J. Vis. Commun. Image R. 48 (2017) 317–328

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Optimal PPVO-based reversible data hiding ☆

Shaowei Weng [a,*], Guohao Zhang [a], Jeng-Shyang Pan [b], Zhili Zhou [c]

[a] School of Information Engineering, Guangdong University of Technology, PR China
[b] Fujian Provincial Key Laboratory of Data Mining and Applications, Funjian University of Technology, Fujian, PR China
[c] School of Computer and Software, NanJing University of Information Science & Technology, PR China

ARTICLE INFO

Article history:
Received 19 October 2016
Revised 4 April 2017
Accepted 12 May 2017
Available online 23 May 2017

Keywords:
Reversible data hiding
Adaptive prediction pattern
Adaptive pixel-modification strategy
Two-stage embedding
PPVO

ABSTRACT

The optimal PPVO-based reversible data hiding (RDH) including adaptive prediction pattern and optimal bin selection is proposed in this paper. By reasonably designing the prediction pattern, each to-be-embedded pixel can be predicted by the n ($n \in \{4, \ldots, 13\}$) neighbors surrounding it. The larger n, the more accurate the prediction, and the higher embedding performance is achieved at low embedding capacity (EC) and vice versa. Since the neighbors surrounding each to-be-embedded pixel can be used for measuring the local complexity, the smoothness classification is more accurate, and furthermore, the prediction performance is increased. Two-stage embedding is exploited in this paper so as to ensure the implementation of full-enclosed-based prediction. Instead of treating separately two stages like in existing methods, two stages are treated jointly to optimize the selection of embedding bins at a given EC. In this way, the optimal embedding bins which can achieve as high visual quality as possible is obtained. Extensive experiments verify the proposed method is effective.

1. Introduction

With the fast development of computer and internet technologies, multimedia data can be accessed, exchanged and copied over the internet at any time and anywhere. Meanwhile, this also makes multimedia data transmitted under threat. Therefore, how to deliver multimedia data safely has been a challenging issue in the information security field, and the main protection approaches including encryption, fingerprinting [1], data hiding [2,3] and so on have been widely studied. Among them, data hiding is a technique that can hide data in a imperceptible manner into multimedia data for copyright protection, content authentication, and secret communications. As we know, data hiding has two related disciplines, i.e., steganalysis [4,5] and forensics [6–8]. Traditional data hiding techniques usually introduce irreversible distortion to multimedia data, and therefore, they cannot be used in some applications requiring no permanent distortion to the original cover image, such as medical and military image processing. To this end, the data hiding with reversibility (RDH) is proposed to provide a solution for these applications. Reversibility means that RDH can losslessly retrieve the original cover image after the embedded data is successfully extracted from the marked image. To ensure reversibility, any data hiding technique which may lead to irreversible distortion cannot be used in RDH methods. Therefore, some methods with reversibility, e.g., lossless compression, difference expansion and histogram shifting, have already been developed for RDH.

The key idea of the lossless-compression-based RDH methods is to generate vacant space for data embedding by losslessly compressing a part of the host images [9–11]. However, the obtained EC is very limited due to the poor compression ratio. Afterwards, Tian proposed the difference expansion (DE) technique, so that every two neighboring pixels can carry at most 1 data bit, i.e., the obtained embedding rate is close to 0.5 bpp (bit per pixel) [12]. Specifically, in Tian's method, the Haar integer transform is used for two neighboring pixels to create a difference value and an average value. Then, this difference value is expanded (i.e., multiplied by 2) to create a vacant least significant bit (LSB), and 1 data bit is embedded into this vacant LSB. Inspired by Haar integer transform in Tian's method, the improved ones have been proposed in [13–18]. Besides, another typical RDH technique is histogram shifting (HS), which embeds data bits into the pixels with the most frequent occurrence in histogram while shift the other pixels for the sake of reversibility [19–21]. In fact, DE can be viewed as a prediction process, in which each pixel to be embedded is predicted by its context only containing one neighbor. Considering the low prediction performance in Tian's method, Thodi

☆ This paper has been recommended for acceptance by Zicheng Liu.
* Corresponding author.
E-mail addresses: wswweiwei@126.com (S. Weng), jspan@cc.kuas.edu.tw (J.-S. Pan), zhou_zhili@163.com (Z. Zhou).

http://dx.doi.org/10.1016/j.jvcir.2017.05.005
1047-3203/© 2017 Elsevier Inc. All rights reserved.

et al. predict a pixel using its three neighbors, and then expand the prediction-error between the prediction and its original value so as to achieve 1-bit data embedding. In addition, HS is incorporated into Thodi et al.'s method so as to efficiently compress the location map, which is used to record the locations of prediction-errors introducing possibly overflow or underflow. The prediction-error expansion (PEE) has been well studied in recent works [22–39]. Among them, Sachnev et al. employed the rhombus predictor (i.e., a full-enclosed predictor), instead of the half-enclosed predictors such as MED predictor [40], to get a sharp PEE histogram. In addition, the sorting technique is employed to priorly select the prediction-errors located in smooth regions for data embedding. By means of the improved prediction performance and the sorting technique, Sachnev et al.'s method achieves good embedding performance, compared with its earlier methods. Dragoi and Coltuc have proposed the use of local prediction based reversible watermarking. In their method, for each pixel, the least square predictor in a square block centered on this pixel is computed. The experimental results show that the local prediction based schemes clearly outperform their global least square and fixed prediction based counterparts (e.g., the median edge detector (MED) [41], gradient-adjusted predictor (GAP) [42] or the simple rhombus neighborhood [23]).

Recently, a new research branch of RDH was proposed by Li et al. [43]. This research branch focuses on achieving very high visual quality when the required EC is not high, e.g., <0.14 bpp for Lena. Motivated by Li et al.'s method, deep research has been carried out to increase image fidelity as much as possible under a limited EC [44–54]. Li et al. proposed a RDH method which combines pixel-value-ordering (PVO) and PEE [43]. In their method, PVO can be deemed as a predictor, in which the maximum (or minimum) is predicted by the second largest (or smallest) pixel in a block. In this way, a histogram of absolute values of prediction errors is generated. Compared with some other predictors, such as the rhombus predictor [23], PVO can produce more accurate prediction results under low EC. HS is utilized to achieve data embedding. Specially, the peak bin (i.e., bin 1) is embedded with 1 bit data, while the bins larger than 1 are shifted. As shown in [43], Li et al.'s method achieve better performance than the previous works [23,27] when the payload is low. Hong et al. have proposed an adaptive method to increase the number of embeddable spaces by referencing a dual binary tree, and meanwhile employ the MED predictor and an error energy estimator to decrease the number of non-embeddable prediction-errors [55]. Ou et al. have proposed a pairwise PEE method [45], in which two adjacent prediction-errors are treated jointly to embed at most $log_2 3$ bits. It is known that two bits of '11' will introduce larger modifications to two adjacent prediction-errors than the other combinations of two bits, i.e., '00', '01' and '10', when these two prediction-errors can be embedded with two bits of data. In order to reduce the amount of image modifications for a given capacity, Ou et al. exclude two bits of '11' from data embedding by exploiting the pairwise PEE. In this way, Ou et al. achieve a significant performance increase. Li et al. have proposed a new RDH method based on PEE for multiple histograms. Specifically, they partition the prediction-error histogram into multiple sub-histograms, and then treat the multiple sub-histograms as an unit to search the optimal bins for data embedding which can achieve the highest visual quality at a given capacity [52]. Dragoi and Coltuc have proposed an adaptive paring reversible watermarking method [54]. In contrast to the fixed pixel pairing used in Ou et al.'s method [45], the adaptive pixel pairing can combine flexibly pixels with similar local complexity into pairs. In this way, their method can increase the number of pairs where both pixels are embedded and decrease the number of shifted pixels. Experimental results also demonstrate their method can achieve a better embedding performance than the state-of-the-art low embedding bit-rate schemes proposed so far.

According the description above, the bin 0 (i.e., the prediction-errors equal to 0) is excluded from data embedding in Li et al.'s method [43]. In fact, the bin 0 is very suitable to carry data. Therefore, in order to exploit the bin 0 for data embedding, Peng et al. proposed an improved PVO (IPVO) method by considering the pixel locations of the maximum and second largest value (or, the minimum and second smallest value) [46]. In this way, the bin 0 can also be used for data embedding in addition to bin 1. Therefore, Peng et al.'s work [46] achieves better performance, compared with Li et al.'s method. In order to fully exploit the blocks of bin 0, Ou et al.'s method has proposed another improvement of PVO called PVO-K, where K is the number of the maximum-valued (or minimum-valued) pixels [47]. In their method, K maximum-valued (or minimum-valued) pixels are modified simultaneously so that the blocks of bin 0 can be expanded to carry data. In fact, PVO-1 is PVO. In order to keep a balance between capacity and distortion, K cannot be set too large. Experiment results in [47] show $K = 2$ can achieve better performance. In a word, PVO, IPVO and PVO-K perform predictions according to a block-by-block manner. In these three methods, since only a small portion of pixels in a block is used for data embedding, a block can only be embedded with at most 2 bits. Inspired by the idea that each pixel can be used for data embedding, Qu et al. has proposed a novel pixel-based PVO (PPVO). In their method, each pixel is predicted using its sorted context pixels [48]. Specifically, for a pixel x to be predicted, its context pixels composed of n right and down neighbors are sorted in ascending order, and then the largest or smallest value in the sorted array is utilized to predict x when all context pixels are either larger or smaller than x. In this way, almost all the pixels can be predicted, and hence, the number of prediction-errors is largely increased. Correspondingly, the number of prediction-errors capable of carrying data bits is increased. The EC of PPVO is larger than that of PVO [43], especially for relatively smooth images.

However, PPVO only utilizes the right and down neighbors of x to predict x, instead of all the neighbors surrounding x. Hence, the prediction performance can be further increased. In PPVO, bin 0, the peak of histogram, is used for data embedding. Since bin 0 has the largest number of prediction-errors, the obtained EC is higher than that provided by other bins. In order to embed data bits into bin 0, the prediction-errors between the peak bin and zero bin (which does not have prediction-errors) need to be shifted by one towards the zero bin to generate an empty bin. Although peak bin 0 can achieve higher EC compared with other bins, a large number of prediction-errors between the peak bin and zero bin need to be modified, and thus the corresponding distortion is also high. Therefore, we conclude that the capacity-distortion performance may not be optimal when bin 0 is used for data embedding. With the above consideration, we argue that PPVO can be further improved.

In the proposed method, by designing reasonably a prediction pattern, each to-be-embedded pixel can be predicted by its n ($n \in \{4, \ldots, 13\}$) pixels surrounding it. It is known that full-enclosed-based prediction is better in accuracy than half-enclosed-based prediction utilized in Qu et al.'s method, and therefore, the prediction performance is largely increased. For each pixel to be embedded, its n neighbors surrounding it must be kept unaltered in the prediction process so as to ensure reversibility. Based on this reason, the host image must be partitioned into two non-overlapped pixel sets A and B. First, one of two sets (e.g., B) is selected for predicting A in the first stage. In the whole embedding process, no modifications are allowed to the pixels in B. Then, in the second stage, the already modified pixels in A are used for predicting B. In this way, almost all pixels can be predicted. This is also the reason that the proposed method must select two-stage embedding.

It is known that, the shape of PEH resembles a Laplacian distribution defined on $(-\infty, +\infty)$ centered at 0. The closer to peak bin, the larger the number of the occurrences of prediction-error is. Although the peak bin can provide higher EC than any other ones, the introduced distortion is also larger due to that large amount of pixels need to be shifted to generate an empty bin. Hence, the peak bin may not be optimal one for capacity-distortion performance. Here, inspired by Hwang et al.'s opinion [56], the exhaustive search is performed to find the optimal embedding bins which can achieve the given EC while maintain the lowest distortion. Experimental results demonstrate that both the capacity and the marked image quality can be improved compared with the methods [43,46–48].

2. Related works

In this section, Qu et al.'s method [48] and Hwang et al.'s method [56] are briefly introduced.

2.1. Qu et al.'s method

In Qu et al.'s method, referring to Fig. 1, the context pixels containing n $(n \in \{4, \ldots, 15\})$ right and down pixels are used to predict a pixel, denoted as p. For simplicity, the minimum and maximum values in C are denoted as $c_{\min}$ and $c_{\max}$, respectively. For the case of $c_{\min} < c_{\max}$, by comparing the pixel p with $c_{\min}$ and $c_{\max}$, respectively, the prediction $\hat{p}$ is obtained as

$$\hat{p} = \begin{cases} c_{\max} & p \geqslant c_{\max}, \\ c_{\min} & p \leqslant c_{\min}, \\ \Phi & c_{\min} < p < c_{\max}, \end{cases} \tag{1}$$

where Φ indicates that the pixel cannot be predicted and will be skipped during data embedding. For the case of $c_{\min} = c_{\max}$, the prediction $\hat{p}$ is calculated as

$$\hat{p} = \begin{cases} c_{\max} & p \geqslant c_{\max}, \\ c_{\max} - 1 & p \leqslant c_{\max} - 1. \end{cases} \tag{2}$$

The marked prediction-error P'_e is obtained as

$$P'_e = \begin{cases} P_e + 1 & p > c_{\max}, \\ P_e + b & p = c_{\max}, \\ P_e - b & p = c_{\min}, \\ P_e - 1 & p < c_{\min}, \end{cases} \tag{3}$$

where b is 1 data bit to be embedded.

Considering that $p' = \hat{p} + P_e$, the histogram modification can also be interpreted in terms of pixel value. The marked pixel value p' is obtained as

$$p' = \begin{cases} p + 1 & p > c_{\max}, \\ p + b & p = c_{\max}, \\ p - b & p = c_{\min}, \\ p - 1 & p < c_{\min}, \\ p & c_{\min} < p < c_{\max}. \end{cases} \tag{4}$$

p	c_1	c_4	c_9
c_2	c_3	c_6	c_{11}
c_5	c_7	c_8	c_{13}
c_{10}	c_{12}	c_{14}	c_{15}

Fig. 1. p is the current pixel, c_i $(i \in \{1, 2, \ldots, 14, 15\})$ is right and down neighbors of p.

2.2. Hwang et al.'s method [50]

Traditional HS method uses T_n and T_p to control the embedding (i.e., expanding) regions and the shifting region, where T_n and T_p are threshold values for both positive and negative sides. In Hwang et al.'s method, the histogram pair shifting (HPS) method uses T_n2, T_{n1}, T_{p1}, and T_{p2} for its embedding algorithm. The threshold values meet the following condition: $T_{n2} \leqslant T_{n1} \leqslant T_{p1} \leqslant T_{p2}$. The HPS embedding algorithm modifies the predicted error as follows:

$$d' = \begin{cases} d + N + 1 & d < T_{n2}, \\ 2(d - T_{n1}) + T_{n1} - 1 + b & T_{n2} \leqslant d \leqslant T_{n1}, \\ d & T_{n1} < d < T_{p1}, \\ 2(d - T_{p1}) + T_{p1} + b & T_{p1} \leqslant d \leqslant T_{p2}, \\ d + p + 1 & d < T_{p2}, \end{cases} \tag{5}$$

where d is a pixel to be embedded, d' is its modified value, $N = T_{n2} - T_{n1}, P = T_{p2} - T_{p1}$. The decoder recovers the original predicted error and extracts the embedded data b from d as follows:

$$d = \begin{cases} d' - N + 1 & d' < T_{n1} + 2N - 1 \\ d' - \lfloor (d' - T_{n1})/2 \rfloor & T_{n2} + 2N - 1 \leqslant d' \leqslant T_{n1} \\ d' & T_{n1} < d' < T_{p1} \\ d' - \lfloor (d' - T_{p1})/2 \rfloor & T_{p1} \leqslant d' \leqslant T_{p1} + 2P + 1 \\ d' - p - 1 & T_{p1} + 2p + 1 < d' \end{cases} \tag{6}$$

A simple example for comparing the HS method with HPS method is illustrated in Fig. 2. From Fig. 2, one can observe that the peak point (i.e., bin 0) is selected for carrying 12 bits in HS method. All bins in the right of the peak are moved to the right by one position, and therefore, the number of the shifted prediction-errors is 27 and the corresponding distortion (denoted as MSE_{HS}) in terms of mean square error (MSE) can be formulated by

$$MSE_{HS} = 6 \times 0^2 + 6 \times 1^2 + 8 \times 1^2 + 6 \times 1^2 + 4 \times 1^2 + 2 \times 1^2 + 1 \times 1^2 = 27$$

From Fig. 2, the peak bin is not a only one which can provide 12 bits. In fact, besides bin 0, the combinations of bins can also provide 12 bits, e.g., each of bins 2 and −2 can be embedded with 6 bits. When bin 2 is used for embedding data, all bins in the right of bin 2 are shifted outwards to generate an empty bin, and the corresponding $MSE_{positive}$ is equal to 10. When bin −2 is used for embedding data, the corresponding $MSE_{negative}$ is 9. The accumulated MSE in the positive side and negative side is 19 (i.e., 10 and 9, respectively). From this example, one can see that the embedding bins in HPS are dynamically selected to optimize the embedding performance.

3. The proposed method

HS is a RDH technique to embed data into the peak bins (i.e., the most frequent pixel values in an image) while shift some other bins to ensure reversibility. The EC is determined by the number of the most frequent pixels. The shifted bins contribute no EC but decrease the image quality. Therefore, it is very desirable to maximize the number of the most frequent pixels while reduce the number of shifted pixels as much as possible. Since the bin 0 (i.e., the bin with the prediction errors equal to 0) is usually the histogram peak, PPVO takes this bin for expansion embedding, whereas the bins with the prediction-errors larger than 0 are taken as shifted ones. Although PPVO achieve the maximum EC for a given histogram, the induced distortion is also the highest because the number of shifted pixels is the largest. Therefore, we conclude that PPVO may not produce the optimal embedding performance.

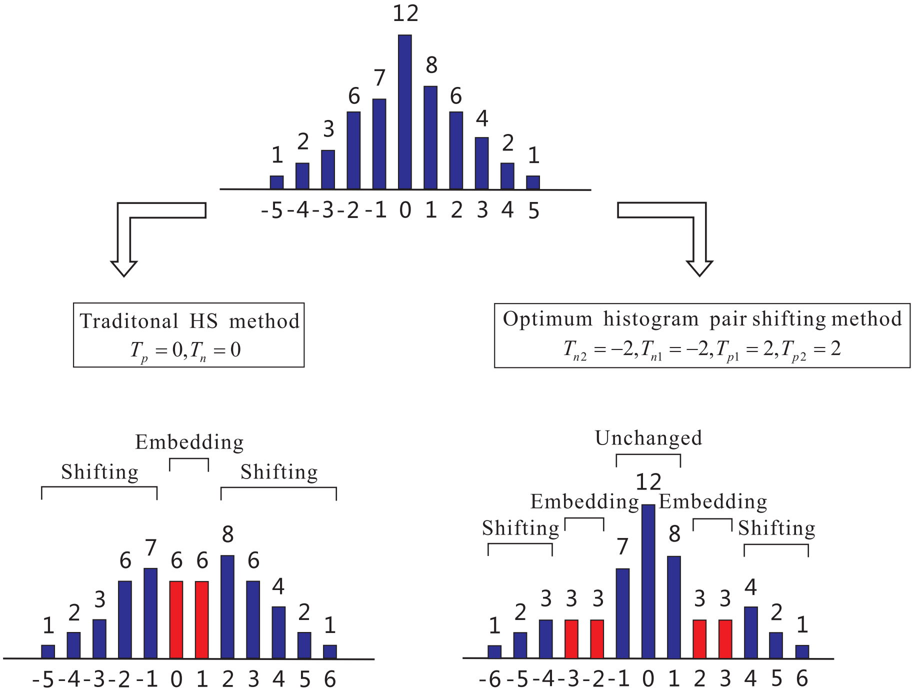


Fig. 2. Optimum histogram pair shifting method.

In this section, we propose an optimal PPVO method. The advantage of our proposed method lies in the following two aspects. Firstly, for a pixel, we utilize the n ($n \in \{4, \ldots, 13\}$) neighbors surrounding it to evaluate its local complexity. The more accurate the evaluation is, the lower the induced distortion is, and the higher the obtained EC is. It is known that the local complexity can be estimated more accurately using the n ($n \in \{4, \ldots, 13\}$) neighbors surrounding each to-be-embedded pixel, compared with the half-enclosed ones used in PPVO. Secondly, two-stage embedding is composed of two embedding stages, which are treated jointly to achieve a given EC. And, an embedding bin in each stage is selected for expansion embedding. Thus, there may exist many combinations of embedding bins for a given EC. To this end, exhaustive search is utilized to search the optimal combination of embedding bins which can achieve the highest PSNR value at a given EC. The detailed techniques used in our proposed method are given in the following subsections.

3.1. Implementation details

Previous RDH works have shown that the full-enclosed-based prediction is better in accuracy than half-enclosed-based prediction. Thus, we design a prediction pattern similar to rhombus prediction [23] and two-stage embedding in our implementation. In our prediction pattern, for each pixel to be embedded, its n ($n \in \{1, 2, \ldots, 13\}$) surrounding pixels are used to predict it. Unlike Sachnev et al.'s method whose context pixels are fixed, our context pixels can be adaptively changed by tuning flexibly the value of n. The larger n is, the more accurate the prediction performance is, the smaller the obtained capacity is, and vice versa.

Given a $W \times H$-sized image I, it is partitioned into two non-overlapped sets A and B, which are denoted as "black" and "blank" in Fig. 3, respectively. One half of the secret message will be embedded into A and the rest half will be embedded into B. In this case, two-stage embedding need to be processed to cover the whole image. Specifically, B is first utilized to predict A in the first stage, and then B is predicted by the already modified pixels in A in the second stage. In the decoding phase, we first extract the embedded message and realize image recovery for pixels in B, and then, extract the embedded message and realize image recovery for pixels in A. Since the two stages are processed similarly, we only take the first stage for illustration.

3.1.1. Smoothness classification by pixel neighborhood

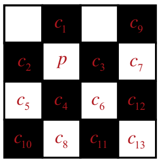
For any p, if it has a surrounding n-pixel neighborhood (refer to Fig. 3), i.e., $\{c_1, \ldots, c_n\}$ ($n \in \{4, \ldots, 13\}$), then this neighborhood constitute a new set, denoted by I_{LSE}. Note that the set I_{LSE} must contain four basic pixels, i.e., c_1, c_2, c_3 and c_4, and therefore, n must be larger than or equal to 4.

The local smoothness estimator is defined as the invariance, denoted by Δ, of all the pixels in I_{LSE}. Δ is used to estimate whether p is located in a smooth region or not. Δ is obtained via

$$\Delta = \sqrt{\frac{\sum_{i \in \{1,\ldots,n\}} (c_i - u_{LSE})^2}{n}}, \tag{7}$$

where μ_{LSE} is the mean value of set I_{LSE}. When $\Delta < vT, p$ is regarded as having a strong relation with its set I_{LSE}, and therefore, it is classified to be within a smooth region, where vT is a predefined threshold which is used for distinguishing which classification p belongs to. Otherwise, p is estimated in a complex region.

All the pixels in A are divided into two parts (S_P and C_P) according to the local complexity. C_P contains all pixels located in complex regions. In general, since larger prediction-errors are usually obtained in complex regions, these regions often embed less but distort more. Based on this reason, C_P is not used for data embed-



	c_1		c_9
c_2	p	c_3	c_7
c_5	c_4	c_6	c_{12}
c_{10}	c_8	c_{11}	c_{13}

Fig. 3. p is the current pixel, c_i ($i \in \{1, 2, \ldots, 13\}$) is the pixels surrounding p.

ding. S_P contains the remaining pixels, i.e., all pixels in smooth regions.

3.1.2. Data embedding for pixels in S_P

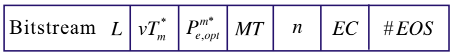
For simplicity, $c_{\min}$ and $c_{\max}$ are still used to denote the maximum and minimum of the set I_{LSE}, respectively. Tables 1–3 are utilized to clearly illustrate how to obtain the predicted value $\hat{p}$, the prediction-error P_e, the marked prediction-error P'_e and the marked value p' of each to-be-embedded p in different cases.

From Table 1, one can observe that when $p \leqslant c_{\min}, P_e = c_{\min} - p$. The reason behind this is to ensure that P_e is non-negative ($P_e \geqslant 0$), and thus, there is only one optimal embedding bin in the positive side which needs to be determined. More importantly, the time cost is reduced. Here, we suppose that the optimal embedding bin is denoted as $p^*_{e,opt}$ for a single stage. Table 2 is used to illustrate that the detailed modification process is performed on p (or P_e) in the case of $C_{\max} = C_{\min}$. From Table 2, it is observed that P_e (i.e., $P_e > P^*_{e,opt}$) is shifted outwards to create vacancies so that data bits are embedded into $P^*_{e,opt}$. This is similar for the case of $C_{\max} \neq C_{\min}$ (refer to Table 3).

3.2. Algorithm description

Some overhead information needs to be recorded before data embedding for reversibility. For the purpose of blind restoration on the decoding side, these overhead information also needs to be embedded into the host image together with the payload. For each single stage, the corresponding overhead information includes five parts: vT_m ($m \in \{1,2\}$), the compressed location map which is utilized to solve potential overflow/underflow problem, the optimal embedding bin $P^{m*}_{e,opt}$ ($m \in \{1,2\}$), the number of stage MT (MT which is generated to distinguish which stages are used for data embedding, and $MT \in \{0,1\}$, where $MT = 0$ means that the required EC can be achieved by only performing the first stage, while $MT = 1$ means that two stages must be performed together so as to obtain the required EC), end of symbol (EOS).

Since two sets (i.e., A and B) are processed in a similar way, we only take A for illustration. For any p, if its marked value p' exceeds the range of $[0, 255]$, i.e., the overflow/unerflow occurs, and then p cannot be used in the data embedding for reversibility. Therefore, we need a location map to determine the overflow and underflow locations. In order to create the map, we define a set D_A to contain all the pixels without overflows or underflows, where $D_A = \{p \in A : 0 \leqslant p' \leqslant 255\}$, and N is the number of pixels in A. According to the local complexity, D_A is further partitioned into two sub-sets, i.e., S_{PA} and C_{PA}, where $S_{PA} = \{p \in D_A : \Delta \leqslant vT_1\}$ and $C_{PA} = \{p \in A : \Delta > vT_1\}$. Clearly, there is no overflow/underflow when the pixels of S_{PA} are used for carrying data. The pixels with $\Delta \leqslant vT_1$ can cause overflow or underflow in the data embedding. Hence, we need another set O_{PA} to contain all these pixels, i.e., $O_{PA} = \{p \notin D_A : \Delta \leqslant vT_1\}$. A location map is obtained in which $p \in S_{PA}$ are marked by '1' while $p \in O_{PA}$ are marked by '0'. This map is compressed losslessly by an arithmetic encoder and the resulting binary sequence is denoted as $\mathcal{L}$, and we suppose that its length is L_S. Here, arithmetic coding is utilized for lossless compressing. The overhead information formats for the first and second stage embedding is shown in Fig. 4, where $m = 1$ represents the first stage while $m = 2$ stands for the second one. Because the first and second stages may not be fully embedded, we need an extra information EC to illustrate the payload size in each stage, which consists of 18 bits (representing EC in each single stage, specifically, $2^{18} = 512 \times 512$). Both of the first and second stages includes the compressed location map (L_S bits), vT_m (8 bits), $p^{m*}_{e,opt}$ (8 bits), MT (1 bit), n (4 bits), EC (18 bits), EOS (8 bits). Hence, for the first and second stages, the overhead information is equal to L_S plus 47 bits, i.e., $L_{\sum} = L_S + 47$, where $L_{\sum}$ is the size of the overhead information. In the experiments, if two-stage embedding cannot achieve the required capacity, the third-stage embedding is combined into two-stage embedding to achieve the required capacity. Correspondingly, the size of MT is increased to 2 bits, where '00' denotes the first-stage embedding, '01' denotes the second-stage embedding, and '10' denotes the third-stage embedding. Similarly, $m \in \{1,2,3\}$. After performing two-stage embedding, the already embedded image is repartitioned into two set A and B. Similar to the first-stage embedding, all the pixels with



Bitstream L	vT^*_m	$P^{m*}_{e,opt}$	MT	n	EC	$\#EOS$

Fig. 4. Overhead information formats in an embedding stage.

Table 3
The data embedding process of p (or P_e) for the case of $c_{\max} = c_{\min}$.

	$c_{\max} = c_{\min}$					
	$p > c_{\max}$			$p \leqslant c_{\min}$		
	$P_e = P^*_{e,opt}$	$P_e > P^*_{e,opt}$	$P_e < P^*_{e,opt}$	$P_e = P^*_{e,opt}$	$P_e > P^*_{e,opt}$	$P_e < P^*_{e,opt}$
P'_e	$P_e + b$	$P_e + 1$	P_e	$P_e + b$	$P_e + 1$	P_e
p'	$p + b$	$p + 1$	p	$p - b$	$p - 1$	p

Table 1
Predicted value of pixel p according to different cases of $c_{\max}$ and $c_{\min}$.

	$c_{\max} \neq c_{\min}$			$c_{\max} = c_{\min}$	
	$p \geqslant c_{\max}$	$p \leqslant c_{\min}$	$c_{\min} < p < c_{\max}$	$p \geqslant c_{\max} + 1$	$p \leqslant c_{\max}$
$\hat{p}$	$c_{\max}$	$c_{\min}$	ϕ	$c_{\max}$	
P_e	$p - c_{\max}$	$c_{\min} - p$		$p - c_{\max}$	$c_{\max} - p$

Table 2
The data embedding process of p (or P_e) for the case of $c_{\max} \neq c_{\min}$.

	$c_{\max} \neq c_{\min}$						
	$p \geqslant c_{\max}$			$p \leqslant c_{\min}$			$c_{\min} < p < c_{\max}$
	$P_e = P^*_{e,opt}$	$P_e > P^*_{e,opt}$	$P_e < P^*_{e,opt}$	$P_e = P^*_{e,opt}$	$P_e > P^*_{e,opt}$	$P_e < P^*_{e,opt}$	ϕ
P'_e	$P_e + b$	$P_e + 1$	P_e	$P_e + b$	$P_e + 1$	P_e	
p'	$p + b$	$p + 1$	p	$p - b$	$p - 1$	p	

$\Delta \leqslant \upsilon T_3$ are divided into two sets: S_{PA1} and C_{PA1}.The third-stage embedding is performed to embed data into the pixels in S_{PA1}.

3.2.1. Optimal-threshold-determination for combined embedding

There can be many possible combinations of bins, each of which can achieve the required EC. Since we do not always select peak bins as the optimal embedding bins, and therefore, we have to determine which bins are utilized for data embedding. The optimal embedding bins cannot only increase the capacity but also reduce the embedding distortion. In this paper, two stages are treated jointly as so to find the optimal embedding bins.

The thresholds υT_1 and υT_2 are utilized respectively in A and B to select only pixels in smooth regions to join the embedding process. For each pixel in A, the variance Δ of I_{LSE} is calculated via Eq. (7). If $\Delta \leqslant \upsilon T_1$, then it is classified into S_{PA}, and otherwise, it belongs to C_{PA}. For each pixel in C_{PA}, it is kept unchanged, i.e., $p' = p$.

After each pixel in S_{PA} is predicted via Table 1 to obtain its prediction-error, the PEH is generated via Eq. (8) by counting the occurrences of prediction-errors in S_{PA}.

$$H_1(k,t) = \#\{1 \leqslant i \leqslant N_s : P_{e,i} = k, \Delta_i = t\} \tag{8}$$

where $H_1(k, \upsilon T_1)$ represents the number of prediction-errors whose values equal k for the case of $\Delta = t$, the subscript 1 of $H_1(\cdot)$ represents the first stage embedding, # denotes the cardinal number of a set, N_s $(N_s \leqslant N)$ denotes the total number of pixels in S_{PA}, and $k \in [0, 255]$.

As we have known (see Section II-C of [31] for details), for the histogram-based RDH, if the maximum modification to pixel values is 1 in data embedding, the expected value of the modification (in l^2-norm) to cover image, denoted as $Dpro(\upsilon T_1, \upsilon T_2, P^{1*}_{e,opt}, P^{2*}_{e,opt})$, is $\frac{1}{2}N_{exp} + N_{shift}$, where N_{exp} and N_{shift} are numbers of expanded and shifted pixels, respectively. N_{exp} can be formulated as

$$N_{exp} = \sum_{t \leqslant \upsilon T_1} H_1(P^{1*}_{e,opt}, t) + \sum_{t \leqslant \upsilon T_2} H_2(P^{2*}_{e,opt}, t) \tag{9}$$

where $P^{1*}_{e,opt}$ denotes the optimal bin in the first stage for the case of $\Delta \leqslant \upsilon T_1$ while $P^{2*}_{e,opt}$ is the optimal one in the second stage for the case of $\Delta \leqslant \upsilon T_2, H_2(\cdot)$ is used to denote the PEH in the second stage embedding.

And, N_{shift} can be written as

$$N_{shift} = \sum_{t \leqslant \upsilon T_1} \sum_{k > P^{1*}_{e,opt}} H_1(k,t) + \sum_{t \leqslant \upsilon T_2} \sum_{k > P^{2*}_{e,opt}} H_2(k,t) \tag{10}$$

The capacity, denoted as $Cpro(\upsilon T_1, \upsilon T_2, P^{1*}_{e,opt}, P^{2*}_{e,opt})$, equals N_{exp}.

The optimal parameter determination is actually equivalent to the following minimization problem.

$$\begin{cases} \displaystyle\mathop{\arg\min}_{0 \leqslant \upsilon T_1, \upsilon T_2 \leqslant \Delta_{\max}} \frac{Dpro(\upsilon T_1, \upsilon T_2, P^{1*}_{e,opt}, P^{2*}_{e,opt})}{Cpro(\upsilon T_1, \upsilon T_2, P^{1*}_{e,opt}, P^{2*}_{e,opt})} \\ \text{subject to } Cpro \geqslant L_{1_{\sum}} + L_{2_{\sum}} + L_w \end{cases} \tag{11}$$

where $\Delta_{\max}$ denotes the maximum variance, L_w is the payload size, $L_{1_{\sum}}$ and $L_{2_{\sum}}$ stand for the length of all overhead information in the first and second layer, respectively.

3.2.2. Simplified parameters determination

In the proposed method, two-stage embedding is utilized jointly to achieve the required payload. As illustrated in Fig. 5, in each stage, the bins which can provide larger than or equal to half of the required EC are considered as possible candidates for the optimal embedding bins. Specifically, in the first stage, for given υT_1, the PEH $H_1(\cdot)$ is formed by counting the occurrences of prediction-errors in S_{PA}. For one bin k, if $H_1(k) \geqslant \frac{L_w}{2}$, then it is a potential candidate, where $H_1(k)$ is short for $H_1(k,t)$. All the candidates are sorted in ascending order to form CS_1, i.e., $CS_1 = \{P^1_{e,lopt}, \ldots, p^1_{e,ropt}\}$, where $P^1_{e,lopt}$ and $P^1_{e,ropt}$ are used to denote the minimum and maximum in CS_1, respectively. We utilize exhaustive search to determine the optimal embedding bin. Specifically, $P^1_{e,opt}$ is taken as the minimum (i.e., $P^1_{e,lopt}$) in CS_1. Then, the pixels in S_{PA} are modified and $H_1(P^1_{e,lopt})$ bits are embedded into P_e whose value is equal to $P^1_{e,lopt}$ according to Section 3.4 such that the marked image I_{WA} is obtained.

Then, in the second stage, the already-modified pixels in A of I_{WA} are used to predict B, and the corresponding PEH $H_2(\cdot)$ is obtained for given υT_2. Similarly, for one bin k, if its $H_2(k)$ satisfies the condition that $H_2(k) + H_1(P^1_{e,lopt}) \geqslant L_w$, then it is a potential candidate, where $H_2(k)$ is short for $H_2(k,t)$. All the candidates are collected into a set called CS_2, i.e., $CS_2 = \{P^2_{e,lopt}, \ldots, p^2_{e,ropt}\}$, where $P^2_{e,lopt}$ is the minimum in CS_2, and $P^2_{e,lopt}$ is the maximum. $P^2_{e,opt}$ is taken as the minimum (i.e., $P^2_{e,lopt}$) in CS_2, and $H_2(P^2_{e,opt})$ bits are embedded into P_e whose value equals to $P^2_{e,opt}$ according to Section 3.4. When the required capacity is just accommodated, stop the embedding process and the marked image I_W is obtained. Meanwhile, the corresponding embedding distortion in terms of MSE is recorded. Then, repeat the above process for all the bins in CS_2, and finally obtain the temporary optimal $(\upsilon T_1, \upsilon T_2, P^1_{e,lopt}, P^{2t}_{e,opt})$ as the one providing the smallest MSE for given $(\upsilon T_1, \upsilon T_2, P^1_{e,lopt})$, where $P^{2t}_{e,opt}$ denotes the temporary optimal bin in the second stage for given $(\upsilon T_1, \upsilon T_2, P^1_{e,lopt})$.

When $(\upsilon T_1, \upsilon T_2)$ is fixed, each bin in CS_1 corresponds to a set CS_2. Repeat the above process for all the bins in CS_1, the optimal $(\upsilon T_1, \upsilon T_2, P^{1*}_{e,opt}, P^{2*}_{e,opt})$ is obtained as the one providing the smallest MSE for given $(\upsilon T_1, \upsilon T_2)$.

In the experiment, for a given EC, we increase υT_1 gradually from the initial value $\upsilon T_1 = 1$ to its maximum (e.g., 10) with a step size of 0.5. For each value of υT_1, we increase υT_2 gradually from the initial value $\upsilon T_2 = 1$ to its maximum (e.g., 10) with a step size of 0.5. For each given $(\upsilon T_1, \upsilon T_2)$, $(P^{1*}_{e,opt}, P^{2*}_{e,opt})$ is obtained as the one providing the lowest MSE. The optimal $(\upsilon T^*_1, \upsilon T^*_2)$ is the one which can yield the lowest MSE among all combinations of $(\upsilon T_1, \upsilon T_2)$.

3.3. A simple example

A simple example is given in Fig. 6 to illustrate the embedding process of our proposed method. For simplicity, we suppose the optimal embedding bin $P^*_{e,opt}$ is 0 for a single stage. From Fig. 6, it is observed that the current pixel p is marked in red, while its context pixels marked in green are composed of $c_1, \ldots, c_{13}$. In Fig. 6(a), since the context pixels are strongly related to each other, and $p = c_{\max} = 162, c_{\max} > c_{\min}, p$ can be embedded with 1 data bit. The marked pixel is 163 due to that the to-be-embedded bit b is 1. Suppose that p in Fig. 6(b) is still located in a smooth region. Since $p > c_{\max}$, the prediction-error equals 3 and it is not the optimal embedding bin. So, p must be increased by 1 for reversibility.

3.4. Embedding procedure

Now, the detailed embedding procedure is described as follows step by step. Here, for clarity of description, we utilize a notation S_P to represent S_{PA} in the first stage or S_{PB} in the second stage. This is similar for O_P and C_P. The notation $\mathcal{P}$ is used to denote the required payload,

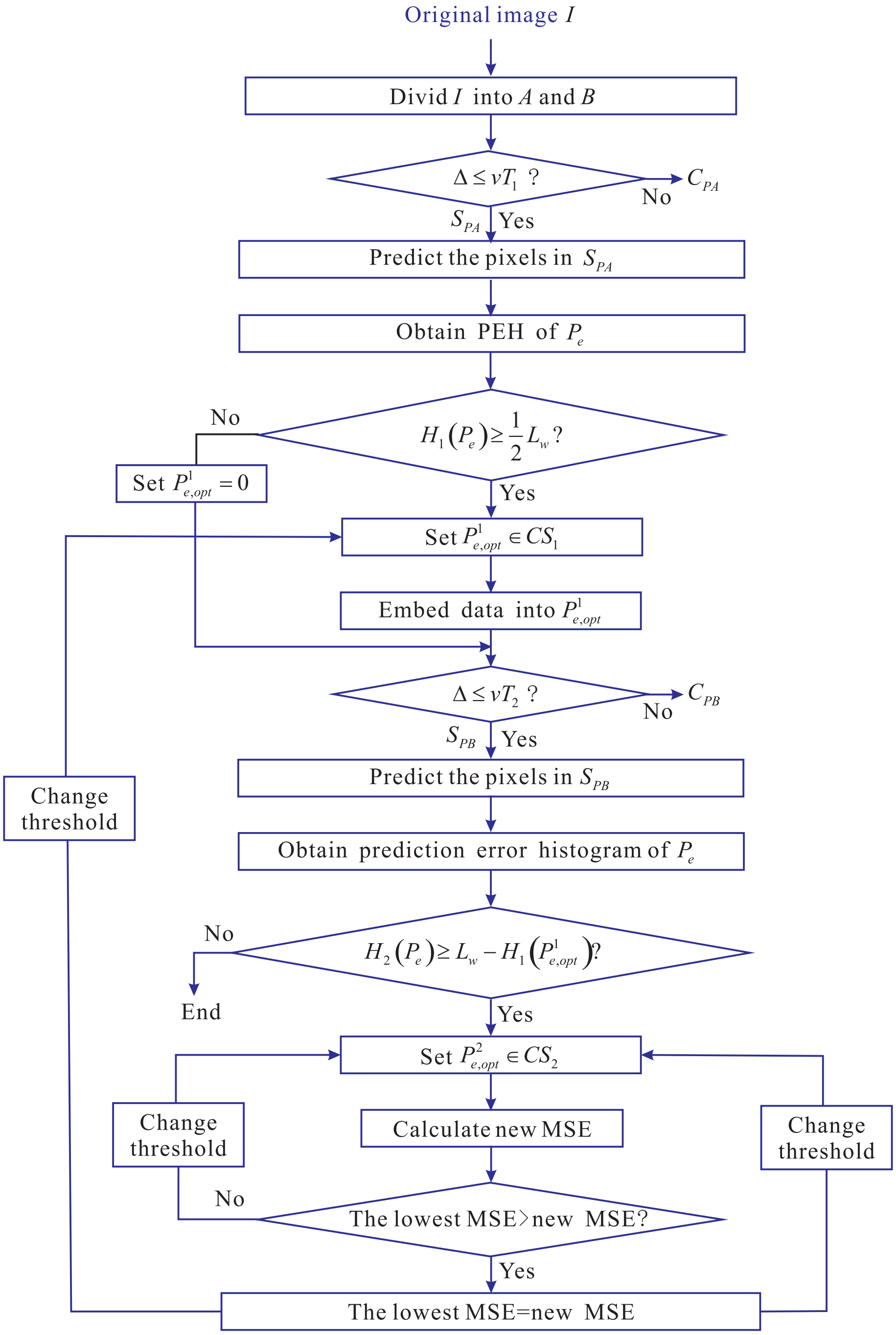

Original image I
Divid I into A and B
$\Delta \le vT_1$?
No
C_{PA}
S_{PA}
Yes
Predict the pixels in S_{PA}
Obtain PEH of P_e
$H_1(P_e) \ge \frac{1}{2} L_w$?
No
Set $P^1_{e,opt} = 0$
Yes
Set $P^1_{e,opt} \in CS_1$
Embed data into $P^1_{e,opt}$
$\Delta \le vT_2$?
No
C_{PB}
S_{PB}
Yes
Change threshold
Predict the pixels in S_{PB}
Obtain prediction error histogram of P_e
$H_2(P_e) \ge L_w - H_1(P^1_{e,opt})$?
No
End
Yes
Set $P^2_{e,opt} \in CS_2$
Change threshold
Calculate new MSE
Change threshold
No
The lowest MSE>new MSE?
Yes
The lowest MSE=new MSE

Fig. 5. The optimal-threshold-determination for two-stage embedding.

Step 1 *Data embedding in a single layer*

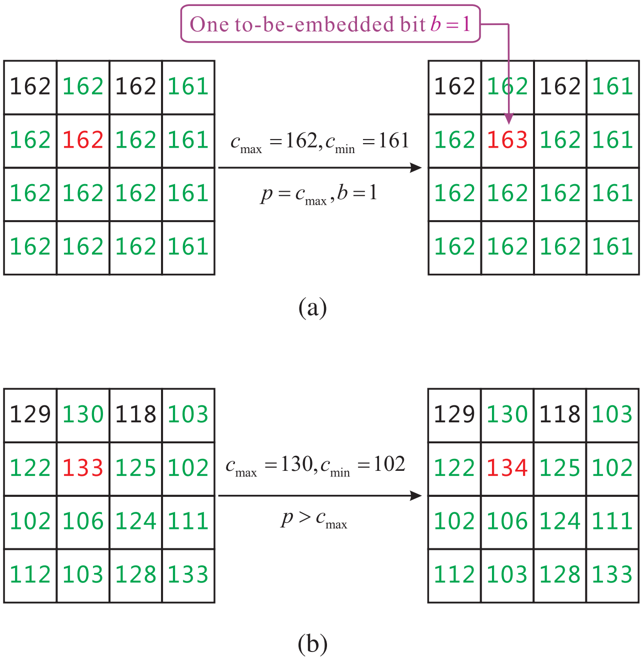
The overhead information of the current stage is obtained. Suppose that the payload to be embedded in the current stage is $\mathcal{P}_C$. First, scan each pixel in A (or B) in a predefined order. If one pixel belongs to C_P or O_P, then it is ignored. If p is in S_P, p is modified according to Tables 2 and 3. After the first $L_{\sum}$ pixels in A (or B) are processed, their LSBs constitute a bit sequence $\mathcal{C}$, and then replaced by the overhead information. Next, the rest part of $\mathcal{P}_C$ and the bit sequence $\mathcal{C}$ are embedded into the remaining pixels in S_P using the same method illustrated in **Step 1.1**.

Step 2 *Obtaining the marked image I_W*

Step 2 is divided into two parts as follows for clarity of description. $\mathcal{P}_C$ is initialized to $\mathcal{P}$, and $|\mathcal{P}_C| = L_w$. If $|\mathcal{P}_C|$ is smaller than or equal to $H_m(P^{m*}_{e,opt})$ $(m \in \{1,2\})$ of this current stage, which implies this stage can provide $\mathcal{P}_C$, then go to **Step 2.1**. Otherwise, do **Step 2.2**.

Step 2.1 *Creating the marked image I_W*

$|\mathcal{P}_C| \leqslant H_m(P^{m*}_{e,opt})$ implies that the current stage is sufficient for providing $\mathcal{P}$. That is to say, this is the last stage. After **Step 1** is implemented, a new marked image I_W is generated.


One to-be-embedded bit $b = 1$
$c_{\max} = 162, c_{\min} = 161$
$p = c_{\max}, b = 1$
(a)
$c_{\max} = 130, c_{\min} = 102$
$p > c_{\max}$
(b)

(a)

162	162	162	161
162	162	162	161
162	162	162	161
162	162	162	161

162	162	162	161
162	163	162	161
162	162	162	161
162	162	162	161

(b)

129	130	118	103
122	133	125	102
102	106	124	111
112	103	128	133

129	130	118	103
122	134	125	102
102	106	124	111
112	103	128	133

Fig. 6. A simple example.

Step 2.2 *Performing the second layer embedding*

Due to $|\mathcal{P}_C| > H_m(P^{m*}_{e,opt})$, we must perform two-stage embedding so as to achieve the required payload $\mathcal{P}$. First, for this stage, perform **Step 1** for the payload of length $H_m(P^{m*}_{e,opt})$ and the corresponding overhead information. After the first stage, the size of the residual payload is $|\mathcal{P}_L| - H_m(P^{m*}_{e,opt})$. Set $|\mathcal{P}_C| = |\mathcal{P}_L| - H_m(P^{m*}_{e,opt})$. Next, go to **Step 1** for next stage embedding.

3.5. Data extraction and image restoration

Step 1 *Extraction of the overhead information*

According to the same order as in embedding, the LSBs of the marked pixels in A (or B) are collected into a bitstream $\mathcal{B}$. By identifying the EOS symbol in $\mathcal{B}$, the bits which contains the compressed map $\mathcal{L}$, are decompressed by an arithmetic decoder to obtain the location map. The location map is compressed losslessly by an arithmetic encoder so as to obtain its length, i.e., L_S. Once L_S is retrieved, $\upsilon T_m, P^{m*}_{e,opt}, n$ and MT will be extracted one by one according to their fixed lengths. We can easily know the number of stages and the embedded bits for the last stage by obtaining EC, respectively.

Step 2 *Extraction of the payload $\mathcal{P}$*

For each marked block p', if its location is marked by '0' in the location map, then it is kept unchanged. If its location is marked by '1' and $\Delta \leqslant \upsilon T_m$, extract data bits and retrieve the original values from p' according to Tables 4–6.

Step 3 *Retrieve of the original image*

If the number of extracted data bits is identical to the capacity calculated by EC, stop extraction and form $\mathcal{P}$. Otherwise, go the next stage extraction and repeat steps above. When the current stage is fully extracted, the extracted bitstream is collected into data bits and the $L_{\sum}$ original pixels in A (or B) are obtained by simple LSB replacement.

4. Experimental results

The 'Lena', 'Baboon' 'Barbara' 'Boat', 'Peppers' and 'Airplane' images with size 512 × 512 are provided by the authors of paper [44]. Fig. 7 shows performance comparisons between the proposed method and the following eight methods: Qu et al. [50], Ou et al. [45,47], Peng et al. [46], Li et al. [44], Sachnev et al. [23], Hong et al. [55] and Hwang et al. [56]. The source codes of Sachnev et al.'s and Ou et al.'s methods are provided by the authors of paper

Table 4
Recovered p_e from the marked prediction-error P'_e according to different cases of $c_{\max}$ and $c_{\min}$.

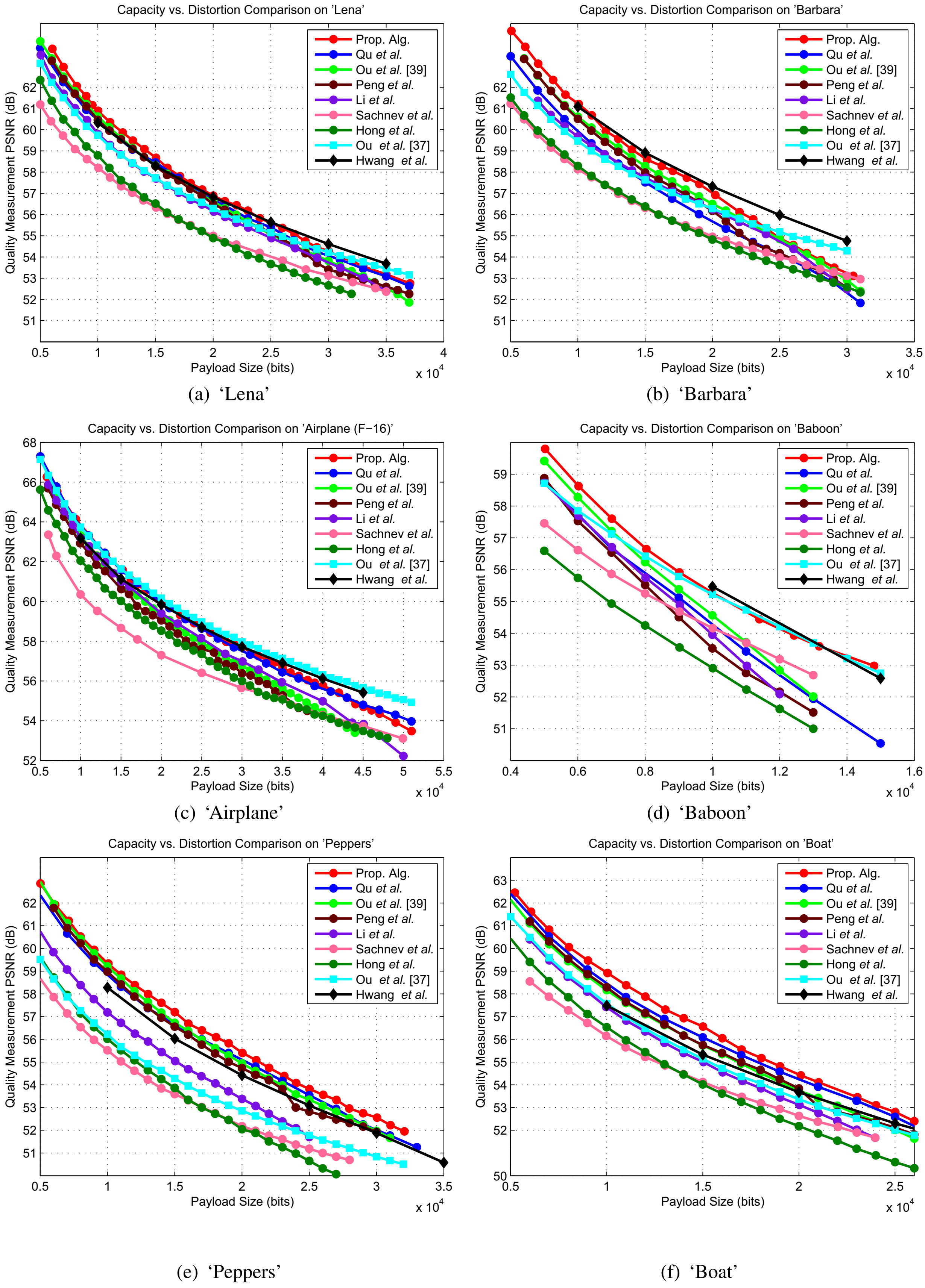
	$c_{\max} \neq c_{\min}$			$c_{\max} = c_{\min}$	
	$p' \geqslant c_{\max}$	$p' \leqslant c_{\min}$	$c_{\min} < p' < c_{\max}$	$p' \geqslant c_{\max} + 1$	$p' \leqslant c_{\max}$
P'_e	$p' - c_{\max}$	$c_{\min} - p'$	ϕ	$p' - c_{\max}$	$c_{\max} - p'$

Table 5
Extracted bit and recovered value from a marked pixel p' for the case of $c_{\max} \neq c_{\min}$.

	$c_{\max} \neq c_{\min}$							
	$p' \geqslant c_{\max}$				$p' \leqslant c_{\min}$			
	$P'_e = P^*_{e,opt}$	$P'_e = P^*_{e,opt} + 1$	$P'_e > P^*_{e,opt}$	$P'_e < P^*_{e,opt}$	$P'_e = P^*_{e,opt}$	$P'_e = P^*_{e,opt}$	$P'_e > P^*_{e,opt}$	$P'_e < P^*_{e,opt}$
p b	p' $b = 0$	$p' - 1$ $b = 1$	$p' - 1$ Bo embedded bit	p'	p' $b = 0$	$p' + 1$ $b = 1$	$p' + 1$ No embedded bit	p'

Table 6
Extracted bit and recovered value from a marked pixel p' for the case of $c_{\max} = c_{\min}$.

	$c_{\max} = c_{\min}$							
	$p' > c_{\max}$				$p' \leqslant c_{\max}$			
	$P'_e = P^*_{e,opt}$	$P'_e = P^*_{e,opt} + 1$	$P'_e > P^*_{e,opt}$	$P'_e < P^*_{e,opt}$	$P'_e = P^*_{e,opt}$	$P'_e = P^*_{e,opt}$	$P'_e > P^*_{e,opt}$	$P'_e < P^*_{e,opt}$
p	$p', b = 0$	$p' - 1, b = 1$	$p' - 1$	$p', b = 0$	$p' + 1, b = 1$	$p' + 1$	$p' + 1$	p'



(a) 'Lena'

(b) 'Barbara'

(c) 'Airplane'

(d) 'Baboon'

(e) 'Peppers'

(f) 'Boat'

Fig. 7. Performance comparisons between the proposed method and following eight methods: Qu et al. [48], Ou et al. [47,45], Peng et al. [46], Li et al. [44], Sachnev et al. [23], Hong et al. [55] and Hwang et al. [56]

[47]. The source codes of Li et al.'s, Peng et al.'s and Hwang et al.'s methods are provided by the authors of paper [44].

In Fig. 7, the capacity range in comparison is limited from an initial capacity of 5000 (or 6000) bits to the maximum capacity of Qu et al.'s method. From Fig. 7(a), one can see that the maximum capacity is 37,000 bits for 'Lena'. In the experiments, we need the third-stage embedding to achieve the maximum capacity. The third-stage embedding means that we re-embed data into an already embedded image, which is obtained after two-stage embedding. The embedding process of third-stage embedding is similar to that of first-stage one. Correspondingly, the idea of the simplified parameters determination in Section 3.2.2 is applied to three-stage embedding, so that the optimal embedding bins of three-stage embedding are obtained. From Fig. 7, one can observe that our method significantly outperforms Qu et al.'s for every image whatever the ER is. Our advantage lies in the utilization of the flexible prediction pattern and the optimal embedding bins. In Qu et al.'s method, PPVO generates predictions in a pixel-by-pixel manner. Since the half-enclosed-based prediction is utilized in PPVO, the number of the obtained prediction-errors is approximately equal to the image size, and correspondingly, the number of embeddable prediction-errors is also high. Depending on a large number of embeddable prediction-errors, PPVO can achieve higher capacity-distortion performance than Ou et al.'s [47] and Peng et al.'s methods. However, for complex texture images such as 'Baboon' and 'Barbara', the half-enclosed-based prediction is unable to evaluate accurately the local complexity, and hence, the performance is weaker than that of Ou et al.'s method [47]. In the proposed method, two-stage embedding is utilized to ensure that almost all the pixels can be predicted. Since in the proposed method, n neighbors surrounding p (i.e., $\{c_1, c_2, \ldots, c_n\}$, and $n \in \{4, \ldots, 13\}$) are used to predict p, the prediction performance is largely increased. Taking 'Baboon' and 'Barbara' for example, our capacity-distortion performance is further superior to that of Ou et al.'s method [47].

Sachnev et al.'s method is a representative PEE method, which performs well by incorporating the sorting technique into the rhombus predictor [23]. From Fig. 7, one can observe that our method achieves higher embedding performance than Sachnev et al.'s one when the payload is low. We will illustrate the reason from the following two aspects. The first one is that our method utilizes more pixels to evaluate the local complexity, compared with Sachnev et al.'s one. The second one is that n neighbors (i.e., $\{c_1, c_2, \ldots, c_n\}$) of each to-be-embedded pixel are involved into prediction, and hence, the prediction is more accurate than that of the rhombus predictor. According to Fig. 3, for the rhombus predictor, in the second stage, each pixel in B is predicted by its neighbors in A which may be not original pixels (i.e., may be have been modified in the first layer). While for Qu et al.'s method, each pixel is predicted by its bottom and right neighbors, which are always original ones. Based on this reason, Qu et al.'s method can obtain the sharpest histogram (see Fig. 8). From Fig. 8, one can observe that the our histogram is slightly lower than those of rhombus prediction scheme and the scheme used in Qu et al. In our method, the peak bin (bin 0) may not be selected for data embedding due to that it may not the optimal embedding bin. From the viewpoint, our method can still achieve good embedding performance.

Li et al.'s method is also a high-fidelity RDH method. The advantage of Li et al.'s method lies in the fact that a pixel pair (x, y) can generate a difference pair (d_1, d_2) by exploiting z, the prediction of y. In this way, a two-dimensional difference histogram of (d_1, d_2) is formed, and it can provide more expanded pixels pairs, in comparison with a one-dimensional difference histogram of d_1 (or d_2). Specially, there exist four cases of (d_1, d_2), i.e., $d_1 = 0, d_2 = 0$, $d_1 = 1$ and $d_2 \geqslant -1$ and $d_1 = -1$ and $d_2 < 0$, for each of which (x, y) can carry 1-bit data. To ensure reversibility, the other cases are shifted, i.e., $d_1 > 1$ and $d_2 \neq 0, d_1 < -1$ and $d_2 < 0, d_1 < 0$ and $d_2 > 0$, and $d_1 = 1$ and $d_2 < -1$. Li et al.'s method utilizes the GAP predictor [42] to obtain z. However, in our method, for a pixel to be embedded, we utilize its n nearest neighbors to predict it, and hence, our method predicts more accurately than that in Li et al.'s method. Based on this reason, our method yields better results than Li et al.'s for most of six test images. Peng et al.'s method utilize PVO to perform prediction. As shown in [18], PVO can obtain better prediction performance than Gap. Therefore, Peng et al.'s method also is superior to Li et al's. However, for 'Airplane', Li et al.'s method achieves better performance than Peng et al's. In fact, in Peng et al.'s method, each block can carry at most two bits. That is to say, a large portion of pixels in a block are not involved in data embedding. For 'Airplane', Peng et al.'s method cannot fully exploit all the pixels located in smooth regions. Based on this reason, Li et al.'s method exceeds Peng et al.'s method in terms of payload and distortion. From Fig. 7(b), one can see that our method obtains a better performance than Li et al's. This is due to the utilizations of the flexible block-partition and the adaptive pixel-modification strategy.

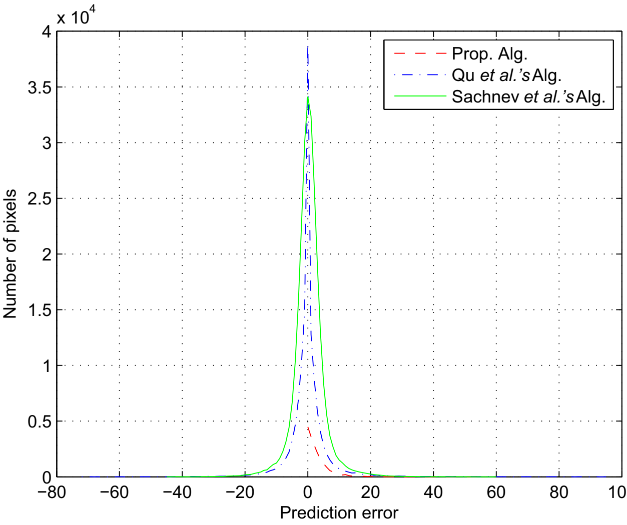


Fig. 8. The prediction-error histograms for Sachnev et al. [23], Qu et al. [48] and the proposed scheme on Lena.

Hong et al. have proposed an adaptive method to increase the number of the prediction-errors capable of carrying data by referencing a dual binary tree. They also employ the MED predictor (a half-enclosed predictor) and an error energy estimator to decrease the number of shifted prediction-errors [55]. In fact, the prediction accuracy is the key of RDH. As aforementioned, our predictor can achieve more accurate prediction than the rhombus one [23] and the MED one when the required capacity is not high. Based on this reason, our method is superior to Hong et al.'s one in the embedding performance (see Fig. 7 for details).

The main purpose of Ou et al. [45] is to reduce the amount of image modifications as much as possible for a given EC. Motivated by this purpose, the pairwise PEE has been proposed by Ou et al. to reduce embedding modifications to some embeddable prediction-error pair containing two neighboring prediction-errors. For a prediction-error pair capable of carrying 2 bits, it is known that two bits of '11' among all four combinations of two bits (i.e., '00', '01', '10' and '11') will introduce the largest modifications to this pair. Therefore, the pairwise PEE is utilized to exclude two bits of '11' from data embedding, so that the modifications to a prediction-error pair are decreased, and accordingly, the high

Table 7
Comparison of time cost on six test images (i.e., 'Lena','Baboon','Barbara','Boat','Airplane','Peppers' and 'Boat'), where the unit of runtime is second.

	'Lena'	'Baboon'	'Barbara'	'Airplane'	'Peppers'	'Boat'
Prop.	11.355	9.7886	9.3493	9.5790	18.0786	16.68
Qu	9.8713	10.760	7.6659	9.2989	10.654	10.301
Peng	0.308975	2.9023	0.322238	0.327275	0.19606	0.2304062
Ou [47]	0.32373	1.1592	0. 50179	0.28214	0.45335	0.68577
Li	5.902122	15.459998	6.11614	4.466062	10.443747	12.523780
Sachnev	9.0086	9.1382	8.9390	8.6698	9.0841	8.9669
Hwang	146.86	90.6714	111.92	133.53	136.61	153.77

visual quality is preserved. From Fig. 7, we can observe that our proposed method can achieve higher embedding performance than Ou et al.'s one [45] for 'Boat' and 'Peppers'. For 'Baboon', our method is superior to Ou et al.'s one [45] when the EC is smaller than or equal to 10,000 bits, and achieves almost the same performance as Ou et al.'s one [45] for the EC of over 10,000 bits. For 'Lena' and 'Barbara', compared with Ou et al.'s method [45], ours can obtain higher PSNR values when the EC is low (e.g., smaller than 30,000 bits for 'Lena' or smaller than 25,000 bits for 'Barbara'). With the capacity further increased, Ou et al.'s method [45] exceeds ours and obtains better performance. For 'Airplane', when the EC is low (i.e., smaller than 30,000 bits), our performance is almost the same as Ou et al.'s one [45]. When the EC is larger than 30,000 bits, our performance is lower than Ou et al.'s one [45].

According to the aforementioned description, Hwang et al.'s method utilizes the optimal embedding bins, instead of the peak bins, for data embedding. Hence, how to find four optimal embedding thresholds in each stage is the key of obtaining as high PSNR value as possible for a given EC in Hwang et al.'s method. Correspondingly, the computational complexity is also very large. From Fig. 7, for 'Boat' and 'Peppers', our method outperforms Hwang et al.'s one at almost all ECs. For the other four images, when the EC is not high (e.g., smaller than 25,000 bits for 'Lena'), our method outperforms Hwang et al.'s one. With the EC increased, Hwang et al.'s method outperforms ours. Although our method cannot achieve better performance than Hwang et al.'s one when the EC is larger, our time cost is lower than Hwang et al.'s one. This is because Hwang et al.'s method must spend a lot of time searching four the optimal thresholds for each stage. However, in our method, we decrease computational complexity by selecting one optimal embedding bin for each stage. From Table 7, one can observe that the time cost of Hwang et al.'s method is higher than ours.

The average time cost of our method, Peng et al.'s, Qu et al.'s, Ou et al.'s [47], Sachnev et al.'s, Li et al.'s and Hwang et al.'s ones for six test images is calculated in Table 7, respectively. We do not obtain the source code of Hong et al.'s method, so we cannot calculate the time cost of their method. The proposed and all compared methods are performed on Matlab R2013a, and all the experiments are carried out on a PC. From Table 7, it can be observed that the time cost of the proposed method varies with images. Our time cost is mainly determined by the time of searching the optimal bin for each stage. Once the optimal bins are obtained, the followed data embedding is fast. From Table 7, one can see that our time cost is higher than those of the other five methods except Hwang et al.'s method. Our time cost is still acceptable in this age because we can decrease it by a high-performance PC or an effective programming language such as C++.

5. Conclusions

In this paper, a novel RDH method based on flexible prediction pattern and the optimal-embedding-bin search scheme is proposed. Flexible prediction pattern has two advantages. One is that for p, its n neighbors surrounding it, i.e., $\{c_1, c_2, \ldots, c_n\}$ ($n \in \{4, \ldots, 13\}$), are utilized to predict p, and accordingly, the predicted value $\hat{p}$ is more close to p. The other is that the local complexity can be estimated accurately by utilizing 13 neighbors. Unlike the existing methods in which the peak bins are selected for data embedding, our method obtains the optimal embedding bins of two stages which can introduce the lowest distortion at a given EC by exhaustive search. Extensive experiments verify that the proposed method outperforms Peng et al.'s, Qu et al.'s, Ou et al.'s, Li et al.'s, Sachnev et al.'s and Hong et al.'s works.

Acknowledgment

This work was supported in part by National NSF of China (Nos. 61571139, 61201393 and 61574049), New Star of Pearl River on Science and Technology of Guangzhou (No. 2014J2200085).

References

[1] C.S. Yuan, X.M. Sun, L.V. Rui, Fingerprint liveness detection based on multi-scale lpq and pca, China Commun. 13 (7) (2016) 60–65.

[2] Z.H. Xia, X.H. Wang, L.G. Zhang, Z. Qin, X.M. Sun, K. Ren, A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing, IEEE Trans. Inf. Forensics Secur. 11 (11) (2016) 2594–2608.

[3] X.Y. Chen, S. Chen, Y.L. Wu, Coverless information hiding method based on the chinese character encoding, J. Internet Technol. 18 (2) (2017) 91–98.

[4] Z.H. Xia, X.H. Wang, X.M. Sun, Q.S. Liu, N.X. Xiong, Steganalysis of LSB matching using differences between nonadjacent pixels, Multimed. Tools Appl. 75 (4) (2016) 1947–1962.

[5] Z.H. Xia, X.H. Wang, X.M. Sun, B.W. Wang, Steganalysis of least significant bit matching using multi-order differences, Secur. Commun. Networks 7 (8) (2014) 1283–1291.

[6] J. Li, X.L. Li, B. Yang, X.M. Sun, Segmentation-based image copy-move forgery detection scheme, IEEE Trans. Inf. Forensic Secur. 10 (3) (2015) 507–518.

[7] Z.L. Zhou, C.-N. Yang, B.J. Chen, X.M. Sun, Q. Liu, Q.M. Jonathan Wu, Effective and efficient image copy detection with resistance to arbitrary rotation, IEICE Trans. Inf. Syst. E99-D (6) (2016) 1531–1540.

[8] Z.L. Zhou, Y.L. Wang, Q.M.J. Wu, C.-N. Yang, X.M. Sun, Effective and efficient global context verification for image copy detection, IEEE Trans. Inf. Forensic Secur. 12 (1) (2017) 48–63.

[9] J. Fridrich, M. Goljan, R. Du, Lossless data embedding-new paradigm in digital watermarking, EURASIP J. Appl. Signal Process. 2002 (2002) 185–196.

[10] M.U. Celik, G. Sharma, A.M. Tekalp, E. Saber, Lossless generalized-lsb data embedding, IEEE Trans. Image Process. 12 (2) (2005) 157–160.

[11] L. Kamstra, H.J.A.M. Heijmans, Reversible data embedding into images using wavelet technique and sorting, IEEE Trans. Image Process. 14 (12) (2005) 2082–2090.

[12] J. Tian, Reversible data embedding using a difference expansion, IEEE Trans. Circuits Syst. Video Technol. 13 (8) (2003) 890–896.

[13] A.M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, IEEE Trans. Image Process. 13 (8) (2004) 1147–1156.

[14] D. Coltuc, J.M. Chassery, Very fast watermarking by reversible contrast mapping, IEEE Signal Process. Lett. 14 (4) (2007) 255–258.

[15] S.W. Weng, Y. Zhao, J.S. Pan, R.R. Ni, Reversible watermarking based on invariability and adjustment on pixel pairs, IEEE Signal Process. Lett. 45 (20) (2008) 1022–1023.

[16] S.W. Weng, Y. Zhao, R.R. Ni, J.S. Pan, Parity-invariability-based reversible watermarking, IET Electron. Lett. 1 (2) (2009) 91–95.

[17] D. Coltuc, Low distortion transform for reversible watermarking, IEEE Trans. Image Process. 21 (1) (2012) 412–417.

[18] F. Peng, X. Li, B. Yang, Adaptive reversible data hiding scheme based on integer transform, Signal Process. 92 (1) (2012) 54–62.

[19] Z. Ni, Y.Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 16 (2006) 354–362.

[20] G.R. Xuan, C.Y. Yang, Y.Z. Zhen, Y.Q. Shi, Reversible data hiding using integer wavelet transform and companding technique, in: Proceedings of IWDW, vol. 5, 2004, pp. 23–26.

[21] X.L. Li, B. Li, B. Yang, T.Y. Zeng, General framework to histogram-shifting-based reversible data hiding, IEEE Trans. Image Process. 22 (6) (2013) 2181–2191.

[22] Y. Hu, H.K. Lee, J. Li, DE-based reversible data hiding with improved overflow location map, IEEE Trans. Circuits Syst. Video Technol. 19 (2) (2009) 250–260.

[23] V. Sachnev, H.J. Kim, J. Nam, S. Suresh, Y.Q. Shi, Reversible watermarking algorithm using sorting and prediction, IEEE Trans. Circuits Syst. Video Technol. 19 (7) (2009) 989–999.

[24] P.Y. Tsai, Y.C. Hu, H.L. Yeh, Reversible image hiding scheme using predictive coding and histogram shifting, Signal Process. 89 (6) (2009) 1129–1143.

[25] W.L. Tai, C.M. Yeh, C.C. Chang, Reversible data hiding based on histogram modification of pixel differences, IEEE Trans. Circuits Syst. Video Technol. 19 (6) (2009) 906–910.

[26] W. Hong, T.S. Chen, C.W. Shiu, Reversible data hiding for high quality images using modification of prediction errors, J. Syst. Softw. 82 (11) (2009) 1833–1842.

[27] L. Luo, Z. Chen, M. Chen, X. Zeng, Z. Xiong, Reversible image watermarking using interpolation technique, IEEE Trans. Inf. Forensic Secur. 5 (1) (2010) 187–193.

[28] W. Hong, An efficient prediction-and-shifting embedding technique for high quality reversible data hiding, EURASIP J. Adv. Signal Process. (2010).

[29] D. Coltuc, Improved embedding for prediction-based reversible watermarking, IEEE Trans. Inf. Forensic Secur. 6 (3) (2011) 873–882.

[30] X. Gao, L. An, Y. Yuan, D. Tao, X. Li, Lossless data embedding using generalized statistical quantity histogram, IEEE Trans. Circuits Syst. Video Technol. 21 (8) (2011) 1061–1070.

[31] X.L. Li, B. Yang, T.Y. Zeng, Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection, IEEE Trans. Image Process. 20 (12) (2011) 3524–3533.

[32] H.-T. Wu, J.W. Huang, Reversible image watermarking on prediction errors by efficient histogram modification, Signal Process. 92 (12) (2012) 3000–3009.

[33] G. Coatrieux, W. Pan, N. Cuppens-Boulahia, F. Cuppens, C. Roux, Reversible watermarking based on invariant image classification and dynamic histogram shifting, IEEE Trans. Inf. Forensic Secur. 8 (1) (2013) 111–120.

[34] J. Li, X.L. Li, B. Yang, Reversible data hiding scheme for color image based on prediction-error expansion and cross-channel correlation, Signal Process. 93 (9) (2013) 2748–2758.

[35] S. Jung, L. Ha, S. Ko, A new histogram modification based reversible data hiding algorithm considering the human visual system, IEEE Signal Process. Lett. 18 (2) (2011) 95–98.

[36] W. Hong, T. Chen, M. Wu, An improved human visual system based reversible data hiding method using adaptive histogram modification, Opt. Commun. 291 (2013) 87–97.

[37] S.W. Weng, J.S. Pan, Reversible watermarking based on multiple predictionmodes and adaptive watermark embedding, Multimed. Tools Appl. 72 (3) (2013) 3063–3083.

[38] Q. Mao, F. Li, C.C. Chang, Reversible image data hiding with oriented and minimized distortions using cascading trellis coding, Inform. Sci. 317 (2015) 170–180.

[39] I.C. Dragoi, D. Coltuc, Local-prediction-based difference expansion reversible watermarking, IEEE Trans. Image Process. 23 (4) (2014) 1779–1790.

[40] M. Chen, Z. Chen, X. Zeng, Z. Xiong, Reversible data hiding using additive prediction-error expansion, in: Proc. of the 11th Workshop on Multimedia and Security, 2009, pp. 19–24.

[41] M. Weinberger, G. Seroussi, S. Sapiro, Loco-i: A low complexity, context-based, lossless image compression algorithm, in Proc. IEEE Data Compression Conf., 1996, pp. 140–149.

[42] X. Wu, N. Memon, Context-based adaptive lossless image coding, IEEE Trans. Commun. 45 (4) (1997) 437–444.

[43] X.L. 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.

[44] X.L. Li, W.M. Zhang, X.L. Gui, B. Yang, A novel reversible data hiding scheme based on two-dimensional difference-histogram modification, IEEE Trans. Inf. Forensic Secur. 8 (7) (2013) 1091–1100.

[45] 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) (2014) 5010–5021.

[46] F. Peng, X.L. Li, B. Yang, Improved pvo-based reversible data hiding, Digit. Signal Process. 25 (2014) 255–265.

[47] B. Ou, X.L. Li, Y. Zhao, R.R. Ni, Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion, Signal Process.: Image Commun. 29 (7) (2014) 198–205.

[48] X.C. Qu, H.J. Kim, Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding, Signal Process. 111 (2015) 249–260.

[49] W. Hong, T.S. Chen, J. Chen, Reversible data hiding using delaunay triangulation and selective embedment, Inform. Sci. 308 (2015) 140–154.

[50] X. Wang, J. Ding, Q.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.

[51] S.W. Weng, J.S. Pan, Adaptive reversible data hiding based on a local smoothness estimator, Multimed. Tools Appl. 74 (23) (2015) 10657–10678.

[52] X.L. Li, W.M. Zhang, X.L. Gui, B. Yang, Efficient reversible data hiding based on multiple histograms modification, IEEE Trans. Inf. Forensic Secur. 10 (9) (2015) 2016–2027.

[53] B. Ou, X.L. Li, J.W. Wang, Improved pvo-based reversible data hiding: a new implementation based on multiple histograms modification, J. Vis. Commun. Image R. 22 (12) (2014) 5010–5021.

[54] I.C. Dragoi, D. Coltuc, Adaptive pairing reversible watermarking, IEEE Trans. Image Process. 25 (5) (2016) 2420–2422.

[55] W. Hong, Adaptive reversible data hiding method based on error energy control and histogram shifting, Opt. Commun. 285 (2) (2012) 101–108.

[56] H.J. Hwang, H.J. Kim, V. Sachnev, S.H. Joo, Reversible watermarking method using optimal histogram pair shifting based on prediction and sorting, KSII Trans. Internet Inf. Syst. 4 (4) (2010) 655–670.