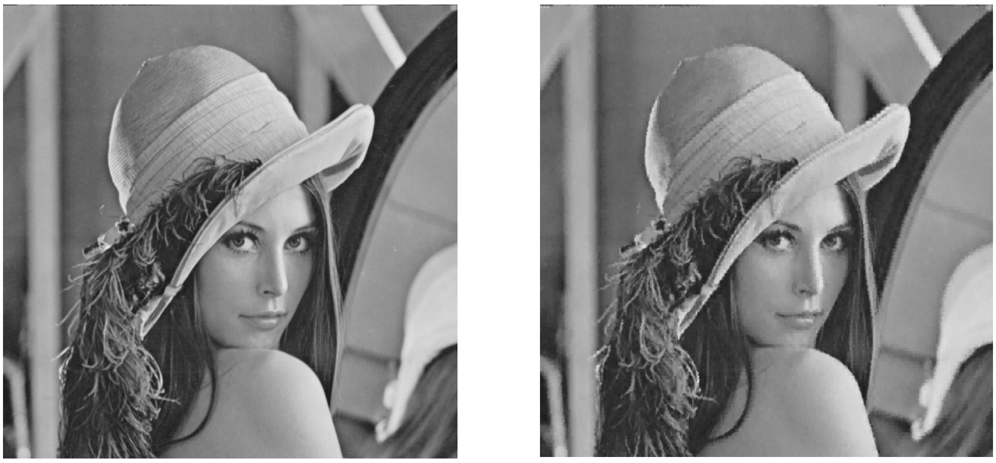
$$\mathrm{NC} = \frac{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}\left[W(i,j)\cdot\tilde{W}(i,j)\right]}{\sum_{i=1}^{M_W}\sum_{j=1}^{N_W}\left[W(i,j)\right]^2}. \qquad \textbf{(10)}$$

We can see that $\mathrm{NC}\in[0,\ 1]$. If we acquire the higher the NC value, there is the more similarity between the embedded watermark and the extracted one. In addition to simulating our algorithm, we also make comparisons with the techniques in [5] under different attacking methods to show the superiority and usefulness of our method.

The simulation results with our algorithm are depicted in Fig. 3, and their corresponding outcomes with the methods in [5] are displayed in Fig. 4, respectively. The extracted watermarks when no attacks applied are shown in Fig. 3(a) and Fig. 4(a), and their counterparts with a variety of attacking methods are illustrated in Fig. 3(b)-(h) and Fig. 4(b)-(h), correspondingly.

Fig. 3(a) and Fig. 4(a) show the extracted watermark when no attacking is applied. Both the NC values are 1.0, and this means that both the algorithms are able to extract the embedded watermarks perfectly because the embedded watermarks and the extracted ones are identical. Fig. 3(b)-(d) and Fig. 4(b)-(d) demonstrate the results under VQ attacks. Assume that there are three codebooks to be trained in advance. Codebook 1 is trained from Lena with size 512, Codebook 2 is obtained by Pepper with size 512, and Codebook 3 is acquired from both Pepper and Baboon with size 256. With the results shown, we are certain that our algorithm is robust under VQ attacks; in contrast, the methods in [5] failed to pass the VQ attacks with various codebooks.

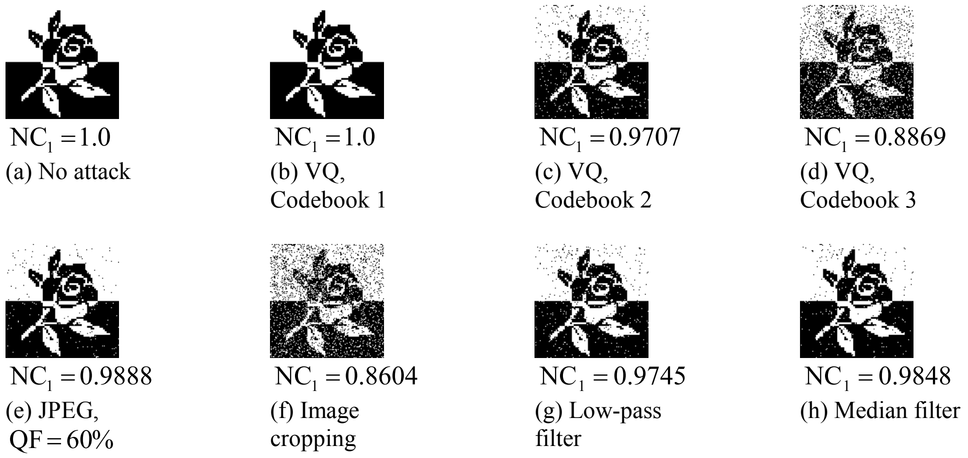
We had other attacking methods, including JPEG compression with different quality factors (QF), image cropping, low-pass filtering and median-filtering techniques [11], on the watermarked image. The extracted watermarks and the NC values are depicted in Fig. 3(e)-(h) and Fig. 4(e)-(h), respectively. Among the extracted watermarks in our algorithm, the ones after JPEG with $\mathrm{QF} = 60\%$, low-pass filtering and median filtering attacks successfully survived because $\mathrm{NC} \rightarrow 1.0$. The NC values in [5] are not as high to compare with our algorithm, because we adopt the information of the neighboring blocks to embed the watermark in order to resist the intentional attacks. Finally, although the NC in the image cropping case is somewhat smaller in our algorithm, the information conveyed therein is still recognizable. Therefore, we are able to claim the robustness and effectiveness of the proposed algorithm and its superiority to compare with the methods in [5].



(a)
Original Lena

(b)
VQ compressed Lena, 31.53 dB

Fig. 2. The test image and VQ compressed one.



$\mathrm{NC}_1 = 1.0$ (a) No attack

$\mathrm{NC}_1 = 1.0$ (b) VQ, Codebook 1

$\mathrm{NC}_1 = 0.9707$ (c) VQ, Codebook 2

$\mathrm{NC}_1 = 0.8869$ (d) VQ, Codebook 3

$\mathrm{NC}_1 = 0.9888$ (e) JPEG, $\mathrm{QF} = 60\%$

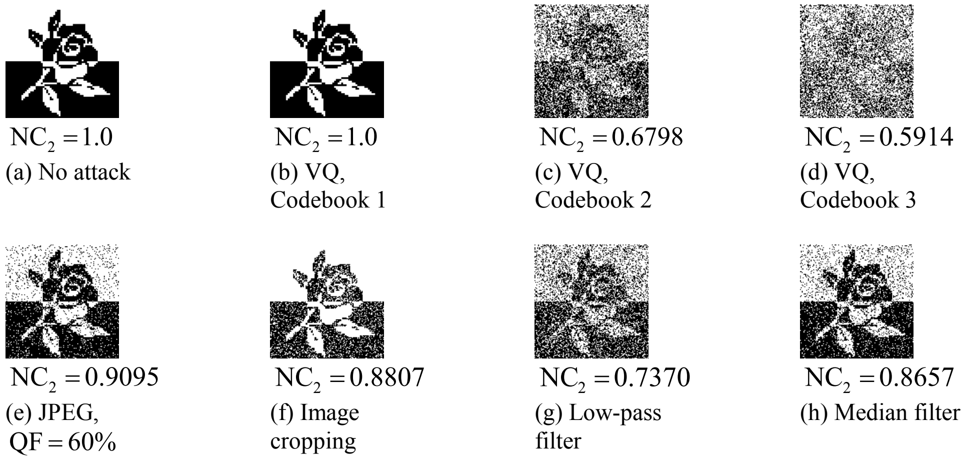
$\mathrm{NC}_1 = 0.8604$ (f) Image cropping

$\mathrm{NC}_1 = 0.9745$ (g) Low-pass filter

$\mathrm{NC}_1 = 0.9848$ (h) Median filter

Fig. 3. The extracted watermarks with size 128×128 and the NC values of the proposed algorithm under various attacking methods.

To sum up, except for the image cropping case, the rest of the extracted watermarks in the algorithm stated in this section have higher NC values. Generally speaking, after experiencing the intentional attacks, our algorithm has better chances to survive. The techniques in [5] failed to pass some of the attacks. This means the effectiveness of the proposed algorithm.



$\mathrm{NC}_2 = 1.0$ (a) No attack

$\mathrm{NC}_2 = 1.0$ (b) VQ, Codebook 1

$\mathrm{NC}_2 = 0.6798$ (c) VQ, Codebook 2

$\mathrm{NC}_2 = 0.5914$ (d) VQ, Codebook 3

$\mathrm{NC}_2 = 0.9095$ (e) JPEG, $\mathrm{QF} = 60\%$

$\mathrm{NC}_2 = 0.8807$ (f) Image cropping

$\mathrm{NC}_2 = 0.7370$ (g) Low-pass filter

$\mathrm{NC}_2 = 0.8657$ (h) Median filter

Fig. 4. The extracted watermarks with size 128×128 and the NC values with the methods in [5] to compare with Fig. 3.

4. Watermarking with VQ Block Means and Variances

We present the multiple-watermark-embedding algorithm for VQ-based binary watermarking with both the means and the variances of blocks.

4.1 The algorithm

Let the input image be $\mathbf{X}$ with size $M \times N$. The goals for performing the algorithm in this section are to embed the robust watermarks with VQ into $\mathbf{X}$, and to output a watermarked reconstruction and the secret keys associated with the embedded watermarks. These secret keys are registered to the third party to certify the ownership of our original multimedia source.

Assume that the three binary-valued watermarks to be embedded are $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{W}_3$, all with sizes $M_W \times N_W$. We perform the VQ operation first [8] and get the codewords with the nearest search algorithm. Afterwards, we are able to embed the watermarks with the characteristics of the indices in the VQ domain. We also perform the pseudo-random number traversing method in this section to permute the watermark to disperse its spatial-domain relationships. With a pre-determined key, key_0 , in the pseudo-random number generating system [9], we have

$$\mathbf{W}_{P,i} = \text{permute}\left(\mathbf{W}_i, \text{key}_0\right), \; i = 1, \; 2, \; 3, \tag{11}$$

then we use the permuted version $\mathbf{W}_{P,i}$ for embedding into the VQ indices.

For embedding the three watermarks into one original image in this section, we need to further extend the use of polarity $\mathbf{P}$, stated in the previous section, with the block-mean, block-variance, and both of them.

Calculating the mean of $\mathbf{y}(m,n)$ and the indices of its surrounding blocks with

$$\mu(m,n) = \frac{1}{9} \sum_{i=m-1}^{m+1} \sum_{j=n-1}^{n+1} \mathbf{y}(i,j). \tag{12}$$

Similarly, obtaining the variance of $\mathbf{y}(m,n)$ and the indices of its neighboring blocks with Eq. (5), also presented here,

$$\sigma^2(m,n) = \left(\frac{1}{9} \sum_{i=m-1}^{m+1} \sum_{j=n-1}^{n+1} \mathbf{y}^2(i,j) \right) - \mu^2(m,n). \tag{13}$$

The polarities based on the means, variances, and both the means and the variances, can be decided with the predetermined threshold value by

$$\mathbf{P}_i = \bigcup_{m=0}^{\frac{M}{M_W}-1} \bigcup_{n=0}^{\frac{N}{N_W}-1} \left\{ P_i(m,n) \right\}, \; i = 1, \; 2, \; 3, \tag{14}$$

Where

$$P_1(m,n) = \begin{cases} 1, & \text{if } \mu(m,n) \geq \text{ threshold;} \\ 0, & \text{otherwise;} \end{cases} \tag{15}$$

$$P_2(m,n) = \begin{cases} 1, & \text{if } \sigma^2(m,n) \geq \text{ threshold;} \\ 0, & \text{otherwise;} \end{cases} \tag{16}$$

And

$$P_3(m,n) = P_1(m,n) \oplus P_2(m,n). \tag{17}$$

We take the threshold $= \frac{L}{2}$ for convenience, and L = codebook size as shown in Eq. (2). Then, we are able to embed $\mathbf{W}_{P,i}$ with $\mathbf{P}_i$ by the exclusive-or operation

$$\text{key}_i = \mathbf{W}_{P,i} \oplus \mathbf{P}_i, \; i = 1, \; 2, \; 3. \tag{18}$$

After the inverse-VQ operation, the reconstructed image, $\mathbf{X}'$, along with the three secret keys, $\text{key}_i, \; i = 1, \; 2, \; 3$, work together to protect the ownership of the original image because the secret keys are registered to the third party to protect the ownership in advance. The image quality of $\mathbf{X}'$ is good because only the error by the VQ is introduced, and it would not be influenced by the information conveyed in the watermarks because of hiding the information into the secret keys.

From another point of view, the proposed algorithm is efficient for implementation with the conventional VQ compression algorithms. Once the codeword of each block is chosen, we are able to determine the polarities of each block; consequently, we get the secret keys. All $\mathbf{X}'$, key_1 , key_2 , and key_3 are transmitted to the receiver.

The block diagram for embedding several watermarks is depicted in Fig. 5.

In extracting the watermarks, we calculate the estimated polarities $\mathbf{P}'_i$ from the means and the variances of $\mathbf{X}'$ first, and then have the exclusive-or operation with key_i to get the estimates of the permuted watermark

$$\mathbf{W}'_{P,i} = \mathbf{P}'_i \oplus \text{key}_i, \; i = 1, \; 2, \; 3. \tag{19}$$

Finally, we can perform the inverse operation of Eq. (9) to get the extracted watermark

$$\mathbf{W}'_i = \text{inverse\_permute}\left(\mathbf{W}'_{P,i}, \text{key}_0\right), \; i = 1, \; 2, \; 3. \tag{20}$$

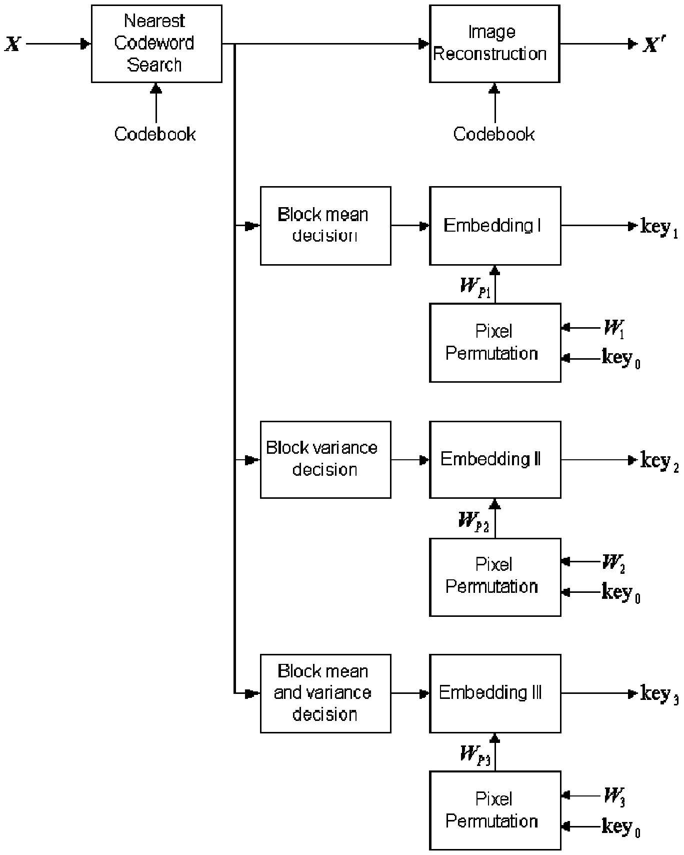


Fig. 5. The block diagram for embedding multiple watermarks.

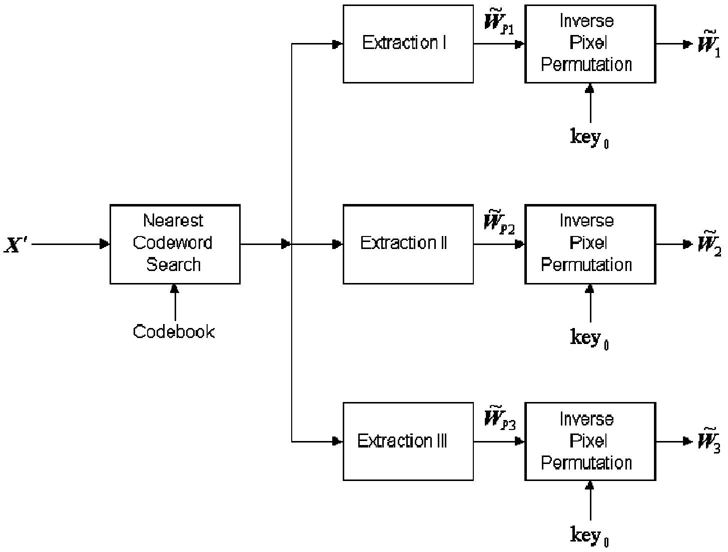


Fig. 6. The block diagram for extracting multiple watermarks.

One advantage for our extracting algorithm is that we can extract the three embedded watermarks in parallel. In addition, owing to the fact that the generated secret keys during the watermark embedding process were registered to the third party in advance, we could prevent the attackers from inserting the counterfeit watermarks to confuse our ownership. The block diagram for the extraction procedure is shown in Fig. 6.

4.2 Simulation results

In our simulation, we take the test image, Lena, with size 512×512, as the original source, and it is shown in Fig. 2(a). The three embedded watermarks, word1, word2, and rose, all having size 128×128, are illustrated in Fig. 7.

The original source is divided into 4×4 block for VQ compression, which also meets the number of bits to be embedded in the watermark. The watermarked reconstruction of the VQ compressed Lena image with LBG algorithm [8] is 31.53 dB with the codebook size 512, also shown in Fig. 2(b). We employ the normalized cross-correlation, NC, demonstrated in Eq. (10), for evaluating the effectiveness of our algorithm [3].

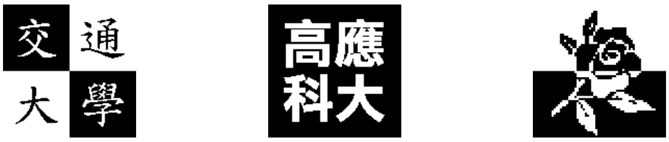
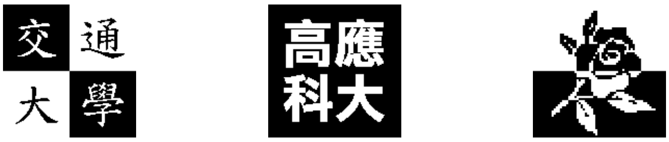

(a) (b) (c)

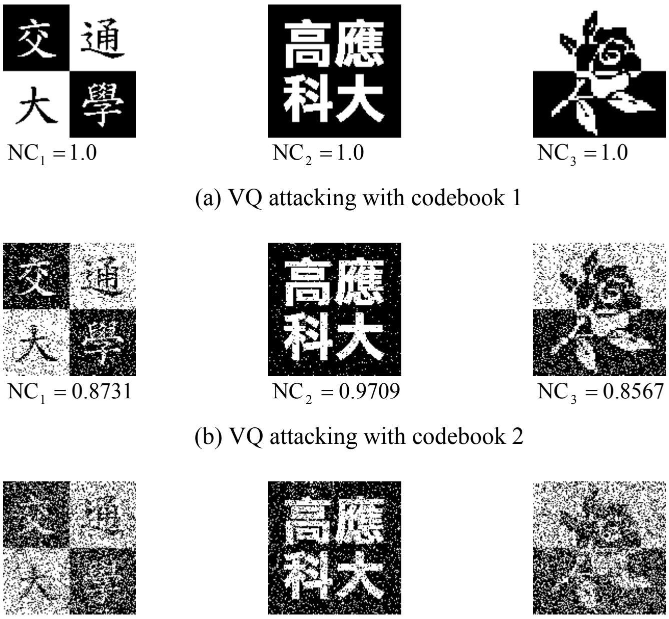
Fig. 7. The three embedded watermarks.

The simulation results with our algorithm are depicted in this section. The extracted watermarks when no attacks applied are shown in Fig. 8. The watermarked image also experiences a variety of attacks, including VQ attack by employing a different codebook other than the original one, the JPEG attack [10], filtering attack [11], and geometric attack, that is, the rotation attack offered by Stirmark [12]. Extracted watermarks after experiencing these attacks are illustrated in Fig. 9 to Fig. 12, respectively.


$\mathrm{NC}_1 = 1.0$ $\mathrm{NC}_2 = 1.0$ $\mathrm{NC}_3 = 1.0$

No attacks applied

Fig. 8. The extracted watermarks when no attacks applied.


$\mathrm{NC}_1 = 1.0$ $\mathrm{NC}_2 = 1.0$ $\mathrm{NC}_3 = 1.0$

(a) VQ attacking with codebook 1

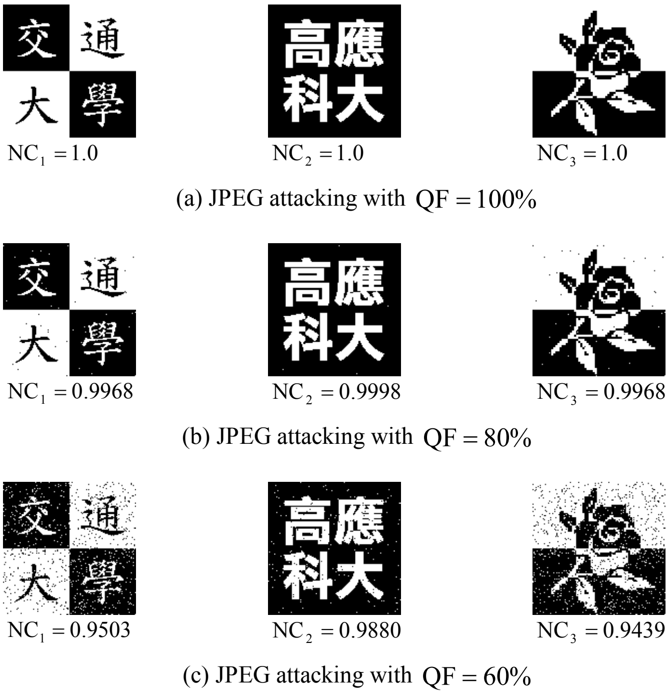
$\mathrm{NC}_1 = 0.8731$ $\mathrm{NC}_2 = 0.9709$ $\mathrm{NC}_3 = 0.8567$

(b) VQ attacking with codebook 2

$\mathrm{NC}_1 = 0.7554$ $\mathrm{NC}_2 = 0.8869$ $\mathrm{NC}_3 = 0.7177$

(c) VQ attacking with codebook 3

Fig. 9. The extracted watermarks under VQ attacking techniques.



$NC_1 = 1.0$ $NC_2 = 1.0$ $NC_3 = 1.0$

(a) JPEG attacking with $QF = 100\%$

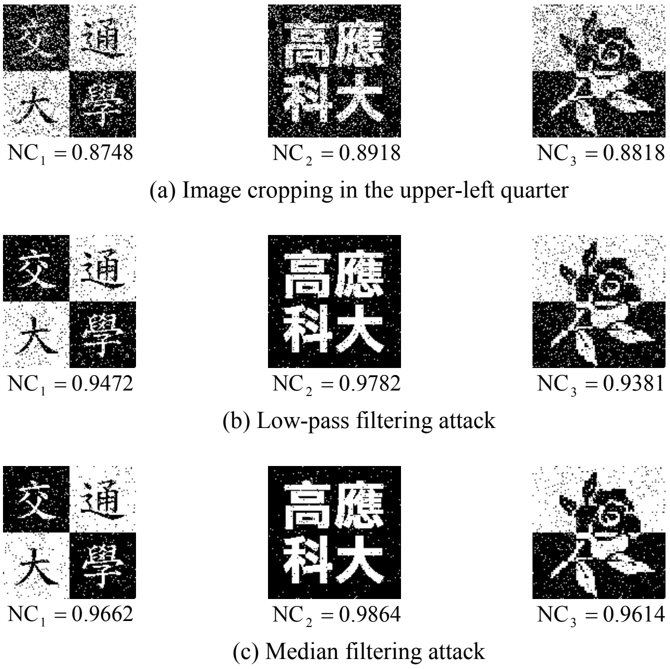
$NC_1 = 0.9968$ $NC_2 = 0.9998$ $NC_3 = 0.9968$

(b) JPEG attacking with $QF = 80\%$

$NC_1 = 0.9503$ $NC_2 = 0.9880$ $NC_3 = 0.9439$

(c) JPEG attacking with $QF = 60\%$

Fig. 10. The extracted watermarks under JPEG compression.



$NC_1 = 0.8748$ $NC_2 = 0.8918$ $NC_3 = 0.8818$

(a) Image cropping in the upper-left quarter

$NC_1 = 0.9472$ $NC_2 = 0.9782$ $NC_3 = 0.9381$

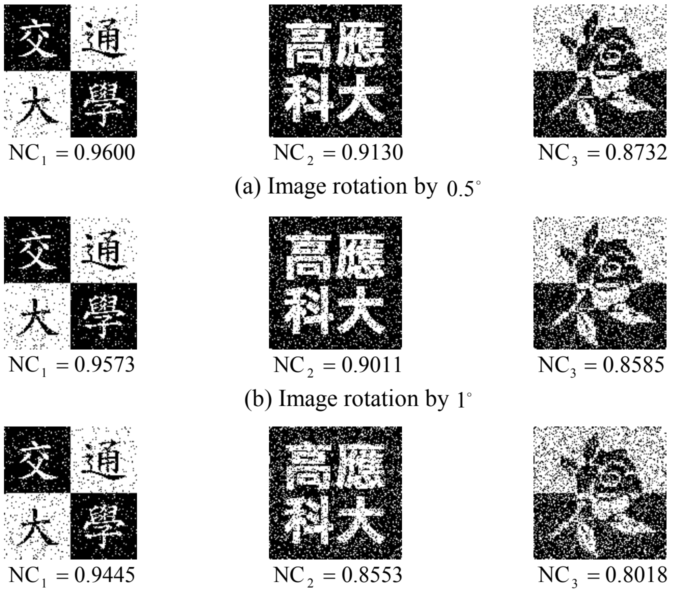
(b) Low-pass filtering attack

$NC_1 = 0.9662$ $NC_2 = 0.9864$ $NC_3 = 0.9614$

(c) Median filtering attack

Fig. 11. The extracted watermarks under the attacking methods in the spatial domain.

Summing up, with the algorithm offered in this section, we can obtain the robust, VQ-based multiple watermarking scheme after imposing a variety kind of attacks. On the one hand, in the category of destruction attacks, except for the VQ attacking case with some other codebooks, the rest of the results in our algorithm have higher NC values. On the other hand, with the synchronization attack such as the geometric attack, the extracted watermarks survive in our simulation both subjectively and objectively. From the data simulated, after experiencing the intentional or unintentional attacks, our algorithm has better chances to survive because of the increase in watermark capacity, and all the three watermarks could survive under the destruction and synchronization attacks.



$NC_1 = 0.9600$ $NC_2 = 0.9130$ $NC_3 = 0.8732$

(a) Image rotation by 0.5°

$NC_1 = 0.9573$ $NC_2 = 0.9011$ $NC_3 = 0.8585$

(b) Image rotation by 1°

$NC_1 = 0.9445$ $NC_2 = 0.8553$ $NC_3 = 0.8018$

(c) Image rotation by 2°

Fig. 12. The extracted watermarks under the geometric attacks.

5. Watermark Embedding with Grey-Level Watermarks

We proposed the procedures for embedding and extracting the gray-level watermark in this section. Fig. 13 and Fig. 14 also depict the flow charts for the embedding and extraction processes.

5.1 The algorithm

Given a codebook $\mathbf{C}$ with L codewords, shown in Eq. (2), a sub-codebook $\mathbf{C}_\mathrm{S}$ is chosen from some codewords in $\mathbf{C}$ first, by referring to one user-key $\mathbf{S}_1$. Hence, $\mathbf{C}_\mathrm{S} \in \mathbf{C}$.

Next, $\mathbf{C}$ is partitioned into p sub-codebooks $\left\{ \mathbf{C}_0, \mathbf{C}_1, ..., \mathbf{C}_{p-1} \right\}$ by referring to another user-key s_2. Here $\mathbf{C}_0 \cup \mathbf{C}_1 \cup ... \cup \mathbf{C}_{p-1} = \mathbf{C}$, $\mathbf{C}_0 \cap \mathbf{C}_1 \cap ... \cap \mathbf{C}_{p-1} = \phi$, and $\mathbf{S}_2$ defines allocations of the codewords in $\mathbf{C}$ to the p sub-codebooks.

In our watermarking scheme, C_S is employed to encode the gray watermark $\mathbf{W}$. In splitting the codebook into $\left\{ \mathbf{C}_0, \mathbf{C}_1, ..., \mathbf{C}_{p-1} \right\}$, we suggest using $p = 2^m$ to hide m watermark bits into each VQ index.

For a gray watermark $\mathbf{W}$, we employ the traditional VQ encoding procedure with codebook $\mathbf{C}_\mathrm{S}$, and we collect the VQ indices as $\mathbf{I}_\mathrm{W}$. In order to embed $\mathbf{I}_\mathrm{W}$ into the host image $\mathbf{X}$ in the VQ-domain, $\mathbf{I}_\mathrm{W}$ has to be split into T parts. Here T is equal to the number of non-overlapping blocks consisting of the host image. The following steps illustrate the details of how to split $\mathbf{I}_\mathrm{W}$ into the considered number of parts.

Step (i) Convert $\mathbf{I}_\mathrm{W}$ into a binary bitstream $\mathbf{B}$.

Step (ii) Decompose $\mathbf{B}$ into T parts, where the length of each part is m bits. We use $\mathbf{B} = \left\{ \mathbf{b}_0, \mathbf{b}_1, \cdots, \mathbf{b}_{T-1} \right\}$ to denote the decomposed result.

Step (iii) Translate each binary element in $\mathbf{B}$ into a decimal integer, and present the collection of integers by J . The rule is according to their translations from binary formats to decimal formats. We denote $J = \left\{ j_0, j_1, \cdots, j_{T-1} \right\}$, where $0 \le j_i < p$, $0 \le i < T$, to represent the translated result. For example, in the case of $m = 2$, if $\mathbf{b}_i = \{1, 0\}$, then the translated result is $j_i = 2$; in the case of $m = 3$, if $\mathbf{b}_i = \{1, 1, 0\}$, then the translated result is $j_i = 6$.

The host image $\mathbf{X}$ is decomposed into T vectors $\{\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_{T-1}\}$ with dimension k. Then, by employing the encoded result J, the following steps illustrate how to hide m bits into each of the vectors.

Step (i) Refer to the i th integer $j_i = 2$ of J, the j_i th sub-codebook $\mathbf{C}_{j_i}$ is selected from $\{\mathbf{C}_0, \mathbf{C}_1, ..., \mathbf{C}_{p-1}\}$ as the default codebook for the VQ encoding procedure.

Step (ii) Find the nearest codeword from $\mathbf{C}_{j_i}$ and use it to replace $\mathbf{x}_i$.

After dealing with all the vectors, the gray watermark can be embedded into the host image. Figure 13 illustrates the flow chart of the embedding procedures.

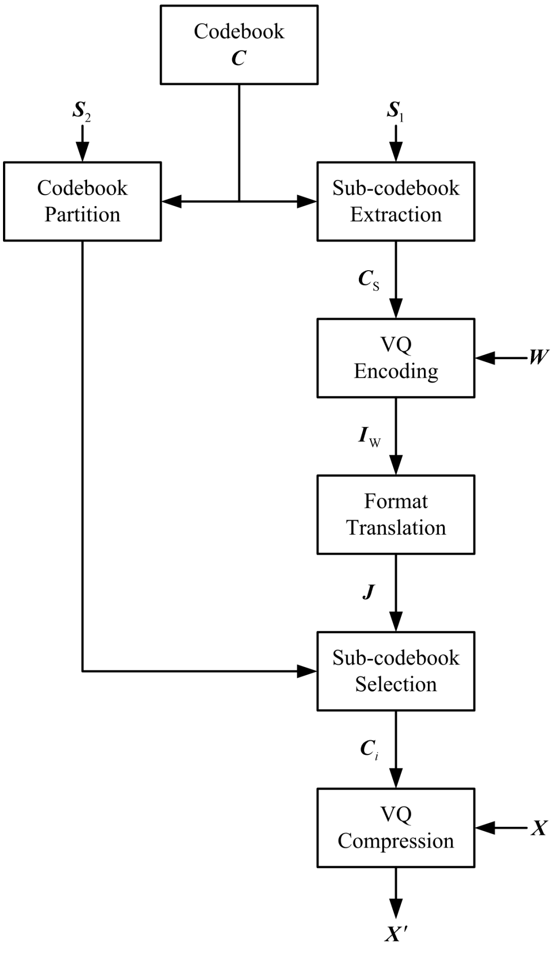


Fig. 13. Watermark embedding procedures.

In the decoder side, the received image $\mathbf{X}'$, which might be attacked during transmission, is divided into T vectors $\{\mathbf{x}'_0, \mathbf{x}'_1, ..., \mathbf{x}'_{T-1}\}$ with dimension k. For the i th vector $\mathbf{x}'_i$, $0 \le i < T$, the steps below are employed to determine the embedded bits.

Step (i) Execute the VQ encoding procedure to find a nearest codeword from codebook $\mathbf{C}$ for $\mathbf{x}'_i$.

Step (ii) Refer to $\mathbf{S}_2$, find the sub-codebook $\mathbf{C}_{j_i}$ where the obtained codeword is belonging to.

Step(iii) Convert the index j_i of $\mathbf{C}_{j_i}$ into a binary stream $\mathbf{b}_i$ with length m; e.g., if $m = 4$ and $j_i = 5$, then $\mathbf{b}_i = \{0, 1, 0, 1\}$.

After obtaining the embedded bits from the watermarked image, the hidden gray watermark can be recovered by reversing the steps above.

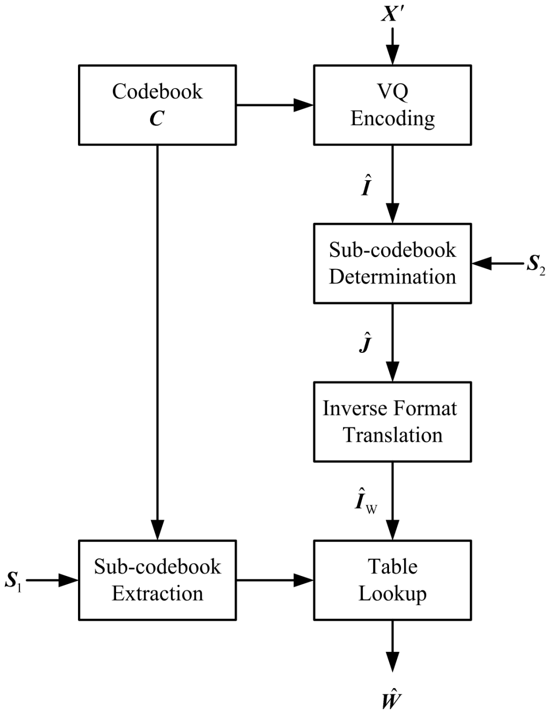


Fig. 14. Watermark extraction procedures.

5.2 Simulation results

In our experiments, the host image is 8-bit per pixel gray image Lena, with size 512×512. The image in Figure 15(a) represents the 8-bit per pixel, gray level watermark, with size 256×256. The codebook of size 256 was trained from Lena with the LBG algorithm [8]. We divide the host image and the watermark into 4×4 blocks, and construct 16384 and 4096 vectors, respectively. The indices obtained from the watermark are converted into a binary bitstream, which is then split into 16384 parts with dimension $m = 2$ bits. To describe the performance, the peak-signal-to-noise ratio (PSNR) is employed. The PSNR value between the host image and the watermarked one is 28.83 dB, and the PSNR value between the embedded watermark and the extracted one under no attack is 32.04 dB. Figure 15 depicts the original watermark and the watermarks extracted from the attacked images. Table I summarizes the capacity of the VQ-based watermarking methods in [13]-0 in addition to the watermarked image quality. It is reasonable to see that if we embed more bits into the host image, the PSNR values in the watermarked images become lower. The results demonstrate the proposed scheme has superior capacity, and it can survive under the VQ compression and the JPEG compression with different quality factors (QF) successfully.

6. Conclusions

Efficient and robust algorithms for VQ-based watermarking have been presented in this paper. We described the schemes for embedding both the binary watermarks and the grey-level ones in the VQ domain. On the one hand, it is efficient because it modifies the VQ indices and makes use of the VQ properties to proceed with the embedding of multiple watermarks, and to hide the information into the secret keys. Besides, the secret keys were registered to the third party in advance to keep other attackers from embedding the counterfeit watermarks. Hence, the watermarked image quality would not be affected by the embedded watermarks. On the other hand, with the simulation results under a variety of attacking techniques, we are able to assert its robustness, effectiveness, and superiority over the existing algorithm. The proposed scheme can also incorporate with the encryption algorithm for further protection. Further work will concentrate on embedding multiple watermarks into the same original source in the VQ and transform domains to protect the original source more effectively.

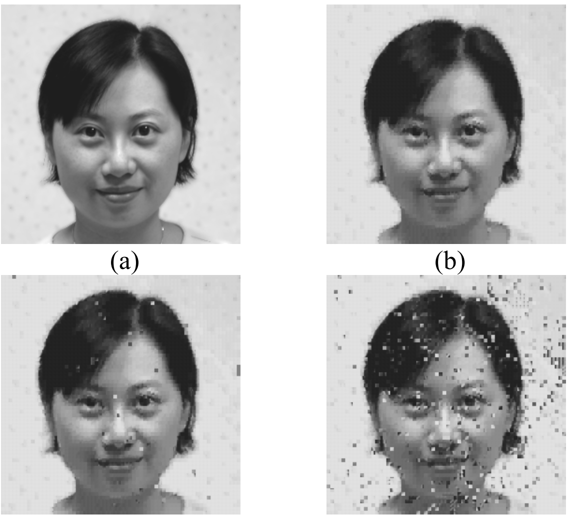

(a) (b) (c) (d)

Fig. 15. The original watermark and the extracted ones (256×256 pixels, 8-bit per pixel gray-scale). (a) Original watermark. (b) Extracted watermark under VQ compression with codebook size 256, PSNR = 32.04 dB. (c) Extracted watermark under JPEG compression with QF = 80%, PSNR = 29.36 dB. (d) Extracted watermark under JPEG compression with QF = 60%, PSNR = 20.00 dB.

Table 1. Comparison of watermark capacity (in bits) and PSNR (in dB) of the watermarked image among different methods in literature.

Methods	Watermark capacity	PSNR
Method In [13]	33199 bits	30.59 dB
Method in [14]	16384 bits 128×128; 1 bit/pixel	31.53 dB
Method in [15]	$\leq 128\times128$ bits	30.99 dB
Proposed	524288 bits 256×256; 8 bits/pixel	28.83 dB

7. Acknowledgement

The authors would like to give their gratitude to Dr. Feng-Hsing Wang for the support of system implementation.

References

[1] R. Barnett, “Digital watermarking: applications, techniques, and challenges,” *IEE Electronics & Communication Engineering Journal*, Vol. 11, No. 4, 1999, pp. 173 – 183.

[2] S. Katzenbeisser and F. Petitcolas, *Information Hiding – Techniques for Steganography and Digital Watermarking*, Artech House, Norwood, 2000.

[3] C. T. Hsu and J. L. Wu, “Hidden Digital Watermarks in Images,” *IEEE Trans. Image Process*., Vol. 8, No. 2, 1999, pp. 58–68.

[4] J. J. K. O’Ruanaidh, W. J. Dowling, and F. M. Boland, “Phase watermarking of digital images,” in *Proc. IEEE Int. Conf. Image Processing*, Vol. 3, Laussane, September 1996, pp. 239–242.

[5] Z. M. Lu and S. H. Sun, “Digital image watermarking technique based on vector quantisation,” *Electron. Lett.*, Vol. 36, No. 4, 2000, pp. 303-305.

[6] K. Sayood, *Introduction to Data Compression*, 2nd Ed., Morgan Kaufmann: San Francisco, CA.

[7] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers: Boston, MA.

[8] Y. Linde, A. Buzo, and R. M. Gray, “An algorithm for vector quantizer design,” *IEEE Trans. Commun*., Vol. 28, No. 1, 1980, pp. 84-95.

[9] J. G. Proakis, *Digital Communications*, McGraw-Hill, New York, 1995.

[10] W. B. Pennebaker, and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[11] R. C. Gonzales and R. E. Woods, *Digital Image Processing*, Addison-Wesley, Reading, Massachusetts, 1992.

[12] F. A. P. Petitcolas and M. G. Kuhn, *StirMark*, http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/, 2004.

[13] Z. M. Lu, J. S. Pan, and S. H. Sun, “VQ-based digital image watermarking method,” *Electron. Lett*., Vol. 36, No. 14, Jul. 2000, pp.1201 – 1202.

[14] H.-C. Huang, F. H. Wang, and J. S. Pan, “Efficient and robust watermarking algorithm with vector quantisation,” *Electron. Lett.*, Vol. 37, No 13, Jun. 2001, pp. 826 – 828.

[15] M. Jo and H. D. Kim, “A digital image watermarking scheme based on vector quantisation,” *IEICE Trans. Inf. & Syst*., Vol. E85-D, 2002, pp.1054 – 1056.