PVO-based reversible data hiding using adaptive multiple histograms generation and modification

Abstract

Keywords:

1. Introduction

Image data hiding has attracted much attention as an effective technology in copyright protection and authentication. Secret data embedded in an image can be extracted as evidence when copyright dispute occurs. In recent years, due to the special requirements for images in the fields of medical, military and political, many reversible data hiding (RDH) methods have been proposed. This kind of data hiding ensures that the cover image can be losslessly recovered after the secret data is extracted. Note that, the capacity-distortion performance is usually used for the evaluation of a RDH method, which measures the embedding distortion (ED) introduced to the cover image for a fixed embedding capacity (EC). Until now, many kinds of RDH methods are developed including compression-based [1, 2], difference expansion (DE) [3–5], histogram shifting (HS) [6–9], prediction-error-expansion (PEE) [10, 13, 14, 17–21], integer-to-integer transform [22–25] and so on.

Some of these kinds of methods are widely studied. DE is first proposed by Tian in [3], in which the correlation between adjacent pixels are exploited and the difference between them are expanded for data embedding. Another well-known category of RDH method, HS, is first proposed by Ni *et al.* in [6]. Principally, the highest two adjacent bins in the image histogram are used for expansion to embed data, while other bins are shifted to provide vacancies. The existence of these histogram vacancies guarantees the reversibility of the RDH method. Different from DE, Thodi and Rodriguez [10] proposed the so-called PEE embedding method to employ the prediction-error of pixels for data embedding instead of the difference of adjacent pixels. In PEE, some bins in the prediction-error histogram are selected for expansion and other bins are shifted for reversibility. Since the predictor used for pixel prediction can further exploit the correlations of neighboring pixels, the embedding performance of PEE outperforms previous methods.

PEE is widely studied and applied due to its efficient capacity-distortion trade-off, and many researchers concentrate on its improvements. Some works focus on designing accurate predictors to generate sharp histograms in PEE. The commonly used predictors include median edge detection (MED) [10], rhombus prediction [18] and gradient-adjusted predictor (GAP) [11]. Among these predictors, the rhombus prediction is widely used as a full-enclosing prediction scheme which employs four closest adjacent neighbors of the target pixel to achieve accurate prediction. In addition, double-layered embedding is utilized in conjunction with rhombus prediction to ensure reversibility. Besides, various strategies are proposed for better performance on PEE, such as, adaptive embedding [11, 18], context modification [12], pairwise PEE [19] and multiple histogram modification (MHM) [21], etc. Among them, adaptive embedding is widely used to better exploit the image redundancy. After sorting pixels based on their local complexity, pixels in smooth areas are selected for data embedding. Since these pixels can be accurately predicted, sharper histograms can be obtained. Pairwise PEE is initially proposed by Ou *et al.*[19], which takes the advantages of the correlations among prediction-errors through a two-dimensional (2D) histogram. In this method, the 2D histogram is generated by combining every two adjacent prediction-errors into a pair and counting their occurrence. Besides, the embedding performance of PEE also can be improved by designing more effective modification manners of the histogram. For instance, a RDH strategy called MHM is proposed in [21]. This method achieves better performance by utilizing more than ten histograms with approximately equal number of prediction-errors instead of a single histogram used in former RDH methods, and optimal expansion bins are adaptively selected in these histograms.

Recently, a new kind of RDH methods is proposed which incorporates pixel-value-ordering (PVO) into PEE and realizes high-fidelity marked image with moderate capacity. Li *et al.* first proposed the PVO-based method [13], in which the prediction-error is calculated based on the sorted pixels in an image block. Specifically, a given cover image is first divided into non-overlapping blocks, and then the prediction-error is calculated in each block by considering the second largest/smallest pixel as the prediction of the largest/smallest pixel. Finally, the maximum and the minimum in each block are used for data embedding. Notice that, after data embedding, for each block, the maximum is either increased by 1 or unchanged, and the minimum is either decreased by 1 or unchanged. Therefore, the PVO of pixels in each block is maintained, which ensures the reversibility. Besides, a block selection strategy is also employed for performance enhancement by selecting blocks with low complexity to embed data. Since the correlation of pixels within each block is properly exploited, high fidelity marked image is obtained in this PVO-based method [13]. Since then, many PVO-based RDH schemes have been proposed. Peng *et al.* proposed the improved PVO method [14], called IPVO in this paper, which improves the original PVO-based method [13] by doing prediction based on not only the pixel values but also their locations. More smooth blocks are used for embedding including the blocks in which the second largest/smallest pixel equals the largest/smallest pixel. That is to say, the maximum/minimum whose prediction-error is 0 are used for embedding in [14]. Notice that, these pixels are not utilized for embedding in original PVO-based method [13] and they are especially suitable for reversible embedding. Therefore, the performance is improved compared to the original PVO [13]. In [15], Ou *et al.* proposed a method called PVO-k, which sets all the k pixels that equal the maximum/minimum in a block as a unit and modifies all the pixels in the unit at the same time. The original PVO-based method corresponds to the case $k = 1$, i.e., the PVO-1 method. In PVO-k, by combining PVO-1 with PVO-2 for performance optimization, the embedding performance is improved compared to the original PVO [13]. Later on, the PVO-based technique is further developed by Wang *et al.* in [16]. Instead of using equal-sized blocks, they designed a dynamic blocking strategy, in which the smooth areas are divided into small sized blocks to increase EC and the complex areas are divided into large sized blocks to decrease ED. In [17], Qu *et al.* proposed to change the block-wise prediction manner used in former PVO-based methods into a pixel-wise prediction manner, called pixel-based PVO (PPVO). Each pixel is predicted by its context pixels in the down right direction, and then every pixel has the chance to embed one data bit, so that the embedding capacity is enlarged. Recently, a novel RDH scheme is proposed by Kim *et al.* [20] in which the half-enclosing predictor used in PPVO is replaced by a more accurate full-enclosing based one. Two prediction values of the target pixel are generated by the predictor, including a high estimation value and a low estimation value. In this way, the two prediction-error histograms are skewed, which means not symmetrical. Then, to reduce shifting distortion compared to the symmetric histograms, only the short tails are used to embed data for better performance.

However, the performance of Kim *et al.*'s method [20] is still not satisfactory enough since the relationship between pixel complexity and pixel embeddability is not fully exploited. In fact, the prediction-errors of pixels with different local complexity hold different distributions. For example, based on Kim *et al.*'s method, in the first layer of the 512×512 sized gray-scale image Lena, when the pixel complexity is 50, 31.5% of its prediction-errors are 0, while only 15.96% of the prediction-errors are 0 when the pixel complexity is 100. Therefore, the embedding performance can be further improved by utilizing the difference of pixel complexity in a more comprehensive way. In this paper, inspired by [18] and [20], a novel PVO-based RDH scheme using adaptive multiple histograms generation and modification is proposed. First, a PVO-based predictor is employed to conduct pixel prediction. Then, by adaptively choosing several thresholds based on image context, prediction-errors for pixels with different complexity levels are grouped to generate multiple histograms. Finally, different histograms are modified in different manners for data embedding based on their own statistical characteristics. Note that, to optimize the embedding performance, the optimal thresholds as well as the modification mechanisms of different histograms are determined based on a minimization problem. Besides, performance improvement can be gained by extending the proposed RDH scheme to the 2D space based on pairwise modification. Experimental results show that the proposed method outperforms the recently proposed Kim *et al.*'s method as well as some other state-of-the-art RDH methods.

The rest of the paper is organized as follows. In Section 2, some existing PVO-based RDH methods are briefly introduced. Then, the proposed RDH scheme based on adaptive multiple histograms generation and modification is presented in Section 3. The experimental results are given in Section 4. Finally, in Section 5, the conclusion is drawn.

2. Related works

In this section, as a preparation, four PVO-based RDH methods are briefly reviewed, including the original PVO [13], IPVO [14], PPVO [17] and a recently proposed method of Kim *et al.* [20].

2.1. The original PVO [13]

The original PVO-based RDH method is proposed by Li *et al.* [13]. In this method, PVO is incorporated into PEE, and a high-quality marked image is achieved by reducing the number of shifted pixels. By modifying the maximum and minimum of each image block, the secret data is embedded into the cover image. The details are as follows.

First, for a given image, divide it into non-overlapping blocks with size of $n_1 \times n_2$. Then, pixels in each divided block are sorted in ascending order based on their values, generating the pixel sequence $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$, where $n = n_1 \times n_2$ and $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ is the unique one-to-one mapping such that $x_{\sigma(1)} \leq \cdots \leq x_{\sigma(n)}$, in which

$$\sigma(i) < \sigma(j), \quad \text{if } x_{\sigma(i)} = x_{\sigma(j)} \text{ and } i < j. \tag{1}$$

Next, in a pixel block, since the second largest pixel is the closest one to the maximum, it is used to predict the maximum, and the prediction-error is defined as

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

Note that, the above defined prediction-error $e_{\max}$ is always positive, and it takes value in $[0, \infty)$. For example, Fig. 1 shows the histogram of $e_{\max}$ of the standard 512×512 gray-scale Lena image. The block size is set to 3×3 and the histogram is generated by counting the occurrence of $e_{\max}$ in all the blocks. Since the bin 1 is always the highest one in the histogram of $e_{\max}$, it is then selected for expansion in the original PVO. Meanwhile, the bin 0 is unchanged while other bins are shifted to the increasing direction by 1. That is to say, the prediction-error $e_{\max}$ is modified to $\widetilde{e}_{\max}$ in the following way to embed data.

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

where $b \in \{0, 1\}$ is one data bit to be embedded. Accordingly, the maximum is modified as

$$\widetilde{x}_{\sigma(n)} = x_{\sigma(n-1)} + \widetilde{e}_{\max}. \tag{4}$$

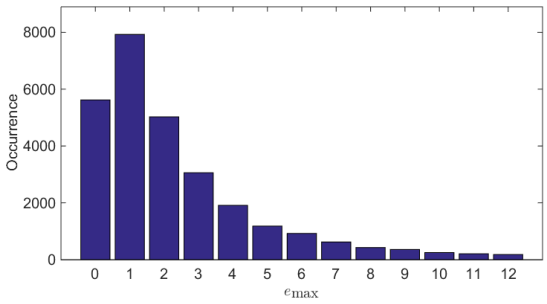


Figure 1: Histogram of $e_{\max}$ defined in Eq. (2) for the standard 512×512 gray-scale image Lena with block size of 3×3.

Moreover, in [13], to enlarge EC, the minimum of each block is utilized for data embedding as well. In detail, the second smallest pixel is used to predict the minimum, and the prediction-error is calculated by

$$e_{\min} = x_{\sigma(1)} - x_{\sigma(2)}. \tag{5}$$

Notice that, $e_{\min}$ is always negative and takes value in $(-\infty, 0]$. Similarly, the bin -1 in the histogram of $e_{\min}$ is selected for expansion. The bin 0 is unchanged and other bins are simply shifted. Specifically, the prediction-error $e_{\min}$ is modified to $\widetilde{e}_{\min}$ as follows to embed data.

$$\widetilde{e}_{\min} = \begin{cases} e_{\min}, & \text{if } e_{\min} = 0 \\ e_{\min} - b, & \text{if } e_{\min} = -1 \\ e_{\min} - 1, & \text{if } e_{\min} < -1 \end{cases} \tag{6}$$

where $b \in \{0, 1\}$ is one data bit to be embedded, and the marked minimum is calculated by

$$\widetilde{x}_{\sigma(1)} = x_{\sigma(2)} + \widetilde{e}_{\min}. \tag{7}$$

In this way, at most two data bits can be embedded into a pixel block by modifying both the maximum and minimum. Moreover, except for the maximum and minimum, other pixels are unchanged in the embedding process. That is to say, $\widetilde{x}_{\sigma(i)} = x_{\sigma(i)}, 2 \le i \le n-1$. Meanwhile, $n = n_1 \times n_2 \ge 4$, which ensures the maximum and the minimum of each block can be utilized for data embedding.

According to Eqs. (4) and (7) by data embedding, the maximum $x_{\sigma(n)}$ is either increased by 1 or unchanged, and the minimum $x_{\sigma(1)}$ is either decreased by 1 or unchanged. Therefore, the PVO of pixels in a block is unchanged after data embedding. In this way, the data extraction and image recovery process can be accordingly conducted. The marked image is first divided into non-overlapping blocks with size of $n_1 \times n_2$, and then pixels in each block are sorted in ascending order to generate the sequence $(\widetilde{x}_{\sigma(1)}, \ldots, \widetilde{x}_{\sigma(n)})$. As $x_{\sigma(i)}, 2 \le i \le n-1$ is unchanged in the embedding process, $\widetilde{x}_{\sigma(i)}$ is simply recovered as itself. Then, calculate the marked prediction-error $\widetilde{e}_{\max}$ as

$$\widetilde{e}_{\max} = \widetilde{x}_{\sigma(n)} - \widetilde{x}_{\sigma(n-1)}. \tag{8}$$

This value is exactly the one defined in Eq. (2). Consequently, the embedded bit and the original maximum in each block can be determined as:

- if $\widetilde{e}_{\max} = 0$, there is no embedded bit and the original maximum is just $\widetilde{x}_{\sigma(n)}$,
- if $\widetilde{e}_{\max} \in \{1, 2\}$, the embedded bit is $b = \widetilde{e}_{\max} - 1$ and original maximum is $\widetilde{x}_{\sigma(n)} - b$,
- if $\widetilde{e}_{\max} > 2$, there is no embedded bit and the original maximum is $\widetilde{x}_{\sigma(n)} - 1$.

The extraction and the recovery process for the minimum is similar with the case of the maximum and the obvious detail is omitted for brevity.

2.2. IPVO [14]

The original PVO-based RDH method takes the maximum and minimum of each block to embed data. However, the bin 0 in the generated histogram is not utilized for expansion while it is unmodified in data embedding. Thus, plenty of pixels in smooth area which are more suitable for reversible embedding are ignored. Inspired by this fact, Peng *et al.* proposed an improved version of the original PVO, called IPVO in this paper. In IPVO, more pixels in smooth area are utilized for expansion by taking the pixel locations into consideration.

The same as the case of PVO, in the data embedding process of IPVO, the cover image is first divided into non-overlapping blocks with size of $n_1 \times n_2$, and pixels in each block are sorted in ascending order to derive the sorted sequence $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$. Then, unlike the original PVO that directly uses the second largest pixel to predict the maximum and always gets positive prediction-errors, IPVO takes the locations of the second largest pixel and the maximum into consideration. By this means, a prediction-error related to a maximum is defined as

$$e_{\max} = x_u - x_v \tag{9}$$

where

$$\begin{cases} u = \min(\sigma(n), \sigma(n-1)) \\ v = \max(\sigma(n), \sigma(n-1)) \end{cases}. \tag{10}$$

Thus, the sign of $e_{\max}$ is determined by the relation of $\sigma(n)$ and $\sigma(n-1)$. Since σ is a one-to-one mapping, two cases may occur:

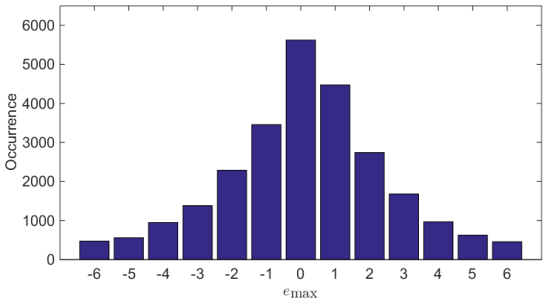


Figure 2: Histogram of $e_{\max}$ defined in Eq. (9) for the standard 512×512 sized gray-scale image Lena with block size of 3×3.

- if $\sigma(n) > \sigma(n-1)$, then $u = \sigma(n-1), v = \sigma(n)$, and $e_{\max} = x_{\sigma(n-1)} - x_{\sigma(n)} \leq 0$,
- if $\sigma(n) < \sigma(n-1)$, then $u = \sigma(n), v = \sigma(n-1)$, and $e_{\max} = x_{\sigma(n)} - x_{\sigma(n-1)} \geq 1$.

Notice that, $e_{\max} = 0$ only occurs in the case $\sigma(n) > \sigma(n-1)$, this is guaranteed by the property of σ in Eq. (1). Therefore, $e_{\max}$ takes values in $(-\infty, \infty)$, and the histogram of $e_{\max}$ obeys a Laplacian-like distribution centered at 0. Take the standard gray-scale image Lena with block size of 3×3 for example, the histogram of $e_{\max}$ is shown in Fig. 2. In this histogram, two bins 0 and 1 are selected for expansion in IPVO, i.e., the modification mechanism of $e_{\max}$ is

$$\widetilde{e}_{\max} = \begin{cases} e_{\max} + b, & \text{if } e_{\max} = 1 \\ e_{\max} - b, & \text{if } e_{\max} = 0 \\ e_{\max} + 1, & \text{if } e_{\max} > 1 \\ e_{\max} - 1, & \text{if } e_{\max} < 0 \end{cases} \tag{11}$$

where $b \in \{0, 1\}$ denotes the bit to be embedded. Accordingly, the maximum is modified by

$$\widetilde{x}_{\sigma(n)} = x_{\sigma(n-1)} + |\widetilde{e}_{\max}| . \tag{12}$$

Besides, IPVO also modifies the minimum to increase EC. Specifically, based on the sorted pixels, the prediction-error related to the minimum is defined as

$$e_{\min} = x_s - x_t, \tag{13}$$

where

$$\begin{cases} s = \min(\sigma(1), \sigma(2)) \\ t = \max(\sigma(1), \sigma(2)) \end{cases} . \tag{14}$$

Similarly, the sign of $e_{\min}$ is determined as follows:

- if $\sigma(1) > \sigma(2)$, then $s = \sigma(2), t = \sigma(1)$, and $e_{\min} = x_{\sigma(2)} - x_{\sigma(1)} \geq 1$,
- if $\sigma(1) < \sigma(2)$, then $s = \sigma(1), t = \sigma(2)$, and $e_{\min} = x_{\sigma(1)} - x_{\sigma(2)} \leq 0$.

In this case, the bins 0 and 1 are selected for expansion as well, and $e_{\min}$ is modified as $e_{\max}$ in the following way.

$$\widetilde{e}_{\min} = \begin{cases} e_{\min} + b, & \text{if } e_{\min} = 1 \\ e_{\min} - b, & \text{if } e_{\min} = 0 \\ e_{\min} + 1, & \text{if } e_{\min} > 1 \\ e_{\min} - 1, & \text{if } e_{\min} < 0 \end{cases} . \tag{15}$$

Accordingly, the marked value of the minimum is calculated by

$$\widetilde{x}_{\sigma(1)} = x_{\sigma(2)} - |\widetilde{e}_{\min}| . \tag{16}$$

Based on Eqs. (12) and (16), like the original PVO-based RDH method [13], the PVO of each block is unchanged in the data embedding process. Consequently, the data extraction and image recovery process can be accordingly conducted by following steps. First, divide the marked image into blocks sized of $n_1 \times n_2$ and sort pixels in each block with ascending order and derive the sequence $(\widetilde{x}_{\sigma(1)}, \ldots, \widetilde{x}_{\sigma(n)})$. As pixels are unchanged in the embedding process except for the maximum and minimum, i.e., $\widetilde{x}_{\sigma(i)} = x_{\sigma(i)}, 2 \leq i \leq n-1$, they are directly recovered as themselves. Then, the marked prediction-error of the maximum $\widetilde{e}_{\max}$ is calculated by

$$\widetilde{e}_{\max} = \widetilde{x}_u - \widetilde{x}_v. \tag{17}$$

The same definitions of u and v in Eq. (10) are also used here, and this prediction-error is exactly the one defined in Eq. (9). Therefore, for each block, the embedded bit and the original maximum can be determined as:

- if $\widetilde{e}_{\max} \in \{0, 1\}$, the embedded bit is $b = 0$ and the original maximum is $\widetilde{x}_{\sigma(n)}$,
- if $\widetilde{e}_{\max} \in \{-1, 2\}$, the embedded bit is $b = 1$ and the original maximum is $\widetilde{x}_{\sigma(n)} - 1$,
- if $\widetilde{e}_{\max} < -1$ or $\widetilde{e}_{\max} > 2$, there is no embedded bit and the original maximum is $\widetilde{x}_{\sigma(n)} - 1$.

The extraction and recovery process of the minimum is similarly with the case of the maximum. Thus, it is omitted.

In summary, by taking the relation of pixel locations into error prediction process, bins 0 and 1 can be selected for expansion. Therefore, more blocks in smooth area are utilized for data embedding, leading to the improvement of embedding performance compared to the original PVO [13].

2.3. PPVO [17]

x	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}

Figure 3: Context pixels (q=15) of x in PPVO.

Since PVO and IPVO conduct prediction in a block-by-block manner where only two pixels are used to embed data in each block, and then at most two bits can be embedded in each block. Therefore, when the block size is large, pixels in smooth areas are not fully exploited. To better use the smooth pixels and increase EC, the so-called PPVO [17] proposed a pixel-by-pixel prediction manner with PVO, in which each cover pixel can be used to embed data. Specific process of this method is as follows. Fig. 3 shows context pixels consisted of 15 pixels used for the prediction of the target pixel x. For a given parameter q, the first q context pixels are used to predict. First, define the maximum and minimum of the context pixels as

$$\begin{cases} c_{\max} = \max\{c_i : 1 \leq i \leq q\} \\ c_{\min} = \min\{c_i : 1 \leq i \leq q\} \end{cases}. \tag{18}$$

Notice that, to avoid the overflow or underflow problem in the embedding process of the RDH method, pixels with values of 0 and 255 are changed to 1 and 254 in the preprocessing phase. Then, three cases are taken into consideration for prediction and data embedding.

Case **1**: if $c_{\max} > c_{\min}$. In this case, the prediction-error e is defined by

$$e = \begin{cases} x - c_{\max}, & \text{if } x \geq c_{\max} \\ x - c_{\min}, & \text{if } x \leq c_{\min} \end{cases}. \tag{19}$$

The sign of e is determined by x, i.e., when $x \geq c_{\max}$, $e \geq 0$, and when $x \leq c_{\min}$, $e \leq 0$. Besides, to ensure the reversibility, pixels satisfying $c_{\min} < x < c_{\max}$ are not utilized for embedding, and they are unchanged in embedding process. Next, e is modified to $\widetilde{e}$ to embed data.

$$\widetilde{e} = \begin{cases} e + b, & \text{if } x = c_{\max} \\ e - b, & \text{if } x = c_{\min} \\ e + 1, & \text{if } x > c_{\max} \\ e - 1, & \text{if } x < c_{\min} \end{cases}. \tag{20}$$

where $b \in \{0, 1\}$ denotes the bit to be embedded. Accordingly, the marked value of x is calculated by

$$\widetilde{x} = x + \widetilde{e}. \tag{21}$$

Since pixels are predicted in the raster-scan order in the embedding process, the extraction and recovery process is performed in reverse order to the embedding. In this way, the context pixels of x are same both in the embedding phase and the extraction phase, and the values of $c_{\max}$ and $c_{\min}$ as well as their relation are unchanged. Therefore, the extraction and recovery process in this case can be accordingly implemented. First, the prediction-error is computed by

$$\widetilde{e} = \begin{cases} \widetilde{x} - c_{\max}, & \text{if } \widetilde{x} \geq c_{\max}, \\ \widetilde{x} - c_{\min}, & \text{if } \widetilde{x} \leq c_{\min}, \end{cases} \tag{22}$$

Then, the embedded bit and the original pixel can be obtained as:

- if $\widetilde{e} = 0$, the embedded bit is $b = 0$ and the original pixel is $\widetilde{x}$,
- if $\widetilde{e} \in \{-1, 1\}$, the embedded bit is $b = 1$ and the original pixel is $\widetilde{x} - \widetilde{e}$,
- if $\widetilde{e} < -1$, there is no embedded bit and the original pixel is $\widetilde{x} + 1$,
- if $\widetilde{e} > 1$, there is no embedded bit and the original pixel is $\widetilde{x} - 1$.

Case **2**: if $c_{\max} = c_{\min}$ and $c_{\min} = 254$. Considering about the special procedure in the preprocessing phase, in this case, only pixels whose value is 254 are used for embedding. And the marked pixel value is calculated by $\widetilde{x} = 254 + b$, in which $b \in \{0, 1\}$ is a data bit to be embedded. Therefore, two cases are included in the extraction and recovery process of case 2:

- if $\widetilde{x} = 254$, the embedded bit is $b = 0$ and the original pixel is 254,
- if $\widetilde{x} = 255$, the embedded bit is $b = 1$ and the original pixel is also 254.

Case **3**: if $c_{\max} = c_{\min}$ and $c_{\min} < 254$. In this case, the prediction-error is defined by

$$e = x - c_{\min}, \quad \text{if } x \leq c_{\min}. \tag{23}$$

Note that, only pixels with value $x \leq c_{\min}$ are considered for embedding in this case, and the prediction-error e satisfies $e \leq 0$. Therefore, e is modified to $\widetilde{e}$ by

$$\widetilde{e} = \begin{cases} e - b, & \text{if } x = c_{\min}, \\ e - 1, & \text{if } x < c_{\min}, \end{cases} \tag{24}$$

Accordingly, the marked value of x is obtained based on Eq. (21).

Furthermore, the extraction and recovery process can be directly conducted by following steps, since the relationship of $c_{\max}$ and $c_{\min}$ can be maintained after data embedding. The prediction-error is calculated at first by

$$\widetilde{e} = \widetilde{x} - c_{\min}, \quad \text{if } \widetilde{x} \leq c_{\min}. \tag{25}$$

Based on the prediction-error $\widetilde{e}$, the embedded bit as well as the original pixel are determined as follows:

- if $\widetilde{e} \in \{0, -1\}$, the embedded bit is $b = -\widetilde{e}$ and the original pixel is $\widetilde{x} - \widetilde{e}$,
- if $\widetilde{e} < -1$, there is no embedded bit and the original pixel is $\widetilde{x} + 1$.

Notice that, the optimal context pixels number q is experimentally determined. For a fixed EC with a given cover image, by testing all the values of q from 1 to 15, the one generating marked image with highest PSNR is finally selected as the optimal one, and the marked image with highest PSNR is determined as the embedding result. In summary, compared to PVO and IPVO, the PPVO method is of better performance, especially in the case with large EC.

2.4. Kim et al.'s method [20]

Unlike traditional RDH methods that conduct modification based on histograms with symmetric Laplacian or Gaussian distributions, a recent work [20] proposed to generate the so-called skewed histograms to reduce the number of shifted pixles. This type of histograms are generated by using extreme predictors based on PVO.

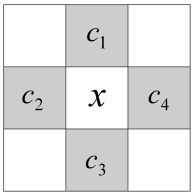


Figure 4: Context pixels $\{c_1, c_2, c_3, c_4\}$ used for the prediction of x.

The embedding process of [20] is introduced here. First of all, four pixels $\{c_1, c_2, c_3, c_4\}$ nearby the target pixel x are used for prediction, as shown in Fig. 4. Then, based on the idea of PVO, $\{c_1, c_2, c_3, c_4\}$ are sorted in ascending order to obtain $p_1 \leq p_2 \leq p_3 \leq p_4$. Next, for the target pixel x, a pair of extreme predictors including a higher estimation, denoted as $\hat{p}_h$, and a lower estimation, denoted as $\hat{p}_l$, can be defined based on these sorted pixels. Therefore, a right skewed histogram can be generated by counting the occurrence of $e = x - \hat{p}_l - 1$, and a left skewed histogram can be generated by counting the occurrence of $e = x - \hat{p}_h$. To reduce the number of shifted pixels, only the short tails of the histograms are used for embedding. Specifically, only pixels satisfying $x \geq \hat{p}_h$ and $x \leq \hat{p}_l$ are employed for embedding, and other pixels remain unchanged. Since $\hat{p}_h \geq \hat{p}_l$ always holds, two cases are taken into consideration, i.e., $\hat{p}_h > \hat{p}_l$ and $\hat{p}_h = \hat{p}_l$. When $\hat{p}_h > \hat{p}_l$, the prediction-error e is calculated by

$$e = \begin{cases} x - \hat{p}_l - 1, & \text{if } x \leq \hat{p}_l \\ x - \hat{p}_h, & \text{if } x \geq \hat{p}_h \end{cases}. \tag{26}$$

And then e is expanded or shifted to $\widetilde{e}$ by

$$\widetilde{e} = \begin{cases} e + b, & \text{if } e = 0 \\ e - b, & \text{if } e = -1 \\ e + 1, & \text{if } e > 0 \\ e - 1, & \text{if } e < -1 \end{cases}. \tag{27}$$

Accordingly, the marked pixel value $\widetilde{x}$ is determined by

$$\widetilde{x} = x + \widetilde{e}. \tag{28}$$

On the other hand, when $\hat{p}_h = \hat{p}_l$, to ensure the reversibility, $\hat{p}_l$ is firstly updated by $\hat{p}_l = \hat{p}_l - 1$. Then, the same procedures described in Eqs. (26-28) can be employed for data embedding.

Essentially, according to the prediction method, the final histogram used for embedding is a sharp symmetric histogram consisted by two skewed histograms. The left part of the symmetric histogram with $e \in (-\infty, -1]$ is corresponding to the left tail of the right skewed histogram obtained by $e = x - \hat{p}_l - 1$, and the right part with

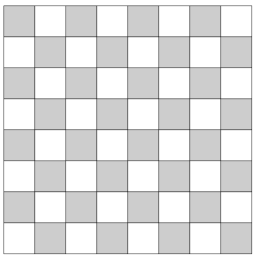


Figure 5: Division mode of double layers.

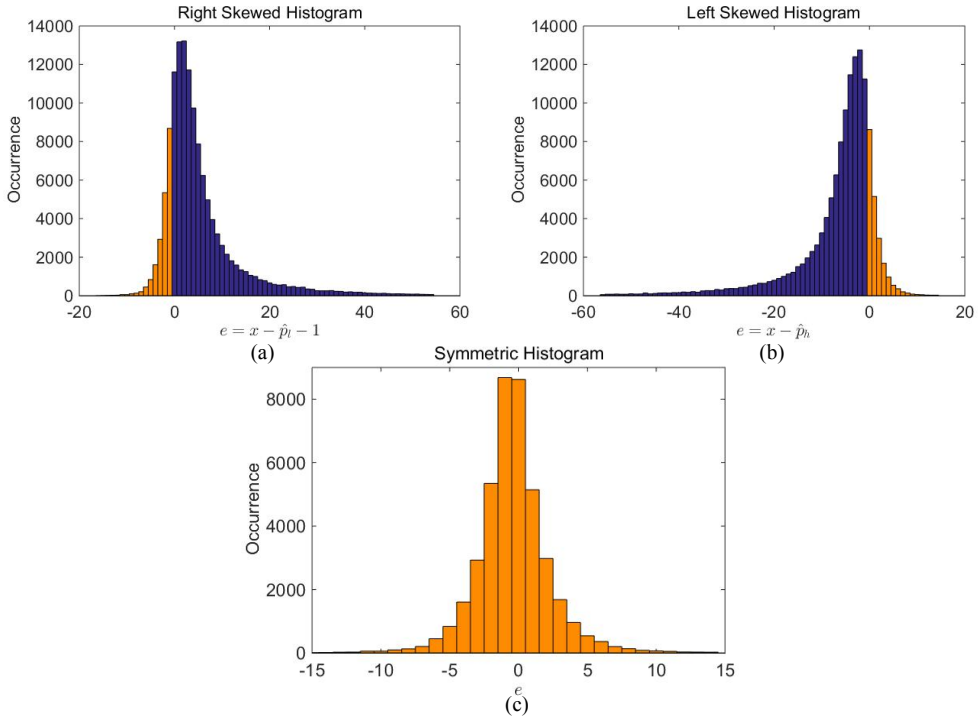


Figure 6: Skewed histograms and symmetric histogram of the first layer of the standard 512×512 sized gray-scale image Lena using predictor 1 in Eq. (29).

$e \in [0, \infty)$ is corresponding to the right tail of the left skewed histogram obtained by $e = x - \hat{p}_h$. Finally, in this symmetric histogram, bins -1 and 0 are used for expansion, and other bins are shifted to create vacancies.

Besides, double-layered embedding is utilized. Fig. 5 shows the division mode of double layers, where the shadow pixels belong to the first layer, and the blank pixels belong to the second layer. Note that, the embedding of the second layer is performed after the first layer is embedded, while the second layer is extracted and recovered before conducting extraction in the first layer. Therefore, the reversibility can be ensured.

As for specific predictors, three different extreme predictors are given based on the sorted pixels $p_1 \le p_2 \le p_3 \le p_4$ as

$$\begin{aligned}
&\text{Predictor 1:} && \hat{p}_h = p_4, && \hat{p}_l = p_1, \\
&\text{Predictor 2:} && \hat{p}_h = \Big[\frac{p_4 + p_3}{2}\Big], && \hat{p}_l = \Big[\frac{p_1 + p_2}{2}\Big], \\
&\text{Predictor 3:} && \hat{p}_h = \Big[\frac{p_4 + p_3 + p_2}{3}\Big], && \hat{p}_l = \Big[\frac{p_1 + p_2 + p_3}{3}\Big].
\end{aligned} \tag{29}$$

where $[\cdot]$ is the rounding function. In practice, all these three predictors are used to embed data, and the one with

best embedding performance will be selected, corresponding marked image with highest PSNR will be chosen as the final embedding result. In order to make the histogram generation process clearer, an example of how to obtain the sharp symmetric histogram based on two skewed histograms is given. By employing predictor 1 in Eq. (29) on the prediction of the first layer of standard 512×512 sized gray-scale image Lena, two skewed histograms and a symmetric histogram are obtained and shown in Fig. 6.

In the extraction and recovery process, since the context pixels of x are from another layer and they are same in the embedding and extraction process, the values of $\hat{p}_h$ and $\hat{p}_l$ are unchanged. Similarly, if $\hat{p}_h = \hat{p}_l$, the $\hat{p}_l$ is updated to $\hat{p}_l - 1$ before conducting extraction. Therefore, the marked prediction-error $\widetilde{e}$ can be calculated by

$$\widetilde{e} = \begin{cases} \widetilde{x} - \hat{p}_h, & \text{if } \widetilde{x} \geq \hat{p}_h \\ \widetilde{x} - \hat{p}_l - 1, & \text{if } \widetilde{x} \leq \hat{p}_l \end{cases} . \tag{30}$$

Based on the values of $\widetilde{e}$, the data bits can be extracted and the cover image can be losslessly recovered. Detail procedures are omitted here, as they are similar to the extraction in above methods.

In summary, through utilizing the PVO-based extreme predictors, a sharp histogram can be generated, leading to the improvement of embedding performance. Experimental results reported in [20] show that this method outperforms some state-of-the-art RDH methods. However, the embedding performance can be further improved since all the selected pixels with different local complexity are processed based on a single histogram and a common modification mechanism. Actually, pixels with different local complexity hold different properties. Therefore, to better exploite the different statistical characteristics of pixels with different local complexity and gain performance improvement, we propose a new RDH scheme using adaptive multiple histograms generation and modification. The details of the proposed RDH scheme are given in the next section.

3. Proposed scheme

Since the different statistical properties of pixels with different local complexity are not fully considered in Kim *et al.*'s method [20], to improve the embedding performance, we propose a novel RDH method based on an adaptive multiple histograms generation and modification strategy. First, the proposed multiple histograms generation and modification strategy is described in a general way. Then, to optimize the embedding performance, an optimal parameters determination method is given. Finally, to make further improvement, the proposed RDH scheme is extended based on pairwise modification.

3.1. Adaptive multiple histograms generation and modification

As mentioned above, in Kim *et al.*'s method [20], all the pixels selected for embedding are modified based on the same modification mechanism regardless of their local complexity difference, leading to the decrease of embedding performance. This phenomenon is verified by the results shown in Table 1. This table gives an example of the relationship between the noise level and embedding performance in the first layer of standard 512×512 sized gray-scale image Lena based on Kim *et al.*'s method. Note that, pixel local complexity is measured by noise level, and its value is calculated according to [20]. Predictor 1 in Eq. (29) is used here for prediction. For each range of noise level, pixels with noise level falling within this range will be predicted, and a symmetric histogram can be derived. Since bins -1 and 0 are used for expansion, EC equals the sum of these two bins. And ED is estimated as half of EC together with the number of all the shifted pixels. Thus, ED/EC represents the distortion caused by embedding one bit. Consequently, the smaller the ED/EC, the better the embedding performance. As for expansion ratio, it measures the proportion of prediction-errors equal -1 or 0 to the total number of prediction-errors, and larger ratio can bring better embedding performance.

According to this table, with the increase of the noise level, ED/EC is sharply increasing, indicating that the embedding efficiency is sharply declining. At the same time, the expansion ratio is decreasing, that is to say the peak of the histogram is getting smaller, and the distribution of histograms generated from pixels with different noise level are different from each other. Therefore, to better utilize the different properties of pixels with different levels of complexity, we propose to generate multiple histograms based on pixels with different ranges of noise levels, and design different modification mechanisms for them.

Table 1: The relationship between noise level and embedding performance in the first layer of standard 512×512 sized gray-scale image Lena based on Kim *et al.*'s method [20].

Noise level	ED/EC	EC (bits)	Expansion ratio
$[0, 40]$	1.2308	988	57.78%
$(40, 80]$	1.6332	6672	46.88%
$(80, 120]$	1.7703	4984	44.05%
$(120, 160]$	2.0366	1992	39.42%
$(160, 200]$	2.2979	871	35.74%

Here is a simple example. It is still carried out on the first layer of standard gray-scale image Lena. Fig. 7 shows the comparison results between Kim *et al.*'s method that based on single histogram and proposed method based on multiple histograms. For the sake of simplicity, only two histograms are used in this example. In order to generate two histograms, two thresholds T_1 and T_2 are utilized. Pixels whose noise level belongs to $[0, T_1]$ are used to generate the first histogram, and the second histogram is obtained based on pixels with noise level falling within $(T_1, T_2]$. For simple implementation, T_1 is chosen as $3/4 \times T_2$, and $T_2 \in \{40, 80, \ldots, 400\}$. As shown in Table 1, the peak of the histogram is getting smaller with the increase of noise level. So that, in the double-histogram-based method, bins -1 and 0 are used for expansion only in the first histogram, and bins -2 and 1 are used for expansion in the second histogram to reduce the number of shifted pixels. Besides, only T_2 is used in Kim *et al.*'s method to obtain a histogram in the range of $[0, T_2]$, and bins -1 and 0 are used for expansion.

As shown in Fig. 7, the red line corresponds to the results of Kim *et al.*'s method, and the black line corresponds to the results of proposed method. Compared to Kim *et al.*'s method that based on single histogram, the values of ED/EC decreased in the proposed double-histogram-based method, indicating that the embedding efficiency is improved by using different modification mechanisms on two histograms generated by pixels with different noise levels. As for EC, it is a little smaller in the proposed method than in Kim *et al.*'s method when T_2 is fixed. However, the thresholds T_1 and T_2 used in the proposed method are not adaptively chosen, and it can be demonstrated that the proposed method can achieve performance improvement under the same payload with Kim *et al.*'s method by adaptively choosing optimal thresholds for a cover image. Therefore, to make full use of the different properties of pixels with different noise levels, we proposed an RDH scheme using adaptive multiple histograms generation and modification.

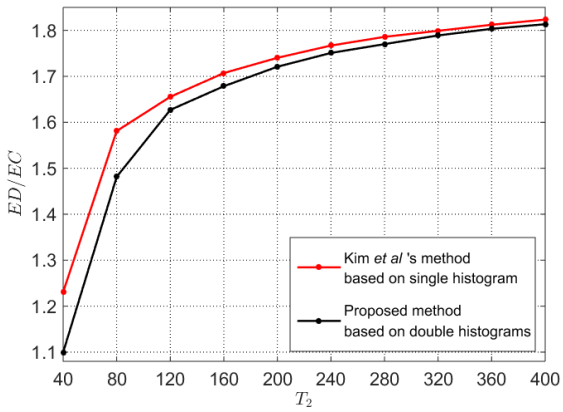


Figure 7: Performance comparison between Kim *et al.*'s method based on single histogram and the proposed based on double histograms on the first layer of standard 512×512 sized gray-scale image Lena.

Now, our idea is presented in detail. First, pixels in the cover image are divided into $n + 1$ classes based on their noise levels. n thresholds $0 \leq T_1 \leq \cdots \leq T_n$ are needed in this process. Pixels whose noise level belongs to $[0, T_1]$ are classified into the first class, and pixels with noise levels in $(T_1, T_2]$ belong to the second class, and so on for

Table 2: Correspondence between noise level, prediction-error histogram and expansion bins.

Noise level	Prediction-error histogram	Expansion bins
$[0, T_1]$	h_1	$-e_1 - 1$ & e_1
$(T_1, T_2]$	h_2	$-e_2 - 1$ & e_2
$\cdots$	$\cdots$	$\cdots$
$(T_{n-1}, T_n]$	h_n	$-e_n - 1$ & e_n
(T_n, ∞)	–	–

other classes. The last class is consisted by pixels whose noise level is bigger than T_n. Then, the prediction-errors of these pixels are calculated based on the PVO-based predictor 1 in Kim *et al.*'s method, as given in Eqs. (26), (29). Therefore, n symmetric histograms can be generated from the first n pixel classes, respectively. Based on the analysis above, we tend to utilize different modification mechanisms on different histograms, since they have different statistical properties. The correspondence between noise level, histogram and expansion bins is shown in Table 2. $\{h_k : k = 1, \cdots, n\}$ is the histogram based on pixels in the k-th range of noise level, and corresponding expansion bins used in the k-th histogram are $-e_k - 1$ and e_k with $e_k \geq 0$. To reduce the number of shifted pixels, bins inside the interval of $(-e_k - 1, e_k)$ remain unchanged and bins outside the interval of $[-e_k - 1, e_k]$ are shifted. Thus, the modification mechanism of the prediction-error e is written as

$$\widetilde{e} = \begin{cases} e, & \text{if } -e_k - 1 < e < e_k \\ e + b, & \text{if } e = e_k \\ e - b, & \text{if } e = -e_k - 1 \\ e + 1, & \text{if } e > e_k \\ e - 1, & \text{if } e < -e_k - 1 \end{cases} \tag{31}$$

where $\widetilde{e}$ is the modified value of e, and $b \in \{0, 1\}$ is one data bit to be embedded. Accordingly, the marked value of x is determined by

$$\widetilde{x} = \begin{cases} x, & \text{if } -e_k - 1 < e < e_k \\ x + b, & \text{if } e = e_k \\ x - b, & \text{if } e = -e_k - 1 \\ x + 1, & \text{if } e > e_k \\ x - 1, & \text{if } e < -e_k - 1 \end{cases}. \tag{32}$$

In this way, EC and ED of the k-th histogram h_k, denoted as EC_k and ED_k, can be calculated by

$$EC_k = h_k(-e_k - 1) + h_k(e_k) \tag{33}$$

and

$$ED_k = \frac{h_k(-e_k - 1) + h_k(e_k)}{2} + \sum_{e<-e_k-1} h_k(e) + \sum_{e>e_k} h_k(e). \tag{34}$$

Therefore, the optimal parameters can be obtained by solving the following minimization problem

$$\begin{cases} \text{minimize} & \dfrac{\sum_{k=1}^{n} ED_k}{\sum_{k=1}^{n} EC_k}, \\ \text{subject to} & \sum_{k=1}^{n} EC_k \geq P \end{cases} \tag{35}$$

where P is the required payload that need to be embedded into the cover image. Finally, for a given cover image and a required payload, the optimal parameters n and $\{(T_k, e_k) : k = 1, \ldots, n\}$ can be determined according to the solution of this minimization problem.

Now, the embedding performance between Kim *et al.*'s method [20] and the proposed method based on double histograms are compared again using optimal thresholds adaptively chosen according to Eq. (35). Suppose that the required payload P is 7660 bits. For Kim *et al.*'s method, bins -1 and 0 are used for expansion in the single histogram generated by pixels whose noise level belongs to $[0, T_2]$. From Table 1, when $T_2 = 80$, EC is 7660 bits, corresponding ED is 12111. And, the value of ED/EC is 1.5811. For proposed method, two histograms are generated based on pixels with noise levels falling within $[0, T_1]$ and $(T_1, T_2]$, respectively. And the expansion bin used in these two histograms are still chosen as $e_1 = 0$ and $e_2 = 1$, indicating that bins -1 and 0 are expanded in the first histogram and bins -2 and 1 are used for expansion in the second histogram. By choosing optimal thresholds for the proposed method, T_1 and T_2 are adaptively determined as $T_1 = 51$ and $T_2 = 103$. Under this situation, EC is 7677 bits, corresponding ED is 11280. Thus, the value of ED/EC is 1.4693. Obviously, ED/EC is greatly decreased, indicating that the embedding performance is improved by using proposed method with double histograms. Note that, in this example, only two histograms are employed, with the use of more histograms, the embedding performance can be further improved. In a word, by exploiting the different properties of pixels with different noise levels, the adaptive multiple histograms generation and modification strategy with optimal parameters can effectively enhance the embedding performance. Then, the problem is how to determine optimal parameters based on the minimization problem. The solution is given at next part.

3.2. Optimal parameters determination

In principle, exhaustive search can be used to determine the optimal parameters, including the number of histograms n, optimal thresholds $\{T_1, \ldots, T_n\}$ and optimal expansion bins $\{e_1, \ldots, e_n\}$. However, there are too many unknown parameters, which can produce a huge solution space and cause unacceptable algorithm complexity. Therefore, to simplify the process of optimal parameters determination, an alternative solution is given in this section.

Before introducing the simplified solution, the calculation method of noise level is given at the beginning. Note that, the noise level in previous subsection is calculated based on [20]. But, its calculation method is too complicated. Therefore, we tend to take a simple calculation way with fewer neighborhood pixels to achieve good performance. Specifically, the noise level of x is calculated referring to a 5×5 sized area surrounding it, shown in Fig. 8. And its value is defined by

$$\begin{aligned} NL(x) = {} & |a-b| + |b-c| + |c-d| + |d-a| + |e-d| + |d-g| + |g-f| + |f-e| + \\ & |g-j| + |j-i| + |i-h| + |h-g| + |c-l| + |l-k| + |k-j| + |j-c|. \end{aligned} \tag{36}$$

Obviously, $NL(x) \geq 0$. And, each pair of pixels that need to calculate the absolute difference are connected by a short line in Fig. 8. For reversibility guarantee, all the neighbors used in calculating noise level are from another layer. As we know, pixels in smooth areas hold higher redundancy and hence higher embedding priority while pixels in complex areas are less embeddable in RDH methods. Therefore, properly exploiting different characteristics of pixels with different noise levels can help us make full use of image redundancy and improve the embedding performance.

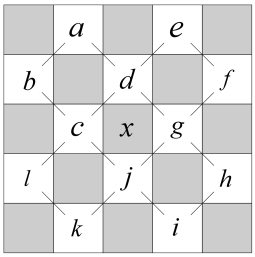


Figure 8: Context pixels used for calculating the noise level of x.

Now, the simplified optimal parameters determination method is given. First of all, some parameters are empirically set. Referring to Table 1, with the increase of noise level, the peak of the histogram is getting smaller, and corresponding tails are becoming more and more thick. If bins -1 and 0 are used for expansion in all these histograms, the distortion caused by shifting will grow bigger and bigger, and the embedding performance will be decreased.

Therefore, in order to reduce the number of shifted pixels while ensuring adequate capacity, the expansion bins are set as

$$e_k = k - 1, \quad k = 1, \ldots, n. \tag{37}$$

Specifically, bins -1 and 0 are used for expansion in h_1, and bins -2 and 1 are used for expansion in h_2, and so on. In this way, the number of shifted pixels in histograms with thick tails are reduced, since pixels whose prediction-error belongs to the interval of $(-e_k - 1, e_k)$ will remain unchanged. Consequently, ED/EC will be reduced.

After simplifying the minimization problem by setting values of $\{e_k : k = 1, \ldots, n\}$, we aim at finding a proper value for the number of histograms n, which is suitable for most images, for further simplification of the problem. Several experiments are conducted to help finding a proper n, and the results are shown in Fig. 9. Three subgraphs in the first column are the results of Lena's first layer under different n, and corresponding results of another standard 512×512 sized gray-scale image Boat are shown in the second column. To make the the subgraphs clearer, only part of the results are displayed, i.e., $EC \in [2000, 10000]$ for Lena and the $EC \in [4000, 11000]$ for Boat. Fig. 9 (a) gives the comparison result of embedding performance between $n = 1$ and $n = 2$. The horizontal axis represents EC, and the vertical axis represents ED/EC. The blue solid line shows the result in the case of $n = 1$, which means that only one threshold is utilized. Therefore, with the increase of the threshold, more pixels are selected for embedding, and EC increases while ED/EC increases too. As for the black dots, they are corresponding to the case of $n = 2$. Two thresholds T_1 and T_2 are used, and two different histograms are generated and further modified by different modification mechanisms determined based on Eq.(37). Specifically, bins -1 and 0 are expanded in the first histogram which is generated based on pixels with noise levels falling within $[0, T_1]$, and bins -2 and 1 are expanded in the second histogram generated based on pixels whose noise level belongs to $(T_1, T_2]$. Since $0 \le T_1 \le T_2 \le NL_{max}$, where NL_{max} means the maximum value of the noise level in a cover image, there are many choices for them. For each choice, two histograms are generated and modified, and the values of EC and ED/EC are calculated based on Eqs. (33-34). Thus, a black dot in the subgraph is obtained. Furthermore, all the black dots can be drew based on different choices of $\{T_1, T_2\}$. Obviously, the black dots below the blue line hold lower ED/EC under the same EC, meaning that the embedding performance is improved by using these choices of $\{T_1, T_2\}$. Since the performance under different choices of $\{T_1, T_2\}$ is quite different from each other, we tend to select out the ones with better performance. Suppose there is a specific $\{T_1, T_2\}$ generating EC_a and ED_a, if there exist another $\{T_1, T_2\}$ with EC_b and ED_b satisfies the following relationship, the first choice with EC_a and ED_a will be given up.

$$EC_b > EC_a \ \& \ ED_b < ED_a. \tag{38}$$

In this way, the threshold pairs with better embedding performance can be selected out. And the black dots with better performance are linked to a black solid line, shown in Fig. 9 (c). Two more experiments are conducted under the cases of $n = 3$ and $n = 4$. Fig. 9 (c) shows the comparison result between $n = 2$ and $n = 3$, and (e) shows the comparison result between $n = 3$ and $n = 4$. Observed from (c), lots of red dots ($n = 3$) are under the black line, indicating that better embedding performance can be achieved by employing three thresholds. And the thresholds $\{T_1, T_2, T_3\}$ with better performance also can be selected out based on Eq. (38), corresponding to the red line in (e). Additionally, in (e), the purple dots are the results in the case of $n = 4$. One can see that, only a small number of dots are under the red line, meaning that the embedding performance are not significantly improved when four thresholds are utilized. But, the computational complexity is dramatically increased in this case. Just as the results of image Lena, similar comparison results can be found in the second column derived from image Boat. Finally, considering about the trade-off between embedding performance and computational complexity, we can make the decision that it is unwise to use more than three thresholds in the proposed multiple histograms generation and modification based RDH scheme and $n = 3$ in this paper.

After determining the values of e_k and n, the solution space of the minimization problem Eq. (35) is much reduced. Therefore, it is easier for us to adaptively choose proper values of $\{T_1, T_2, T_3\}$ based on the image content. According to the above analysis, for a specific image, some threshold sets with better embedding performance can be filtered out based on Eq. (38). A large set consisted by these threshold sets can be obtained, denoted as D. And the final decision of $\{T_1, T_2, T_3\}$ is a set of thresholds chosen from D whose EC is just over the required payload.

In summery, three thresholds are employed in the proposed RDH scheme, and corresponding bins used for expansion in three histograms are bin 0, bins ± 1 and bins ± 2, respectively. Besides, the values of $\{T_1, T_2, T_3\}$ are adaptively determined based on the image content and the required payload. For a fixed payload, the threshold set which can

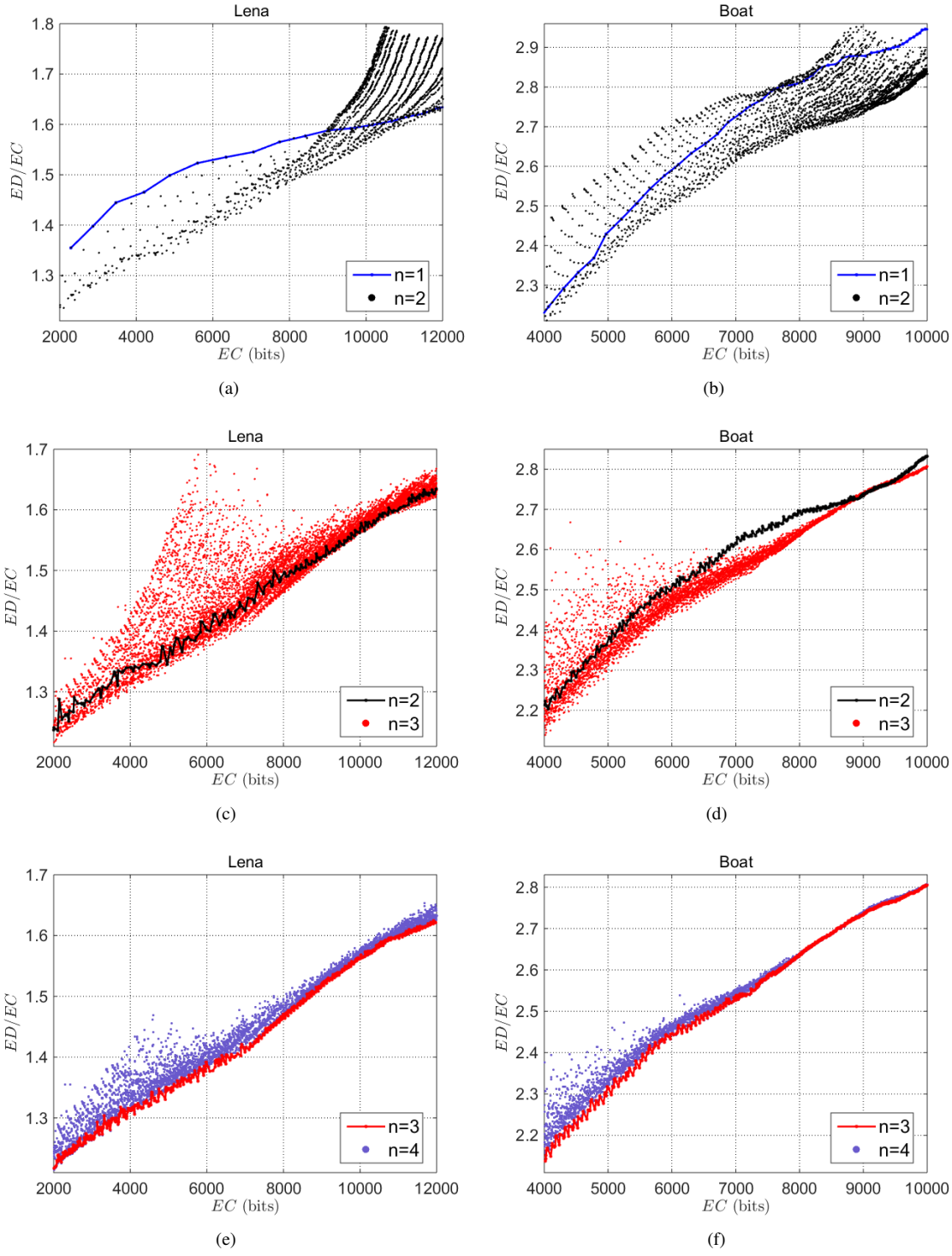


Figure 9: Embedding performance comparison with different n in Lena (the left column) and Boat (the right column).

generate a marked image with highest PSNR will be chosen, and corresponding marked image will be considered as the final embedding result.

3.3. Extension based on pairwise modification

In this part, the proposed method is further extended based on the idea of pairwise modification. To better exploit the correlations among prediction-errors, the traditional one-dimensional (1D) histogram modification approach can be extended to the 2D space. This method is proposed by Ou *et al.* in [19], called pairwise PEE, which combined two adjacent prediction-errors into a pair and modified them based on a 2D histogram modification mapping. The main idea of pairwise PEE is introduced at the beginning.

Firstly, the prediction-errors of the image pixel sequence $(x_1, \ldots, x_N)$ is calculated based on a certain predictor, recorded as $(e_1, \ldots, e_N)$. Then, every two adjacent prediction-errors are combined to form a pair (e_{2i-1}, e_{2i}), $i = 1, 2, \ldots, N/2$. By counting the occurrence of each prediction-error pair, a 2D prediction-error histogram can be obtained. Next, in order to expand or shift bins of the 2D histogram in a less distorted direction as much as possible, a unique 2D mapping is designed, as shown in Fig. 10. For simplicity, only the mapping in the first quadrant is given, as the other three quadrants take similar modification mechanisms. In fact, the modification mechanism of 1D histogram also can be transformed into a 2D space in an equivalent way by directly combining the modification of two adjacent prediction-errors. For example, two prediction-errors that equal 0 can form the prediction-error pair $(0, 0)$, and this pair can be mapped to $(0, 0), (0, 1), (1, 0)$ and $(1, 1)$. Thus, the main difference between pairwise PEE and 1D histogram modification is the special embedding manner of the prediction-error pairs $(0, 0)$ and $(1, 1)$. As for $(0, 0)$, it can be expanded into four pairs to embed two bits in 1D histogram based methods, while $(0, 0)$ is only expanded to three pairs including $(0, 0), (0, 1)$ and $(1, 0)$ to embed $\log_2 3$ bits. The expansion to $(1, 1)$, which has the largest distortion, is discarded. Although the number of the embedded bits is decreased from two bits to $\log_2 3$ bits, the embedding distortion is reduced by half. Besides, for $(1, 1)$, it is also expanded to embed one bit in pairwise PEE, while it is only shifted in the 1D methods. In a word, the special mapping is designed to reduce the embedding distortion at the expense of a less amount of bits can be embedded in the most smooth areas. More importantly, according to the experimental results in [19], the embedding performance is improved by using pairwise modification.

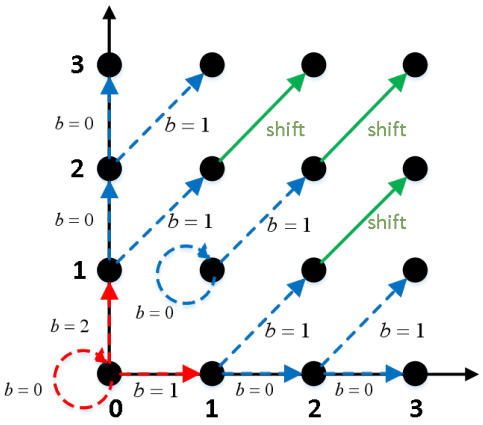


Figure 10: 2D modification mechanism of pairwise PEE.

Inspired by the idea of pairwise PEE, to derive sharp histograms as well as properly exploit the correlations of adjacent prediction-errors, we extends the proposed adaptive multiple histograms generation and modification strategy to 2D space for performance improvements through using pairwise modification. Details are presented as follows.

To implement pairwise modification, the first thing is to combine prediction-errors into pairs. According to previous analysis, the 1D histograms generated by pixels with different noise levels hold different statistical properties as well as different modification manners. Therefore, to increase the correlation of the prediction-errors within a pair, the pairing process is separately conducted in each class. As we know, a sharply distributed histogram can increase the capacity while decrease the distortion caused by shifting. By separately conducting pairing process, the two elements in a pair take similar statistical properties, hence higher correlation, which contributes to the generation of sharp 2D histogram as well as better embedding performance. After pairing prediction-errors, multiple 2D histograms can be generated by counting the occurrence of each pair.

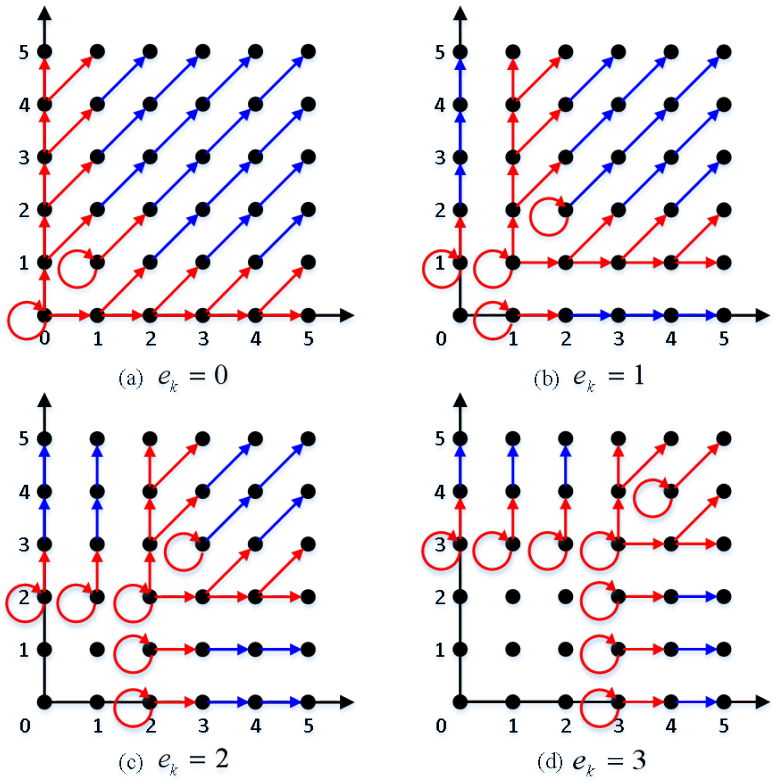


Figure 11: Examples of 2D mappings used in multiple 2D histograms.

For the sake of simplicity, the 2D mappings used in multiple classes are designed based on the modification manner used in pairwise PEE. Some examples are shown in Fig. 11. Note that, only the first quadrant is given, as the other three quadrants take similar modification mechanisms. The red arrows correspond to the expansion and the blue arrows is the process of shifting. Specifically, when $e_k = 0$, bin 0 in the 1D histogram is used for expansion, which is the same as pairwise PEE, so that the 2D mapping shown in Fig. 10 is directly used in this case. When $e_k = 1$, the expansion bin used in 1D histogram is bin 1, and the special mapping for $(0, 0)$ in (a) is moved to $(1, 1)$, as shown in (b). That is to say, the pair (1,1) can be expanded to (1,1), (1,2) and (2,1) to embed two bits. At the same time, the pair (2,2) are expanded to (2,2) and (3,3) to embed one bit. Since the prediction-error 0 is unchanged in this case, in pairs that contain 0, only the non-zero prediction-errors are expanded or shifted. Just like pair (1,0), it is expanded to (1,0) and (2,0), and 0 in this pair is never changed. Besides, when $e_k > 1$, other 2D mappings can be obtained in similar way. Two more examples are given in (c) and (d) corresponding to the case of $e_k = 2$ and $e_k = 3$. Finally, the modification of multiple 2D histograms can be conducted based on corresponding 2D mappings designed for different classes.

Notice that, the optimal parameters found in the 1D-histograms-based adaptive multiple histograms generation and modification method are also employed in the extended 2D method for simplicity. Specifically, the optimal number of threshold are still set to three, meaning that still three thresholds as well as three different modification mechanisms with different expansion manners are utilized, while the modification mechanisms are based on 2D histograms in this case. As for the specific values of these thresholds, they are chosen from the set D that contains many kinds of values of $\{T_1, T_2, T_3\}$ with better embedding performance in the 1D method. According to the analysis of pairwise PEE, due to the EC of a cover image may be slightly decreased when it is extended to a 2D version, the values of thresholds whose EC is just over the required payload may not suitable in the 2D method. Therefore, we tend to conduct embedding process based on many different thresholds successively taken from D. And the EC values of the selected thresholds in the 1D scheme are all bigger than the required payload. Finally, the thresholds whose EC in the

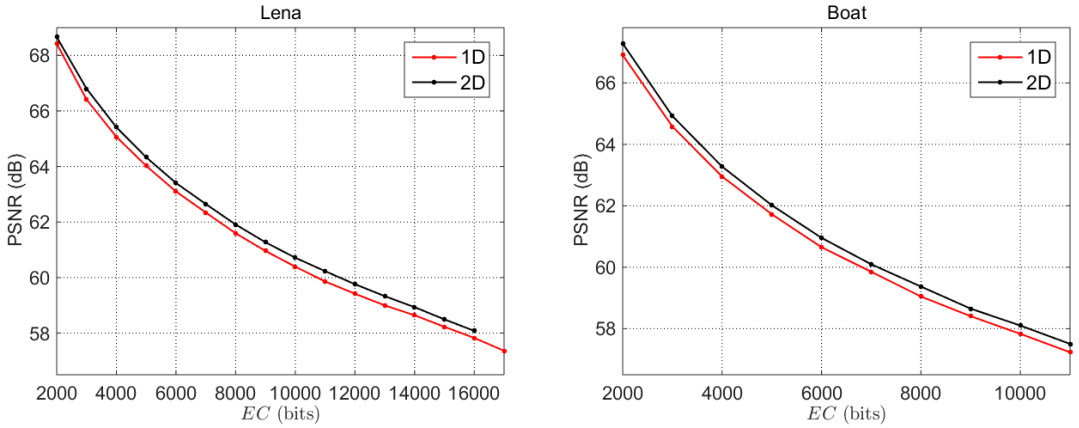


Figure 12: Embedding performance comparison between proposed 1D RDH scheme and the 2D version.

2D method is just over the required payload will be utilized to generate the marked image.

To illustrate the efficiency of extending the proposed 1D scheme to 2D space, we take the first layers of two standard 512×512 gray-scale images Lena and Boat for example. The performance comparison results between the 1D scheme and the 2D scheme are shown in Fig. 12. The red lines are the results based on the proposed 1D scheme, and the black lines correspond to the 2D version. Obviously, the PSNR values are higher in the 2D case under different values of EC in both images.

Double-layered embedding is also used in the proposed 2D RDH method. In the embedding phase, the second layer is processed only after the first layer embedding is completed. While in the extraction phase, the second layer is first extracted and recovered. Since the context pixels are all from another layer, the values of $\hat{p}_h$ and $\hat{p}_l$ in the extraction process are same as the ones in the embedding process. Besides, according to Eq. (32), in the embedding phase, if $x \geq \hat{p}_h$, i.e. $e \geq e_k$, x is either increased by 1 or unchanged. And if $x \leq \hat{p}_l$, i.e. $e \leq e_k$, x is either decreased by 1 or unchanged. That is to say, the relationship between $\widetilde{x}$ and $\hat{p}_h$ or $\hat{p}_l$ is also unchanged in the extraction phase.

Therefore, the extraction and recovery process can be conducted by following procedures. Based on the pixel $\widetilde{x}$ in the marked image, the prediction-error $\widetilde{e}$ is determined by

$$\widetilde{e} = \begin{cases} \widetilde{x} - \hat{p}_h, & \text{if } \widetilde{x} \geq \hat{p}_h \\ \widetilde{x} - \hat{p}_l, & \text{if } \widetilde{x} \leq \hat{p}_l \end{cases} . \tag{39}$$

Note that, if $\hat{p}_h = \hat{p}_l$, $\hat{p}_l$ should be updated by $\hat{p}_l = \hat{p}_l - 1$ before calculating the prediction-error. By calculating the NL of pixels, three different classes of pixels as well as their prediction-errors can be obtained. In order to extract data bits and recover the image based on the 2D mapping used in embedding phase, the prediction-errors should also be updated based on Eq. (39). Finally, the embedded payload can be extracted by the inverse mapping of Fig. 10, and the image pixels can be recovered accordingly.

3.4. Implementation details

Some implementation details of the proposed RDH scheme are given in this part. For simplicity, only the details of the extended 2D scheme based on pairwise modification are presented.

First of all, in order to achieve blind data extraction and image restoration, some side information should be embedded to the cover image. It mainly includes three parts:

- Location map: The overflow and underflow problem that may occur in the embedding process should be avoided by doing preprocessing on the cover image. In detail, since all the pixels may be modified at most 1 in the proposed method, pixels whose value is 0 or 255 may be changed to -1 or 256, and these two values are beyond the range [0,255] in normal gray-scale images. This phenomenon will cause the RDH to fail. To overcome this

problem, we perform the preprocess to turn 0 and 255 to 1 and 254. For losslessly image recovery, the positions of these pixels are recorded by a location map (LM). Pixels with modified value of 1 or 254 are recorded as 1 in the LM, and pixels whose original value is 1 or 254 are recorded as 0 in the LM. Other pixels are not recorded. After that, LM is losslessly compressed by arithmetic coding, and the compressed location map is denoted as CLM, which is considered as part of the payload. Meanwhile, 18 bits are used to record the length of CLM for its correct extraction.

- Optimal parameters: The optimal parameters needed to be recorded including three thresholds for the first layer and another three thresholds for the second layer. 10 bits are given to record each of them. Therefore, totally $10 \times 6 = 60$ bits are employed to record these parameters.

- The replaced LSBs: Since all these parameters should be known before the extraction process, they are embedded at the first line of the image by replacing the least significant bits (LSBs) of some pixels. Considering about the length of CLM and the optimal parameters, totally $t = 18 + 60 = 78$ bits are needed. Therefore, the original LSBs of first 78 pixels in the first row of the image should be recorded. In the proposed RDH scheme, these original LSBs together with CLM are linked at the beginning of the payload for further embedding.

Now, the embedding process as well as the extraction and recovery process can be performed by following steps. Firstly, the steps of the embedding process is given. For the sake of simplicity, only the embedding and extraction procedures of the first layer is presented.

Step 1. Generate the location map except the four edge lines of the image. Losslessly compress it and obtain CLM along with its length S_{CLM}.

Step 2. Extract the LSBs of the first t pixels at the first line, and temporarily set their LSBs to 0.

Step 3. Link the original LSBs of the t pixels as well as the CLM at the beginning of the payload.

Step 4. Parameters Selection and Data Embedding.

- Perform pixel prediction and calculate NL for each of them.

- Find the thresholds set D based on the optimal parameters selection method in the 1D scheme. This set contains many values of $\{T_1, T_2, T_3\}$ with good performance for different EC.

- Successively taken groups of $\{T_1, T_2, T_3\}$ from D that satisfying corresponding EC in the 1D scheme is bigger than the payload. Conduct 2D data embedding based on current values of $\{T_1, T_2, T_3\}$ to find if it is sufficient to embed the payload.

- Finally, the proper values of $\{T_1, T_2, T_3\}$ can be found, and the payload are embedded into the cover image based on the 2D method.

Step 5. Replace the LSBs of the pixels in the first line to embed the length of CLM as well as the optimal thresholds. After this step, the embedding process of the first layer is completed.

The steps of the extraction and recovery process are as follows.

Step 1. By reading the LSBs of the first t pixels at first line, obtain all the parameters required in the extraction process. And temporarily set their LSBs to 0.

Step 2. Perform pixel prediction and NL calculation as the same way with the embedding process.

Step 3. Based on the parameters obtained from step 1, extract the payload from the image and recover these pixels based on the corresponding reverse mappings of the pairwise modification mechanism.

Step 4. Take first t bits of the extracted payload and recover the first t pixels in the first line of the image. Take out the following S_{CLM} bits and uncompress it to obtain the original location map and further recover the modified pixels in the locations with 1. Finally, the payload is extracted and the image is losslessly recovered.

4. Experimental results

In this section, several experiments are conducted to evaluate the performance of the proposed method by comparing it with some state-of-the-art methods. The methods to compare includes IPVO, PPVO and the newly published

Table 3: Optimal thresholds in the first layer and the second layer along with pl_1 for EC of 10,000 bits .

Images	First layer			Second layer			pl_1
	T_1	T_2	T_3	T_1	T_2	T_3	
Lena	20	40	54	18	38	52	5530
Baboon	268	282	912	40	450	478	5687
Airplane	14	18	28	12	24	26	6567
Barbara	16	54	76	0	40	74	5565
Boat	44	50	92	38	54	86	5623
Elaine	70	72	88	52	70	80	5990

skewed-histogram-shifting-based Kim *et al.*'s work, along with the pairwise PEE. All these experiments are implemented on Matlab.

First of all, six standard gray-scale images with size of 512×512 are utilized for performance comparison between these methods, including Lena, Baboon, Airplane, Babara, Boat and Elaine. Note that, in the proposed method, the value of n is set to 3, and the values of e_k are set as Eq. (42). The first experiment aims to analyze the optimal parameters used in the embedding phase. Six thresholds for each image including three used in the first layer and three used in the second layer along with the value of pl_1 for EC of 10,000 bis are shown in Table. 2. From this table, one can observe that the best thresholds are various on different images, indicating that the adaptive choosing of the thresholds used in the proposed method helps achieve better performance on different kinds of images. Also, there is no certain relationship among the thresholds in each image, and hence it is necessary to test all possible sets of the thresholds for good performance. Besides, the optimal values of pl_1 which represents the number of the bits embedded in the first layer are all bigger that half of the payload, especially the pl_1 of Airplane. Thus, it is reasonable to find a suitable pl_1 in the embedding process rather than simply embed half of the payload in each layer.

Table 4: Comparison of PSNR (in dB) between the proposed method and some advanced methods with the embedding capacity of 10,000 bits.

Images	Ou *et al.*	Peng *et al.*	Qu *et al.*	Kim *et al.*	Proposed
Lena	59.70	60.47	60.36	59.92	61.23
Baboon	55.22	53.55	54.11	55.99	55.23
Airplane	63.66	62.96	63.76	63.66	64.31
Barbara	59.43	60.55	60.09	59.95	61.60
Boat	57.49	58.27	58.38	57.67	58.65
Elaine	58.06	57.36	58.30	58.33	58.42
Average	58.93	58.86	59.17	59.25	59.91

Table 5: Comparison of PSNR (in dB) between the proposed method and some advanced methods with the embedding capacity of 20,000 bits.

Images	Ou *et al.*	Peng *et al.*	Qu *et al.*	Kim *et al.*	Proposed
Lena	56.22	56.54	56.66	56.67	57.60
Airplane	60.15	59.07	60.02	60.12	60.67
Barbara	56.23	56.20	56.29	56.52	57.43
Boat	53.32	53.84	54.19	53.73	54.76
Elaine	52.91	52.61	53.55	53.11	53.51
Average	55.77	55.65	56.14	56.03	56.79

Furthermore, experiments are conducted to compare the performance between the proposed method and some state-of-the-art methods. The comparison results of PSNR under different embedding capacities are shown in Fig. 11.

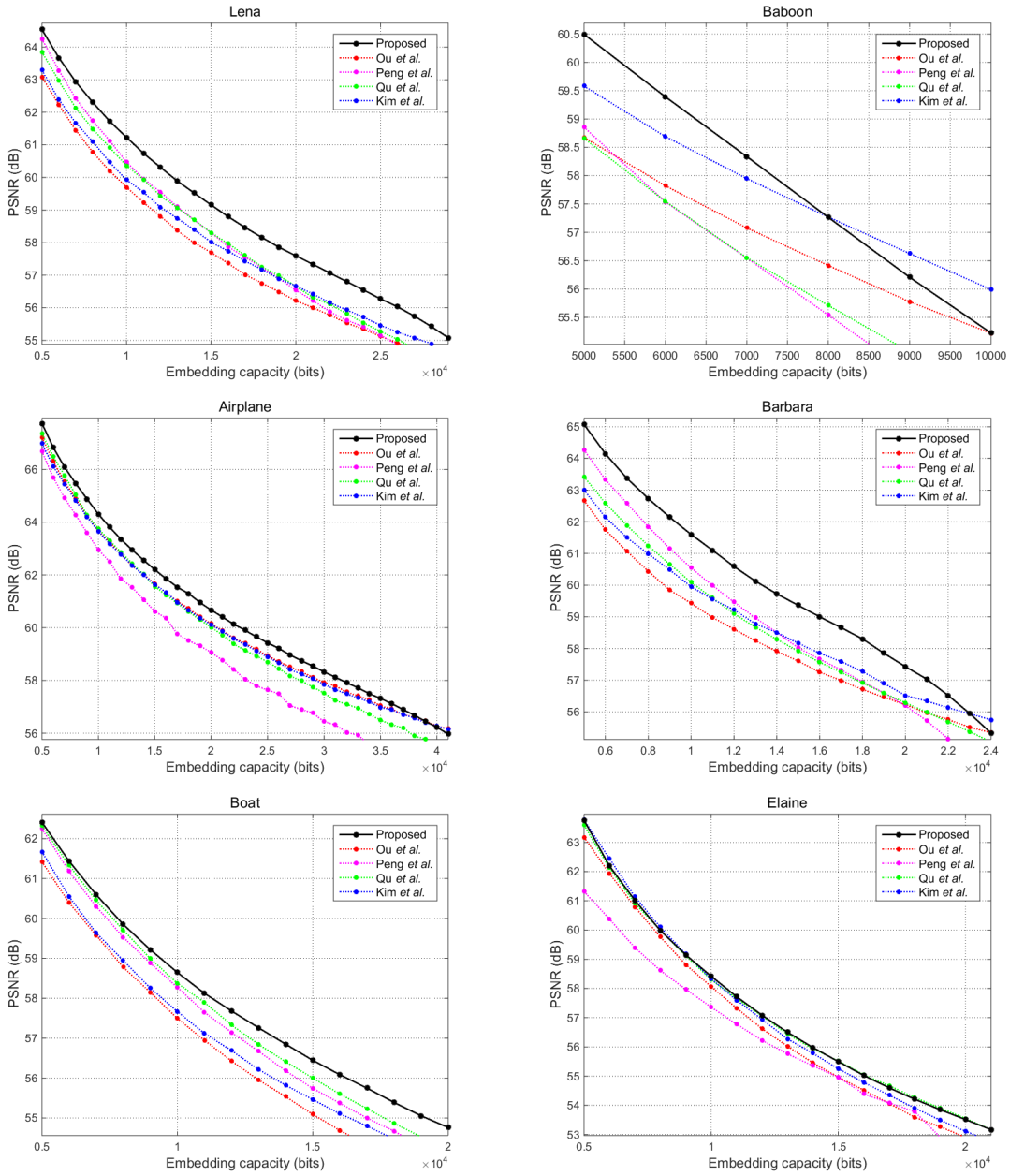


Figure 13: Performance comparison between the proposed method and some state-of-the-art methods.

Notice that, in the proposed method, the embedding capacity is varies from 5,000 bits to its maximum with a step of 1,000 bits. From the results of these images, it is obvious that the proposed method outperforms others in most images and under various embedding capacities. Besides, more specific comparison results of the these methods under the embedding capacity of 10,000 bits and 20,000 bits are presented in Table 3 and Table 4, respectively. As one can see from the tables, the average PSNR of the proposed method is the highest one both in the cases with the embedding capacity of 10,000 and 20,000 bits.

Note that,the implementation details of all these compared methods are consisted with the conditions used in their original article. As for IPVO, the block size used in the implementation of the IPVO method is chosen from 2×2 to

5×5, totally 16 times embedding are performed and the block size leading to highest PSNR is used. As mentioned in Section 2.1, compared to the original PVO-based method, IPVO improves the embedding performance by taking the positions into consideration and further utilizing more pixels in smooth area to embed. Although the embedding performance is improved compared to the original PVO, it still has much room for improvement both in the procedures of error prediction and PEH modification. Compared with IPVO, our method utilizes a more accurate predictors as well as an adaptive multiple PEHs generation and modification strategy and obtains better performances on all these images as shown in Fig. 11.

For PPVO, 15 times of embedding are performed under each embedding capacity in every image to find a proper number of context pixels from 1 to 15, and the results with highest PSNR are shown in Fig. 11. This method changes the block-by-block manner used in PVO and IPVO to pixel-by-pixel manner to predict each pixel and improves the performance by using more pixels in smooth area than IPVO. However, its performance can be further improved by a more accurate full-enclosing predictor as well as a variable bin used for extension in multiple PEHs. Specifically, the proposed methods outperforms PPVO in almost all the images except for the slight weakness in Elaine when EC is bigger than 17,000, and it improves PPVO by 0.74 dB for an embedding capacity of 10,000 bits and 0.65 dB for 20,000 bits in average according to Table 3 and Table 4.

In a newly published work, Kim *et al.* [20] proposes 3 pairs of extreme predictors based on PVO to improve the prediction. Each pair of predictors can generate two skewed PEHs and it improves the performance mainly based on reducing the shifting distortion. However, since a fixed bin is always chosen to be expanded in its embedding process, it can be improved by adaptively applying different modification mechanisms on PEHs generated by pixels of different NLs. Based on the comparison results shown in Fig. 11, the proposed methods gains higher values of PSNR in most cases. While this method achieves better performance in Baboon when EC is bigger than 8,000 bits, and this is mainly due to the predictor 2 which does not only use the maximum and minimum for prediction. Overall, the proposed method shows better performance that Kim's, and it improves this method by 0.66 dB and 0.76 dB in average for the embedding capacity of 10,000 bits and 20,000 bits, respectively. What's more, the improvement on Barbara even reaches 1.65 dB for the embedding capacity of 10,000 bits, which is a considerable improvement.

Furthermore, pairwise PEE is also compared. As a 2D-PEH modification based method, pairwise PEE achieves better performance than many 1D-PEH based methods [yinyong]. Since the embedding in this method is based on a single 2D-PEH and a fixed 2D-mapping, the proposed method can achieve better performance based on the modification of multiple PEHs. Experimental results verify that the proposed method is of higher PSNRs. And, it has 0.98 dB and 1.02 dB increase over pairwise PEE in average as shown in Table 1 and Table 2.

Next experiments is conducted on the Kodak database. 24 color images whose size is 512×768 or 768×512 are included. Note that, all of them are converted to gray-scale images before embedding. Since the performance of Qu *et al.*'s method and Kim *et al.*'s method is more closer to the proposed method, only these two methods are compared to in this experiment. Table 5 shows the comparison results of 24 images with the embedding capacity of 10,000 bits. It is clear that the proposed method achieves the best results, whose average PSNR is higher than Qu *et al.*'s method and Kim *et al.*'s method by 0.56 dB and 0.49 dB, respectively. Furthermore, the proposed method has the highest PSNR values in 21 images in these 24 images.

To further examine the effectiveness of proposed method, one more experiment is conducted on a large database BossBase v1.01, which contains 10,000 gray-scale images. All these images are 512×512 sized coming from scaled and cropped natural images with various sizes. Each image is embedded with 10,000 bits based on Qu *et al.*'s method, Kim *et al.*'s method and the proposed method, respectively. The obtained results is shown in Fig. 14 through the distribution of the PSNR increase. Note that, the value of PSNR increase of each image is calculated by subtracting the PSNR of the compared method using the PSNR of the proposed method. (a) shows the comparison between the proposed method and Qu *et al.*'s method, and the proposed method outperforms Qu *et al.*'s with an increase of 0.76 dB in average. (b) shows the comparison between the proposed method and Kim *et al.*'s method, and the proposed method has better embedding performance than Kim *et al.*'s method by 0.68 dB in average. In a word, the superiority of the proposed method is also demonstrated on large image database.

5. Conclusion

In this paper, a novel RDH method based on PVO and a multiple histograms generation and modification strategy is proposed. Instead of doing modification on a single PEH which has been used in many former methods, the

Table 6: Comparison of PSNR (in dB) between the proposed method and the methods of Qu *et al.*[17] and Kim *et al.*[20] with the embedding capacity of 10,000 bits in Kodak database.

Images	Qu *et al.*	Kim *et al.*	Proposed
kodim01	64.11	64.85	65.02
kodim02	64.36	64.63	65.20
kodim03	65.04	65.03	66.05
kodim04	63.99	64.53	65.00
kodim05	62.47	64.69	64.95
kodim06	66.07	66.48	67.03
kodim07	64.79	64.47	65.63
kodim08	57.35	58.17	58.19
kodim09	63.31	62.67	63.67
kodim10	62.60	62.34	63.14
kodim11	65.40	64.88	65.45
kodim12	64.67	64.71	65.39
kodim13	57.78	59.32	58.19
kodim14	62.51	64.13	64.24
kodim15	62.85	63.34	63.45
kodim16	65.02	64.99	65.71
kodim17	64.36	63.68	64.61
kodim18	60.98	60.90	61.58
kodim19	63.24	62.96	64.03
kodim20	58.04	56.43	56.22
kodim21	63.54	62.58	63.87
kodim22	62.99	62.59	63.44
kodim23	64.39	64.15	64.72
kodim24	62.04	61.12	60.56
Average	63.00	63.07	63.56

proposed method classified pixels with different degrees of noise level into several categories and further generate several different histograms. Reasonable modification are performed based on their own characteristics to get better performance. Experimental results demonstrate that, compared to many state-of-the-art RDH methods, the proposed method is of more superiority in the capacity-distortion performance. Since the proposed 2D version is simply conducted based on a fixed mapping, the future work is to develop more suitable mappings for the modification of multiple PEHs.

References

[1] M. U. Celik, G. Sharma, A. M. Tekalp, E. Saber, Lossless generalized-LSB data embedding, IEEE Trans. Image Process. 14 (2) (2005) 253-C266.

[2] J. Fridrich, M. Goljan, R. Du, Lossless data embedding-new paradigm in digital watermarking, EURASIP J. Appl. Signal Process. 2002 (2) (2002) 185–196.

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

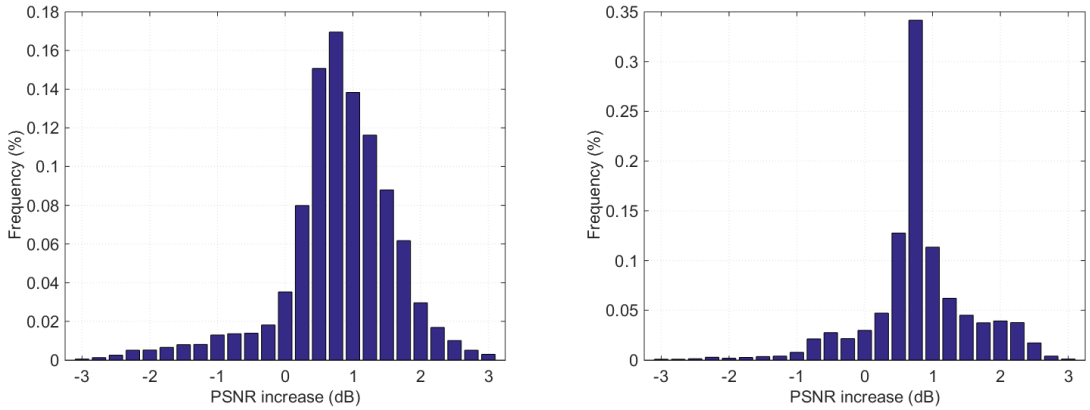
[4] Y. Hu, H. K. Lee, J. Li, De-based reversible data hiding with improved overflow location map, IEEE Trans. Circuits Syst. Video Technol. 19 (2) (2009) 250–260.

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

[6] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 16 (3) (2006) 354–362.

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

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



(a) Proposed method vs Qu *et al.*'s method.

(b) Proposed method vs Kim *et al.*'s method.

Figure 14: PSNR comparison between the proposed method and the methods of Qu *et al.* and Kim *et al.* with EC of 10,000 bits in BossBase v1.01 database.

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

[10] D. M. Thodi, J. J. Rodrguez, Expansion embedding techniques for reversible watermarking, IEEE Trans. Image Process. 16 (3) (2007) 721–730.

[11] 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.

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

[13] 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.

[14] F. Peng, X. Li, B. Yang, Improved pvo-based reversible data hiding, Digit. Signal Process. 25 (2) (2014) 255–265.

[15] B. Ou, X. Li, Y. Zhao, R. Ni, Reversible data hiding using incariant pixel-value-ordering and prediction-error expansion, Signal Process. 29 (7) (2014) 760–772.

[16] X. Wang, J. Ding, Q. Pei, A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition, Inf. Sci. 310 (2015) 16–35.

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

[18] 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.

[19] 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.

[20] S. Kim, X. Qu, V. Sachnev,H.J. Kim, Skewed Histogram Shifting for Reversible Data Hiding using a Pair of Extreme Predictions, IEEE Trans. Circuits Syst. Video Technol. Early Access.

[21] X. Li, W. Zhang, X. Gui, B. Yang, Efficient Reversible Data Hiding Based on Multiple Histograms Modification, IEEE Trans. Inf. Forens. Secur. 10(9) (2015) 2016–2027.

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

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

[24] D. Coltuc, Low distortion transform for reversible watermarking, IEEE Trans. Image Process. 21 (1) (2011) 412–417.

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