High Capacity Reversible Data Hiding in Encrypted Images Based on Pixel Value Preprocessing and Block Classification

Tao Zhang, Ju Zhang, Yicheng Zou, and Yu Zhang(✉)

College of Computer and Information Science, Southwest University, Chongqing 400715, China
zhangyu@swu.edu.cn

Abstract. Reversible data hiding in encrypted images (RDHEI) can simultaneously achieve secure transmission of images and secret storage of embedded additional data, which can be used for cloud storage and privacy protection. In this paper, an RDHEI scheme based on pixel value preprocessing and block classification (PVPBC-RDHEI) is proposed. The content owner first preprocesses the pixel values of the original image and then encrypts the image by combining bitwise exclusive-or operation and block permutation, so that the correlation between neighboring pixels in the encrypted image block is preserved. Upon receiving the encrypted image, the data hider classifies all blocks into six types based on the number of continuously consistent value bit planes counted from top-down in the pixel block, and generates corresponding indicators for six block types using Huffman coding. Therefore, each pixel block in the image can vacate room for data embedding. The receiver can separately extract the additional data or recover the original image according to the different keys. The experimental results show that the embedding rate of the proposed scheme is significantly superior to the state-of-the-art schemes, and the security and reversibility are guaranteed.

Keywords: Reversible data hiding · Encrypted images · Pixel value preprocessing · Block classification

1 Introduction

With the rapid development of information technology, people's demand for data security and privacy protection is becoming increasingly urgent. To protect sensitive information from unauthorized access and tampering, data hiding technology has been extensively studied. Traditional data hiding methods can cause permanent distortion of the image by embedding additional data into the carrier image [1]. To solve this problem, reversible data hiding (RDH) technology has emerged. Reversible data hiding can recover the original image losslessly after extracting the embedded data. Due to the reversibility, RDH can be applied to some special scenarios, e.g., medical and military imagery, law forensics and

S. Rudinac et al. (Eds.): MMM 2024, LNCS 14555, pp. 14–27, 2024.
https://doi.org/10.1007/978-3-031-53308-2_2

precision machining. To date, numerous RDH methods have been proposed and they can be roughly classified into five categories: differential expansion (DE) [2–4], histogram shifting (HS) [5–7], lossless compression [8–10], prediction-error expansion (PEE) [11,12], and pixel value ordering (PVO) [13,14].

In recent years, with the popularity of cloud storage, more and more users tend to upload data to the cloud for storage or sharing. However, due to certain security risks in cloud storage, the problem of data privacy leakage has become increasingly prominent. In response to this challenge, the combination of reversible data hiding and image encryption, i.e., reversible data hiding in encrypted images (RDHEI) has become a new research hotspot. This approach enables both secure transmission of images and secret storage of embedded additional data. Up to now, many RDHEI methods have been proposed. According to the point of view of vacating room, the existing methods can mainly be divided into two categories: reserving room before encryption (RRBE) [15–18] and vacating room after encryption (VRAE) [19–25].

In RRBE schemes, the content owner preprocesses the original image using its spatial correlation to reserve room before image encryption. The data hider uses the spare room to embed additional data after receiving the encrypted image. Ma et al. [15] first proposed the RRBE method by using the traditional RDH method to embed the least significant bit (LSB) of some pixels into other pixels to reserve room. Chen and Chang [16] combined extended run-length encoding and a block-based most significant bit (MSB) plane rearrangement mechanism to compress the MSB planes of the image, thus generating room for high-capacity embedding. Puteaux and Puech [17] recursively processed each bit-plane of an image from MSB to LSB by combining error prediction, reversible adaptation, encryption and embedding. Gao et al. [18] located prediction errors by generating the error location binary map, and marked the location of prediction errors with error blocks and embedded additional data with message blocks.

Unlike RRBE schemes, in VRAE schemes, the content owner only needs to choose a suitable encryption algorithm to encrypt the original image. Then, the data hider reserve room in the encrypted image to embed additional data. Zhang [19] used stream cipher technology to encrypt the original image, and then the data hider divides the encrypted image into non-overlapping blocks and embeds a secret bit by flipping the last three LSBs of half pixels in each block. Qian and Zhang [20] proposed a separable method for encrypted images using Slepian-Wolf sources coding which achieved a high embedding payload and good image reconstruction quality. Later, block-based image encryption schemes have been proposed and applied to VRAE. Qin et al. [21] employed an efficient encryption method to encrypt each original block while transferring redundancy from MSB to LSB. The additional data is then embedded into the LSB of the encrypted block by the sparse matrix encoding method. Fu et al. [22] divided the encrypted blocks into embeddable and non-embeddable blocks by analyzing the distribution of MSB layers and then used Huffman coding to adaptively compress embeddable blocks according to the occurrence frequency of the MSB, and the vacated room can be used for additional data embedding. Wang et al. [23] classified all blocks into usable blocks (UBs) and unusable blocks (NUBs) while

preserving intra-block pixel correlation. Then, since the pixels in the block share the same MSBs, the UB is reconstructed to vacate room for data embedding. Wang et al. [24] classified and encoded all encrypted blocks based on the number of MSB bit planes where all values are '1' or '0'. After embedding the indicators that are generated by Huffman coding into the first MSB bit plane of each block, the remaining bits of these MSB bit planes can be vacated for additional data embedding. Wang and He [25] proposed a novel RDHEI method based on adaptive MSB prediction. The encrypted image is first divided into 2×2 sized non-overlapping blocks and then the other three pixels are predicted with the upper-left pixel within the encrypted block such that the embedding room is vacated. Although the embedding rate of these block-based encryption schemes has increased, there is still some room for improvement.

In this paper, we propose an efficient reversible data hiding scheme for encrypted images based on pixel value preprocessing and block classification (PVPBC-RDHEI) to further improve the embedding rate. In our scheme, the content owner first preprocesses the pixel values of the original image and then encrypts the original image block by block. The data hider classifies all blocks into six types based on the number of continuously consistent value bit planes counted from top-down in the pixel block, and uses Huffman coding to generate corresponding indicators for six block types. Therefore, each pixel block in the image can embed data. On the receiving side, data extraction and image recovery can be separated according to the image encryption key and the data hiding key. The embedding rate of our scheme is superior to some existing RDHEI schemes and the main contributions of our scheme are summarized as follows:

1. By preprocessing the pixel values of the original image, each pixel block in the image can embed data.
2. Block-level encryption of the image combined with bitwise XOR operation and block permutation preserves the correlation of pixels within the block.
3. Our scheme achieves a higher embedding rate than state-of-the-art schemes, and the data extraction and image recovery are also separable and error-free.

The rest of this paper is organized as follows. Sect. 2 describes the detailed procedures of the proposed PVPBC-RDHEI scheme. In Sect. 3, the experimental results and analysis are provided. Finally, conclusions are presented in Sect. 4.

2 The Proposed PVPBC-RDHEI Scheme

This section describes our proposed PVPBC-RDHEI scheme. Figure 1 shows the framework of the proposed PVPBC-RDHEI scheme. There are three types of users: content owner, data hider and receiver. First, the content owner preprocesses the pixel values of the original image, then encrypts the image in combination with bitwise XOR operation and block permutation, and embeds the location map generated by preprocessing the pixel values into the encrypted image. On the data hider side, the data hider first classifies all blocks of the encrypted image, then encrypts the additional data using the data hiding key

to enhance security and embeds the encrypted additional data into the received encrypted image. Finally, the receiver can extract additional data or recover the original image by using different keys.

2.1 Image Encryption

In our scheme, encryption of the image includes three steps: pixel value preprocessing, block-level stream encryption and block permutation, and location map embedding.

Pixel Value Preprocessing. For the original image I of size $M \times N$, the pixel value $x(i,j)$ of each pixel is preprocessed to a value less than or equal to 31, in which $1 \le i \le M$, $1 \le j \le N$.

Step 1: Make each pixel value $x(i,j)$ less than or equal to 127 by Eq. (1),

$$x(i,j) = \begin{cases} 255 - x(i,j), x(i,j) > 127 \\ x(i,j), otherwise \end{cases} \tag{1}$$

Step 2: Use Eq. (2) to make each pixel value $x(i,j)$ less than or equal to 63,

$$x(i,j) = \begin{cases} 127 - x(i,j), x(i,j) > 63 \\ x(i,j), otherwise \end{cases} \tag{2}$$

Step 3: Obtain the final pixel value for each pixel by Eq. (3).

$$x(i,j) = \begin{cases} 63 - x(i,j), x(i,j) > 31 \\ x(i,j), otherwise \end{cases} \tag{3}$$

Note that to correctly determine whether the pixels in the original image have changed in three steps, three location maps sized $M \times N$ are required for marking, with '1' indicating no change and '0' indicating a change.

Block-Level Stream Encryption and Block Permutation. After preprocessing the pixel values of the original image I, the image is encrypted using block-level stream encryption and block permutation. First, the $M \times N$ sized image I is divided into non-overlapping blocks sized 2×2 given by

$$I = \left\{ P_k(i,j) \middle| 1 \le i \le \lfloor \frac{M}{2} \rfloor, 1 \le j \le \lfloor \frac{N}{2} \rfloor, 1 \le k \le 4 \right\} \tag{4}$$

where $P_k(i,j)$ denote the k-th pixel of the block located at (i,j).

Next, a $\lfloor \frac{M}{2} \rfloor \times \lfloor \frac{N}{2} \rfloor$ sized pseudo-random matrix H is generated using the image encryption key K_e. Then, each block is stream ciphered by

$$P_k^{en}(i,j) = P_k(i,j) \oplus h(i,j), 1 \le k \le 4 \tag{5}$$

where $h(i,j)$ denote the element located at (i,j) in the pseudo-random matrix H and $P_k^{en}(i,j)$ denote the k-th encrypted pixel of the block located at (i,j) and $\oplus$ denotes the bitwise exclusive-or operation.

After the stream encryption, the cover image is divided into 2×2 sized non-overlapping blocks once again. Then, all blocks are permutated using the same encryption key K_e to generate the encrypted image I_e.

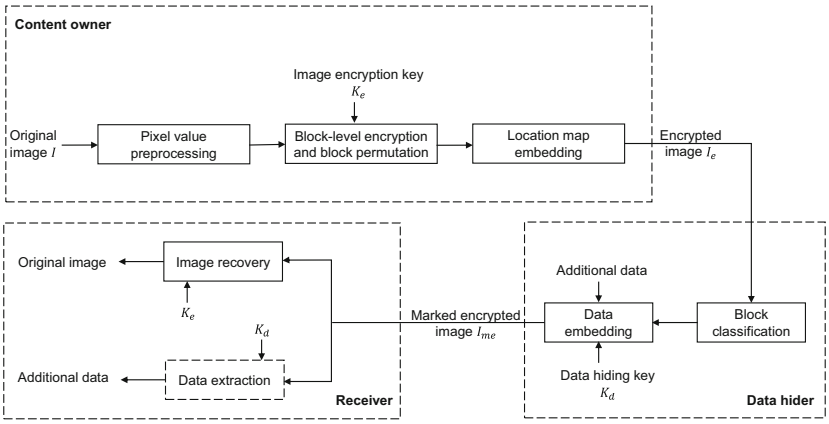

Fig. 1. The framework of the proposed scheme.

Location Map Embedding. Embed the three location maps into the encrypted image I_e as three bit planes, as shown in Fig. 2. Specifically, all pixels of the encrypted image I_e are converted into the 8-bit binary sequence, and 8-bits of all encrypted pixels are extracted sequentially to generate 8 encrypted bit planes. Then, the last three LSB bit planes are moved to the front as the first three MSB bit planes of the encrypted image, while keeping the position of the two bit planes in the middle unchanged. Next, the three location maps are embedded in the encrypted image as the last three LSB bit planes in turn. Finally, the 8-bit binary sequence of all pixels is converted to decimal system to regain the encrypted image I_e.

2.2 Block Classification

After receiving the encrypted image I_e, the embedding room for data hiding is vacated by the data hider. First, all pixels of the encrypted image are converted into the 8-bit binary system, and 8-bits of all encrypted pixels are extracted sequentially to generate 8 encrypted bit planes. Then, the last three LSB bit planes are extracted to obtain the previously embedded three location maps. Next, the first three MSB bit planes are extracted and recovered to the original LSB position of the encrypted image. Finally, the values of the first three MSB bit planes of the encrypted image are set to '0' so that all pixel values of the encrypted image are less than or equal to 31. Note that to ensure reversibility, the three location maps need to be compressed and saved as auxiliary information. This paper uses the bit-plane rearrangement and the bit-stream compression scheme [16] to compress the location map.

Next, the encrypted image I_e is divided into 2×2 sized non-overlapping pixel blocks. The structure of pixel block is shown in Fig. 3 (a). Then, for each block, all pixels are converted into the 8-bit binary system. For one block, 8-bits of all encrypted pixels are extracted sequentially to generate 8 encrypted bit planes

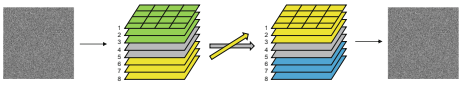

Fig. 2. Location map embedding process.

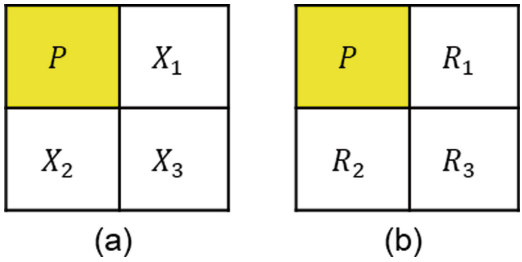

Fig. 3. The structure of a block.

E_λ ($\lambda = 1, 2, \cdots, 8$), and the first MSB encrypted bit plane is denoted as E_1. As shown in Fig. 3 (a), each pixel block consists of four pixels P, X_1, X_2 and X_3. The pixel P marked by yellow is selected as the reference pixel. First, three values R_1, R_2 and R_3 are obtained by

$$R_i = P \oplus X_i, i = 1, 2, 3 \tag{6}$$

Then, the three pixels X_1, X_2 and X_3 in the pixel block are replaced with R_1, R_2 and R_3 respectively. The reconstructed pixel block is shown in Fig. 3 (b). With R_1, R_2 and R_3, the block type-label T for each block can be calculated as

$$T = 8 - lenbin\left(\max\left(R_1, R_2, R_3\right)\right) \tag{7}$$

where $lenbin\left(d\right)$ returns the number of bits in the binary sequence of the decimal number d. For example, we have $lenbin\left(\max\left(30, 16, 9\right)\right) = 5$ since $\max\left(30, 16, 9\right) = 30 = \left(11110\right)_2$. The block type-label T records the number of consecutive uniformly valued bit-planes, i.e., totally '0' or '1', in the block counted from top-down. Therefore, according to how many bit planes are all '0' or all '1', all blocks can be scanned and classified into six different types as listed in Table 1. Note that the value of the block type-label T ranges from '3' to '8'.

Finally, Huffman coding is applied to generate corresponding indicators for six block types. Six Huffman codes are predefined to represent the six labels, namely $\{0, 10, 110, 1110, 11110, 11111\}$. Sort the six block type-labels by the number of blocks and use the shorter code to represent the label with the larger number of blocks. Taking the Lena image as an example, the distribution of each block type and its corresponding Huffman coding are shown in Table 2.

After the Huffman coding is determined, all block indicators are linked into a sequence of block type indicators and its total length LH can be calculated by

$$LH = \sum_{t=3}^{8} \left(n_t \times l_t\right), t = 3, 4, \cdots, 8 \tag{8}$$

where n_t is the number of pixel blocks with block type-label $T = t$, and l_t is the length of the corresponding Huffman code.

In the proposed scheme, for each encrypted block, the first three bits of the reference pixel P in the upper-left corner can be used for data embedding, while the last five bits remain unchanged to ensure reversibility. And according to the type-label T of each encrypted block, the first T bits of the other three pixels R_1, R_2 and R_3 are vacated for data embedding, while the last $(8 - T)$ bits remain unchanged. In the above way, each pixel block labeled as T can vacate $(3 \times T + 3)$ bits room. Figure 4 shows an example of six types of pixel blocks.

Table 1. Six block types and corresponding descriptions.

Block types	Descriptions
Type 3	E_1 to E_3 are all '0'
Type 4	E_1 to E_4 are all '0' or all '1'
Type 5	E_1 to E_5 are all '0' or all '1'
Type 6	E_1 to E_6 are all '0' or all '1'
Type 7	E_1 to E_7 are all '0' or all '1'
Type 8	E_1 to E_8 are all '0' or all '1'

Table 2. Lena's block type distribution and Huffman coding.

Block types	Amount	Indicator
Type 3	21125	0
Type 4	19860	10
Type 5	16192	110
Type 6	6685	1110
Type 7	1349	11110
Type 8	325	11111

2.3 Data Embedding

In this phase, two sets of data, auxiliary information and additional data, are embedded into the encrypted image I_e after block classification to obtain the marked encrypted image I_{me}. Detailed descriptions are as follows.

To reversibly recover the image, auxiliary information includes the total length of the block type indicator sequence, the block type indicator sequence, the length of the compressed location map, the compressed location map, and a 20-bits Huffman coding list. The Huffman coding list, the total length of the block type indicator sequence, the block type indicator sequence, the length of the three compressed location maps, and their corresponding compressed location maps are linked in turn to form complete auxiliary information. To ensure that the Huffman coding list and block type indicator sequence can be completely extracted in later phases, auxiliary information is first embedded into the first three bits of the upper-left pixel of each block. Then, the data hider uses the data hiding key K_d to encrypt the additional data and embeds the encrypted additional data into the remaining vacated room.

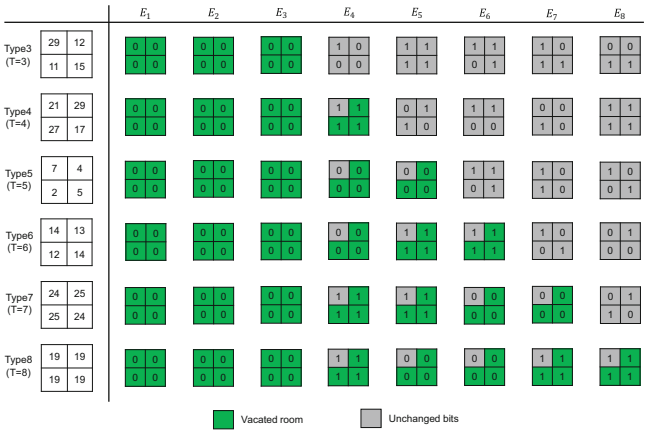

Fig. 4. Examples of block classification.

2.4 Data Extraction and Image Recovery

For the proposed scheme, data extraction and image recovery can be performed separately. The permission for data extraction and image recovery depends on the number of keys for the receiver. There are three cases:

Case 1: The receiver only has the image encryption key K_e, the original image can be recovered losslessly. The process is as follows.

1. Divide the marked encrypted image I_{me} into 2×2 sized non-overlapping blocks and extract the first three bits of the upper-left reference pixel in each block. Next, the Huffman coding list and the block type indicator sequence are extracted from the extracted data to determine the block type-label T for each block.
2. Extract three compressed location maps from the extracted data, and then decompress them to obtain the original three location maps.
3. For each pixel block, firstly, fill the first three bits of the upper-left reference pixel in the block with '0' to obtain P. Then, according to the block type-label T of each block, fill the first T bits of the other three pixels in the block with '0' to obtain R_1, R_2 and R_3. Next, X_1, X_2 and X_3 are recovered as

$$X_i = P \oplus R_i, i = 1, 2, 3 \tag{9}$$

4. After the encrypted pixel values of all blocks are recovered, the original image with the pixel values preprocessed is obtained by decryption using the key K_e.
5. Finally, apply three location maps in reverse order to recover each pixel value to obtain the original image I.

Case 2: The receiver only has the data hiding key K_d. The receiver first determines the block type-label T for each block as described in the previous

case. Then, the required amount of auxiliary information is accumulated through the guidance of the block type-label. After skipping the auxiliary information in the embeddable space, the receiver can extract the encrypted additional data stream and decrypt it to obtain the embedded additional data.

Case 3: The receiver has both the data hiding key K_d and the image encryption key K_e. The receiver can extract the additional data correctly and recover the original image perfectly.

Fig. 5. Six test images. (a) Lena; (b) Airplane; (c) Baboon; (d) Barbara; (e) Peppers; (f) Lake.

3 Experimental Results and Analysis

Experiments are conducted to evaluate the performance of the proposed scheme. First, in Sect. 3.1, the embedding rate of the proposed scheme is calculated and the reversibility is analyzed. The security of the proposed scheme is then analyzed in Sect. 3.2. In Sect. 3.3, three image datasets including BOSSBase [26], BOWS-2 [27] and UCID [28] are used to illustrate the universality of the proposed scheme. Finally, the proposed scheme is compared with some state-of-the-art RDHEI schemes in Sect. 3.4. Six commonly used 512×512 sized grayscale images: Lena, Airplane, Baboon, Barbara, Peppers, and Lake are selected to perform the experiments, as shown in Fig. 5. The embedding rate (ER) presented by bpp (bits per pixel) is the key indicator. In addition, PSNR (Peak signal-to-noise ratio) and SSIM (structural similarity) are also used to verify reversibility.

3.1 Embedding Rate and Reversibility

The embedding rate of the proposed scheme on six test images is calculated. Table 3 presents the distribution of pixel blocks of different block types, the total length of auxiliary information and the ER on six test images, where NT_k denote the number of pixel blocks labeled with block type-label $T = k\,(k = 3, 4, \cdots, 8)$, and LAU denote the total length of auxiliary information. The ER of the $M \times N$ sized image is calculated as

$$ER = \frac{1}{M \times N} \times \left(\sum_{k=3}^{8} (3 \times k + 3) \times NT_k - LAU \right). \tag{10}$$

As shown in Table 3, for the image 'Baboon', the LAU is 590314 bits and the ER is 1.0941 bpp. For the image 'Airplane', the LAU is 364376 bits and the ER is

2.7356 bpp. 'Airplane' is known to be much smoother than 'Baboon'. Therefore, it can be concluded that the ER generally depends on image smoothness.

Table 3. The distribution of pixel blocks of different block types on six test images.

Image	NT_3	NT_4	NT_5	NT_6	NT_7	NT_8	$LAU\,(bits)$	$ER\,(bpp)$
Lena	21125	19860	16192	6685	1349	325	348861	2.5769
Airplane	18600	16195	16170	9585	3865	1121	364376	2.7356
Baboon	42424	17100	5045	836	118	13	590314	1.0941
Barbara	28651	16999	12883	5684	1136	183	454114	2.0147
Peppers	20972	23744	15387	4658	682	93	361635	2.4408
Lake	29857	21792	9033	3492	1143	219	432787	1.9899

After theoretically calculating the ER of the image, the reversibility of the proposed scheme can be analyzed experimentally. Taking the Lena image as an example, Fig. 6 is the experimental results of each phase obtained by our scheme. The results show that the ER of the proposed scheme on the Lena image reaches 2.5769 bpp. Figure 6 (d1) and (a1) are identical, confirming that the proposed scheme can completely recover the original image. In addition, the quality of the recovered image is evaluated by PSNR and SSIM in the simulation results, where PSNR is close to $+\infty$ and SSIM is equal to 1. This indicates that the recovered image is the same as the original image. The theoretical analysis and simulation results verify the reversibility of the proposed scheme.

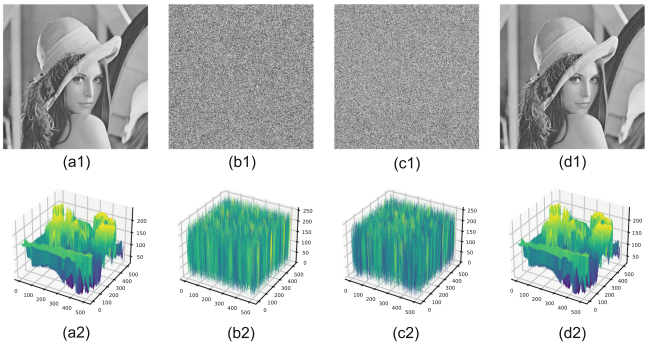

Fig. 6. Results in different phases for Lena. (a1) Original image; (b1) Encrypted image; (c1) Marked encrypted image with the ER = 2.5769 bpp; (d1) Recovered image (PSNR $\rightarrow +\infty$, SSIM = 1); (a2-d2) Pixel value distributions of the images a1-d1.

3.2 Security Analysis

To evaluate the security level of the proposed scheme, the statistical features of the images obtained at different stages are analyzed first. As can be seen from

Fig. 6 (a2-c2), the pixel value distribution of the original image has significant feature information, while the pixel value distribution of both the encrypted image and the marked encrypted image are uniform. It is difficult to find any content of the original image from the encrypted image or the marked encrypted image by using statistical analysis, which reinforces the security.

To further illustrate the security of this scheme, we analyze the key space of the encryption method. Our scheme is based on block-level stream encryption and block permutation to encrypt images. When dividing the $M \times N$ sized image into $s \times s$ non-overlapping blocks, the number of blocks is $n = \frac{M \times N}{s \times s}$. Therefore, the number of possible combinations of block-level stream encryption is $\theta_1 = 256^n$ and the number of possible combinations of block permutation is $\theta_2 = n!$. Overall, the entire key space of the encryption method is $\theta = \theta_1 \times \theta_2 = 256^n \times n!$. Assume that the 512×512 sized image is divided into 2×2 sized non-overlapping pixel blocks, the total number of image blocks is $n = 65536$. Therefore, the entire key space is $\theta = 256^{65536} \times 65536!$, which is a large number and it is almost impossible to decrypt the image correctly without the key. This can guarantee the security of the encrypted image.

Table 4. Experimental results on three entire image datasets.

Dataset	metric	Best	Worst	Average
BOSSBase	ER(bpp)	6.3852	0.4781	3.0207
	PSNR	$+\infty$	$+\infty$	$+\infty$
	SSIM	1	1	1
BOWS-2	ER(bpp)	5.8511	0.2652	2.8910
	PSNR	$+\infty$	$+\infty$	$+\infty$
	SSIM	1	1	1
UCID	ER(bpp)	4.7000	0.0824	2.3669
	PSNR	$+\infty$	$+\infty$	$+\infty$
	SSIM	1	1	1

3.3 Applicability to Image Datasets

To further verify the applicability of the proposed scheme to various feature images, three entire image datasets including BOSSBase [26], BOWS-2 [27] and UCID [28] are used to test the embedding rate. Among these datasets, BOSSBase and BOWS-2 both have 10000 grayscale images sized 512×512 while UCID has 1338 color images sized 384×512 or 384×512. For UCID, we first used the Image module in Python's Pillow library to convert 1338 color images into grayscale images. Table 4 lists the best, worst, and average cases on the three image datasets. It can be seen that the PNSRs of all images are close to $+\infty$ and the SSIMs of all images are equal to 1. This indicates that all images can be recovered losslessly. Consequently, the proposed scheme has good embedding performance in terms of ER.

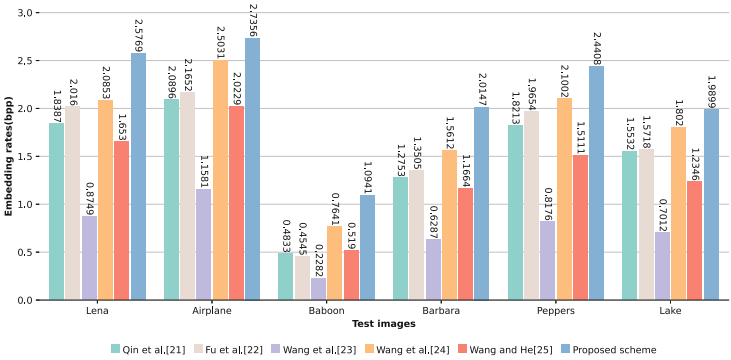

Fig. 7. Comparison of ER (bpp) on six test images.

3.4 Comparisons with Other State-of-the-art Schemes

In order to better illustrate the superiority of our scheme, the proposed scheme is compared with five state-of-the-art schemes [21–25]. All five schemes are VRAE-based scheme. As shown in Fig. 7, we first compare the ER on the six test images as shown in Fig. 5. It can be observed that the ER of the six test images obtained by our scheme are higher than these five schemes. In addition, our scheme can achieve good performance for rough images such as 'Baboon'.

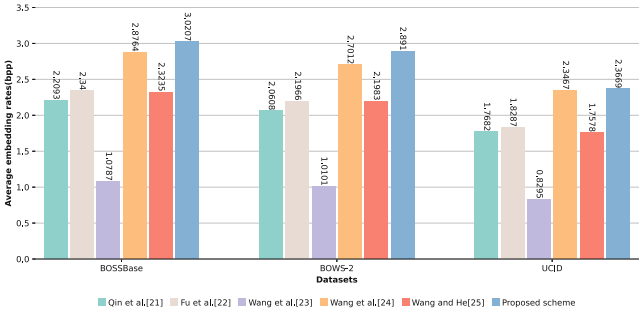

Fig. 8. Average ER (bpp) comparison on three image datasets.

Moreover, the experimental results on the three datasets of BOSSBase [26], BOWS-2 [27] and UCID [28] further confirm that our scheme outperforms the comparison schemes [21–25]. As shown in Fig. 8, the average ER of the proposed scheme on BOSSBase, BOWS-2 and UCID are 3.0207 bpp, 2.8910 bpp and 2.3669 bpp, respectively, which are higher than the average ER of the comparison schemes.

4 Conclusions

In this paper, an efficient reversible data hiding scheme for encrypted images based on pixel value preprocessing and block classification (PVPBC-RDHEI) is proposed. In the proposed scheme, the image is encrypted by preprocessing the pixel values of the original image and combining bitwise exclusive-or operation and block permutation. In addition, based on the number of consecutive consistent value bit planes counted from top-down in the pixel block, all blocks can be classified into six types and corresponding indicators can be generated for the six block types using Huffman coding. Therefore, each pixel block in the image can embed data, thus exploring a larger data embedding space. The experimental results and analysis show that the embedding rate of the proposed scheme is significantly better than the state-of-the-art schemes, and the security and reversibility are guaranteed. At the same time, the data extraction and image recovery are separable and error-free. In the future, we will investigate more efficient RDHEI schemes to further improve the embedding rate.

References

1. Jiang, R., Zhou, H., Zhang, W., Yu, N.: Reversible data hiding in encrypted three-dimensional mesh models. IEEE Trans. Multimedia **20**(1), 55–67 (2017)
2. Tian, J.: Reversible data embedding using a difference expansion. IEEE Trans. Circuits Syst. Video Technol. **13**(8), 890–896 (2003)
3. Al-Qershi, O.M., Khoo, B.E.: High capacity data hiding schemes for medical images based on difference expansion. J. Syst. Softw. **84**(1), 105–112 (2011)
4. He, W., Xiong, G., Weng, S., Cai, Z., Wang, Y.: Reversible data hiding using multi-pass pixel-value-ordering and pairwise prediction-error expansion. Inf. Sci. **467**, 784–799 (2018)
5. Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible data hiding. IEEE Trans. Circuits Syst. Video Technol. **16**(3), 354–362 (2006)
6. Li, X., Li, B., Yang, B., Zeng, T.: General framework to histogram-shifting-based reversible data hiding. IEEE Trans. Image Process. **22**(6), 2181–2191 (2013)
7. Wang, J., Ni, J., Zhang, X., Shi, Y.Q.: Rate and distortion optimization for reversible data hiding using multiple histogram shifting. IEEE Trans. Cybern. **47**(2), 315–326 (2016)
8. Zhang, W., Hu, X., Li, X., Yu, N.: Recursive histogram modification: establishing equivalency between reversible data hiding and lossless data compression. IEEE Trans. Image Process. **22**(7), 2775–2785 (2013)
9. Lin, C.C., Liu, X.L., Tai, W.L., Yuan, S.M.: A novel reversible data hiding scheme based on AMBTC compression technique. Multimedia Tools Appl. **74**, 3823–3842 (2015)
10. Lin, C.C., Liu, X.L., Yuan, S.M.: Reversible data hiding for VQ-compressed images based on search-order coding and state-codebook mapping. Inf. Sci. **293**, 314–326 (2015)
11. Ou, B., Li, X., Zhao, Y., Ni, R., Shi, Y.Q.: Pairwise prediction-error expansion for efficient reversible data hiding. IEEE Trans. Image Process. **22**(12), 5010–5021 (2013)

12. Kumar, R., Jung, K.H.: Enhanced pairwise IPVO-based reversible data hiding scheme using rhombus context. Inf. Sci. **536**, 101–119 (2020)
13. Li, X., Li, J., Li, B., Yang, B.: High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. Signal Process. **93**(1), 198–205 (2013)
14. Qu, X., Kim, H.J.: Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding. Signal Process. **111**, 249–260 (2015)
15. Ma, K., Zhang, W., Zhao, X., Yu, N., Li, F.: Reversible data hiding in encrypted images by reserving room before encryption. IEEE Trans. Inf. Forensics Secur. **8**(3), 553–562 (2013)
16. Chen, K., Chang, C.C.: High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. J. Vis. Commun. Image Represent. **58**, 334–344 (2019)
17. Puteaux, P., Puech, W.: A recursive reversible data hiding in encrypted images method with a very high payload. IEEE Trans. Multimedia **23**, 636–650 (2020)
18. Gao, G., Tong, S., Xia, Z., Shi, Y.: A universal reversible data hiding method in encrypted image based on MSB prediction and error embedding. IEEE Trans. Cloud Comput. **11**(2), 1692–1706 (2022)
19. Zhang, X.: Reversible data hiding in encrypted image. IEEE Signal Process. Lett. **18**(4), 255–258 (2011)
20. Qian, Z., Zhang, X.: Reversible data hiding in encrypted images with distributed source encoding. IEEE Trans. Circuits Syst. Video Technol. **26**(4), 636–646 (2015)
21. Qin, C., Qian, X., Hong, W., Zhang, X.: An efficient coding scheme for reversible data hiding in encrypted image with redundancy transfer. Inf. Sci. **487**, 176–192 (2019)
22. Fu, Y., Kong, P., Yao, H., Tang, Z., Qin, C.: Effective reversible data hiding in encrypted image with adaptive encoding strategy. Inf. Sci. **494**, 21–36 (2019)
23. Wang, Y., Cai, Z., He, W.: High capacity reversible data hiding in encrypted image based on intra-block lossless compression. IEEE Trans. Multimedia **23**, 1466–1473 (2021)
24. Wang, X., Chang, C.C., Lin, C.C.: Reversible data hiding in encrypted images with block-based adaptive MSB encoding. Inf. Sci. **567**, 375–394 (2021)
25. Wang, Y., He, W.: High capacity reversible data hiding in encrypted image based on adaptive MSB prediction. IEEE Trans. Multimedia **24**, 1288–1298 (2022)
26. Bas, P., Filler, T., Pevný, T.: Break our steganographic system: the ins and outs of organizing BOSS. In: Filler, T., Pevný, T., Craver, S., Ker, A. (eds.) IH 2011. LNCS, vol. 6958, pp. 59–70. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24178-9_5
27. Bas, P., Furon, T.: Image database of bows-2. Accessed 20 Jun 2017
28. Schaefer, G., Stich, M.: UCID: an uncompressed color image database. In: Storage and Retrieval Methods and Applications for Multimedia 2004. vol. 5307, pp. 472–480. SPIE (2003)