International Journal of Innovative Computing, Information and Control
Volume **9**, Number **6**, June **2013**

ICIC International ©2013 ISSN 1349-4198
pp. **2521–2531**

HYBRID MATRIX CODING AND ERROR-CORRECTION CODING SCHEME FOR REVERSIBLE DATA HIDING IN BINARY VQ INDEX CODESTREAM

Junxiang Wang[1,2], Jiangqun Ni[1,*] and Zheming Lu[3,*]

[1]School of Information Science and Technology
Sun Yat-Sen University
No. 135, West Xingang Road, Guangzhou 510275, P. R. China
wjx851113851113@yahoo.cn; *Corresponding author: issjqni@mail.sysu.edu.cn

[2]School of Mechanical and Electronic Engineering
Jingdezhen Ceramic Institute
Jingdezhen 333403, P. R. China

[3]School of Aeronautics and Astronautics
Zhejiang University
No. 38, Zhada Road, Xihu District, Hangzhou 310027, P. R. China
*Corresponding author: zheminglu@zju.edu.cn

Received March 2012; revised July 2012

Abstract. *In this paper, we propose a novel reversible data hiding framework to hide secret data into a binary codestream of VQ indices invertibly, in which matrix encoding is used to efficiently embed secret data, and error correction coding (ECC) is employed for lossless restoration of the marked codestream at the receiver side. Due to the unchanged structure of the codestream during the embedding process, our scheme can significantly improve the security of our scheme. The use of ECC, however, leads to a decreased compression ratio. As a lossless compression method for VQ index tables, search-order coding is used as a remedy to improve the compression performance. According to the experimental results, it is testified that the proposed scheme achieves a high stego-image quality, a large embedding capacity, a high transmission efficiency and an acceptable compression performance. Moreover, our scheme achieves the relatively high performance in terms of the security.*

Keywords: Reversible data hiding, Matrix encoding, (7,4) Hamming code, VQ index

1. **Introduction.** To ensure the security of data transmission and avoid malicious attacks, data hiding techniques have been widely developed, which embed secret data into cover images with imperceptible degradation. However, the distortions of cover medium, caused by the data embedding process in traditional data hiding schemes, are often inevitable and irreversible. In some special fields such as military, medical and forensic fields, it is crucial to restore the original cover image without any distortion. Thus, reversible data hiding algorithms have been proposed, which can completely restore the cover image without any distortions after extraction of the secret data.

Most existing reversible data hiding schemes are classified into three categories, i.e., the spatial domain [1,2], transform domains [3-5], and compressed domains [6-18]. In order to save the storage space and transmission bandwidth for stego-images, the vector quantization (VQ) technique, as an important image compression algorithm due to the characteristics of easy implementation and high compression ratio, is often applied for reversible data hiding [6-18]. In our opinion, the existing VQ based reversible data hiding schemes can also be divided into three categories, such as data hiding during the VQ encoding process [6,7], data hiding on the index table [8-12] and data hiding on the binary codestream of VQ indices [13-18]. For the first category, two similar codewords with small Euclidean distance would be constructed by a criterion to encode the input pixel block according to the secret data content '0' or '1'. Yang et al. [6] used the nearest adjacent block or another similar adjacent one to encode the current block by using the least distortion criterion between the selected adjacent block and the mean value of four adjacent blocks. Chang et al. [7] found the best-match codeword from the state codebook of SMVQ or an approximate codeword to encode each input block. The criterion is shared with both sender and the receiver. The inverse restoration was implemented to recover the original image. There are two shortcomings for these schemes. Due to the difference between two similar codewords for each block encoding, the accumulative distortions will gradually increase. Thus, these methods usually degrade the stego-image quality. In addition, the security level for these schemes is somewhat low owing to no separation of the encoding and data embedding processes as mentioned in [6].

For the second category, the secret data embedding process was performed directly on the VQ index table to generate a marked VQ index table, which meant the index values were changed by some rules to other values and meanwhile concatenated by some flag bits to distinguish different cases. For these algorithms [8-12], the VQ indices were partitioned into several groups by declustering techniques and the mapped relationship between the corresponding indices in different groups were established. When the embedding process was implemented, one index value can be kept unchanged or changed to the mapped value to hide secret data '0' or '1'. To denote different cases, the flag bits should be added as the indicator. Among the schemes, literatures [8,9] utilized the minimum-spanning tree, short-spanning path method and principal components analysis (PCA) for declustering, respectively, while, literatures [10,11] employed the frequency of the occurrence of the index values, namely the referred counts for declustering. In addition, Tsai [12] utilized the histogram shifting technique to determine the mapping relationship between VQ indices and then performed the data embedding process. There is a main shortage for these schemes in terms of the security. For these schemes, the added indicator leads to the change in the structure of the index table, which may induce unwanted suspicion from the attackers and thus reduce the level of security.

For the third type, secret data were embedded into the binary codestream of the VQ indices and then transmitted to the receiver. In [13], Chang et al. proposed a novel joint neighboring coding (JNC) approach to encode the VQ indices into a binary form and hided message into the codestream during the JNC process. Later, Lu et al. [14] and Wang and Lu [15] improved the JNC-based scheme [13] by choosing different start points and paths to achieve a large capacity and a high embedding efficiency. Yang et al. [16] further improved the scheme in [14] at the aspect of the compression ratio by using the Huffman code. During the period, Chang et al. [17] and Yang and Lin [18] proposed the other two reversible schemes based on the locally adaptive coding technique (LAC), respectively. However, since the coding methods employed in above mentioned schemes, i.e., JNC and LAC are not commonly used before, the special decoder for these codes should be individually designed, which may inevitably induce the suspicion and thus degrade the security level.

In our paper, we propose a novel reversible data hiding framework based on the common communication channel, which employs the efficient matrix encoding to hide secret data into the binary codestream of the VQ indices and utilizes the error-correction encoding to recover the stego-codestream at the receiver side. The advantages of our paper can be summary into three points. 1) Efficient matrix encoding can greatly increase the embedding capacity and thus ensures a high transmitting efficiency. 2) Commonly used error-correction encoding utilized for restoration of the marked codestream could not lead to any suspicion, which improves the security level of our scheme. 3) Due to the separation of the encoding/embedding process and data embedding process by employment of both matrix encoding and Error-Correction coding, our scheme could achieve a higher security level.

The rest of the paper is organized as follows. Section 2 briefly reviews the concepts of VQ and the basic framework for reversible data hiding techniques in the compression. Section 3 presents the proposed scheme and the associated techniques in detail. Section 4 provides experimental results and evaluates the performance of our proposed scheme. Finally, we draw the conclusion in Section 5.

2. **Related Works.**

2.1. **Vector quantization and the corresponding index table generation.**

Vector quantization (VQ) is a commonly used lossy data compression technique whose process can be mentioned as follows. First, VQ generates a representative codebook Y from a number of training vectors using the LBG clustering algorithm [19], where $Y = \{\boldsymbol{y_i};\ i = 1, 2, \ldots, N\}$ and $\boldsymbol{y_i}$ is called a codeword. In the VQ encoding phase, the image to be encoded is divided into non-overlapping $r \times s$-sized blocks, where $r \times s = t$, and each block as a t-dimensional input vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_t)$ is compared with each codeword in the codebook to find the best matching codeword by using the squared Euclidean distance as follows.

$$d(\boldsymbol{x}, \boldsymbol{y_i}) = \sum_{j=1}^{t} (x_j - y_{ij})^2 \quad (i = 1, 2, \ldots, N) \tag{1}$$

where x_j is the j-th component of the input vector $\boldsymbol{x}$, and y_{ij} is the j-th component of the codeword $\boldsymbol{y_i}$.

Finally, the VQ index of the best matching codeword with the minimum squared Euclidean distance is chosen to encode the corresponding block and transmitted to the receiver. For example, if the codeword $\boldsymbol{y_i}$ is chosen, the subscript i is acquired and transmitted as the corresponding VQ index. For an $R \times S$-sized grayscale image, after each block has been encoded by the corresponding VQ index, an $(R/r) \times (S/s)$-sized index table is generated. Obviously, in comparison with transmitting the original image, it is more efficient to transmit the VQ indices in the aspect of saving the bandwidth. In the VQ decoding stage, the received indices are used to look up the corresponding codewords from the same codebook so that the input blocks can be reconstructed. It is worthwhile to note that the specially designed indicators in the embedding process will make the designed VQ decoder different from the conventional one. Thus the security level of these schemes, such as [8-12], is decreased.

2.2. **Basic framework of reversible data hiding in compressed domains.**

Yang et al. [6] presented a basic framework of reversible data hiding in compressed domains and then proposed a reversible data hiding scheme based on MFCVQ encoding. In this sub-section, we only focus on the framework as follows.

As shown in Figure 1, I_0 is the original image, E is a lossy encoder containing a quantizer, and C is the code stream, resulting from E, as the carrier for reversible data hiding. D is a decoder corresponding to E. From Figure 1, we can see that the embedder and extractor are on the both sides of the channel.

According to [6], the design rules relevant to the framework for reversible data hiding are given as follows.

1) *Reversibility*: the perfect restoration of C and w, i.e., $C = C_R$ and $w = w_r$;

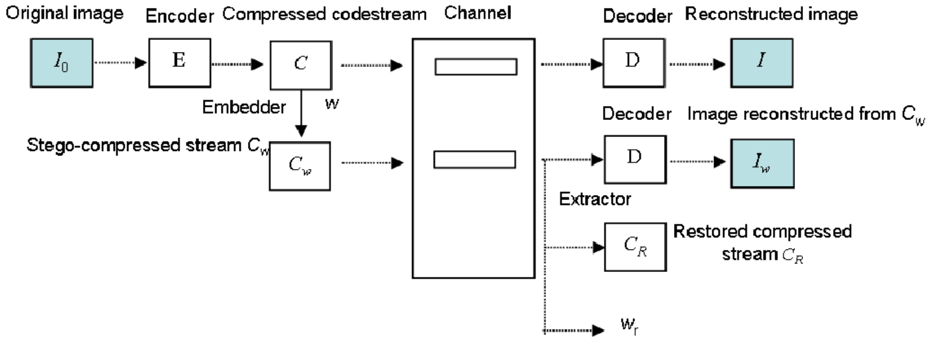

FIGURE 1. Framework of reversible data hiding in the compressed domain

2) *Fidelity*: the reconstructed image I_w should be similar to its original version I_0 without perceptual distortions;

3) *Indistinguishability*: the same decoder D could not tell the difference between stego-codestream C_w and its conventional version C. Specifically, the embedder E should guarantee the same size and structure for the C_w and C without adding any specially designed indicators.

To increase the security of the framework, another two rules are offered.

4) *Compatibility*: the encoder E and decoder D should be as compatible with existing lossy compression schemes (VQ) as possible;

5) *Separation of the encoder and the embedder*: this rule facilitates the individually processing of the lossy image coding and data hiding.

In our paper, the security of VQ based schemes could be measured based on the aforementioned five criterions.

3. The Proposed Scheme.

In the section, we propose a common framework for reversible data hiding in a binary codestream of the VQ indices. The proposed scheme features the hybrid matrix encoding and error-correction encoding with $(7, 4)$ Hamming code, where the matrix encoding is utilized to achieve efficient data embedding and the error-correction coding (ECC) is adopted for the lossless restoration. Since the adoption of ECC leads to an increased stego-codestream length, SOC technique [20] can be alternatively utilized in the embedding process to improve the compression performance. In this section, two related techniques, i.e., matrix encoding and ECC, are briefly reviewed first. Then the embedding and extraction processes are provided.

3.1. Matrix encoding.

Matrix encoding was proposed by Crandall [21] to enhance the embedding efficiency by decreasing the number of required changed bits. Later, Westfeld [22] implemented the matrix encoding in the famous steganographic scheme (F5). Matrix encoding is generally used in the least significant bits (LSBs) of the coefficients for steganography, e.g., quantized DCT coefficients [22]. In our paper, the high embedding efficiency motivates us to apply matrix encoding to hide secret data into the binary codestream of the VQ indices.

The matrix encoding with Hamming code $(1, n, k)$ $k \geq 3$ can embed k-bit message into an n-bit binary codeword by changing at most 1 bit, where $n = 2^k - 1$. The embedding process divides the binary codestream C and message data w, into n-bit sized and k-bit sized segments respectively. Let CS and WS denote the i-th cover codestream and message segments respectively, where $CS = (cb_{n(i-1)+1}, \cdots, cb_{ni})$ and $WS = (w_{k(i-1)+1}, \cdots, w_{ki})$. The matrix encoding process is preformed as follows.

To embed k-bit message WS into n-bit codestream segment CS, the flip bit position α in CS is determined by

$$(\alpha)_2 = WS \oplus b(CS) \tag{2}$$

where '$\oplus$' means the bit-wise XOR operation, $(\alpha)_2$ is the k-bit sized binary form of the decimal value α and the function $b(\bullet)$ is defined as follows.

$$b(CS) = \bigoplus_{j=1}^{n} [(cb_{n(i-1)+j}) \times (j)_2] = (cb_{n(i-1)+1} \times (1)_2) \oplus \cdots \oplus (cb_{n(i-1)+n} \times (n)_2) \tag{3}$$

where $cb_{n(i-1)+j}$ is a binary bit from $CS = (cb_{n(i-1)+1}, \cdots, cb_{ni})$, and $(i)_2$, $(i = 1, 2 \cdots, n)$ is the k-bit sized binary output of the decimal value i. Therefore, $b(CS)$ is the k-bit sized binary output.

If $\alpha \neq 0$, the α-th bit in the segment of CS should be flipped, i.e., '1' to '0' or '0' to '1', to hide secret data WS. Therefore, the stego-codestream segment SS is then obtained by

$$SS = \begin{cases} CS, & \text{if } \alpha = 0 \\ cb_{n(i-1)+1}, \dots, \overline{cb_{n(i-1)+\alpha}}, \dots, cb_{ni} & \text{if } \alpha \neq 0 \end{cases} \tag{4}$$

where $\overline{cb_{n(i-1)+\alpha}}$ means the flipping operation on $cb_{n(i-1)+\alpha}$.

On the decoder side, the k-bit message WS is extracted from an n-bit sized SS as follows

$$WS = b(SS) \tag{5}$$

The embedding rate r for Hamming code $(1, n, k)$ is defined by

$$r = \frac{k}{n} = \frac{k}{2^k - 1} \tag{6}$$

It is obvious that the embedding rate r decreases with the increase of the parameter $k\,(k \geq 3)$. To obtain a high data hiding capacity, the matrix encoding with the Hamming code $(1, 7, 3)$ is adopted in our paper.

3.2. **Error-correction coding.**

Error-correction coding is an error control system for data transmission. During the system, the sender adds systematically generated redundant bits to the message bits, which allows the receiver to detect and correct a limited number of errors occurring everywhere in the message. In accordance with the Hamming code $(1, 7, 3)$ used in matrix encoding, the $(7, 4)$ Hamming code is utilized in our paper for lossless restoration of the cover codestream and introduced as follows.

The $(7, 4)$ Hamming code (HC) as a detailed case encodes 4 bits of data to a 7-bit codeword by adding 3 parity check bits. In this way, $(7, 4)$ Hamming code can correct any single-bit error, or detect all single-bit and two-bit errors. Therefore, when only one bit is flipped by matrix encoding for data hiding, the bit can be corrected by the $(7, 4)$ Hamming code. For a 4-bit data $D = (d_1, d_2, d_3, d_4)$, the $(7, 4)$ Hamming encoding is performed to generate a 7-bit codeword $D_G = (q_1, q_2, d_1, q_3, d_2, d_3, d_4)$ with 3-bit additional parity check data q_i $(i = 1, 2, 3)$ by

$$D_G = \mathbf{G}D = \text{mod2}\left(\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix}\right) = (q_1, q_2, d_1, q_3, d_2, d_3, d_4)^{\mathrm{T}} \tag{7}$$

where $\mathbf{G}$ is the generator matrix for $(7, 4)$ Hamming code and the function mod2$(\bullet)$ is the modulo-2 operation.

When the binary data D is encoded by $(7, 4)$ Hamming code, the generated codeword D_G instead of D would be transmitted to receiver. Let the received data be $D'_G =$

$(q_1', q_2', d_1', q_3', d_2', d_3', d_4')^{\mathrm{T}}$. If any correctable error occurs during the transmission process, the parity check and error correction operation should be performed by the receiver as follows.

$$Z = \mathbf{H}D_G' = \mathrm{mod}2\left(\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}(q_1', q_2', d_1', q_3', d_2', d_3', d_4')^{\mathrm{T}}\right) \tag{8}$$

where $\mathbf{H}$ is the parity-check matrix for the $(7, 4)$ Hamming code.

Z in (8) is referred as a syndrome. If $Z = (0, 0, 0)^{\mathrm{T}}$, no error occurs during the transmission process and we have $D_G = D_G'$. When $Z = (z_1, z_2, z_3)^{\mathrm{T}} \neq (0, 0, 0)^{\mathrm{T}}$, the syndrome (z_1, z_2, z_3) is considered as a binary form and transformed to a decimal value, denoted as z, which indicates the z-th bit was corrupted. Thus, an error has been detected in the codestream and can be corrected (simply flipped) to recover D_G. For example, if $Z = (1, 0, 1)^{\mathrm{T}}$ and $D_G{}' = (0, 1, 1, 0, 1, 1, 1)^{\mathrm{T}}$, we can judge the 5-th bit has been corrupted and then correct $D_G{}'$ to be $D_G = (0, 1, 1, 0, \overline{1}, 1, 1)^{\mathrm{T}} = (0, 1, 1, 0, 0, 1, 1)^{\mathrm{T}}$.

According to the corrected $(7, 4)$ Hamming codeword D_G, the original data D can be extract by

$$\begin{aligned} D &= (\mathrm{E,0}) \cdot D_G \\ &= \mathrm{mod}2\left(\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot (q_1, q_2, d_1, q_3, d_2, d_3, d_4)^{\mathrm{T}}\right) \\ &= (d_1, d_2, d_3, d_4)^{\mathrm{T}} \end{aligned} \tag{9}$$

In our proposed scheme, we embed 3-bit secret message into each 7-bit Hamming codeword and then continuously transmit them to the receiver.

3.3. **Embedding process.** For an $R \times S$ sized 8-bit grayscale cover image I_0, the message w and the codebook Y with N codewords, denoted as $Y = \{y_i;\ i = 1, 2, \ldots, N\}$, the embedding process based on the codestream of VQ indices is illustrated as follows and shown in Figure 2.

1) Generate the index table IT. Partition the cover image I_0 into $r \times s$ sized non-overlapped blocks and then calculate the corresponding VQ index value of each block as mentioned in Subsection 2.1 to generate a VQ index table IT;

2) Transform the index table IT into the binary codestream C. For different purposes, two approaches can be alternatively adopted as follows;

2.1) Traditional VQ. Scan the index table IT in the raster scan order and generate an index array $P = \{p_i | i \in [1, (R/r) \times (S/s)]\}$, where each index value p_i is represented with its $\lceil \log_2 N \rceil$-bit binary version. Then, the index array P is transformed into a binary form, denoted as C;

2.2) SOC_VQ. The index table IT is firstly scanned to obtain the index array P same as Step 2.1 and then each index value p_i is encoded by using the SOC method to generate the binary codestream C;

3) Embed the message w into the codestream C. Divide C into 4-bit sized segments. For each segment CS, $(7, 4)$ Hamming coding is performed as shown in Subsection 3.2 and thus a 7-bit codeword CS_G is generated. Then matrix encoding with Hamming code $(1, 7, 3)$ is then performed on CS_G to hide 3-bit secret data WS and generate the stego-codestream segment SS, namely one bit of CS_G is flipped to hide 3-bit secret data as

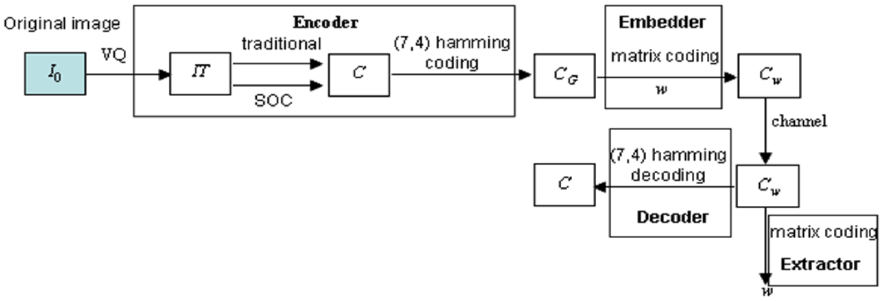

FIGURE 2. Flow chart of the proposed scheme

described in Subsection 3.1. When all the segments of C have been processed, the stego-codestream C_w is generated, which is then transmitted to the receiver. In addition, to enhance the security, w can be encrypted by a secret key before the data hiding process.

3.4. **Extracting process.** With the received stego-codestream C_w, the embedded message w is extracted and the original codestream is recovered from C_w as follows.

1) Extract the message w. Partition the marked codestream C_w into 7-bit sized segments. For each stego-segment SS, 3-bit secret data WS can be extract by Equation (5). When all the stego-segments have been processed, each extracted 3-bit secret data can be concatenated to form the whole message w.

2) Restore the cover codestream C. Partition the marked codestream C_w into 7-bit sized segments. For each stego-segment SS of 7-bit, at most 1 bit is flipped for 3-bit message embedding. Then SS is corrected to its original version CS_G by using Equation (8). Finally, the 4-bit sized D, which denotes the VQ indices, can be extracted from CS_G by using Equation (9). After all the stego-segments have been restored, the original codestream C is recovered.

Finally, it is noted that, when the capacity of message w is small, only a portion of the codestream C is coded with Hamming coding to enhance the compression ratio.

4. **Main Results.** In this section, we present the experimental results for six typical $8 \times 512 \times 512$-bit graylevel images to evaluate the performance of our scheme in terms of embedding capacity, stego-image quality, compression ratio, embedding rate and the security.

In the experiment, the size of each non-overlapping image block is set to be 4×4, the size of codebook is denoted as N, the number of neighborhood indices for SOC m is set to be 8. Meanwhile, the VQ codebook is trained by using the well-known LBG algorithm [19]. Meanwhile, the embedding capacity and Stego-image quality are measured by parameter *capacity* and Peak signal-to-noise ratio ($PSNR$), respectively. The compression performance is denoted by the parameter CB, which is defined by

$$CB = \frac{L}{R \times S} = \frac{L}{512 \times 512} \tag{10}$$

where L is the number of total bits of the output codestream.

In addition, the embedding rate is measured by the parameter r, which is defined to evaluate how many bits of secret data can be embedded under the same output codestream length as follows.

$$r = \frac{capacity}{CB \times R \times S} = \frac{capacity}{L} \tag{11}$$

It is noted that the scheme with high embedding rate is preferred for practical applications. Finally, the security will be individually discussed according to the criterions as mentioned in Subsection 2.2.

4.1. **The performances of our proposed scheme under different situations.** In this subsection, we evaluate the performance of our proposed scheme under different situations. The results are summarized in Table 1.

As shown in Table 1, the stego-images have the same visual quality as the ones with the VQ encoding according to $PSNR$, which is expected due to the "reversible" nature. According to Table 1, the capacity and compression performance for the Tra_VQ encoder are the same for different test images on the same codebook size N. Reason for the phenomenon is that the encoding and data hiding processes for the codestream in our scheme are regular, namely each 4-bit binary segment is encoded with $(7, 4)$ Hamming code and then utilized to embed 3-bit secret data by using matrix encoding. Therefore, when $N = 256$, $CB = \frac{8}{4\times4} \times \frac{7}{4} = 0.875$ and $capacity = CB \times R \times S \times \frac{3}{7} = CB \times 512 \times 512 \times \frac{3}{7} = 98304$. With the increase of the codebook size N, the length of VQ index codestream increases, which leads to the raise of embedding capacity (*capacity*) and compression performance (CB) for both Tra_VQ and SOC_VQ. The same regularity is observed for the SOC_VQ situation. The only difference between Tra_VQ and SOC_VQ encoders with the same codebook size is that SOC_VQ compresses the VQ index table and produces a decreased VQ index codestream.

Finally, due to the same embedding structure, the embedding rate is the same for the proposed scheme under different situations, e.g., the embedding rate $r = 3/7 = 0.4286$ for (7, 4) Hamming code as shown in Table 1.

Table 1. Performances of our proposed scheme under different situations

codebook size	Parameter		Lena	Peppers	Mandrill	Boat	Goldhill	F16
$N = 256$	VQ encoding quality (*PSNR*)		30.328	29.800	23.224	29.157	30.288	30.753
	Stego-image quality (*PSNR*)		30.328	29.800	23.224	29.157	30.288	30.753
	Tra_VQ	*capacity* (bit)	98304	98304	98304	98304	98304	98304
		CB (bpp)	0.875	0.875	0.875	0.875	0.875	0.875
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286
	SOC_VQ	*capacity* (bit)	63898	62620	79430	62226	63603	61735
		CB (bpp)	0.5687	0.5574	0.7070	0.5539	0.5661	0.5495
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286
$N = 512$	VQ encoding quality (*PSNR*)		31.216	30.561	23.887	30.038	31.231	31.608
	Stego-image quality (*PSNR*)		31.216	30.561	23.887	30.038	31.231	31.608
	Tra_VQ	*capacity* (bit)	110592	110592	110592	110592	110592	110592
		CB (bpp)	0.9844	0.9844	0.9844	0.9844	0.9844	0.9844
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286
	SOC_VQ	*capacity* (bit)	73217	73880	91574	74205	73430	72329
		CB (bpp)	0.6517	0.6576	0.8151	0.6605	0.6536	0.6438
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286
$N = 1024$	VQ encoding quality (*PSNR*)		32.037	31.285	24.069	30.788	32.061	32.345
	Stego-image quality (*PSNR*)		32.037	31.285	24.069	30.788	32.061	32.345
	Tra_VQ	*capacity* (bit)	122880	122880	122880	122880	122880	122880
		CB (bpp)	1.09375	1.09375	1.09375	1.09375	1.09375	1.09375
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286
	SOC_VQ	*capacity* (bit)	86880	86140	104080	87610	87120	80980
		CB (bpp)	0.7733	0.7667	0.9264	0.7798	0.7755	0.7208
		r	0.4286	0.4286	0.4286	0.4286	0.4286	0.4286

4.2. **Comparisons with other schemes.** In this subsection, our proposed scheme with SOC_VQ encoder is compared with several latest published VQ based reversible data hiding schemes [11,13,15,16].

As shown in Table 2, all the schemes can perfectly reconstruct stego-images with the same quality as the corresponding VQ compressed images. With the efficient matrix encoding, our proposed scheme achieves the highest embedding capacity among all the schemes in Table 2. It is observed, however, that the proposed scheme has no advantage in compression performance according to the parameter CB. That is because the adoption of $(7,4)$ Hamming coding in our scheme almost doubles the length of its output codestream. Meanwhile, other schemes employed different compression methods during the index encoding process, such as SMVQ [11], joint neighboring coding [13,15] and Huffman coding [16], which further increases the gap of compression performance between our scheme and other schemes. It is fortunate that, by taking advantage of SOC_VQ, our scheme achieves an accepted compression performace compared with other schemes. Obviously, there is a tradeoff between the embedding capacity (*capacity*) and compression performance (CB), which means that a better compression performance leads to a shorter length of output codestream and thus results in a smaller embedding capacity.

To fairly compare the effects of the two tradeoff metrics, the metric of embedding rate (r) is defined. Figure 3 shows the performance of embedding rate *vs.* test images for different schemes, which demonstrates that our proposed scheme significantly outperforms other schemes, which means proposed scheme can embed most secret data when the output of the same length is transmitted.

Finally, the issue on the security is discussed. Since scheme [11] hided the message into the index table or its codestream by ether directly changing the index values or adding some specially designed flag bits in front of the index values, the structure of VQ index table was modified, which required a specifically designed decoder to recover the stego-codestream and the modified index table. Therefore, scheme [11] is not compatible with the design rule 3 and 4 as mentioned in Subsection 2.2 and their security levels are low. While for the schemes [13,15,16], the specifically designed joint neighboring code (JNC) for encoding the index values, which was not commonly used, leaded to some suspicion from the attackers. Thus, these schemes [13,15,16] do not meet the design rule 4 and also have a low security level. By comparison, our proposed scheme employs commonly used

Table 2. Comparisons between several VQ-based reversible data hiding schemes

Algorithm	Parameter	Lena	Peppers	Mandrill	Boat	Goldhill	F16
Proposed with SOC_VQ ($N = 512$, $m = 8$)	VQ encoding quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	Stego-image quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	capacity (bit)	73217	73880	91574	74205	73430	72329
	CB (bpp)	0.6517	0.6576	0.8151	0.6605	0.6536	0.6438
Scheme [11] ($N = 512$)	Stego-image quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	capacity (bit)	46560	48240	32676	34856	34006	47908
	CB (bpp)	0.500	0.500	0.573	0.509	0.5132	0.509
Scheme [13] ($N = 512$)	Stego-image quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	capacity (bit)	16129	16129	16129	16129	16129	16129
	CB (bpp)	0.582	0.572	0.659	0.584	0.590	0.577
Scheme [15] ($N = 512$)	Stego-image quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	capacity (bit)	32004	32004	32004	32004	32004	32004
	CB (bpp)	0.641	0.632	0.691	0.665	0.664	0.649
Scheme [16] ($N = 512$)	Stego-image quality (*PSNR*)	31.216	30.561	23.887	30.038	31.231	31.608
	capacity (bit)	19492	20779	16882	21179	19796	23400
	CB (bpp)	0.537	0.532	0.599	0.534	0.554	0.498

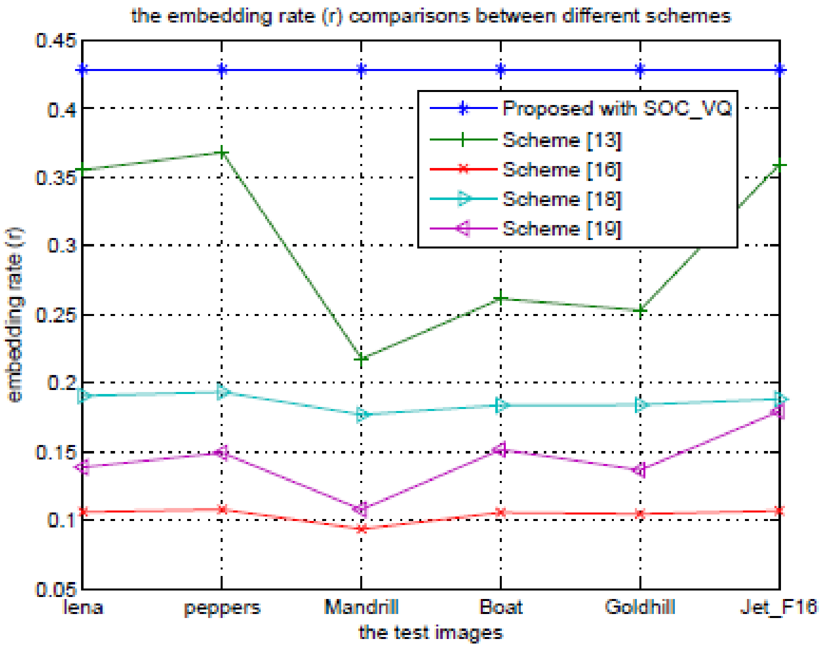

Figure 3. Embedding rate comparisons between different schemes

error-correction coding and matrix encoding for data hiding and thus meets the design rule 3 and 4 as mentioned in Subsection 2.2. Therefore, our scheme obtains an improved security level.

5. Conclusion.

This paper presents a novel reversible data hiding framework based on VQ index coderstream by using hybrid matrix encoding and error-correction coding ($(7, 4)$ Hamming coding). The proposed scheme utilizes the matrix encoding to embed secret data into the VQ index codestream and error-correction coding to losslessly recover the marked codestream in receiver side, respectively. With the high efficient matrix encoding, our scheme achieves a larger capacity and higher embedding rate than other VQ based schemes. Meanwhile, Hamming code technique, as the commonly used unit for data transmission, could not raise any suspicion and thus enhances the security level. In addition, the separation of VQ encoding and data hiding processes in our scheme can further improve the security level. Finally, it is noted that our proposed framework can be extended to other compressed codestreams, e.g., JPEG and JPEG2000.

Acknowledgment. This work is supported by National Natural Science Foundation of China (No. 60970145 and No. 61171150), National Research Foundation for the Doctoral Program of Higher Education of China (No. 20120171110037), the Key Program of Natural Science Foundation of Guangdong (No. S2012020011114) and Education Department Science and Technology Project of Jiangxi Province (No. GJJ12512).

REFERENCES

[1] C. D. Vleeschouwer, J. F. Delaigle and B. Macq, Circular interpretation of bijective transformations in lossless watermarking for media asset management, *IEEE Transactions on Multimedia*, vol.5, no.1, pp.97-105, 2003.

[2] M. U. Celik and A. M. Tekalp, Lossless generalized-LSB data embedding, *IEEE Transactions on Image Processing*, vol.14, no.2, pp.253-266, 2005.

[3] J. Fridrich, M. Goljan and R. Du, Invertible authentication watermark for JPEG images, *Proc. of the 2th IEEE Conf. on Information Technology: Coding and Computing*, Las Vegas, USA, pp.223-227, 2001.

[4] C. C. Chang, C. C. Lin, C. S. Tseng and W. L. Tai, Reversible hiding in DCT-based compressed images, *Information Sciences*, vol.177, no.13, pp.2768-2786, 2007.

[5] I. Usman, A. Khan and A. Ali, Reversible watermarking based on intelligent coefficient selection and integer wavelet transform, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(A), pp.4675-4682, 2009.

[6] B. Yang, Z. M. Lu and S. H. Sun, Reversible watermarking in the VQ-compressed domain, *Proc. of the 5th IASTED International Conference on Visualization, Imaging, and Image Processing*, Benidorm, Spain, pp.298-303, 2005.

[7] C. C. Chang, W. L. Tai and C. C. Lin, A reversible data hiding scheme based on side match vector quantization, *IEEE Transactions on Circuits and Systems for Video Technology*, vol.16, no.10, pp.1301-1308, 2006.

[8] C. C. Chang, Y. P. Hsieh and C. Y. Lin, Lossless data embedding with high embedding capacity based on declustering for VQ-compressed codes, *IEEE Transactions on Information Forensics and Security*, vol.2, no.3, pp.341-349, 2007.

[9] C. C. Chang and C. Y. Lin, Reversible steganographic method using SMVQ approach based on declustering, *Information Sciences*, vol.177, no.8, pp.1796-1805, 2007.

[10] 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, 2009.

[11] J. D. Lee, Y. H. Chiou and J. M. Guo, Reversible data hiding based on histogram modification of SMVQ indices, *IEEE Transactions on Information Forensics and Security*, vol.5, no.4, pp.638-648, 2010.

[12] P. Tsai, Histogram-based reversible data hiding for vector quantisation-compressed images, *IET Image Process*, vol.3, no.2, pp.100-114, 2009.

[13] C. C. Chang, T. D. Kieu and W. C. Wu, A lossless data embedding technique by joint neighboring coding, *Pattern Recognition*, vol.42, no.7, pp.1597-1603, 2009.

[14] Z. M. Lu, J. X. Wang and B. B. Liu, An improved lossless data hiding scheme based on image VQ-index residual value coding, *The Journal of Systems and Software*, vol.82, no.6, pp.1016-1024, 2009.

[15] 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, 2009.

[16] C. H. Yang, S. C. Wu, S. C. Huang and Y. K. Lin, Huffman-code strategies to improve MFCVQ-based reversible data hiding for VQ indexes, *Journal of Systems and Software*, vol.84, no.3, pp.388-396, 2011.

[17] 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, 2009.

[18] C. H. Yang and Y. C. Lin, Fractal curves to improve the reversible data embedding for VQ-indexes based on locally adaptive coding, *Journal of Visual Communication and Image Representation*, vol.21, no.4, pp.334-342, 2010.

[19] Y. Linde, A. Buzo and R. M. Gray, An algorithm for vector quantization design, *IEEE Transactions on Communications*, vol.28, no.1, pp.84-95, 1980.

[20] C. H. Hsieh and J. C. Tsai, Lossless compression of VQ index with search-order coding, *IEEE Transactions on Image Processing*, vol.5, no.11, pp.1579-1582, 1996.

[21] R. Crandall, Some notes on steganography, *Posted on Steganography*, http://os.inf.tu-dresden.de/westfeld/crandall.pdf, 1998.

[22] A. Westfeld, F5 – A steganographic algorithm: High capacity despite better steganalysis, *Proc. of the 4th International Workshop on Information Hiding*, Pittsburgh, USA, pp.289-302, 2001.