Multimedia Tools and Applications (2019) 78:21223–21244
https://doi.org/10.1007/s11042-019-7425-7

A low bit-rate SOC-based reversible data hiding algorithm by using new encoding strategies

Zhibin Pan [1] · Erdun Gao [1] · Ruoxin Zhu [1] · Lingfei Wang [1]

Received: 30 May 2018 / Revised: 29 January 2019 / Accepted: 25 February 2019 /
Published online: 15 March 2019
© Springer Science+Business Media, LLC, part of Springer Nature 2019

Abstract
Search order coding (SOC) benefits a lot from the correlation of neighboring blocks for vector quantization (VQ)-compressed images. SOC selects a number of different indices in its search path as candidate search order codes. In this work, we present a low bit-rate SOC-based reversible data hiding algorithm benefiting from novel encoding strategies which exploit the information of the placeholders. The blocks are classified into two categories by the placeholders, where different encoding strategies are designed, respectively. Firstly, for the block in smooth region, the placeholders in its neighborhood are employed to compress the VQ index. Secondly, for the block in complex region, SOC is employed to compress the VQ index. Finally, for the blocks that cannot be processed by its placeholders and SOC, an effective prediction method named accurate gradient selective prediction (AGSP) and Huffman coding are introduced. After encoding phase, the size of the output bit stream is reduced so that space is saved for data embedding. Experiment results show that our proposed method outperforms other state-of-the-art SOC-based algorithms in bit rate and embedding capacity.

Keywords Reversible data hiding · Image compression · Search order coding (SOC) · Vector quantization (VQ) · Placeholder (PH) of repetition search point

1 Introduction

Recently, due to the new characteristics of the Internet communication, information resource sharing and intellectual property protection, steganography is demanded to keep pace with the development of new technologies. To protect the digital information against security issues, cryptography schemes, such as DES [7] and RSA [30] are proposed. However, these methods aiming at protecting secret information show no ability to ensure the security of cover media.

✉ Zhibin Pan
zbpan@mail.xjtu.edu.cn

[1] School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, People's Republic of China

To solve this problem, data hiding was proposed to embed a large number of secret data imperceptibly into various kinds of digital medias without attracting the attention of malicious attackers.

Data hiding mainly includes irreversible data hiding algorithms and reversible data hiding algorithms. In irreversible data hiding algorithms [1], severe degradation in the quality of original cover image may arise after secret information is extracted, which may act as a trigger for many disputes. Conversely, in reversible data hiding (RDH) algorithms, both the original cover media and secret information can be recovered completely at the meantime. On this account, reversible data hiding algorithms are used in various aspects in recent decades, especially in military and medical fields. The researches on data hiding can be performed in spatial domain [17, 32], transformed domain [3, 13, 18] and compressed domain [2, 4–6, 8, 15, 16, 19, 20, 24–29, 33, 34]. In future, we should find new forms of cover media which faces the same security problem for extending the fields of RDH applications, such as sensor data and hyperspectral data. Only if we could totally ensure the integrity of cover data, the following feature extraction and recognition works [22, 23] could be guaranteed.

Speaking of reversible data hiding in compressed domain, there are two main compression methods named vector quantization (VQ) and discrete cosine transform (DCT), respectively. Especially, vector quantization [21] is an attractive and practical technique because of its simple compression procedures and low bit rate. Consequently, VQ-based data hiding algorithms introduce great benefit to image storing and transmitting. Afterwards, based on that correlation among adjacent blocks could be further exploited, side match vector quantization (SMVQ) was proposed in 1992 [14], which generated a state codebook for each block. As the size of state codebook is always smaller than main codebook, SMVQ performs better than VQ in compression rate. However, the process of SMVQ is more complex and time consuming. Four years later, search order coding (SOC), regarded as an efficiently improved compression algorithm for VQ indices, was initially proposed by Hsieh and Tsai [11]. Considering that VQ is an important form for transmitting multimedia information, the first VQ-based scheme was designed to ensure the security of data by Lin and Wang in 1999 [20]. Later on, owing to the high compression rate of SOC, many researchers were studying on reversible data hiding algorithms based on SOC. Firstly, Chang et al. proposed an information hiding algorithm based on SOC in 2004 [2], which realized the function to compress the image and embed the secret data at the same time. The visual quality of the stego-image was maintained comparing to VQ. To reduce the code length of the stego-images, Pan et al. proposed a novel encoding and embedding rule in 2013 [25], which refrains from the transformation from SOC to original VQ and apparently improves the embedding rate. Shortly afterwards, Chang et al. developed this method by jointly utilizing the SOC and SMVQ algorithms in 2015 [5]. Benefiting from the small size of state codebook, this scheme achieved a higher compression rate and embedding rate comparing to Chang et al.'s [5] method. Then, Wang et al. proposed a technique called adjoining state-codebook mapping (ASCM) [33]. Two state-codebooks were constructed by the left and top adjacent indices of the current index to map the blocks to the corresponding indices in their state-codebooks. Therefore, the output bit stream was shortened. Besides, Lee et al. proposed another reversible data hiding algorithm [16]. In their scheme, correlative sub-codebooks were generated for image compression and data hiding. Later on in 2015, Lin et al. proposed a novel reversible data hiding algorithm to improve the embedding capacity and compression rate [19]. In their scheme, after the process of SOC, ASCM was adopted to further encode the blocks which cannot be processed by SOC. In 2016, Qin et al. proposed a reversible data hiding algorithm through the improved searching order coding (ISOC) [27], in which all indices were categorized into three types named SOC type, relative addressing (RA) type and VQ type, based on the correlation between the index and its neighborhood.

Through the above analysis, we could come to the conclusion that, for enlarging the embedding capacity of the given cover image, it should be a good way to reduce the coding length of the indexes. Considering that search order coding (SOC) has no adaptability to the variety of different blocks and it is really an inefficient coding way, we should try to design a new coding method which could show the capability for adaptively processing blocks with different complexity. By the experimental data, we found that the correlations among the neighboring blocks could be further exploited to evaluate the complexity of the current block. By utilizing the placeholders of the repetition search points (SPs) rather than conventional SOC, we could significantly compress the indexes of blocks which locate in extremely smooth regions. Importantly, to our best knowledge, our proposed method is the first method to investigate the importance of the placeholders in the predefined search path in extremely smooth region which are ignored by all other SOC-based methods.

Another obvious weakness of SOC is that the coding method would regress to original index value (OIV) when the current block is too complex. That is to say, we could further enhance the coding performance by giving a more efficient way to process those complex blocks. In our method, we introduce the new prediction method named accurate gradient selective prediction (AGSP) predictor and Huffman coding [12] to our method for further reducing the size of the output bit stream. By applying these two new strategies, we propose a reversible data hiding algorithm based on SOC, in which the secret data are embedded based on VQ index map. Compared to other compression algorithms based on SOC in VQ-compressed domain, our proposed method can achieve better compression rate and embedding capacity.

To illustrate the procedures of our proposed method and make it self-contained, the rest of this paper is arranged as follows: the related works are reviewed in Section 2. After that, our proposed reversible data hiding algorithm is described in Section 3 in detail, including the classification of the index block, compression procedure, embedding procedure and the extracting procedure. Moreover, the experimental results are shown and discussed in Section 4, which demonstrate the advantages of our proposed method by comparing with the latest reversible data hiding algorithms based on SOC. In the end, the conclusion part is given in Section 5.

2 Related works

In this section, several previous works related to our proposed method are described. To be specific, the basic concept of VQ and SOC algorithms are presented in Section 2.1 and 2.2, respectively. Then, in Section 2.3, Lin et al.'s scheme which jointly uses SOC and ASCM is introduced. Notations used in our paper and the explanations are listed in Table 1.

2.1 Vector quantization [10]

Vector quantization (VQ) was first proposed by Gary [10] in 1984 for the purpose of image compression. The process of VQ can be summarized into three steps, namely the generation of codebook, VQ encoding process, and VQ decoding process, respectively. VQ goes through the process like a mapping function which converts the pixels in a block into an index map, and

Table 1 The notation utilized in this paper

Notation	Explanation
B	Secret data
C	The codebook
N	The size of the codebook
$X= (X_0, X_1, \ldots, X_{k-1})$	The current block
(P)	Placeholder of repetition search point
x	The index of the current block X
$\overline{x}$	The prediction index value of x
d	The difference between x and $\overline{x}$
I	The index map
w	The output length of each block
CC	The output compression code
r_1, r_2	The indicator used in the receiver end
m	The length of search order code

then utilizes the index of the closest codeword in the codebook to represent the current image block which could markedly reduce the encoding rate. The procedures of VQ are conducted as following.

Firstly, the training images are divided into k-dimensional non-overlapping training blocks. Then these training image blocks are trained for generating a practical N-sized codebook. In most cases, the codebook used nowadays is trained by Linde-Buzo-Gary (LBG) algorithm [21], which could achieve a satisfied visual quality of the decompressed images. Secondly, VQ chooses the closest codeword C_b from the codebook C for the current block X based on the minimum distortion criterion. The Euclidian distance distortion between X and C_i is calculated using eq. (1) and the choosing rule is given by eq. (2), where $X = (X_0, X_1, \ldots, X_{k-1})$ denotes the current block, C_i and X denote the $i-$th component of codebook C and current image block, and $C_{i,j}$ denotes the $j-$th component of the codeword C_i. Finally, the index of best-match codeword C_b can be used to encode the current block X.

$$ED(X, C_i) = \|X - C_i\|^2 = \sum_{j=0}^{k-1} \left(X_j - C_{i,j}\right)^2 \tag{1}$$

$$ED(X, C_b) = \min_{0 \le i \le N-1} \{ED(X, C_i)\} \tag{2}$$

After all the input blocks are processed in such a manner, an index table can be obtained [9], which significantly reduces the bit rate. At the decoding end, the decoder can restore the images according to the received index table and the same codebook, so that the quality of the recovered image is the same as VQ-compressed image in the encoding end.

2.2 Search order coding [11]

Due to the high correlation among the neighboring blocks, the adjacent indices in VQ index map are approximate to one another. On the basis of such a fact, Hsieh and Tsai [11] proposed search order coding (SOC) to compress the index map generated by VQ in 1996. In this method, the core aim is to seek the same index as the current index in the previously-encoded blocks in a predefined search path. Then, SOC code rather than VQ index is used to shorten the length of compression code. After all the blocks are processed in the raster scan order, the VQ index map can be compressed further. When an index needs to be encoded, it is called as the search center, and the previously-encoded blocks are called the search points (SP). Denote the length of the SOC code as m, and 2^m different index values are sequentially considered as possible SOC code one by one according to the search path, in which the repeated index values that are called placeholders (PHs) of repetition search points are completely ignored. In most cases, *m* is set as two or three. If the index of the search center *X* is identical to one of the SOC indices, it can be encoded by SOC, else it can only be encoded by the original index value (OIV). To make this algorithm reversible, one bit indicator is used to distinguish whether the current block is encoded by SOC or OIV. For example, “1” means that the current block *X* is encoded by SOC, and “0” means that the current block *X* is encoded by OIV.

Figure 1 shows an example of SOC algorithm and the arrows show the predefined search path. Suppose the size of the codebook *N* is 256, two bits are used as SOC indices, and the coordinates of the search center are (4, 3) with an original VQ index value of 34. After the searching process, the SOC indices are (33, 32, 35, 34), and each of them can be encoded as $((00)_2, (01)_2, (10)_2, (11)_2)$ respectively. Apparently, the fourth index encoded as $(11)_2$ is identical to the current index 34. Because indicator “1” means the current block can be encoded by SOC, the current block can be finally encoded as “1||11”, i.e., the current block can be compressed to 3 bits rather than 8 $(\log_2 N) \gg 3$ bits.

2.3 Reversible data hiding scheme of Lin et al. [19]

In 2015, Lin et al. proposed a reversible data hiding algorithm based on SOC and adjoining state-codebook mapping (ASCM) [19]. In this scheme, the bit rate is significantly decreased.

ASCM maps the current block *X* to a small sized corresponding state-codebook, which is generated by its adjacent blocks (the upper block *U* and the left block *L* as shown in Fig. 2). To generate an *Ns*-sized state-codebook of the block *U*, the similarity between *U* and each codeword is calculated to resort the codebook in ascending order temporarily, and the top *Ns* indices which better match the block *U* are selected to construct the state-codebook. The similarity between two codewords is calculated using eq. (3):

$$ED\left(C_{sp}, C_i\right) = \sum_{j=0}^{k-1} \left(C_{sp,j} - C_{i,j}\right)^2 \tag{3}$$

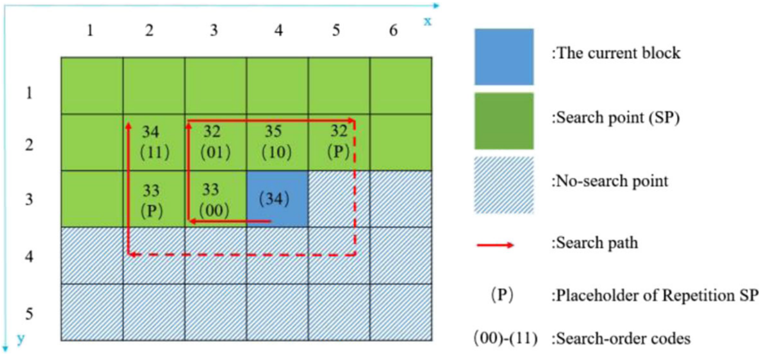


Fig. 1 Example of SOC algorithm

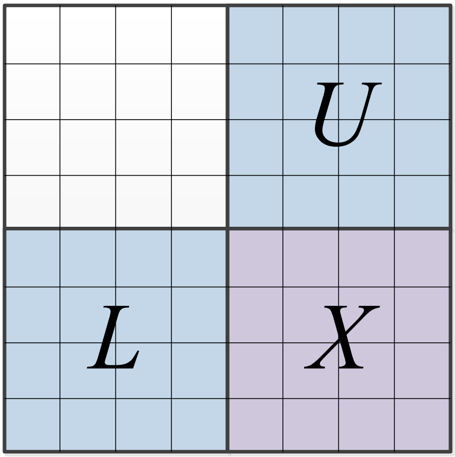


Fig. 2 The diagram of neighboring blocks of current block X

where C_{sp} denotes the codeword in codebook C for the current SP, $C_{sp,j}$ is the j-th element of C_{sp}, C_i is the i-th codeword in C, and $C_{i,j}$ is the j-th element of C_i. The closest codeword is that with the smallest ED.

The indices in the state-codebook of the block L are different from the already generated state-codebook of the block U. After two state-codebooks are obtained, the current block X will be investigated in these two state-codebooks. If there is an index that is equal to the index x of the current block X, x can be encoded by its order in the corresponding state-codebook.

In the scheme of Lin et al., SOC is firstly employed to reduce the bit rate of the output bit stream. Then, for the blocks that cannot be encoded by SOC, ASCM is introduced. The state-codebooks are generated by the current search-order codes. Assume that N is the size of the main codebook, m is the length of SOC index, w is the output length of each block, and all the blocks are encoded in the raster scan order. For an index from the VQ index map, Lin et al.'s embedding scheme is as follows:

Case1: If the current block X can be encoded by SOC, the indicator "1" is attached to the output bit stream, followed by one of the 2^m search-order codes and $(w-(m+1))$ bits secret data.

Case2: If the current block X cannot be encoded by SOC, ASCM is performed. If the current index can be found in the state-codebooks, the indicator "01" is attached to the output bit stream, followed by $(\mathrm{m}+\log_2 Ns)$ bits location indicator and $(w-(\mathrm{m}+\log_2 Ns))$ bits secret data.

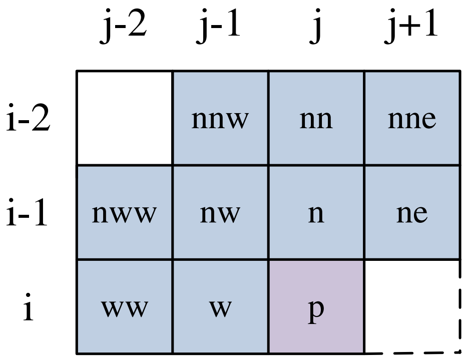
Case3: If the current block X cannot be encoded by Case 1 and Case 2, it has to be encoded by indicator "00" and followed by $\log_2 N$ bits original VQ index value.

After all the blocks in the non-seed area are encoded, the test image can be compressed in a low bit rate. And the extraction steps are exactly the reverse process of the embedding process.

2.4 AGSP predictor [31]

The accurate gradient selective prediction (AGSP) predictor calculates the gradient of current pixel using the nine neighboring pixels listed as n, w, nw, ne, ww, nn, nne, nnw and nww shown in Fig. 3. The estimated gradient is calculated as follows:

Fig. 3 The diagram of neighboring pixels of current pixel p



	j-2	j-1	j	j+1
i-2		nnw	nn	nne
i-1	nww	nw	n	ne
i	ww	w	p	

Horizontal:

$$D_1 = (2|w{-}ww| + 2|n{-}nw| + 2|ne{-}n| + |nn{-}nnw| + |nne{-}nn| + |nw{-}nww|)/9 + 1 \tag{4}$$

Vertical:

$$D_2 = (2|w{-}nw| + 2|n{-}nn| + |ne{-}nne| + |ww{-}nww| + |nw{-}nnw|)/7 + 1 \tag{5}$$

+45 degree:

$$D_3 = (2|w{-}n| + 2|n{-}nne| + |ww{-}nw| + |nw{-}nn|)/6 + 1 \tag{6}$$

−45 degree:

$$D_4 = (2|w{-}nww| + 2|n{-}nnw| + |ne{-}nn|)/5 + 1 \tag{7}$$

Accordingly, the corresponding causal pixel of gradient D_1, D_2, D_3 and D_4 are defined to be n, w, ne and nw, respectively. Then, two directions with the two smaller gradient values are chosen as D_{min1} and D_{min2}. Meanwhile, the corresponding causal pixels which are denoted as C_{min1} and C_{min2} on the same directions of D_{min1} and D_{min2} are chosen. Therefore, the prediction value $\overline{p}$ of the current pixel p is calculated as follows:

$$\overline{p} = \frac{D_{\min1} \times C_{\min2} + D_{\min2} \times C_{\min1}}{D_{\min1} + D_{\min2}} \tag{8}$$

3 The proposed method

The algorithm procedures and our new encoding strategy are shown in detail in this section. As it can be seen in Fig. 4, the cover image is non-overlapping divided into the equal-size blocks according to the predefined width and height. Afterwards, for each block, we search the codebook to find the most matching codeword and obtain the index for the current block. And then, for the coding procedure, we introduce the new encoding strategy which classifies all the blocks into smooth areas or complex areas according to the distribution of the neighboring indexes. Then, for different areas, we propose to use different coding methods to adaptively exploit the correlations among adjacent blocks. After finding the proper indexes for all the blocks, secret data are embedded into the codes to generate the codes stream for transmitting.

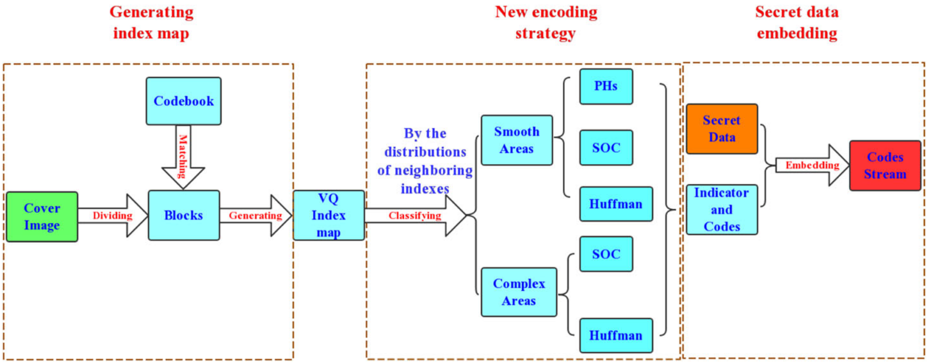


Fig. 4 Overview of the proposed method

The encoding and embedding procedures start from the second row and second column of the index table (the non-seed area) in the raster scanning order, and the seed areas are encoded by VQ. In this paper, the classification operation of the current block using the placeholders (PHs) of the repetition SPs is defined before the new encoding and embedding procedures in Section 3.1. In Section 3.2 and Section 3.3, the embedding scheme and extracting scheme are described in detail, respectively.

3.1 The classification of the block

In conventional SOC encoding scheme, commonly a fixed length of SOC index or VQ index is used to encode each block. In order to search a certain number of search order codes, the SPs are searched in the predefined search path, and the repeated indices named placeholders of repetition SPs are completely ignored. However, the placeholders of repetition SPs could be exploited for extracting lots of important information. For example, if the current block has plenty of placeholders in its search path, it is more likely that the current block has the same index value as the values of the placeholders. Especially, once the index values of four neighboring blocks are identical (there are three the same placeholders in the four neighboring indices), it is likely that the current block also has the same index value as its neighborhoods.

However, in order to find SOC indices, the search scope has to be expanded so that the following search order codes have lower correlation with the current block. To improve this deficiency of SOC algorithm, more efficient encoding strategies based on SOC using the information of placeholders are proposed in this paper.

The idea of our proposed encoding method is to investigate the importance of the placeholders in the predefined search path, and then adaptively use one of the three encoding strategies to encode each block according to the complexity of the current region. To our best knowledge, none of the existing SOC-based schemes focused on the placeholders and aimed at improving the encoding rules of the blocks that are in smooth area. That is to say, the blocks can already be encoded by original SOC algorithm. And the blocks which cannot be encoded by original SOC can also be compressed further in our proposed method.

Because the four neighboring indices usually have the strongest correlation with the current block, the complexity of the current block X is judged by its four adjacent processed blocks, which are counted from the left block of the current block as shown in Fig. 5. The index values of the placeholders in the four neighboring blocks are classified into the placeholder type. The image blocks are classified into two types using the information of PH – (1) smooth areas and

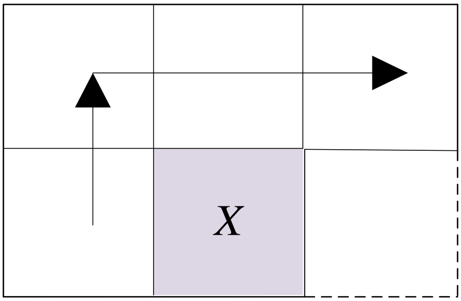
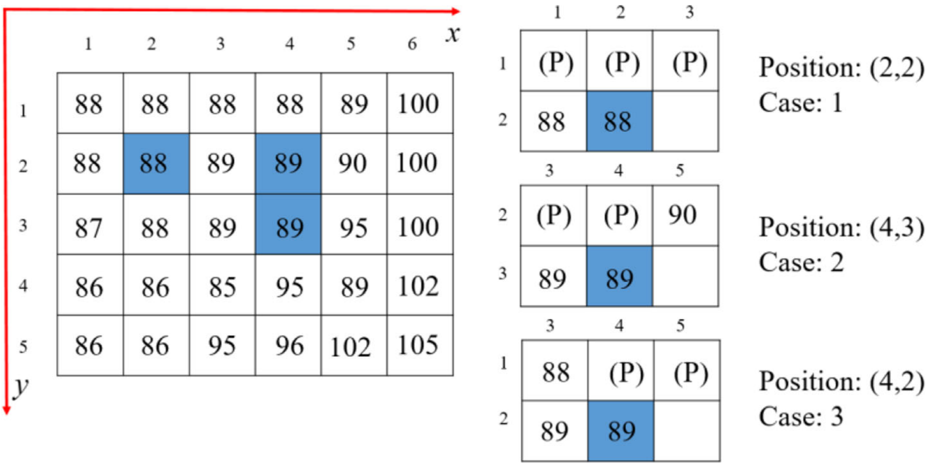


Fig. 5 Search order of the neighboring blocks of the current block *X*

(2) complex areas. We define a block in smooth region is the block that has more than one placeholder in its four neighboring blocks. Taking Fig. 6 as an example, the smooth region consists of the following three cases as shown in Table 2, where *Y* and *Z* denote two different index values. In fact, in Table 2, '*YYYY*', '*YYYZ*' and so on just represent the different types of distributions of the four neighboring indexes. Taking Case1 in Fig. 6 as an example, the four neighboring indexes are 88, 88, 88, 88. So it is just the type of '*YYYY*' and *Y* represents 88 in this example. Therefore, the three placeholders 'P' is 88 and also *Y*. For Case2, the four neighboring indexes are 89, 89, 89, 90. So it belongs to the type of '*YYYZ*', in which *Y* represents 89 and *Z* represents 90, respectively. And for Case3, the four neighboring indexes are 89, 88, 88, 89. So it belongs to the type of "*YZZY*", where *Y* represents 89 and *Z* represents 88, respectively. There are two different placeholders which are *Y* and *Z* in Case3. Especially for Case3, in the encoding procedure, we need one bit of code to distinguish that the current index is equal to which value, *Y* or *Z*. However, in Case1 and Case2, no more bit is needed. Because only if the current index is the same value as *Y*, we would use PHs to encode it.

Case 1: The four neighboring index values are all equal to *Y* so that there are three placeholders in its four neighboring blocks, and the index value *Y* is classified into PH;



	1	2	3	4	5	6
1	88	88	88	88	89	100
2	88	88	89	89	90	100
3	87	88	89	89	95	100
4	86	86	85	95	89	102
5	86	86	95	96	102	105

	1	2	3
1	(P)	(P)	(P)
2	88	88	

Position: (2,2)
Case: 1

	3	4	5
2	(P)	(P)	90
3	89	89	

Position: (4,3)
Case: 2

	3	4	5
1	88	(P)	(P)
2	89	89	

Position: (4,2)
Case: 3

Fig. 6 A fragment of an index map and examples of the placeholders "(P)"

Table 2 Types of three different cases in smooth areas

Case	Indices of the neighboring blocks	PH	Examples	
			Position	Index value
Case1	*YYYY*	*Y*	(2, 2)	88
Case2	*YYYZ, YYZY, YZYY, ZYYY*	*Y*	(4, 3)	89
Case3	*YYZZ, YZYZ, YZZY*	*Y, Z*	(4, 2)	89

Case 2: Three of the four neighboring index values are equal to *Y* so that there are two placeholders in its four neighboring blocks, and the index value *Y* is classified into PH;

Case 3: Two neighboring index values are equal to *Y*, and the other two neighboring index values are equal to *Z* so that there are two placeholders in its four neighboring blocks, and the index value *Y*, *Z* are classified into PH.

3.2 Encoding and embedding phase

Four neighboring indices of the current block X (x represents its index) are compared to the three cases in Table 2 to judge whether the current block locates in smooth areas. If it belongs to smooth areas, the index values in PH type can be determined at the same time. In order to decode the blocks reversibly, indicators are introduced to distinguish three encoding strategies. For the block in smooth region, if it can be encoded by PH type, indicator '0' is used; else if it can be encoded by SOC, indicator '10' is used, else indicator '11' is used. For the block in complex region, if it can be encoded by SOC, indicator '0' is used, else indicator '1' is used. The indicators used in the new encoding strategies are shown in Table 3 and Fig. 7. In the above encoding strategies, if the current block cannot be processed by PH and SOC, an efficient prediction algorithm called AGSP is utilized to obtain the prediction index value $\overline{x}$ of x and the prediction error $d = x\text{-}\overline{x}$. AGSP can make full use of the gradient information of the current block so that it shows good prediction accuracy on complex area. As a result, because of the concentration distribution of the prediction error d using the equation of $d = x\text{-}\overline{x}$, Huffman coding [12] can achieve the highest compression ratio in most cases.

In the embedding phase, the output code length of each in-process block is set to w, which is defined as one unit. Consequently, the proposed encoding and embedding algorithms are as follows. Specially, the "||" means the concatenation operation.

- Input: Index map I of the gray scale cover image, the secret data B

Table 3 The indicator utilized in the proposed algorithm

Encoder	In the smooth region			In the complex region	
	PH	SOC	HUFFMAN	SOC	HUFFMAN
Indicator	0	10	11	0	1

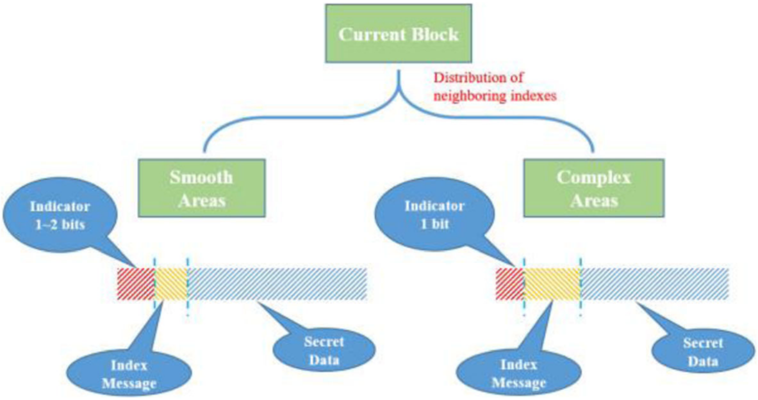


Fig. 7 Data structure of the one unit in the code steams

- Output: Output bit stream *CC* of the image embedded with secret message
- Step 1: Read an index value *x* from the residual of the index map *I*, and determine which region it is in;
- Step 2: If the current block is in complex region, go to Step 4, otherwise perform one of the following steps:
- Step 2.1: If current block belongs to Case1 or Case2, and if the current index equals to the value of PHs, the index *x* is denoted by indicator "0" and (*w*-1) bits secret data can be embedded, i.e., 0|| (*w*-1) bits *B*, else go to Step 3;
- Step 2.2: If current block belongs to Case3, and if the current index equals to *Y* or *Z*, 1 bit more indicator is used ("0" indicates the first index *Y* and "1" indicates the second index *Z*), so that the index *x* is denoted by indicator "1", another 1 bit indicator and (*w*-2) bits secret data can be embedded, i.e., 0||1 bit of location indicator|| (*w*-2) bits *B*, else go to Step 3;
- Step 3: Determine 2 bits SOC code. If the current block can be encoded by SOC, the index *x* is denoted by indicator "10", 2 bits SOC index and (*w*-4) bits secret data can be embedded, i.e., 10||2 bits SOC|| (*w*-4) bits *B*, else go to Step 5;
- Step 4: Determine 2 bits SOC index. If the current block can be encoded by SOC, the index *x* is denoted by indicator "0" and 2 bits SOC index and (*w*-3) bits secret data can be embedded, i.e., 0||2 bits SOC|| (*w*-3) bits *B*, else go to Step 5;
- Step 5: Utilize AGSP to obtain the prediction index $\overline{x}$ of the current block, and calculate the prediction error $d = x$-$\overline{x}$. Then Huffman code is utilized to encode *d*. Moreover, if the current block is in smooth area, the index *x* is denoted by indicator "11" and Huffman code, i.e., 11||Huffman code; else the index *x* is denoted by indicator "0" and Huffman code, i.e., 1||Huffman code.
- Step 6: Repeat Step1 to Step 5 until all the indices in index map *I* have been encoded, and the secret message *B* has been embedded into the output bit stream *CC*.

Finally, the output bit steam can be obtained after the encoding and embedding procedures. Generally, it is sent to the receiver without additional message. Taking Fig. 6 as an example, assume that *w* is 9 and secret data of *B* are "111,010,101". If the in-process index *x* is 89 at position (4, 2), and the neighboring indices are 89, 88, 88 and 89. Based on the previous definition, the current block is in smooth area and it belongs to Case3, and PH type includes the indices 89 and 88. Because 89 equals to the current index value, the current block can be encoded as “0||0||1,110,101”.

3.3 Extracting phase

After receiving the output bit stream *CC*, the receiver can easily extract secret message *B* and index map *I* in a reverse process of the embedding phase. In this section, the extraction phase is described in detail.

- Input: Compression code *CC* containing the secret data *B*
- Output: The index map *I* and the secret message *B*
- Step 1: Set secret message *B* and index map *I* to be empty;
- Step 2: If the current block is in smooth area, go to Step 3, else go to Step 4;
- Step 3: The extracting process of the blocks that belongs to Case1 is taken as an example. Determine the composition of PH, extract 1 bit indicator from the compression code *CC* and denote it as r_1, then perform one of the following Steps:
- Step 3.1: If $r_1 = 0$, the reconstructed index equals the index value in PH, then extract next (w-1) bits and concatenate them into secret data *B*;
- Step 3.2: If $r_1 = 1$, read next 1 bit indicator (denote as r_2). If $r_2 = 0$, extract next 2 bits from *CC* as SOC index and decode them using SOC decoder as the reconstructed index of x. Then, extract next (w-4) bits and concatenate them into secret data *B*. If $r_2 = 1$, extract the different prefix code, suppose the Huffman code has m bits and decoded them by Huffman decoder as the prediction error d. The index x of the current block can be obtained by the prediction index value $\bar{x}$ and the prediction error d. Extract next (w-2-m) bits and concatenate them into secret message *B*;
- Step 4: Extract 1-bit indicator denoted as r_1, and then perform one of the two following steps:
- Step 4.1: If $r_1 = 0$, extract the next 2 bits SOC code and decode them as the reconstructed index of x. Extract next (w-3) bits and concatenate them into secret message *B*;
- Step 4.2: If $r_1 = 1$, extract the different prefix code, suppose the Huffman code has m bits and decoded them as the prediction error d. Then the index of the current block can be obtained. Finally, extract next (w-1-m) bits and concatenate them into secret message *B*;

Following the completion of the four steps above, secret message *B* and index map *I* can be reconstructed reversibly at the same time.

4 Experimental results

For algorithms of vector quantization (VQ)-based reversible data hiding, with the given cover image and secret data, the target is to invisibly transmit the secret data by hiding it into the cover image. Therefore, the stability of algorithm could be guaranteed if it could gain the best embedding efficiency on most of the various of test images. The experiments are performed with ten 512 × 512-sized, 8-bit gray level typical images, and each of them is partitioned into non-overlapping blocks sized 4 × 4. Secret data B was created using a pseudo-random number generator. The test images are shown in Fig. 8, namely “Lena”, “Peppers”, “Airplane”, “Boat”,

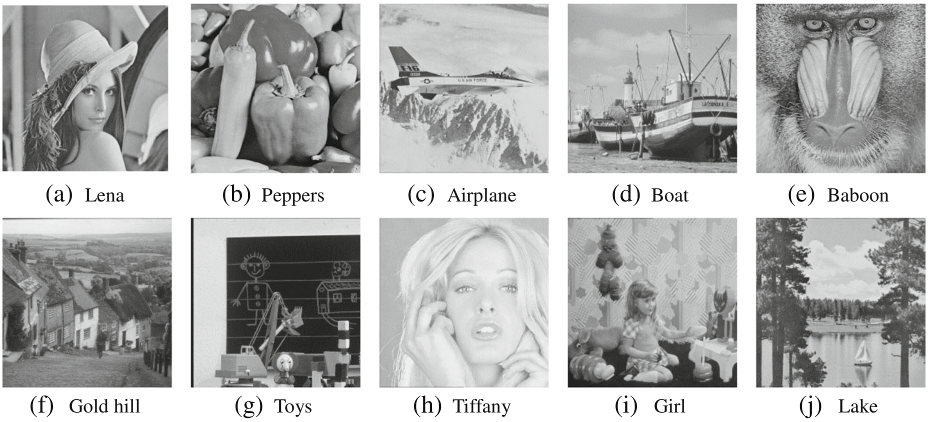

(a) Lena (b) Peppers (c) Airplane (d) Boat (e) Baboon
(f) Gold hill (g) Toys (h) Tiffany (i) Girl (j) Lake

Fig. 8 Ten standard test images

"Baboon", "Gold hill", "Toys", "Tiffany", "Girl", and "Lake". Additionally, the 256-sized and 512-sized 16-dimensional codebook which are trained by the LBG algorithm is implemented to simulate our proposed method. The criterions, named bit rate (BR) and embedding efficiency (EE), are considered to evaluate the performance of algorithms. In this section, experiments are conducted to verify the performance of our proposed method compared to those of Lin's scheme [19], Wang's scheme [33], Lee's scheme [16], Chang's scheme [2] and Pan's scheme [25].

Our proposed method improves the embedding phase of the blocks that can be encoded by SOC and the blocks that cannot be encoded by SOC. In order to verify the superiority, the comparison results are shown in: (1) the performance of the blocks in smooth area or complex area that can or cannot be processed by SOC; (2) the performance of the whole image blocks.

In (1), the compression rate is mainly discussed; in (2), the embedding rate and the embedding efficiency are mainly discussed.

The first criterion is bit rate (*BR*), and it represents the ratio of the size of output bit stream and the size of the original cover image, which is measured by bit per index (*bpi*). *CR* is defined by eq. (9), where $\|CC\|$ is the length of the output bit stream *CC*, and N_i is the total number of indices in the index table in the VQ compression domain. A smaller *BR* benefits both the senders and receivers in terms of the bandwidth of the network.

$$BR = \|CC\|/N_i \tag{9}$$

The second criterion is embedding rate (*ER*), which is defined in eq. (10). *ER* denotes the number of the embedded secret bits in each compressed index and is utilized to evaluate the embedding rate of the compression scheme.

$$ER = \|B\|/N_i \tag{10}$$

where $\|B\|$ is the number of secret bits embedded into the compression codes of the cover image. Additionally, *bpi* is utilized to measure the number of secret bits embedded into each index. Generally, larger embedding rate is preferred.

The third criterion is the embedding efficiency (*EE*), which is defined in eq. (11). *EE* denotes the number of the embedded secret bits in each bit of the output bit stream. A larger embedding rate and a smaller compression rate can generate a higher embedding efficiency.

$$EE = \frac{\|B\|}{\|CC\|} \times 100\% = \frac{ER}{BR} \tag{11}$$

As we have talked before, PHs could show the advantages in the given three cases. In addition, the percentages of these three cases on the test images are given in Table 4.

On 10 test images, the average percentages of Case1, Case2, Case3 are 12.53%, 11.57% and 8.31%, respectively. That is to say, the coding length of these parts are potential to be reduced. Especially for Toys, the percentages of Case1 and Case2 are 37.95% and 19.16%, which show that the redundancy of Toys could be further exploited.

In fact, the encoding method based on PHs is to reduce the code length of SOC. By doing so, more coding vacancy are saved for embedding more secret data. To show the advantage of PHs, the direct way is to compare the code length between PHs and SOC [2] in smooth areas. As shown in Table 5, comparing to SOC which evenly uses 3 bits for encoding every index, our proposed method reduces the coding length to 2.51 bits (by 16.3%, (3–2.51)/3 × 100%) averagely. Moreover, it works better for the smooth test images, such as "Toys" and "Tiffany" than the complex test images, such as "Baboon". This difference is due to the fact that the proposed method mainly makes use of the neighboring correlative information, so that the smoother the area has the higher correlation is in the neighboring indices. That is also the reason why we only introduce the utilization of PHs in smooth areas.

Table 6 compares the performance of the proposed method and the original SOC algorithm for the blocks that cannot be processed by SOC, which means the blocks are in complex area and such blocks do not have much correlation to their neighbors compared to the blocks in smooth area. In the proposed method, an efficient predictor called AGSP is utilized to reduce the redundancy in complex area. AGSP is distinguished from other predictors for it makes fully use of the gradient information of the complex area. In order to illustrate the efficiency of AGSP predictor, the distribution of code length in ten test images is shown in Fig. 9, which is encoded by Huffman coding. Compared to the original SOC algorithm, which needs 1 + 8 = 9 bits ($N = 256$) to encode the blocks which cannot be processed by SOC, fewer bits are required in the proposed encoding scheme. Table 6 shows that the compression rate of proposed method is reduced by 15.4% ((9–7.61)/9 × 100%) on average compared to the original SOC scheme.

Table 7 shows the comparison between our proposed method and five existing schemes in terms of bit rate of the whole image using 256-sized codebook. We know that for vector quantization (VQ)-based methods of reversible data hiding, different codebooks would lead to different experimental results. So in our comparing experiments, all the schemes use the same codebook. As the baseline, [2] uses the original SOC to encode the indexes, in which no more parameter is needed to affect the final results. Pan et al. [25] reduces bit rate by avoiding the

Table 4 Percentage (%) of three cases on test images ($N = 256$)

Image	Lena	Airplane	Boats	Baboon	Peppers	Lake	Tiffany	Toys	Girl	Gold hill	Average
Case1	2.90	14.92	13.97	0.15	7.38	6.99	27.49	37.95	5.79	7.72	12.53
Case2	8.31	18.93	14.96	1.54	11.94	10.76	15.66	19.16	7.06	7.38	11.57
Case3	8.78	12.12	8.80	3.14	8.94	8.95	11.81	8.28	7.12	5.19	8.31

Table 5 Comparison between the proposed method and SOC on *BR* (bpi) of flat parts ($N = 256$)

Image	Lena	Airplane	Boats	Baboon	Peppers	Lake	Tiffany	Toys	Girl	Gold hill	Average
SOC	3	3	3	3	3	3	3	3	3	3	3
Proposed	2.77	2.38	2.38	3.03	2.58	2.60	2.04	1.98	2.63	2.73	2.51

transform from SOC coding to OIV (original index values), which also has no additional parameter to affect the results. In [19], the search-order code length (SOL) and the state-codebook length (SCL) are two crucial parameters. When we re-realized this algorithm, we set SOL = 2, SCL = 8 as suggested in their paper. As described in [33], the number of state-codebook-index (SCI) is the most important parameter, which affects the capacity for embedding the secret data in an image, as well as the bit rate of the output code-stream. Therefore, as recommended, we set SCI = 3bits. In Lee et al.'s method [16], z represents the half length of a Coarse sub-codebook, which is required to be properly selected to yield the best performance. In our re-realization, z = 3 is set to achieve the best performance. Table 7 shows that compared to these five existing schemes, the bit rate of our proposed method is reduced from 8.7% to 13.6%. This superiority is due to the fact that our proposed method improves the encoding process of all the blocks no matter whether it can be encoded by SOC or not. Notably, for the blocks that can be processed by SOC in the original schemes, it can be encoded by fewer bits in our proposed method because of the usage of the placeholders of repetition search points. On the other hand, for the blocks that cannot be encoded by SOC, our proposed method makes fully use of the gradient information and the most efficient encoding rule of Huffman coding.

Additionally, Table 8 shows the comparison between our proposed method and these five existing schemes using 512-sized codebook, and the bit rate of our proposed method is reduced from 15.4% to 17.1%.

Tables 9 and 10 show the comparison of the proposed embedding method with the existing reversible data hiding schemes based on SOC in terms of embedding performance when 16,129 bits secret data are embedded using 256-sized and 512-sized codebook respectively.

Experimental results show that the proposed method achieves the best embedding efficiency. Specifically, the average bit rate of the proposed method is 5.90 *bpi* using 256-sized codebook. By contrast, Lin et al.'s scheme achieved 6.38 *bpi* under the same condition, which makes it distinctive from other five existing embedding methods obviously. Experimental results in Table 9 show that our proposed scheme reduces 7.5% ((6.38–5.90)/6.38 × 100%) of the bit rate compared to Lin et al.'s scheme using 256-sized codebook. Likewise, experimental results in Table 10 show that our proposed scheme reduces 16.7% ((8.18–6.81)/8.18 × 100%) of the bit rate compared to Lin et al.'s scheme using 512-sized codebook.

We conducted another experiment to evaluate the performance of data embedding phase in terms of the maximum embedding capacity at a fixed bit rate. Because of the limit of the embedding algorithm, in Pan et al.'s [25] and Chang et al.'s [2] schemes, only 1 bit can be embedded into each index so that the maximal embedding capacity of these two schemes is

Table 6 Comparison between the proposed method and SOC on *BR* (bpi) of complex parts ($N = 256$)

Image	Lena	Airplane	Boats	Baboon	Peppers	Lake	Tiffany	Toys	Girl	Goldhill	Average
SOC	9	9	9	9	9	9	9	9	9	9	9
Proposed	7.38	8.02	8.01	8.00	7.39	8.06	6.47	7.85	7.5	7.43	7.61

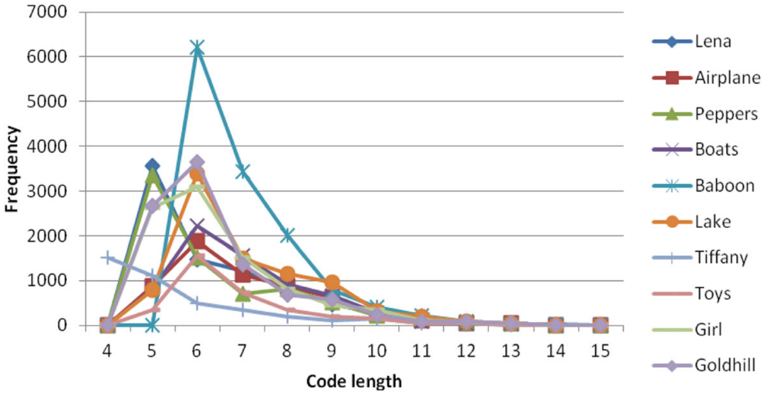


Fig. 9 The distribution of code length of prediction error encoded by Huffman coding ($N = 256$)

fixed as 16,129 ((128–1) × (128–1)) bits. In Fig. 10, the proposed method is compared to remaining other three embedding schemes, i.e., Lin et al.'s [19], Wang et al.'s [33] and Lee et al.'s [16] schemes, because these three schemes utilized the neighboring correlation to encode the cover images and to embed secret data. However, Lin et al.'s [19] and Wang et al.'s [33] schemes used two or four neighboring blocks to obtain the corresponding state-codebook, which leads to the additional computational cost. In this experiment, we also use the suggested optimal experimental settings in each of the three schemes.

In Fig. 10, the maximal embedding capacity is shown when the bit rate is fixed at 8 bpi using 256-sized codebook, which has the same bit rate as traditional vector quantization index. Obviously, more secret bits can be embedded by the proposed method compared to any of three existing schemes in all of the ten test images, which means that the proposed scheme performs extremely well in the data embedding phase. On average 18.0% ((50,274–42,613)/42,613 × 100%) more secret data can be embedded when the bit rate is fixed at 8 bpi than the Lin et al.'s [19] scheme, which also outperforms the other two embedding schemes.

Comparing to the original SOC method in [2], it is unavoidable that all the following schemes enlarge the computing complexity a little. For example, in [33], adjoining state-codebook mapping (ASCM) was designed to map the blocks to the corresponding indices in its state-codebooks. In [19], ASCM was adopted to further encod the blocks which cannot be

Table 7 Comparison of *BR* (bpi) between the proposed method and five other schemes ($N = 256$)

Images	Schemes					
	Lin et al. [19]	Wang et al. [33]	Lee et al. [16]	Chang et al. [2]	Pan et al. [25]	Proposed
Lena	5.46	5.93	5.97	6.04	6.04	5.13
Airplane	4.91	5.82	5.04	5.22	5.22	4.48
Boats	5.21	5.96	5.26	5.51	5.51	4.74
Baboon	7.48	7.33	7.94	7.90	7.90	7.11
Peppers	5.24	5.82	5.66	5.76	5.76	4.83
Lake	5.77	6.23	6.07	5.89	5.89	5.50
Tiffany	4.15	5.52	4.18	4.35	4.35	3.21
Toys	4.19	5.55	3.87	4.19	4.19	3.30
Girl	5.76	6.24	6.43	6.00	6.00	5.42
Gold hill	5.80	6.04	6.31	6.17	6.17	5.53
Average	5.40	6.04	5.67	5.70	5.70	4.93

Table 8 Comparison of *BR* (bpi) between the proposed method and five other schemes ($N = 512$)

Images	Schemes					
	Lin et al. [19]	Wang et al. [33]	Lee et al. [16]	Chang et al. [2]	Pan et al. [25]	Proposed
Lena	6.87	7.86	7.84	7.22	7.22	6.09
Airplane	6.31	7.31	6.88	6.23	6.23	5.37
Boats	6.74	7.81	6.73	6.43	6.43	5.60
Baboon	9.01	8.90	9.76	9.23	9.23	8.28
Peppers	6.65	7.75	7.58	6.99	6.99	5.84
Lake	7.32	7.98	7.65	7.29	7.29	6.50
Tiffany	6.95	6.73	5.42	5.19	5.19	3.70
Toys	6.99	6.94	4.79	4.82	4.82	3.86
Girl	7.62	7.86	8.18	7.63	7.63	6.42
Gold hill	7.54	7.54	8.20	7.86	7.86	6.61
Average	7.20	7.67	7.30	6.89	6.89	5.83

processed by SOC. In our method, we firstly classify the blocks into smooth or complex areas. For blocks in different areas, we apply different coding methods. This classification operation would a little bit enlarge the complexity of original SOC algorithm. However, for testing the efficiency of RDH schemes, we simulate it on the PC and all the coding procedures take only several seconds and limited memory.

5 Conclusions

In this paper, a novel effective reversible data hiding scheme based on SOC in VQ domain is proposed. By further exploiting the correlation among the neighboring blocks, the original

Table 9 Comparison of bit rate and embedding efficiency between the proposed method and other five schemes for embedding 16,129 bits ($N = 256$)

Images		Schemes					
		Lin et al. [19]	Wang et al. [33]	Lee et al. [16]	Chang et al. [2]	Pan et al. [25]	Proposed
Lena	BR	6.44	6.92	6.96	7.20	6.71	6.12
	EE	0.15	0.14	0.14	0.14	0.15	0.16
Airplane	BR	5.89	6.81	6.03	6.58	6.55	5.46
	EE	0.17	0.14	0.16	0.15	0.15	0.18
Boats	BR	6.20	6.95	6.25	6.83	6.78	5.72
	EE	0.16	0.14	0.16	0.14	0.15	0.17
Baboon	BR	8.46	8.32	8.93	9.21	9.40	8.09
	EE	0.12	0.12	0.11	0.11	0.10	0.12
Peppers	BR	6.22	6.81	6.65	6.91	6.63	5.81
	EE	0.16	0.14	0.15	0.14	0.15	0.17
Lake	BR	6.76	7.22	7.06	7.38	7.53	6.48
	EE	0.15	0.14	0.14	0.13	0.13	0.15
Tiffany	BR	5.13	6.51	5.17	5.82	4.49	4.20
	EE	0.19	0.15	0.19	0.17	0.22	0.23
Toys	BR	5.17	6.54	4.86	5.68	6.21	4.28
	EE	0.19	0.15	0.20	0.17	0.16	0.23
Girl	BR	6.74	7.23	7.42	7.47	7.45	6.40
	EE	0.15	0.14	0.13	0.13	0.13	0.15
Gold hill	BR	6.78	7.03	7.30	7.66	8.12	6.52
	EE	0.15	0.14	0.13	0.13	0.12	0.15

Table 10 Comparison of bit rate and embedding efficiency between the proposed method and other five schemes for embedding 16,129 bits ($N = 512$)

Images		Schemes					
		Lin et al. [19]	Wang et al. [33]	Lee et al. [16]	Chang et al. [2]	Pan et al. [25]	Proposed
Lena	BR	7.85	8.84	8.82	10.37	9.58	7.07
	EE	0.12	0.11	0.11	0.09	0.10	0.14
Airplane	BR	7.25	8.44	7.56	9.46	8.39	6.35
	EE	0.13	0.12	0.13	0.10	0.12	0.15
Boats	BR	7.72	8.79	7.71	9.63	8.62	6.58
	EE	0.13	0.11	0.13	0.10	0.11	0.15
Baboon	BR	9.99	9.88	10.74	12.24	12.03	9.26
	EE	0.10	0.10	0.09	0.08	0.08	0.11
Peppers	BR	7.63	8.73	8.56	10.16	9.31	6.82
	EE	0.13	0.11	0.11	0.10	0.11	0.14
Lake	BR	8.30	8.96	8.63	10.42	9.66	7.48
	EE	0.12	0.11	0.11	0.09	0.10	0.13
Tiffany	BR	7.93	7.71	6.40	8.52	7.14	4.68
	EE	0.12	0.13	0.15	0.12	0.14	0.21
Toys	BR	7.97	7.92	5.77	8.17	6.68	4.84
	EE	0.12	0.12	0.17	0.12	0.15	0.20
Girl	BR	8.60	8.84	9.16	10.75	10.07	7.40
	EE	0.11	0.11	0.11	0.09	0.10	0.13
Gold hill	BR	8.52	8.52	9.18	10.96	10.36	7.59
	EE	0.12	0.12	0.11	0.09	0.09	0.13

SOC scheme is enhanced in two aspects. Firstly, for the blocks in smooth area, our proposed method highlights the importance of the placeholders of repetition search points in the SOC search path and uses the placeholders to encode the blocks with less bits compared to its SOC code. Secondly, for the blocks in complex area, we introduce the AGSP predictor by utilizing the gradient information. As a result, the output code-stream can be reduced by encoding the prediction errors using Huffman coding. Finally, experimental results show the superiority of our proposed with higher embedding efficiency. Compared to Lin et al.’s scheme, our proposed method can decrease the bit rate by 7.5% and 16.7% on average when 16,129 bits

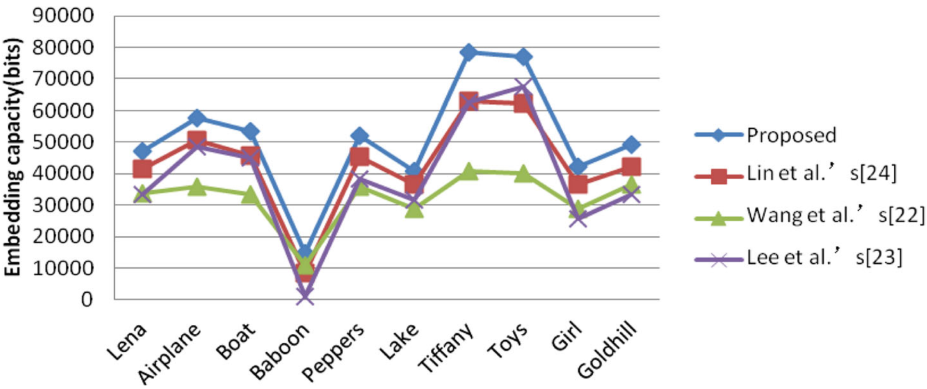


Fig. 10 Maximal embedding capacity when bit rate is fixed at 8 bpi for the proposed scheme and other schemes (N = 256)

of secret data are embedded using 256-sized and 512-sized codebook respectively, and can embed 18.0% more secret bits when the bit rate is fixed at 8 bpi using 256-sized codebook.

Acknowledgements This work is supported in part by the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) (Grant No. 201800030).

References

1. Celik MU, Sharma G, Tekalp AM, Sable E (2005) Lossless generalized-LSB data embedding. IEEE Trans Image Process 14(2):253–266
2. Chang CC, Chen GM, Lin MH (2004) Information hiding based on search-order coding for VQ indices. Pattern Recogn Lett 25(11):1253–1261
3. Chang CC, Lin CC, Tseng CS, Tai WL (2007) Reversible hiding in DCT-based compressed images. Inf Sci 177:2768–2786
4. Chang CC, Lu TC, Horng G, Huang YH (2016) Very efficient variable-length codes for the lossless compression of VQ indices. Multimed Tools Appl 75(6):3537–3552
5. Chang CC, Nguyen TS, Lin CC (2015) A reversible compression code hiding using SOC and SMVQ indices. Inf Sci 300:85–99
6. Chang CC, Wu WC (2009) A lossless data embedding technique by joint neighboring coding. Pattern Recogn 42(7):1597–1603
7. Diffie W, Hellman ME (1977) Exhaustive cryptanalysis of the NBS data encryption standard. Computer 10(6):74–84
8. Fridrich J, Goljan M, Du R (2001) Invertible authentication watermark for JPEG images. Proc. ITCC, Las Vegas
9. Gersho A, Gray RM (1992) Vector Quantization and Signal Compression. Kluwer, Norwell
10. Gray RM (1984) Vector quantization. IEEE Acoust Speech Signal Process 1:4–29
11. Hsieh CH, Tsai JC (1996) Lossless compression of VQ index with search-order coding. IEEE Trans Image Process 5(11):1579–1582
12. Huffman DA (1952) A method for the construction of minimum redundancy codes. Proc IRE 40(9):1098–1101
13. Kamstra L, Heijmans HJAM (2005) Reversible data embedding into images using wavelet techniques and sorting. IEEE Trans Image Process 14(12):2082–2090
14. Kim T (1992) Side match and overlap match vector quantizers for images. IEEE Trans Image Process 1(4):170–185
15. Lee J, Chiou Y, Guo J (2013) Lossless data hiding for VQ indices based on neighboring correlation. Inf Sci: 419–438
16. Lee JD, Chiou YH, Guo JM (2013) Lossless data hiding for VQ indices based on neighboring correlation. Inf Sci 221:419–438
17. Li X, Zhang W, Gui X (2017) Efficient Reversible Data Hiding Based on Multiple Histograms Modification. IEEE Transactions on Information Forensics & Security 10(9):2016–2027
18. Lin YK (2012) High capacity reversible data hiding scheme based upon discrete cosine transformation. J Syst Softw 85(10):2395–2404
19. Lin CC, Liu XL, Yuan SM (2015) Reversible data hiding for VQ-compressed images based on search-order coding and state-codebook mapping. Inf Sci 293:314–326
20. Lin YC, Wang CC (1999) Digital images watermarking by vector quantization. Proc Natl Comput Symp: 76–87
21. Linde Y, Buzo A, Gray RM (1980) An algorithm for vector quantizer design. IEEE Trans Commun 28(1): 84–95
22. Liu Y, Nie L, Han L, et al (2016) Action2Activity: recognizing complex activities from sensor data. 1617–1623
23. Liu Y, Nie L, Liu L et al (2016) From action to activity: Sensor-based activity recognition. Neurocomputing 181:108–115
24. Pan ZB, Hu S, Ma XX (2015) A new lossless data hiding method based on joint neighboring coding method. JVis Commun Image Represent:14–23
25. Pan ZB, Ma XX, Deng XM, Hu S (2013) Low bit-rate information hiding method based on search-order-coding technique. J Syst Softw 86:2863–2869

26. Qin C, Chang CC, Chiu YP (2014) A novel joint data-hiding and compression scheme based on SMVQ and image inpainting. IEEE Trans Image Process 23(3):969–978
27. Qin C, Hu Y (2016) Reversible data hiding in VQ index table with lossless coding and adaptive switching mechanism. Signal Process:48–55
28. Qin C, Ji P, Wang J, Chang CC (2017) Fragile image watermarking scheme based on VQ index sharing and self-embedding. Multimed Tools Appl 76(2):2267–2287
29. Ramroach S (2015) A reversible steganographic scheme for VQ indices based on joint neighboring coding. Expert Syst Appl 42(2):713–722
30. Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
31. Tang H, Kamata S (2006) A gradient based predictive coding for lossless image compression. IEICE Trans Inf Syst E89-D(7):2250–2256
32. Tian J (2003) Reversible data embedding using a difference expansion. IEEE Trans Circuits Syst Video Technol 13(8):890–896
33. Wang WJ, Huang CT, Liu CM, Su PC, Wang SJ (2013) Data embedding for vector quantization image processing on the basis of adjoining state-codebook mapping. Inf Sci 246:69–82
34. Wang JX, Lu ZM (2009) A path optional lossless data hiding scheme based on VQ joint neighboring coding. Inf Sci 179(19):3332–3348

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Zhibin Pan received the B.S. degree in Information and Telecommunication Engineering in 1985 and the M.S. degree in Automation Science and Technology in 1988 from Xi'an Jiaotong University, P.R. China, respectively. He received the Ph.D. degree in Electrical Engineering in 2000 from Tohoku University, Japan. He is a professor in the Department of Information and Telecommunication Engineering, Xi'an Jiaotong University, P.R. China. His current research interests include image compression, multimedia security and object recognition.

Erdun Gao received the B.S. degree in Information Engineering from Xi’an Jiaotong University, in 2017. He is currently working towards the Master degree in School of Electronic and Information Engineering at Xi’an Jiaotong University, Xi’an, P.R. China. His research interests include image processing and computer vision.

Ruoxin Zhu received the B.S. degree of Information Engineering in 2014 and M.S. degree of Multimedia Engineering in 2017 from Xi’an Jiaotong University, Xi’an, P.R. China. Her research interests include information hiding and image processing.

Lingfei Wang received the B.S. degree in 2011 and the Ph.D. degree in 2018 from Xi'an Jiaotong University, Xi'an, P.R. China. His research interests include information hiding and image processing.