Signal Processing 93 (2013) 198–205

Contents lists available at SciVerse ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion

Xiaolong Li [a,b], Jian Li [a,d], Bin Li [c], Bin Yang [a,*]

[a] Institute of Computer Science and Technology, Peking University, Beijing 100871, China
[b] Beijing Key Laboratory of Internet Security Technology, Peking University, Beijing 100871, China
[c] College of Information Engineering, Shenzhen University, Shenzhen 518060, China
[d] College of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China

ARTICLE INFO

Article history:
Received 25 January 2012
Received in revised form
29 June 2012
Accepted 17 July 2012
Available online 26 July 2012

Keywords:
Reversible data hiding
Pixel-value-ordering
Prediction-error expansion

ABSTRACT

This paper presents a high-fidelity reversible data hiding scheme for digital images based on a new prediction strategy called pixel-value-ordering (PVO) and the well-known prediction-error expansion (PEE) technique. Specifically, a host image is first divided into non-overlapped equal-sized blocks. Then the maximum and minimum values of each block are predicted by other pixels of the block according to their pixel value orders. With such a PVO-based predictor, data embedding is implemented via PEE. The incorporation of PVO into PEE has an advantage in reducing the number of shifted pixels, and thus it can alleviate the degradation in image quality. Consequently, the proposed method can embed adequate data into a host image with rather limited distortion. The PSNR of a marked image versus its original one is guaranteed to be above 51.14 dB. In addition, a solution is provided to further improve the embedding performance by priorly using the flat blocks to embed data while leaving the rough ones unchanged. We demonstrate the proposed method outperforms some advanced prior arts with abundant experimental results.

1. Introduction

Reversible data hiding aims to embed secret data into a host image by slightly modifying its pixels, and, more importantly, the original image as well as the embedded data should be completely restored from the marked image [1]. Reversible data hiding is a special type of information hiding, and the reversibility is quite desirable in some applications such as medical and military image processing.

Many reversible data hiding algorithms have been proposed so far. Among them, Tian's difference expansion (DE)-based method [2] has attracted much attention from researchers. The basic idea of DE is to employ the pixel difference instead of the pixel itself to embed data. Compared with the previously introduced compression-based reversible data hiding [3,4], Tian's method can provide a much higher embedding capacity (EC) while keeping the distortion low. Later on, the DE technique has been widely investigated and developed in many aspects [5–18].

There are three major extensions of DE. One tries to generalize DE into integer transform [5–7]. Another one tries to reduce the size of location map in DE [8–11]. The third one is called prediction-error expansion (PEE) where the difference value is replaced by the prediction-error in expansion embedding [12–14].

To our best knowledge, PEE usually leads to a good embedding performance since this approach has the

* Corresponding author. Tel.: +86 10 82529693; fax: +86 10 82529207.
E-mail addresses: lixiaolong@pku.edu.cn (X. Li), ljian20@gmail.com (J. Li), libin@szu.edu.cn (B. Li), yang_bin@pku.edu.cn (B. Yang).

0165-1684/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sigpro.2012.07.025

potential to well exploit the spatial redundancy in natural images. The first PEE-based method was proposed by Thodi and Rodriguez [12]. Afterwards, this method was improved by Hu et al. [13] by reducing the size of location map. Recently, the PEE technique was further improved by Li et al. [19] by incorporating in PEE two new strategies, adaptive embedding and pixel selection. Besides, it is worth mentioning that some double-layered PEE-based methods are also proposed [20–22]. Compared with [12–14] which utilize half-enclosing casual pixels in prediction, the double-layered embedding can employ full-enclosing pixels to predict the center one. In this way, a more accurate prediction is made and a better performance is achieved.

Besides DE, the histogram-based method proposed by Ni et al. [23] is another important work of reversible data hiding, in which the peak point of image histogram is utilized to embed data. In this method, each pixel value is modified at most by 1, and thus the visual quality of marked image is guaranteed. Notice that PEE can also be viewed as an extension of the histogram-based method, since PEE is implemented by modifying the prediction-error histogram.

In this paper, a high-fidelity reversible data hiding scheme is proposed based on incorporating a new prediction strategy in PEE. By the proposed method, at most half of image pixels are modified by 1 in value, and thus a high visual quality is guaranteed. The PSNR of a marked image versus its original one is at least 51.14 dB. We first propose a predictor based on pixel-value-ordering (PVO). Specifically, we divide the host image into non-overlapped equal-sized blocks. For each block, we use the second largest value to predict its maximum, and the second smallest value to predict its minimum. Then we implement data embedding in a block-wise manner by applying PEE to the prediction-error histograms. The combination of PVO-based predictor and PEE is advantageous in reducing the number of shifted pixels in expansion embedding. As a result, the proposed method can embed adequate data into a host image with rather limited distortion. Moreover, inspired by our previous work [19], we also provide a solution to select the flat blocks to embed data while ignoring the rough ones, which can further improve the embedding performance. Experimental results demonstrate that our method outperforms some previous PEE-based methods.

The rest of the paper is organized as follows. Section 2 introduces the proposed predictor and the new reversible data hiding scheme. The experimental results as well as the comparison with the prior arts are shown in Section 3. Section 4 concludes this paper.

2. Proposed method

In this section, we first introduce the PVO-based predictor, and then present the data embedding and extraction procedures.

2.1. PVO-based predictor and reversible data embedding

In PEE-based reversible data hiding, the prediction-error histogram is divided into two regions, one is called inner region and the other outer region. The prediction-errors in inner region are expanded to carry data, whereas those in outer region are shifted so that the inner and outer regions still separate from each other after data embedding. If there is only one or two bins in inner region, i.e., the maximum modification to pixel values is 1 in data embedding, it can be derived that the expected value of the modification (in l^2-norm) to host image is $0.5N_c+N_s$, where N_c and N_s are EC (in bits) and the number of shifted pixels, respectively [19]. Clearly, for a given EC N_c, the modification is less if N_s is less. Hence, the proportion of shifted pixels, i.e.,

$$P_{\text{shifted}} = \frac{\#\{\text{shifted pixels}\}}{\#\{\text{expanded or shifted pixels}\}} = \frac{N_s}{N_c+N_s} \tag{1}$$

is a benchmark of PEE-based reversible data hiding for evaluating the embedding performance, where "#" denotes the cardinal number of a set. The smaller the P_{shifted} is, the less modification to host image and the better performance is.

We now propose a new predictor for PEE. Its advantage over some existing predictors can be demonstrated using the benchmark P_{shifted}.

First, the host image is divided into non-overlapped equal-sized blocks. For a given block X containing n pixels, sort its value $(x_1,\ldots,x_n)$ in ascending order to obtain $(x_{\sigma(1)},\ldots,x_{\sigma(n)})$, where $\sigma:\{1,\ldots,n\}\rightarrow\{1,\ldots,n\}$ is the unique one-to-one mapping such that: $x_{\sigma(1)}\le\cdots\le x_{\sigma(n)}$, $\sigma(i)<\sigma(j)$ if $x_{\sigma(i)}=x_{\sigma(j)}$ and $i<j$. Then we use the second largest value, $x_{\sigma(n-1)}$, to predict the maximum $x_{\sigma(n)}$. The corresponding prediction-error is

$$PE_{\max} = x_{\sigma(n)} - x_{\sigma(n-1)}. \tag{2}$$

Take 2×2 sized blocks as an example, the histogram of $PE_{\max}$ for the image Lena is shown in Fig. 1.

Based on this predictor and PEE, the reversible data embedding/extraction can be described as follows (see Fig. 2 for an illustration). Since the bin 1 (i.e., the bin with $PE_{\max}=1$) is usually the histogram peak, similar to the previous works such as [23,21], we simply consider this bin as inner region and the bins larger than 1 as outer region. In this situation, to embed data via PEE, the

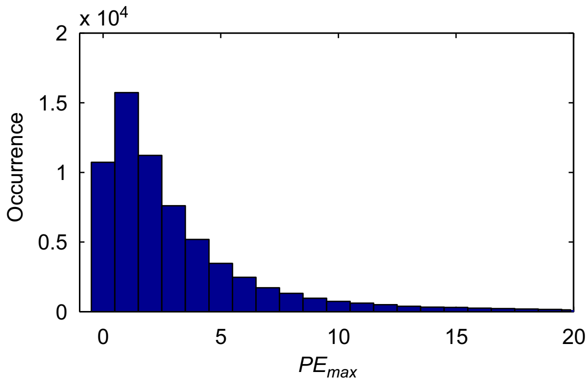

Fig. 1. Histogram of $PE_{\max}$, for the standard 512×512 sized gray-scale image Lena.

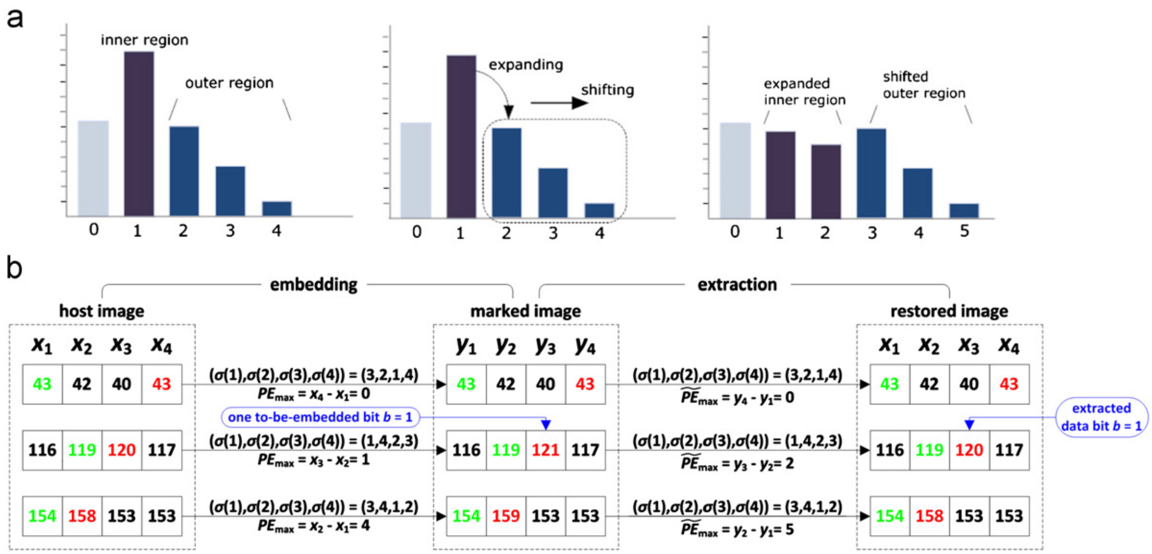

Fig. 2. Illustration of PVO-based PEE embedding (by modifying the maximum of a block). (a) The bin 0 is not involved in data embedding, the inner region is the bin 1, and the outer region is formed by other bins. (b) The maximum and the second largest value of each block are marked in red and green, respectively. In data embedding, the upper block is unchanged, the middle block is expanded to carry a data bit $b=1$, and the lower block is shifted. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Comparison of the value of P_{shifted} between the proposed PVO-based PEE embedding (with different block sizes) and two methods [13,20], for six standard 512×512 sized gray-scale images. Except Barbara, all the images are downloaded from USC-SIPI database.

Image	Hu et al. [13]	Sachnev et al. [20]	Proposed (2×2)	Proposed (3×3)	Proposed (4×4)	Proposed (5×5)
Lena	0.795	0.752	0.713	0.660	0.630	0.618
Baboon	0.933	0.916	0.895	0.860	0.836	0.819
Barbara	0.836	0.805	0.755	0.694	0.659	0.638
Airplane (F-16)	0.694	0.694	0.597	0.543	0.504	0.472
Peppers	0.874	0.835	0.757	0.696	0.667	0.652
Fishing boat	0.873	0.845	0.795	0.743	0.713	0.697

prediction-error $PE_{\max}$ is modified to

$$\widetilde{PE}_{\max}=\begin{cases}PE_{\max} & \text{if } PE_{\max}=0,\\ PE_{\max}+b & \text{if } PE_{\max}=1,\\ PE_{\max}+1 & \text{if } PE_{\max}>1,\end{cases} \tag{3}$$

where $b\in\{0,1\}$ is a data bit to be embedded. Accordingly, the maximum $x_{\sigma(n)}$ is modified to

$$\tilde{x}=x_{\sigma(n-1)}+\widetilde{PE}_{\max}=\begin{cases}x_{\sigma(n)} & \text{if } PE_{\max}=0,\\ x_{\sigma(n)}+b & \text{if } PE_{\max}=1,\\ x_{\sigma(n)}+1 & \text{if } PE_{\max}>1,\end{cases} \tag{4}$$

whereas other values $x_{\sigma(1)},\ldots,x_{\sigma(n-1)}$ keep unchanged. Hence, the marked value of X is $(y_1,\ldots,y_n)$, where $y_{\sigma(n)}=\tilde{x}$ and $y_i=x_i$ for every $i\neq\sigma(n)$.

In the above procedure, since the maximum $x_{\sigma(n)}$ is either unchanged or increased, the pixel value order (i.e., the mapping σ) remains unchanged. Therefore, for a marked block whose value is $(y_1,\ldots,y_n)$, by computing the prediction-error

$$\widetilde{PE}_{\max}=y_{\sigma(n)}-y_{\sigma(n-1)} \tag{5}$$

the data extraction and image restoration can be realized as follows:

- If $\widetilde{PE}_{\max}=0$, the block is unchanged in data embedding and its original value is just $(y_1,\ldots,y_n)$ itself.
- If $\widetilde{PE}_{\max}\in\{1,2\}$, the block is expanded to carry hidden data in data embedding. The embedded data bit is $b=\widetilde{PE}_{\max}-1$ and the original value is $(x_1,\ldots,x_n)$, where $x_{\sigma(n)}=y_{\sigma(n)}-b$ and $x_i=y_i$ for every $i\neq\sigma(n)$.
- If $\widetilde{PE}_{\max}>2$, the block is shifted in data embedding and its original value is $(x_1,\ldots,x_n)$, where $x_{\sigma(n)}=y_{\sigma(n)}-1$ and $x_i=y_i$ for every $i\neq\sigma(n)$.

Particularly, for the above PVO-based PEE embedding, the quantity P_{shifted} defined in (1) can be reformulated as

$$P_{\text{shifted}}=\frac{\#\{PE_{\max}>1\}}{\#\{PE_{\max}\geq1\}} \tag{6}$$

where $PE_{\max}$ is the prediction-error of host image defined in (2).

Referring to Table 1, it presents the value of P_{shifted} for the PVO-based PEE embedding (with different block sizes) and two typical existing PEE embeddings, i.e., the method [13] which is based on the context adaptive predictor median-edge-detector (MED [24]) and the double-layered embedding method [20]. Recall that in [20], a pixel is predicted by the average of its four neighbors. For this table, the results are directly derived from the prediction-error histograms, other factors influencing the performance (e.g., the location map, the sorting technique used in [20], etc.) are not taken into account. The comparison demonstrates that PVO-based PEE embedding is better than the prior arts in terms of benchmark P_{shifted}. Moreover, for PVO-based PEE embedding, better performance can be achieved if using larger sized blocks. The latter observation is reasonable since, by taking larger sized blocks, the spatial redundancy in natural images can be better exploited.

So far we have only introduced the data embedding scheme by modifying the maximum of a block. We can in fact consider modifying the minimum as well. Specifically, the second smallest value, $x_{\sigma(2)}$, can be used to predict the minimum $x_{\sigma(1)}$, and PEE embedding can be implemented by modifying the corresponding prediction-error $PE_{\min} = x_{\sigma(1)} - x_{\sigma(2)}$ (see Fig. 3 for an illustration). In this case, the prediction-error $PE_{\min}$ is modified to

$$\widetilde{PE}_{\min} = \begin{cases} PE_{\min} & \text{if } PE_{\min} = 0, \\ PE_{\min} - b & \text{if } PE_{\min} = -1, \\ PE_{\min} - 1 & \text{if } PE_{\min} < -1, \end{cases} \tag{7}$$

where $b \in \{0,1\}$ is a data bit to be embedded. The marked value of $x_{\sigma(1)}$ can be determined accordingly and the obvious details are omitted for sake of brevity.

By considering both the maximum and minimum, at most two bits can be embedded into a block at the same time, which may efficiently increase EC. For example, when the conditions $x_{\sigma(n)} - x_{\sigma(n-1)} = 1$ and $x_{\sigma(1)} - x_{\sigma(2)} = -1$ are both satisfied, one can embed two bits into the block X (the middle block in Fig. 2(b) is such an example). The detailed data embedding procedure including the construction of location map will be given in the next subsection.

Finally, we remark that the utilization of flat blocks is more favorable for reversible data hiding [19]. In this light, we will use priorly the flat blocks to embed data while leaving the rough ones unchanged. Here, the complexity of a block X is measured by the difference $x_{\sigma(n-1)} - x_{\sigma(2)}$, and a block is taken as a flat one if its complexity is less than a predefined threshold T. For example, for the three blocks in Fig. 2(b), their complexities are $x_1 - x_2 = 43 - 42 = 1$, $x_2 - x_4 = 119 - 117 = 2$ and $x_1 - x_4 = 154 - 153 = 1$, respectively. Notice that the quantity $x_{\sigma(n-1)} - x_{\sigma(2)}$ is invariant after data embedding, so the encoder and decoder can find exactly the same flat blocks. This guarantees the reversibility of PVO-based PEE embedding in the case of using flat blocks.

2.2. Data embedding procedure

The data embedding procedure is described as follows step by step.

*Step*1 (*Image partition*): Divide the host image into k non-overlapped blocks $\{X_1, \ldots, X_k\}$ such that each X_i contains n pixels. Sort the pixel values of X_i in ascending order to obtain $(x_1, \ldots, x_n)$. Notice that, unlike the previous subsection, the notation $(x_1, \ldots, x_n)$ is used here to denote the ordered pixel values, instead of the original values. Also, we omit the block-index i in definition of pixel values of X_i for clarity.

*Step*2 (*Location map construction*): The overflow/underflow location map LM is defined in this step. For a block X_i, if one of the following two cases occurs,

- $x_n - x_{n-1} \geq 1$ and $x_n = 255$
- $x_1 - x_2 \leq -1$ and $x_1 = 0$

it is an exceptional block that would cause overflow/underflow after being expanded or shifted in PEE. And in this case we set $LM(i) = 1$. Otherwise, we take $LM(i) = 0$. Consequently, LM is a binary sequence of length k. Then losslessly compress LM using arithmetic coding. Denote the length of the compressed location map as l_{clm}. Here, we remark that there is usually only a few exceptional blocks that may cause overflow/underflow, so only a few "1" in the sequence LM. Consequently, using lossless compression may significantly reduce the size of location map.

*Step*3 (*Data embedding*): Successively embed the data bits into the host image, i.e., for $i \in \{1, \ldots, k\}$,

- if $LM(i) = 1$, the overflow/underflow would occur and we do nothing with X_i.
- if $LM(i) = 0$ and $x_{n-1} - x_2 \geq T$, X_i is a rough block and we do nothing with X_i.
- if $LM(i) = 0$ and $x_{n-1} - x_2 < T$, there is no overflow/underflow and X_i is a flat block. In this case, the maximum and minimum of X_i will be shifted or expanded to carry data:
  - for the maximum x_n
    - if $x_n - x_{n-1} > 1$, x_n is shifted to $x_n + 1$.
    - if $x_n - x_{n-1} = 1$, x_n is expanded to $x_n + b$, where $b \in \{0,1\}$ is a data bit to be embedded.
    - if $x_n - x_{n-1} = 0$, x_n will not be modified.

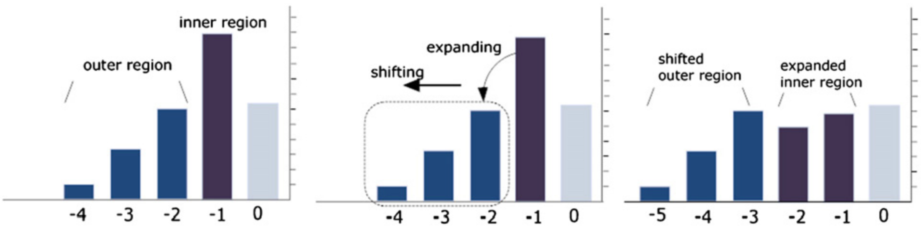

Fig. 3. Illustration of PVO-based PEE embedding (by modifying the minimum of a block).

- for the minimum x_1
  - if $x_1 - x_2 < -1$, x_1 is shifted to $x_1 - 1$.
  - if $x_1 - x_2 = -1$, x_1 is expanded to $x_1 - b'$, where $b' \in \{0,1\}$ is another data bit to be embedded.
  - if $x_1 - x_2 = 0$, x_1 will not be modified.

This step will stop if all data bits are embedded, and we denote k_{end} as the index of last data-carrying block. A critical feature here is that the pixel value order in each block remain unchanged after data embedding.

*Step*4 (*Auxiliary information and location map embedding*): Record the least significant bits (LSB) of first $16 + 2\lceil \log_2 N \rceil + l_{\text{clm}}$ image pixels to obtain a binary sequence S_{LSB}, where $N = kn$ is the total number of image pixels and $\lceil \cdot \rceil$ is the ceiling function. Then replace these LSB by the following auxiliary information and the compressed location map defined in *Step* 2

- block size parameters n_1 (4 bits) and n_2 (4 bits), where $n = n_1 \times n_2$,
- block complexity threshold T (8 bits),
- end position k_{end} ($\lceil \log_2 N \rceil$ bits),
- length of the compressed location map l_{clm} ($\lceil \log_2 N \rceil$ bits).

Finally, embed the sequence S_{LSB} into the remaining blocks $\{X_{k_{\text{end}}+1}, \ldots, X_k\}$ using the same method in *Step* 3 to obtain the marked image.

In the above procedure, apart from a few blocks that carry the auxiliary information and location map, only the maximum and minimum of a block will be modified by 1 in value whereas other pixels remain unchanged. So, approximately, the PSNR of a marked image versus its original one is at least

$$10 \log_{10} \frac{255^2 n}{2} \geq 10 \log_{10}(2 \cdot 255^2) = 51.14 \text{ dB} \tag{8}$$

when the block size $n \geq 4$. Therefore, by our method, the visual quality of marked image is well guaranteed.

Finally, we remark that in our method, to use priorly the flat blocks, the threshold T is taken as the smallest positive integer such that the secret data can be successfully embedded. This threshold can be determined iteratively.

2.3. Data extraction procedure

The corresponding data extraction procedure is detailed as follows.

*Step*1 (*Auxiliary information and location map extraction*): Read LSB of first $16 + 2\lceil \log_2 N \rceil$ pixels of marked image to get the auxiliary information including the values of n_1, n_2, T, k_{end} and l_{clm}. Then read LSB of next l_{clm} pixels to get the compressed location map. Determine the location map LM by decompressing the compressed location map.

*Step*2 (*Sequence S_{LSB} extraction and image restoration*) The same as data embedding, divide the marked image into k non-overlapped blocks $\{X_1, \ldots, X_k\}$ such that each X_i contains n pixels. In this step, for blocks $\{X_{k_{\text{end}}+1}, \ldots, X_k\}$, we will extract the sequence S_{LSB} defined in *Step* 4 of data embedding and meanwhile realize the restoration. For a block X_i ($i > k_{\text{end}}$) whose values in ascending order are $(y_1, \ldots, y_n)$:

- if $LM(i) = 0$ and $y_{n-1} - y_2 < T$
  - for the maximum y_n
    - if $y_n - y_{n-1} > 2$, there is no hidden data and original value of y_n is $y_n - 1$.
    - if $y_n - y_{n-1} \in \{1,2\}$, the hidden data is $b = y_n - y_{n-1} - 1$ and original value of y_n is $y_n - b$.
    - if $y_n - y_{n-1} = 0$, there is no hidden data and original value of y_n is itself.
  - for the minimum y_1
    - if $y_1 - y_2 < -2$, there is no hidden data and original value of y_1 is $y_1 + 1$.
    - if $y_1 - y_2 \in \{-1,-2\}$, the hidden data is $b' = y_2 - y_1 - 1$ and original value of y_1 is $y_1 + b'$.
    - if $y_1 - y_2 = 0$, there is no hidden data and original value of y_1 is itself.
- otherwise, there is no hidden data and original values of y_n and y_1 are themselves.

Moreover, in any case, for every $j \in \{2, \ldots, n-1\}$, the original value of y_j is itself since it is unchanged in data embedding. This step will stop if the sequence S_{LSB} is extracted.

*Step*3 (*Data extraction and image restoration*): Replace LSB of first $16 + 2\lceil \log_2 N \rceil + l_{\text{clm}}$ image pixels by the sequence S_{LSB} extracted in *Step*2. Then use the same method of *Step*2 to extract the hidden data from the blocks $\{X_1, \ldots, X_{k_{\text{end}}}\}$, and meanwhile to realize restoration for those blocks. Finally, the embedded data is extracted and the host image is recovered.

3. Experimental results

We first consider the impact of block size $n_1 \times n_2$ to the embedding performance. Referring to Fig. 4, it can be observed that generally, the performance on PSNR is better for a larger block size. However, larger sized blocks provide lower maximum EC. For example, we can only embed a maximum of 6000 bits into the image Lena when taking 5×5 sized blocks. In contrast, we can embed 32,000 bits when taking 2×2 sized blocks. In this light, to achieve the best performance while providing high EC,

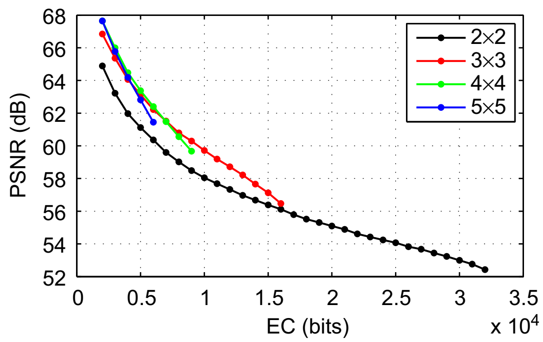

Fig. 4. Performance of the proposed method with different block sizes, for the image Lena.

we propose to take capacity-dependent blocks. Specifically, for a given EC, the embedding procedure is carried out for different block sizes, then the one which achieves the best performance is taken as the final block size. Here, the cases $n_1,n_2 \in \{2,3,4,5\}$ are considered and the embedding procedure is repeated for 16 times to determine the best block size. In consequence, the computational complexity is a little high whereas our gain is the best performance.

Fig. 5 shows the performance comparison of our method and four recently proposed PEE-based methods of Hu et al. [13], Sachnev et al. [20], Luo et al. [21], and Li et al. [19]. For our method, we vary EC from 5000 bits to its maximum with a step size of 1000 bits. The figure clearly illustrates that our method outperforms Hu et al.'s and Luo et al.'s, since our method has a higher PSNR for every image whatever EC is. Besides the effective predictor for PEE, our superiority is mainly dependent on the strategy of flat blocks selection.

For Sachnev et al.'s method [20], it provides also an embedding position selection strategy similar to ours, and this method is better than ours for some images in the case of high EC (e.g., for the image Lena and an EC larger than 28,000 bits, for the image Baboon and an EC larger

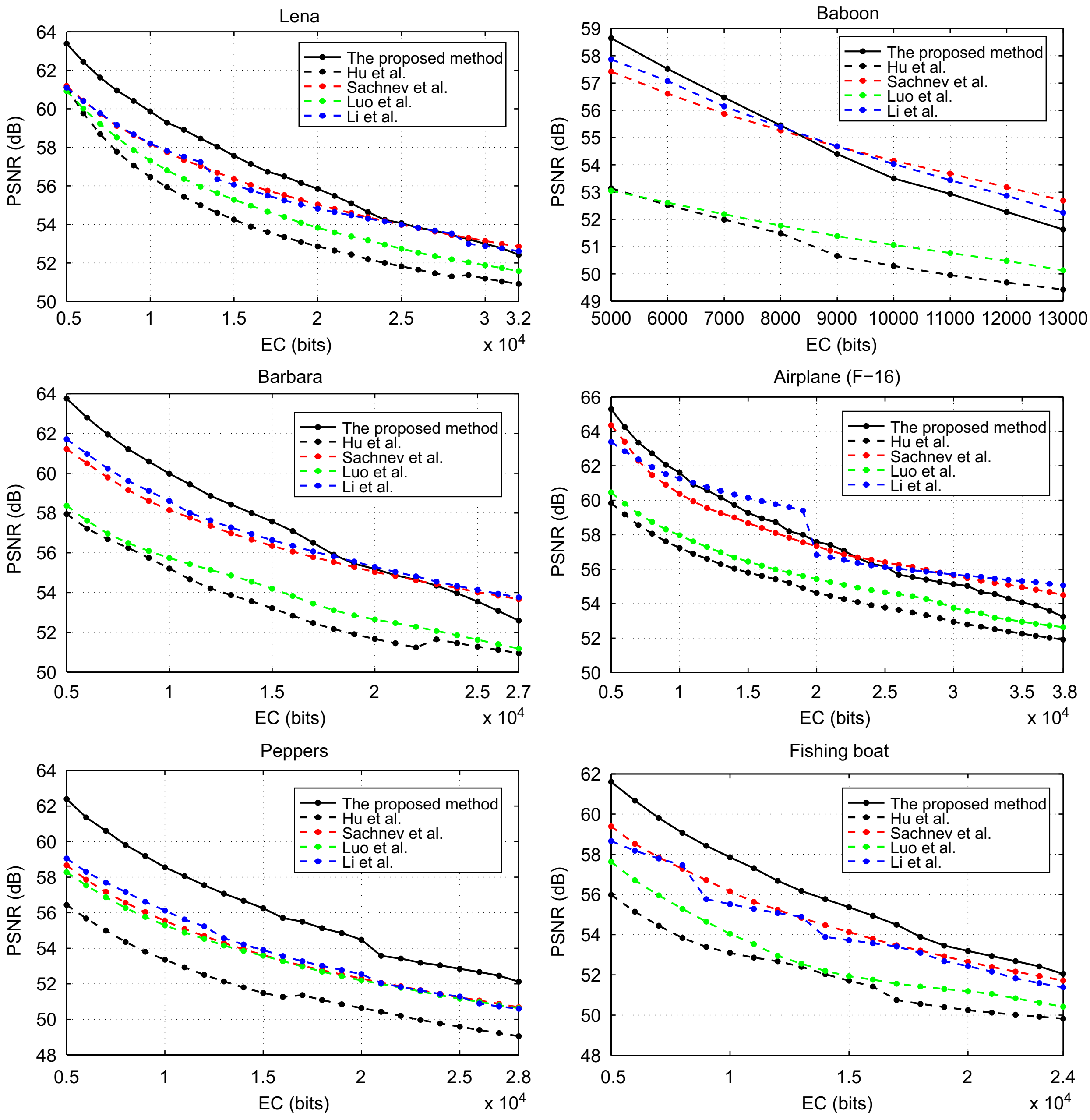

Fig. 5. Performance comparison between our method and four methods of Hu et al. [13], Sachnev et al. [20], Luo et al. [21], and Li et al. [19].

Table 2
Comparison of PSNR (in dB) between our method and four methods of Hu et al. [13], Sachnev et al. [20], Luo et al. [21], and Li et al. [19], for an EC of 10,000 bits.

Image	Hu et al. [13]	Sachnev et al. [20]	Luo et al. [21]	Li et al. [19]	Proposed
Lena	56.46	58.18	57.31	58.20	59.86
Baboon	50.29	54.15	51.06	54.03	53.50
Barbara	55.21	58.15	55.74	58.61	59.98
Airplane (F-16)	57.24	60.38	57.97	61.26	61.61
Peppers	53.36	55.55	55.29	56.12	58.55
Fishing boat	53.09	56.15	54.04	55.52	57.85
Average	**54.28**	**57.09**	**55.24**	**57.29**	**58.56**

than 8000 bits, etc., see Fig. 5). The reason is that in such cases, since EC is high, our method should take necessarily 2×2 sized blocks and cannot well exploit the advantage of pixel correlation. In addition, for those cases, the flat blocks are insufficient and our method should use necessarily rough blocks to embed data. This is also unfavorable to the embedding performance. However, for moderate capacities, our method is better. Referring to Table 2, it can be seen that our method improves Sachnev et al.'s by 1.47 dB in average for an EC of 10,000 bits.

For Li et al.'s method [19], it is also better than ours for some images in the case of high EC, since this method can also take advantage of flat blocks. However, similar to the case of Sachnev et al.'s method, our method is better for moderate capacities. Referring to Table 2, it can be seen that our method improves Li et al.'s by 1.27 dB in average for an EC of 10,000 bits.

In addition, we remark that, from Table 2, it can be observed that the improvement of our method over Hu et al.'s and Luo et al.'s is significant, say, larger than 3.3 dB in average for an EC of 10,000 bits.

It should be noticed that the methods [13,19–21] can provide a much higher maximum EC (about 1 bit per pixel or even larger) than the proposed one (e.g., only 0.122 bits per pixel, for the image Lena). This is a drawback of our method. However, this level of EC is sufficient for many practical applications. For example, Coatrieux et al. [25] pointed out that an EC of 3500 bits (about 0.014 bits per pixel for a 512×512 sized image) is enough for the application of reversible data hiding in medical image sharing. At the current stage, we only utilize the bin 1 to carry data and do not apply multi-pass embedding. Improving EC is beyond the scope of this paper, and we will investigate this issue in our future work.

4. Conclusion

In this paper we presented a new PEE-based reversible data hiding method. Our main contributions are as follows. First, based on ordering the pixel values in image block, an effective predictor is proposed for PEE. Second, the flat blocks are priorly selected to embed data, which is helpful to improve the embedding performance. Compared with the prior arts, the proposed method can achieve a higher PSNR under the same EC. But using only one bin as inner region may not be able to provide sufficient embedding positions. In our future work we will try to improve EC.

Acknowledgments

Supported by National Natural Science Foundation of China (61103174) and Foundation for Distinguished Young Talents in Higher Education of Guangdong (LYM10116).

References

[1] R. Caldelli, F. Filippini, R. Becarelli, Reversible watermarking techniques: an overview and a classification, EURASIP Journal on Information Security (2010). article ID 134546.
[2] J. Tian, Reversible data embedding using a difference expansion, IEEE Transactions on Circuits and Systems for Video Technology 13 (8) (2003) 890–896.
[3] J. Fridrich, M. Goljan, R. Du, Lossless data embedding - new paradigm in digital watermarking, EURASIP Journal on Applied Signal Processing 2002 (2) (2002) 185–196.
[4] M.U. Celik, G. Sharma, A.M. Tekalp, E. Saber, Lossless generalized-LSB data embedding, IEEE Transactions on Image Processing 14 (2) (2005) 253–266.
[5] A.M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, IEEE Transactions on Image Processing 13 (8) (2004) 1147–1156.
[6] X. Wang, X. Li, B. Yang, Z. Guo, Efficient generalized integer transform for reversible watermarking, IEEE Signal Processing Letter 17 (6) (2010) 567–570.
[7] F. Peng, X. Li, B. Yang, Adaptive reversible data hiding scheme based on integer transform, Signal Processing 92 (1) (2012) 54–62.
[8] L. Kamstra, H.J.A.M. Heijmans, Reversible data embedding into images using wavelet techniques and sorting, IEEE Transactions on Image Processing 14 (12) (2005) 2082–2090.
[9] S. Weng, Y. Zhao, J.S. Pan, R. Ni, Reversible watermarking based on invariability and adjustment on pixel pairs, IEEE Signal Processing Letters 15 (2008) 721–724.
[10] H.J. Kim, V. Sachnev, Y.Q. Shi, J. Nam, H.G. Choo, A novel difference expansion transform for reversible data embedding, IEEE Transactions on Information Forensics and Security 4 (3) (2008) 456–465.
[11] M. Liu, H.S. Seah, C. Zhu, W. Lin, F. Tian, Reducing location map in prediction-based difference expansion for reversible image data embedding, Signal Processing 92 (3) (2012) 819–828.
[12] D.M. Thodi, J.J. Rodriguez, Expansion embedding techniques for reversible watermarking, IEEE Transactions on Image Processing 16 (3) (2007) 721–730.
[13] Y. Hu, H.K. Lee, J. Li, DE-based reversible data hiding with improved overflow location map, IEEE Transactions on Circuits and Systems for Video Technology 19 (2) (2009) 250–260.
[14] W. Hong, T.S. Chen, C.W. Shiu, Reversible data hiding for high quality images using modification of prediction errors, Journal of Systems and Software 82 (11) (2009) 1833–1842.
[15] C.C. Lin, N.L. Hsueh, A lossless data hiding scheme based on three-pixel block differences, Pattern Recognition 41 (4) (2008) 1415–1425.
[16] W.L. Tai, C.M. Yeh, C.C. Chang, Reversible data hiding based on histogram modification of pixel differences, IEEE Transactions on Circuits and Systems for Video Technology 19 (6) (2009) 906–910.
[17] P. Tsai, Y.C. Hu, H.L. Yeh, Reversible image hiding scheme using predictive coding and histogram shifting, Signal Processing 89 (6) (2009) 1129–1143.

[18] D. Coltuc, Improved embedding for prediction-based reversible watermarking, IEEE Transactions on Information Forensics and Security 6 (3) (2011) 873–882.
[19] X. Li, B. Yang, T. Zeng, Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection, IEEE Transactions on Image Processing 20 (12) (2011) 3524–3533.
[20] V. Sachnev, H.J. Kim, J. Nam, S. Suresh, Y.Q. Shi, Reversible watermarking algorithm using sorting and prediction, IEEE Transactions on Circuits and Systems for Video Technology 19 (7) (2009) 989–999.
[21] L. Luo, Z. Chen, M. Chen, X. Zeng, Z. Xiong, Reversible image watermarking using interpolation technique, IEEE Transactions on Information Forensics and Security 5 (1) (2010) 187–193.
[22] W. Hong, An efficient prediction-and-shifting embedding technique for high quality reversible data hiding, EURASIP Journal on Advances in Signal Processing (2010). article ID 104835.
[23] Z. Ni, Y.Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Transactions on Circuits and Systems for Video Technology 16 (3) (2006) 354–362.
[24] M.J. Weinberger, G. Seroussi, G. Sapiro, The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS, IEEE Transactions on Image Processing 9 (8) (2000) 1309–1324.
[25] G. Coatrieux, C.L. Guillou, J.M. Cauvin, C. Roux, Reversible watermarking for knowledge digest embedding and reliability control in medical images, IEEE Transactions on Information Technology in Biomedicine 13 (2) (2009) 158–165.