J. Vis. Commun. Image R. 46 (2017) 58–69

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Efficient PVO-based reversible data hiding using multistage blocking and prediction accuracy matrix ☆

Wenguang He *, Jie Cai, Ke Zhou, Gangqiang Xiong

School of Information Engineering, Guangdong Medical University, Guangdong 524023, China

ARTICLE INFO

Article history:
Received 30 July 2016
Revised 23 February 2017
Accepted 6 March 2017
Available online 8 March 2017

Keywords:
Reversible data hiding
Pixel value ordering
Multistage blocking
Prediction accuracy matrix

ABSTRACT

In recent years, pixel value ordering based reversible data hiding has become a hot research topic for its high-fidelity. In this approach, only the maximum and minimum of pixel block are predicted and modified to embed data and the preservation of pixel values order guarantees the reversibility. So far, the optimal block size can only be exhaustively searched until Wang et al. propose the dynamic blocking strategy which enables the combination of two various-sized blocks. By further dividing flat block into four sub-blocks to retain larger embedding capacity, dynamic blocking can employ less high complexity blocks for a given embedding capacity. However, the lack of host image dependent automatic block classification mechanism still exposes the fact that their work is far from efficient and comprehensive. In this paper, to address this drawback and to better exploit image redundancy, a really efficient and more comprehensive blocking strategy namely multistage blocking is proposed. High efficiency lies in prediction accuracy matrix based thresholds determination, which enables infinitely extended multistage blocking in theory. The superiority of the proposed scheme is also experimentally verified.

1. Introduction

Data hiding [1] is a technique that embeds secret message into host media to ensure copyright protection, authentication, and so on. As an important branch of data hiding, reversible data hiding (RDH) can recover the host image exactly as well as extracting the embedded data [2] and thus has been applied to many quality sensitive fields such as military, medical imaging and remote sensing. In general, the performance of RDH scheme can be evaluated by the embedding capacity (EC) and embedding introduced distortion.

Early RDH schemes are mainly based on lossless compression [3–5]. Usually, these schemes cannot provide high EC while keeping distortion low. Later on, more efficient RDH schemes based on histogram modification and expansion technique have been devised. Difference expansion (DE) technique is firstly proposed by Tian [6] with the idea of embedding the secret message by expanding the difference between adjacent pixels. Although its EC is bounded by 0.5 bpp, DE can achieve better performance and thus has been widely investigated and developed, mainly in the aspects of integer-to-integer transformation [7–9] and prediction error expansion (PEE) [10,11]. Nowadays, PEE has become the most effective and extensively exploited RDH technique. Instead of using the difference between adjacent pixels, the difference between pixel intensity and its predicted value computed from the context namely prediction error is utilized for expansion embedding. As the local correlation of a larger neighborhood is exploited in PEE, better performance can be derived compared with DE. Taking into account that better predictor always produces errors with smaller magnitude and hence embedding distortion can be reduced, many prediction methods are investigated for prediction accuracy improvement, such as median edge detector [10], rhombus prediction [12], interpolation techniques [13], gradient adjusted prediction [14] and partial differential equation [15]. Moreover, the PEE technique can also be developed in other aspects such as location map reduction [16], context modification [17], adaptive embedding [18,19], two-dimensional histogram modification [20,21] and pixel value ordering [22–25].

Besides DE, the histogram modification based scheme proposed by Ni [26] is another remarkable work of RDH, in which the intensity histogram bins between the peak point and zero point are shifted before the peak points are employed for data embedding. However, its EC is limited despite high marked image quality and low computational complexity. To construct a sharper histogram, Lee [27] proposed to utilize the difference histogram instead and Tsai [28] proposed to utilize the prediction error histogram.

☆ This paper has been recommended for acceptance by M.T. Sun.

* Corresponding author.
E-mail address: 56207403@qq.com (W. He).

http://dx.doi.org/10.1016/j.jvcir.2017.03.010
1047-3203/

Afterwards, difference/prediction error histogram modification is also adopted in many works [29–33].

Recently, several PEE-based schemes have attracted much attention since they can achieve very high visual quality [19–25]. As only certain histogram bins are expanded to carry secret data, maximum modification of pixel values can be ensured at most 1 and hence high visual quality is guaranteed at the cost of limited EC. So far, the idea of pixel value ordering (PVO) firstly proposed by Li [22] has been proved outperforms conventional PEEs [12,13,16,18]. In PVO, pixel values within block are firstly sorted and then the second largest/smallest one is used to predict the largest/smallest one to produce prediction errors further employed for expansion embedding. Due to the high correlation of pixels within block, a rather sharp histogram which means remarkable embedding performance can be obtained. Moreover, Li et al. also proposed to measure block complexity by the difference between the second largest pixel value and the second smallest one. Then flat blocks are preferentially utilized to embed data. Later on, Ou [23] and Peng [24] improved Li et al.'s work [22] based on the same consideration that smooth blocks cannot be fully exploited since bin 0, which is usually the second-highest bin, is excluded from data embedding. In [23], Ou proposed a new embedding strategy called PVO-k, where k is the number of maximum-valued (minimum-valued) pixels. For data embedding, all maximum-valued (minimum-valued) pixels will be modified as a unit to embed one bit data. What is more, a new block complexity computed from block context was proposed. Peng [24] brought bin 0 into embedding scheme by designing a new prediction error and similarly achieved better performance than conventional PVO.

In the above PVO-based schemes [22–24], the optimal block size has to be exhaustively searched according to EC requirement for the reason that larger block size generally provides a relatively smaller EC whereas higher PSNR. In view of this, Wang [25] proposed an effective extension namely dynamic blocking. That is, a large block size is used in textured areas to ensure high PSNR whereas a small block size is used in smooth areas to achieve large EC. In practice equal-sized blocks are classified into rough, normal and flat according to block complexity, then rough block is excluded from data embedding and flat block is further divided into four sub-blocks. Taking into account that such further division can obtain three more flat embedding primitive ideally, less high complexity blocks can be employed and thus better performance can be achieved. However, Wang et al.'s work is still far from efficient and comprehensive. Experimental results show that only large EC evidently benefits from the combination of two various-sized blocks. For small EC, even worse performance is achieved. Another deficiency lies in the inefficiency in block classification. So far, the best combination of two thresholds employed for dynamic blocking can only be exhaustively searched and such operation is really time-consuming.

Specially, the root cause of such phenomenon can be attributed to the incomplete blocking strategy and the lack of host image dependent automatic block classification mechanism. As supposed by Wang et al., the worse performance can be avoided or reversed by combining three or even more than three various-sized blocks. However, the existing guideless further division is even incapable of combining two various-sized blocks with reasonable computational complexity. Following Wang et al.'s work, we present an extended and generalized blocking strategy namely multistage blocking in this paper. Compared with Wang et al.'s dynamic blocking, multistage blocking has two major advantages. First, multistage blocking proposes to fulfill image partition in the form of binary tree. In theory, n-stage blocking enables the combination of n various-sized blocks. Second, the attached host image dependent automatic block classification mechanism matching with multistage blocking is also presented. With the idea of prediction accuracy matrix, the best combination of n thresholds employed for multistage blocking can be easily determined.

The rest of the paper is organized as follows. In Section 2, the related works are introduced. Section 3 presents the proposed scheme in details. Experimental results and performance comparisons with other schemes are shown in Section 4. Finally, we conclude the paper in Section 5.

2. Related work

In this section, as a review, previous PVO-based schemes [22,24,25] are briefly introduced.

2.1. PVO-based scheme of Li et al.

Li et al.'s work [22] effectively utilizes the similarity among pixels within block to provide sufficient EC with low distortion. Taking maximum-modification-based embedding as an example, the overall process can be described as follows.

Step1. divide the host image into a set of non-overlapped blocks.

Step2. For each block consisted of n pixels, sort all pixel values $(x_1, x_2, \ldots, x_n)$ in ascending order to obtain $(x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)})$ where $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ is the unique one-to-one mapping such that: $x_{\sigma(1)} \leqslant x_{\sigma(2)} \leqslant \cdots \leqslant x_{\sigma(n)}, \sigma(i) < \sigma(j)$ if $x_{\sigma(i)} = x_{\sigma(j)}$ and $i < j$. Then the prediction error is obtained by using $x_{\sigma(n-1)}$ to predict $x_{\sigma(n)}$ as

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

Step3. Data bit $b \in \{0, 1\}$ is embedded by modifying prediction error PE_{max} to

$$\widetilde{PE}_{max} = \begin{cases} PE_{max}, & \text{if } PE_{max} = 0 \\ PE_{max} + b, & \text{if } PE_{max} = 1 \\ PE_{max} + 1, & \text{if } PE_{max} > 1 \end{cases} \tag{2}$$

Accordingly, the maximum is modified to

$$\widetilde{x}_{\sigma(n)} = x_{\sigma(n-1)} + \widetilde{PE}_{max} = \begin{cases} x_{\sigma(n)}, & \text{if } PE_{max} = 0 \\ x_{\sigma(n)} + b, & \text{if } PE_{max} = 1 \\ x_{\sigma(n)} + 1, & \text{if } PE_{max} > 1 \end{cases} \tag{3}$$

Notice that only the maximum is either unchanged or increased. Thus, the pixel values order remains unchanged and reversibility is guaranteed. At the decoder, the prediction error $\widetilde{PE}_{max}$ can be obtained in the same way. Then data bit can be extracted as

$$b = \widetilde{PE}_{max} - 1, \text{if } \widetilde{PE}_{max} \in \{1, 2\} \tag{4}$$

And the original value is recovered

$$x_{\sigma(n)} = \begin{cases} \widetilde{x}_{\sigma(n)}, & \text{if } \widetilde{PE}_{max} = 0 \\ \widetilde{x}_{\sigma(n)} - b, & \text{if } \widetilde{PE}_{max} \in \{1, 2\} \\ \widetilde{x}_{\sigma(n)} - 1, & \text{if } \widetilde{PE}_{max} > 2 \end{cases} \tag{5}$$

Similarly to the maximum, the minimum is decreased or unchanged after embedding data into the minimum. For the minimum, the difference PE_{min} is defined by $PE_{min} = x_{\sigma(1)} - x_{\sigma(2)}$ and the highest bin -1 is used for expansion. Moreover, the block complexity 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.

2.2. PVO-based scheme of Peng et al.

Peng [24] proposed to improve PVO performance by bringing bin 0 into embedding scheme. To this end, a new prediction error is designed as

$$d_{max} = x_u - x_v \tag{6}$$

where $u = min(\sigma(n), \sigma(n-1)), v = max(\sigma(n), \sigma(n-1))$. In this way, the new error d_{max} is no longer always positive and the largest bin usually switches to bin 0 (which is bin 1 in [22]). In addition, the histograms of PE_{max} and d_{max} have the following correspondence:

$$\begin{cases} PE_{max}(0) = d_{max}(0) \\ PE_{max}(k) = d_{max}(k) + d_{max}(-k) \end{cases} \tag{7}$$

As only $PE_{max}(1)$ is used for expansion in [22], whereas $d_{max}(0)$ and $d_{max}(1)$ are used here, larger EC can be obtained by $d_{max}(0) + d_{max}(1) - PE_{max}(1) = d_{max}(0) - PE_{max}(-1) > 0$. Next, data bit $b \in \{0,1\}$ is embedded by modifying prediction error d_{max} to

$$\widetilde{d}_{max} = \begin{cases} d_{max} + b, & \text{if } d_{max} = 1 \\ d_{max} + 1, & \text{if } d_{max} > 1 \\ d_{max} - b, & \text{if } d_{max} = 0 \\ d_{max} - 1, & \text{if } d_{max} < 0 \end{cases} \tag{8}$$

Accordingly, the maximum is modified to

$$\widetilde{x}_{\sigma(n)} = x_{\sigma(n-1)} + \widetilde{d}_{max} = \begin{cases} x_{\sigma(n)} + b, & \text{if } d_{max} = 1 \\ x_{\sigma(n)} + 1, & \text{if } d_{max} > 1 \\ x_{\sigma(n)} + b, & \text{if } d_{max} = 0 \\ x_{\sigma(n)} + 1, & \text{if } d_{max} < 0 \end{cases} \tag{9}$$

Similarly, pixel values order remains unchanged as only the maximum is either unchanged or increased. Then the prediction error $\widetilde{d}_{max}$ can be obtained in the same way to extract data bit as

$$b = \begin{cases} \widetilde{d}_{max} - 1, & \text{if } \widetilde{d}_{max} \in \{1,2\} \\ -\widetilde{d}_{max}, & \text{if } \widetilde{d}_{max} \in \{-1,0\} \end{cases} \tag{10}$$

And the original value is recovered

$$x_{\sigma(n)} = \begin{cases} \widetilde{x}_{\sigma(n)} + b, & \text{if } \widetilde{d}_{max} \in \{1,2\} \\ \widetilde{x}_{\sigma(n)} - 1, & \text{if } \widetilde{d}_{max} > 2 \\ \widetilde{x}_{\sigma(n)} - b, & \text{if } \widetilde{d}_{max} \in \{-1,0\} \\ \widetilde{x}_{\sigma(n)} - 1, & \text{if } \widetilde{d}_{max} < -1 \end{cases} \tag{11}$$

The minimum-modification-based embedding is similar to the maximum-based one described above. The prediction error is calculated accordingly as

$$d_{min} = x_s - x_t \tag{12}$$

where $s = min(\sigma(1), \sigma(2)), t = max(\sigma(1), \sigma(2))$. Next, the marked value of the minimum is determined as

$$\widetilde{x}_{\sigma(1)} = \begin{cases} x_{\sigma(1)} - b, & \text{if } d_{min} = 1 \\ x_{\sigma(1)} - 1, & \text{if } d_{min} > 1 \\ x_{\sigma(1)} - b, & \text{if } d_{min} = 0 \\ x_{\sigma(1)} - 1, & \text{if } d_{min} < 0 \end{cases} \tag{13}$$

At the decoder, the prediction error $\widetilde{d}_{min}$ can also be obtained in the same way to extract data bit as

$$b = \begin{cases} \widetilde{d}_{min} - 1, & \text{if } \widetilde{d}_{min} \in \{1,2\} \\ -\widetilde{d}_{min}, & \text{if } \widetilde{d}_{min} \in \{-1,0\} \end{cases} \tag{14}$$

And the original value is recovered

$$x_{\sigma(1)} = \begin{cases} \widetilde{x}_{\sigma(1)} + b, & \text{if } \widetilde{d}_{min} \in \{1,2\} \\ \widetilde{x}_{\sigma(1)} + 1, & \text{if } \widetilde{d}_{min} > 2 \\ \widetilde{x}_{\sigma(1)} + b, & \text{if } \widetilde{d}_{min} \in \{-1,0\} \\ \widetilde{x}_{\sigma(1)} + 1, & \text{if } \widetilde{d}_{min} < -1 \end{cases} \tag{15}$$

2.3. PVO-based scheme of Wang et al.

Wang [25] further extended Peng et al.'s work [24] with the idea of dynamic blocking which enables the combination of two various-sized blocks. In this approach, 4×4 blocks are classified into rough, normal and flat according to block complexity. Then rough block is excluded from data embedding and flat block is further divided into four 2×2 sub-blocks. Obviously, the block complexity plays a more important role here since it not only tells whether a block is flat or not, but also tells whether a block is flat enough to be further divided.

To ensure that the block complexity preserves after data embedding, a new definition of block complexity is given. Suppose a 4×4 block is divided into four 2×2 sub-blocks $S_k = \left\{x^k_{\sigma(1)}, x^k_{\sigma(2)}, x^k_{\sigma(3)}, x^k_{\sigma(4)}\right\}, k \in [1,4]$. We can see that no matter data embedding occurs in one 4×4 block or in four 2×2 sub-blocks, eight pixel values $\left\{x^k_{\sigma(2)}, x^k_{\sigma(3)}\right\}, k \in [1,4]$ always remain unchanged. Thus, block complexity can be measured as $NL = max(x^1_{\sigma(3)}, x^2_{\sigma(3)}, x^3_{\sigma(3)}, x^4_{\sigma(3)}) - min(x^1_{\sigma(2)}, x^2_{\sigma(2)}, x^3_{\sigma(2)}, x^4_{\sigma(2)})$. With complexity NL and two thresholds $T_1, T_2, (-1 \leqslant T_2 \leqslant T_1 \leqslant 255)$, 4×4 block is classified as follows:

- Rough block with $NL > T_1$ is excluded from data embedding;
- Normal block with $T_2 < NL \leqslant T_1$ can be used for embedding;
- Flat block with $NL \leqslant T_2$ can be further divided to embed more data bits.

3. Proposed scheme

For previous PVO-based schemes, only one or two bins of the prediction error histogram are expanded to carry data and the maximum modification to pixel values is 1 in data embedding. Hence, under the premise of providing sufficient expandable errors, the number of to-be-shifted errors can directly reflect embedding performance. The fewer to-be-shifted errors is, the less distortion and better performance is.

Although to-be-shifted error is inevitable according to the prediction mechanism of PVO, one can reduce its number by various strategies such as block complexity measurement and optimal block size. In [22–24], the embedding procedure has to be carried out for numerous block sizes to determine the best one and the computational complexity is a little high in consequence. In [25], computational complexity switches to lie in seeking the best combination of two thresholds T_1, T_2 which determines whether a block is available for data embedding or not and whether an available block should be further divided or not. Experimental results show that such seeking caused computational complexity has become the fatal deficiency of Wang et al.'s work and indicates the infeasibility of introducing more various-sized block.

Before presenting our solution, an overall evaluation of PVO prediction is first given and then a generalized multistage blocking strategy is designed.

3.1. Evaluation of PVO prediction

As known in [22–24], larger block size provides smaller EC whereas higher PSNR. Wang [25] summarized this as that the

difference between the second largert/smallest pixel and the largest/smallest pixel tends to be smaller in a larger block. Here, we propose to explain this with prediction accuracy.

For a given host image and specified block size, we can easily get the number of expandable errors totally obtained from blocks with complexity t (denoted as N_e^t) and the number of corresponding to-be-shifted errors (denoted as N_s^t). Then we define prediction accuracy PA as

$$PA = \frac{N_e^t}{N_e^t + N_s^t} \tag{16}$$

Generally, the smaller t is, the higher PA is. Taking image Lena as an example, the PA curves calculated by Peng et al.'s scheme using various block sizes are shown in Fig. 1(a). Fig. 1(a) confirms that for all block sizes, prediction accuracy PA decreases with the increase of block complexity t. Moreover, larger block size generally produces higher PA for the same block complexity t.

Notice that higher PA does not guarantee sufficient EC. Here we extend PA as

$$PA = \frac{N_e^T}{N_e^T + N_s^T} = \frac{\sum_{t=0}^{T} N_e^t}{\sum_{t=0}^{T} N_e^t + \sum_{t=0}^{T} N_s^t} \tag{17}$$

If without taking into account other factors caused consumption, N_e^T is also the maximum EC. Thus, N_e^T and PA evaluate the performance of PVO embedding. Fig. 1(b) shows the $N_e^T - PA$ curves using various block sizes, which also points out a method for determining the optimal block size according to EC requirement. In the following, this method will be extended and applied in thresholds determination.

3.2. Generalized multistage blocking

In this section, we define the framework of n-stage blocking. Suppose the host image is divided into a set of $u \times v$ sized non-overlapped blocks namely 1-stage block or root block (denoted as B_r). Then for a given n, further divide each root block into a set of $\frac{u}{2^{\lfloor n/2 \rfloor}} \times \frac{v}{2^{\lfloor n/2 \rfloor}}$ sized sub-blocks, called n-stage block or leaf block (denoted as B_l). As illustrated in Fig. 2, No.4 to No.7 blocks would become leaf blocks if $n = 2, 3$ and No.16 to No.31 blocks would become leaf blocks if $n = 4, 5$.

Besides root block (1-stage block) and leaf block (n-stage block), all other blocks are identified as intermediate block (denoted as B_i). Apparently, intermediate block comes from the union of leaf blocks or the union of the next stage intermediate blocks. As for the union problem, we propose to define the median of each leaf block as $ME = (sl(B_l) + sm(B_l))/2$, where $sl(.)$ and $sm(.)$ denote selection functions for the second largest and the second smallest pixel values of specified block. After sorting all leaf blocks according to ME in ascending order, the union of two adjacent leaf blocks forms the (n-1)-stage intermediate block. For example, the union of No.16 and No.17 blocks in Fig. 2 forms intermediate block No.8 block.

Another kind of union is the union of intermediate blocks. As intermediate block is essentially the union of leaf blocks such as $B_i = \left\{B_l^1, B_l^2, \ldots\right\}$, the median of each intermediate block can be defined as $ME = (max(sl(B_l^1), sl(B_l^2), \ldots) + min(sm(B_l^1), sm(B_l^2), \ldots))/2$. After sorting the same stage intermediate blocks according to ME in ascending order, the union of two adjacent intermediate blocks forms a double-sized intermediate block or even the root block. For example, the union of No.8 and No.9 blocks in Fig. 2 forms intermediate block No.4 block.

Finally we introduce a series of thresholds and define the complexity of root block to fulfill n-stage blocking. As root block is also the union of leaf blocks, we define its complexity as $NL = max(sl(B_l^1), sl(B_l^2), \ldots) - min(sm(B_l^1), sm(B_l^2), \ldots)$. Then each root block can be processed as described in Fig. 3. For 3-stage blocking, three thresholds T_1, T_2, T_3 are employed to fulfill root block classification. Specially, once $T_3 = -1$, 3-stage blocking would turn to 2-stage blocking. Similarly, once $T_5 = -1$, 5-stage blocking would turn to 4-stage blocking. Here, for simplicity, only 3-stage blocking is taken into account in this paper. In this case, we simply identify root block with $NL > T_1$ as Type-I block, root block with $T_2 < NL \leqslant T_1$ as Type-II block, root block with $T_3 < NL \leqslant T_2$ as Type-III block and root block with $NL \leqslant T_3$ as Type-IV block.

3.3. Thresholds determination

Since expandable and to-be-shifted errors can be produced from three types of blocks, prediction accuracy can be further extended in our method

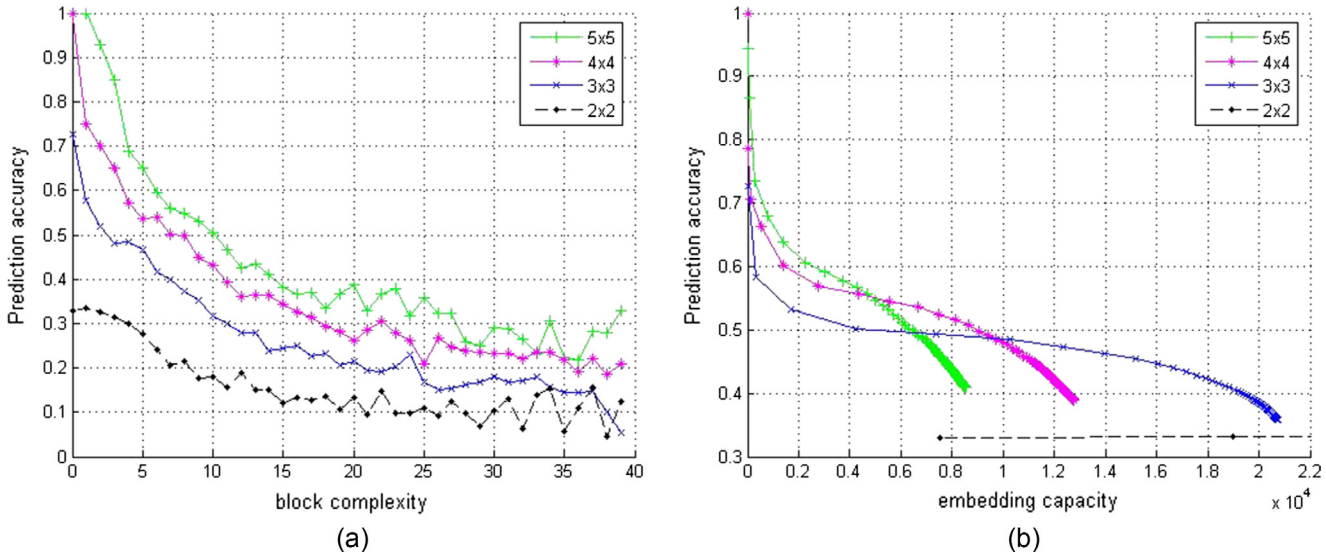

Fig. 1. PA curves and its corresponding $N_e^T - PA$ curves obtained by Peng et al.' scheme using various block sizes.

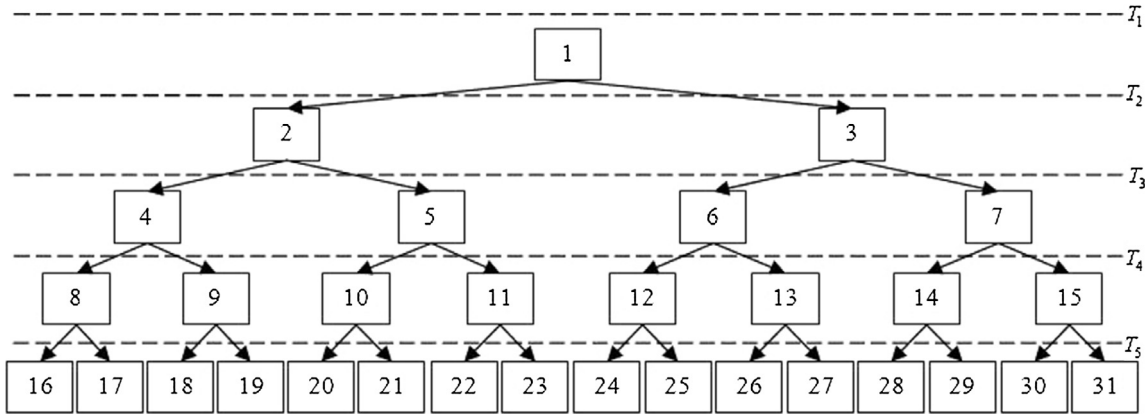

Fig. 2. Illustration of n-stage blocking.

```
if NL > T1
    %do nothing
elseif NL > T2
    %process root block as a whole one
elseif NL > T3
    %process root block as two sub – blocks
elseif NL > T4
    %process root block as four sub – blocks
elseif ...
    %...
end
```

Fig. 3. Framework of multistage blocking based embedding and extraction.

Table 1
Meaning of symbols used.

Symbols	Meaning
N^t_{el}, N^t_{sl}	Numbers of expandable and to-be-shifted errors from Type-IV blocks
N^t_{ei}, N^t_{si}	Numbers of expandable and to-be-shifted errors from Type-III blocks
N^t_{er}, N^t_{sr}	Numbers of expandable and to-be-shifted errors from Type-II blocks

$$
\begin{aligned}
N_e^T &= \sum_{t=0}^{T_3} N_{el}^t + \sum_{t=T_3+1}^{T_2} N_{ei}^t + \sum_{t=T_2+1}^{T_1} N_{er}^t \\
N_s^T &= \sum_{t=0}^{T_3} N_{sl}^t + \sum_{t=T_3+1}^{T_2} N_{si}^t + \sum_{t=T_2+1}^{T_1} N_{sr}^t \\
PA &= \frac{N_e^T}{N_e^T + N_s^T}
\end{aligned}
\tag{18}
$$

Table 1 lists the symbols' meaning used in Eq. (18). Now, according to the EC requirement, we are required to figure out the best combination of $\{T_1, T_2, T_3\}$ which produces the highest PA under the premise of providing sufficient EC.

First, we simply suppose all root blocks are Type-IV, Type-III and Type-II blocks respectively to obtain the statistics shown in Table 1. Table 2 shows part of the results obtained on image Lena using 4×4 root block size. Referring to Table 2, we can see that

Table 2
Part of statistic results obtained on image Lena.

	N^t_{er}	N^t_{sr}	N^t_{ei}	N^t_{si}	N^t_{el}	N^t_{sl}
$t=0$	3	1	6	2	9	7
$t=1$	67	57	142	106	272	224
$t=2$	504	324	916	740	1604	1708
$t=3$	1201	983	2265	2103	3867	4869
$t=4$	1566	1278	2766	2922	4813	6563
$t=5$	1601	1475	2884	3268	4869	7435

Type-IV block always produces the most expandable errors, followed by Type-III block. However, the newly added expandable errors are always accompanied by newly added to-be-shifted errors. Here, we propose to measure the efficiency of such EC improvement by

$$
\begin{aligned}
PA(t,0) &= \frac{N_{er}^t}{N_{er}^t + N_{sr}^t} \\
PA(t,1) &= \frac{N_{ei}^t - N_{er}^t}{(N_{ei}^t - N_{er}^t) + (N_{si}^t - N_{sr}^t)} \\
PA(t,2) &= \frac{N_{el}^t - N_{ei}^t}{(N_{el}^t - N_{ei}^t) + (N_{sl}^t - N_{si}^t)}
\end{aligned}
\tag{19}
$$

where $PA(t,0)$ measures the efficiency in turning Type-I blocks into Type-II blocks, $PA(t,1)$ measures the efficiency in turning Type-II blocks into Type-III blocks, and $PA(t,2)$ measures the efficiency in turning Type-III blocks into Type-IV blocks. Then the prediction accuracy matrix (PAM) is constructed as

$$
\begin{bmatrix}
PA(0,0) & PA(0,1) & PA(0,2) \\
\vdots & \vdots & \vdots \\
PA(255,0) & PA(255,1) & PA(255,2)
\end{bmatrix}
\tag{20}
$$

PAM provides a fast and effective method for thresholds determination. As $PA(t,0), PA(t,1), PA(t,2)$ are independent of each other, we are free to chase larger EC by enlarging T_1, T_2, T_3 while maintaining relatively highest prediction accuracy. Using the statistic results from Table 2, the corresponding PAM for image Lena can be calculated as

$$
\begin{bmatrix}
0.75 & 0.75 & 0.375 \\
0.54 & 0.605 & 0.524 \\
0.609 & 0.498 & 0.415 \\
0.55 & 0.487 & 0.367 \\
0.551 & 0.422 & 0.36 \\
0.52 & 0.417 & 0.323 \\
\vdots & \vdots & \vdots
\end{bmatrix}
\tag{21}
$$

T_1	0	1	2	3	4	4	5
T_2	0	1	1	1	1	1	1
T_3	0	0	0	0	0	1	1

Fig. 4. Process of the change of three thresholds.

Suppose the EC requirement is 5,000 bits, we first initialize $\{T_1, T_2, T_3\}$ by $\{0, 0, 0\}$. Then we enlarge T_2 by 1 since $PA(T_2 + 1, 1)(0.605)$ is the largest one among $\{PA(T_1 + 1, 0), PA(T_2 + 1, 1), PA(T_3 + 1, 2)\}$. Notice that T_1 should be always not smaller than T_2 and T_2 is always not smaller than T_3, so here T_1 got enlarged by 1 automatically. Next, T_1 is chosen to be enlarged by 1 since $PA(T_1 + 1, 1)(0.609)$ is the largest one among $\{PA(T_1 + 1, 0), PA(T_2 + 1, 1), PA(T_3 + 1, 2)\}$. The enlargement process will be executed over and over until required EC is satisfied. Fig. 4 shows the whole process of the change of three thresholds. By taking $T_1 = 5, T_2 = 1$ and $T_3 = 1$, we finally achieve a maximum EC of 5153 and a highest PA of 55%.

To directly observe the influence of multistage blocking on PVO prediction, here we take images Lena and Barbara as test images. For our method, we vary EC from 2000 bits to 21,000 bits with a step size of 1000 bits. As shown in Fig. 5, much higher PA is obtained compared with that of Peng et al.'s using various block sizes.

3.4. Implementation for the proposed scheme

First, the proposed multistage blocking data embedding procedure will be introduced.

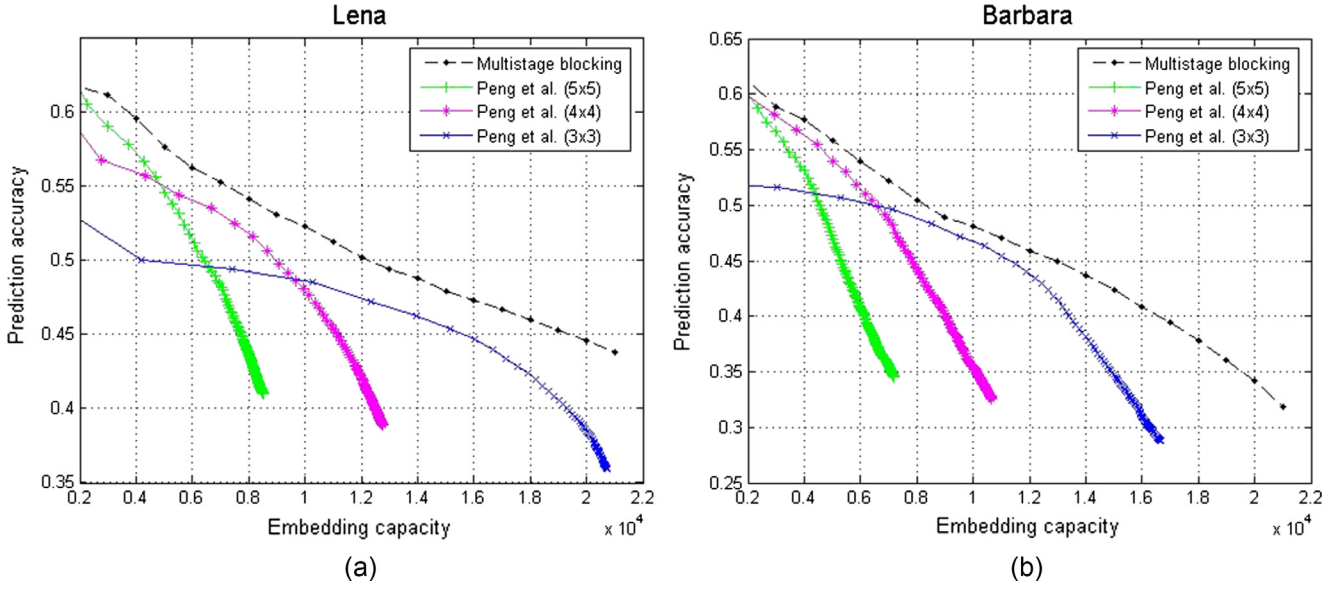

Fig. 5. Prediction accuracy comparison between our method and Peng et al.' using various block sizes.

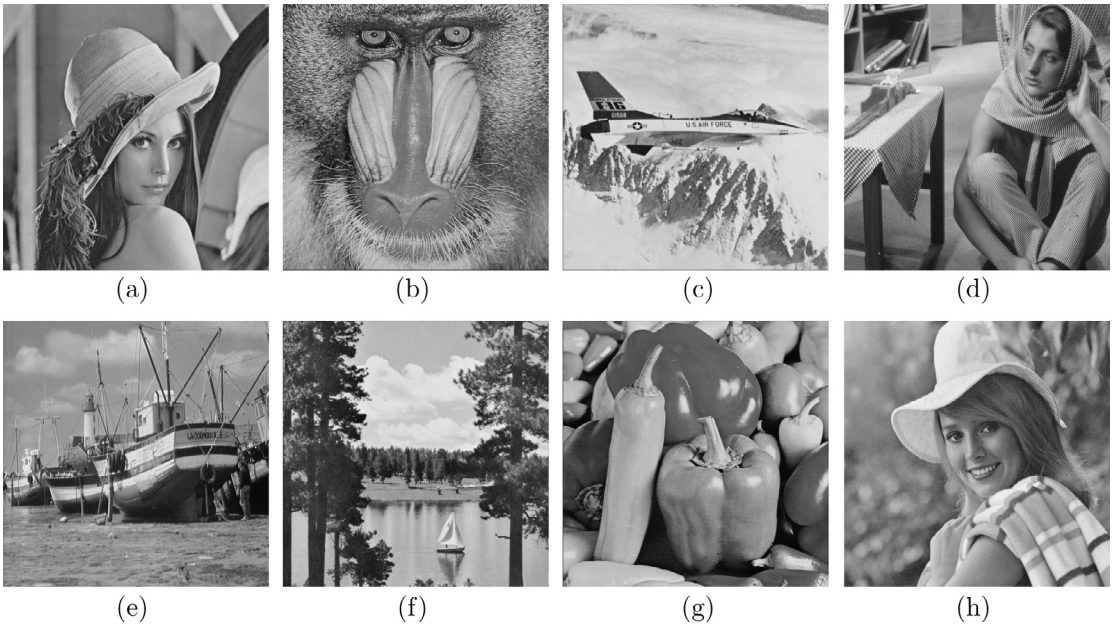

Fig. 6. Original test images: (a) Lena, (b) Baboon, (c) Airplane, (d) Barbara, (e) Boat, (f) Lake, (g) Peppers, (h) Elaine.

Step 1. Image partition divide host image I into a set of $u \times v$ sized blocks $\left\{B_r^1, B_r^1, \ldots, B_r^N\right\}$.

Step 2. Location map construction
For each block, if any pixel value within equals to 255 or 0, we mark this block as exceptional by $LM(i) = 1$ since it would cause over/underflow. Otherwise, we take $LM(i) = 0$. Then losslessly compress the location map using arithmetic coding.

Step 3. Data embedding
Successively embed the data bits into blocks. For each block B_r:

- If $LM(i) = 1$, the over/underflow would occur and we do nothing with it.
- If $LM(i) = 0$ and $NL > T_1$, it is unsuitable for embedding and we do nothing with it either.
- If $LM(i) = 0$ and $T_2 < NL \leqslant T_1$, the maximum and minimum of it are shifted or expanded to embed data according to (9) and (13).
- If $LM(i) = 0$ and $T_3 < NL \leqslant T_2$, it is divided into two intermediate blocks $B_r = \left\{B_i^1, B_i^2\right\}$. For each of them, the maximum and minimum are shifted or expanded to embed data according to (9) and (13).
- If $LM(i) = 0$ and $NL \leqslant T_3$, it is divided into four leaf blocks $B_r = \left\{B_l^1, B_l^2, B_l^3, B_l^4\right\}$. For each of them, the maximum and minimum are shifted or expanded to embed data according to (9) and (13).

This step continues until all data bits have been embedded, and we denote E_B as the index of last data-carrying block.

Step 4. Auxiliary information and location map embedding

Record the least significant bits of first $32 + 2\lceil \log_2(N) \rceil + Lclm$ image pixels (denoted as S_{LSB}), where $Lclm$ is the length of the compressed location map. Replace these LSBs by the following auxiliary information and the compressed location map:

- Root block size parameters u (4 bits) and v (4 bits),
- Block complexity thresholds T_1 (8 bits), T_2 (8 bits) and T_3 (8 bits),
- End position E_B ($\lceil \log_2(N) \rceil$ bits),
- Length of the compressed location map $Lclm$ ($\lceil \log_2(N) \rceil$ bits).

Finally, embed the binary sequence S_{LSB} into the remaining blocks to obtain the marked image.

As the inverse process of data embedding, data extraction and image recovery are described below.

Step 1. Auxiliary information and location map extraction
Read the LSBs of the first $32 + 2\lceil \log_2(N) \rceil$ pixels of the marked image to retrieve the auxiliary information, including the values of u, v, T_1, T_2, T_3, E_B, and $Lclm$. Then read the LSBs of the next $Lclm$ pixels to retrieve the compressed location map. Recover the location map LM by decompressing the compressed location map.

Step 2. Sequence S_{LSB} extraction and image restoration
First, divide the marked image into a set of blocks and calculate their complexities. Then extract the sequence from blocks $\left\{B_r^{E_B+1}, \ldots, B_r^N\right\}$. For each block B_r:

- If $LM(i) = 0$ and $NL \leqslant T_3$, it is divided into four leaf blocks $B_r = \left\{B_l^1, B_l^2, B_l^3, B_l^4\right\}$. For each of them, extract hidden data bits and recover original values according to (10,11,14,15).
- If $LM(i) = 0$ and $T_3 < NL \leqslant T_2$, it is divided into two intermediate blocks $B_r = \left\{B_i^1, B_i^2\right\}$. For each of them, extract hidden data bits and recover original values according to (10,11,14,15).
- If $LM(i) = 0$ and $T_2 < NL \leqslant T_1$, directly extract hidden data bits and recover original values according to (10,11,14,15).
- Otherwise, there is no hidden data and changed values.

Step 3. Data extraction and image restoration

Replace the LSBs of first $32 + 2\lceil \log_2(N) \rceil + Lclm$ image pixels by the sequence just extracted. Then use the same method of step2 to extract the hidden data from blocks $\left\{B_r^1, \ldots, B_r^{E_B}\right\}$, and meanwhile to realize restoration for those blocks. Finally, the embedded data is extracted and the host image is recovered.

4. Experimental results

This section presents experimental results from the proposed scheme. The performance of the proposed scheme will be compared with those of the Sachnev et al. [12], Li et al. [20], Ou et al. [23], Peng et al. [24] and Wang et al. [25] schemes. For these experiments, eight gray-scale images served as test images. Except Barbara, all images are downloaded from the USC-SIPI image database, as depicted in Fig. 6.

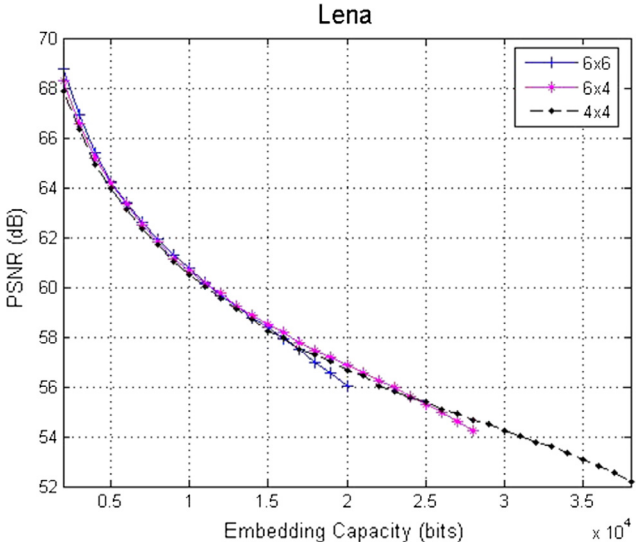

Fig. 7. Performance of the proposed scheme with different root block sizes for image Lena.

Table 3
Run times (unit: second) of the proposed scheme for test images.

Image	6 × 6	6 × 4	4 × 4	Total
Lena	2.27	3.72	6.48	12.47
Baboon	1.08	1.67	2.96	5.71
Airplane	2.33	4.16	7.94	14.43
Barbara	1.63	2.84	4.79	9.26
Boat	1.79	3.11	5.72	10.62
Lake	1.53	2.42	4.45	8.4
Peppers	2.02	3.3	6.99	12.31
Elaine	0.88	2.6	5.42	8.9
Average	1.69	2.98	5.59	10.26

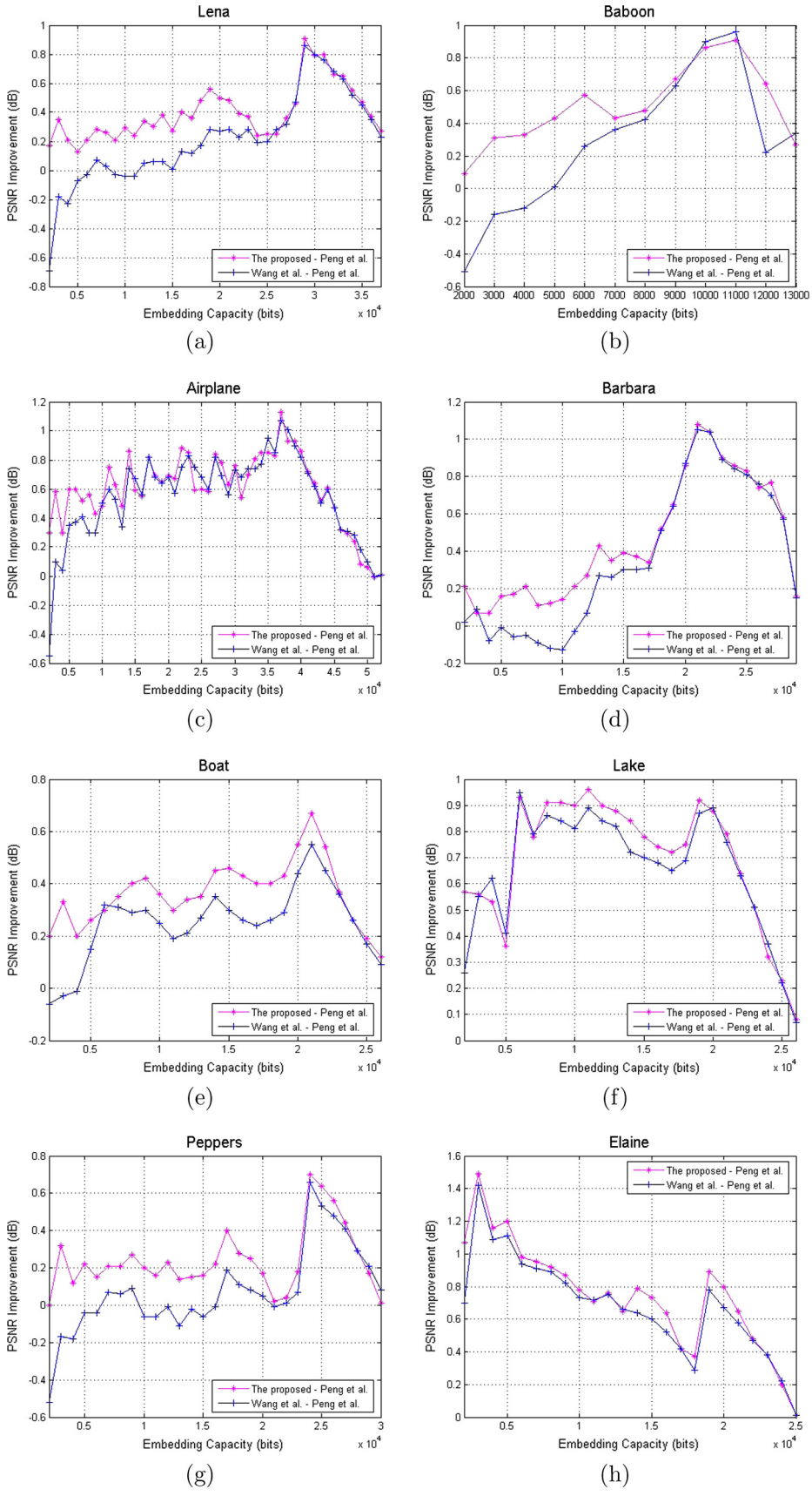

Fig. 8. Comparison of improvements over Peng et al.'s scheme obtained by the proposed scheme and Wang et al.'s scheme.

We first consider the impact of root block size to embedding performance. Referring to Fig. 7, it can be observed that the performance of various root block sizes mainly differ in maximum EC. For example, we can embed a maximum of 20,000 bits into image Lena when taking 6×6 root block size. In contrast, we can embed 28,000 bits when taking 6×4 root block size and 38,000 bits when taking 4×4 root block size. However, similar to Wang et al.'s scheme, the proposed scheme does not improve the maximum EC either. Taking into account the feature of PVO-based schemes that small block size leads to larger EC whereas lower PSNR, the maximum EC of the proposed scheme is almost the same as that of Peng et al.'s scheme using 2×2 blocks. In terms of PSNR, it is obvious that larger root block size produces slightly higher PSNR. 6×6 root block size improves 6×4 root block size by 0.47 dB at most when EC is smaller than 11,000 bits. Similarly, 6×4 root block size improves 4×4 root block size by 0.26 dB at most when EC is in range of [12000, 24000].

To achieve the best performance, the embedding procedure still has to be carried out for three root block sizes: $6 \times 6, 6 \times 4, 4 \times 4$. For our scheme, we vary EC from 2,000 bits to its maximum with a step of 1,000 bits. Table 3 shows the computation times for all test images, which are measured on an Intel Pentium CPU (3.0 GHz) windows 7 PC with 4.0 GB RAM. When taking 4×4 root block size, the proposed scheme spent the longest time on image Airplane (7.94 s) due to its smoothness. Considering a maximum EC of 52,000 bits, the embedding procedure has been carried out 51 times. Thus, the average time consumption for one loop embedding is about 0.15 s. In Wang et al.'s scheme, they proposed to attempt all combinations of $(T_2, T_1), -1 \leqslant T_2 \leqslant T_1 \leqslant 255$ in data embedding. As there could be $(1 + 257) * 257/2$ kinds of combination of (T_2, T_1) if without any filtering, such attempt will lead to incredible time consumption. Suppose the EC requirement is 2000 bits, the runtime of such attempt on image Lena reaches up to about 100 min. Based on this, it can be concluded that our scheme shows great superiority over Wang et al.'s in computational complexity.

Besides computational complexity, our scheme also achieves higher PSNR than that of Wang et al.'s for every EC. Based on the consideration that both of them announce to achieve superior performance to Peng et al.' work, here we compare their improvements in Fig. 8. Fig. 8 confirms the poor performance of Wang et al.'s scheme when EC is small. In contrast, our scheme improves the performance of Peng et al.'s for all ECs. Taking image Peppers as an example, the maximum improvement is 0.66 dB for embedding 24,000 bits of data in Wang et al.'s scheme. However, they even perform worse than Peng et al.'s before this. The proposed scheme obviously overcomes this deficiency by always achieving positive PSNR improvement. When EC is smaller than 24,000, there is no negative PSNR improvement and the average PSNR improvement is about 0.2 dB. For other test images, there is similarly no negative PSNR improvement either, which is the most important advantage of our scheme.

Referring to Fig. 8, it can also be seen that the proposed scheme improves Wang et al.'s by 0.86 dB, 0.6 dB, 0.85 dB, 0.27 dB, 0.36 dB, 0.31 dB, 0.52 dB, 0.37 dB at most on all test images respectively. It should be noted that none of them is obtained by using 4×4 root block size. Once the root block size is determined as 4×4, the additional 4×2 block will be the only difference between the proposed scheme and Wang et al.'s. In that case, the proposed scheme can only achieve slight superiority over Wang et al.'s. For example, 4×4 root block size is adopted on image Lena since EC is larger than 24,000 bits. From Fig. . 8(a) we can see that the corresponding superiority is not evident. Relatively evident superiority appears on textured images Boat, Lake and Elaine, on which 4x4 root block size is adopted since a very small EC (e.g., 6,000 bits for Boat, 3,000 bits for Lake and Elaine). On those test images, the proposed scheme improves Wang et al.'s by 0.17 dB, 0.12 dB, 0.15 dB respectively at most when taking 4×4 root block size.

To further verify the advantages of the proposed scheme, a comprehensive comparison was performed on 100 images of UCID database. Fig. 9(a) presents the comparison result, with EC varying from 2000 to 40,000 bits (only a few images can provide an EC as large as 40,000 bits), and the PSNR is the average value for each EC. Referring to Fig. 9(a), we can see that the proposed scheme achieves a much better average performance on a large image database on all ECs and the superiority of the proposed scheme becomes more obvious as EC decreases. Specially, when EC is smaller than 15,000 bits, the increase in PSNR over Wang et al.'s scheme is greater than 0.2 dB. When EC is smaller than 10,000 bits, the increase in PSNR over Wang et al.'s scheme is greater than 0.33 dB. Nevertheless, the comparison result is still relatively conservative due to the limited EC. Among the 100 randomly selected images with size of 512×384 or 384×512, the proposed scheme can achieve a maximum EC as large as 35,000 bits on only 34 ones.

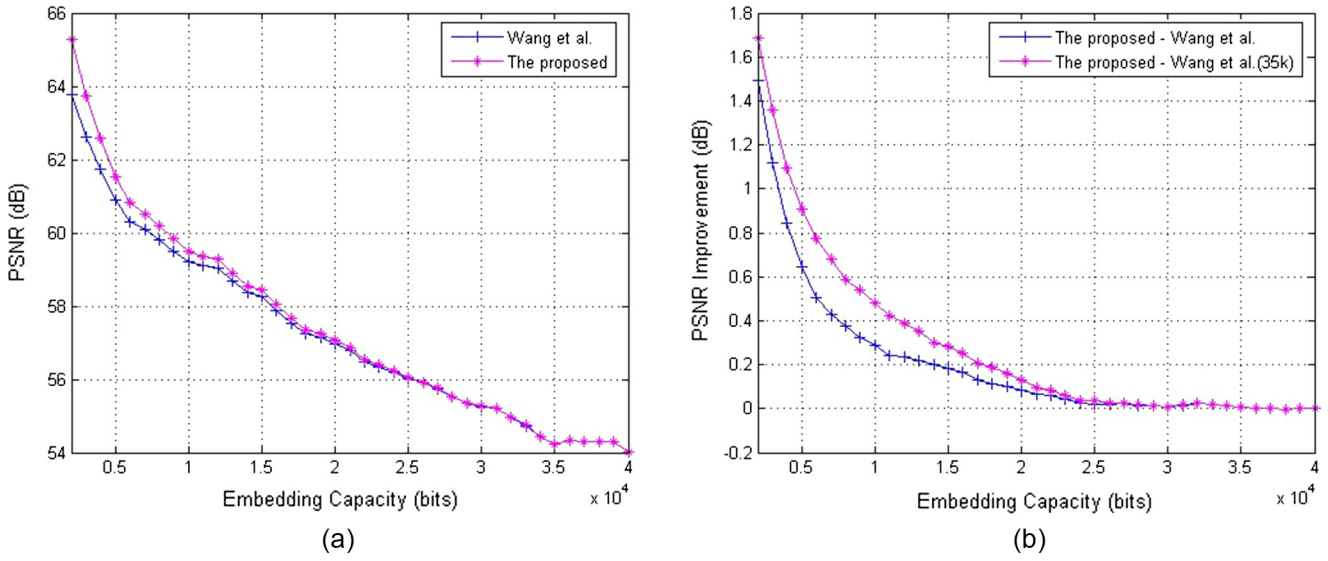

Fig. 9. Performance comparison on the UCID image database.

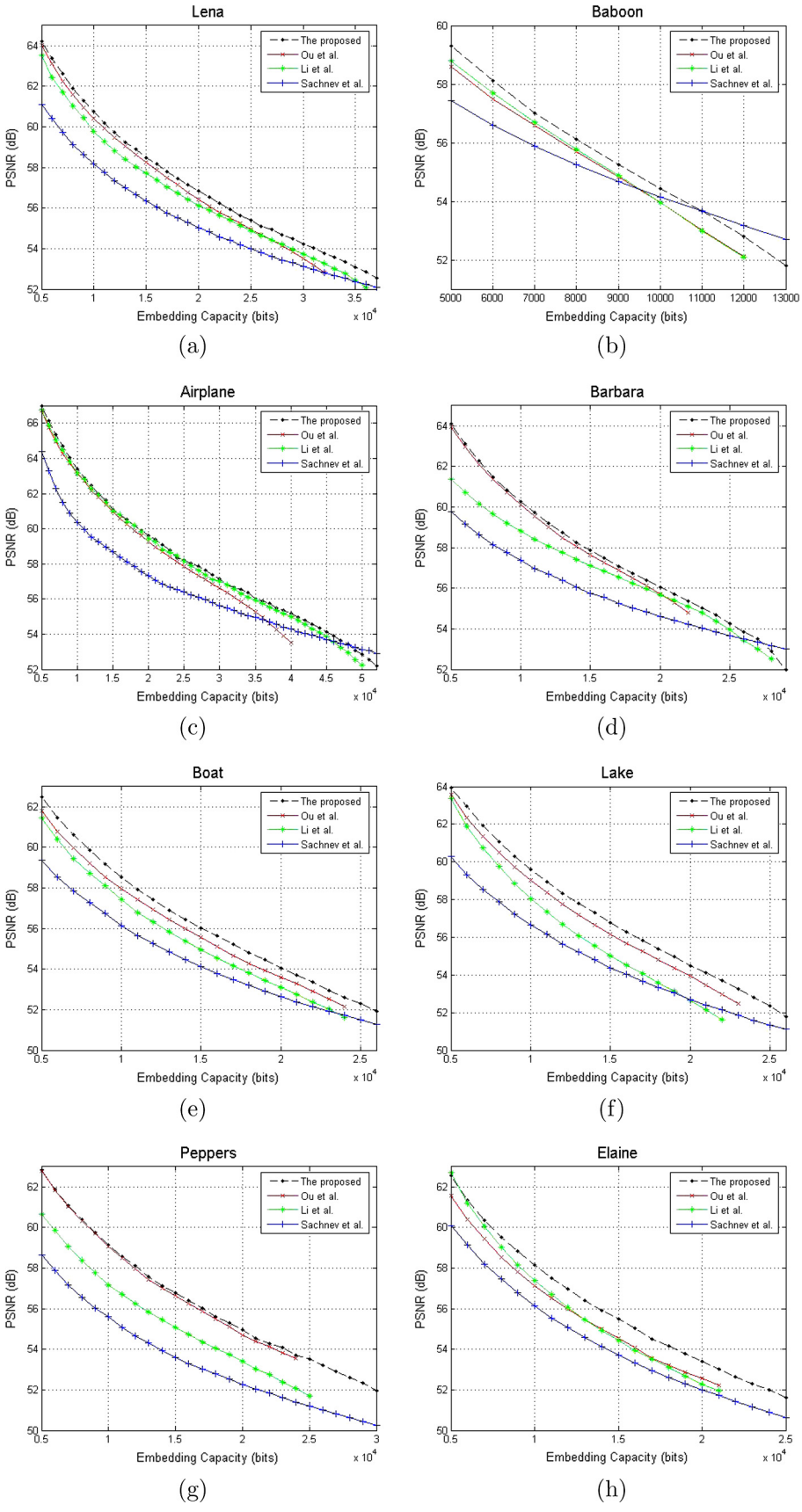

Fig. 10. Performance comparison between the Ou et al. [23], Li et al. [20], Sachnev et al. [12] and the proposed scheme.

Table 4
Comparison of PSNR (in dB) between the Li et al. [20], Sachnev et al. [12], Ou et al. [23] schemes and the proposed scheme for an EC of 10,000 bits.

Image	Li et al.	Sachnev et al.	Ou et al.	The proposed
Lena	59.76	59.19	60.42	60.78
Baboon	53.96	54.14	53.95	54.44
Airplane	63.18	60.33	63.14	63.38
Barbara	59.51	58.98	60.13	60.28
Boat	57.44	56.16	57.96	58.53
Lake	58.08	56.65	59.01	59.6
Peppers	57.11	55.48	59.07	59.15
Elaine	57.38	56.12	57.12	58.14
Average	58.3	57.13	58.85	59.29

Table 5
Comparison of PSNR (in dB) between the Li et al. [20], Sachnev et al. [12], Ou et al. [23] schemes and the proposed scheme for an EC of 20,000 bits.

Image	Li et al.	Sachnev et al.	Ou et al.	The proposed
Lena	56.12	55.02	56.41	56.86
Baboon	–	49.36	–	–
Airplane	59.42	57.28	59.26	59.59
Barbara	55.78	54.58	55.69	56.03
Boat	53.11	52.63	53.61	54.06
Lake	52.67	52.71	53.95	54.5
Peppers	53.41	52.31	54.7	54.95
Elaine	52.29	51.97	52.56	53.4
Average	54.69	53.79	55.17	55.63

Fig.9(b) compares the improvement over Wang et al.'s scheme obtained on those 34 images with the original one. Obviously the proposed scheme can achieve more desirable superiority.

Fig. 10 shows the comparison between the proposed scheme and other schemes [12,20,23]. Among them, Sachnev et al.'s scheme has been verified better than many PEE-based schemes. For this scheme, prediction error is obtained using rhombus prediction where the four neighboring pixels are used to predict the center one. Moreover, pixels with small local variance are preferentially utilized for embedding. Referring to Fig. 10, our scheme outperforms Sachnev et al.'s in most cases, except when EC approaches its maximum (e.g., 11,000 bits for Baboon, 48,000 bits for Airplane and 28,000 bits for Barbara). In this case, the advantage of multistage blocking is greatly weakened due to the extremely high proportion of minimum (2×2) blocks. This results in a more and more similar performance to Peng et al.'s scheme, as shown in Fig. 8. Hence, the reason for this is the same as with Peng et al.'s scheme: not only the advantage of pixel correlation can not be well exploited by taking 2×2 blocks, but also rough blocks are necessarily used to embed data. However, our scheme achieves evident superiority over Sachnev et al.'s for moderate EC. Referring to Table 4 and 5, our scheme improves Sachnev et al.'s by 2.16 dB on average for an EC of 10,000 bits, and 1.84 dB for an EC of 20,000 bits. It should be noted that among the proposed scheme and other schemes [12,20,23], only Sachnev et al.'s scheme can provide an EC as large as 20,000 bits on image Baboon.

Li et al.'s scheme [20] can be viewed as an extension of Lee et al.'s work [27] since it succeeded to turn part of errors from to-be-shifted to expandable. Fig. 10 shows that the proposed scheme significantly outperforms Li et al.'s for most test images. Specifically, the average increase in PSNR is far more than 1 dB for Lake and Peppers (e.g. 2.09 dB for Lake and 1.77 dB for Peppers) while the least increase appears on image Airplane (e.g. 0.19 dB for Airplane). It is reasonable because 4×4 root block size is adopted since an EC of 12,000 bits on image Airplane due to its smoothness. Hence, the proposed scheme cannot benefit a lot from multistage blocking. On the other hand, Li et al.'s pixel-pair-selection strategy shows high accuracy in smooth areas and therefore better performance can be achieved. Table 4 and 5 shows that the proposed scheme improves Li et al.'s by 0.99 dB on average for an EC of 10,000 bits, and 0.94 dB for an EC of 20,000 bits.

Ou et al.'s POV-k reversible embedding strategy [23] is an effective extension of conventional PVO. By taking $k \in \{1, 2\}$, an optimal combined embedding is proposed to fully exploit image redundancy. However, similar to Wang et al.'s, the high computational complexity is also the main weakness of this scheme. The runtime of the combined PVO-k with $k \in \{1, 2, 3\}$ reaches up to two minutes and such a time cost is unacceptable in practical application. Fig. 10 shows that the proposed scheme slightly outperforms Ou et al.'s. Specifically, the average increase in PSNR is 0.41 dB, 0.55 dB, 0.51 dB, 0.23 dB, 0.53 dB, 0.61 dB, 0.16 dB and 0.95 dB for all test images respectively. Table 4 and 5 shows that the proposed scheme improves Ou et al.'s by 0.44 dB on average for an EC of 10,000 bits, and 0.46 dB for an EC of 20,000 bits.

5. Conclusion

On the basis of Wang et al.'s work, an efficient and more comprehensive multistage blocking strategy is proposed in this paper. For pixel value ordering based RDH schemes, it can be noticed that the maximum modification is always 1 at most. This enables us to achieve prediction accuracy guided multistage blocking. Experimental results demonstrate that our scheme not only achieves great superiority over Wang et al.'s work in computational complexity, but also guarantees less embedding distortion for every given EC. Finally, it can be inferred that prediction accuracy matrix based multistage blocking can be further extended to combine more various-sized blocks. When combining n various-sized blocks, the $256 \times n$ sized prediction accuracy matrix will be the assurance of high efficiency.

Acknowledgments

This work is supported by the National Scientific Fund of China (Nos. 61170320, 81201763).

References

[1] M. Wu, H. Yu, B. Liu, Data hiding in image and video: part II-designs and applications, IEEE Trans. Image Process. 12 (6) (2003) 696–705.
[2] A. Khan, A. Siddiqa, S. Munib, S.A. Malik, A recent survey of reversible watermarking techniques, Inform. Sci. 279 (20) (2014) 251–272.
[3] J. Fridrich, M. Goljan, R. Du, Lossless data embedding-new paradigm in digital watermarking, EURASIP J. Appl. Signal Process 2002 (2) (2002) 185–196.
[4] M.U. Celik, G. Sharma, A.M. Tekalp, E. Saber, Lossless generalized-LSB data embedding, IEEE Trans. Image Process. 14 (2) (2005) 253–266.
[5] C.-C. Chang, C.-C. Lin, C.-S. Tseng, W.-L. Tai, Reversible hiding in DCT-based compressed images, Inform. Sci. 177 (13) (2007) 2768–2786.
[6] J. Tian, Reversible data embedding using a difference expansion, IEEE Trans. Circ. Syst. Video Technol. 13 (8) (2003) 890–896.
[7] A.M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, IEEE Trans. Image Process. 13 (8) (2004) 1147–1156.
[8] S. Weng, Y. Zhao, J.S. Pan, R. Ni, Reversible watermarking based on invariability and adjustment on pixel pairs, IEEE Signal Process. Lett. 15 (2008) 721–724.
[9] D. Coltuc, Low distortion transform for reversible watermarking, IEEE Trans. Image Process. 21 (1) (2012) 412–417.
[10] D.M. Thodi, J.J. Rodriguez, Expansion embedding techniques for reversible watermarking, IEEE Trans. Image Process. 16 (3) (2007) 721–730.
[11] X. Gao, L. An, Y. Yuan, D. Tao, X. Li, Lossless data embedding using generalized statistical quantity histogram, IEEE Trans. Circ. Syst. Video Technol. 21 (8) (2011) 1061–1070.
[12] V. Sachnev, H.J. Kim, J. Nam, S. Suresh, Y.Q. Shi, Reversible watermarking algorithm using sorting and prediction, IEEE Trans. Circ. Syst. Video Technol. 19 (7) (2009) 989–999.
[13] L. Luo, Z. Chen, M. Chen, X. Zeng, Z. Xiong, Reversible image watermarking using interpolation technique, IEEE Trans. Inform. Forensics Security 5 (1) (2010) 187–193.
[14] Q. Pei, X. Wang, Y. Li, H. Li, Adaptive reversible watermarking with improved embedding capacity, J. Syst. Softw. 86 (11) (2013) 2841–2848.
[15] B. Ou, X. Li, Y. Zhao, R. Ni, Reversible data hiding based on PDE predictor, J. Syst. Softw. 86 (10) (2013) 2700–2709.
[16] Y. Hu, H.K. Lee, J. Li, DE-based reversible data hiding with improved overflow location map, IEEE Trans. Circ. Syst. Video Technol. 19 (2) (2009) 250–260.
[17] D. Coltuc, Improved embedding for prediction-based reversible watermarking, IEEE Trans. Inform. Forensics Security 6 (3) (2011) 873–882.
[18] X. Li, B. Yang, T. Zeng, Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection, IEEE Trans. Image Process. 20 (12) (2011) 3524–3533.
[19] W. Hong, Adaptive reversible data hiding method based on error energy control and histogram shifting, Opt. Commun. 285 (2) (2012) 101–108.
[20] X. Li, W. Zhang, X. Gui, B. Yang, A novel reversible data hiding scheme based on two-dimensional difference-histogram modification, IEEE Trans. Inform. Forensics Security 8 (7) (2013) 1091–1100.
[21] B. Ou, X. Li, Y. Zhao, R. Ni, Y.-Q. Shi, Pairwise prediction-error expansion for efficient reversible data hiding, IEEE Trans. Image Process. 22 (12) (2013) 5010–5021.
[22] X. Li, J. Li, B. Li, B. Yang, High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion, Signal Process. 93 (1) (2013) 198–205.
[23] B. Ou, X. Li, Y. Zhao, R. Ni, Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion, Signal Process.: Image Commun. 29 (7) (2014) 760–772.
[24] F. Peng, X. Li, B. Yang, Improved pvo-based reversible data hiding, Digit. Signal Process. 25 (2014) 255–265.
[25] X. Wang, J. Ding, Q. Pei, A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition, Inform. Sci. 310 (2015) 16–35.
[26] Z. Ni, Y.-Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Trans. Circ. Syst. Video Technol. 16 (3) (2006) 354–362.
[27] S.K. Lee, Y.H. Suh, Y.S. Ho, Reversible image authentication based on watermarking, in: Proc. IEEE ICME, 2006, pp. 1321–1324.
[28] P. Tsai, Y.C. Hu, H.L. Yeh, Reversible image hiding scheme using predictive coding and histogram shifting, Signal Process. 89 (6) (2009) 1129–1143.
[29] Y.C. Li, C.M. Yeh, C.C. Chang, Data hiding based on the similarity between neighboring pixels with reversibility, Digit. Signal Process. 20 (4) (2010) 1116–1128.
[30] Z.-Z. Zhao, H. Luo, Z.-M. Lu, J. -S Pan, Reversible data hiding based on multilevel histogram modification and sequential recovery, J. Electron. Commun. 65 (2011) 814–826.
[31] H. Luo, F.-X. Yu, H. Chen, Z.-L. Huang, H. Li, P.-H. Wang, Reversible data hiding based on block median preservation, Inform. Sci. 181 (2) (2011) 308–328.
[32] Z.-B. Pan, S. Hu, X.-X. Ma, L. F W, Reversible data hiding based on local histogram shifting with multilayer embedding, J. Vis. Commun. Image R. 31 (2015) 64–74.
[33] D.-S. Fu, Z.-J. Jing, S. -G Zhao, J. Fan, Reversible data hiding based on prediction-error histogram shifting and EMD mechanism, J. Electron. Commun. 68 (2014) 933–943.