Signal Processing 108 (2015) 278–287

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Reversible data hiding with high payload based on referred frequency for VQ compressed codes index

Tai-Yuan Tu, Chih-Hung Wang*

Department of Computer Science and Information Engineering, National Chiayi University, Chiayi City 60004, Taiwan

ARTICLE INFO

Article history:
Received 8 May 2014
Received in revised form
15 September 2014
Accepted 16 September 2014
Available online 28 September 2014

Keywords:
Reversible data hiding
Vector quantization
Image compression
Steganography

ABSTRACT

In 2007, Chang et al. proposed a reversible data hiding scheme for secret communication that hides secret information in the compression codes of a cover image, but their scheme has low hiding capacity and introduces extra m bits to reverse the original vector quantization (VQ) index of the cover image after the secret data are extracted. Instead of introducing m bits (where the size of the codebook is m bits) and using only one-third of the VQ indices of the cover image to hide the secret bit, we propose using only one bit to distinguish between indices of two clusters (i.e., cluster_2 and cluster_3). Not only the indices in cluster_1 but also those in cluster_2 and cluster_3 can hide the secret bits. Our proposed scheme reduces the number of extra bits and increases the hiding capacity. The experiment results clearly showed that our proposed scheme outperforms Chang et al.'s hiding scheme.

1. Introduction

With the rapid development of the Internet and telecommunication techniques, people frequently need to send, receive, or store private information. Furthermore, the publishing and broadcasting industries have rapidly grown in recent years. With the advent of digital formats, high-quality data can be provided even when copied multiple times. This has led to more information being transmitted via the Internet than ever. Network security is becoming increasingly important with the amount of data being exchanged over the Internet. The copyright marks and serial numbers of digital media such as digital films, audio recordings, books, and multimedia products can lead to large-scale unauthorized copying. Therefore, confidentiality and data integrity are necessary to protect against illicit access. Researchers have focused on finding a solution to the problem of unauthorized copying. Invisible information can be embedded into digital media so that it cannot be easily extracted without specialized techniques. This has resulted in the explosive growth of information hiding techniques.

Information hiding [2,26,30] is defined as the procedure of embedding important information such as a secret message into cover media. Applications of information hiding techniques include steganography, watermarking, fingerprinting, and copyright marks for digital media. These approaches have many differences; in this paper, we mainly focus on steganography.

Steganography is the art of storing information so that its existence is hidden. The goal of steganography is to communicate securely in a completely undetectable manner and to hide data well enough that unintended recipients do not suspect that the steganographic medium contains hidden data. Therefore, the required quality of the stego-image needs to be high for the embedded secret message; the resulting image must not be degraded so much that there is any perceivable difference between the stego-image and the original cover image. Digital images are ideal for hiding secret

* Corresponding author. Tel.: +886 5 2717736.
E-mail address: wangch@mail.ncyu.edu.tw (C.-H. Wang).

http://dx.doi.org/10.1016/j.sigpro.2014.09.021
0165-1684/

information. In the information hiding process, the sender first embeds the secret message into a digital image called a cover image, and the cover image containing the secret message becomes the stego-image. Then, the sender sends the stego-image to the receiver. The receiver can extract the secret message from the stego-image.

In the past decades, many approaches have been proposed for secure communications via the Internet. Most hiding methods directly embed the secret data into the pixels of a cover image to generate the stego-image [11,16,20,28]. For example, Wu et al. [27] proposed a secret image sharing scheme by applying optimal pixel adjustment process to enhance the image quality under different payload capacity and various authentication bits conditions. Chang et al. [3] proposed a scheme to hide secret data in the least significant bit (LSB) of image pixels by using a dynamic programming strategy. Chan and Cheng [1] proposed a simple LSB substitution-based hiding technique, and Wang et al. [24] proposed an image-hiding method based on optimal LSB substitution and a genetic algorithm.

In order to speed up the transmission time over the Internet and reduce bandwidth usage, data compression is commonly used to reduce the amounts of data traveling over a communication network. Several widely accepted compression methods are vector quantization (VQ) [12], discrete wavelet transformation (DWT) [23], and discrete cosine transformation (DCT) [10]. One of the most common compression algorithms is VQ, which is an attractive option because of its simplicity and cost-effectiveness. Thus, some hiding techniques have proposed using it to hide secret data into the compression code of host multimedia [4,6,9,12,15,17,22,29]. For example, Lu and Sun [19] proposed an image watermarking technique based on VQ, and Jo and Kim [14] designed a VQ-based spreading image watermarking scheme. Both schemes degrade the image quality after the secret data are embedded into the host image. They are irreversible, which means that the original cover image cannot be completely recovered after the secret information is extracted. Hence, several researchers have concentrated on studying reversible or lossless data hiding methods [5,7,13,21,25].

In 2007, Chang et al. [8] proposed a lossless recovery data embedding method based on VQ. The method first sorts a codebook generated by the Linde–Buzo–Gray (LBG) algorithm in descending order according to the referred frequencies of code words in advance. Then, the sorted codebook is partitioned into three clusters of the same size. The cluster with the highest frequency is used to embed the secret data, and the other two clusters with lower frequencies are used for recovery from the stego-image.

Although Chang et al.'s method is easy to implement, the disadvantage is that the capacity and transmitted bits of the method depend on the distribution of indices in the VQ index table. In particular, when the size of codebook becomes large, the embedding capacity decreases, and the number of codes increases rapidly.

In this paper, we propose a new reversible data hiding scheme based on VQ with high embedding capacity. The proposed method can reduce the number of codes and possesses higher embedding capacity than Chang et al.'s method [8].

The remainder of this paper is organized as follows. Section 2 introduces related works on compression schemes and reviews Chang et al.'s scheme [8]. Section 3 describes the proposed scheme in detail. Experimental results are presented in Section 4. Finally, the conclusions are presented in Section 5.

2. Related works

The basic concept of VQ is described in Section 2.1. We review Chang et al.'s method [8] in Section 2.2.

2.1. VQ algorithm

VQ is a lossy compression technique especially designed for digital images. It can be defined as mapping the function Q from $r \times l$-dimensional Euclidean space $R^{r \times l}$ to a finite subset Y of $R^{r \times l}$; that is, Q: $R^{r \times l} \rightarrow Y$, where $Y = \{y_i | i = 1, 2, \ldots, N\}$ is the codebook of size N and y_i is the ith code word in Y. Fig. 1 shows the VQ encoding and decoding processes.

The VQ procedure consists of three phases: (1) codebook generation, (2) encoding, and (3) decoding. In the codebook generation phase, some grayscale images are initially picked out as the sample training set and then used to create a codebook containing the most representative code words. This constructed codebook is then used by both the encoder and decoder. Because the VQ-compressed image quality is significantly influenced by the quality of the codebook, a suitable algorithm needs to be chosen to generate the codebook. The LBG algorithm by Linde et al. [18] can be applied to those sample training images to generate a representative codebook.

Before the grayscale image is encoded, the original image is first partitioned into non-overlapping blocks of $r \times l$ pixels, so each block can be represented by an $r \times l$-dimensional vector. In the encoding phase, each block of the original image, which is represented by the vector $X = \{x_0, x_1, \ldots, x_{r \times l}\}$, $X \in R^{r \times l}$, is compared with the code words in the codebook to find the closest code word in the codebook. The distance between X and the code word Y_i, $i = 1, 2, \ldots, n$ is determined by the Euclidean distance $d(X, Y_i)$:

$$d(X, Y_i) = \|X - Y_i\| = \sqrt{\sum_{j=0}^{r \times l - 1} \left(x_j - y_{ij}\right)^2},$$

where x_j and y_{ij} are the jth elements of vectors X and Y_i, respectively. When the closest code word Y_i of X is found, index i is used to encode vector X; the original image is represented by indices of these closest code words. Since the number of bits used to represent the index is always smaller than that of vector X, the encoded image is thus compressed.

In the decoding phase, the decoder has the same codebook as the encoder. The decoder has index i as the input and only performs a simple table lookup operation to obtain the decoded code word x_i, which it then uses to reconstruct the input vector X approximately.

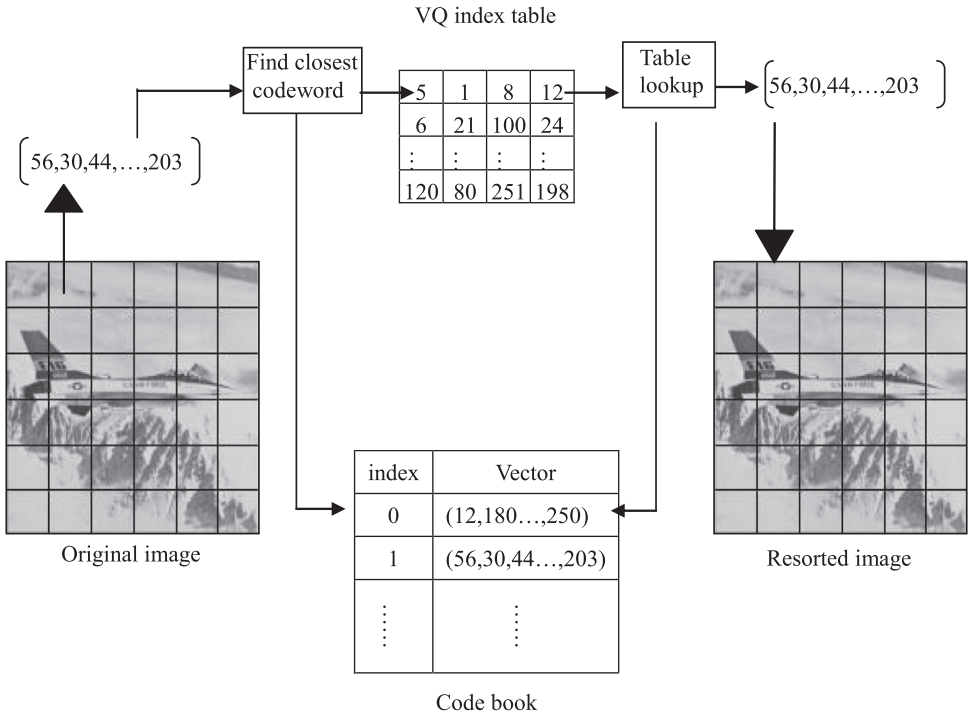

Fig. 1. VQ encoding and decoding process.

2.2. Chang et al.'s method [8]

The embedding method for VQ images proposed by Chang et al. [8] first calculates the referred frequency count of each code word in the original codebook. Then, the original codebook is sorted in descending order according to the referred frequency count of each code word in the codebook.

The front $\lfloor (N-2^{B-1})/(2^{B-1}\times 3)\rfloor \times (2^{B-1}\times 3)$ code words in the sorted codebook of size N code words are extracted to form a new codebook. Therefore, the new codebook is equally partitioned into $2^{B-1}\times 3$ clusters with the same size. The surplus 2^{B-1} clusters not included in the new codebook are used as indicators, where B denotes the size of the secret data hidden into each image block.

As an example, consider the case $B=1$; the codebook with $\lfloor (N-1)/3\rfloor \times 3$ code words is divided into three clusters with the same size. The front k code words $(k=\lfloor (N-1)/3\rfloor)$ with the highest referred count value form the first cluster; the next k code words are in the second cluster, and the rest of the code words of the codebook are in the third cluster. The three code words located individually in three different clusters form a code word trio in the codebook and are replaced with one another. The members of the code word trio have relatively identical positions in the clusters themselves.

During the data embedding phase, the input index has to determine in which cluster the corresponding code word is located. Only the index situated in the first cluster can be used to carry the secret data. Given a code word C_1 in the first cluster, if the one-bit secret s is equal to 1, the code word C_3 in the third cluster is used to replace C_1. If the one-bit secret s is equal to 0, the code word C_2 in the second cluster is used to replace C_1, where C_1, C_2, and C_3 are in the same code word trio. However, this condition will lead to mistakes in the decoding process. For this reason, the original indices of the second and third clusters must be modified. The indices of the two clusters are transformed into indices of the first cluster. That is, both C_2 and C_3 are replaced with C_1. However, this causes confusion when code word C_1 is recovered in the reversion process. In order to recover code word C_1 correctly in the reversion process, an indicator is introduced to mark the transformed indices of code word C_3 in the third cluster. The index value 0, which is called the preserved index, is put in front of the transformed index value C_1.

Although the method is easy to implement, only the front one-third of the indices in the codebook can be employed to hide the secret bit. In extreme cases, if all indices are situated in the first cluster, the number of embedded secret bits is very high, and there is no extra bit. At the other extreme, if all indices are situated in the third cluster, then the number of embedded secret bits is zero. In order to determine whether the corresponding index comes from the second or third cluster, the indicator is added in front of the transformed index of the third cluster. The index is encoded with an additional length of $\lceil \log_2^N \rceil$ – bit. That is, this kind of index cannot be used to hide secret data, but it dramatically increases the number of extra bits and the bits per pixel (bpp). Assuming that the indices are uniformly distributed in the codebook, only one-third of the cover image blocks can hide the secret data. Therefore, the hiding capacity is not high.

3. Proposed algorithm

The hiding capacity of Chang et al.' s method [8] is not high because they use extra bits to act as indicators, which increases the number of transmitted bits and is not cost-efficient. In order to increase the hiding capacity and reduce the number of extra bits, we propose a reversible data hiding method with a high payload that is inspired from Ref. [8]. Next, we describe the proposed algorithm in detail, which consists of three phases: initial; embedding; and extracting and reversion.

3.1. Initial phase

To facilitate the later embedding and extracting phases, some images are first picked up as test sample images in the initial phase. The initial phase algorithm is shown as follows:

Input: Test sample images.

Output: Sorted codebook in descending order.

- Step 1: Each image is divided into non-overlapping image blocks; all are encoded by VQ to determine the closest code word in the codebook.
- Step 2: The referred frequencies of the N code words in the codebook are set to zero.
- Step 3: When the closest code word of one image block in the codebook is determined, the referred frequency of the corresponding code word in the codebook is increased by 1.
- Step 4: Repeat Step 3 until all image blocks of all sample images are processed.
- Step 5: After the referred frequencies of the code words in the VQ codebook are calculated, the original codebook is sorted in descending order according to the referred frequency of each code word in the codebook.
- Step 6: Output sorted codebook in descending order.

The front $\lfloor N/3 \rfloor \times 3$ code words in the sorted codebook of size N code words are extracted to form the new codebook δ'. Therefore, the new codebook is partitioned into three clusters of the same size. The surplus $(N-\lfloor N/3 \rfloor) \times 3$ code words not included in the new codebook δ' are discarded.

An example is demonstrated here, where the codebook δ' with $\lfloor N/3 \rfloor \times 3$ code words is divided into three clusters of the same size. The front k code words ($k=\lfloor N/3 \rfloor$) with the highest referred frequency form the first cluster, which is called cluster$_1$. The next k code words are in cluster$_2$, and the rest of the code words of the codebook δ' are in cluster$_3$. To hide the secret data, we have to establish a mapping function in the code words among the three clusters. The three code words located individually in three different clusters form a code word trio in the codebook δ'. The members of the code word trio have relatively identical positions in the clusters so that they can be replaced with one another. As shown in Fig. 2, the code words C_r of cluster$_1$, C_{r+m} in cluster$_2$, and C_{r+2m} in cluster$_3$ are in the same code word trio.

3.2. Embedding phase

In the embedding phase, m bits of secret data are assumed to be embedded in each index, where $m \geq 1$. Here, we have to determine in which cluster the input index is located. There are two cases that must be considered: there may only be one or more than one secret bits embedded in each index. The necessary steps required for this process are described below.

Case 1. Only one bit is embedded into each index (i.e., $m=1$).

Step 1: First, we have to determine in which cluster the index that carries the secret bit is located. The following conditions need to be considered:

(1) The input index is situated in cluster$_1$.
  (1.1) If the value of the secret bit is equal to 0, the index is directly transformed into the corresponding index in cluster$_2$.
  (1.2) If the value of the secret bit is equal to 1, the index is directly transformed into the corresponding index in cluster$_3$.

Let Γ denote the input index and σ denote the secret bit. The above two conditions can be represented as the following equation:

$$\Gamma \Rightarrow \begin{cases} \text{cluster}_2, & \text{if} \quad (\Gamma \in \text{cluster}_1 \wedge \sigma = 0) \\ \text{cluster}_3, & \text{if} \quad (\Gamma \in \text{cluster}_1 \wedge \sigma = 1) \end{cases}, \tag{1}$$

where the symbol "⇒" means "transformed into the corresponding index".

(2) When the input index is situated in cluster$_2$ or cluster$_3$, the input index is transformed into the

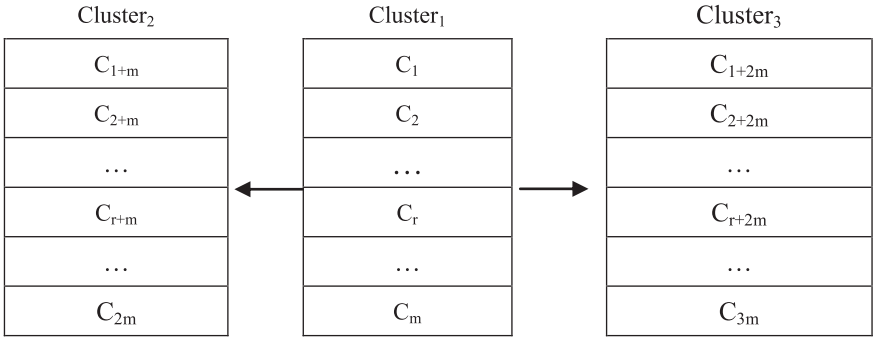

Fig. 2. The codeword-trio of cluster-mapping.

corresponding index in cluster$_1$. An extra one bit I needs to be attached to the rear of the transformed index of the input index in cluster$_1$ as an indicator.

(2.1) If the input index is situated in cluster$_2$, then the value of the indicator bit I is set to 0.

(2.2) If the input index is situated in cluster$_3$, then the value of the indicator bit I is set to 1.

The above two conditions can be represented as

$$I = \begin{cases} 0, & \text{if } (\Gamma \in \text{cluster}_2) \\ 1, & \text{if } (\Gamma \in \text{cluster}_3) \end{cases} \quad (2)$$

Step 2: If the extra indicator bit I exists, then I needs to be attached to the rear of the transformed index of the input index in cluster$_1$ as an indicator. The one secret bit is then appended to the rear of the indicator bit I of the transformed index.

Step 3: Repeat Steps 1 and 2 until all input indices are processed.

Case 2. More than one bit (i.e., $m > 1$) is embedded into each index.

Step 1: Similar to the Case 1, we have to determine in which cluster the index that carries the secret bit is located according to the following conditions:

(1) The input index is situated in cluster$_1$.
  (1.1) If the first bit of the m secret bits is 0, the index is transformed into the corresponding index in cluster$_2$.
  (1.2) Otherwise, if the first bit of the m secret bits is 1, the index is transformed into the corresponding index in cluster$_3$.

(2) When the input index is situated in cluster$_2$ or cluster$_3$, the input index is first transformed into the corresponding index in cluster$_1$.
  (2.1) If the corresponding index comes from cluster$_2$, the indicator bit I is set to 0.
  (2.2) If the corresponding index comes from cluster$_3$, the indicator bit I is set to 1.

Step 2: If the extra indicator bit I exists, then the I plus m secret bits (i.e., $m+1$ bits) are appended to the rear of the corresponding index in cluster$_1$; otherwise, only $m-1$ secret bits (removing the first bit of the secret bits) are directly appended to the rear of the transformed index.

Step 3: Repeat Steps 1 and 2 until all input indices are processed.

As an illustrative example, suppose that the original codebook contains 16 code words (i.e., $N=16$) and that the sorted codebook δ' consists of 15 code words. The secret bit stream is $(100111011001)_2$. Two bits can be embedded into each index of the VQ compression image. The codebook δ' is divided into three clusters of the same size ($|\text{cluster}_i|_{i=1,2,3} = 5$), as shown in Fig. 3. We determine the first index, which has a value of 4, to be situated in cluster$_1$, and the first bit of secret bits 10 is 1. Then the index having a value of 4 is transformed into the index having a value of 14 in cluster$_3$. The secret bit 0 (removing the first bit 1) is attached to the rear of the index. The second index having a value of 12 in cluster$_3$ that will be embedded in 01 is transformed into an index having a value of 2. The indicator bit I is set to 1, and the secret bits 01 are attached to the rear of the transformed index. Likewise, the third index having a value of 6 is located in cluster$_2$, and the value of the indicator bit I is set to 0. Then, the secret bits 11 are appended to the rear of indicator bit I. Fig. 3 shows the resulting table of the embedding procedure, where the underlined indices denote the indicator values added to the rear of the transformed indices.

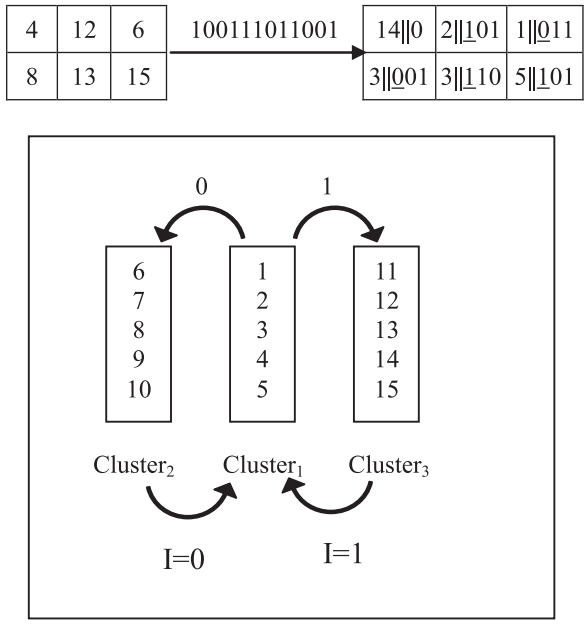

Fig. 3. Example of hiding two-bit secret data.

3.3. Extracting and reversion phase

In the extracting and reversion phase, the receiver first has to acquire the codebook δ' either privately or publicly. Then, the codebook δ' is equally partitioned into three clusters, as in the embedding phase, and the final result is employed to extract the secret bits and reconstruct the original VQ index table of the cover image. To correctly retrieve the hidden secret bits and reconstruct the original image index table, the following two cases should be considered and the algorithm is presented below.

Case 1. One secret bit is embedded in each index.

In this case, only the cluster look-up operation needs to be performed in the data-extracting procedure. The following conditions need to be considered.

(1) If the index value belongs to cluster$_2$, the hidden secret bit is 0. The index in cluster$_2$ should be transformed into the corresponding index in cluster$_1$.

(2) If the index value is located in cluster$_3$, the hidden secret bit is equal to 1. The index in cluster$_3$ should be transformed into the corresponding index in cluster$_1$.

(3) If the index is situated in cluster$_1$, the two bits that follow the index are extracted. The first bit of the two extracted bits is the indicator bit and the second bit is the secret bit.
  (3.1) If the indicator bit following the index is equal to 0, the index in cluster$_1$ should be transformed into the corresponding index in cluster$_2$.
  (3.2) Otherwise, if the indicator bit is equal to 1, the original index should be recovered by the corresponding index in cluster$_3$.

That is, the secret bit σ can be extracted by

$$\sigma = \begin{cases} 0, & \text{if } (\Gamma \in \text{cluster}_2 \vee (\Gamma \in \text{cluster}_1 \wedge \text{the extracted second bit} = 0)) \\ 1, & \text{if } (\Gamma \in \text{cluster}_3 \vee (\Gamma \in \text{cluster}_1 \wedge \text{the extracted second bit} = 1)) \end{cases}. \tag{3}$$

The index value Γ can be recovered by the following transformation:

$$\Gamma \Rightarrow \begin{cases} \text{cluster}_1, & \text{if } (\Gamma \in \text{cluster}_2 \vee \Gamma \in \text{cluster}_3) \\ \text{cluster}_2, & \text{if } (\Gamma \in \text{cluster}_1 \wedge I = 0) \\ \text{cluster}_3, & \text{if } (\Gamma \in \text{cluster}_1 \wedge I = 1) \end{cases} \tag{4}$$

Case 2. More than one bit is embedded in each index.

The following conditions need to be considered:

(1) If the index value belongs to cluster$_2$ or cluster$_3$, the $m-1$ bits of secret data that follow the index are extracted.
  (1.1) If the index value belongs to cluster$_2$, it should be transformed into the corresponding index in cluster$_1$. The secret bits can be recovered by concatenating "0" with the extracted $m-1$ bits.
  (1.2) If the index value belongs to cluster$_3$, it should be transformed into the corresponding index in cluster$_1$. The secret bits can be recovered by concatenating "1" with the extracted $m-1$ bits.
(2) If the index is situated in cluster$_1$, the $(m+1)$ bits attached to the rear of the index are extracted and comprise the secret data except for the first bit.
  (2.1) If the indicator bit following the index is equal to 0, the index in cluster$_1$ should be transformed into the corresponding index in cluster$_2$.
  (2.2) Otherwise, if the indicator bit is equal to 1, the original index should be recovered by the corresponding index in cluster$_3$.

After each index of image blocks is looked up, the original cover image index table is completely reconstructed and all hidden secret bits are extracted correctly.

4. Experimental results and discussions

4.1. Experimental results

We carried out some experiments to demonstrate the performance of our proposed reversible data hiding scheme with a high payload. Fig. 4 shows the eight 512×512 pixel gray-scale images that were used as the test images: i.e., Lena, Airplane, Toys, Tiffany, Zelda, Boat, GoldHill, and Sailboat. The LGB algorithm was used to generate three kinds of codebooks with 256, 512, and 1024 code words, where each code word was a vector with 16 parameters. The five standard test images of Lena, Airplane, Tiffany, Zelda, and Toys were used for the training process (i.e., trained images). The other three test images of Boat, GoldHill, and Sailboat were used only for testing (i.e., non-trained images). To facilitate our proposed scheme, the three codebooks were trained according to the frequencies of occurrence of code words in the five trained images, and the three sorted codebooks with sizes of 256, 512, and 1024 code words were created in descending order. A pseudo random number generator was used to generate a sequence of secret data consisting of 0's and 1's.

We focused on comparing our scheme with Chang et al.'s method [8] and other previous methods [7,14] in terms of the hiding capacity (bits), cost (bits), and neat capacity (bits). The cost denotes the extra bits the method needs to hide the secret, and the neat capacity denotes the result of subtracting the cost from the hiding capacity.

Table 1 shows that the trained images had a better distribution of code words than the non-trained images. The distribution of the code words in the non-trained images increased the cost of Chang et al.'s method [8] and decreased the hiding capacity. Tables 2–4 show the comparison of the proposed method and the schemes by Chang et al.'s method [7], Chang et al.'s method [8] and Jo and Kim' method [14] in terms of the hiding capacity, cost, and neat capacity for different codebook sizes. In all cases the hiding capacity of the proposed scheme outperforms theirs. Chang et al.'s method [7] adopted the modification SMVQ technique to perform lossless embedding, so the hiding capacity is fixed regardless of what kind image it is. Moreover, Jo and Kim' method [14] cannot support the service of lossless recovery. Since the proposed method is reversible and provide additional bit to completely recover original VQ index table. So the neat capacity of the proposed scheme is slightly smaller than that of Jo and Kim' method [14].

The experimental results demonstrated that the hiding capacities of the images are not affected whether or not they are involved in training. When the size of the codebook was increased, the hiding capacities of both the trained and non-trained images decreased, and the costs increased. Increasing the number of indices in cluster$_2$ or cluster$_3$ decreased the hiding capacities. When the indices located in cluster$_3$ were greatly increased, this not only decreased the hiding capacities but dramatically increased the cost. When the codebook was increased, the hiding capacity decreased, and the cost increased dramatically. Compared with Chang et al.'s method [8], the proposed method had higher hiding capacity and lower cost regardless of the images that were used. The results revealed that the proposed method consistently performed better than Chang et al.'s method [8].

4.2. Discussions

Chang et al.'s scheme [8] introduce m bits if the size of codebook is m bits and uses only one-third of the VQ

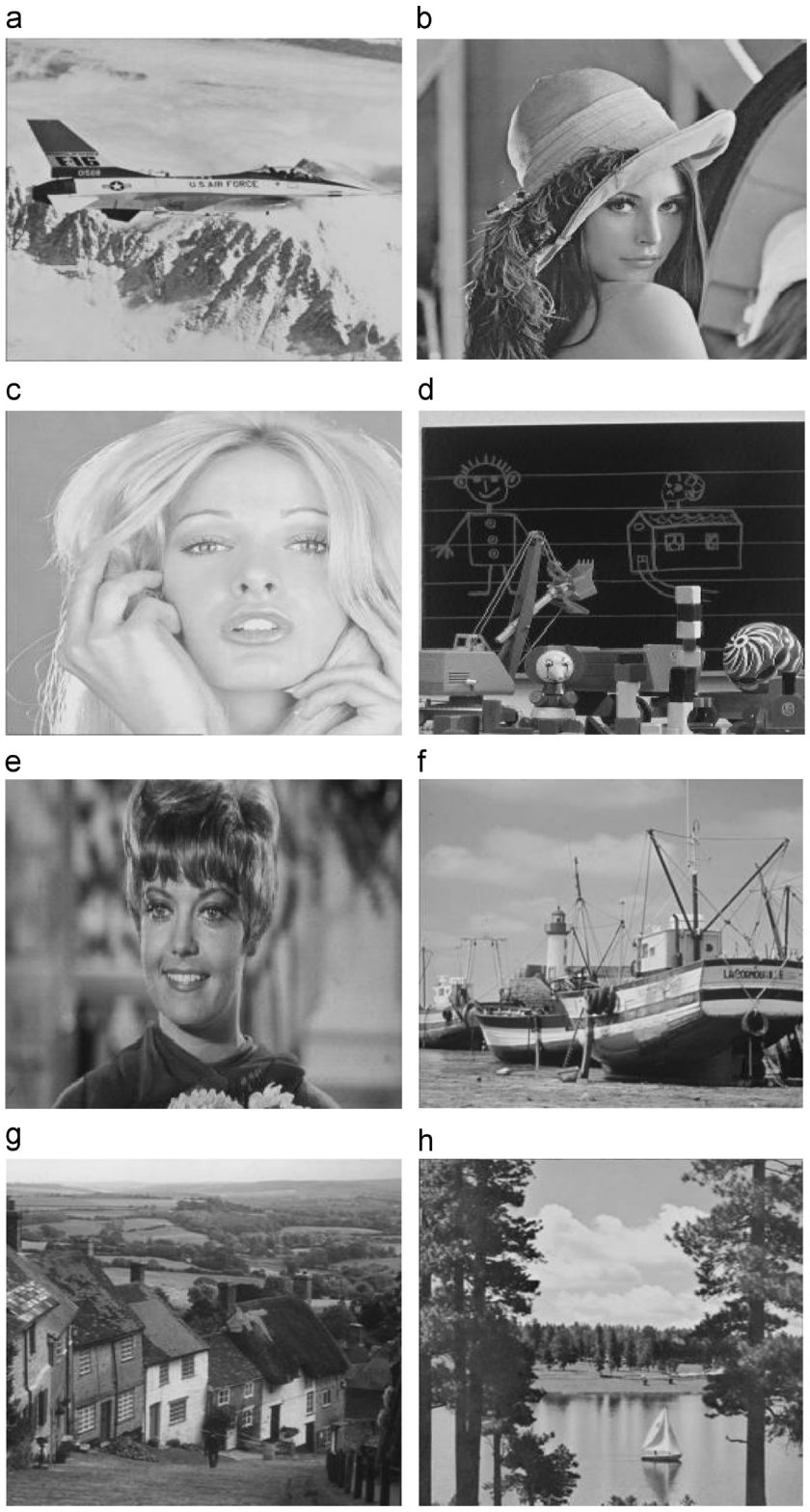

Fig. 4. Test images: (a) Airplane; (b) Lena; (c) Tiffany; (d) Toys; (e) Zelda; (f) Boat; (g) GoldHill and (h) Sailboat.

indices of the cover image to hide the secret bit. However, our proposed method uses only one bit to distinguish indices between cluster_2 and cluster_3. Not only the indices in cluster_1 but also those in cluster_2 and cluster_3 can hide the secret bits. Our proposed scheme reduces the extra bits and increases the hiding capacity.

Table 1
The occurrence frequencies of the codewords for each test image among the three clusters in 256, 512 and 1024 codebook sizes.

Codebook size	Image							
	Trained					Non-trained		
	Airplane	Lena	Tiffany	Toys	Zelda	Boat	GoldHill	Sailboat
256								
$Cluster_1$	12,951	12,598	15,396	14,141	13,061	12,135	10,191	11,523
$Cluster_2$	2423	2783	864	1706	2672	2643	4133	3120
$Cluster_3$	1056	1040	156	560	687	1642	2094	1771
512								
$Cluster_1$	12,658	11,973	15,039	13,451	12,414	11,307	8830	10,262
$Cluster_2$	2746	3290	1041	2285	3207	3103	5123	3800
$Cluster_3$	1114	1217	380	692	835	2049	2482	2370
1024								
$Cluster_1$	13,082	11,582	15,498	13,354	12,695	11,314	9208	10,037
$Cluster_2$	2683	3655	737	2409	2982	3083	4761	3752
$Cluster_3$	765	1284	240	723	832	2102	2500	2671

Table 2
Comparisons of the four data hiding methods (codebook size: 256).

Scheme	Image							
	Trained					Non-trained		
	Airplane	Lena	Tiffany	Toys	Zelda	Boat	GoldHill	Sailboat
Proposed scheme								
Capacity	16,384	16,384	16,384	16,384	16,384	16,384	16,384	16,384
Cost	6958	7646	2040	4532	6718	8570	12,454	9782
Neat capacity	9426	8738	14,344	11,852	9666	7814	3930	6602
Chang et al.'s scheme [8]								
Capacity	12,951	12,598	15,396	14,141	13,061	12,135	10,191	11,523
Cost	8448	8320	1248	4480	5496	13,136	16,752	14,168
Neat capacity	4503	4278	14,148	9661	7565	−1001	−6561	−2645
Chang et al.'s scheme [7] (SCB=16)								
Capacity	8065	8065	8065	8065	8065	8065	8065	8065
Cost	0	0	0	0	0	0	0	0
Neat capacity	8065	8065	8065	8065	8065	8065	8065	8065
Jo and Kim's scheme [14] (TH=80)[a]								
Capacity	15,388	15,650	–	15,344	16,115	–	15,656	–
Cost	0	0	–	0	0	–	0	–
Neat capacity	15,388	15,650	–	15,344	16,115	–	15,656	–

[a] Jo and Kim's scheme cannot support the service of lossless recovery (i.e. it is irreversible), while our proposed scheme is reversible.

For example, if one bit is embedded in each index of the cover image, the cover image with a total number C of VQ indices can be partitioned into three parts: that is, C_1, C_2, and C_3. These are the numbers of indices of the cover image in $cluster_1$, $cluster_2$, and $cluster_3$, respectively, and $C=C_1\cup C_2\cup C_3$, $C_1\cap C_2\cap C_3=\varphi$. The size of the codebook is eight bits; that is, there are 256 code words in the codebook. According to Table 1, the number of indices situated in $cluster_1$ (C_1) is much more than the total number of indices in $cluster_2$ (C_2) and $cluster_3$ (C_3). In other words, the inequality $C_1 \gg (2C_2+2C_3)$ makes sense. Then, we have $C_1 \gg C_2+C_3$. This proves that the hiding capacity of the proposed method is greater than the cost. The proposed scheme is cost-efficient. Compared with Chang et al.'s scheme, the proposed scheme costs less because the inequality $2C_2+2C_3 \le 8C_3$ holds. In summary, our proposed scheme offers improved performance compared to Chang et al.'s scheme.

5. Conclusion

We propose a VQ index scheme with high data hiding capacity for grayscale images. Our proposed scheme not only hides information in the indices of the VQ compression

Table 3
Comparisons of the four data hiding methods (codebook size: 512).

Scheme	Image							
	Trained					Non-trained		
	Airplane	Lena	Tiffany	Toys	Zelda	Boat	GoldHill	Sailboat
Proposed scheme								
Capacity	16,384	16,384	16,384	16,384	16,384	16,384	16,384	16,384
Cost	7720	9014	2842	5954	8084	10,304	15,210	12,340
Neat capacity	8664	7370	13,542	10,430	8300	6080	1174	4044
Chang et al.'s scheme [8]								
Capacity	12,658	11,973	15,039	13,451	12,414	11,307	8830	10,262
Cost	10,026	10,953	3420	6228	7515	18,441	22,338	21,330
Neat capacity	2632	1020	11,619	7223	4899	−7134	−13,508	−11,068
Chang et al.'s scheme [7] (SCB=16)								
Capacity	8065	8065	8065	8065	8065	8065	8065	8065
Cost	0	0	0	0	0	0	0	0
Neat capacity	8065	8065	8065	8065	8065	8065	8065	8065
Jo and Kim's scheme [14] (TH=80)[a]								
Capacity	15,744	15,905	–	15,735	15,321	–	15,110	–
Cost	0	0	–	0	0	–	0	–
Neat capacity	15,744	15,905	–	15,735	15,321	–	15,110	–

[a] Jo and Kim's scheme cannot support the service of lossless recovery (i.e. it is irreversible), while our proposed scheme is reversible.

Table 4
Comparisons of the three data hiding methods (codebook size: 1024).

Scheme	Image							
	Trained					Non-trained		
	Airplane	Lena	Tiffany	Toys	Zelda	Boat	GoldHill	Sailboat
Proposed scheme								
Capacity	16,384	16,384	16,384	16,384	16,384	16,384	16,384	16,384
Cost	6896	9878	1954	6264	7623	10,370	14,522	12,846
Neat capacity	9488	6506	11,430	10,120	8761	6014	1862	3538
Chang et al.'s scheme [8]								
Capacity	13,082	11,582	15,498	13,354	12,695	11,314	9208	10,037
Cost	7650	12,840	2400	7230	8320	21,020	25,000	26,710
Neat capacity	5432	−1258	13,098	6124	4375	−9706	−15,792	−25,703
Chang et al.'s scheme [7] (SCB=16)								
Capacity	8065	8065	8065	8065	8065	8065	8065	8065
Cost	0	0	0	0	0	0	0	0
Neat capacity	8065	8065	8065	8065	8065	8065	8065	8065

image without image distortion but also recovers the original indices to reconstruct the VQ compression image from the hidden indices without any loss.

Typically, each index in the index table can hide one bit of secret data. In other words, the amount of information embedded in an image depends on the number of blocks the image is divided into by the VQ algorithm. Our proposed scheme uses only one extra bit to enable the indices situated in cluster$_2$ or cluster$_3$ to embed secret data. It can increase the hiding capacity of the cover image more than Chang et al.'s scheme [8].

When the codebook is increased, the proposed scheme still provides the highest amount of payload capacity. When the number of embedded secret bits in each index is extended, the proposed scheme can still efficiently work in the three clusters unlike Chang et al.'s method [8]. The experimental results demonstrated that the proposed scheme can achieve both high capability and reversibility.

References

[1] C.K. Chan, L.M. Cheng, Hiding data images by simple LSB substitution, Pattern Recognit. 37 (2004) 469–474.
[2] C.C. Chang, G.M. Chen, M.H. Lin, Information hiding based on search-order coding for VQ indices, Pattern Recognit. Lett. 25 (2004) 1253–1260.
[3] C.C. Chang, J.Y. Hsiao, C.S. Chan, Finding optimal least-significant bit substitution in image hiding by dynamic programming strategy, Pattern Recognit. 36 (2003) 1583–1595.
[4] C.C. Chang, C.Y. Lin, Y.P. Hsieh, Data hiding for vector quantization images using mixed-base notation and dissimilar patterns without loss of fidelity, Inf. Sci. 201 (2012) 70–79.
[5] C.C. Chang, T.C. Lu, Reversible index-domain information hiding scheme based on side-match vector quantization, J. Syst. Softw. 79 (2006) 1120–1129.
[6] C.C. Chang, T.S. Nguyen, C.C. Lin, A novel VQ-based reversible data hiding scheme by using hybrid encoding strategies, J. Syst. Softw. 86 (2013) 389–402.
[7] C.C. Chang, W.L. Tai, M.H. Lin, A reversible data hiding scheme with modified side match vector quantization, in: Proceedings of the IEEE 19th International Conference on Advanced Information Networking and Applications (AINA'05), Taipei, Taiwan, vol. 1, 2005, pp. 947–952.
[8] C.C. Chang, W.C. Wu, Y.C. Hu, Lossless recovery of a VQ index table with embedded secret data, J. Vis. Commun. Image Represent. 18 (2007) 207–216.
[9] Y.H. Chiou, J.D. Lee, Reversible data hiding based on search-order coding for VQ-compressed images, J. Converg. Inf. Technol. (JCIT) 6 (12) (2011) 177–184.
[10] C. Christopoulos, A. Skodras, T. Ebrahimi, The JEPG2000 still image coding system: an overview, IEEE Trans. Consum. Electron. 46 (4) (2000) 1103–1127.
[11] Z. Eslami, J.Z. Ahmadabadi, Secret image sharing with authentication chaining and dynamic embedding, J. Syst. Softw. 84 (5) (2011) 803–809.
[12] R.M. Gray, Vector quantization, IEEE Acoust. Speech Signal Process. Mag. 1 (1984) 4–29.
[13] X. Gui, X. Li, B. Yang, A high capacity reversible data hiding scheme based on generalized prediction-error expansion and adaptive embedding, Signal Process. 98 (2014) 370–380.
[14] M. Jo, H.D. Kim, A digital image watermarking scheme based on vector quantization, IEICE Trans. Inf. Syst. E85-D (6) (2002) 1054–1056.
[15] T. Kim, Side match and overlap match vector quantizers for images, IEEE Trans. Image Process. 1 (1992) 170–185.
[16] C.F. Lee, H.L. Chen, A novel data hiding scheme based on modulus function, J. Syst. Softw. 83 (5) (2010) 832–843.
[17] C.Y. Lin, C.C. Chang, Hiding data in VQ-compressed images using dissimilar pairs, J. Comput. 17 (2) (2006) 3–10.
[18] Y. Linde, A. Buzo, R.M. Gray, An algorithm for vector quantizer design, IEEE Trans. Commun. 36 (1980) 84–95.
[19] Z.M. Lu, S.H. Sun, Digital image watermarking technique based on vector quantization, Electron. Lett. 36 (4) (2000) 303–305.
[20] B. Ou, X. Li, Y. Zhao, R. Ni, Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion, Signal Process.: Image Commun. 29 (2014) 760–772.
[21] C. Qin, C.-C. Chang, Y.-C. Chen, Efficient reversible data hiding for VQ-compressed images based on index mapping mechanism, Signal Process. 93 (9) (2013) 2687–2695.
[22] P. Rahmani, G. Dastghaibyfard, Reversible data hiding for VQ-compressed images based on an index replacement technique, Int. J. Comput. Electr. Eng. 4 (3) (2012) 359–362.
[23] G.K. Wallance, The JPEG still picture compression standard, IEEE Trans. Consum. Electron. 38 (1) (1992) 17–34.
[24] R.Z. Wang, C.F. Lin, J.C. Lin, Image hiding by optimal LSB substitution and genetic algorithm, Pattern Recognit. 34 (3) (2001) 671–683.
[25] L. Wang, Z. Pan, X. Ma, S. Hu, A novel high-performance reversible data hiding scheme using SMVQ and improved locally adaptive coding method, J. Vis. Commun. Image Represent. 25 (2) (2014) 454–465.
[26] R.Z. Wang, Y.D. Tsai, An image-hiding method with high hiding capacity based on best-block matching and *k*-means clustering, Pattern Recognit. 40 (2007) 398–409.
[27] C.C. Wu, S.J. Kao, M.S. Hwang, A high quality image sharing with steganography and adaptive authentication scheme, J. Syst. Softw. 84 (2011) 2196–2207.
[28] Z. Xinpen, W. Shuozhong, Q. Zhenxing, F. Guorui, Reference sharing mechanism for watermark self-embedding, IEEE Trans. Image Process. 20 (2) (2011) 485–495.
[29] C.H. Yang, Y.C. Lin, Fractal curves to improve the reversible data embedding for VQ-indexes based on locally adaptive coding, J. Vis. Commun. Image Represent. 21 (2010) 334–342.
[30] Y.H. Yu, C.C. Chang, Y.C. Hu, Hiding secret data in images via predictive coding, Pattern Recognit. 38 (2005) 691–705.