Reversible Data Hiding Scheme for VQ Indices Based on Absolute Difference Trees

Chin-Chen Chang[1, 2], Thai-Son Nguyen[1, 3], and Chia-Chen Lin[4]
[1] Department of Information Engineering and Computer Science, Feng Chia University,
Taichung 40724, Taiwan, R.O.C,
[e-mail: alan3c@gmail.com]
[2] Department of Biomedical Imaging and Radiological Science, China Medical University
Taichung 40402, Taiwan, R.O.C,
[3] Department of Information Technology, Tra Vinh University
Tra Vinh Province, Vietnam
[e-mail: thaison@tvu.edu.vn]
[4] Department of Computer Science and Information Management,
Providence University, Taichung 43301, Taiwan, R.O.C,
[e-mail: ally.cclin@gmail.com]
*Corresponding author: Chia-Chen Lin

Received February 12, 2014; revised April 22, 2014; accepted May 17, 2014; published July 29, 2014

Abstract

Reversible data hiding is a technique for recovering original images without any distortion after secret data are extracted from the image. The technique continues to attract attention from many researchers. In this paper, we introduce a new reversible data hiding scheme based on the adjacent index differences of vector quantization (VQ) indices. The proposed scheme exploits the differences between two adjacent indices to embed secret data. Experimental results show that our scheme can achieve a lower compression rate than an earlier scheme by Yang and Lin. Our scheme's average compression rate, 0.44 bpp, outperforms that of Yang and Lin's scheme, which averages 0.53 bpp. Moreover, the embedding capacity of our scheme can rise to 1.45 bpi, which also is superior to that of Chang et al.'s scheme [35] (1.00 bpi)Yang and Lin's scheme [27] (0.91 bpi) as well as Chang et al.'s scheme [26] (0.74 bpi).

Keywords: Data hiding, image compression, reversible, steganography, VQ compression

http://dx.doi.org/10.3837/tiis.2014.07.021

1. Introduction

With the rapid progress in multimedia and Internet technologies, most digital data can be transmitted over the Internet without time or space constraints. However, the Internet's open environment means that transmitted data may be exposed to illegal or unauthorized modification or may be eavesdropped upon. Therefore, protecting transmitted data from such attacks has become a critical issue. Over the past ten years, lots of research results have been dedicated to find solutions that enhance the security of transmitted data. Generally, data protection techniques can be divided into two approaches: cryptography and data hiding. Cryptography transforms secret data into a meaningless form by using cryptographic algorithms such as DES [1] or AES [2]. The meaningless form gives the transmitted data some protection, but it also may raise the suspicions of attackers. By contrast, data hiding hides secret data in meaningful cover media, such as images, videos, audios or texts, before it is transmitted over the Internet. The original nature of the cover media remains even through it carries the secret data. Therefore, it is more difficult for malicious attackers to distinguish whether a media object carries secret data. In other words, the security of the hidden data is guaranteed.

There are two basic types of data hiding schemes. The first is irreversible data hiding, which embeds secret data into a cover image to generate a stego-image. The use of irreversible data hiding maintains the security of secret data when it is transmitted over the Internet. Many irreversible data hiding schemes [3-13] have been proposed. The merit of irreversible data hiding is that its hiding capacity is quite high. Take for example least significant bit (LSB) substitution [13], one of the most popular irreversible data hiding schemes. With LSB substitution, the number of hidden secret bits of each pixel in a cover image can range from 1 bit to 4 bits. However, the one weakness in irreversible data hiding is that it does not allow a cover image to be recovered exactly after secret data are extracted.

The second type of data hiding is reversible data hiding, also called lossless data hiding. Reversible data hiding compensates for the weakness in irreversible data hiding with its reversibility feature. This reversibility enables the cover image to be reconstructed exactly after secret data are extracted. Many reversible data hiding schemes [14-22] have been proposed in the past five years. In 2002, Fridich et al. [16] introduced a reversible data hiding scheme that compresses the least significant bits (LSBs) of pixels in the cover image, appends the secret data to be compressed, and encrypts the data and, finally, replaces the LSB of the cover image with the encrypted codes. However, Fridich et al.'s scheme does not offer a high embedding capacity. Also in 2002, Celik et al. [15] proposed a new reversible data hiding scheme that improved on Fridich et al.'s scheme by using vector quantization and lossless compression. Celik et al.'s scheme offers lower distortion of the stego-image and higher hiding capacity than those of Fridich et al.'s scheme. In 2003, Tian [21] proposed a reversible data hiding scheme that relies on the difference expansion between a pair of pixels. With Tian's scheme, one secret bit can be embedded by depending on the expansion difference between a pair of pixels. To improve the hiding capacity of Tian's scheme, in 2004 Alattar [14] presented a new reversible data hiding scheme that uses a vector in place of the pixel pair in Tian's scheme. Alattar's scheme successfully embeds more bits into each vector; therefore, it not only achieves computational efficiency but also increases hiding capacity.

Typically, cover images used in data hiding schemes can be divided into three domains: the spatial domain, the transform domain and the compression domain. In spatial domain schemes

[5, 6, 9-16], secret data are embedded by modifying the cover image's spatial information. In the transform domain [8, 23-25], secret data are embedded by shifting the coefficient values of the cover image. However, these two techniques require larger bandwidth and storage space to transmit stego-media because the stego-media become larger than the cover media after secret data are embedded. Therefore, spatial and transform domain schemes are not suitable when transmission bandwidth is limited. Compression domain schemes are considered the best choice in this case because the compression codes are significantly smaller than the cover image before data hiding. Even with the hidden data, the compression code size is still smaller than the cover image. Research over the past five years has relied on compression codes generated by VQ-based or SMVQ-based compression [26-28, 29, 30, 31, 34, 35], BTC [32] and DCT [33], to carry secret data simply because of this characteristic.

In these compression domain schemes, many scholars have tried to embed secret data into VQ-compressed images [26-28, 30, 31]. In 2006, Lin and Chang [31] introduced a data hiding scheme that relies on a pair of dissimilar codewords in the sorting codebook to embed one secret bit. In 2007, Chang et al. [26] proposed a reversible data hiding scheme by clustering the VQ codebook into three groups. Only the VQ indices located in group with the highest referred counts are used to hide one secret bit of data. The other two groups are used for the future recovery of image data. In 2009, Chen and Huang [28] proposed a hybrid scheme that combines the dynamic tree coding scheme (DTCS) and modified search order coding (MSOC) to encode the index table completely and efficiently without any extra coding distortion. By depending on the useful information in the left and upper blocks adjacent to the current processing block, they applied the HLIC method to embed secret information. To improve the embedding capacity and compression rate of Chang et al.'s scheme [26], Yang and Lin [27] proposed new VQ-based reversible data hiding scheme. In this scheme, Yang and Lin sort and divide the VQ codebook into 2^B clusters, where B is the number of secret bits embedded into each VQ index, and half of the clusters are used to embed secret data. However, the average compression rate and average embedding rate of Yang and Lin's scheme are still around 0.5 bpp and 0.9 bpi, respectively. In 2013, Chang et al. [35] proposed a new compression and information hiding hybrid scheme based on SMVQ and search order coding (SOC). Their scheme obtained both the low compression rate and the high embedding capacity when compare with Chang et al.'s scheme [26] and Yang and Lin [27]. However, the embedding rate obtained by their scheme is small because most of the indices simply embed one bit of secret data. To further improve the embedding rate while achieving a better compression rate, in this paper, we introduce a new reversible data hiding scheme based on the differences between neighboring index values in the index table. Experimental results confirm that our proposed scheme offers better compression and embedding rates than the schemes proposed by Chang et al. [26, 35] and Yang and Lin [27], while maintaining the same visual image quality as those three schemes.

The rest of this paper is organized as follows. Section 2 reviews VQ and Yang and Lin's scheme. Details of the proposed scheme are described in Section 3. Experimental results are discussed in Section 4. Finally, some conclusions are given in Section 5.

2. Related Work

In this section, we first introduce traditional vector quantization (VQ) because our proposed scheme embeds secret data into VQ-generated indices. Then, Yang and Lin's VQ-based reversible data hiding scheme is introduced.

2.1 Vector quantization

In 1984, Gray [3] proposed vector quantization, also called VQ. The VQ technique is well-known for both its lossy data compression and its efficient clustering. The main idea behind VQ is that the k-dimensional space R^k is mapped into its finite subset $Y = \{y_i ; i = 1,2, \ldots, N\}$, where y_i is called a codeword and the set Y is called a codebook.

In VQ, a codebook must be generated by a set of training images and clustering algorithm in advance. The well-known iterative generalized Lloyd algorithm also called Linde-Buzo-Gray algorithm (LBG) [3] is often used to generate codebooks for VQ. As **Fig. 1** shows, in VQ, an image is first decomposed into several non-overlapping blocks, with each block treated as a vector and all blocks encoded sequentially in raster scan order. In the encoding phase, each k-dimensional input vector $x = (x_1, x_2, \ldots, x_k)$ is compared with the codewords in the codebook to determine the nearest codeword, which is based on the Euclidian distance defined in Equation (1).

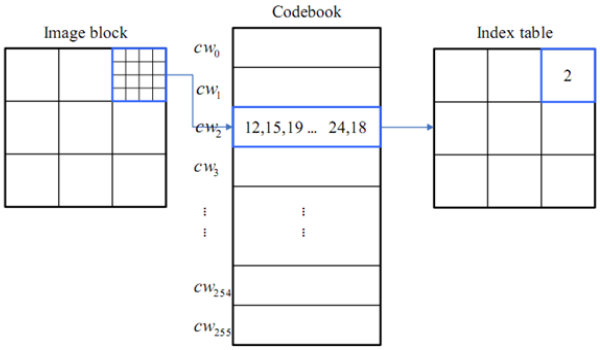


Fig. 1. Encoding flowchart of the VQ encoder

Here, the similarity between the input vector x and a codeword y_i of the codebook is measured by the Euclidian distance shown in Equation (1):

$$d(x, y_i) = \sqrt{\sum_{j=1}^{k} (x_j - y_{ij})}, \tag{1}$$

where x_j is the jth component of the input vector x, and y_{ij} is the jth component of the codeword y_i. Once a codeword with the nearest Euclidian distance is found, it is used to encode the current processing input vector. An index table is generated after all vectors are processed, and the index table is sent to the receiver.

In the decoding phase, the same codebook is used to find the corresponding codeword for each index of the received index table. Then the original image can be recovered. The index table is the result of VQ encoding. Because the index table is smaller than the original image, the objective of VQ compression is achieved. Assume an original image size 512×512 is divided into a vector with 16 dimensions, and a codebook is sized 256. Each vector requires only 8 bits after VQ compression; therefore, the VQ compression rate is 1/2 (bpp). However, by using the Euclidian distance formula in Equation (1), the index values of the nearest codewords are found during compression processing. Some distortion between the original image and the reconstructed image may occur. Thus, VQ is a lossy compression scheme.

2.2 Yang and Lin's reversible data hiding scheme

In 2009, Yang and Lin [27] proposed a new reversible data hiding scheme based on VQ. The purpose of Yang and Lin's scheme is to improve the embedding capacity of Chang et al.'s scheme [26]. Assume that codebook $Y=(y_0, y_1,\ldots, y_{n-1})$ with size n is given. Y' is the sorted codebook with a size $n'=\left\lfloor \frac{n-2^{B-1}}{2^B} \right\rfloor \times 2^B$ that is created and divided into 2^B groups, where B is the number of secret bits embedded into each VQ index. All groups are the same size as $m=\left\lfloor \frac{n-2^{B-1}}{2^B} \right\rfloor$. In this scheme, half of the indices in Y' are used to embed secret data and at least 2^{B-1} indices are used as indicators. For an image I, each block I_i is encoded into index y_i by VQ coding. y_i is then transformed into y_i', where y_i' is the corresponding index in a group and sometimes includes an indicator. To better explain Yang and Lin's scheme, the function $tranj(y_i)$ is used to represent the transformation function, where y_i is transformed into the corresponding index in group C_j. **Fig. 2** shows an example of Yang and Lin's scheme when B is 1. In Yang and Lin's scheme, if index y_i belongs to group C_0, y_i' is set to $trans_0(y_i)$ when secret bit s=0 is hidden and y_i' is set to $trans_1(y_i)$ when secret bit s=1 is hidden. If the index y_i belongs to group C_1, indicator I_0 is added ahead by setting y_i' to $I_0 \| trans_1(y_i)$, and no secret data is hidden.

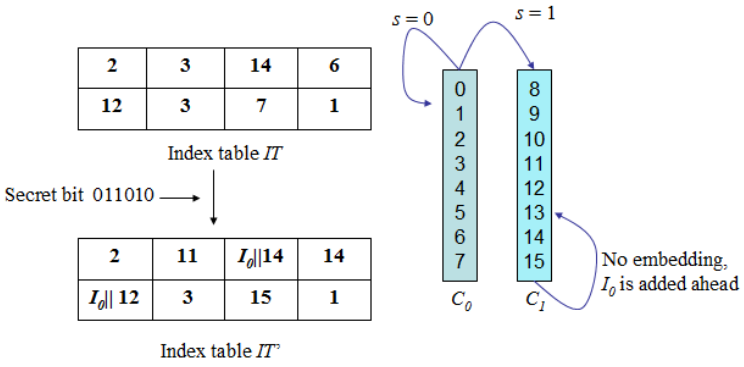


Fig. 2. Example of Yang and Lin's scheme with B=1

In the embedding phase, the grayscale image I sized $W \times H$ is used to embed secret data and the secret message is denoted as $S=(s_0, s_1,..., s_r)$, where $s_j=\{0,1\}$ and $0 \le j \le r$. The following algorithm shows the embedding phase in detail.

Embedding algorithm

Input: Cover image I, secret message B, codebook Y' divided into 2 groups, C_0, and C_1.
Output: Transformed index table IT'.
Step 1: Encode cover image I by using the VQ encoder to produce index table IT.
Step 2: Read index y from index table IT and read secret bit s from the secret message S.
Step 3: If $y \in C_0$, and s =0, y is unchanged by setting $y'=trans_0(y)$.
Step 4: If $y \in C_0$, and s =1, transfer y to group C_1 by setting $y'= trans_1(y)$.

Step 5: If $y \in C_1$, no secret data is hidden, and y is transferred by setting $y' = I_0 || trans_1(y)$, where I_0 is the indicator.
Step 6: Repeat Steps 2 to 5 until all indices in the index table are processed.
After the entire index table IT is transferred into the transformed index table IT', the sender sends IT' and codebook Y' with its two groups, C_0, and C_1, to the receiver. When the receiver gets this information from the sender side, the receiver performs the following steps to obtain secret message S and recover the original index table IT.
Extracting algorithm
Input: Transformed index table IT' and codebook Y', which is divided into two groups, C_0, and C_1.
Output: Recovered index table IT and secret message S.
Step 1: Read index y' from the index table IT'.
Step 2: If y' has no indicator, original index y is recovered by using the formula $y = trans_0(y')$ and 1-bit secret data s is extracted according to the following rules. If $y' \in C_k$, then $s = k$ (i.e., if $y' \in C_1$, s=1. Otherwise, if $y' \in C_0$, s=0).
Step 3: If y' has indicator I_0 and $y' = I_0 || y''$ recover y by using the formula $y = trans_1(y'')$, and no secret data is extracted in this case.
Step 4: Repeat Steps 1 to 3 until index table IT' is processed completely.

After the extracting phase is completed, secret message S is extracted and the original index table IT is reconstructed exactly. Then the traditional VQ decoder can be applied to reconstruct cover image I.

3. Proposed Scheme

To achieve the high compression and embedding rate performance of Yang and Lin's scheme [27] while maintaining the same image quality in the reconstructed image, this section presents a novel reversible data hiding scheme based on the difference between adjacent indices in the index table. The proposed scheme encodes the VQ index table into a binary code stream and hides secret data during the encoding process. Comparing with some previous data hiding schemes [26, 27, 35] in the VQ domain, the proposed scheme achieves both a higher embedding capacity and a higher compression rate.
For this discussion, the cover image is grayscale image I sized $W \times H$, with codebook $Y = (y_0, y_1, \ldots, y_{n-1})$, and the secret message is denoted as $S = (s_0, s_1, ..., s_r)$, where $s_j = \{0,1\}$ and $0 \le j \le r$. Our proposed reversible data hiding scheme can be divided into two phases: data embedding phase and data extracting phase. These two phases are discussed in Subsections 3.1 and 3.2, respectively.

3.1 Data embedding phase

Fig. 3 is a flowchart of our data embedding phase, which involves three operations: determining the scan path and computing adjacent indices' differences, constructing the absolute difference tree and data embedding.

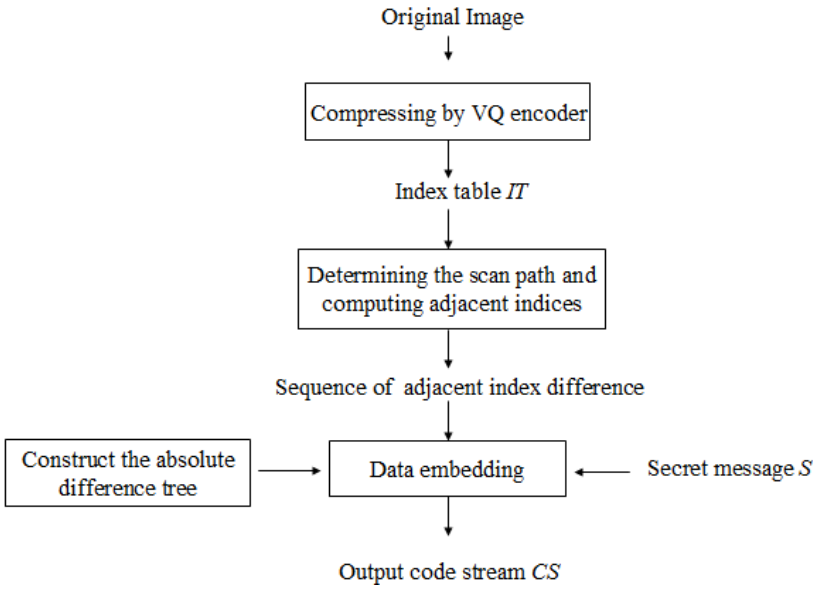


Fig. 3. Flowchart of data embedding algorithm

The determining the scan path and computing adjacent indices' differences phase is first operation. To obtain all the directional adjacent indices' differences in the index table of the cover image, nine scan paths must be used in this phase. To better explain this step, the basic scan paths are shown in detail in the 4×4 indices in **Fig. 4**. After applying one of the scan paths in **Fig. 4**, the sequence of index values is obtained as $v_1, v_2, \ldots, v_k$, where k is the size of the index table.

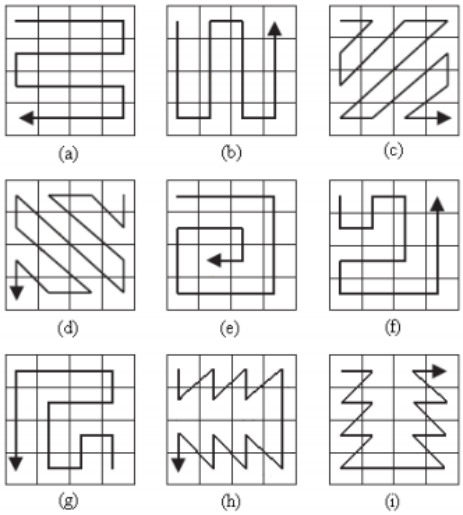


Fig. 4. Nine basic scan paths

After the sequence of index values $v_1, v_2, \ldots, v_k$ is obtained according to the determined scan path, a sequence of adjacent index differences, $D = \{d_1, d_2, \ldots, d_k\}$, is calculated by using Equation (2). Adjacent index difference d_i can be a negative or positive value.

$$d_i = \begin{cases} v_{i+1} - v_i & 1 \le i \le k \\ v_i & i = k \end{cases}, \tag{2}$$

Four steps must be performed to construct the absolute difference tree in the second operation. In an absolute difference tree, each node represents the absolute difference value of two neighboring indices based on the determined scan path. The absolute difference tree can be divided into four levels. As **Fig. 5** shows, the root is level 0 and the four child nodes of the root make up level 1. For level 2, 12 nodes (i.e., node 0 to node 11) which are the child nodes of the level 2 are generated. In the same way, level 3 consists of the child nodes of level 2, which are labeled node 0 to node 31. Note that each absolute difference value larger than 31 also belongs to level 3. For each level, a different bit number is used to encode the members in the different level. For example, 4 bits is used for level 0, 7 bits for level 1, 9 bits for level 2 and 11 bits for level 3, respectively. Note that 7 bits contains a 2-bit indicator, 1 bit a sign of the difference value and a 4-bit binary representation of the corresponding child value, as described in Step 5.2 of the data embedding algorithm. Clearly, the absolute difference value belongs to the smaller level; thus, fewer bits are required for encoding. In this case, if a larger level is created (i.e., level 4), the compression rate will be less. Therefore, the proposed scheme uses only four levels.

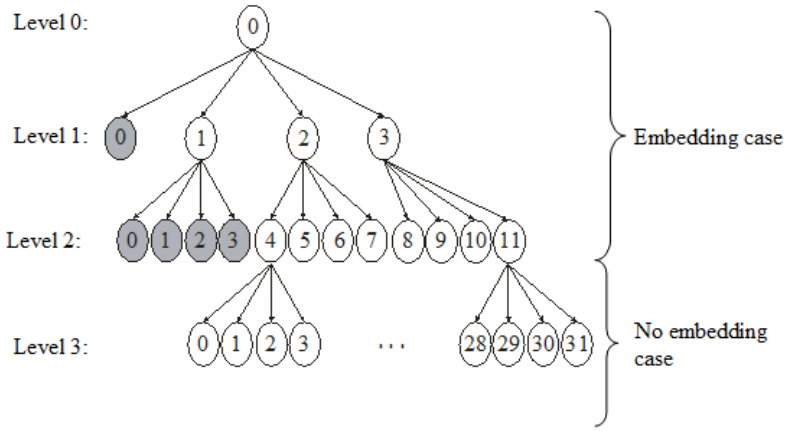


Fig. 5. Example of the absolute difference tree

Step 1: The value 0 is first chosen as the root node of the absolute difference tree.
Step 2: The four child nodes of the root are added and their order is set as 0, 1, 2, and 3.
Step 3: Choose 4 child nodes for each of nodes 1, 2 and 3. Note that the order of the child nodes starts with 0, and the 12 nodes generated for level 2 are labeled node 0 to node 11.
Step 4: Keep choosing a new child for each node that has value between 4 and 11. Again, the order of new child nodes starts with 0, and the 32 nodes generated for level 3 are labeled node 0 to node 31. Construction of the absolute difference tree stops when Step 4 is completed.

As **Fig. 5** shows, once the four steps are completed the absolute difference tree is created. The purpose of constructing an absolute difference tree is to provide important information for the embedding process. In **Fig. 5**, some gray nodes do not have child nodes because the gray nodes have already used in a lower level. For example, the white node 0 in level 0 already has four child nodes; therefore, the gray node 0 in level 1 does not have child nodes. The nodes of the absolute difference tree are classified into two cases. If the absolute difference value d belongs to the elements of level 0, level 1, or level 2 (absolute difference value ranges from 0 to 11), it is classified in Case 1 and two secret bits are embedded. Otherwise, the absolute difference value d belongs to Case 2 and no secret bit can be embedded.

Our proposed data embedding phase can be broken into six steps. The corresponding algorithm is presented in detail here.

Data embedding algorithm

Input: Grayscale cover image I, codebook Y, secret message S.

Output: Code stream CS in binary form.

Step 1: Encode cover image I by using a VQ encoder to obtain index table IT.

Step 2: Determine a scan path. Next, compute adjacent indices' differences to generate the sequence of adjacent indices' differences D. Then, construct the absolute difference tree using the four steps described earlier. Finally, depending on the absolute difference tree, encode each index difference value of the sequence of adjacent index differences D.

Step 3: Read index difference value d from the sequence of adjacent indices' differences D.

Step 4: If $11 < |d|$, this case belongs to Case 2 of the absolute difference tree, and no secret bit is hidden. Therefore, d is encoded by r=00$\| sign(d) \|(|d|)_2$, and r is concatenated into code stream CS, where $sign(d)$ is defined in Equation (3), $(|d|)_2$ is the 8-bit binary representation of the absolute value of d, and "||" represents the concatenation operation.

$$sign(d) = \begin{cases} 0 & if\ d < 0 \\ 1 & if\ d > 0 \end{cases}. \tag{3}$$

Step 5: Otherwise (i.e., $|d| \le 11$), this is Case 1 of the absolute difference tree, the embedding case. The next two secret bits $s_1\ s_2$ are read from the secret message B.

Step 5.1: If d = 0, encode d by r=01$\| s_1\ s_2$ and concatenate r into code stream CS.

Step 5.2: If $1 \le |d| \le 3$, encode d by r=10$\| sign(d) \| w_2$, where $sign(d)$ is defined in Equation (3). w is the child node of d in the absolute difference tree in **Fig. 5**, w is calculated as in Equation (4), and w_2 is the 4-bit binary representation of w. Finally, concatenate r into code stream CS.

$$w = \begin{cases} (|d|-1)\times 4 & if\ \ s_1 s_2 = 00 \\ (|d|-1)\times 4+1 & if\ \ s_1 s_2 = 01 \\ (|d|-1)\times 4+2 & if\ \ s_1 s_2 = 10 \\ (|d|-1)\times 4+3 & if\ \ s_1 s_2 = 11 \end{cases}. \tag{4}$$

Step 5.3: If $4 \le |d| \le 11$, encode d by r=11|| $sign(d)$ ||q_2, where $sign(d)$ is defined in Equation (3), q is the child node of d in the absolute difference tree in **Fig. 5**, q is calculated by Equation (5), and q_2 is the 6-bit binary representation of q. Finally, concatenate r into code stream CS.

$$q = \begin{cases} (|d|-4)\times 4 & if \quad s_1 s_2 = 00 \\ (|d|-4)\times 4+1 & if \quad s_1 s_2 = 01 \\ (|d|-4)\times 4+2 & if \quad s_1 s_2 = 10 \\ (|d|-4)\times 4+3 & if \quad s_1 s_2 = 11 \end{cases}. \tag{5}$$

Step 6: Repeat Steps 2 to 5 until all difference values in D are processed.

After all steps are completed, code stream CS is obtained. Finally, the sender sends the code stream CS and the determined scan path to the receiver. To better explain our data embedding phase, an example is shown in **Fig. 6**. Assume VQ encoding is conducted and one index table sized 4×4 is generated as shown in **Fig. 6**. A scan path as shown in **Fig. 3(a)** is determined.

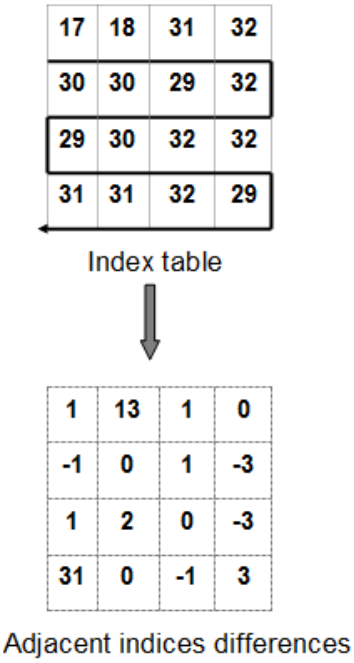


Fig. 6. Example of the embedding phase of our scheme with a 4×4 index table and the scan path of **Fig. 3(a)**.

Along the **Fig. 3(a)** scan path, the adjacent indices' differences are generated by using Equation (2). Assume secret message S is "100011011011" and the absolute difference tree is generated as in **Fig. 5**. The first difference value is "1", so $|d|$=1≤11. This is Case 1 of the absolute difference tree, the embedding case. According to Step 5.2, the 2-bit indicator should be "10", and the two secret bits s_1s_2=10 are read from secret message S. Finally, d is encoded by r =10||0||0010, and r is concatenated into code stream CS. The second difference index value is 13, so $|d|$=13 ≥12. This case belongs to Case 2 of the **Fig. 5**. Therefore, no secret data is embedded. Finally, r = 00||0||00001101 and is concatenated into code stream CS according

to Step 4. For the fourth difference index value, d=0, and two secret bits s_1s_2=11 are encoded by r=01||11. Then r is concatenated into code stream CS. Depending on the data embedding algorithm described earlier, the entire sequence of adjacent index differences will be encoded and code stream CS will be obtained.

3.2 Data extracting phase

When the receiver gets code stream CS from the sender, the receiver extracts secret message S and reconstructs original index table IT by performing the following steps to recover original image I. A detailed flowchart of our proposed extracting phase is presented in **Fig. 7**.

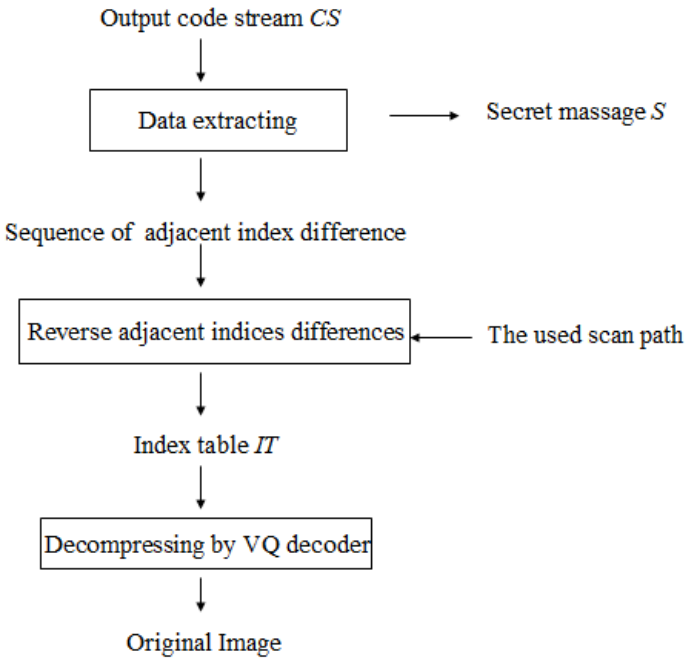


Fig. 7. Flowchart of our extracting phase

Data extracting algorithm
Input: Code stream CS.
Output: Secret message S and the original index table IT.
Step 1: Read 2 bits p from code stream CS.
Step 2: If p=00, no secret data is hidden. The original difference index value d is reconstructed according to the following operations. First, read 1 following bit as the sign of d by using Equation (3). Next, read the 8 next bits and convert them into decimal values as $|d|$. Finally, d is sent to the sequence of adjacent indices' differences D.
Step 3: If p=01, read the two next bits as two secret bits s_1s_2. The original difference index d is recovered as 0. Then concatenate s_1s_2 into secret message S and insert d into the sequence of adjacent indices' differences D.
Step 4: If p=10, then reconstruct d by using following processes. First, read 1 next bit as the sign of d. Then, read the 4 next bits and change them to decimals as w, then calculate $|d|$ by using Equation (6).

$$|d| = \left\lfloor \frac{w}{4} \right\rfloor + 1. \tag{6}$$

Two secret bits s_1s_2 is extracted by using Equation (7).

$$s_1s_2 = \begin{cases} 00 & if\ w = (|d| - 1) \times 4 \\ 01 & if\ w = (|d| - 1) \times 4 + 1 \\ 10 & if\ w = (|d| - 1) \times 4 + 2 \\ 11 & if\ w = (|d| - 1) \times 4 + 3 \end{cases}. \tag{7}$$

Then, send d to sequence D and concatenated s_1s_2 into secret message S.

Step 5: If p=11, then reconstruct d by using following processes. Read 1 next bit as the sign of d. Read 6 next bits and change them to decimals as w. Then compute $|d|$ by using Equation (8).

$$|d| = \left\lfloor \frac{w}{4} \right\rfloor + 4. \tag{8}$$

After that, extract 2 secret bits s_1s_2 by using Equation (9).

$$s_1s_2 = \begin{cases} 00 & if\ w = (|d| - 4) \times 4 \\ 01 & if\ w = (|d| - 4) \times 4 + 1 \\ 10 & if\ w = (|d| - 4) \times 4 + 2 \\ 11 & if\ w = (|d| - 4) \times 4 + 3 \end{cases}. \tag{9}$$

Then, the original difference index value d is sent to the sequence of adjacent indices' differences D, and s_1s_2 is concatenated into secret message S.

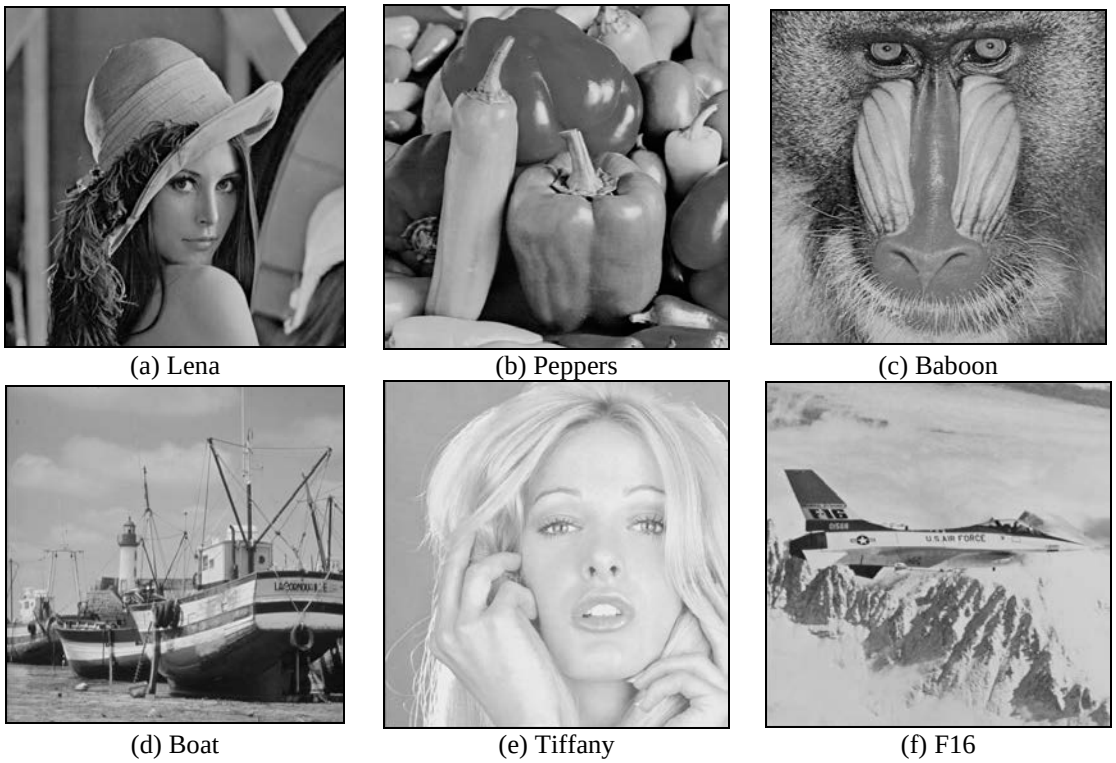
Step 6: Repeat Steps 1 to 5 until the entire code stream CS is processed.

Step 7: Recover the original index table by reversing the adjacent index difference process depending on the scan path used.

After the extracting phase is completed, the original index table is reconstructed and secret message S is extracted. Then, original image I can be regenerated from the reconstructed index table by using the VQ decoder.

4. Experimental Classification Results and Analysis

This section describes the experiments conducted to demonstrate the performance of our proposed scheme in hiding capacity, compression rate and visual quality of the recovered image. Three existing schemes, Chang et al.'s schemes [26, 35] and Yang and Lin's scheme [27], were compared with ours. In our experiments, the size of each non-overlapping image block was 4×4, with each block a 16-dimensional vector. The codebook used in our experiments was sized 256 and was trained by using the well-known LBG algorithm [3] with three training grayscale images, "Lena", "F16" and "Boat". The six general grayscale test images shown in **Fig. 8**, "Lena", "Boat", "Peppers", "F16", "Tiffany", and "Baboon", were used in our experiments. All computing was performed on a PC with a 2.1GHz Intel(R) Core™2 CPU and a 1-GB RAM. The operating system was Windows 7 Professional and our algorithm was programmed by Microsoft Visual Studio 2005 C#.



(a) Lena (b) Peppers (c) Baboon

(d) Boat (e) Tiffany (f) F16

Fig. 8. Six 512×512 grayscale test images

We evaluated the performance of our scheme against three criteria: hiding capacity, compression rate and visual quality of the recovered image. The evaluation criteria are described in the following paragraphs. The compression rate (*CR*) is the ratio of the size of the compression output code stream to the size of the original cover image and is defined in Equation (10).

$$CR = \frac{\|I'\|}{H \times W}(bpp), \tag{10}$$

where $\|I'\|$ is the size of the output code stream, and H and W are the height and width of the cover image. Bit per pixel (bpp) is the *CR* unit. The use of *CR* allows us to estimate compression performance. Basically, the smaller the compression rate the better the compression performance. In contrast, the larger the compression rate the poorer the compression performance. **Fig. 9** compares the results for our scheme, Chang et al.'s scheme and Yang and Lin's scheme in compression rate.

In addition, to evaluate the hiding performance of our proposed scheme the embedding rate (*ER*) is defined as in Equation (11). *ER* shows how many secret bits can be embedded into each index of the cover image.

$$ER = \frac{\|S\|}{N_{IDX}}(bpi), \tag{11}$$

where $\|S\|$ is the number of bits that can be embedded into a cover image and N_{IDX} is the number of indices in the index table generated by VQ compression. The higher the embedding rate is, the larger the secret message can be embedded. The embedding performance results for

our scheme and the two existing schemes used for comparison are shown in **Fig. 10**.

The third measure is visual quality of the recovered image because visual quality is used to estimate the degree of distortion between the original cover image and the recovered image. Peak signal-to-noise (*PSNR*) ratio is used to evaluate image quality, as shown in Equation (12).

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right), \tag{12}$$

where the mean square error (*MSE*) for a $W \times H$ grayscale image is defined as in Equation (13).

$$MSE = \frac{1}{H \times W}\sum_{i=1}^{H}\sum_{j=1}^{W}\left(I_{ij} - I_{ij}^{'}\right)^2, \tag{13}$$

where I_{ij} and $I^{'}_{ij}$ are the pixel values of the cover and stego-images, respectively. However, our scheme, Chang et al.'s schemes and Yang and Lin's scheme are all VQ-based reversible data hiding schemes. Therefore, the visual quality for all four schemes is exactly the same.

Because our proposed scheme has nine scan paths, three test images, "Baboon", "Tiffany" and "F16", were selected to show the possible embedding rate and the compression rate with the nine scan paths, as shown in **Table 1**.

Table 1. Comparisons of different scan paths in ER and CR with our scheme

Images / Types of scan path	Baboon		Tiffany		F16	
	ER	CR	ER	CR	ER	CR
Scan path 1	0.81	0.58	1.86	0.37	1.48	0.45
Scan path 2	0.79	0.59	1.81	0.38	1.52	0.43
Scan path 3	0.70	0.60	1.78	0.39	1.40	0.47
Scan path 4	0.68	0.61	1.77	0.39	1.39	0.47
Scan path 5	0.79	0.59	1.86	0.37	1.49	0.44
Scan path 6	0.81	0.59	1.81	0.38	1.48	0.45
Scan path 7	0.82	0.59	1.86	0.37	1.53	0.43
Scan path 8	0.74	0.60	1.78	0.39	1.44	0.45
Scan path 9	0.74	0.60	1.81	0.38	1.42	0.46

Table 1 makes it clear that scan path 7 is the best path in our scheme because it achieves an average compression rate of 0.46 bpp and an average embedding rate of 1.4 bpi. Moreover, scan path 7 also works well with complex images. As for smooth images, there are three scan paths, paths 1, 5 and 7 that can achieve the same average compression rate (0.37 bpp) and average embedding rate (1.86 bpi). Therefore, scan path 7 is used with our scheme to compare with three existing schemes in the rest experiments.

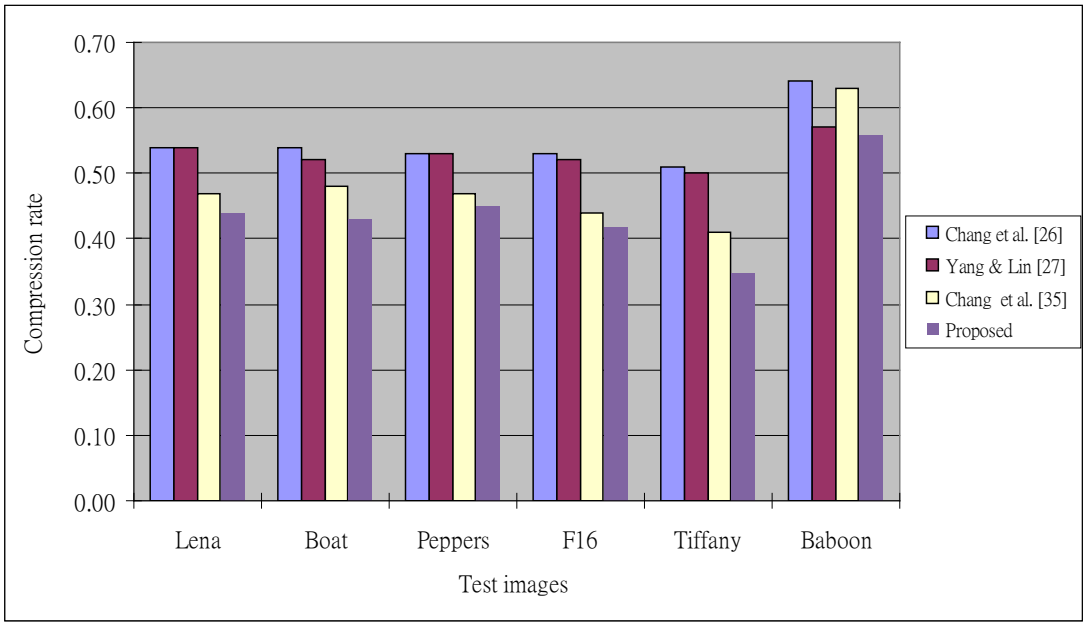


Fig. 9. Comparison of compression rate for our proposed scheme using scan path 7 in **Fig. 3(g)**, and for Chang et al.'s schemes and Yang and Lin's scheme

Fig. 9 shows the compression performance of our proposed scheme, Chang et al.'s schemes [26, 35] and Yang and Lin's scheme [27]. Clearly, our scheme outperforms the other two schemes in compression rate. The average compression rate of our scheme is 0.44 bpp. By contrast, the average compression rate of Chang et al.'s scheme [35], Yang and Lin's scheme [27], and Chang et al.'s scheme [26] is only 0.48bpp, 0.53 bpp, and 0.55 bpp, respectively. Our proposed scheme's superior compression rate results from encoding the difference value of two adjacent indices instead of the original index value. In most cases, each adjacent index value generated by our proposed computing adjacent indices' values phase can be encoded by using fewer bits. For example, if the absolute difference value of two adjacent indices d is "0", the difference value is encoded by indicator 01, with two secret bits concatenated. Only 4 bits are required for this case. In contrast, if the absolute value of the adjacent index difference value equals 20, our scheme must use 11 bits to encode the current adjacent index difference (i.e., 00||0||00010100). Luckily, a large absolute value of the adjacent index difference occurs infrequently. It only occurs when the content of an image has significant variation. Therefore, our scheme offers a better compression rate for smooth images than for complex images. Take the test images "Baboon" and "Tiffany", for example. With our scheme the compression rate for "Baboon" is larger than that for "Tiffany" because "Tiffany" is smoother than "Baboon". Note that for the complex image "Baboon", the compression rate with our scheme remains 0.56 bpp, which is still lower than the rates achieved by Yang and Lin's scheme [27] (0.57 bpp), Chang et al.'s scheme [35] (0.63 bpp), or Chang et al.'s scheme [26] (0.64 bpp).

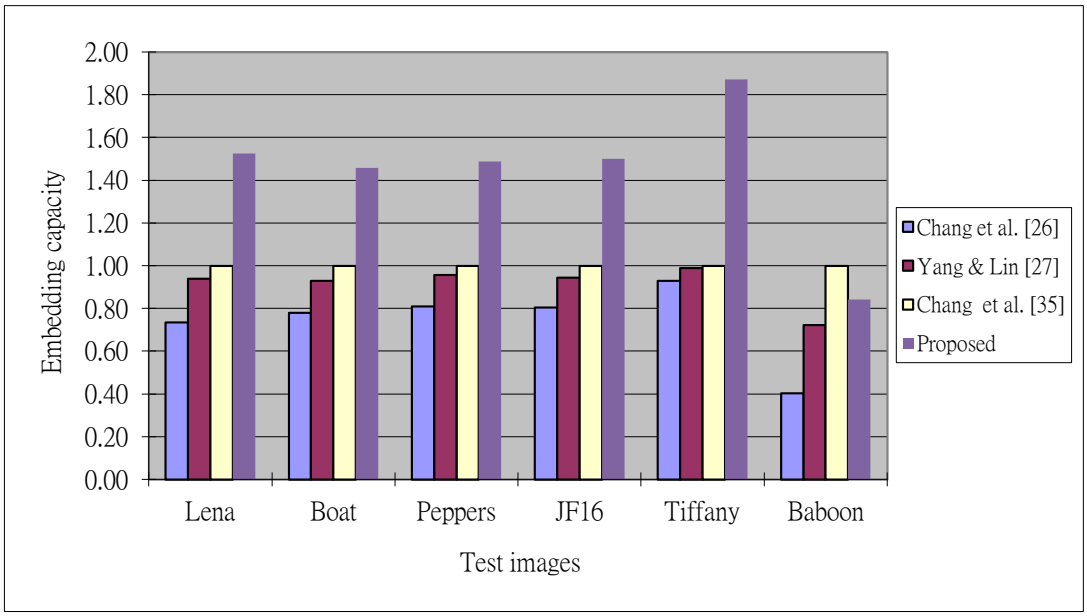


Fig. 10. Embedding rate results of our proposed scheme with scan path 7 of **Fig. 3(g)** compared with the two existing schemes

Fig. 10 compares embedding performance for our scheme, Chang et al.'s schemes [26, 35] and Yang and Lin's scheme [27]. Clearly, our scheme's embedding rate is significantly higher than the others. The average embedding rate of our scheme is approximately 1.45 bpi, followed by Chang et al.'s scheme [35] (1.00 bpi) and Yang and Lin's scheme [27] (0.91 bpi). The worst embedding rate is Chang et al.'s scheme [26] (0.74 bpi). Our scheme's embedding rate is superior because two bits can be embedded in most adjacent index differences.

5. Conclusion

This paper proposes a novel reversible data hiding based on adjacent indices' differences. The average compression rate of our proposed scheme is 0.44 bpp, which is superior to Yang and Lin's scheme (0.53 bpp) or Chang et al.'s scheme (0.54 bpp). Moreover, our scheme's compression rate is also better than that offered by traditional VQ compression schemes (i.e., 0.5 bpp for a codebook sized 256). In addition, the average embedding rate of our scheme can be as much as 1.45 bpi, which outperforms Chang et al.'s scheme [35] (1.00 bpi), Yang and Lin's scheme [27] (0.91 bpi), and Chang et al.'s scheme [26] (0.74 bpi). Finally, the visual quality of our scheme is as good as Yang and Lin's and Chang et al.'s schemes (*PSNR*>30dB). Therefore, the overall superior performance offered by our scheme enables us to conclude that it is efficient and flexible enough to be applied directly in some applications such as digital libraries or online multimedia transmission.

References

[1] R. M. Davis, "The data encryption standard in perspective," *IEEE Commun. Mag.*, vol. 16, no. 6, pp. 5-9, 1978. Article (CrossRef Link)
[2] R. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signature and public key

cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp 120-126, 1978. Article (CrossRef Link)
[3] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, no. 2, pp. 4-29, 1984. Article (CrossRef Link)
[4] W. Bender, N. Morimoto, A. Lu, "Technique for data hiding," *IBM Systems Journal*, vol. 35, pp. 313-336, 1996. Article (CrossRef Link)
[5] C. K. Chan, L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognition*, Vol. 37, pp. 469-474, 2004. Article (CrossRef Link)
[6] C. K. Chan, L. M. Cheng, "Improved hiding data in images by optimal moderately significant-bit replacement," *IEEE Electronic Letters*, vol. 37, pp. 1017-1018, 2001. Article (CrossRef Link)
[7] K. L. Chung, C. H. Shen, L. C. Chang, "A novel SVD and VQ-based image hiding scheme," *Pattern Recognition*, vol. 2, pp. 1051-1058, 2001. Article (CrossRef Link)
[8] L. M. Marvel, C. G. Boncelet, C. T. Retter, "Spread spectrum image steganography," *IEEE Transactions on Image Processing*, vol. 8, pp. 1075-1083, 1999. Article (CrossRef Link)
[9] [9] J. Mielikanen, "LSB matching revisited," *IEEE Signal Processing Letter*, vol. 13, pp. 285-297, 2006. Article (CrossRef Link)
[10] C. C. Thien, J. C. Lin, "A simple and high hiding capacity method for hiding digit-by-digit data in images based on modulus function," *Pattern Recognition*, vol. 36, pp. 2875-2881, 2003. Article (CrossRef Link)
[11] R. Z. Wang, C.F. Lin, J. C. Lin, "Hiding data in images by optimal moderately significant-bit replacement," *IEEE Electronic Letters*, vol. 36, pp. 2069-2070, 2000. Article (CrossRef Link)
[12] R. Z. Wang, C.F. Lin, J. C. Lin, "Image hiding by optimal LSB substitution and generic algorithm," *Pattern Recognition*, vol. 34, pp. 671-683, 2001. Article (CrossRef Link)
[13] X. Zhang, S. Wang, "Steganography using multiple-base notational system and human vision sensitivity," *IEEE Signal Processing Letters*, vol. 12, pp. 67-70, 2005. Article (CrossRef Link)
[14] A. M. Alattar, "Reversible watermarking using the difference expansion of a generalized integer transform," *IEEE Transactions on Image Processing*, vol. 13, pp. 1147-1156, 2004. Article (CrossRef Link)
[15] M. U. Celik, G. Sharma, A.M. Tekalp, E. Saber, "Reversible data hiding," in *Proc. of IEEE Int. Conf. on Image Processing*, pp. 157-160, 2002. Article (CrossRef Link)
[16] J. Fridich, M. Goljian, R. Du, "Lossless data embedding - new paradigm in digital watermarking," *EURASIP Journal on Applied Signal Processing*, pp. 185-196, 2002. Article (CrossRef Link)
[17] J. Fridich, M. Goljian, R. Du, "Invertible authentication," in *Proc. of SPIE Photonics West, Security and Watermarking of Multimedia Contents III, San Jose, CA*, pp. 197-208, 2001. Article (CrossRef Link)
[18] M. Goljan, J. Fridrich, R. Du, "Distortion-free data embedding," in *Proc. of 4th Information Hiding Workshop, Pittsburgh, PA*, pp. 27-41, April 2001. Article (CrossRef Link)
[19] C. W. Honsinger, P. Jones, M. Rabbani, J. C. Stoffel, "Lossless recovery of an original image containing embedded data," *US Patent 6278791 B1*, August 21, 2001. Article (CrossRef Link).
[20] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, "Reversible data hiding," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 16, pp. 354-362, 2006. Article (CrossRef Link)
[21] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, pp. 890-896, 2003. Article (CrossRef Link)
[22] C. D. Vleeschouwer, J. F. Delaigle, B. Macq, "Circular interpretation of histograms for reversible watermarking," in *Proc. of IEEE International Multimedia Signal Processing Workshop France*, pp. 345–350, October 2001. Article (CrossRef Link)
[23] H. C. Huang, F. H. Wang, J. S. Pang, "Efficient and robust watermarking algorithm with vector quantization," *Electron. Lett.*, vol. 37, no. 13, pp.826-828, 2001. Article (CrossRef Link)
[24] H. C. Huang, F. H. Wang, J. S. Pan, "A VQ-based robust multi-watermarking algorithm," *IEICE Trans. Fundam.*, vol. E-85A, no. 7, pp.1719-1726, 2002. Article (CrossRef Link)
[25] C. Y. Lin, C. C. Chang, "Hiding data in VQ-compressed images using dissimilar pairs," *J. Comput.*, vol. 17, no. 2, pp. 3-10, 2006. Article (CrossRef Link)
[26] C. C. Chang, W. C. Wu, and Y. C. Hu, "Lossless recovery of a VQ index table with embedded secret data," *Journal of Visual Communication and Image Representation*, vol. 18, no. 3, pp.

207-216, June 2007. Article (CrossRef Link)
[27] C. H. Yang and Y. C. Lin, "Reversible data hiding of a VQ index table based on referred counts," *Journal of Visual Communication and Image Representation*, vol. 20, no. 6, pp. 399-407, August 2009. Article (CrossRef Link)
[28] W. J. Chen, and W. T. Huang, "VQ indices compression and information hiding using hybrid lossless index coding," *Digital Signal Processing*, vol. 19, no. 3, pp. 433-443, May 2009. Article (CrossRef Link)
[29] C. C. Chang, W. L. Tai, C. C. Lin, " A reversible data hiding scheme based on side match vector quantization," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 16, no. 10, pp. 1301-1308, 2006. Article (CrossRef Link)
[30] C. C. Chang, T. D. Kieu, and Y. C. Chou, "Reversible information hiding for VQ indices based on locally adaptive coding," *Journal of Visual Communication and Image Representation*, vol. 20, no. 1, pp. 57-64, Jan. 2009. Article (CrossRef Link)
[31] C. Y. Lin, C. C. Chang, "Hiding data in VQ-compressed images using dissimilar pairs," *J. Comput.*, vol. 17, no. 2, pp. 3-10, 2006. Article (CrossRef Link)
[32] C. C. Chang, C. Y. Lin, Y. H. Fan, "Lossless data hiding for color images based on block truncation coding," *Pattern Recognition*, vol. 41, no. 7, pp. 2347-2357, July 2008. Article (CrossRef Link)
[33] C. C. Lin, P. F. Shiu, "DCT-based reversible data hiding scheme," *Journal of Software*, vol. 5, no. 2, pp. 214-224, 2010. Article (CrossRef Link)
[34] S. F. Chiou, Y. C. Lu, I. E. Liao, M. S. Hwang, "An efficient reversible data hiding scheme based on SMVQ," *The Imaging Science Journal*, Vol. 61, No. 6, pp. 467-474, July 2013. Article (CrossRef Link)
[35] C. C. Chang, T. S. Nguyen, and C. C. Lin, "A reversible data hiding scheme for VQ indices using locally adaptive coding," *The Journal of Systems and Software*, Vol. 86, pp. 389-402, 2013. Article (CrossRef Link)

Chin-Chen Chang received the B.S. degree in applied mathematics and the M.S. degree in computer and decision sciences from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1977 and 1979, respectively. He received the Ph.D degree in computer engineering from National Chiao Tung University, Hsinchu, in 1982. From July 1998 to June 2000, he was Director of the Advisory Office, Ministry of Education, R.O.C. From 2002 to 2005, he was a Chair Professor at National Chung Cheng University. From February 2005, he has been a Chair Professor at Feng Chia Univeristy. In addition, he was severed as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression, and data structures.

Thai-Son Nguyen received the bachelor degree in information technology from Open University, HCM city, Vietnam, in 2005. From December 2006, he has been lecturer of TraVinh University, TraVinh, Vietnam. In 2011, he received M.S. degree in computer sciences from FengChia University, TaiChung, Taiwan. He is currently pursuing the Ph.D. degree with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan. His current research interests include data hiding, image processing, database security and information security.

Chia-Chen Lin (M'08) received the B. S. degree in information management from the Tamkang University, Taipeo, Taiwan, R.O.C., in 1992. She received the M.S. degree in information management and the Ph.D. degree in information management from Chiao Tung University, Hsinchu, Taiwan, in 1994 and 1998, respectively. She was a Visiting Associate Professor at Business School, University Illinois at Urbana Champain, during August 2006 to July 2007. She is currently a Professor in the Department of Computer and Information Management, Providence University, Sha-Lu, Taiwan. Her research interests include image and signal processing, image data hiding, mobile agent, and electronic commerce.