CMC, vol.59, no.1, pp.119-133, 2019

Reversible Data Hiding in Encrypted Image Based on Block Classification Permutation

Qun Mo[1], Heng Yao[1], Fang Cao[2], Zheng Chang[3] and Chuan Qin[1, *]

Abstract: Recently, reversible data hiding in encrypted image (RDHEI) has attracted extensive attention, which can be used in secure cloud computing and privacy protection effectively. In this paper, a novel RDHEI scheme based on block classification and permutation is proposed. Content owner first divides original image into non-overlapping blocks and then set a threshold to classify these blocks into smooth and non-smooth blocks respectively. After block classification, content owner utilizes a specific encryption method, including stream cipher encryption and block permutation to protect image content securely. For the encrypted image, data hider embeds additional secret information in the most significant bits (MSB) of the encrypted pixels in smooth blocks and the final marked image can be obtained. At the receiver side, secret data will be extracted correctly with data-hiding key. When receiver only has encryption key, after stream cipher decryption, block scrambling decryption and MSB error prediction with threshold, decrypted image will be achieved. When data hiding key and encryption key are both obtained, receiver can find the smooth and non-smooth blocks correctly and MSB in smooth blocks will be predicted correctly, hence, receiver can recover marked image losslessly. Experimental results demonstrate that our scheme can achieve better rate-distortion performance than some of state-of-the-art schemes.

Keywords: Reversible data hiding, image encryption, image recovery.

1 Introduction

Digital image security is one of the important research topics in many fields, such as medical, military and some forensic images. In the past few decades, digital image has become a popular way for communication. Many schemes have been proposed for plaintext images. Generative model [Duan, Song, Qin et al. (2018)], optimal iterative BTC [Qin, Ji, Chang et al. (2018)] and image hashing [Qin, Chen, Luo et al. (2018)] are applied for image security. However, in some important areas, such as military and medical fields, we not only need to protect the security of secret information, but also need to recover original image on the receiver side, which is called as reversible data hiding (RDH). Different from robust watermarking, RDH emphasizes the extraction of

[1] School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai, 200093, China.

[2] College of Information Engineering, Shanghai Maritime University, Shanghai, 201306, China.

[3] ICD-LOSI, UMR CNRS 6281, University of Technology of Troyes, 10000 Troyes, France.

[*] Corresponding Author: Chuan Qin. Email: qin@usst.edu.cn.

CMC. doi:10.32604/cmc.2019.05770

secret data and high quality of image recovery [Yu, Zhu, Li et al. (2013); Zhou, Sun, Dong et al. (2015)]. A lot of RDH schemes for plaintext images have already been designed, such as difference expansion [Tian (2003); Thodi and Rodriguez (2007)] and histogram shifting [Ni, Shi, Ansari et al. (2006)] schemes. More recently, to improve embedding capacity, some schemes were proposed: prediction-error expansion [Ou, Li, Zhao et al. (2013)], multiple histograms modification [Li, Zhang, Gui et al. (2015)] and histogram shifting mechanism for inpainting-assisted reversible steganographic scheme [Qin, Chang, Huang et al. (2013)] were all applied for plaintext image. In 2014, Qin et al. [Qin, Chang and Chui (2014)] proposed a scheme which realized data hiding and compression at the same time. However, these schemes cannot be applied in encrypted image because the redundancy in original image is rare. To solve this problem, RDH in encrypted image allows the data hider to embed additional data into the encrypted image and realize data hiding and protect the privacy of content owner simultaneously.

In 2011, reversible data hiding in encrypted image (RDH-EI) was proposed by Zhang [Zhang (2011)], in which the three least significant bits (LSB) were flipped to hide the additional secret data. Later, improved schemes based on this scheme began to emerge. Among them, the error rate of extraction was improved, and some schemes [Hong, Chen and Wu (2012); Liao and Shu (2015)] were proposed, the capacity of embedding was increased, and some schemes proposed to embed additional data in compressed encrypted images and realize lossy compression for selective encrypted image with image inpainting [Qin, Zhou, Cao et al. (2018)].

Generally speaking, existing schemes are classified into two categories: 1) vacating room after encryption and 2) vacating room before encryption. In 2012 [Zhang (2012)], a separable RDH-EI scheme was proposed, original image was encrypted by bitwise XOR operation by an encryption key, the data-hider compressed the LSB of encrypted image to create spare space to embed additional data. This scheme can achieve data extraction, image decryption with high quality and recovered image. However, when the amount of additional data was larger than 0.04 bit per pixel (bpp), it was difficult to recover original image correctly. In 2013, Ma et al. [Ma, Zhang, Zhao et al. (2013)] proposed a separable scheme with good quality of recovery and embedding rate, however, before the encryption, a pre-processing operation by the sender should be applied, and the scheme was actually a conventional RDH in plaintext image. In 2014, Wu et al. [Wu and Sun (2014)] proposed a RDH-EI scheme based on prediction error, additional data was embedded in the most significant bit (MSB) of encrypted image. The problems of this scheme were: (1) the quality of decrypted image is low, even if the payload was 0.016 bpp, PSNR value was about 35 dB for Lena and 24 dB for Baboon; (2) for the texture image, such as Baboon, the quality of recovered image was very low. Later, Qian et al. [Qian and Zhang (2016); Qian, Zhang and Feng (2016)] used distributed source encoding and progressive recovery to realize reversible data hiding in encrypted images, respectively. In 2018, Qin et al. [Qin, Zhang, Cao et al. (2018)] proposed a novel scheme via block selection to embed secret data in encrypted images.

In summary, the above schemes cannot guarantee high embedding rate together with a high quality of recovered image. To solve this problem, this work proposes a new scheme based on block classification permutation, in which original image is divided into non-overlapping blocks and all blocks are classified into smooth blocks and non-smooth blocks according to threshold. The classification permutation aims to scramble the pixel positions and make the data-hider classify the smooth and non-smooth blocks easily. Secret data are embedded in the MSB of encrypted pixels in smooth blocks. At the receiver side, if the receiver only has data hiding key, secret information will be extracted correctly. When encryption key is possessed, MSB in all blocks will be predicted with threshold and decrypted image can be achieved. When data hiding key and encryption key are both obtained, receiver can find the smooth and non-smooth blocks correctly and MSB in smooth blocks will be predicted correctly and final recovered image will be obtained. Experimental results demonstrate that the quality of recovered image and embedding rate of our scheme outperform those of the above schemes.

The rest of the paper is organized as follows. The proposed scheme is described in detail in Section 2. Section 3 provides experimental results and analysis. Finally, the conclusion is drawn and future work is mentioned in the last part.

2 Proposed scheme

2.1 Overview

The schematic diagram of the proposed scheme is shown in Fig. 1. For the content owner, block classification, block permutation and image encryption are conducted on original image; for the data hider, data embedding with MSB substitution is conducted on smooth blocks of encrypted image; for the receiver, secret information, decrypted image and recovered image can be obtained with different keys.

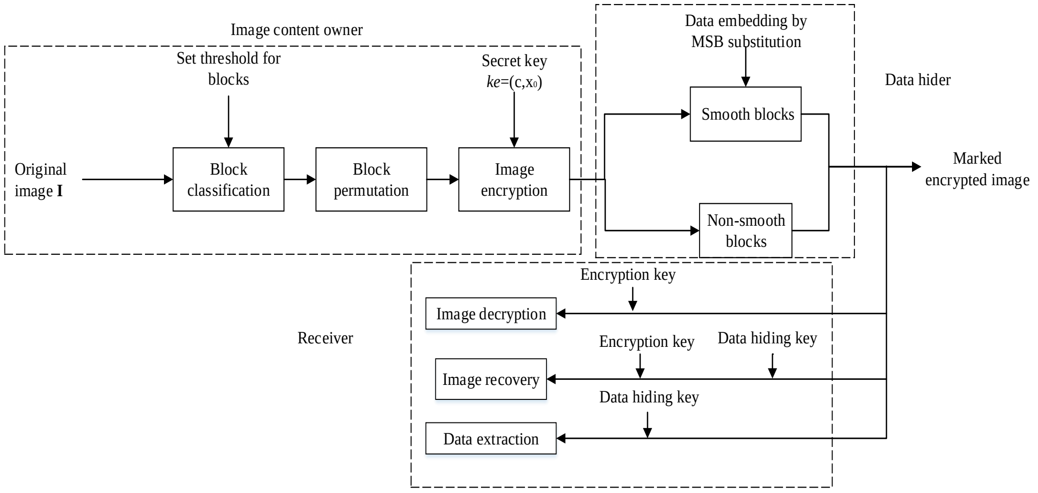


Figure 1: Schematic diagram of the proposed scheme

2.2 Block classification and permutation

2.2.1 Block classification

Assume the original image **I** with a size of $M \times N$ pixels and each pixel $\mathbf{I}_{x,y}$ with gray value falling into [0, 255] is represented by 8 bits, where both M and N are power of 2, and

$1<x<M$, $1<y<N$. In this work, first, the original image **I** is divided into $K\times K$ non-overlapping blocks. For each block, we denote the maximum and the minimum of the pixels as p_{max} and p_{min}, respectively. The difference between p_{max} and p_{min} can be calculated as:

$$D_z=p_{max}-p_{min} \tag{1}$$

where D_z represents the difference between p_{max} and p_{min} in the z-th block. We set a same threshold D_{T} for all blocks, if $D_z<D_{\mathrm{T}}$, the block is classified as a smooth block, otherwise, as a non-smooth block. Generate a matrix **U** with the size of $(M/K)\times(N/K)$ to indicate whether the block is smooth or not. Detailedly, each smooth and non-smooth block are marked by 0 and 1, respectively.

2.2.2 Block permutation

After block classification, for the sake of security, Josephus traversing permutation is applied on image blocks, which contains three parameters, including the number of blocks of original image **I** recorded as n, starting position J_{s}, and the counting period J_{c}. With these three parameters, image **I**′ can be obtained with a Josephus block permutation:

$$\mathbf{I}'=J\,(n, J_{\mathrm{s}}, J_{\mathrm{c}}) \tag{2}$$

To provide an example, assuming n=9, J_{s} and J_{c} are set as 2 and 3, respectively andthe total initial sequence can be represented as:

$$\mathbf{I}=\{\mathbf{B}_0, \mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3, \mathbf{B}_4, \mathbf{B}_5, \mathbf{B}_6, \mathbf{B}_7, \mathbf{B}_8\} \tag{3}$$

where $\mathbf{B}_i$ represents the i-th block of the original image **I** (i=0, 1, ..., 8). Apply our parameters to Eq. (2), i.e., $\mathbf{I}'=J\,(9, 2, 3)$, a scrambled sequence is produced:

$$\mathbf{I}'=\{\mathbf{B}_3, \mathbf{B}_6, \mathbf{B}_0, \mathbf{B}_4, \mathbf{B}_8, \mathbf{B}_5, \mathbf{B}_2, \mathbf{B}_7, \mathbf{B}_1\} \tag{4}$$

According to the new image **I**′ after block scrambling, a new mark matrix **U**′ can be obtained, where $n=(M/K)\times(N/K)$, J_s and J_c is set by the encrypted key. Fig. 2 shows an example, in which 4×4 blocks are shown, where K=4, D_{T}=10 and J=(16, 2, 3). These 16 blocks are classified into smooth and non-smooth blocks according to D_{T}, after permutation, a new image **I**′ with block classification and permutation is generated.

Block1	Block2	Block3	Block4
Block5	Block6	Block7	Block8
Block9	Block10	Block11	Block12
Block13	Block14	Block15	Block16

(a)

1	0	0	1
0	1	0	0
0	1	1	0
1	0	1	1

(b)

Block4	Block7	Block10	Block13
Block16	Block3	Block8	Block12
Block1	Block6	Block14	Block5
Block15	Block11	Block2	Block9

(c)

1	0	1	1
1	0	0	0
1	1	0	0
1	1	0	0

(d)

Figure 2: Example of block classification permutation. (a) Initial blocks of original image **I**, (b) Mark matrix **U**, (c) Blocks after permutation of image **I**′, (d) Mark matrix **U**′

2.3 Image encryption

2.3.1 Stream cipher encryption

After block classification and permutation, a new image **I′** is generated. In order to make the image **I′** indistinguishably, we encrypt it by using an encryption key: k_e=(c, x_0). The elements of this key are used as parameters of chaotic generator. First, the image owner turns the new image **I′** into plain bits by decomposing each pixel into 8 bits as:

$$b_{x,y,u} = \lfloor I'_{x,y} / 2^u \rfloor \bmod 2, u = 0,1,2...7 \tag{5}$$

where $I'_{x,y}$ is the pixel of image **I′** with coordinate (x, y). By using this chaotic generator, a sequence of pseudo-random bytes $s_{x,y,u}$ is obtained and bitstream of the encrypted image can be calculated through exclusive-or(XOR) operation:

$$e_{x,y,u} = b_{x,y,u} \oplus s_{x,y,u}, u = 0,1,2,...7 \tag{6}$$

Accordingly, an encrypted image **E** can be constructed by:

$$E_{x,y} = \sum_{u=0}^{7} e_{x,y,u} \cdot 2^u \tag{7}$$

where $E_{x,y}$ is the pixel of encrypted image with coordinate (x, y).

2.3.2 Self-embedding of *U′*

In order to extract embedded secret information correctly and recover the original image losslessly, mark matrix **U′** needs to be embedded in the encrypted image. The p-th element of mark matrix **U′** is stored in the LSB of encrypted pixel with coordinate (K^2, K^2) of the p-th block, where $0<p<(M/K)\times(N/K)$. At the same time, the value of original LSB located in coordinate (K^2, K^2) of the p-th block is stored as additional secret information to embed in encrypted image.

2.4 Data embedding

After image encryption, with the embedded mark matrix **U′**, the data hider can classify all divided blocks into smooth and non-smooth blocks respectively. For smooth blocks, MSB of each pixel in addition to the first pixel of smooth blocks are substituted by b_k with:

$$E'_{s(x,y)} = b_k \cdot 128 + (E_{s(x,y)} \bmod 128) \tag{8}$$

where $0<k<M\times N$, $E_{s(x,y)}$ represents pixel of smooth blocks in image **E′** with coordinate (x, y), $E'_{s(x,y)}$ represents pixel of smooth blocks after data embedding. Hence, a final marked image **E′** can be obtained.

2.5 Data extraction

In the data extraction phase, firstly, if the receiver only has data-hiding key, the receiver can classify the marked image **E′** into $K\times K$ non-overlapping blocks and LSB of each pixel with coordinates (K^2, K^2) in the current block can be obtained. In other words, the concrete position and number of smooth blocks and non-smooth blocks can be acquired, the number of smooth blocks is recorded as n_1 and the number of non-smooth blocks is recorded as n_2, meanwhile, the MSB of each pixel of smooth blocks are extracted by:

$$b_k = E'_{s(x,y)} / 128 \tag{9}$$

where $0<k<n_1\times K^2$, therefore, the secret information embedded in encrypted image can be obtained correctly. At the same time, the original LSB embedded as additional information will also be achieved.

2.6 Image decryption and recovery

2.6.1 Image decryption

When the receiver only has the encryption key (stream ciphers key and scrambling encryption key), the receiver can obtain the decrypted image $\mathbf{I}_\mathrm{d}$, the specific steps are as follows:

Step 1: The stream cipher key k_e is used to generate the sequence $s(x,y)$, with $M\times N$ pseudo-random bytes. The pixels of the marked encrypted image are browsed in the scan line order and all pixels are decrypted by conducting XOR for the value $e_{x,y,u}$ with the associated binary sequence $s_{x,y,u}$ in the pseudo-random stream:

$$b'_{x,y,u} = e_{x,y,u} \oplus s_{x,y,u} \tag{10}$$

Accordingly, a decrypted image $\mathbf{I}_\mathrm{d}$ can be constructed by:

$$I_{d(x,y)} = \sum_{u=0}^{7} b'_{x,y,u} \cdot 2^u \tag{11}$$

Note that, secret data is embedded in the MSB of pixels in smooth blocks, and their MSB cannot be decrypted correctly.

Step 2: In the case where the scramble key is known, blocks before permutation can be achieved. We can get the image $\mathbf{I}_\mathrm{d}$ before block classification permutation in which there are some pixels whose MSB may be error in smooth blocks. To obtain the decrypted image with high quality, the pixel values of smooth blocks are adjusted by estimating its MSB.

Step 3: The decrypted image is divided into $K\times K$ non-overlapping blocks, for each block, difference between each pixel and the first pixel can be calculated and recorded as $D'_{x,y}$. If $D'_{x,y} > 0$, i.e., other pixel value is larger than the first pixel value in the current block, when $128 - D_\mathrm{T} < \left|D'_{x,y}\right| < 128 + D_\mathrm{T}$, the current decrypted pixel value can be calculated by:

$$I_{d(x,y)} = I_{d(x,y)} - 128 \tag{12}$$

If $D'_{x,y} < 0$, i.e., other pixel value is smaller than the first pixel in the current block, when $128 - D_\mathrm{T} < \left|D'_{x,y}\right| < 128 + D_\mathrm{T}$, the current decrypted pixel value can be calculated by:

$$I_{d(x,y)} = I_{d(x,y)} + 128 \tag{13}$$

In this way, the decrypted image $\mathbf{I}_\mathrm{d}$ can be obtained.

2.6.2 Image recovery

If the receiver both has data hiding key and encryption key, the original image **I** can be recovered with high quality. When data hiding key is achieved, smooth and non-smooth blocks can be achieved. For pixels in non-smooth blocks, the pixel values remain unchanged, and for those pixel values in smooth blocks, we calculate difference between each pixel value and the first pixel value and the differences are recorded as $D'_{s(x,y)}$. If $D'_{s(x,y)} > 0$, i.e., other pixel value is larger than the first pixel in the current block, when $128 - D_{\mathrm{T}} < \left|D'_{s(x,y)}\right| < 128 + D_{\mathrm{T}}$, the current recovered pixel value can be calculated by:

$$I_{s(x,y)} = I_{s(x,y)} - 128 \tag{14}$$

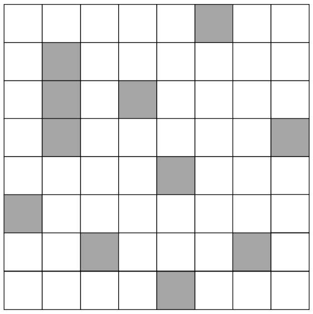
Where $I_{s(x,y)}$ represents the pixel in smooth blocks.

If $D'_{s(x,y)} < 0$, i.e., other pixel value is smaller than the first pixel in the current block, when $128 - D_{\mathrm{T}} < \left|D'_{s(x,y)}\right| < 128 + D_{\mathrm{T}}$, the current recovered pixel value can be calculated by:

$$I_{s(x,y)} = I_{s(x,y)} + 128 \tag{15}$$

In this way, the recovered image **I** can be obtained.

For example, Fig. 3 shows an example, in which the process of MSB prediction and image recovery is shown. Fig. 3(a) shows the image after stream ciphers decryption and block permutation decryption. It contains 64 blocks, each block includes 4×4 pixel values. The smoothness corresponding to each block is shown in Fig. 3(b), in which, 0 represents smooth blocks and 1 represents non-smooth blocks. We mark the non-smooth blocks with gray and use white to mark the smooth blocks. Pixel values in non-smooth blocks are kept unchanged. With Eq. (14) and Eq. (15), the MSB value can be recovered correctly. Five blocks identified in red is used as a concrete example, the pixel values in the five blocks are shown in Fig. 3(c). When threshold D_{T}=15, by calculating the difference $D'_{s(x,y)}$, we correct the MSB value of pixel in current smooth block. After correcting all pixel values, we can get the recovered image with high quality.



(a)

0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0
0	1	0	0	0	0	0	1
0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0

(b)

				107	104	95	100				
				99	95	88	95				
				97	100	98	100				
				103	103	97	102				
94	98	98	102	97	109	107	99	97	224	99	225
99	227	223	103	97	109	107	99	225	225	224	98
103	101	97	227	92	99	93	96	97	97	220	102
99	104	94	226	96	97	101	99	99	98	98	101
				101	95	228	234				
				221	226	92	92				
				225	93	96	227				
				219	222	227	95				

(c)

				107	104	95	100				
				99	95	88	95				
				97	100	98	100				
				103	103	97	102				
94	98	98	102	97	101	101	97	97	96	99	97
99	99	95	103	97	109	107	99	97	97	96	98
103	101	97	99	92	99	93	96	97	97	92	102
99	104	94	98	96	97	101	99	99	98	98	101
				101	95	100	106				
				93	98	92	92				
				97	93	96	99				
				91	94	99	95				

(d)

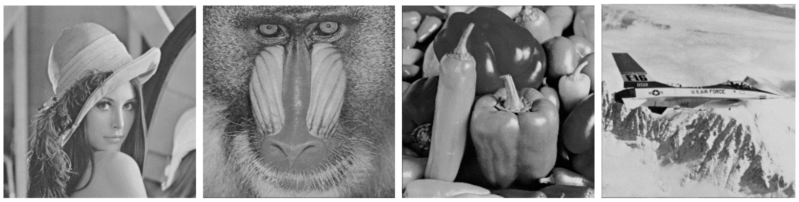
Figure 3: (a) Initial blocks after encryption and classification rearrangement, (b) Mark matrix **U**′, (c) Pixels in blocks marked with red, (d) Pixels after MSB prediction

3 Experimental results and comparisons

In this section, we present the obtained experimental results, Section 3.1 gives a full example for different test images and shows the obtained results. Then, in Section 3.2 we analysis performance of our proposed scheme from different statistical metrics. Finally, in Section 3.3, we compare embedding rate and recovered image quality of our proposed scheme and other three state-of-the-art schemes.

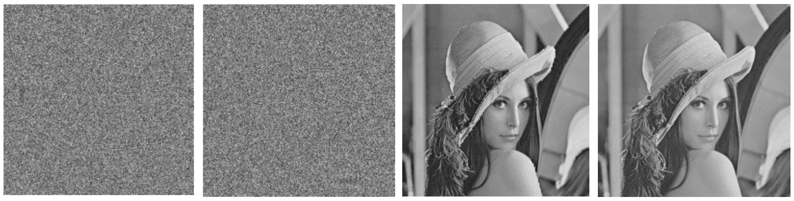
3.1 Results of our scheme

We first applied our scheme on four different test original images of 512×512 pixels, illustrated in Fig. 4. Fig. 5 gives an example of our proposed scheme for *Lena* sized 512×512. Fig. 5(a) shows encrypted image *Lena*. Fig. 5(b) is the marked image with secret bits embedded. Fig. 5(c) is the decrypted image and Fig. 5(d) is the recovered image.



(a) (b) (c) (d)

Figure 4: Test image. (a) *Lena*, (b) *Baboon*, (c) *Peppers*, (d)*Airplane*



(a) (b) (c) (d)

Figure 5: An example of the proposed scheme with K=4 and the threshold D_T is 15. (a) The encrypted image after classification and permutation, (b) Marked image with 111489 bits embedded, (c) Decrypted image after MSB prediction with PSNR=31.72 dB, (d) The losslessly recovered image

3.2 Performance analysis

We perform an analysis to evaluate performance of our proposed scheme. We use different statistical metrics: horizontal, vertical and diagonal correlation coefficients; Shannon entropy; PSNR between the original image and decrypted and recovered image and embedding rate of secret information.

(a) Shannon entropy:

$$H(\beta) = -\sum_{i=1}^{h} P(\beta_i)\log_2 P(\beta_i) \tag{16}$$

where β denotes the source consisting of h symbols, i.e., β_1, β_2, …, β_h; $P(\beta_i)$ denotes the probability of the symbol β_i.

(b) Horizontal, vertical and diagonal correlation coefficients:

$$Corr = \frac{S\sum_{i=1}^{S}(c_i \times d_i) - \sum_{i=1}^{S} c_i \times \sum_{i=1}^{S} d_i}{\sqrt{(S\sum_{i=1}^{S} c_i^{\,2} - (\sum_{i=1}^{S} c_i)^2) \times (S\sum_{i=1}^{S} d_i^{\,2} - (\sum_{i=1}^{S} d_i)^2)}} \tag{17}$$

where c_i and d_i are two data sequence of adjacent pixels including horizontal, vertical and diagonal directions of the plain image and encrypted image, respectively. S is the sequence length.

(c) Peak-signal-to noise ratio (PSNR):

$$PSNR = 10\log_{10}\frac{255^2}{\frac{1}{M \times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(I_d(x,y) - I(x,y))^2} \tag{18}$$

where $\mathbf{I_d}$ is the decrypted image and $\mathbf{I}$ is original image

(d) Embedding rate:

$$rate = \frac{n_1 \times (K \times K - 1) - 2 \times (N/K) \times (M/K)}{N \times M} \tag{19}$$

Embedding rate is expressed in bit per pixel, where n_1 represents the number of smooth blocks and K represents the block size. The embedding amount of mark matrix **U** and the original LSB values embedded as extra secret information should be considered.

Shannon's entropy provides a standard to evaluate randomness of image, which can be calculated with Eq. (16). The entropy should be close to 8 for an ideal random image with 256 grey levels. The information entropies of original image and encrypted image are shown in Tab. 1.

We can find that the calculated entropies of encrypted images are extremely close to the ideal value 8. Meanwhile, the correlation between two adjacent pixels in encrypted image is also a good standard to evaluate good encryption performance. Correlation coefficients will be calculated with Eq. (17) and the results are shown in Tab. 2, we can find that the correlation coefficients of adjacent pixels of plain image are close to 1, while the correlation coefficients of adjacent pixels of encrypted image are close to 0, which indicates that our proposed scheme have a good encryption performance.

Table 1: Entropy analysis of our proposed scheme

	Entropy values (bpp)	
Image	Original image	Encrypted image
Lena	7.4474	7.9906
Baboon	7.1391	7.9907
Peppers	7.5715	7.9699
Airplane	5.5716	7.9906
Barbara	7.4664	7.9907
Couple	7.0581	7.9907

Table 2: Correlation analysis of our proposed scheme

	Horizontal correlation coefficients	Vertical correlation coefficients	Diagonal correlation coefficients	Horizontal correlation coefficients	Vertical correlation coefficients	Diagonal correlation coefficients
Image	Original image	Original image	Original image	Encrypted image	Encrypted image	Encrypted image
Lena	0.9676	0.9843	0.9558	0.0018	0.0115	-0.0204
Baboon	0.8673	0.7552	0.7264	-0.0024	0.0231	0.0128
Peppers	0.9764	0.9817	0.9653	-0.0147	-0.0108	0.0074
Airplane	0.9689	0.9518	0.9253	0.0132	0.0148	-0.0055
Barbara	0.8533	0.9589	0.8431	-0.0032	0.0052	-0.0003
Couple	0.9436	0.9495	0.9088	-0.0034	-0.0103	-0.0105

Table 3: PSNR values and embedding rate of decrypted and recovered image *Lena*

			Decrypted image		Recovered image
Test images	$K \times K$	D_{T}	PSNR(db)	Embedding rate	PSNR
Lena	4×4	30	30.55	0.5774	+∞
		25	30.82	0.4774	
		20	31.13	0.4639	
		15	31.72	0.3628	
	8×8	30	24.78	0.4800	+∞
		25	25.22	0.4203	
		20	25.61	0.3369	
		15	26.10	0.2233	
	16×16	30	19.89	0.2627	+∞
		25	20.07	0.2061	
		20	20.58	0.1514	
		15	21.22	0.0488	

Table 4: PSNR values and embedding rate of decrypted and recovered image *Baboon*

		Decrypted image			Recovered image
Test images	*K*×*K*	D_T	PSNR(db)	Embedding rate	PSNR
Baboon	4×4	30	30.90	0.1635	+∞
		25	31.34	0.1014	
		20	31.85	0.0363	
		15	32.45	0.0178	
	8×8	30	27.20	0.0935	+∞
		25	28.15	0.0408	
		20	29.26	0.0020	
		15	30.41	--	
	16×16	30	23.67	0.0170	+∞
		25	24.90	0.0020	
		20	26.36	--	
		15	28.06	--	

Table 5: PSNR values and embedding rate of decrypted and recovered image *Peppers*

		Decrypted image			Recovered image
Test images	*K*×*K*	D_T	PSNR(db)	Embedding rate	PSNR
Peppers	4×4	30	27.61	0.5900	+∞
		25	28.10	0.5384	
		20	28.69	0.4545	
		15	29.37	0.3094	
	8×8	30	20.79	0.4513	+∞
		25	21.10	0.3732	
		20	21.45	0.2640	
		15	22.10	0.1050	
	16×16	30	16.30	0.1950	+∞
		25	16.61	0.1318	
		20	17.01	0.0596	
		15	17.55	0.0020	

In the process of encryption, the original image is divided into *K*×*K* blocks. We set a threshold D_T to classify smooth and non-smooth blocks. The size of image block and threshold will directly affect the quality of decrypted image. We test PSNR values and embedding rate of the decrypted and recovered image under different *K* and threshold. The results are shown in Tab. 3, Tab. 4, Tab. 5 and Tab. 6. For decrypted image, when *K* remains unchanged, the embedding rate becomes larger as the threshold increases, while

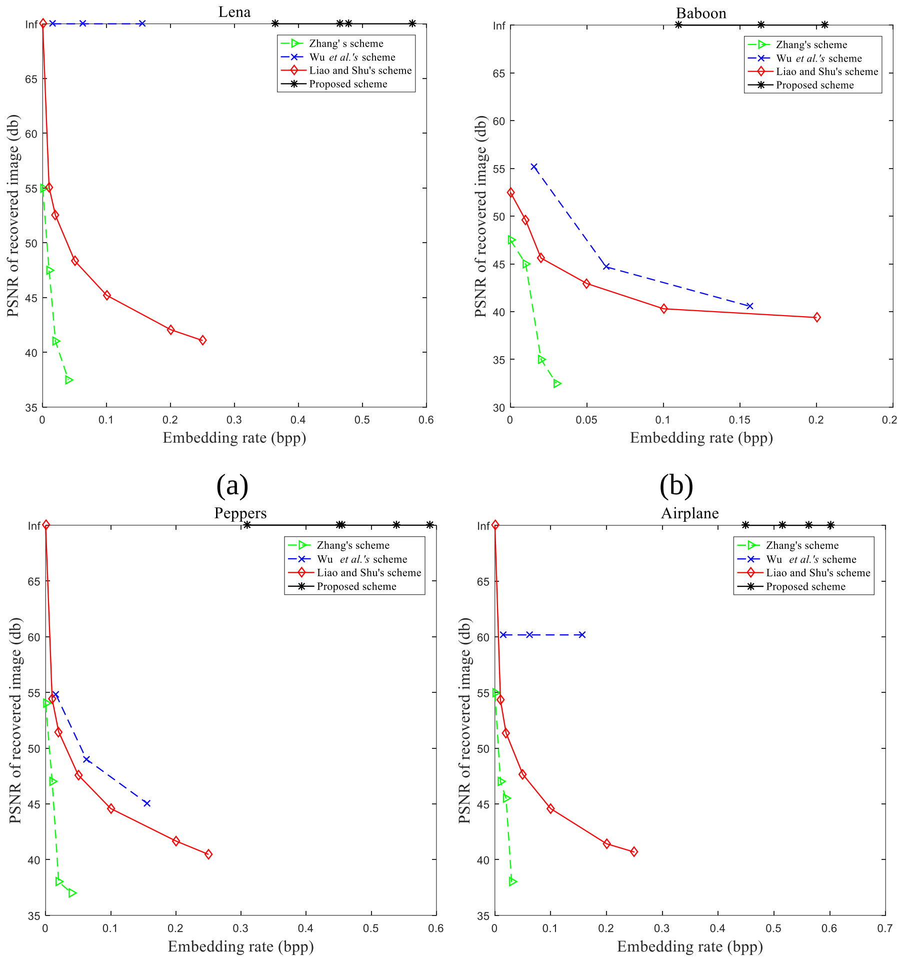
PSNR value decreases. When threshold value remains unchanged, both embedding rate and PSNR value decrease with the increase of *K*. For the four test images, complete reversibility can be obtained by our proposed scheme. Therefore, according to different images, setting a reasonable threshold and block size *K* will achieve a good balance between embedding rate and PSNR values of decrypted image.

Table 6: PSNR values and embedding rate of decrypted and recovered image *Airplane*

			Decrypted image		Recovered image
Test images	*K*×*K*	D_T	PSNR(db)	Embedding rate	PSNR
Airplane	4×4	30	26.86	0.6014	+∞
		25	27.24	0.5628	
		20	27.66	0.5152	
		15	29.36	0.4494	
	8×8	30	22.50	0.5000	+∞
		25	24.00	0.4816	
		20	25.55	0.4388	
		15	26.33	0.3278	
	16×16	30	17.01	0.3939	+∞
		25	17.84	0.3511	
		20	18.67	0.2587	
		15	19.58	0.1449	

3.3 Comparisons with three state-of-the-art schemes

We conduct some comparisons in terms of embedding rate and PSNR of recovered image between our scheme and three schemes including Zhang's scheme [Zhang (2011)], Wu et al.'s scheme [Wu and Sun (2014)] and Liao et al.'s scheme [Liao and Shu (2015)]. First of all, Fig. 6 shows comparison results of rate-distortion curves for the relationship between embedding rate and visual quality of recovered image. It can be found that, our scheme can acquire better rate-distortion performance than other three schemes for the four test images sized 512×512, including *Lena*, *Baboon*, *Peppers* and *Airplane*. For the recovered image, our scheme allows us to have a higher embedding rate and PSNR value than other schemes. Under the same PSNR value, the embedding rate of our scheme is higher than other three schemes.



(a) (b)

(c) (d)

Figure 6: Comparisons rate-distortion curves for recovered image among the proposed scheme and other three schemes. (a) *Lena*. (b) *Baboon*. (c) *Peppers*. (d) *Airplane*

4 Conclusions

In this paper, a novel scheme of reversible data hiding in encrypted image based on block classification and permutation is proposed. Before encryption, we set a threshold to classify original image into smooth and non-smooth blocks, respectively. Then all blocks are permuted according to Josephus scrambling and mark matrix **U** is generated to mark the smooth and non-smooth blocks respectively. In the encryption process, the image is encrypted with the stream cipher. In data embedding process, secret data is embedded in the MSB of each encrypted pixel expect the first pixel in current smooth blocks. At the receiver side, when data-hiding key is possessed, secret data can be extracted correctly. When encryption key is possessed, MSB in all blocks will be predicted with threshold and decrypted image can be achieved. In order to recover the original image reversibly, the receiver should both has data hiding key and encryption key. Compared with three proposed schemes, the rate-distortion performance of our proposed scheme is better. In the future work, the storage space of the mark matrix should be decreased and the quality of decrypted image should be further improved.

Acknowledgement: This work was supported by the National Natural Science Foundation of China (61672354, 61702332). The authors would like to thank the anonymous reviewers for their valuable suggestions.

References

Duan, X. T.; Song, H. X.; Qin, C.; Muhammad, K. K. (2018): Coverless steganography for digital images based on a generative model. *Computers, Materials & Continua*, vol. 55, no. 3, pp. 483-493.

Hong, W.; Chen, T.; Wu, H. Y. (2012): An improved reversible data hiding in encrypted images using side match. *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199-202.

Li, X. L.; Zhang, W. M.; Gui, X. L.; Yang, B. (2015): Efficient reversible data hiding based on multiple histograms modification. *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016-2027.

Liao, X.; Shu, C. W. (2015): Reversible data hiding in encrypted images based on absolute mean difference of neighboring pixels. *Journal of Visual Communication and Image Representation*, vol. 28, pp. 21-27.

Ma, K. D.; Zhang, W. M.; Zhao, X. F.; Yu, N. H.; Li, F. H. (2013): Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553-562.

Ni, Z.; Shi, Y. Q.; Ansari, N.; Su, W. (2006): Reversible data hiding. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354-362.

Ou, B.; Li, X, L.; Zhao, Y.; Ni, R. R.; Shi, Q. Y. (2013): Pairwise prediction-error expansion for efficient reversible data hiding. *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010-5021.

Qian, Z. X.; Zhang, X. P. (2016): Reversible data hiding in encrypted image with distributed source encoding. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 636-646.

Qian, Z. X.; Zhang, X. P.; Feng, G. R. (2016): Reversible data hiding in encrypted images based on progressive recovery. *IEEE Signal Processing Letters*, vol. 23, no. 11, pp. 1672-1676.

Qin, C.; Chang, C. C.; Huang, Y. H.; Liao, L. T. (2013): An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 7, pp. 1109-1118.

Qin, C.; Chang, C. C.; Chiu, Y. P. (2014): A novel joint data-hiding and compression scheme based on SMVQ and image inpainting. *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 969-978.

Qin, C.; Ji, P.; Chang, C. C.; Dong, J.; Sun, X, M. (2018): Non-uniform watermark sharing based on optimal iterative BTC for image tampering recovery. *IEEE Multimedia*, vol. 25, no. 3, pp. 36-48.

Qin, C.; Chen, X. Q.; Luo, X. Y.; Zhang, X. P.; Sun, X. M. (2018): Perceptual image hashing via dual-cross pattern encoding and salient structure detection. *Information Sciences*, vol. 423, pp. 284-302.

Qin, C.; Zhou, Q.; Cao, F.; Dong, J.; Zhang, X. P. (2018): Flexible lossy compression for selective encrypted image with image inpainting. *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1.

Qin, C.; Zhang, W.; Cao, F.; Zhang, X. P.; Chang, C. C. (2018): Separable reversible data hiding in encrypted images via adaptive embedding strategy with block selection. *Signal Processing*, vol. 153, pp. 109-122.

Tian, J. (2003): Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890-896.

Thodi, D. M.; Rodriguez, J. J. (2007): Expansion embedding techniques for reversible watermarking. *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721-730.

Wu, X. T.; Sun, W. (2014): High-capacity reversible data hiding in encrypted images by prediction error. *Signal Processing*, vol. 104, pp. 387-400.

Yu, J.; Zhu, G. P.; Li, X. L.; Yang, J. Q. (2013): An improved algorithm for reversible data hiding in encrypted image. *Lecture Notes in Computer Science*, vol. 7809, pp. 384-394.

Zhang, X. P. (2011): Reversible data hiding in encrypted image. *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255-258.

Zhang, X. P. (2012): Separable reversible data hiding in encrypted image. *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826-832.

Zhou, J.; Sun, W.; Dong, L.; Liu, X.; Au, O. C. et al. (2016): Secure reversible image data hiding over encrypted domain via key modulation. *IEEE Transaction Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 441-452.