Lossless Information Hiding in the VQ Index Table

Chin-Chen Chang
Feng Chia University, Taichung, Taiwan
Email: alan3c@gmail.com

[1]Kuo-Nan Chen, [2]Zhi-Hui Wang and [2,3]Ming-Chu Li
[1]National Chung Cheng University, Chiayi, Taiwan
[2] Dalian University of Technology, Dalian, China
Email: [1]ckn95p@gmail.com; [2]wangzhihui1017@yahoo.cn; [3]li_mingchu@yahoo.com

***Abstract*—In this paper, a reversible data hiding scheme based on a Vector Quantization (VQ) index table is proposed. To satisfy different needs of users, our proposed scheme is designed with flexibility in adjusting hiding capacity and compression rate. First, the codebook is rearranged referring to the index occurrence frequency in the VQ index table. After that, the codewords in the newly generated codebook are clustered into a number of groups for the usage of embedding secret digits in our proposed scheme. Note that, the decision of group numbers determines the capability of hiding capacity and compression rate of a to-be-embed image in our proposed scheme. Based on the well-defined embedding strategies, the experimental results show that the performance of our proposed scheme outperforms the scheme proposed by Yang and Lin in 2009.**

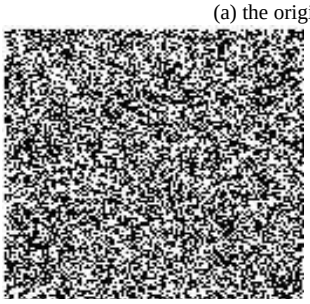
***Index Terms*—vector quantization, reversible, bit rate, hiding capacity, data hiding**

I. INTRODUCTION

With the rapid improvement of computer and networking technologies, it becomes more and more popular that people exchange information (including digital images) with each other via the Internet. The convenient communication ways are accompanied by problems of information security and the usage of networking bandwidth. Several researches in cryptography have been proposed, such as AES [15], DES [16] and RSA [17], for minimizing the information security problem caused by the public properties of the Internet. Research in cryptography guarantees the privacy of the transmitted data by transforming the transmitted data from plaintext (meaningful) to ciphertext (meaningless) with keys. That is, the original ciphertext is hard to be decrypted without the appropriate keys. In this way, the transmitted data can be transmitted more securely via the Internet. However, the meaningless ciphertext is singular, and it is easy to attract attention of malicious attackers. Hence, several data hiding schemes have been proposed to overcome the singular problems, such as secret sharing (meaningful shares) [1] [6], steganographic technology [5] [13] [19], and reversible data hiding technologies [4] [8]. Researches mentioned above aim to embed secret message into cover images, and makes the secert message embedded images (stego images) indistinguishable from the original cover images. Therefore, the stego images are hard to gain attention by potential attackers. The differences of the secret embedded results between the cryptography and data hiding technology based on images are shown in Fig. 1.

The usage of networking bandwidth is also a serious problem due to the huge transmission of high quality images. To solve these problems, many data compression schemes are proposed, and they can be classified into lossless schemes [11] and lossy schemes [14]. Among these compression schemes for digital images, the vector quantization (VQ) compression scheme, which was first proposed by Gray in 1984 [7], is an efficient compression scheme that has a high bit rate and satisfactory image

(a) the original image



(b) the final result by cryptography

(c) the final result by data hiding technologies

Figure 1. The differences of final results between cryptography and data hiding technology based on images.

doi:10.4304/jsw.8.3.547-553

quality. Based on the advances of VQ, many researchers aim to apply lossless data hiding schemes on VQ images [2] [18] [20]. In this way, the users can get a compact secret embedded result with the ability to recover the original cover images.

In this paper, in terms of cryptography and image compression concepts, we propose a reversible data hiding scheme that has high hiding capacity and a satisfactory compression rate based on VQ images. In addition, the original VQ images can be losslessly recovered while the secret information is completely extracted. The remainder of this paper is organized as follows. In Section 2, the VQ compression technology is reviewed, and, in Section 3, the proposed scheme is illustrated. In Section 4, the experimental results, which show the effectiveness of our proposed scheme, are presented. Finally, our conclusions are presented in Section 5.

II. RELATED WORKS

In this section, the VQ compression technology is briefly reviewed. VQ was first proposed by Gray in 1984 [7]. The concept of VQ is that it uses an index number to represent a set of non-overlapping pixels to compress the original image. The details are described as follows. Before the compressing, a codebook is needed, and it can be generated by various codebook generation algorithms. Among these algorithms, the LBG algorithm, which was proposed by Linde, Buzo, and Gray in 1980 [12], is one of the most frequently used algorithms, and it is illustrated as follows. To generate a codebook sized c using the LBG algorithm, a set of representative images is selected and divided into non-overlapping blocks, called block set B, in which each block is sized $m \times n$. For training a codebook sized c, a set of blocks $P1 = \{p1_i \mid i = 0,1,...,c-1\}$ is randomly selected from block set B, and $P1$ is treated as the initial centroid point set for grouping B into c clusters to form the cluster set $C1 = \{c1_i \mid i = 0,1,...,c-1\}$. For cluster set $C1$, a new centroid point set $P2 = \{p2_i \mid i = 0,1,...,c-1\}$ is generated by finding the centroid point $v2_i$ from $c1_i$. Subsequently, $P2$ is used for grouping B into a new cluster set $C2 = \{c2_i \mid i = 0,1,...,c-1\}$. The finding centroid point set and grouping new cluster set processes take turns until the centroid point set $Pk = \{pk_i \mid i = 0,1,...,c-1\}$ is stable compared with the previous one. Then, a codebook sized c, in which each codeword is an $m \times n$-dimensional vector, is generated. After that, the generated codebook Pk is used for helping the VQ compression processes described as follows. To compress an image I sized $M \times N$, the to-be-compressed image is first divided into $(M \times N / m \times n)$ non-overlapping blocks, where $m \times n$ is the size of each block. Every $m \times n$ pixels in a block can be treated as a $m \times n$-dimensional vector and replaced with a one-dimensional vector (index number) by searching for the closest codeword in the codebook Pk. The efficiency of VQ can be measured by bit rate, which has the units of bits per pixel, as defined in Equation 1. Here, $bit\_rate = \frac{\lceil \log_2 c \rceil}{m \times n}$.

$$bit\_rate = \frac{\text{Compression Code (bits)}}{\text{The image size (pixels)}}. . \quad (1)$$

For instance, to compress an image size 12×12 with a codebook sized 256, where each codeword in the codebook is a sixteen-dimensional vector, first, the to-be-compressed image is divided into $(12 \times 12)/(4 \times 4)$ non-overlapping blocks, in which each block is sized 4×4. For every sixteen pixels in a block, the sixteen values are replaced with an index number where the codeword it refers to is closest to these sixteen pixels compared with the other codewords in the codebook. The example of a VQ encoder is shown in Fig. 2. After all the blocks are replaced with index numbers, the process of the VQ encoder is completed, and the compressed result is called the VQ index table. To decompress a VQ index table into a VQ image, the index value in the VQ index table is replaced with the sixteen-dimensional vector (codeword) according to the codebook and arranges the sixteen-dimensional vector into a 4×4 block. In this way, each index value is expanded by sixteen values. After all index values are expanded, the VQ index table can be decompressed to obtain the VQ image. The example of VQ decoder is shown in Fig. 3.

III. THE PROPOSED SCHEME

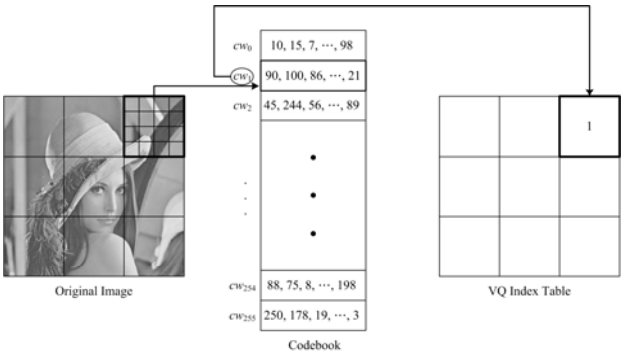


Figure 2. The example of VQ encoder.

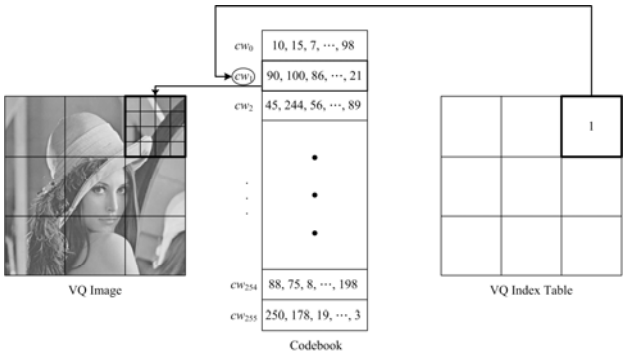


Figure 3. The example of VQ decoder.

In the proposed scheme, the secret information is encoded with the VQ index table according to the frequency of occurrence of the index values. At the receiving end, the secret information can be extracted exactly, and the original VQ image can also be reversed. The proposed scheme includes two procedures, i.e., the embedding procedure and the extracting and recovering procedure, and they are illustrated in Subsections 3.1 and 3.2, respectively.

A. Embedding Procedure

To embed secret information into a VQ index table, first, the corresponding codebook is rearranged from high to low according to the index occurrence frequency in the to-be-embedded VQ index table. After the rearranged codebook $R=\{r_i \mid i=0 \text{ to } c-1\}$ sized c has been generated, it is clustered sequentially into G groups. Note that c must be multiples of G. Afterwards, the secret information is transformed into base-(G-1) equivalence (uses the digits 0 to G-2) for further processing. To embed the base-(G-1) equivalence secret information into the VQ index table, it has to reference an index mapping table that is created as follows. For each index r_j in the first group, where $j=0 \text{ to } \frac{c}{G}-1$, it is mapped with an un-embeddable index $r_{(c-1)-(j\times G)-S}$ with the base-(G-1) secret digit S, where S = 0 to G-1, as shown in Figure 4. That is, to embed secret information into embeddable indices (in the first group), the embeddable indices are replaced with the un-embeddable index values according to the index mapping table. On the other hand, if an un-embeddable index occurs in the VQ index table, it would be replaced with an embeddable index by checking the index mapping table with some indicator bits appended at the end of it for recovery purposes. The indicator bits for each un-embeddable index are generated by transforming the corresponding base-G secret digits into $\lceil \log_2 G \rceil$ bits binary representation.

Example 1: In the rearranged codebook $R=\{r_i \mid i=0 \text{ to } 255\}$, c = 256, G = 4, assume that the base-4 secret digits are (0, 2) and that the original index values are (0, 1, 65, 255). First, the codebook is clustered into four groups, and the indices in the first group are r_0, r_1, …, r_{63}. For the first original index 0 to embed the corresponding secret digit 0, according to the embedding rules discussed above, the un-embeddable index 255 is used to replace index 0 for embedding. For the second original index 1, since the corresponding to-be-embedded secret digit is 2, it is replaced with the un-embeddable index 250. For the third original index 65, since it is an un-embeddable index (out of the first group), it is replaced with the embeddable index 63 with two ($\lceil \log_2 4 \rceil$) binary bits 1 and 0 appended to the end of it. For the final original index 255, since it also belongs to un-embeddable index, it is replaced with the embeddable index 0 with two binary bits 0 and 0 appended to the end of it. The embedding process of Example 1 is shown in Fig. 5.

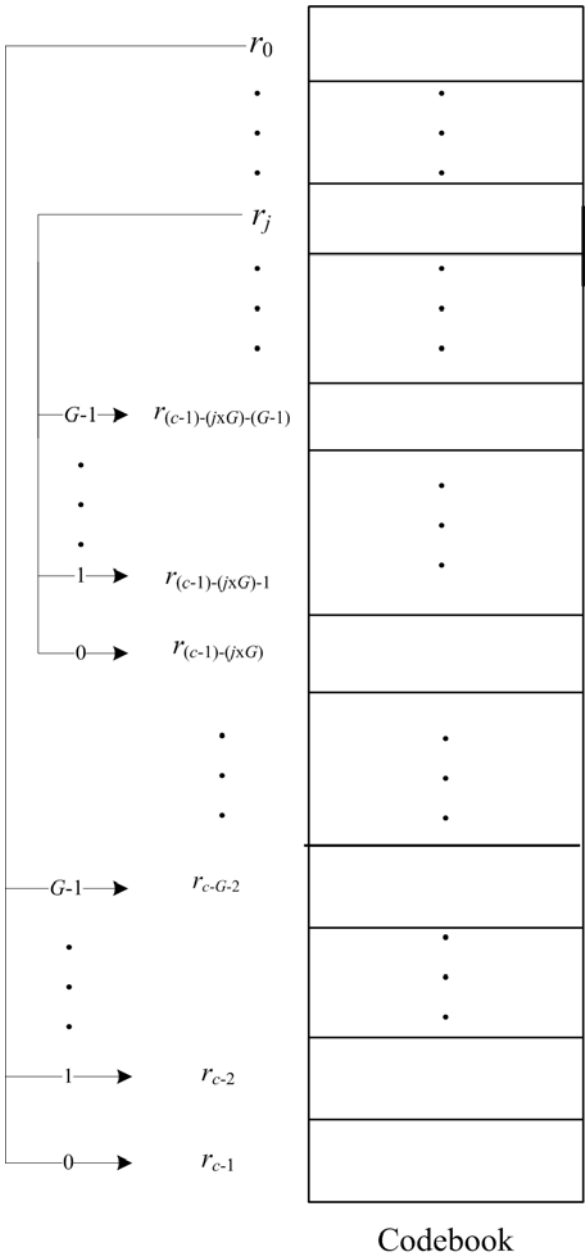


Figure 4. The index mapping table.

0	1
65	255

(a) the original index table

255	250
$(63)_{10}\|\|(10)_2$	$(0)_{10}\|\|(00)_2$

(b) the final index table with secret digits embedded

Figure 5. The embedding processes of Example 1.

B. Extracting and Recovering Procedure

At the receiving end, the user only has to know the parameter of G; then, he/she can extract the embedded secret digits, and the original VQ index table can also be reversed as described below. The final codestream that the receiver gets is decoded by every $\lceil \log_2 c \rceil$ bits first, where c is the size of the codebook. Similar to the embedding procedure, the codebook is divided into G groups, and the decoding processes fall into two cases:

Case 1: the index value V transformed from the $\lceil \log_2 c \rceil$ bits falls into the first group

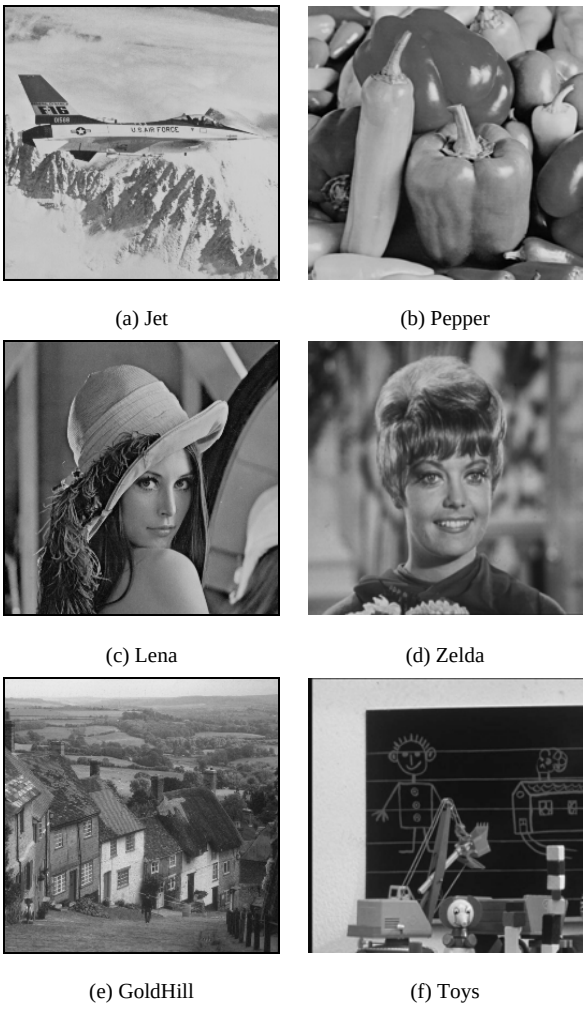
Case 2: the index value V transformed from the $\lceil \log_2 c \rceil$ bits is out of the first group

In Case 1, the original index can be reversed by replacing the decoded index with the corresponding un-embeddable index according to the index mapping described as follows. The following $\lceil \log_2 G \rceil$ bits of V are decoded as the indicators. After transforming $\lceil \log_2 G \rceil$ bits into base-G digits, the recovered index can be mapped by treating these transformed results as to-be-embedded secret digits in the index table, which is shown in Fig. 4. Note that the index mapping table can be created by the receiver since he/she knows parameters G and c. In Case 2, the decoded index value can be reversed into the original index value by substituting the corresponding embeddable index value, and the base-G secret digits can be extracted concurrently according to the index mapping table.

Example 2: As in Example 1, assume that the final embedded result is shown as Fig. 5(b), G = 4, c = 256, and the index mapping table is shown as Fig. 4. Since the first eight bits extracted from the final codestream whose decimal value equals 255, which is an un-embeddable index value, this means that it is transformed from an embeddable index value with secret digits embedded. Therefore, the decoding procedure falls into Case 2, so the original index value 0 is reversed, and the base-4 secret digit 0 is extracted by mapping Fig. 4. For the next eight extracted bits whose decimal value equals to 250, the decoding process also falls into Case 2. Similarly, the original index value 1 is reversed, and the base-4 secret digit 2 is extracted according to the index mapping table. For the next eight extracted bits whose decimal value equals to 63, it belongs to an embeddable index, and the decoding process falls into Case 1, such that the original index value 65 is recovered by transforming the following next two bits, 1 and 0 into base-4 digits, and then mapping it to the index mapping table. Next, the following eight bits are extracted whose decimal value equals to 0. Here, the decoding process falls into Case 2. Similarly, the original index value 255 is reversed according to the index mapping table. Finally, the base-4 secret digits (0, 2) are extracted exactly, and the original VQ index table is totally recovered as shown in Fig. 5(a).

IV. EXPERIMENTAL RESULTS

Six VQ images sized 512×512, Jet, Pepper, Lena, Zelda, GoldHill, and Toys, which are shown in Figure 6, are used as text images in the experimental results to show the performance of our proposed scheme. Each codeword of the codebook in the following experiments is a sixteen-dimensional vector.



(a) Jet (b) Pepper

(c) Lena (d) Zelda

(e) GoldHill (f) Toys

Figure 6. Test images.

Table I shows the hiding capacity in bits and the bit rate of our proposed scheme, with the parameter G equal to 4, 8, and 16, and the size of the codebook is 256. Table II shows the same things except that the codebook size is

TABLE I
THE HIDING CAPACITY AND THE BIT RATE OF OUR PROPOSED SCHEME (CODEBOOK SIZE = 256)

Images \ G		4	8	16
Jet	hiding capacity	21790	34219	42562
	bit_rate (bpp)	0.520	0.548	0.584
Pepper	hiding capacity	20431	27810	27145
	bit_rate (bpp)	0.527	0.574	0.644
Lena	hiding capacity	18755	23727	21482
	bit_rate (bpp)	0.535	0.591	0.665
Zelda	hiding capacity	20910	26920	24078
	bit_rate (bpp)	0.524	0.578	0.656
Goldhill	hiding capacity	18082	23935	22484
	bit_rate (bpp)	0.538	0.590	0.662
Toys	hiding capacity	23182	37301	46180
	bit_rate (bpp)	0.513	0.535	0.570

TABLE II.
THE HIDING CAPACITY AND THE BIT RATE OF OUR PROPOSED SCHEME (CODEBOOK SIZE = 512)

Images \ *G*		4	8	16
Jet	hiding capacity	21209	32292	38542
	bit_rate (bpp)	0.585	0.618	0.662
Pepper	hiding capacity	19024	25620	25387
	bit_rate (bpp)	0.596	0.646	0.713
Lena	hiding capacity	18125	21989	19492
	bit_rate (bpp)	0.600	0.660	0.736
Zelda	hiding capacity	20579	26089	24164
	bit_rate (bpp))	0.588	0.644	0.718
Goldhill	hiding capacity	17737	22216	21304
	bit_rate (bpp)	0.602	0.659	0.729
Toys	capacity	23008	37601	47724
	bit_rate (bpp)	0.577	0.597	0.626

TABLE III.
COMPARISON BETWEEN OUR PROPOSED SCHEME AND THE SCHEME PROPOSED BY YANG AND LIN

Images	Yang and Lin's scheme				Proposed scheme			
	Hiding capacity		*bit_rate* (bpp)		Hiding capacity		*bit_rate* (bpp)	
	No swap	With swap	No swap	With swap	*G*=8	*G*=16	*G*=8	*G*=16
Jet	28840	31010	0.595	0.588	34219	42562	0.548	0.584
Pepper	28818	30172	0.609	0.608	27810	27145	0.574	0.644
Lena	26980	29874	0.647	0.644	23727	21482	0.591	0.665
Zelda	27304	32190	0.654	0.618	26920	24078	0.578	0.656
Goldhill	23594	28850	0.697	0.686	23935	22484	0.590	0.662
Toys	30692	31790	0.573	0.572	37301	46180	0.535	0.570
Averages	27705	30650	0.629	0.619	28985	30655	0.569	0.630

512. It is obvious that the bit rate here is just slightly degraded compared with the VQ compression whose bit rate equals to 0.5 when the codebook size is 256 (Table I) and 0.5625 when the codebook size is 512 (Table II). By observing Tables I and II, it can be seen that the hiding capacity increases as parameter *G* increases, but the bit rate is degraded comparatively. To further show the performance of our proposed scheme, we compared it to Yang and Lin's scheme [21], and the comparison results are shown in Table III. By observing Table III, it is obvious that our proposed scheme outperforms Yang and Lin's scheme in both hiding capacity and bit rate.

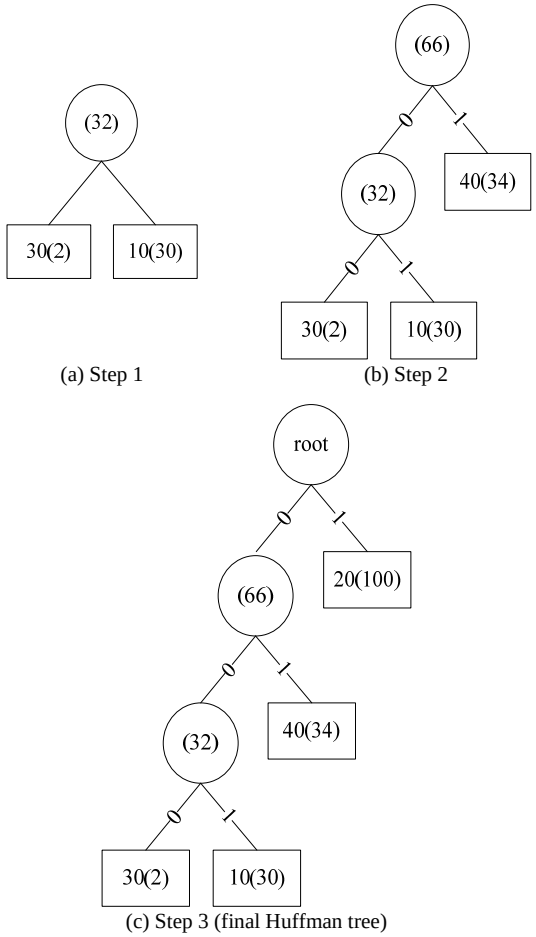
V. ANALYSIS

In this paper, we proposed a reversible image hiding scheme based on VQ images. Basically, VQ is an image compression scheme, and the concept of this paper is that we try to analyze the characteristic of the compression code (VQ index table) and embed the secret bits to it. For satisfying some stricter requirements and special applications, the size of the compression result (VQ index table) can be further reduced by using some suitable compression scheme such as Huffman coding [9] and search order coding [10]. They are briefly introduced as follows.

A. Huffman coding

In 1952, Huffman proposed Huffman codes which are widely used for compression data. The basic concept of it is to use longer bits to represent infrequent symbols and shorter bits to represent frequent symbols. We use an example to illustrate how to generate Huffman codes. Assume an index table IT contains four different index values: 10, 20, 30, and 40, and the frequencies of occurrence of them are 30, 100, 2, and 34, respectively. First, these four different index values are sorted according to frequencies of occurrence, and the result is 30(2), 10(30), 40(34), and 20(100), where the number in the parentheses is frequency of occurrence. After sorting, a Huffman tree (binary tree) is going to be created for later encoding. Note that, each different index values are treated as a leaf node. Step 1: the two least-frequent nodes (30 and 10) are merged into a new node where its frequency of occurrence is the sum of child nodes, and the left branch is labeled as 0 and 1 for the right branch as shown in Figure 6(a). Step 2: continue merging the two least-frequent and the result is shown in Figure 6(b). Step 3: merge the last two nodes, and the Huffman tree is built completed which is shown in Figure 6(c).

The codeword table which is generated according to the Huffman tree (Figure 6(c)) is listed in Figure 7. Assume the codebook size used here is 256, then the size of VQ index table $8\times(2+30+34+100)=1328$ bits can be reduced to $(3\times2+3\times30+2\times34+1\times100)=264$ bits. However, since Huffman codes is variable-length code, how to embed secret bits in it according to our proposed



(a) Step 1 (b) Step 2

(c) Step 3 (final Huffman tree)

Figure 6. The processes of building Huffman tree.

scheme is our future work.

B. Search order coding

In 1996, Hsieh and Tsai proposed Search order coding (SOC) to compress the VQ index table. The authors found that, for an index value, an identical index value can be found frequently around of it. On the basis of this concept, we can try to search an identical index value in the left, top left, top, and top right directions for a to-be-compressed index value. Since these four positions can be represented with two bits, the original index value is replaced by the position code if an identical index value is found. Otherwise, the index value is incompressible and keeps its value unchanging. Note that, an addition bit is required to inform that the processing compressed code is position code or original index value. If we are hard to find an identical index value by using four positions, we can expand the position numbers to eleven, which is shown as Figure 8, where the gray parts are processed, and the white parts are unprocessed. Note that the first column of VQ index table is incompressible, and the processing order is from left to right, and top to bottom. The index values compressed by using SOC can be classified into two cases: compressible and uncompressible. Each of them is fixed-length code, so it is workable for applying our proposed scheme to it and therefore it can satisfy with the users who want to have smaller size of results and also want to embed some secret bits in it.

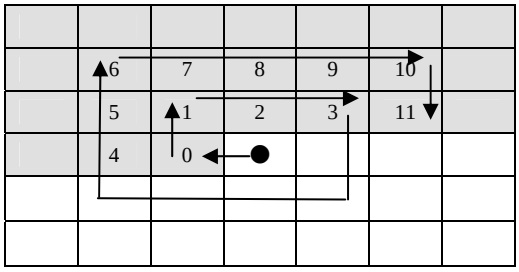


Figure 7. The search order for SOC.

VI. CONCLUSIONS

We propose a reversible image hiding scheme based on VQ images which has the flexibility in adjusting the data hiding capacity and the compression rate. The flexibility makes the proposed scheme suitable for more applications, including users who want high hiding capacity and those who want a compact embedded result. In the future, we will try to design a more powerful reversible image hiding scheme that can further reduce the size expansion of the embedded results.

Index value	**Codeword**
10	001
20	1
30	000
40	01

Figure 7. The codeword table which is generated according to the Figure 6(c).

REFERENCES

[1] C. C. Chang, C. C. Lin, C. H. Lin, and Y. H. Chen, "A novel secret image sharing scheme in color images using small shadow images," *Information Sciences*, vol. 178, no. 11, pp. 2433–2447, June 2008.

[2] 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, 2007.

[3] Y. F. Chen, Y. K. Chan, C. C. Huang, M. H. Tsai, and Y. P. Chu, "A multiple-level visual secret-sharing scheme without image size expansion," *Information Sciences*, vol. 177, pp. 4696–4710, 2007.

[4] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless Generalized-LSB Data Embedding," *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 253–265, February 2005.

[5] W. Y. Chen, "Color image steganography scheme using DFT, SPIHT codec, and modified differential phase-shift keying techniques," *Applied Mathematics and Computation*, vol. 196, no. 1, pp. 40–54, February 2008.

[6] Z. Eslami, S.H. Razzaghi, and J. Zarepour Ahmadabadi, "Secret image sharing based on cellular automata and steganography," *Pattern Recognition*, vol. 43, pp. 397–404, 2010.

[7] R. M. Gray, "Vector quantization," *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 4–29, 1984.

[8] Y. A. Ho, Y. K. Chan, H. C. Wu, Y. P. Chu, "High-capacity reversible data hiding in binary images using pattern substitution," *Computer Standards & Interfaces*, vol. 31, no. 4, pp. 787–794, June 2009.

[9] C. H. Hsieh and J. C. Tsai, "Lossless Compression of VQ Index with Search-Order Coding," *IEEE Transactions on Image Processing*, vol. 5, no. 1, pp. 1579–1582, November 1996.

[10] D. A. Huffman, "A Method for the Construction of Minimum Redundancy Codes," *Proceedings of the Institute of Radio Engineers*, vol. 40, pp. 1098–1101, September 1952.

[11] A. B. Hussein, "A Novel Lossless Data Compression Scheme Based on the Error Correcting Hamming Codes," *Computers & Mathematics with Applications*, vol. 56, no. 1, pp. 143–150, July 2008.

[12] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, pp. 84–95, January 1980.

[13] C. L. Liu and S. R. Liao, "High-performance JPEG steganography using complementary embedding strategy," *Pattern Recognition*, vol. 41, no. 9, pp. 2945–2955, September 2008.

[14] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of Multivariate Mixed Data via Lossy Coding and Compression," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 9, pp. 1546–1562, September 2007.

[15] National Institute of Standards & Technology, "Announcing the advanced encryption standard (AES) ," *Federal Information Processing Standards Publication*, vol. 197, no. l, 2001.

[16] National Institute of Standards & Technology, "Data encryption standard (DES)," *Federal Information Processing Standards Publication*, vol. 46, January 1977.

[17] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.

[18] P. Tsai, "Histogram-based reversible data hiding for vector quantization-compressed images," *IET Image Processing*, vol. 3, no. 2, pp. 100–114, 2009.

[19] C. M. Wang, N. I. Wu, C. S. Tsai, and M. S. Hwang, "A high quality steganographic method with pixel-value differencing and modulus function," *Journal of Systems and Software*, vol. 81, no. 1, pp. 150–158, January 2008.

[20] J. X. Wang and Z. M. Lu, "A path optional lossless data hiding scheme based on VQ joint neighboring coding," *Information Sciences*, vol. 179, no. 19, pp. 3332–3348, September 2009.

[21] 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.

Chin-Chen Chang received his Ph.D in computer engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. During the academic years of 1980-1983, he was on the faculty of the Department of Computer Engineering at the National Chiao Tung University. From 1983-1989, he was on the faculty of the Institute of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan. From August 1989 to July 1992, he was the head of, and a professor in, the Institute of Computer Science and Information Engineering at the National Chung Cheng University, Chiayi, Taiwan. From August 1992 to July 1995, he was the dean of the college of Engineering at the same university. From August 1995 to October 1997, he was the provost at the National Chung Cheng University. From September 1996 to October 1997, Dr. Chang was the Acting President at the National Chung Cheng University. From July 1998 to June 2000, he was the director of Advisory Office of the Ministry of Education of the R.O.C. From 2002 to 2005, he was a Chair Professor of National Chung Cheng University. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures.

Kuo-Nan Chen received his BS degree in Information Engineering and Computer Science from Feng Chia University in 2002, and the MS degree in Graduate Institute of Educational Measurement and Statistics from National Taichung University in 2006. Currently, he is a Ph.D. student in the department of Computer Science and Information Engineering of National Chung Cheng University, Taiwan. His research interests in image data hiding technologies.

Zhi-Hui Wang received the BS degree in software engineering in 2004 from the North Eastern University, Shenyang, China and the MS degree in software engineering in 2007 from the Dalian University of Technology, Dalian, China. She is currently pursuing her PhD degree in computer software and theory from the Dalian University of Technology, Dalian, China. Her research interests include data hiding, and image processing.

Ming-Chu Li received a Ph.D. degree in computer science from the University of Toronto,Toronto, Canada in 1998. His research interests include Hamiltonian Graph Theory, NP-Theory and Algorithms, Network and Information Security, Reputation Systems,and Grid computing and its applications. During 1997-2002, he worked as a system software Engineer in north america, where he helped in the design and implementation of algorithms and the structures of projects. In 2002, he was a Full Professor of Computer Science at Tianjin University (Tianjin, China). In 1993, he was a Full Associate Professor at the University of Science and Technology Beijing (Beijing, China). Prof. Li is currently a Full Professor of Computer Science at DaLian University of Technology (DLUT) (Dalian, China), where he has been since September 2004. He is also Vice Dean of School of Software of DLUT.