Reversible Watermarking Using Adaptive Edge-Guided Interpolation

Ningjie Dai, Guorui Feng and Qian Zeng
School of Communication and Information Engineering, Shanghai University, Shanghai 200072
[e-mail: {chinadai, grfeng, zxqian}@shu.edu.cn]
*Corresponding author: Guorui Feng

Received January 21, 2011; revised February 25, 2011; accepted March 23, 2011; published April 29, 2011

Abstract

Reversible watermarking is an open problem in information hiding field, with embedding the encoded bit '1' or '0' into some sensitive images, such as the law enforcement, medical records and military images. The technique can retrieve the original image without distortion, after the embedded message has been extracted. Histogram-based scheme is a remarkable breakthrough in reversible watermarking schemes, in terms of high embedding capacity and low distortion. This scheme is lack of capacity control due to the requirement for embedding large-scale data, because the largest hidden capacity is decided by the amount of pixels with the peak point. In this paper, we propose a reversible watermarking scheme to enlarge the number of pixels with the peak point as large as possible. This algorithm is based on an adaptive edge-guided interpolation, furthermore, hides messages by interpolation-error, i.e. the difference between the original and interpolated image value. Simulation results compared with other state-of-the-art reversible watermarking schemes in this paper demonstrate the validity of the proposed algorithm.

Keywords: Reversible watermarking, adaptive interpolation-error expansion, edge-guided interpolation, histogram shift

DOI: 10.3837/tiis.2011.04.013

1. Introduction

Image reversible watermarking techniques hide unperceivable data in a host signal and recover the original image after extracting the embedded information. Achieving distortion-free data embedding in some sensitive applications, such as law enforcement, medical records and military images, belongs to the specified purpose of the reversible watermarking. On these occasions, it is prohibited to change any pixel of the image. Here, we both rely on the hidden data and original image, for any change will affect the authentication of the image and the access to the original data. And even, the lightest change in the pixel value is intolerable. In the reversible watermarking field, how to embed information, extract the hidden data and recover the original image is the task of the reversible watermarking algorithms. The goal of the reversible watermarking researchers is to embed more hidden message in original multimedia with lower distortion.

The earliest idea and application of reversible watermarking, to the authors' knowledge, could be traced to Barton [1] in a US patent. Barton proposed a fragile reversible watermarking scheme based on the spatial domain. A module-256 addition function was utilized, but problems of salt-and-pepper noise hindering watermark retrieval could not be avoided. Later, an improved approach was proposed by De Vleeschouwer et al. [2] to reduce visual artifacts.

From that on, dozens of reversible watermarking schemes had been proposed during the past decade, and achieved the performance of great payload capacity and low visual distortion. These reversible watermarking methods could be simply classified into three categories: methods based on lossless compression of image components, difference expansion, and histogram shifting, respectively.

The first class of these methods is conceptually rather simple. In 2001, Miroslav Goljan et al.[3] presented a steganalytic method for detection of LSB (least significant bit) embedding. But the amount of hidden data was limited, and the distortion still existed. At the same year, Fridrich et al. [4] proposed two techniques for authentication of digital images using invertible watermarking. The first method was based on robust spatial additive watermarks connected with modulo addition and the other resorted to lossless compression and encryption of bit-planes. Celik et al. [5] presented a LSB data-embedding reversible watermarking, which introduced additional operating pointed on the capacity-distortion curve.

The second type of these methods (DE, difference expansion) was first proposed by Tian [6]. His method divided the image into pairs of pixels, then embedded one bit of information into each pair. So it could not exceed the embedding rate of 0.5 bits per pixel (bpp) in a single layer embedding. Then, Alttar [7] expanded Tian's method, which divided the image into quads (a set of four pixels) and embedded three bits in each quad. In 2005, Kamstra et al. [8] improved Tian's method by sorting to increase the efficiency of lossless compression.

Histogram operation, known as an effective strategy for reversible watermarking methods, was first presented by Ni et al. [9] in 2006. This method hid the message by tuning the zero point and peak point of an image histogram. Their idea was very simple, embedding bit '1' or '0' into the pixels with the peak point by expanding it additively or leaving it unchanged, while expanding the pixels with the value between the peak and zero point by one bit. Therefore, this method got slight distortion with low complexity, but its capacity relied on the number of pixels with the peak point was limited. For example, the peak point of *Lena* is 154 and the amount of pixels with the peak point is 2716, which means the largest hidden

capacity of this image is 2716. So if we want to hide large embedded message in image, the concept of this histogram scheme must be extended.

Later, Hwang et al. [10] extended Ni's scheme and employed the peak point of the image histogram and location map in their algorithm. One of the improvements was that the side information didn't need to transmit to the receiver. In 2007, Thodi et al. [11] combined the histogram-shift with prediction-error expansion to solve the location map problem which generated by DE [6], achieving a significant improvement in the quality of the watermarking image compared with DE algorithm. Lin and Hsueh [12] applied block differences by dividing the image into thousands of three-pixel blocks which contain two absolute differences -- the difference between pixel one and two, and pixel two and three, then, embedded hidden data in the peak point of pixel difference. In 2009, Hu et al. [13] applied the DE mixed methods to prediction-errors including expandable, shiftable, unexpandable and unshiftable (overflow /underflow) predicted errors to make the size of the side information package for blind detection as small as possible. In the same year, Tsai et al. [14] proposed a new scheme that achieved high capacity and low distortion, where a residue image indicated a difference between the base-pixel and other pixels in a non-overlapping block. Recently, Luo et al. [15] proposed a reversible watermarking scheme based on additive interpolation-error expansion, achieving the performance of low distortion and high capacity.

From the recent conventional schemes, to our knowledge, the methods of combining interpolation with histogram shifting are the key breakthrough for good performance of high capacity and low distortion. Most of these schemes embedded hidden information in the difference between the original value and its interpolated version. So the better interpolation method is obtained, the higher performance of reversible watermarking can be achieved. As is well known to all, interpolation methods are most commonly used in the demosaicing to recover color images. Next we introduce demosaicing methods (image reconstruction from color samples) briefly.

In color samples, there are many existing kinds of color filter array (CFA) patterns, and the most common pattern is the Bayer [16] pattern as presented in **Fig. 1-(a)** (the interpolation pattern of grey scale image in **Fig. 2-(b)-(e)**, is similar to the B channel of the color image in **Fig. 1-(b)**). A lot of interpolation methods had been presented since that. In the early time, the usual applications were non-adaptive methods, such as pixel duplication, bilinear interpolation, and bi-cubic convolution interpolation. These methods had advantages in simplicity and low cost, however, there were many inherent defects, such as block effects, blurred detail and ringing artifacts around edges. What's worse, the peak signal-to-noise ratio (PSNR) value of benchmark images is unsatisfactory. Recently, many methods were proposed based on the high correlation between different color channels or adjacent pixels. In 2006, Zhang et al. [17] proposed a demosaicing scheme to estimate the different signals by linear minimum mean square-error estimation (LMMSE) technique. The contribution of section 2-2 is derived from this theory.

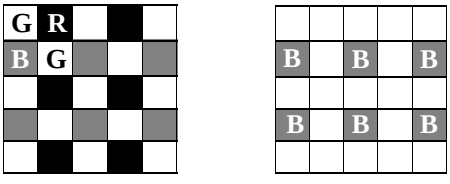

Fig. 1. **(a)** Bayer CFA **(b)** B channel in Bayer CFA.

The structure of this paper is organized as follows. In Section 2, we discuss the related theories of demosaicing and interpolation. Then in Section 3 we will describe the proposed method. Simulation results and data analysis are demonstrated in Section 4. In Section 5, conclusions of this paper are presented.

2. Adaptive Gray Image Edge-guided Interpolation

In this section, firstly, the equation of gray image is explicitly denoted in terms of trigonometric function. Secondly, the derivation of non-border pixels interpolation and the calculation of the border pixels interpolation will simultaneously be given. Thirdly, the order of embedding process is described. Finally, we describe the additive interpolation-error expansion.

2.1 Formulation of Gray Image

From [18], we can divide a complete image into four subsample images. In each subsample images, we use three-quarter's pixels (gray pixels) to interpolate quarter's pixels (white pixels), shown in **Fig. 2**. The expression of the four quarter-images can be scribed in terms of cosine functions as follows:

$$\begin{aligned} Z_1(i,j) &= O(i,j) - \frac{1}{4}(1-\cos\pi i)(1-\cos\pi j)O(i,j) \\ Z_2(i,j) &= O(i,j) - \frac{1}{4}(1-\cos\pi i)(1+\cos\pi j)O(i,j) \\ Z_3(i,j) &= O(i,j) - \frac{1}{4}(1+\cos\pi i)(1-\cos\pi j)O(i,j) \\ Z_4(i,j) &= O(i,j) - \frac{1}{4}(1+\cos\pi i)(1+\cos\pi j)O(i,j) \end{aligned}, \tag{1}$$

where (i, j) indicates the pixel coordinate, starting with (1,1), $O(i, j)$ denotes the original pixel value and $Z_k(i, j)$ denotes the subsample images' value, k=1,2,3,4.

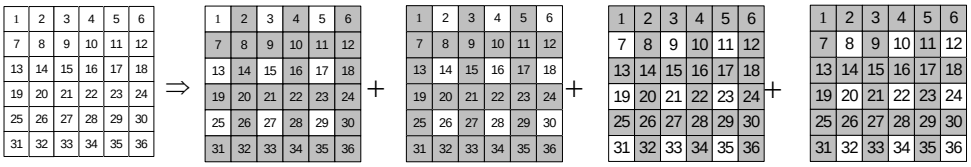

Fig. 2. System overview. **(a)** A complete image, **(b)-(e)** Interpolation of white pixels by three-quarter gray pixels.

2.2 Derivation of Four Directional Adaptive Edge-guided Interpolations for Non-border pixels

Since missing samples have high correlation with their neighbours in the edge, they can be interpolated by neighbor pixels. Luo *et al.* [15] succeed in using Zhang *et al.* [17] interpolation method into reversible watermarking based on two-direction estimation. Now, we derive four- direction estimation based on Zhang's method, and consider **Fig. 2-(e)** as example shown in **Fig. 3**. We can recover the missing pixel $x_w(2i,2j)$ or $x_w(n)$ along four directions: horizontal direction, vertical direction, 45^0 diagonal direction and 135^0 diagonal

direction. And four directional interpolation results which are denoted by $\hat{x}_h(2i,2j)$, $\hat{x}_v(2i,2j)$, $\hat{x}_{45}(2i,2j)$ and $\hat{x}_{135}(2i,2j)$, can be computed as

$$\begin{cases} \hat{x}_h(2i,2j) &= (x_l(2i\quad,2j-1)+x_l(2i\quad,2j+1))/2 \\ \hat{x}_v(2i,2j) &= (x_l(2i-1,2j\quad)+x_l(2i+1,2j\quad))/2 \\ \hat{x}_{45}(2i,2j) &= (x_l(2i-1,2j+1)+x_l(2i+1,2j-1))/2 \\ \hat{x}_{135}(2i,2j) &= (x_l(2i-1,2j-1)+x_l(2i+1,2j+1))/2 \end{cases}. \tag{2}$$

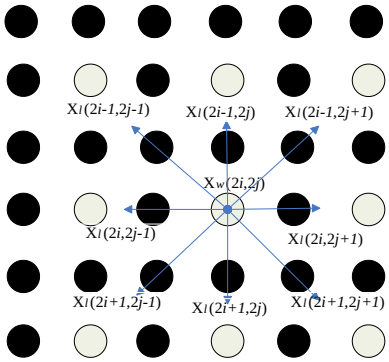

Fig. 3. Formation of a low-resolution image.

Let $\tilde{x}_h(2i,2j)$, $\tilde{x}_v(2i,2j)$, $\tilde{x}_{45}(2i,2j)$ and $\tilde{x}_{135}(2i,2j)$ be the corresponding estimation errors, then

$$\begin{cases} \tilde{x}_h(2i,2j) &= x(2i,2j)-\hat{x}_h\quad(2i,2j) \\ \tilde{x}_v(2i,2j) &= x(2i,2j)-\hat{x}_v\quad(2i,2j) \\ \tilde{x}_{45}(2i,2j) &= x(2i,2j)-\hat{x}_{45}\quad(2i,2j) \\ \tilde{x}_{135}(2i,2j) &= x(2i,2j)-\hat{x}_{135}(2i,2j) \end{cases}. \tag{3}$$

Four variables in (4) exploit the correction of $x(2i,2j)$ with its neighbors in a particular direction. The more optimal estimate of $x(2i,2j)$ can be calculated by fusing the four directional LMMSE method. It can be expressed as

$$\hat{x}_w(n) = w_h(n).\hat{x}_h(n)+w_v(n).\hat{x}_v(n)+w_{45}(n).\hat{x}_{45}(n)+w_{135}(n).\hat{x}_{135}(n). \tag{4}$$

where $w_h(n)+w_v(n)+w_{45}(n)+w_{135}(n)=1$. From ref.[17], in two orthogonal directional $w_h(n)/w_v(n)=\sigma^2_{\tilde{x}_v}(n)/\sigma^2_{\tilde{x}_h}(n)$, where $\sigma^2_{\tilde{x}_h}(n)$ and $\sigma^2_{\tilde{x}_v}(n)$ are the variances of estimation errors $\hat{x}_h(n)$ and $\hat{x}_v(n)$. Now, in four directions, we assume

$$\begin{cases} w_h(n)/w_v(n)=\sigma^2_{\tilde{x}_v}(n)/\sigma^2_{\tilde{x}_h}(n) \\ w_{45}(n)/w_{135}(n)=\sigma^2_{\tilde{x}_{135}}(n)/\sigma^2_{\tilde{x}_{45}}(n) \end{cases}. \tag{5}$$

The weight $w_h(n)$, $w_v(n)$, $w_{45}(n)$ and $w_{135}(n)$ are determined to minimize the mean square error (MSE) of $\hat{x}_w(n)$

$$\sigma^2_{\tilde{x}_w}(n) = E[\tilde{x}^2_w(n)] = E[(x(n) - \hat{x}_w(n))^2]. \quad (6)$$

Substituting (3), (4), (5) into (6), we obtains

$$\begin{aligned}\sigma^2_{\tilde{x}_w}(n) &= w_h^2(n).\sigma^2_{\tilde{x}_h}(n) + w_v^2(n).\sigma^2_{\tilde{x}_v}(n) + w_{45}^2(n).\sigma^2_{\tilde{x}_{45}}(n) + w_{135}^2(n).\sigma^2_{\tilde{x}_{135}}(n) \\ &+ 2w_h(n).w_v(n).E[\tilde{x}_h(n).\tilde{x}_v(n)] + 2w_h(n).w_{45}(n).E[\tilde{x}_h(n).\tilde{x}_{45}(n)] \\ &+ 2w_h(n).w_{135}(n).E[\tilde{x}_h(n).\tilde{x}_{135}(n)] + 2w_v(n).w_{45}(n).E[\tilde{x}_v(n).\tilde{x}_{45}(n)] \\ &+ 2w_v(n).w_{135}(n).E[\tilde{x}_v(n).\tilde{x}_{135}(n)] + 2w_{45}(n).w_{135}(n).E[\tilde{x}_{45}(n).\tilde{x}_{135}(n)] \\ &\approx w_h^2(n).\sigma^2_{\tilde{x}_h}(n) + w_v^2(n).\sigma^2_{\tilde{x}_v}(n) + w_{45}^2(n).\sigma^2_{\tilde{x}_{45}}(n) + w_{135}^2(n).\sigma^2_{\tilde{x}_{135}}(n)\end{aligned}. \quad (7)$$

In Eq. (7), the correlation between the variables $\tilde{x}_i$ and $\tilde{x}_j$ is weak in a natural image, so $E[\tilde{x}_i(n).\tilde{x}_j(n)] \approx 0$, where i, j = h, v, 45, 135 and i ≠ j. To minimize $\sigma^2_{\tilde{x}_w}(n)$ (that means $\sigma^2_{\tilde{x}_w}(n) = 0$), combining with $w_h(n) + w_v(n) + w_{45}(n) + w_{135}(n) = 1$ and Eq.(5), we can obtain

$$w_h(n) = \frac{\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n)}{\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) + \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) + \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{135}}(n) + \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)}. \quad (8)$$

We take *u* as the average of neighbor pixels,

$$u = \frac{x_l(2i-1,2j-1) + x_l(2i-1,2j) + x_l(2i-1,2j+1) + x_l(2i,2j-1) + x_l(2i,2j+1) + x_l(2i+1,2j-1) + x_l(2i+1,2j) + x_l(2i+1,2j+1)}{8}. \quad (9)$$

and

$$\sigma^2_{\tilde{x}_h}(2i,2j) = \frac{1}{3}\sum_{k=1}^{3}(S_h(k) - u)^2, \quad (10)$$

where $S_h = \{x_l(2i,2j-1), \hat{x}_h(2i,2j), x_l(2i,2j+1)\}$.

Other three functions of $\sigma^2_{\tilde{x}_v}(n)$, $\sigma^2_{\tilde{x}_{45}}(n)$, $\sigma^2_{\tilde{x}_{135}}(n)$ are similar to $\sigma^2_{\tilde{x}_h}(n)$ and $w_v(n)$, $w_{45}(n)$, $w_{135}(n)$ are similar to $w_h(n)$. In fact, the Euclidean distance of four neighbor pixels in 45^0 and 135^0 diagonal is $\sqrt{2}$ times as those in horizontal and vertical direction. In a natural image, the farther a pixel is from the missing pixel, the lower correction between the pixel and the missing pixel. Thus, we revise the weight of 45^0 and 135^0 diagonal to $\sqrt{2}/2$. Eq.(8) can be turned into

$$w_h(n) = \frac{\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n)}{\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) + \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) + \frac{\sqrt{2}}{2}\sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{135}}(n) + \frac{\sqrt{2}}{2}\sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)}. \quad (11)$$

And do the similar revision in $w_v(n), w_{45}(n)$ and $w_{135}(n)$.

If $\sigma^2_{\tilde{x}_{45}}(n) >> \sigma^2_{\tilde{x}_h}(n), \sigma^2_{\tilde{x}_{45}}(n) >> \sigma^2_{\tilde{x}_v}(n)$ and $\sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_h}(n), \sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_v}(n)$, then

$$\begin{aligned}
&\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{135}}(n)\\
&\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)\\
&\sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{135}}(n)\\
&\sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_{45}}(n)\sigma^2_{\tilde{x}_{135}}(n) >> \sigma^2_{\tilde{x}_h}(n)\sigma^2_{\tilde{x}_v}(n)\sigma^2_{\tilde{x}_{45}}(n)
\end{aligned} . \tag{12}$$

The formulation (11) can be revised as

$$w_h(n) = \frac{\sigma^2_{\tilde{x}_v}(n)}{\sigma^2_{\tilde{x}_v}(n) + \sigma^2_{\tilde{x}_h}(n)} . \tag{13}$$

And the $w_v(n)$, $w_{45}(n)$ and $w_{135}(n)$ should be done the same revision. For above, four directional edge-guided interpolations is the extension of the 2 directional interpolations.

In conclusion, we use Eq. (4) for predicting the missing pixel, Eq.(11) for the weight and Eq.(10) for local variances computations in section 2-2.

2.3 Bilinear Interpolation to Border Pixels

On the border of a natural image, there are only 3 or 5 neighbors instead of 8, such as pixels $x_l(1,1)$ and $x_l(5,3)$, shown in **Fig. 4**. In this case, we can't use Section 2-2 to interpolate these pixels. Now we apply bilinear interpolation to interpolate the missing pixels' value. First of all, we sort the border pixels into two types: pixels in the corner such as $x_l(1,1)$ and in the sideline such as $x_l(5,3)$. The prediction of the corner pixel and the sideline pixel can be calculated as follows:

$$\begin{aligned}
x_l(1,1) &= \frac{1}{2}(x_l(1,2) + x_l(2,1)),\\
x_l(5,3) &= \frac{1}{3}(x_l(5,2) + x_l(4,3) + x_l(5,4)).
\end{aligned} \tag{14}$$

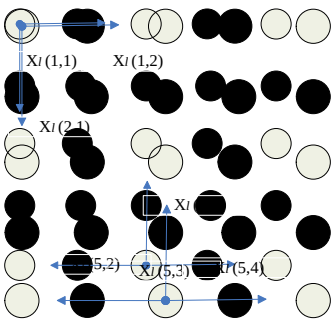

Fig. 4. Bilinear interpolation for border pixels.

2.4 The Order of Embedding Process

We recover all pixels by interpolating one quarter pixels four times. So there exist many orders of embedding process, and a better order of embedding process can achieve larger capacity of hidden data and higher PSNR. Before the discussion of the order of the

embedding process, we assume that in each quarter interpolation, the number of pixels with the peak point of interpolation-error histogram (the histogram of the difference between the original value and the interpolated value) is the same, and the number of pixels with the second peak point is close to that with the peak point, all marked as ***S***. And mark the number of pixels in the image as ***T***. Next, we discuss three main methods of embedding process order. The others are similar to order 2) or 3).

1) Embed hidden data in the peak point and the second peak point, and expand the interpolation-error in two direction (add or subtract 1 bit) at the same time. In this case, the capacity of each interpolation is 2***S*** and the number of changed pixels is ***T*/4**. After quarter interpolation four times, the total capacity of this order is 8***S*** and the number of changed pixels is ***T***.

2) First, embed hidden data in the peak point and expand the interpolation-error in the right direction (add 1 bit) for the first quarter interpolation. Then interpolate the first quarter again, embed hidden data in the peak point and expand in the left direction (subtract 1 bit). In the first embedding process, the payload is ***S*** and the number of changed pixels is ***T***/8; while in the second embedding process, ***T***/16 pixels from expanding in the right direction recover to original value and ***T***/16 original pixels subtract 1 bit. After above two embedding process, the capacity is 2***S*** and the number of changed pixels is ***T***/16. Do the same operation in the second to forth quarter sub-image embedding process. So the total capacity of this order is 8***S*** and the number of changed pixels is ***T***/2.

3) First, embed hidden data in the peak point and expand the interpolation-error in the right direction (add 1 bit) in the first quarter interpolation, and do the same operation in the second to forth quarter interpolation. Then expand the interpolation-error in the left direction (subtract 1 bit) in the first quarter interpolation, the same as in the second to forth quarter interpolation. The payload and the number of changed pixels are the same as the order 2).

From mentioned above, the order 2) and 3), can obtain the same performance which are better than order 1). But after two embedding processes, the changed pixels' number of the order 2) is less than the order 3). We know, the more pixels changed, the less accuracy in later interpolation. So in this paper, we take the second order to embed hidden data. However, the derivations mentioned above are based on the ideal conditions. In fact, the calculation of capacity and the number of changed pixels are more complex, and order 2) is much better than order 3) apparently.

2.5 Additive Interpolation-error Expansion

Firstly, using (4) or (14) to calculate x', with the interpolated value of the missing pixel x, we can obtain the interpolation-error (marked as ***e***) as

$$e = x - x' \tag{15}$$

Secondly, define ***M*** as the number of pixels with the peak point of interpolation-error histogram and ***m*** as the number of pixels with the zero point. Then, do following operation for the interpolation-error expansion

$$e' = \begin{cases} e \pm b, & e = M \text{ and } b \in \{0,1\} \\ e \pm 1, & e \text{ between } M \text{ and } m \\ e \quad , & \text{otherwise} \end{cases} \tag{16}$$

If $M < m$, take $'\pm'$ as $'+'$, else take $'\pm'$ as $'-'$. After revising the interpolation-errors, the image value becomes

$$x'' = x' + e' \tag{17}$$

In the extracting process, using the same interpolation scheme, we can obtain the same interpolated value x', then

$$e' = x'' - x' \tag{18}$$

And do the inverse process of (14), we can extract the hidden data and recover the original interpolation-errors

$$\begin{cases} e = e', & b = 0, \quad e' = M \\ e = e' \mp 1, & b = 1, \quad e' = M \pm 1 \\ e = e' \mp 1, & e' \text{ between } M \pm 2 \text{ and } m \\ e = e', & \text{otherwise} \end{cases} \tag{19}$$

If $M < m$, take $'\mp'$ as $'-'$ and $'\pm'$ as $'+'$, else take $'\mp'$ as $'+'$ and $'\pm'$ as $'-'$. So, the original image value can be obtained by

$$x = x' + e \tag{20}$$

From section 2-4, we know the fact that we can expand interpolation-error by adding 1 bit and subtracting 1 bit alternately. Accordingly, we should choose $\boldsymbol{M} < \boldsymbol{m}$ and $\boldsymbol{m} < \boldsymbol{M}$ alternately in the interpolation process.

3. Proposed Method

In this section, we demonstrate the detail of the proposed reversible watermarking scheme.

3.1 Overhead information

Note that it is possible for image receiver to recover original image and extract the embedded data by some overhead information, which includes the location of overflow/underflow grayscale values, the value of $\boldsymbol{M}$ and $\boldsymbol{m}$, *etc*.

In each embedding process, the pixel changes 1 bit or remains unchanged, so overflow/underflow only happens when the pixel valued 255 or 0. And a pixel valued 255 or 0, is called boundary pixel. We only embed data into pixels valued from 0 to 254 by applying additive interpolation-error expansion or embed data into pixels valued from 1 to 255 by applying subtractive interpolation-error expansion. Firstly, we embed hidden data into pixels valued 254 (additive) or 1 (subtractive). Secondly, scan all pixels, put '1' or '0' into location map when the boundary pixel is unchanged or changed accordingly. Thirdly, record the initial 16 LSB bits of the original image at the beginning of the location map. Fourthly, embed the $\boldsymbol{M}$ and $\boldsymbol{m}$ (each uses 8 bits) into the initial 16 bits of the original image. Finally, embed data (the remaining hidden data and location map) into pixels valued 0 to 253 or 2 to 255.

3.2 Example for Comparing with Other Methods

Consider some examples for demonstrating the accuracy in predicting pixels by the proposed scheme and other methods, as shown in **Fig. 5-6**. In **Fig. 5-6**, **(a)** the original image *Lena*, **(b)** some values of original image (white background pixels are the original sample values, while the black background pixels are the missing values to be interpolated), **(c)** using Sachnev *et al.*[19] to interpolate the missing values, **(d)** using Luo *et al.*[15] (horizontal direction and vertical direction) to interpolate the missing values, **(e)** using Section 2-2 to interpolate the missing values. To compare with other schemes, we show MSE to calculate the distortion between the interpolated values and the original values.

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \| I_{org}(i,j) - I_{mod}(i,j) \| . \tag{21}$$

And $I_{org}(i, j)$-$I_{mod}(i, j) = x - x' = e$, so in **Fig. 5**, the *MSE*s are,

$MSE_{Sac.}$=((136-134)^2+(80-87)^2+(48-50)^2+(105-101)^2+(45-56)^2+(45-44)^2+(59-70)^2+(50-45)^2+(43-42)^2)/9 =38.00

MSE_{Luo}=((136-135)^2+(80-87)^2+(48-50)^2+(105-102)^2+(45-56)^2+(45-45)^2+(59-70)^2+(50-45)^2+(43-42)^2)/9 =36.77

$MSE_{Pro.}$=((136-137)^2+(80-84)^2+(48-50)^2+(105-101)^2+(45-54)^2+(45-45)^2+(59-67)^2+(50-46)^2+(43-43)^2)/9 =22.00

Do the same calculation in the **Fig. 6**, $MSE_{Sac.}$=5.11, MSE_{Luo}=4.00, $MSE_{Pro.}$=3.44.

From the data, we can obtain two important messages. 1) The proposed method is best in interpolating among the three methods in edge-region, smooth-region. 2) e=0 is the peak point of interpolation-error histogram. So the proposed method can get the maximum number of pixels with the peak point of interpolation-error histogram. Therefore the proposed method can obtain the largest capacity accordingly.

164	152	135	110	80	52	51
149	136	113	80	59	48	46
144	123	86	65	47	46	45
127	105	71	45	45	45	45
112	85	54	44	47	40	42
88	59	51	50	41	43	47
63	57	46	45	45	43	43

(a) **(b)**

164	152	135	110	80	52	51
149	134	113	87	59	50	46
144	123	86	65	47	46	45
127	101	71	56	45	44	45
112	85	54	44	47	40	42
88	70	51	45	41	42	47
63	57	46	45	45	43	43

164	152	135	110	80	52	51
149	135	113	87	59	50	46
144	123	86	65	47	46	45
127	102	71	56	45	45	45
112	85	54	44	47	40	42
88	70	51	45	41	42	47
63	57	46	45	45	43	43

164	152	135	110	80	52	51
149	137	113	84	59	50	46
144	123	86	65	47	46	45
127	101	71	54	45	45	45
112	85	54	44	47	40	42
88	67	51	46	41	43	47
63	57	46	45	45	43	43

(c) **(d)** **(e)**

Fig. 5. Comparisons of the interpolation of proposed **(e)**, with Sachnev's [19] **(c)** and Luo's [15] **(d)** in edge-region

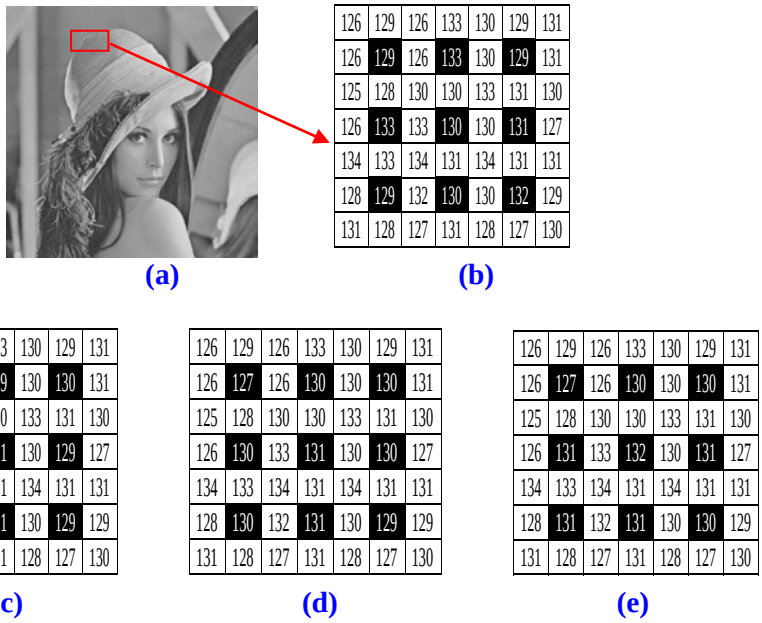

Fig.6 Comparisons of the interpolation of proposed **(e)**, with Sachnev's [19] **(c)** and Luo's [15] **(d)** in smooth-region.

3.3 Encoder

Fig. 7 presents a simple flowchart for implementing the proposed algorithm's embedding and watermark process. Without loss of generality, in the below mention steps, we only consider the single layer embedding process. And the multi-layer embedding process can be similar to multiple single-layer embedding process. Before embedding, all the pixels should be divided into four sets, just like **Fig. 2-(b)**, **(c)**, **(d)** and **(e)**. The detailed embedding steps are listed as follows.

1) Find all pixels according to the **Fig. 2-(b)**, dividing into sample pixels and non-sample pixels (missing pixels).

2) Using Section 2-2 to interpolate missing pixels value x' in the non-border and Section 2-3 to border-pixel.

3) Using (13) to calculate the interpolation-error.

4) Generate the histogram of the interpolation-error, and determine the peak point $\boldsymbol{M}$, and the zero point $\boldsymbol{m}$, record the first 16 original LSB bits at the beginning of location map $\boldsymbol{B}$ and embed the $\boldsymbol{M}$ and $\boldsymbol{m}$ (each uses 8 bits) at the beginning 16 bits of original image.

5) Scan all interpolation-errors. If the original value of pixel x is equal to 255 or 0, put a '1' into location map $\boldsymbol{B}$ and move to the next one. Else, if expand or embed x (x = 254) by (15) (hidden data as $\boldsymbol{W_1}$) and obtain x'' is equal to 255 or 0, put a '0' into location map $\boldsymbol{B}$ and move to the next one.

6) Assemble the location map $\boldsymbol{B}$ and the remaining watermark data $\boldsymbol{W_2}$ to form payload $\boldsymbol{P}$. Skip the beginning 16 pixels. From the 17th pixel on, in the remaining pixels (the value is 0 to 254 or 1 to 255) if the interpolation-error is $\boldsymbol{M}$, embed a bit; else the interpolation-error is between $\boldsymbol{M}$ and $\boldsymbol{m}$, expand the interpolation-error by adding or subtracting 1 bit.

7) Repeat operation as steps 2) to 6) twice. The first time, embed or expand the pixels by additive and the second time by subtraction.

Repeat the same operation as steps 2) to 7) according to the **Fig. 2-(b)**, **(c)**, or **(d)**.

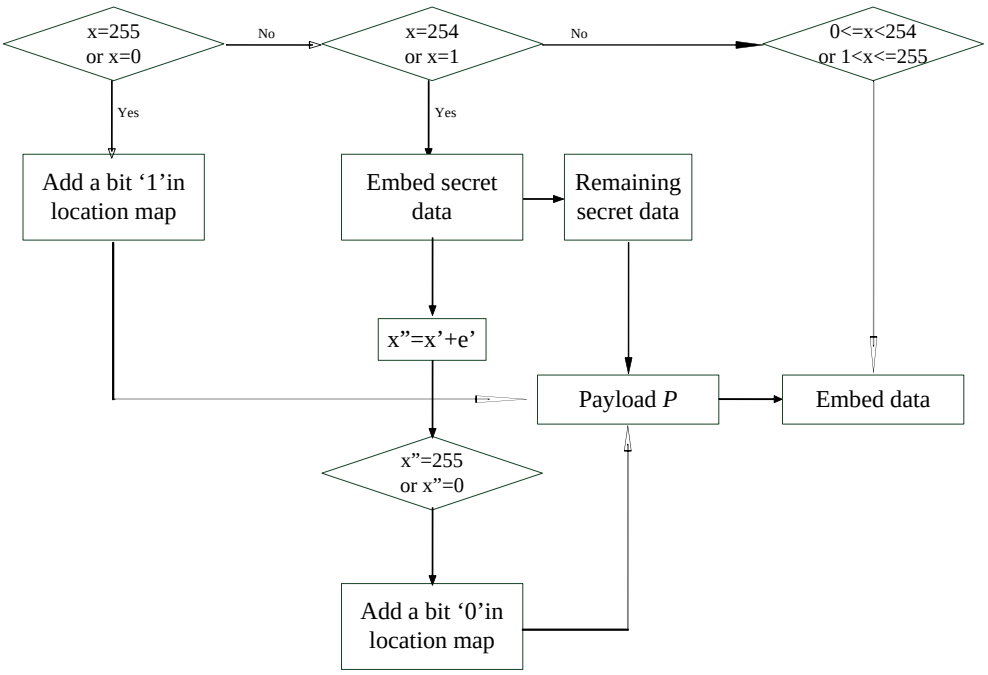

Fig. 7. The flowchart of embedding process.

3.4 Decoder

On the other hand, the extraction of hidden data is the recovery of embedding process. The extraction process in the watermark image can be performed as follows, the flowchart shown in **Fig.8**.

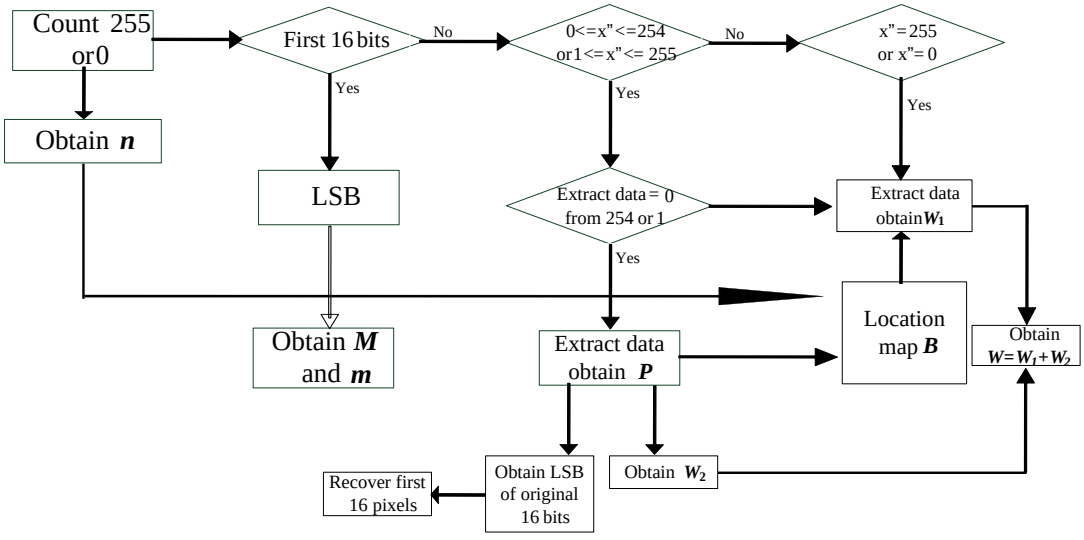

Fig. 8. The flowchart of extracting process.

1) Find all pixels according to the **Fig. 2-(d)**, dividing into sample pixels and non-sample pixels.

2) Using Section 2-2 to interpolate non-sample pixels value x' in the non-border and Section 2-3 to border-pixel.

3) Using (16) to calculate the changed interpolation-error $\boldsymbol{e}'$.

4) Count the number of pixel 0 or 255, marked as $\boldsymbol{n}$.

5) Calculate the LSB of the beginning 16 bits of watermark image, and obtain the $\boldsymbol{M}$ and $\boldsymbol{m}$ from the 16 bits (each 8 bits).

6) If $\boldsymbol{x}''=1$ (or $\boldsymbol{x}''=254$) and the extracted bit is 1 or $1<\boldsymbol{x}''<=255$ (or $0<=\boldsymbol{x}''<254$), extract the hidden data by (17) and obtain payload $\boldsymbol{P}$ (including the LSB of original first 16 bits, $\boldsymbol{n}$ bits location map $\boldsymbol{B}$ and watermark data $\boldsymbol{W_1}$.

7) When x"=0 (or x"=255), if the marked bit in location map $\boldsymbol{B}$ is '0', extract the hidden data, obtain watermark data $\boldsymbol{W_1}$ and recover the interpolation-error by (17), else, if the marked bit is '1', keep the interpolation-error unchanged.

8) Repeat operation as steps 2) to 7) twice. The first time, extract or narrow the pixels by subtraction and the second time by additive. Contrast with the embedding process 7).

9) Assemble the watermark data $\boldsymbol{W_1}$ with $\boldsymbol{W_2}$, obtain the original watermark data $\boldsymbol{W}$. And using (18) to recover the original image.

Repeat the same operation as steps 2) to 9) according to the **Fig. 2-(d)**, **(c)** or **(b)**.

Fig. 9. Benchmark images. **(a)** Lena, **(b)** Baboon, **(c)** Airplane, **(d)** Lake, **(e)** Peppers, **(f)** Couple.

4. Experimental Result

Section 2 offers the proof that the proposed scheme has the reversibility. In this section, we have implemented the proposed reversible watermarking scheme using MATLAB compared with Hwang *et al*. [10], Lin *et al*. [12], Hu *et al*. [13], Luo *et al*. [15] in single-layer and C. C. Lin *et al*. [12][20] and Luo *et al*. [15] in multi-layer using typical 512×512 grayscale images in **Table 1** and **Fig. 10**. The grayscale images, such as *Lena*, *Baboon*, *Airplane*, *Lake*, *Pepper*

and *Couple* are placed in **Fig. 9**. Images available at [21] are used. More details will be given in **Table 2**, comparing with Luo *et al*. by 2 directional edge-guided interpolation.

Table 1. PSNR values of the proposed method other four methods (single-layer capacity).

Scheme	Lena image		Baboon image		Airplane image		Sailboat image	
	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR	Capacity	PSNR
Hwang *et al*.[10]	5,336	48.22	5,208	48.20	15,300	48.40	7,051	48.25
Lin *et al*.[12]	59,900	46.60	19,130	47.61	80,006	47.99	37,644	47.60
Hu *et al*.[13]	60,241	48.69	21,411	48.34	77,254	48.86	28,259	48.40
Luo et al.[15]	60,564	48.66	19,806	48.42	85,516	48.90	40,282	48.47
Proposed	64,856	48.70	20,223	48.39	92,275	48.97	44,324	48.51

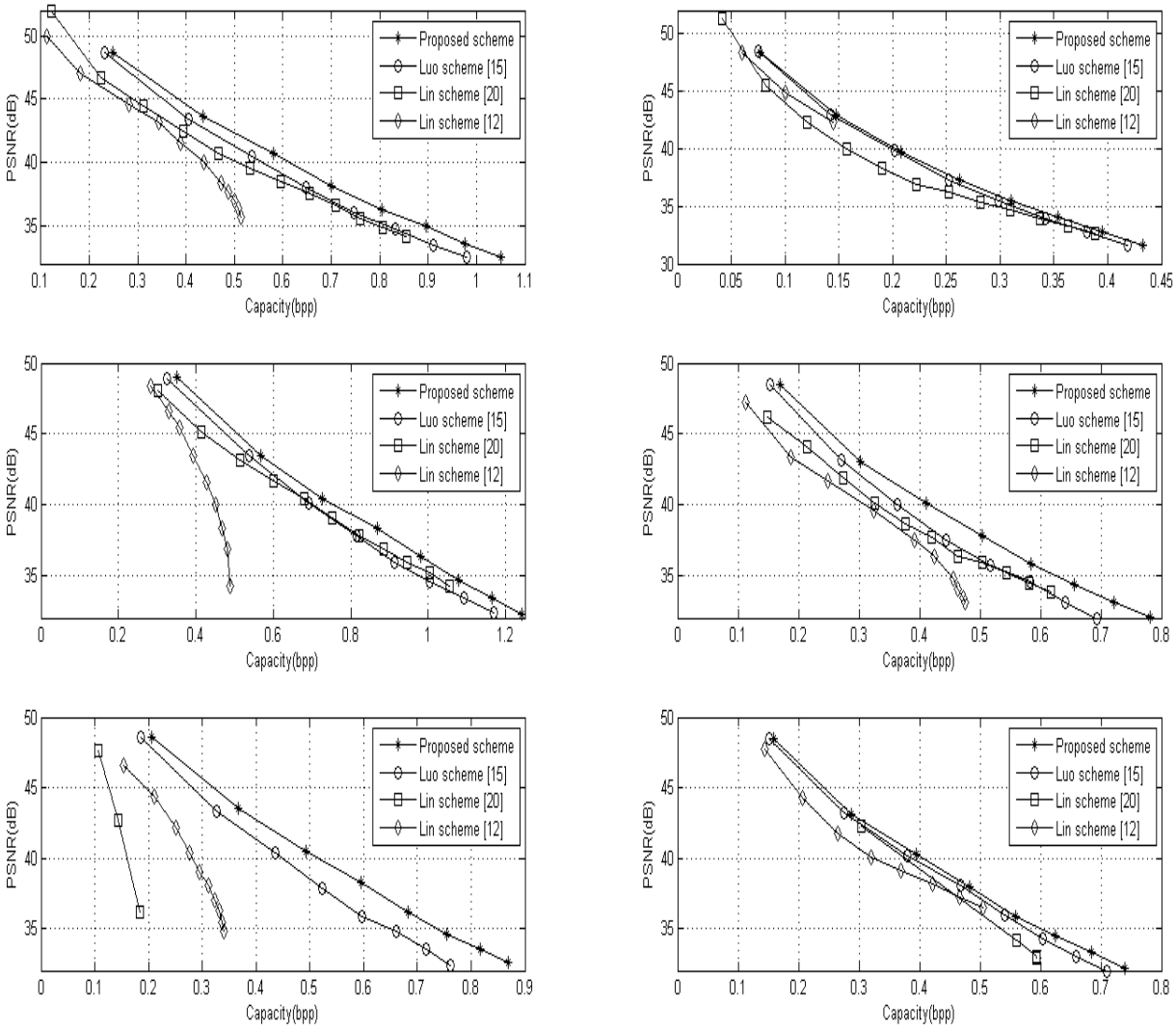

Fig. 10. Comparsions among the proposed scheme and other schemes.

Table 2. PSNR values of the proposed scheme and Luo [15] scheme(multi-layer capacity).

Layer		Lena		Baboon		Airplane	
		Luo et al.	Proposed	Luo et al.	Proposed	Luo et al.	Proposed
1	Payload	60,564	64,856	19,806	20,223	85,516	92,275
	PSNR	48.67	48.70	48.43	48.39	48.91	48.98
2	Payload	106,008	113,820	37,337	38,613	141,231	147,522
	PSNR	43.45	43.61	42.94	42.87	43.42	43.50
3	Payload	140,547	152,067	52,916	54,492	181,479	190,443
	PSNR	40.53	40.67	39.81	39.70	40.09	40.38
4	Payload	169,900	183,608	66,236	68,679	213,457	227,516
	PSNR	38.08	38.16	37.27	37.30	37.82	38.31
5	Payload	195,872	210,850	78,303	81,254	239,578	257,103
	PSNR	36.06	36.22	35.50	35.46	35.93	36.34
6	Payload	218,147	234,581	89,581	92,856	263,069	282,895
	PSNR	34.79	34.93	33.91	34.01	34.57	34.68
7	Payload	238,693	256,066	99,882	103,502	286,531	305,277
	PSNR	33.49	33.62	32.76	32.78	33.42	33.44
8	Payload	256,870	275,577	109,647	113,426	306,545	325,688
	PSNR	32.54	32.58	31.61	31.66	32.33	32.30

Layer		Lake		Peppers		Couple	
		Luo et al.	Proposed	Luo et al.	Proposed	Luo et al.	Proposed
1	Payload	40,282	44,324	48,837	54,062	39,868	41,703
	PSNR	48.48	48.52	48.56	48.61	48.48	48.50
2	Payload	71,203	79,404	85,558	96,365	72,280	75,280
	PSNR	43.11	43.09	43.35	43.49	43.17	43.14
3	Payload	95,439	107,771	114,257	129,259	99,631	103,672
	PSNR	40.00	40.11	40.34	40.45	40.16	40.25
4	Payload	116,522	131,982	136,968	156,000	122,655	126,699
	PSNR	37.51	37.84	37.86	38.31	38.07	37.91
5	Payload	135,553	153,362	156,286	178,742	141,877	146,416
	PSNR	35.76	35.82	35.89	36.12	35.95	35.80
6	Payload	152,730	172,131	173,162	197,771	158,311	163,622
	PSNR	34.55	34.38	34.78	34.63	34.25	34.47
7	Payload	167,886	189,251	187,474	214,017	172,880	179,368
	PSNR	33.13	33.07	33.53	33.52	33.02	33.28
8	Payload	181,888	205,001	199,778	227,691	186,168	194,021
	PSNR	31.90	32.10	32.40	32.57	31.99	32.21

From **Table 1**, we know in the single-layer capacity, the proposed scheme is better than most of the other schemes (except the Hu *et al.* scheme in *Baboon*, but only 1200 less than Hu *et al.*), especially in *Airplane* the capacity is 7.9% larger than the other four methods at least. The reason for the larger capacity is shown in **Fig. 5-6**. In the single-layer, the pixel

value will be either added or subtracted by one at most. Therefore in the worst case the MSE will be equal to one. In this case, the PSNR of the stego-image is $10\times\log_{10}(255^2/\text{MSE}) \geq 48.13\text{dB}$. While using the same method, the best case is that embedding the hidden data in all pixel, and only when the hidden data is '1', the pixel shifts one bit and when the hidden data is '0' ,the pixel keeps unchanged. Therefore the MSE is equal to 1/2, under the constraint of 0-1 equal probability distribution. In this case, the capacity of single-layer is 256×256 = 65,536 and the PSNR is 51.17dB. But this is difficult to be achieved, for the interpolation scheme can't accurately interpolate all pixels and the overhead problem can't be ignored in the embedding process.

For multi-layer embedding, we compare results with Luo [15] and C. C. Lin [12][20] in six test images. Just like in the single-layer embedding, the proposed scheme achieves better performance in the *Airplane* than *Baboon*. For *Airplane* is the smoothest among six test images and *Baboon* is the most complex, the overall prediction is more accurate in *Airplane* than *Baboon*. As a result, we can get larger capacity in *Airplane* than *Baboon*.

In **Table 2** the PSNR values of the proposed method and Luo *et al*. [15] method are provided. Luo *et al*. utilizing two directions to interpolate the pixels, while 4 directions are used in the proposed scheme. Therefore, the accuracy of interpolation is improved in all six test images and larger capacity is achieved. In the embedding process, Luo *et al*. shifts one grey scale value by added and subtracted at each embedding process, while the proposed scheme uses either added or subtracted one bit. So the PSNR values are improved in the same embedding capacity.

5. Conclusion

In this paper, a high-capacity algorithm combined with an additive edge-guided interpolation and histogram shifting has been proposed for reversible watermarking with low distortion and high capacity. The new four-direction edge-guided interpolation that has also been proposed is the key skill for overcoming the limit capacity problem of histogram shifting by Ni *et al*. [9]. And a good order of embedding process has the capacity to keep distortion low. Experimental results show that the proposed method has better results than Hwang *et al*. [10], Lin *et al*. [12] and Luo *et al*. [15] and other schemes in most cases.

Acknowledgement

This work was supported by the Natural Science Foundation of Shanghai, China (09ZR1412400, 11ZR1413200), the Innovation Program of Shanghai Municipal Education Commission (10YZ11, 11YZ10).

References

[1] J. M. Barton, “Method and apparatus for embedding authentication information within digital data,” *U.S. Patent* 5 646 997, 1997. Article (CrossRef Link)

[2] D. Vleeschouwer, J. F. Delaigle and B. Macq, “Circular interpolation of bijective transformations in lossless watermarking for media asset management,” *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 97-105, Mar. 2003. Article (CrossRef Link)

[3] M. Goljan, J. Fridrich and R. Du, “Distortion-free data embedding for images,” in *Proc. of the 4th Information Hiding Workshop, Lecture Notes in Computer Science*, Springer, New York, vol. 2137, pp. 27-41, Apr. 2001. Article (CrossRef Link)

[4] J. Fridrich, M. Goljan and R. Du, "Invertible authentication watermark for JPEG images," in *Proc. of SPIE, Information Technology: Coding and Computing*, pp. 223-227, Jan. 2001. Article (CrossRef Link)
[5] M. U. Celik, G. Sharma, A. M. Tekalp, et al., "Lossless generalized LSB data embedding," *IEEE Transactions on Image Processing*, vol. 14, pp. 2, pp. 253-266, Feb. 2005. Article (CrossRef Link)
[6] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890-896, Aug. 2003. Article (CrossRef Link)
[7] A. M. Alattar, "Reversible watermarking using difference expansion of quads," in *Proc. of IEEE International Conference on Acoustics*, Speech, Signal Processing, vol. 3, pp. 377-380, May 2004. Article (CrossRef Link)
[8] L. Kamstra and H. J. A. M. Heijmans, "Reversible data embedding into images using wavelet technique and sorting," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2082-2090, Dec. 2005. Article (CrossRef Link)
[9] Z. Ni, Y. Q. Shi, N. Ansari, et al., "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354-362, Mar. 2006. Article (CrossRef Link)
[10] J. H. Hwang, J. W. Kim and J. U. Choi, "A reversible watermarking based on histogram shifting," *Lecture Notes in Computer Science*, vol. 4283, pp. 348-361, 2006. Article (CrossRef Link)
[11] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721-730, Mar. 2007. Article (CrossRef Link)
[12] C. C. Lin and N. L. Hsueh, "A lossless data hiding scheme based on three-pixel block differences," *Pattern Recognition*, vol. 41, no, 4, pp. 1415-1425, Apr. 2008. Article (CrossRef Link)
[13] Y. Hu, H. K. Lee and J. Li, "De-based reversible data hiding with improved overflow location map," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 2, pp. 250-260, Feb. 2009. Article (CrossRef Link)
[14] P. Tsai, Y. C. Hu and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," *Signal Processing*, vol. 89, no. 6, pp. 1129-1143, 2009. Article (CrossRef Link)
[15] L. X. Luo, Z. Y. Chen, M. Chen, et al., "Reversible image watermarking using interpolation technique," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 187-193, Mar. 2010. Article (CrossRef Link)
[16] B. E. Bayer, "Color imaging array," *U.S. Patent* 3 971 065, Jul. 1976. Article (CrossRef Link)
[17] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2226-2238, Aug. 2006. Article (CrossRef Link)
[18] X. Li, B. Gunturk and L. Zhang, "Image demosaicing: a systematic survey," in *Proc. of SPIE*, vol. 6822, pp. 68221J, 2008. Article (CrossRef Link)
[19] V. Sachnev, H. J. Kim, J. Nam, et al., "Reversible watermarking algorithm using sorting and prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989-999, Jul. 2009. Article (CrossRef Link)
[20] C. C. Lin, S. P. Yang and N. L. Hsueh, "Lossless data hiding based on difference expansion without a location map," in *Proc. of Congress on Image and Signal Processing*, vol. 2, pp. 8-12, May 2008. Article (CrossRef Link)
[21] http://decsai.ugr.es/cvg/dbimagenes/g512.php.

Ningjie Dai received the B.S. degree in electronic and information engineering from China Jiliang University, China, in 2009. He is currently pursuing the M. S. Degree in Communication and Information Engineering in the School of Communication and Information Engineering, Shanghai University, China. His research interests include image processing and data hiding.

Guorui Feng received the B.S. and M. S. degree in computational mathematic from Jilin University, China, in 1998 and 2001 respectively. He received Ph. D. degree in electronic engineering from Shanghai Jiaotong University, China, 2005. From January 2006 to December 2006, he was an assistant professor in East China Normal University, China. During 2007, he was a research fellow in Nanyang Technological University, Singapore. Now he is with the School of Communication and Information Engineering, Shanghai University, China. His current research interests include image processing, image content security and computational intelligence.

Qian Zeng received the B.S. degree in communication engineering from Jilin University, China, in 2007. Now he is pursuing the M.S. degree in Communication and Information Engineering at Shanghai University.