 CMC, vol.59, no.3, pp.925-941, 2019

Reversible Data Hiding Based on Pixel-Value-Ordering and Pixel Block Merging Strategy

Wengui Su[1, 2], Xiang Wang[3, *] and Yulong Shen[1]

Abstract: With the reversible data hiding method based on pixel-value-ordering, data are embedded through the modification of the maximum and minimum values of a block. A significant relationship exists between the embedding performance and the block size. Traditional pixel-value-ordering methods utilize pixel blocks with a fixed size to embed data; the smaller the pixel blocks, greater is the embedding capacity. However, it tends to result in the deterioration of the quality of the marked image. Herein, a novel reversible data hiding method is proposed by incorporating a block merging strategy into Li et al.'s pixel-value-ordering method, which realizes the dynamic control of block size by considering the image texture. First, the cover image is divided into non-overlapping 2×2 pixel blocks. Subsequently, according to their complexity, similarity and thresholds, these blocks are employed for data embedding through the pixel-value-ordering method directly or after being emerged into 2×4, 4×2, or 4×4 sized blocks. Hence, smaller blocks can be used in the smooth region to create a high embedding capacity and larger blocks in the texture region to maintain a high peak signal-to-noise ratio. Experimental results prove that the proposed method is superior to the other three advanced methods. It achieves a high embedding capacity while maintaining low distortion and improves the embedding performance of the pixel-value-ordering algorithm.

Keywords: Reversible data hiding, pixel-value-ordering, prediction error expansion, dynamic block partition.

1 Introduction

Reversible data hiding (RDH), also known as lossless data hiding, can recover both secret information and host image without loss after the extraction of embedded data [Caldelli, Filippini and Becarelli (2010)]. Currently, RDH has been applied in quality sensitive fields such as copyright protection, medical image processing, and military image protection.

Based on lossless compression, early RDH methods realized reversible information embedding [Celik, Sharma, Tekalp et al. (2005)]. The embedding capacity (EC) is highly limited, as only the compressed redundant space is utilized for embedding. To improve

[1] College of Computer Science and Technology, Xidian University, Xi'an, 710071, China.

[2] Guangxi Key Laboratory of Manufacturing System & Advanced Manufacturing Technology, College of Mechanical Engineering, Guangxi University, Nanning, 530004, China.

[3] College of Telecommunications Engineering, Xidian University, Xi'an, 710071, China.

[*] Corresponding Author: Xiang Wang. Email: wangxiang@xidian.edu.cn.

the embedding capacity and maintain a low distortion, many efficient RDH techniques have been introduced and developed over time. Among them, difference expansion (DE) [Tian (2003); Pei, Wang, Li et al. (2013)] and histogram shifting (HS) [Ni, Shi and Ansari (2006); Li, Li, Yang et al. (2013)] are two fruitful schemes.

The difference expansion scheme proposed by Tian [Tian (2003)] is a basic RDH technique. In recent years, the DE scheme has been extensively researched and developed in the aspects of integer transformation [Wang, Li, Yang et al. (2010)], reducing bitmap size [Hu, Lee and Li (2009)], prediction of error expansion (PEE) [Yao, Liu, Tang et al. (2018); He, Zhou, Cai et al. (2017)], and pixel-value-ordering (PVO) [Li, Li, Li et al. (2013); Ou, Li and Wang (2016)].

The HS algorithm proposed by Ni et al. [Ni, Shi and Ansari (2006)] is another important scheme for RDH. This scheme embeds secret information by shifting the intensity histogram bins between the peak point and zero point to obtain a higher quality embedded image. However, the maximum embedding capacity is limited by the number of pixels at the peak of the histogram. To improve the embedding capacity, many scholars have performed further improvements based on HS technology in recent years [Qin, Chang, Huang et al. (2013); Wang, Ye, Wang et al. (2018); Rad, Wong and Guo (2016); Du, Yin and Zhang (2018)]. For example, Du et al. [Du, Yin and Zhang (2018)] introduced the HS-based method in JPEG bitstream to optimize histogram modification. This novel method achieves lossless embedding with high payload and less file size expansion in an identical payload.

Thodi et al. [Thodi and Rodriguez (2007)] proposed the PEE method by combining DE and HS. The PEE algorithm introduces HS to compress the location map efficiently and utilizes the prediction error instead of the adjacent pixel difference for expansion embedding. The local correlation of the larger neighborhood of the image rather than the two adjacent pixels is considered in the prediction; therefore, the PEE method achieves a tradeoff between embedding capacity and fidelity. Because the prediction accuracy has an important impact on the performance of the PEE method, many related PEE methods have also been presented for efficient predictions, such as the median edge detector (MED) [Thodi and Rodriguez (2007)], gradient adjusted prediction (GAP) [Wu and Memon (2015)], adaptive embedding [Cao, An, Yao et al. (2018)], rhombus prediction [Sachnev, Kim, Nam et al. (2009)], and pixel value ordering [Li, Li, Li et al. (2013)]. In most of the improved PEE methods, both local complexity (LC) and prediction error (PE) can be served as metrics to determined block smoothness. The hypothesis that a direct proportion exists between LC and PE is typically exploited to reduce image distortion. However, in practice, some cases exist that do not conform to this hypothesis. Hence, Chen et al. [Chen, Ni, Hong et al. (2017)] devised a directionally enclosed prediction and expansion method to limit data embedding to pixels proportional to LC and PE. Experiments proved that this method yields high image fidelity while providing a considerable payload.

In recent years, the PVO-based RDH [Li, Li, Li et al. (2013)] has attracted increasing attention. The advantage of this method is that it can achieve better visual quality at low EC. In the PVO-based method [Li, Li, Li et al. (2013)], the image is divided into equal-sized pixel blocks; subsequently, the second largest/smallest pixel in the block is

used to predict the largest/smallest pixel that is modified to embed data and reversibility is achieved by maintaining the order of pixel values within each block before and after embedding. Because the pixels in the same block are often highly correlated, the PVO-based RDH method presents an excellent embedding performance. Meanwhile, the block selection technique is adopted to select smooth blocks for data embedding and a more concentrated histogram is obtained that improves the embedding performance. However, bin0, which is typically the second-highest bin, was not used in Li et al.'s method [Li, Li, Li et al. (2013)]. Peng et al. [Peng, Li and Yang (2014)] extended Li et al.'s work by introducing the relative position of the pixels into the prediction to construct a bilateral prediction error histogram. In his method, bin1 and bin0 were exploited for data embedding to better utilize image redundancy and achieve better embedding performance than that of Li et al. [Li, Li, Li et al. (2013)]. The work of Weng et al. [Weng, Pan, Deng et al. (2018)] is an improvement of Peng et al.'s work [Peng, Li and Yang (2014)]. They developed a novel pixel modification strategy. Three largest (or smallest) pixels in a smooth block were exploited for prediction; hence, a smooth block can carry at most $\log_2 3$ bits. Furthermore, two-layer embedding and n block partition ways were combined to achieve high embedding capacity and low distortion.

To our knowledge, the PVO embedding performance has a significant correlation with block size; therefore, many RDH methods are dedicated to introducing the block selection strategy in PVO embedding. Wang et al. [Wang, Ding and Pei (2015)] further improved Peng et al.'s work [Peng, Li and Yang (2014)] and proposed a PVO-based method with a dynamic block strategy. In accordance with the block complexity, the pixel blocks are divided into rough blocks, normal blocks, and flat blocks. Subsequently, the rough blocks are excluded from data embedding, whereas the flat blocks are further divided into four sub-blocks to increase the embedding capacity. A larger block size is selected in the texture area to ensure high peak signal-to-noise-ratio (PSNR), while a smaller block size is adopted in the smoothing area to achieve higher EC. This method can provide higher embedding capacity and lower distortion than the PVO-based RDH algorithm proposed in Li et al. [Li, Li, Li et al. (2013)] and [Peng, Li and Yang (2014)]. Further, Weng et al. [Weng, Liu, Pan et al. (2016)] proposed a reversible data hiding method based on flexible block-partition and the adaptive pixel modification strategy. The image is partitioned in accordance with the local complexity metric, and a high-correlation block is further divided into sub-blocks of size 1×3, and each block can be embedded with two data bits. A moderate-correlation block is divided into two sub-blocks of different sizes, each of which can embed up to four data bits. A low-correlation block can embed up to two bits. This method outperforms some of the state-of-the-art methods. Based on the dynamic block strategy proposed by Wang et al. [Wang, Ding and Pei (2015)]; He et al. [He, Cai, Zhou et al. (2017)] developed a multistage blocking strategy for n-threshold and n-level blocks. The image was first divided into non-overlapping root blocks. The root blocks were further divided into intermediate blocks or leaf blocks in accordance with the threshold. Further, different embedding schemes were selected in accordance with the block type. The algorithm guarantees less embedding distortion under the given EC, and outperforms Wang et al.'s work in computational complexity. Meikap et al. [Meikap and Jana (2018)] developed a PVO-based method with varying block sizes and multi-direction overlapping embedding.

In this method, a parameter α was utilized to maintain the order of pixel in a block to ensure that the method is suitable for different block sizes. Three directions: horizontal, vertical, and diagonal of each block were embedded for high embedding capacity. Experiments indicate that the method yields a better performance.

To achieve larger embedding capacity and high fidelity, many improved PVO-based methods aim to exploit the pixel correlation to obtain more suitable pixel block for data hiding. Inspired by this, a novel PVO-based algorithm based on the block merging strategy is proposed herein. In Li et al.'s work [Li, Li, Li et al. (2013)], the cover image was divided into blocks of equal size and the pixels in each block were sorted in accordance with the pixel values; subsequently, data embedding was realized by modifying the maximum and minimum values of each block. For a given EC, blocks of different sizes, i.e., $n_1 \times n_2$, $n_1, n_2 \in \{2,3,4,5\}$ were tested in the embedding process; subsequently, that with the best embedding performance was determined to be the final block size. A larger block produces a more concentrated histogram that improves the image PSNR but reduces the EC significantly. A smaller block helps to achieve higher EC, but typically results in the deterioration in image quality. The PVO-based embedding performance has a significant correlation with the block size and is highly limited by the uniform-blocking manner. To achieve high embedding capacity while preserving good visual quality, we propose a pixel block merging strategy based on the local complexity, similarity, and thresholds. We first divide the image into non-overlapping 2×2 blocks; subsequently, these pixel blocks are employed for PVO-based embedding or after being merged, which is decided according to a local complexity measurement. Experimental results indicate that the proposed method exploits the advantage of pixel correlation well and yields high embedding capacity without deteriorating the image quality.

The remainder of this paper is structured as follows. Section 2 provides an introduction of Li et al.'s PVO-based RDH algorithms. Section 3 introduces the proposed PVO-based embedding and extraction procedures in detail. Section 4 provides the experimental results and Section 5 concludes this paper.

2 Related works

Li et al.'s method [Li, Li, Li et al. (2013)] fully utilizes the similarity of pixels and can create the space for data embedding by modifying the maximum and minimum pixel blocks, thus enabling sufficient payloads to be embedded into images with low distortion. Take the maximum-modification-based embedding phases as an example. First, divide the cover image into non-overlapping and equal-sized blocks. For a given block including n pixels $X = \{x_1, x_2,...,x_n\}$, sort the pixel values $(x_1, x_2,...,x_n)$ in the ascending order, and obtain a sequence $(x_{\sigma(1)},...,x_{\sigma(n)})$, where $\sigma:(1,2,...,n)\rightarrow(1,2,...,n)$ denotes the unique one-to-one mapping such that $x_{\sigma(1)} \le x_{\sigma(2)} \le ... \le x_{\sigma(n)}, \sigma(i) < \sigma(j)$ if $x_{\sigma(i)} = x_{\sigma(j)}$ and $i<j$, subsequently, the second largest value $x_{\sigma(n-1)}$ can be used for predicting the maximum $x_{\sigma(n)}$, and the corresponding prediction error can be expressed as follows:

$$PE_{\max} = x_{\sigma(n)} - x_{\sigma(n-1)} \tag{1}$$

A histogram for PE_{max} can be obtained after calculating the prediction errors of all blocks.

The value of PE_{max} is always positive and bin1 (the bin of PE_{max} =1) is typically the peak value of the histogram, implying that this bin can be regarded as the interior region for data embedding. Furthermore, to ensure reversibility, bins larger than 1 are used as the external regions and are shifted to create the space for data embedding. Therefore, the maximum $x_{\sigma(n)}$ can be modified as follows:

$$\hat{x}_{\sigma(n)} = \begin{cases} x_{\sigma(n)}, & \text{if } PE_{\max} = 0 \\ x_{\sigma(n)} + b, & \text{if } PE_{\max} = 1 \\ x_{\sigma(n)} + 1, & \text{if } PE_{\max} > 1 \end{cases} \tag{2}$$

where $b \in \{0,1\}$ refers to the data to be embedded.

Let the marked value of X be represented as $(y_1,...,y_n)$, and sort its values in the ascending order and obtain $(y_{\sigma(1)},...,y_{\sigma(n)})$. Subsequently, all $i \neq \sigma(n)$ satisfies the condition that $y_i = x_i$ and $\hat{x}_{\sigma(n)} = y_{\sigma(n)}$. In addition, the order of pixel values is consistent in the embedding process, implying that the mapping of σ remains unchanged before and after data embedding.

When decoding, the prediction error can be described as follows:

$$\hat{PE}_{\max} = y_{\sigma(n)} - y_{\sigma(n-1)} \tag{3}$$

Only when $\hat{PE}_{\max} \in \{1,2\}$ can $\hat{PE}_{\max}$ include the embedded data, and the process of decoding is detailed as follows:

(1) If $\hat{PE}_{\max} \in \{1,2\}$, the embedded data will be $b = \hat{PE}_{\max} - 1$ and the original value will be $x_{\sigma(n)} = y_{\sigma(n)} - b$.

(2) If $\hat{PE}_{\max} > 2$, the prediction error will not include the embedded data and the original value will be $x_{\sigma(n)} = y_{\sigma(n)} - 1$.

(3) If $\hat{PE}_{\max} = 0$, the marked value will be the same as the original value and the original value will be $x_{\sigma(n)} = y_{\sigma(n)}$.

3 Proposed method

In the PVO-based RDH scheme [Li, Li, Li et al. (2013); Peng, Li and Yang (2014)], the image will be divided into pixel blocks of equal size, which subsequently are utilized for data embedding in a block-by-block manner. As shown in Fig. 1, the larger blocks lead to higher PSNR for low EC, whereas the smaller blocks yield a better embedding performance for high EC. Moreover, the block size is related to the maximum EC. The smaller blocks can provide higher EC. For PVO-based works, the tradeoff between capacity and fidelity is achieved by adjusting the block size.

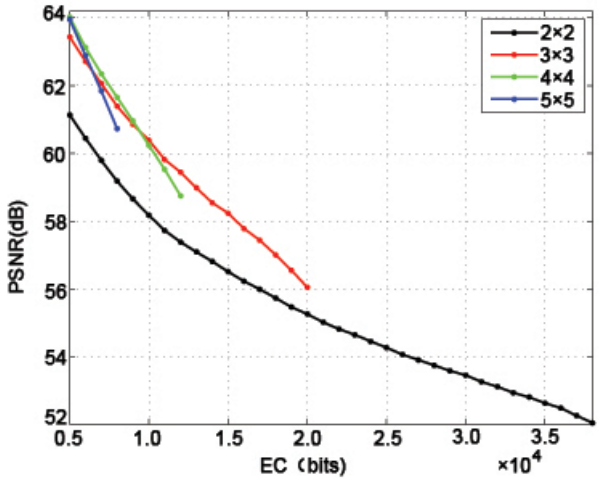

Figure 1: Performance of the scheme proposed by Peng et al. [Peng, Li and Yang (2014)] with different block sizes for the image Lena

With the above consideration, a novel PVO-based RDH using the block merging strategy is proposed to better exploit the pixel correlation within a block and create more embedded space with low distortion. In the smooth region where the pixels exhibit similar pixel values and good prediction accuracy, a smaller block size is adopted to improve the EC. Meanwhile, in the texture region where the pixels exhibit a weak correlation, a larger block size is employed to preserve good visual quality.

First, the host image is divided into non-overlapping 2×2 pixel blocks, and are classified into smooth blocks or normal blocks according to the local complexity measurement. The smooth 2×2 pixel blocks are directly embedded via the PVO-based method [Li, Li, Li et al. (2013)]. For normal pixel blocks, the similarity of the adjacent blocks is measured; subsequently, the similar blocks are merged into larger 2×4 or 4×2 blocks to perform PVO-based embedding. The unused merged blocks are further merged into 4×4 pixel blocks in accordance with the similarity and thresholds for PVO-based embedding. Hence, the proposed method achieves the dynamic control of the block sizes according to the image texture possible and achieves a better payload with lower distortions.

3.1 Classification of pixel blocks

3.1.1 Definition of complexity

In this study, we utilized the block complexity (denoted by NL) to differentiate a block I_i in the smooth region from the others. For the blocks with different texture complexities, different sizes of the embedding blocks should be determined. The image blocks can be classified by the NL and threshold T. Smooth blocks can be embedded via the PVO-based method directly, while ordinary blocks should be embedded via block merging before embedding.

Suppose the cover image I is a grayscale image of size $W \times H$; divide I into n $r \times c$ non-overlapping blocks, where $I=\{I_1, I_2,...,I_n\}$. Sort a block I_i by its pixel values in the

ascending order to obtain $I_i = (x_1, x_2, ..., x_k)$, where $x_1 \le x_2 \le ... \le x_k$. Thus, the block complexity NL_i of I_i can be defined as follows:

$$\begin{aligned} \bar{x} &= \frac{1}{k-2}\sum_{i=2}^{k-1} x_i \\ NL_i &= \frac{1}{k-2}\sum_{i=2}^{k-1} (x_i - \bar{x})^2 \end{aligned} \tag{4}$$

3.1.2 Smoothness classification

For a one-pixel block I_i (i=1, 2,…, n) of I, Eq. (4) can be used for calculating the local complexity NL_i (i=1,..., n). Further, whether a pixel block is smooth can be determined using the threshold T_j (j=1, 2, 3, $-1 \le T_j \le 255$). All pixel blocks can be divided into two sets, E and G, according to their local complexities NL_i and thresholds T_j. Set E contains all smooth blocks that are employed to embed data. The set G contains ordinary blocks that cannot be embedded into the data directly but must be further merged to maintain low distortion according to the similarity. The detailed steps of the block merging strategy are described in Section 3.2.2. As for the classification of the pixel blocks, two cases exist:

Case 1: if $NL_i \le T_j$, the block I_i is classified as a smooth block and added to set E. In this case, I_i locates in the smooth region where the blocks have similar pixel values and can be used for embedding. According to the PVO-based method in Li et al. [Li, Li, Li et al. (2013)], two bits of data (one bit for maximum and one bit for minimum) are directly embedded into block I_i.

Case 2: if $NL_i > T_j$, I_i is denoted as a normal block. In this case, I_i likely locates in the texture region where using a larger block size will exhibit evident advantages and higher PSNR. Therefore, the normal block I_i is added sequentially into set G and labeled with $G_i (i = 1, ..., l)$ for further processing in Section 3.2.

Reversibility is guaranteed by maintaining the complexity NL of each block invariant after embedding. In the embedding process of the PVO-based method, the pixels of a normal block remain unchanged; thus, the complexity remains unchanged as well. Meanwhile, for the smooth block, only the maximum x_k and minimum x_1 are modified to accomplish the PVO-based embedding. In such situations, the pixel values $(x_2, x_3 ..., x_{k-1})$ remain unchanged. Therefore, the complexity of a smooth block remains unchanged according to Eq. (4).

3.2 Pixel merging strategy

3.2.1 Definition of the similarity between blocks

After completing the block classification, adjacent blocks with high similarity in set G will be merged to create a larger block before performing data embedding to maintain higher PSNR and further improve the performance of the PVO-based method.

For two adjacent non-overlapping pixel blocks $G_i = (x_1, x_2, ..., x_k)$ and $G_{i+1} = (y_1, y_2, ..., y_k)$ $(i = 1,2,...,l-1)$ of size $r \times c$ in G, the definition of similarity (denoted by *Sim*) between blocks is:

$$Sim_i = \sum_{i=2}^{k-1} |x_i - y_i| \quad (i = 1,...,l-1) \tag{5}$$

The threshold S is used to measure the similarity of two blocks. If $Sim_i \le S$, G_i is similar to G_{i+1}, and these two blocks can be merged into a larger block with the size of $r \times 2c$ or $2r \times c$. Similar to the local complexity *NL*, the similarity *Sim* remains unchanged during embedding and extraction, thus ensuring reversibility.

3.2.2 Pixel merging strategy

Specifically, we use a threshold S to estimate the similarity of two adjacent blocks in the set G. According to the similarity *Sim* of two r×c blocks G_i and G_{i+1} (i=1,2,...,l-1) and the threshold S, two situations must be considered.

Case 1: If $Sim_i \le S$, G_i has high correlation with G_{i+1}, merge these two r×c blocks to form a new pixel block of size of r×2c or 2r×c. After this process, the better correlation among the pixels within the merged block will be maintained to ensure better visual effects in the texture regions after embedding. Subsequently, the block classification method depicted in Section 3.1.2 is applied again to estimate the smoothing level of the new merged block to determine whether the new pixel block should be embedded or further merged.

Case 2: If $Sim_i > S$, a significant difference appears between the pixel blocks G_i and G_{i+1}, and will lead to severe distortions and thus not suitable for embedding.

3.3 Embedding procedure

In this section, the embedding procedure of the presented method will be described in detail. Here are the specific steps.

Step 1: image partition

Divide the host grayscale image I into n non-overlapping pixel blocks of size 2×2, $I=\{I_1, I_2,..., I_n\}$. For each block I_i, sort its pixel values in the ascending order as $\{x_1, x_2, ..., x_4\}$, such that $x_1 \le x_2 \le ... \le x_4$.

Step 2: location map (LM) construction

For pixel values $(x_1, x_2, ..., x_4)$ in each block I_i, if $\max(x_1, x_2, ..., x_4) = 255$ or $\min(x_1, x_2, ..., x_n) = 0$, then this block may cause an overflow or underflow and should be excluded from embedding. In this case, we set the location map *LM(j)*=1, otherwise *LM(j)*=0. Therefore, *LM* is a binary sequence of length ξ. *LM* is consequently compressed losslessly using arithmetic coding and the length of *LM* is denoted as $l_{clm.}$

Step 3: block Classification and Embedding

Each block I_i should be classified according to the method in Section 3.1 with the given threshold T_1.

● If *LM(i)*=1, then the block is an exception that will cause overflow/underflow; skip this block and add it into set *C*.

● If *LM(i)*=0 and $NL_i \le T_1$, then the block is smooth. Subsequently, add I_i to set E with label E_i (i=1,..,p). Subsequently, utilize the PVO-based method in Li et al. [Li, Li, Li et al. (2013)] to embed two bits into each block.

● If *LM(i)*=0 and $NL_i > T_1$, then the block is normal. Subsequently, add the pixel block I_i in sequence to set $G(G_j(j = 1,...,l)$.

Step 4: the first merging and embedding process

With the given threshold *S* and the similarity Sim_i of adjacent blocks G_j and G_{j+1} in *G*, the first merging and embedding procedure are performed in this step.

1. If $Sim_i \le S$, subsequently merge G_i and G_{i+1} into a larger pixel block of size 2×4 or 4×2. Subsequently, re-estimate the block complexity of the merged block with the given threshold T_2 and the method described in Step 3. If smooth, the merged block is added to set *E* ($E_i(i=p+1,...,q)$) for embedding data. Further, two bits of data should be embedded into each of the smooth blocks. If normal, the merged block is grouped into a set *M*, where $M_k = G_j \cup G_{j+1}$ (k = 1,2,..., n ; j = 1,2,...,l-1) for the second merging process.

2. If $Sim_i > S$, neither block G_j nor G_{j+1} is used for merging and embedding. Subsequently, add these two blocks into set C.

Step 5: the second merging and embedding process

The second merging is performed in set *M* using the same steps in Step 4 and the threshold T_3. All the smooth blocks generated in this step are added into set *E* ($E_i(I=q+1,..., \xi_{end})$ and the remaining blocks in set *C*. For every smooth block, two bits of data are embedded. Repeat this step until all embeddable blocks are embedded and denote the index of the last data-carrying block as ξ_{end} .

The three threshold values T_1, T_2, T_3, and the similarity threshold *S* are related to specific images and the embedding capacity. Test all combinations; subsequently, the optimal threshold for different capacities can be determined.

Step 6: auxiliary information and the embedding procedure

Record the least-significant-bit (LSB) of the first $32 + 2\lceil log_2(16\xi)\rceil + l_{clm}$ pixels of the image to generate a binary sequence S_{LSB}, which will be embedded into the image as part of the watermark. Subsequently, these LSBs are replaced by the following auxiliary information and the compressed location map *LM:*

- Thresholds T_1 (8 bits), T_2 (8 bits), and T_3 (8 bits) for block complexity;
- Thresholds *S* (8 bits) for similarity;
- The end position ξ_{end} ($\lceil \log_2(16\xi) \rceil$ bits);
- The length of the compressed location map l_{dm}($\lceil \log_2(16\xi) \rceil$ bits);
- The location map (l_{dm} bits).

Finally, the sequence S_{LSB} is embedded into the remaining blocks $\{E_{\xi_{end}+1},...,E_{\xi}\}$ using the same method in Step 3 to generate the marked image. A flow chart of the embedding procedure is shown in Fig. 2.

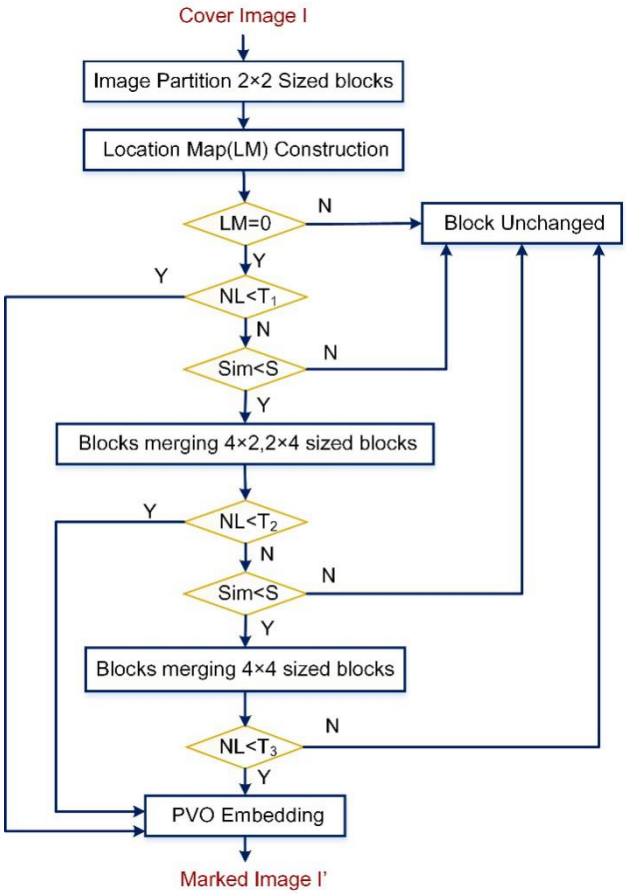

Figure 2: Embedding procedure

3.4 Extraction procedure

The extraction procedure is an inverse process of embedding. The detailed procedure consists of the following four steps.

Step 1: auxiliary information and location map extraction

Record the LSB of the first $32+2\lceil log_2(16\xi)\rceil$ pixels of the marked image to extract the auxiliary information including T_1, T_2, T_3, S, ξ_{end}, and l_{clm}. Subsequently, read the LSB of the latter l_{clm} pixels to extract the compressed LM. The LM is recovered through the decompression of the compressed map.

Step 2: image partition and the preprocessing

Using the same method as Step 1 in Section 3.3, partition the marked image I' into non-overlapping 2×2 blocks. Following the block classification and merging methods depicted in Section 3.3, Steps 3-5, and group all the blocks into sets E and C.

Step 3: sequence S_{LSB} extraction and image restoration

Extract the sequence S_{LSB} defined in Section 3.3, Step 6, from the block sequence $\{E_{\xi_{end}+1},...,E_{\xi}\}$ using the PVO-based decoding method depicted in Section 2 and perform image restoration.

Step 4: data extraction and image restoration

Replace the LSB of the first $36+2\lceil log_2(16\xi)\rceil+l_{clm}$ image pixels by the sequence S_{LSB} extracted in Step 3. Extract the hidden data and recover the host image from blocks $\{E_1,...,E_{\xi_{end}}\}$ using the method depicted in Step 3. For each block in set C, no data is hidden and its pixels remain unchanged. Thus, both the host image and secret data are fully recovered.

4 Experimental results

We first consider the determination of thresholds and the impact of the location map size to the embedding performance. The determination of thresholds T_1, T_2, T_3, and S depends on the given EC and the host image, and no direction relationship exists between these thresholds. The best combination of these thresholds is determined iteratively to be the smallest values such that minimal image distortion can be achieved.

To enable a lossless reconstruction of the host image at the decoder, the location map must record the positions of the expandable differences. The location map is compressed losslessly and subsequently embedded into the carrier along with the payload. Despite the compression, the location map still occupies part of the embedding capacity. Hence, the size of the location map will affect the embedding performance to some extent. To minimize the impact of location map on the embedding performance, adaptive block partition is adopted in our method. Smaller blocks are exploited for smooth areas and larger blocks for texture areas, thus improving the distribution of 0 and 1 in the location map and further reducing the size of the compressed location map. From Tab. 1, it is clear that the sizes of the LMs of different images have relatively small magnitudes compared with the EC; therefore, the LM has less influence on the embedding performance.

Table 1: The sizes of the LMs of different images

Image	Lena	Baboon	Airplane	Peppers	Boat
LM(bit)	96	352	96	136	208

The proposed method was implemented in MATLAB. For the performance evaluation, three state-of-the-art RDH methods, namely, those of Li et al. [Li, Li, Li et al. (2013)], Peng et al. [Peng, Li and Yang (2014)], and Sachnev et al. [Sachnev, Kim, Nam et al. (2009)], were re-implemented and compared to the proposed method. The embedding performance was evaluated by comparing with other RDH schemes from two perspectives: visual image quality and embedding capacity. Eight grayscale images including Airplane, Elaine,

Barbara, Baboon, Boat, Lake, Lena, and Peppers served as test images. Except Barbara, all images were downloaded from the USC-SIPI image database.

Li et al.'s work [Li, Li, Li et al. (2013)] is a high-fidelity PVO-based RDH method. Our method is a direct improvement of their work. Fig. 3 shows that the embedding performance of our method is significantly better than that of Li et al.'s work for all test images. Referring to Tab. 2 and Tab. 3, the performance of the proposed method improves by 0.83 dB on average for an EC of 10,000 bits and by 0.83 dB for an EC of 20,000 bits compared to the performance of Li et al.'s work. In particular, for the smooth image of Airplane, the proposed method provides higher maximum EC (0.198 bit per pixel) than that of Li et al. [Li, Li, Li et al. (2013)] (0.145 bits per pixel). Moreover, the proposed method obtains a greater gain in PSNR for the large capacity cases. In this case, because the EC is high, the 2×2 blocks that are applied in Li et al. [Li, Li, Li et al. (2013)] cannot fully utilize the pixel correlation. In addition, because there are no sufficient smooth blocks at the high EC, the method proposed in [Li, Li, Li et al. (2013)] employed necessary rough blocks for data embedding that will result in image deterioration. However, with the proposed method, a high EC can be achieved using 2×2 blocks in the smooth regions. If the EC increases and there are no sufficient smooth blocks, the small blocks can be merged into 2×4, 4×2, or 4×4 pixel blocks to obtain a smaller prediction error and maintain higher PSNR.

Peng et al.'s scheme [Peng, Li and Yang (2014)] is an extension of Li et al.'s work. In their work, bin0 was introduced into the embedding scheme and larger blocks were utilized to guarantee sufficient EC. Peng et al.'s scheme achieved a better performance than the PVO-based scheme [Li, Li, Li et al. (2013)]. According to Fig. 3, the proposed method achieves a substantial improvement in PSNR for every image regardless of the EC. Considering the results in Tab. 2 and Tab. 3, the PSNR improves by utilizing the proposed method when the capacities of the 10,000 bits and 20,000 bits are 0.29 dB and 0.36 dB, respectively. The maximum EC of the proposed method is almost the same as that of Peng et al.'s scheme. For a relatively large EC, the proposed method produces higher PSNR. The reason is that the capacity-dependent block strategies [Peng, Li and Yang (2014)] tend to use smaller blocks such as 2×2 or 3×3 when increasing the EC. Hence, it derives PNSR decreases, whereas a merging strategy is adopted in the proposed method to better exploit the correlations within a block and yields greater gain in the PSNR for the large capacity cases.

Sachnev et al.'s scheme [Sachnev, Kim, Nam et al. (2009)] is well known for its highly accurate prediction and has been verified better than many state-of-the-art works. As shown in Fig. 3, the proposed method outperforms Sachnev et al.'s in most cases, except when the EC approaches its maximum (e.g., 9300 bits for Baboon, 48000 bits for Airplane, and 29000 bits for Barbara). This is because the numbers of rough blocks have to be employed when aiming for maximum EC. However, the proposed scheme achieves superiority over Sachnev et al.'s scheme for a moderate EC. From Tab. 2 and Tab. 3, it is clear that the proposed scheme improves Sachnev et al.'s by 2.25 dB on average for an EC of 10000 bits and 1.74 dB for an EC of 20000 bits.

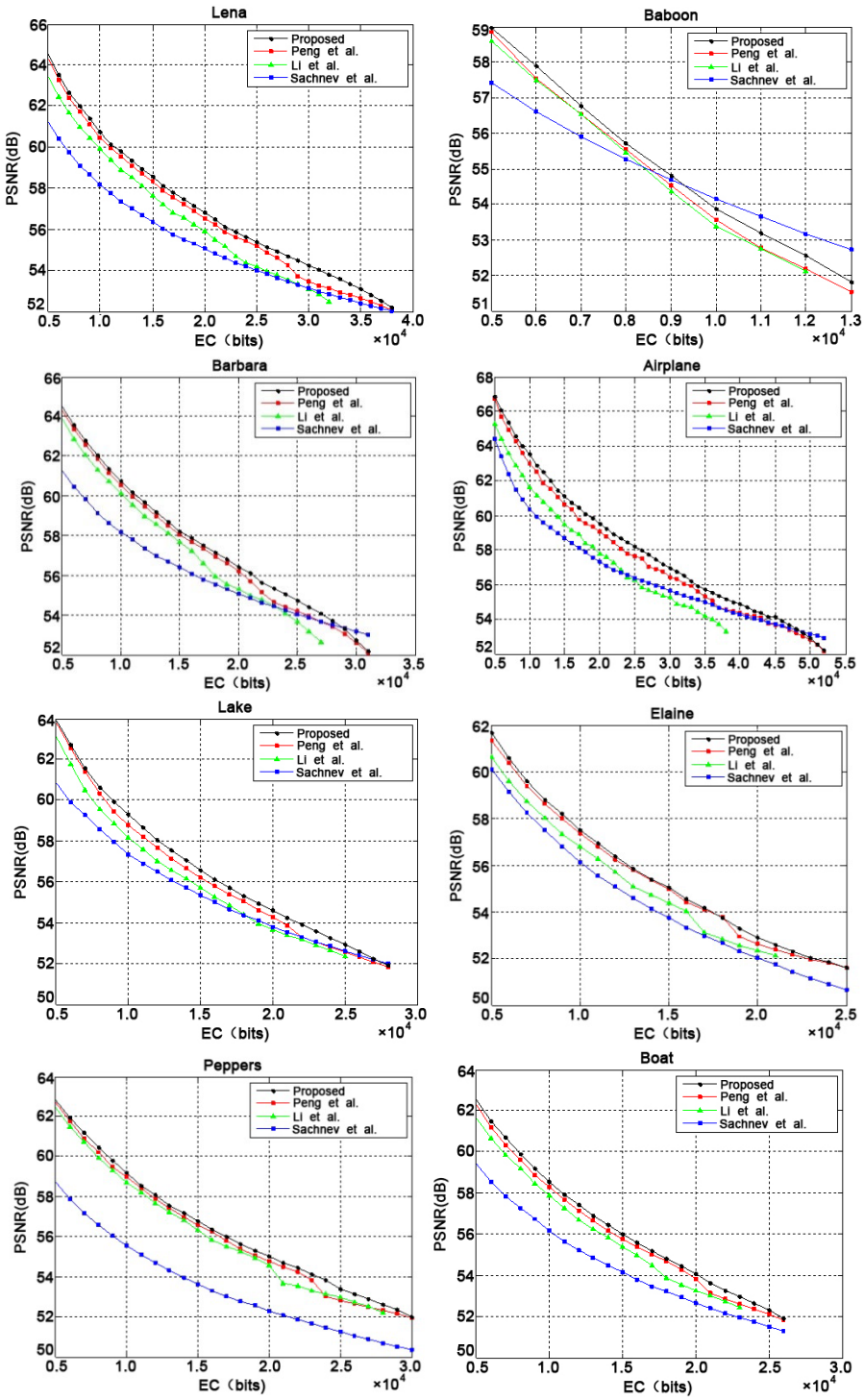

Figure 3: Performance comparison between the proposed method and the three methods

of Sachnev et al. [Sachnev, Kim, Nam et al. (2009)], Li et al. [Li, Li, Li et al. (2013)], and Peng et al. [Peng, Li and Yang (2014)]

Table 2: Comparison of PSNR (in dB) between the methods of Peng et al. [Peng, Li and Yang (2014)], Sachnev et al. [Sachnev, Kim, Nam et al. (2009)], and Li et al. [Li, Li, Li et al. (2013)], and the proposed approach for an EC of 10,000 bits

Image	Lena	Baboon	Barbara	Airplane	Peppers	Boat	Elaine	Lake	Average
Proposed	60.74	53.84	60.79	63.58	59.16	58.53	57.42	59.21	59.16
Peng, Li and Yang (2014)	60.47	53.55	60.55	62.96	58.98	58.27	57.36	58.87	58.87
Sachnev, Kim, Nam et al. (2009)	59.19	54.15	58.14	60.34	55.56	56.14	56.13	56.65	56.91
Li, Li, Li et al. (2013)	59.76	53.96	59.67	63.18	57.19	57.42	57.39	58.08	58.33

Table 3: Comparison of PSNR (in dB) between the methods of Peng et al. [Peng, Li and Yang (2014)], Sachnev et al. [Sachnev, Kim, Nam et al. (2009)], and Li et al. [Li, Li, Li et al. (2013)], and the proposed approach for an EC of 20,000 bits, for a capacity of 20,000 bits. The results for Baboon are eliminated herein because the proposed methods of Peng et al. and Li et al. cannot achieve such a payload

Image	Lena	Baboon	Barbara	Airplane	Peppers	Boat	Elaine	Lake	Average
Proposed	56.58	-	56.45	59.89	55.05	54.16	52.76	54.26	55.59
Peng, Li and Yang (2014)	56.54	-	56.20	59.07	54.77	53.84	52.61	53.60	55.23
Sachnev, Kim, Nam et al. (2009)	55.04	55.04	55.04	57.31	52.29	52.64	51.99	52.70	53.85
Li, Li, Li et al. (2013)	56.14	-	56.24	59.45	53.39	53.12	52.31	52.66	54.76

5 Conclusion

A reversible data hiding method that combined the PVO-based method with the block merging strategy was presented. Instead of uniform blocks, 2×2 blocks were adopted in smooth regions while 2×4, 4×2, or 4×4 blocks were employed in the rough regions in the embedding procedure. The proposed method increased the embedding capacity and reduced the distortions efficiently. Compared with three state-of-the-art reversible watermarking algorithms, the proposed method could well exploit the advantages of block correlation by adaptively selecting blocks of different sizes according to the image characteristics, hence demonstrating excellent performance. Based on this research, the best compromise between embedding performance and complexity can be further studied in the future.

Acknowledgments: This work was supported by the Key Basic Research Plan in Shaanxi Province (Grant No. 2017ZDXM-GY-014), and Guangxi Key Laboratory of Manufacturing System & Advanced Manufacturing Technology (Grant No. 17-259-05S007).

Reference

Caldelli, R.; Filippini, F.; Becarelli, R. (2010): Reversible watermarking techniques: an overview and a classification. *Eurasip Journal on Information Security*, vol. 2010, no. 134546, pp. 1-19.

Cao, F.; An, B. W.; Yao, H.; Tang, Z. J. (2018): Local complexity based adaptive embedding mechanism for reversible data hiding in digital images. *Multimedia Tools and Applications*, pp. 1-16.

Celik, M. U.; Sharma, G.; Tekalp, A. M.; Saber, E. (2005): Lossless generalized-LSB data embedding. *IEEE Transactions on Image Processing*, vol.14, no. 2, pp. 253-266.

Chen, H. S.; Ni, J. Q.; Hong, W.; Chen, T. S. (2017): High-fidelity reversible data hiding using directionally enclosed prediction. *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 574-578.

Du, Y.; Yin, Z. X.; Zhang, X. P. (2018): Improved Lossless Data Hiding for JPEG images based on histogram modification. *Computers, Materials & Continua*, vol. 55, no. 3, pp. 495-507.

He, W. G.; Cai, J.; Zhou, K.; Xiong, G. Q. (2017): Efficient PVO-based reversible data hiding using multistage blocking and prediction accuracy matrix. *Journal of Visual Communication and Image Representation*, vol. 46, pp. 58-69.

He, W, G.; Zhou, K.; Cai, J.; Wang, L.; Xiong, G, Q. (2017): Reversible data hiding using multi-pass pixel value ordering and prediction-error expansion. *Journal of Visual Communication and Image Representation*, vol. 49, pp. 35-360.

Hu, Y. J.; Lee, H. K.; Li, J. W. (2009): DE-based reversible data hiding with improved overflow location map. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 2, pp. 250-260.

Li, X. L.; Li, J.; Li, B.; Yang, B. (2013): High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. *Signal Processing*, vol. 93, no. 1, pp. 198-205.

Li, X. L.; Li, B.; Yang, B.; Zeng, T. Y. (2013): General framework to histogram-shifting-based reversible data hiding. *IEEE Transactions on Image Processing,* vol. 22, no. 6, pp. 2181-2191.

Meikap, S.; Jana, B. (2018): Directional PVO for reversible data hiding scheme with image interpolation. *Multimedia Tools and Applications*, vol. 77, no. 23, pp. 31281-31311.

Ni, Z. C.; Shi, Y. Q.; Ansari, N. (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.; Wang, J. W. (2016): High-fidelity reversible data hiding based on pixel-value-ordering and pairwise prediction-error expansion. *Journal of Visual Communication and Image Representation*, vol. 39, pp. 12-23.

Peng, F.; Li, X. L.; Yang, B. (2014): Improved PVO-based reversible data hiding. *Digital Signal Processing*, vol. 25, no. 2, pp. 255-265.

Pei, Q. Q.; Wang, X.; Li, Y.; Li, H. (2013): Adaptive reversible watermarking with improved embedding capacity. *Journal of Systems and Software*, vol. 86, no. 11, pp. 2841-2848.

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 System for Video Technology*, vol. 23, no. 7, pp. 1109-1118.

Rad, R. M.; Wong, K. S.; Guo, J. (2016): Reversible data hiding by adaptive group modification histogram of prediction errors. *Signal Processing*, vol. 125, pp. 315-328.

Sachnev, V.; Kim, H. J.; Nam, J.; Suresh, S; Shi, Y. Q. (2009): Reversible watermarking algorithm using sorting and prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989-999.

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.

Wang, X.; Ding, J.; Pei, Q. Q. (2015): A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition. *Information Sciences*, vol. 310, pp. 16-35.

Wang, X.; Li, X. L.; Yang, B.; Guo, Z. M. (2010): Efficient generalized integer transform for reversible watermarking. *IEEE Signal Processing Letters*, vol. 17, no. 6, pp. 567-570.

Wang, W. Q.; Ye, J. Y.; Wang, T. Q.; Wang, W, F. (2018): A high capacity reversible data hiding scheme based on right-left shift. *Signal Processing*, vol. 150, pp. 102-115.

Weng, S. W.; Liu, Y. J.; Pan, J. S.; Cai, N. (2016): Reversible data hiding based on flexible block-partition and adaptive block-modification strategy. *Journal of Visual Communication and Image Representation*, vol. 41, pp. 185-199.

Weng, S.; Pan, J. S.; Deng, J. H.; Zhou, Z. L. (2018): Pairwise IPVO-based reversible data hiding. *Multimedia Tools and Applications*, vol. 77, no. 11, pp. 13419-13444.

Wu, X.; Memon, N. (2015): Context-based, adaptive, lossless image coding. *IEEE Transactions on Communications*, vol. 45, no. 4, pp. 437-444.

Yao, H.; Liu, X.; Tang, Z.; Hu, Y.; Qin, C. (2018)：An improved image camouflage technique using color difference channel transformation and optimal prediction-error expansion. *IEEE Transactions on Signal Processing*, vol. 6, pp. 40569-40584.