Received December 18, 2019, accepted January 1, 2020, date of publication January 8, 2020, date of current version January 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964830

A Reversible Steganography Method With Statistical Features Maintained Based on the Difference Value

TENGFEI LI, HUIFENG LI, LIANG HU, AND HONGTU LI
Computer Science and Technology, Jilin University, Changchun 212006, China

Corresponding author: Hongtu Li (lihongtu@jlu.edu.cn)

This work was supported in part by the National Key Research and Development Plan of China under Grant 2017YFA0604500, in part by the National Sci-Tech Support Plan of China under Grant 2014BAH02F00, in part by the National Natural Science Foundation of China under Grant 61701190, in part by the Youth Science Foundation of Jilin Province of China under Grant 20160520011JH and Grant 20180520021JH, in part by the Youth Sci-Tech Innovation Leader and Team Project of Jilin Province of China under Grant 20170519017JH, in part by the Key Technology Innovation Cooperation Project of Government and University for the Whole Industry Demonstration under Grant SXGJSF2017-4, and in part by the Key scientific and technological Research and Development Plan of Jilin Province of China under Grant 20180201103GX.

ABSTRACT Reversible data hiding (RDH) is a technique that slightly alters digital media (e.g. images or videos) to embed secret messages while the original digital media can be completely recovered without any error after the hidden messages have been extracted. In the past more than one decade, hundreds of RDH algorithms have been reported, and among these algorithms, the difference histogram shifting (DHS) based methods have attracted much attention. With DHS-based RDH, high capacity and low distortion can be achieved efficiently. But there occurs one problem that, with DHS, the difference values to embed secret bits are explored, and the other difference values are shifted to create vacant spaces, it will cause the difference value histogram changing significantly and draw the attention of steganalyzers. So, this paper proposed a new idea for RDH based on the difference value and with statistical features maintained (SFM) with simple implementation and high scalability, we embed the secret messages by keeping the difference values that need to be modified in the original range, and the other difference values would not be shifted. In addition, we need the original difference values as the key to extract the secret messages. In order to expand the embedding capacity further, we designed two algorithms that embed message in two different difference values and four different difference values, and these two methods are named SFM_A and SFM_B respectively. SFM_B can support greater amount of embedded message than SFM_A, but brings greater changes to the original image, which could lead to the decline of PSNR and SSIM. The experimental results show that through our method, the histogram of difference values is well maintained, and the degree of distortion of the image is improved at the same time.

INDEX TERMS Reversible steganography, difference value, statistical feature, embedding capacity, PSNR, SSIM.

I. INTRODUCTION

Reversible data hiding (RDH) can imperceptibly hide data into digital images, and more importantly, the original image can be reconstructed completely after the embedded data have been extracted out [3]. RDH can find many applications, e.g., for medical and military image processing.

The classical RDH schemes have been proposed based on three fundamental strategies: lossless compression [1], difference expansion (DE) [2], and histogram shifting (HS) [3]. Among them, the HS based methods have attracted much attention. Nowadays, difference histogram shifting (DHS) [22] and prediction-error histogram shifting (PEHS) [23] based methods have received much more attention because of their large embedding capacity and high fidelity. For these two advanced approaches, the main idea is to explore the correlation between neighboring pixels in a host image, and thus a difference or prediction-error histogram with higher peaks can be generated. Then the additional message can be reversibly embedded into the host

The associate editor coordinating the review of this manuscript and approving it for publication was Yap-Peng Tan.

image via modifying the difference histogram or prediction-error histogram.

Numerous DHS and PEHS based algorithms have been proposed in the past few years: Khan *et al.* [4] discussed the performances of different reversible watermarking schemes on the basis of various characteristics of watermarking, and the major focus of this survey is on prediction-error expansion (PEE) based reversible watermarking techniques. Wu *et al.* [5] proposed an RDH algorithm that enhances the contrast of a host image to improve its visual quality instead of trying to keep the PSNR value high. By splitting the highest two histogram bins into four, data embedding and histogram equalization are performed simultaneously, and it seems that this is the first algorithm that achieves image contrast enhancement by RDH, they chose the highest two bins to get bits embedded, improved the embedded capacity to some extent. However, the statistical features of the image were modified seriously, and looking for the largest number of two-pixel values also brought additional time overhead. Li *et al.* [6] proposed a new RDH method based on PEE for multiple histograms. Unlike the previous methods, in this paper, they considered a sequence of histograms and devised a new embedding mechanism based on multiple histograms modification (MHM) and got a better performance than two-dimensional PEH-based methods, but in this method of difference expansion, most of the original image's pixel values also have been modified. Qiu *et al.* [7] proposed a novel RDH method with an adaptive embedding capability by extending the generalized integer transformation (GIT) and achieved a better embedding efficiency of the net payload, the method could embed $(n-1) \times log_2 k$ bits into n pixels, this is a considerable amount of embedded capacity, but for the PSNR value, it is lower than some typical RDH methods and its safety has been affected to some extent. Kumar and Chand [8] proposed a new RDH scheme that used pixel value adjusting feature, it has two phases. In the first phase, it scans the image diagonally from left to right and hides some of the secret messages into the odd valued pixels, and in the second phase, it also scans the image diagonally but in the right to left order, it hides the secret messages into the even valued pixels, it is easy to perform, but it modified a lot of pixels and also has some overhead in hiding secret messages. Chen *et al.* [9] proposed a new algorithm based on the alteration of difference values relating to original image and with the characteristics of ease of implementation, it employed the block-based, multi-round prediction to look for enhanced performances, they applied four rounds for data embedding as depicted in the paper, and used different weighting factors in different rounds for producing predicted image, as the number of rounds increases, the effect of the image is enhanced, but this also posed the problem of increased message embedding and image restore time overhead. Luo *et al.* [10] presented a novel sparse recovery based RDH method using the human visual system (HVS). They proposed a sparse recovery based predictor to improve the low accuracy of existing predictors, built the concentrated PEH to obtain good embedding performance, designed a new embedding strategy based on just noticeable difference (JND), and utilized the PEE technique to embed data. A series of methods based on pixel value ordering (PVO) [11], [12] have proposed to produce camousflaged pixels of good image quality combined with PEE. Ou *et al.* [13] proposed a novel RDH framework based on the so called pairwise PEE, and used the pairwise PEE to embed data in a 2D PEH. Then in [14], Ou *et al.* proposed two new techniques for histogram generation and modification, called adaptive pixel pairing (APP) and adaptive mapping selection respectively, to further improve the high-dimensional pairwise PEE.

Some scholars have tried to apply RDH to an encrypted domain: Zhang *et al.* [15] proposed some new methods to improve the performance by reversing the order of encryption and vacating room. Li *et al.* [16] proposed a method to embed additional data into block compressed sensing (BCS) and got a nice performance in encrypted images. Qian and Zhang [17] proposed a novel scheme of reversible data hiding in encrypted images using distributed source coding. Fangjun *et al.* [18] proposed a new simple yet effective framework for RDH in encrypted domain, the plain image were first divided into sub-blocks with the size of $m \times n$, and before data hiding, they encrypted each block with the same key and then disordered the blocks, because the correlation between neighboring pixels/coefficients have been preserved, some RDH methods could be applied directly in encrypted domain. Xiong [24] proposed a scheme of reversible watermarking using a complementary embedding strategy in the spatial domain. A complementary embedding strategy is designed to increase the embedding capacity and decrease the distortion of the watermarked image in the vertical direction embedding. With the recent popularity of artificial intelligence, there are some schemes that apply artificial intelligence methods to steganography. Duan et al. [25] proposed a new image steganography scheme based on a U-Net structure. The scheme compresses and distributes the information of the embedded secret image into all available bits in the cover image, which not only solves the obvious visual cues problem, but also increases the embedding capacity.

These papers used their own methods to solve the difficulties of using steganography due to image encryption, such as estimating some pixels before encryption, so that additional data could be embedded in the estimating errors, combining the compression and encryption of the image, compressing a series of selected bits taken from the encrypted image to make room for secret messages, and so on, these methods have made great strides in how to successfully apply RDH to the encrypting images, but how to improve the performance of RDH itself and apply it to the encryption domain are still problems we need to study.

With the rapid development of steganalysis technology and the coming era of cloud computing, a content owner may not trust the server manager, he/she will encrypt the data first and then upload it to the server, and in this case, steganography could be an excellent solution, and with the development of steganalysis, kinds of secure methods for the steganography are attracting more and more attention. As the statistical features of the image is an important analysis of steganalysis basis, so maintaining the statistical features of images is one of the most important ways to enhance the security of steganography. At the same time, storage is also one of the concerns of cloud computing, we hope to save as much storage space as possible when encrypting data. Therefore, how to improve the embedded capacity, or balance the security and embedded capacity (usually, higher embedding capacity means lower security) is also an important issue which steganographic algorithm designers need to consider.

We can see that scholars are committed to developing algorithms that improve RDH performance, but few take into account the statistical features of the carrier. In this paper, we propose an RDH method ensuring the embedded capacity while maintaining the statistical features of the image, we name our method SFM (Statistical Features Maintained), which could resist some steganalysis for traditional RDH algorithms, such as "histogram analysis". For convenience, we use gray-scale images for illustration but our proposed method is applicable to both the gray-scale and color images. The main idea of our method is keeping the difference values that need to be modified in the original range, for example, if we choose the pixel blocks with difference pair $\{-1, 0\}$ to embed secret messages, after the embedding process, the difference values of these blocks are still in $\{-1, 0\}$, and that is an example of embedding a piece of secret message in two difference values. In this paper, we design to embed message in two different difference values (SFM_A) and four different difference values (SFM_B), SFM_B increases the embedding capacity, but at the cost of a larger modification to the original image, and later we expanded a comparative experiment on the statistical histogram and the distortion level (measured by PSNR and SSIM) of the embedded image.

Our main contributions in this work are listed as follows:

1) We propose two new DHS methods (named SFM_A and SFM_B, respectively) for RDH. The main idea of the proposed schemes is ensuring the embedded capacity while maintaining the global statistical features of an image by dividing the image into sub-blocks.
2) Some experiments are conducted to compare our proposed schemes to some other state-of-the-art algorithms on the histogram of the difference value, PSNR, SSIM and the capacity-distortion trade-off. The results demonstrate, compared to other state-of-the-art algorithms, SFM_A obviously has the best performance at relatively lower embedding rate, SFM_B outperforms other algorithms at higher embedding rate and has a wider range in the embedding rate because its PSNR falls gently with the increasing of embedding rate.

The rest of this paper is organized as follows. In Section II, we review some related works about RDH. In Section III, a new method for RDH based on difference value is proposed, this section has 2 parts: embedding message in two different difference values and four different difference values, which have their own characteristics. Experimental results are presented in Section IV. Finally, we conclude the paper in Section V.

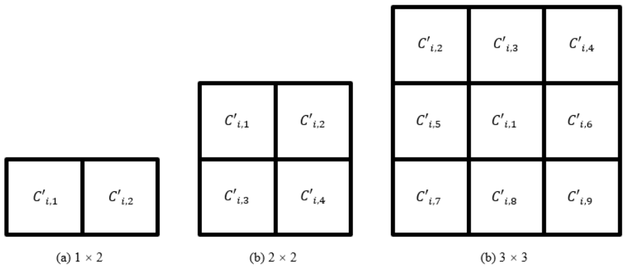


FIGURE 1. Different sub-blocks of the host image.

II. RELATED WORKS

We start our presentation by introducing Tian's difference expansion (DE) algorithm [2], that is an important work of RDH. In DE algorithm, the host image is divided into pixel pairs (as seen in FIGURE 1.(a)), and the difference value of two pixels in a pair is expanded to carry one data bit. This method can provide an embedding rate (ER) up to 0.5 bit per pixel (BPP). In particular, Tian employed a location map to record all expandable locations, and afterwards, the technique of location map was widely adopted by most RDH algorithms. Later on, Tian's work has been improved in many aspects.

Alattar [19] generalized DE to a triple or a quad of pixels (as seen in FIGURE 1.(b)) which can increase the maximum ER from 0.5 BPP to 0.75 BPP. Kamstra and Heijmans [21] utilized low-pass image to predict expandable locations so that the location map can be remarkably compressed.

HS is another fundamental strategy and has attracted much attention. They can also be divided into three different approaches, i.e., (original) HS [3], DHS [22], and PEHS [23]. In Ni *et al.*'s method [3], the data embedding was implemented by shifting one-dimensional image histogram, and [22], [23] applied the histogram translation to other histograms. The main idea of HS is as follows.

Consider a gray-scale image I. For a given integer a, $1 \leq a \leq 253$, the hidden data is embedded into I in the following way to get the marked image I':

$$I'_{i,j} = \begin{cases} I_{i,j} - 1, & \text{if } I_{i,j} < a \\ I_{i,j} - m, & \text{if } I_{i,j} = a \\ I_{i,j} + m, & \text{if } I_{i,j} = a + 1 \\ I_{i,j} + 1, & \text{if } I_{i,j} > a + 1. \end{cases} \tag{1}$$

where (i, j) is a pixel location and $m \in \{0, 1\}$ is a data bit to be embedded. The decoder can extract the embedded data and restore the original image by simply reading marked pixel values.

1) If $I'_{i,j} < a - 1$, there is no hidden data in the pixel and its original value is $I'_{i,j} + 1$;

2) If $I'_{i,j} \in \{a-1, a\}$, the pixel is used to carry hidden data and its original value is a. The embedded data bit is $m = a - I'_{i,j}$;
3) If $I'_{i,j} \in \{a+1, a+2\}$, the pixel is also used to carry hidden data and its original value is $a+1$. The embedded data bit is $m = I'_{i,j} - (a+1)$;
4) If $I'_{i,j} > a+2$, similar to the first case, there is no hidden data and the original value is $I'_{i,j} - 1$.

The idea of DE could be applied in Ni *et al.*'s method [3], the variable 'a' could represent a difference value. The host image is divided into sub-blocks and pixels in a host image is represented by $C_{i,j}(1 \leq i \leq N_I, 1 \leq j \leq m_I \times n_I)$.To avoid the saturation (i.e., the overflow or underflow) during the embedding process, the saturated pixels (pixels with value 0 or 255) have to be preprocessed by modifying one gray-scale unit and noting in a location map L (initialized to empty) as that in [20]. To do this, visit pixels sequentially and append a bit '0' to L when $C_{i,j} \in [1, 254]$. If $C_{i,j} \in \{0, 255\}$, append a bit '1' to L and modify $C_{i,j}$ to $C'_{i,j}$ by using the following equation:

$$C'_{i,j} = \begin{cases} 254, & \text{if } C_{i,j} = 255 \\ 1, & \text{if } C_{i,j} = 0 \\ C_{i,j}, & \text{otherwise.} \end{cases} \quad (2)$$

The length of L is equal to the number of pixels with values 0, 1, 254 and 255, and L could be used to restore the host image later, As seen, after overflow or underflow processing, a new image whose pixels are in the range [1, 254] is obtained. The images with 1×2, 2×2, 3×3 sub-blocks are exemplified in FIGURE 1, the difference value in each block is computed as:

$$D_{i,j} = C'_{i,j} - C'_{i,1} \quad (3)$$

where $1 \leq i \leq N_I, 2 \leq j \leq m_I \times n_I$ in Eq.(3).

As higher points of the difference histogram are situated at the points 0 and ± 1, we can divide Bin 0 and Bin -1 of the difference histogram into the inner region, and the rest of the bins are divided into the outer region. The embedding algorithm of the RDH scheme can be described as follows.

$$C''_{i,j} = \begin{cases} C'_{i,j} - 1, & \text{if } D_{i,j} < -1 \\ C'_{i,j} - b, & \text{if } D_{i,j} = -1 \\ C'_{i,j} + b, & \text{if } D_{i,j} = 0 \\ C'_{i,j} + 1, & \text{if } D_{i,j} > 0. \end{cases} \quad (4)$$

In Eq.(4), $b \in \{0, 1\}$ is a message bit to be embedded, and $C''_{i,j}$ is the corresponding pixel value in the marked image. And the message extraction and image restoration can be described as follows.

$$b^* = \begin{cases} 0, & \text{if } C''_{i,j} - C_{i,1} = 0, -1 \\ 1, & \text{if } C''_{i,j} - C_{i,1} = 1, -2. \end{cases} \quad (5)$$

$$C'_{i,j}{}^* = \begin{cases} C''_{i,j} - 1, & \text{if } C''_{i,j} - C''_{i,1} > 0 \\ C''_{i,j} + 1, & \text{if } C''_{i,j} - C''_{i,1} < -1 \\ C''_{i,j}, & \text{otherwise.} \end{cases} \quad (6)$$

where b^* and $C'_{i,j}{}^*$ represent the extracted message bit and the restored pixel value. After message extraction and image restoration, the original image can be recovered via using the extracted location map L. If the appended bit in the location map L is '1', then

$$C_{i,j}{}^* = \begin{cases} 255, & \text{if } C'_{i,j}{}^* = 254 \\ 0, & \text{if } C'_{i,j}{}^* = 1. \end{cases} \quad (7)$$

Else,

$$C_{i,j}{}^* = C'_{i,j}{}^* \quad (8)$$

At this point, the HS-based RDH process ends, the whole process is shown in FIGURE 2, and we can clearly see the feasibility of this algorithm. While considering the DHS method, we could discover that, DHS algorithm will destroy the statistical features of the image difference histogram, and it will lead to the attention of steganalyzer. So, we propose an improved idea to maintain the statistical features, keep the difference values that need to be modified in the original range and difference histogram is not going to be shifted. We design to embed messages in two different difference values (SFM_A) and four different difference values (SFM_B), the methods are shown in the next part.

III. PROPOSED METHOD

In this section, we introduce our proposed method in detail, we use gray-scale images for illustrating, but our method is easily extended to color images. Refer to the DHS method, the idea of this scheme is that after the embedding process, the difference values that need to be modified are in the original range, for example, if we choose the pixels with difference value pair $\{-1, 0\}$ to embed secret messages, after the embedding process, the difference values of these pixels are still in $\{-1, 0\}$. Here we propose two algorithms using the methods named SFM_A and SFM_B, the first one embeds secret messages in pixels with difference values -1 and 0, the second one embeds secret messages in pixels with difference values -2, -1, 0 and 1.

A. GENERAL PROCESS

As introduced in Section II, the host image is firstly divided into sub-blocks as FIGURE 1 shows, pixels in a host image is represented by $C_{i,j}(1 \leq i \leq N_I, 1 \leq j \leq m_I \times n_I)$, where N_I represents the number of sub-blocks, and the size of sub-blocks is represented by $m_I \times n_I$. The difference value in each block is computed as Eq.(3) and we get the difference value in each block $D_{i,j}$, $1 \leq i \leq N_I, 2 \leq j \leq m_I \times n_I$. The next two algorithms that will be presented are both based on this process.

B. SFM_A:EMBEDDING SECRET MESSAGES INTO PIXELS WITH DIFFERENCE VALUE −1 AND 0

This algorithm embeds secret messages in pixels with difference value -1 and 0, and the embedding algorithm of the

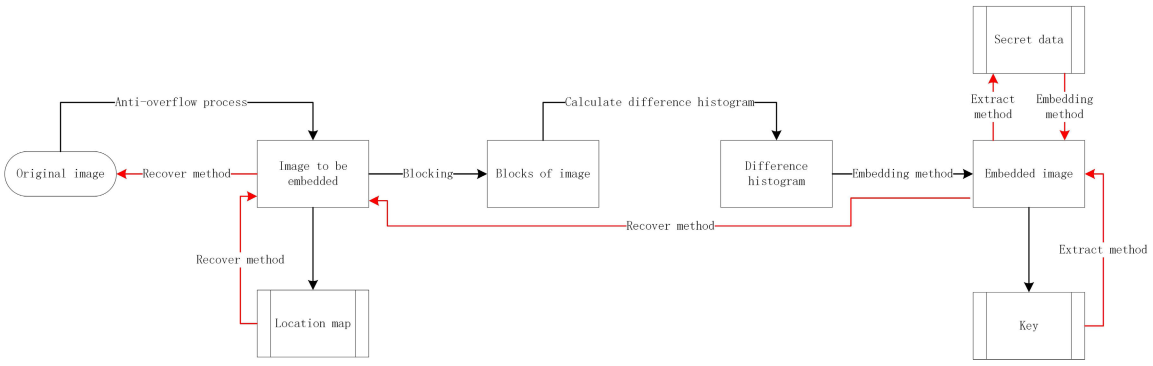


FIGURE 2. Secret messages embedding and extracting process.

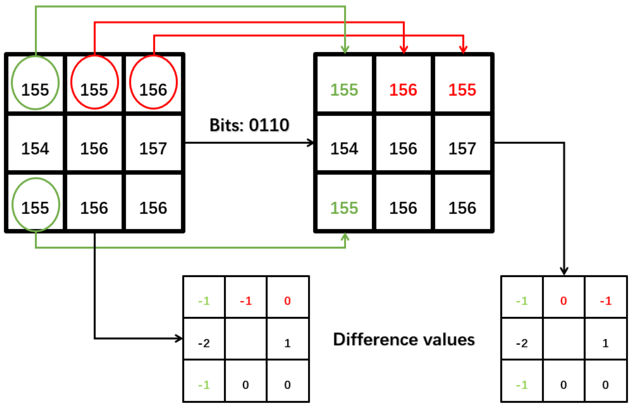


FIGURE 3. Bits 0110 embed in one sub-block(3×3) applying SFM_A.

RDH scheme can be described as follows.

$$C''_{i,j} = \begin{cases} C'_{i,j} + b, & \text{if } D_{i,j} = -1 \\ C'_{i,j} - b, & \text{if } D_{i,j} = 0 \\ C'_{i,j}, & \text{otherwise.} \end{cases} \quad (9)$$

In Eq.(9), $b \in \{0, 1\}$ is a message bit to be embedded, and $C''_{i,j}$ is the corresponding pixel value in the marked image, $C'_{i,j}$ is the corresponding pixel value after modification, and it is calculated as Eq.(2). The embedding process is shown in FIGURE 3 (bits 0110). The difference between Eq.(4) and Eq.(9) is that the difference values modified are still in $\{-1, 0\}$ and the other difference values are unchanged, the process is shown in FIGURE 3. Because we don't know if there is an embedded bit in the pixel if we get a difference -1 or 0, so we need the $D_{i,j}$, $1 \le i \le N_I$, $2 \le j \le m_I \times n_I$ to extract secret messages and restore the host image.

After the embedding process, the message extraction and image restoration can be described as follows.

$$b^* = \begin{cases} 0, & \text{if } C''_{i,j} - C''_{i,1} \in \{0, -1\} \text{ and } C''_{i,j} - C''_{i,1} = D_{i,j} \\ 1, & \text{if } C''_{i,j} - C''_{i,1} \in \{0, -1\} \text{ and } C''_{i,j} - C''_{i,1} \neq D_{i,j}. \end{cases} \quad (10)$$

$${C'_{i,j}}^* = \begin{cases} C''_{i,j} - 1, & \text{if } C''_{i,j} - C''_{i,1} > D_{i,j} \\ C''_{i,j} + 1, & \text{if } C''_{i,j} - C''_{i,1} < D_{i,j} \\ C''_{i,j}, & \text{otherwise.} \end{cases} \quad (11)$$

To avoid the saturation (i.e., the overflow or underflow) during the embedding process, traditional DHS method has to modify the saturated pixels (pixels with value 0 or 255) as Eq.(2). In this algorithm, if $D_{i,j} = -1$, $C_{i,j}$ has a maximum of 254 and a minimum of 0, so after the process in Eq.(2), there will not exist overflow phenomenon; if $D_{i,j} = 0$, $C_{i,j}$ has a maximum of 255 and a minimum of 0, if $C_{i,j} = 0$, an underflow will occur. So, modifying $C_{i,j}$ to $C'_{i,j}$ follows the equation

$$C'_{i,j} = \begin{cases} 0, & \text{if } {C'_{i,j}}^* = 1 \text{ and } i, j \text{ is in } L_p \\ {C'_{i,j}}^*, & \text{otherwise.} \end{cases} \quad (12)$$

Comparing with DHS, this algorithm successfully implements the embedding and restoring of secret messages, and we use the original difference value as the key to extract the secret messages. Obviously, this algorithm modifies fewer bits, and better maintains the statistical features of the difference value histogram. The experimental result of this part is shown in Section IV.

C. SFM_B:EMBEDDING SECRET MESSAGES INTO PIXELS WITH DIFFERENCE VALUES -2, -1, 0 AND 1

In order to further increase the embedding capacity, we propose this algorithm that embeds secret messages in pixels with four different difference values, and the principle is also maintaining the statistical features of the difference value histogram. Before embedding, we put two consecutive bits as an embedded unit. The embedding algorithm of the RDH scheme can be described as follows.

$C'_{i,j}$ is the corresponding pixel value of $C_{i,j}$ after modification, and it is calculated as Eq.(2). The operation of the pixels with difference values $-2, -1, 0$ and 1 is shown as TABLE 1, and after the operation we will get the corresponding pixel value in the marked image $C''_{i,j}$. The embedding process is shown in FIGURE 4 (bits 01101100). And we can easily calculate that the different values $D'_{i,j}$ after the embedding process, as shown in TABLE 2.

Obviously, the same difference corresponds to the same embedding unit, so we can get the message extraction as

TABLE 1. Operation of the pixels in embedding process. The row represents the difference value $D_{i,j}$, and the column represents the embedding unit.

	-2	-1	0	1
00	Unchanged	-1	-2	-3
01	+1	Unchanged	-1	-2
10	+2	+1	Unchanged	-1
11	+3	+2	+1	Unchanged

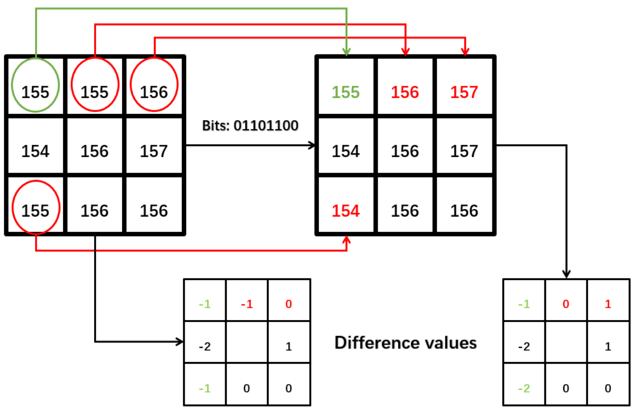


FIGURE 4. Bits 01101100 embed in one sub-block (3 × 3) applying SFM_B.

TABLE 2. New difference values $D'_{i,j}$ after embedding process. The row represents the difference value, and the column represents the embedding unit.

	-2	-1	0	1
00	-2	-2	-2	-2
01	-1	-1	-1	-1
10	0	0	0	0
11	1	1	1	1

TABLE 3. Operation of the pixels in image restoration process. The row represents the difference value $D_{i,j}$, and the column represents new difference values $D'_{i,j}$ after embedding process.

	-2	-1	0	1
-2	Unchanged	+1	+2	+3
-1	-1	Unchanged	+1	+2
0	-2	-1	Unchanged	+1
1	-3	-2	-1	Unchanged

Eq.(13).

$$b^* = \begin{cases} 00, & \text{if } C''_{i,j} - C''_{i,1} = -2 \\ 01, & \text{if } C''_{i,j} - C''_{i,1} = -1 \\ 10, & \text{if } C''_{i,j} - C''_{i,1} = 0 \\ 11, & \text{if } C''_{i,j} - C''_{i,1} = 1. \end{cases} \quad (13)$$

The image restoration process could be completed by TABLE 3, it is the reverse process of the embedded operation, we will get ${C'_{i,j}}^*$ after the process.

To avoid the saturation (i.e., the overflow or underflow) during the embedding process, we should also modify the saturated pixels by modifying $C_{i,j}$ to $C'_{i,j}$ follows Eq.(14). To recover the host image, we must record the position (i, j) of modified pixel $C_{i,j}$ and the modified value in two lists: L_p and L_s, and these two lists have the same length.

$$C'_{i,j} = \begin{cases} 252, & \text{if } D_{i,j} = -2 \text{ and } C_{i,j} > 252 (\text{only 253 in fact}) \\ 1, & \text{if } D_{i,j} = -1 \text{ and } C_{i,j} < 1 (\text{only 0 in fact}) \\ 253, & \text{if } D_{i,j} = -1 \text{ and } C_{i,j} > 253 (\text{only 254 in fact}) \\ 2, & \text{if } D_{i,j} = 0 \text{ and } C_{i,j} < 2 (0 \text{ or } 1) \\ 254, & \text{if } D_{i,j} = 0 \text{ and } C_{i,j} > 254 (\text{only 255 in fact}) \\ 3, & \text{if } D_{i,j} = 1 \text{ and } C_{i,j} < 3 (1 \text{ or } 2). \end{cases} \quad (14)$$

And the host image could be recovered as follows:

for t from the beginning to the end of L_p :

$$C^*_{i,j} = \begin{cases} {C'_{i,j}}^* - L_s(t), & \text{if } {C'_{i,j}}^* \leq 3 \text{ and } i, j \text{ is in } L_p \\ {C'_{i,j}}^* + L_s(t), & \text{if } {C'_{i,j}}^* \geq 252 \text{ and } i, j \text{ is in } L_p. \end{cases} \quad (15)$$

In Eq.(15), 't' is an index of the list L_p, the equation means for every element in the list, we will find the corresponding modified pixel and restore it following list L_s.

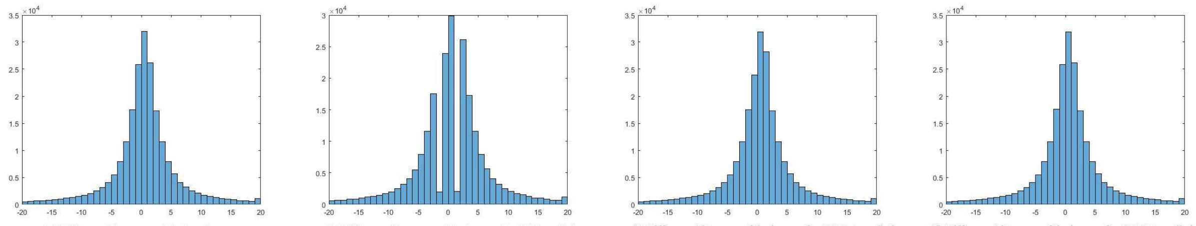
In this algorithm, because that we regard two consecutive bits as an embedded unit, the embedding capacity will be greatly improved. Some pixels are unchanged before and after the operation, that could improve the embedding efficiency (calculated by Number of embedded bits/ Number of modified bits), the difference values are still in $\{-2, -1, 0, 1\}$, so the statistical features of the difference value histogram are well maintained. The experimental result of this part is shown in Section IV.

IV. ANALYSIS AND EXPERIMENT

The experiment consists of five parts: comparative experiment of the difference histogram, comparative experiment of the PSNR, comparative experiment of the SSIM and comparative experiment of the capacity-distortion trade-off. In each part, we compare some state-of-the-arts algorithms with the two algorithms we proposed in Section III. Here in order to be convenient, the algorithm of embedding secret messages into pixels with difference value 0 and -1 is called SFM_A, and the algorithm of embedding secret messages into pixels with difference values -2, -1, 0 and 1 is called SFM_B as described in Section III.

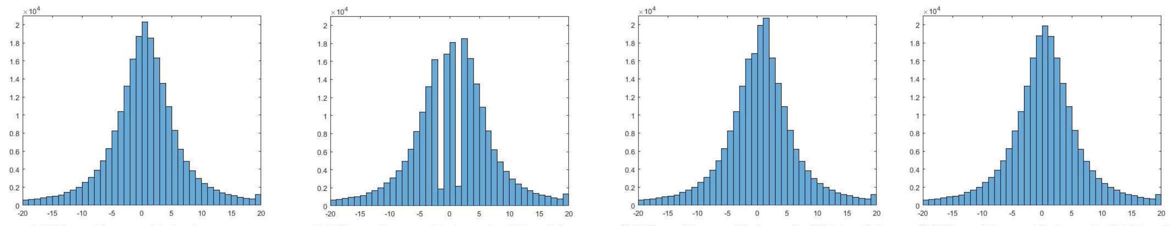
A. PRE-PROCESS FOR SATURATION PREVENTION

To avoid the saturation (i.e., the overflow or underflow) during the embedding process, in SFM_A, if $D_{i,j} = -1$, $C_{i,j}$ has a maximum of 254 and a minimum of 0, so after the process in Eq.(2), there will not exist overflow phenomenon; if $D_{i,j} = 0$, $C_{i,j}$ has a maximum of 255 and a minimum of 0, if $C_{i,j} = 0$, an underflow will occur. So, modifying $C_{i,j}$ to $C'_{i,j}$ follows Eq.(2). And in SFM_B, we should also modify the saturated pixels by modifying $C_{i,j}$ to $C'_{i,j}$ follows Eq.(14).



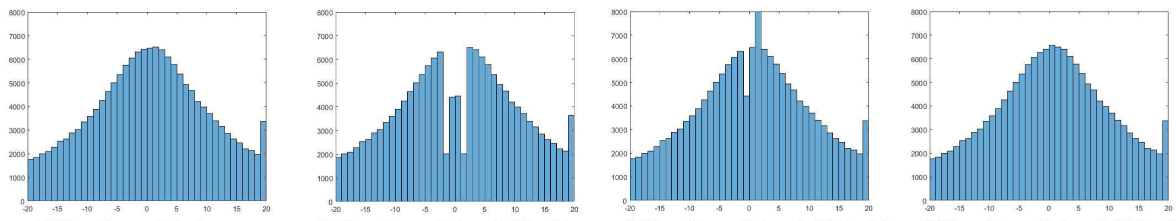
(a) Difference histogram of the host image. (b) Difference histogram of the image after DHS applied. (c) Difference histogram of the image after SFM_A applied. (d) Difference histogram of the image after SFM_B applied.

FIGURE 5. **Histogram of the difference value of "Airplane".**



(a) Difference histogram of the host image. (b) Difference histogram of the image after DHS applied. (c) Difference histogram of the image after SFM_A applied. (d) Difference histogram of the image after SFM_B applied.

FIGURE 6. **Histogram of the difference value of "Lena".**



(a) Difference histogram of the host image. (b) Difference histogram of the image after DHS applied. (c) Difference histogram of the image after SFM_A applied. (d) Difference histogram of the image after SFM_B applied.

FIGURE 7. **Histogram of the difference value of "Baboon".**

B. EXPERIMENT ON THE HISTOGRAM OF THE DIFFERENCE VALUE

The main idea of our method is statistical features maintaining, so we do experiment on the histogram of the difference value, compare the histogram of the image after the operation of DHS algorithm, SFM_A and SFM_B getting applied with the histogram of host image. In the experiment, a piece of secret message with 4000 bits length is generated, we use the gray-scale image "Airplane"(512×512) which is shown in FIGURE 8.(a) as the carrier and the result of this part is shown as follows:

Among figures illustrated as FIGURE 5, FIGURE 6 and FIGURE 7. They all display similar statistical features. We employ FIGURE 5 for explaining. From FIGURE 5.(b), it is not hard to observe that the parts of the histogram corresponding to 0 and -1 are dispersed, the Bins corresponding to less than -1 are shifted to the left and larger than 0 are shifted to the right. The result in FIGURE 5.(c) and FIGURE 5.(d) shows that the two algorithms applying our method better maintained the statistical features of the differential histogram, and in terms of the difference histogram, SFM_B does a better job than SFM_A.

As Eq.(9) shows, SFM_A changes the difference value when the bit to be embedded is '1', under these circumstances, with the number of '1's in the embedded bit increasing, there will be more difference going from '0' to '-1' or from '-1' to '0'. Now, we assume that the difference values '0' and '-1' are completely randomly distributed in pixels, this means that when we embed bit '1' in the image, the probability of changing the difference value '0' or '-1' depends on the number of difference values '0' and '-1'. Now assume a situation, the number of difference values '0' in the image is 1000, and the number of '-1' is 500, we embed 900 bits '1' into the image. Based on the previous assumptions, we can easily calculate that there are 600 '0's to '-1' and 300 '-1's to '0', now the number of difference value '0' is 700 ($1000-600+300 = 700$), and the number of difference value '1' is 800 ($500-300+600 = 800$), in other words, the number of difference values '0' and '-1' may be reversed. As FIGURE 5 shows, compared to the number in the original histogram, the number of '0' is much more than '-1', the number of difference values '0' and '-1' is significantly closer to be equal than the original image. Compared with SFM_A, the difference values of SFM_B are more varied, and the histogram of difference value is closer to the histogram of the original image as FIGURE 5.

C. EXPERIMENT ON THE PSNR

Here we choose 8 different images with size of 512×512 from reference [14] as the carriers, which are shown in FIGURE 8,

FIGURE 8. The test images.

TABLE 4. PSNR of the marked images with 8000 bits message embedded.(I = image, S = scheme).

S \ I	DHS			SFM_A			SFM_B			Ou's [14]		
	1×2	2×2	3×3	1×2	2×2	3×3	1 ×2	2×2	3×3	Pro-FM	Pro-AMG	Pro-AMO
Airplane	52.32	50.56	49.83	66.28	66.28	66.28	62.33	62.31	62.37	64.93	65.23	66.15
Baboon	51.29	49.54	48.85	66.28	66.28	66.28	62.25	62.16	62.25	56.92	57.22	56.99
Barbara	51.58	49.93	49.21	66.28	66.28	66.28	62.41	62.45	62.49	60.89	61.53	61.50
Boat	51.51	49.81	49.11	66.28	66.28	66.28	62.41	62.45	62.49	59.19	59.45	59.43
Elaine	51.54	49.80	49.10	66.28	66.28	66.28	62.83	62.88	62.98	60.12	60.15	60.43
Lake	51.53	49.83	49.14	66.28	66.28	66.28	62.32	62.43	62.39	60.34	60.59	60.65
Lena	51.77	50.10	49.39	66.28	66.28	66.28	62.47	62.56	62.56	61.06	61.56	61.79
Peppers	51.61	49.92	49.24	66.28	66.28	66.28	62.45	62.45	62.43	57.81	58.19	56.52

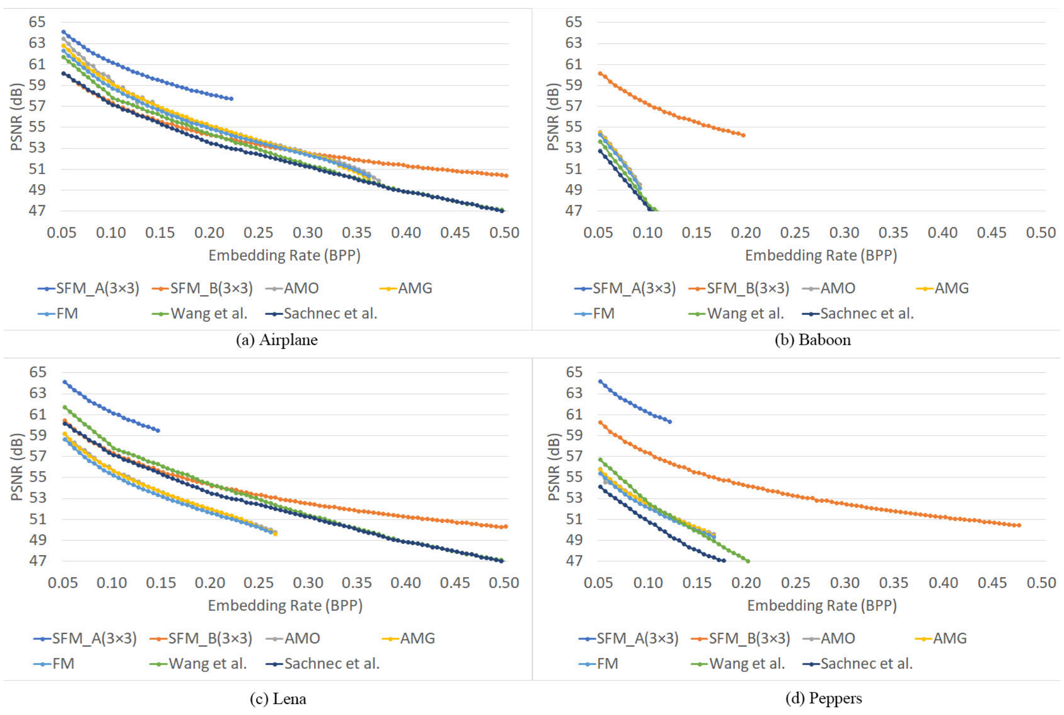
secret message with length 8000 is generated, three kinds of sub-block dividing methods shown in FIGURE 1 are experimented respectively, the PSNR of the images after DHS, SFM_A, SFM_B and three methods proposed in [14] (It was developed based on PEE rather than DHS, so we compare the three methods proposed in this scheme to our proposed algorithm without sub-block division) getting applied are shown in TABLE 4.

For SFM_A, after the pixel modification of embedding process, the pixel matrix of the embedded image need to make difference with that of the original image when computing the PSNR, the resulted matrix is a binary matrix with 0 and 1. With the same secret message to be embedded, the number of elements equal to 1 in the resulted matrix is the same for all kinds of carrier images. This is why the PSNR of the embedded images are equal.

Compare the results in TABLE 4, we can see that both of the two algorithms we proposed get better performance in the quality of the marked image than DHS algorithm. Compared to some of Ou's most advanced algorithms [14], when the embedded secret bits are the same, SFM_A gets better effects better in all images and SFM_B also performs better in most images. And note that the SFM_A gets a larger PSNR value than SFM_B, that means, embedding secret bits into four different difference values will introduce greater flaws than two difference values. While an obvious advantage of SFM_B is that the embedding capacity has been greatly improved.

D. EXPERIMENT ON THE SSIM

To illuminate the structural change between the original images and the embedded ones, we compare the SSIM of our proposed two schemes (in three different ways of blocking: 1×2, 2×2, 3×3) to Ou's [14] in TABLE 5. To show more clear contrast, we adopt two embedding rates as adopted in [14], namely 0.1 and 0.2. As in TABLE 5, our schemes provide a high SSIM value, which means the image structures are maintained well after the message embedding. When the embedding rate was set to 0.1, the SSIM of SFM_A reached 0.9999 for almost all images that can achieve this embedding capacity (except the case when Airplane is blocked



(a) Airplane (b) Baboon (c) Lena (d) Peppers

FIGURE 9. Performance comparisons of the proposed methods with some states-of-the-art schemes.

as 3 × 3, the SSIM value is 0.9998); What's more, for the image Baboon, all of Ou's three schemes cannot complete the embedding, while the SSIM of our SFM_B still reached up to 0.9999. When the embedding rate was set to 0.2, our schemes had similar SSIM with Ou's schemes, but the number of images which can be embedded with our schemes are more than Ou's. The reasons are the following. On one hand, in SFM_A, we split the images into different block sizes, and in each block, the pixels that meet the embedding criteria are picked strictly (the difference between embedding pixels and reference pixel is restricted to only 0 and -1), and after embedding, the maximum modification for the embedded pixel is 1. Therefore, the modification to a image is small. While this rule may cause a lower embedding rate to SFM_A. On the other hand, SFM_B, whose embedding criteria is also strict ($-2, -1, 0, 1$) but more relaxed than that of SFM_A, can provide higher embedding rate. SFM_B still has a high SSIM because the block mechanism makes the pixel change small before and after embedding. In addition, similar to SFM_A, the modification of a pixel for embedding is also maximum to only 1, so the SSIM of SFM_B is slightly lower than that of SFM_A after embedding but the embedding capacity is improved greatly.

E. EXPERIMENT ON THE CAPACITY-DISTORTION TRADE-OFF

We compare Pro-AMO, Pro-AMG and Pro-FM which are proposed in Ou's [14] and another scheme [26] listed in TABLE 5 to our two schemes in terms of PSNR. The PSNR of different schemes with embedding rate ranging from 0.05 to 0.5 can be found in FIGURE 9, where SFM_A and SFM_B are applied by splitting images into 3 × 3 blocks. According to Sectoon III, 3 × 3 blocking has a higher embedding rate but a lower PSNR in our proposed scheme. We apply the above seven schemes on four images which were tested and proved to be available for embedding more message in. As shown in FIGURE 9, in the four testing images *Airplane*, *Baboon*, *Lena* and *Peppers*, SFM_A all has the largest PSNR at the same embedding rate. But the cost of high PSNR is that the range of its embedding rate is limited to a relatively lower level. In the image *Airplane*, SFM_A has the highest embedding rate than the left three images, which can reach 0.22. In this image, SFM_B has a poor PSNR in lower embedding rate, but with the increasing of the embedding rate, the line of SFM_B descends gradually. When the embedding rate increases to 0.3, SFM_B becomes the best one. [26]'s scheme also has a gently descending line, but its overall performance is inferior to SFM_B's. SFM_B also performs very well in image *Baboon*, its PSNR reaches around 60 with the embedding rate 0.05, and when embedding rate increases to 0.2, the PSNR can still reach up to 54. However, the PSNRs of other schemes are only less than 55 when the embedding rate comes to 0.05, and fall rapidly below 47 when the embedding rate increases to 0.1. When it comes to the image *Lena*, the overall performance of each scheme is similar to that in image *Airplane*. The difference is SFM_B achieves relatively better performance in the initial low embedding rate, and with the increasing of the embedding rate, the PSNR of SFM_B exceeds other schemes' and becomes the best scheme from the embedding rate 0.27 because of its gentle descent.

TABLE 5. Performance comparison in terms of SSIM. The symbol "–" denotes that the corresponding method cannot fulfill the capacity.

Schemes	Capacity	Images					
	(BPP)	Lena	Baboon	Airplane	Barbara	Lake	Boat
SFM_A (1×2)	0.1	-	-	0.9999	-	-	-
	0.2	-	-	-	-	-	-
SFM_A (2×2)	0.1	0.9999	-	0.9999	0.9999	-	-
	0.2	-	-	-	-	-	-
SFM_A (3×3)	0.1	0.9999	-	0.9999	0.9999	0.9999	-
	0.2	-	-	0.9998	-	-	-
SFM_B (1×2)	0.1	0.9997	0.9999	0.9997	0.9998	0.9997	0.9998
	0.2	0.9995	-	0.9993	0.9995	0.9995	0.9996
SFM_B (2×2)	0.1	0.9997	0.9999	0.9997	0.9998	0.9997	0.9997
	0.2	0.9994	-	0.9992	0.9996	0.9994	0.9996
SFM_B (3×3)	0.1	0.9997	0.9999	0.9997	0.9998	0.9997	0.9997
	0.2	0.9994	-	0.9993	0.9996	0.9994	0.9995
Pro-AMG	0.1	0.9997	-	0.9998	0.9998	0.9997	0.9997
	0.2	0.9996	-	0.9996	0.9996	-	-
Pro-AMO	0.1	0.9997	-	0.9998	0.9998	0.9997	0.9997
	0.2	0.9996	-	0.9996	0.9997	-	-
Pro-FM	0.1	0.9997	-	0.9998	0.9998	0.9997	0.9997
	0.2	0.9995	-	0.9996	0.9996	-	-
[13]	0.1	0.9997	-	0.9998	0.9998	0.9997	0.9997
	0.2	0.9995	-	0.9996	-	-	-
[26]	0.1	0.9997	0.9995	0.9997	0.9997	0.9997	0.9996
	0.2	0.9994	0.9983	0.9995	0.9994	0.9990	0.9991
[27]	0.1	0.9999	0.9996	0.9999	0.9999	0.9999	0.9999
	0.2	0.9998	0.9996	0.9999	0.9998	0.9996	0.9997
[28]	0.1	0.9994	-	0.9997	0.9996	0.9997	0.9995
	0.2	0.9991	-	0.9995	0.9994	-	-

The line of each scheme in Pepper is similar to those in *Baboon*, but the decline is more gradual. SFM_B always keeps the highest PSNR, when the embedding rate increases to 0.2, the PSNRs of other schemes fall below 47, but the PSNR of SFM_B can reach up to 54, and keeps going down gently. When the embedding rate reaches up to 0.47, SFM_B can still gain PSNR more than 50. The reason of the above phenomenon is given as follows. For small data payload (ER $\leq$ 0.4 bits per pixel), the three methods of [14] and the method of [26] outperform similar with each other, while in *Baboon* and *Peppers*, our scheme outperforms consistently the others. For large data payload (ER > 0.4 bits per pixel), our scheme tends to be similar to [26] in shape but has a higher PSNR value in the same embedding rate. Because the method in [26] and methods in [14] are all implemented on the errors of rhombus prediction. At high ER, the important bins with high frequency in the histogram should be utilized by all the involved schemes to hide large payload. It is also observed that [26] and [14] cannot compete for Pepper image at high data payload. This is because the *Pepper* image has many potentially overflowed/underflowed pixels located around both sides of the histogram, i.e., 0 and 255.

V. CONCLUSION

In this paper, we propose an improved RDH method based on DHS named SFM, the main idea is keeping the difference values that need to be modified in the original range, and the other difference values will not be shifted, this will help maintain the statistical features of the image and improve the security. We also propose two algorithms SFM_A and SFM_B based on the method and provide a concrete implementation to overflow/underflow prevention, embedding and extraction, SFM_A embeds secret messages into pixels with difference values 0 and -1, while the SFM_B into -2, -1, 0 and 1, the experiments show that, both the two algorithms better maintain the statistical features of the differential histogram; as for the quality of the embedded images, SFM_A got a better performance than some state-of-the-arts, namely got a larger PSNR value, SFM_B also got a larger PSNR value in some experimental sample images with higher capacity. SFM_A is the best choice if it can satisfy the embedding capacity, when length of message to be embed in the image is beyond the capacity of SFM_A, applying SFM_B to this image could reach a good performance. There are some interesting directions for future research. Firstly, note that both the SFM_A and SFM_B only have distinct advantages for specific requirements, it is worthwhile to explore a scheme that could meet more needs for embedding messages. Secondly, with the development of deep learning, it has been applied to many fields and achieved remarkable results. In the next step, we will try to apply the method of deep learning to steganography in order to obtain better results.

REFERENCES

[1] J. M. Barton, "Method and apparatus for embedding authentication information within digital data," U.S. Patent 5 646 997, Jul. 8, 1997.

[2] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.

[3] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.

[4] A. Khan, A. Siddiqa, S. Munib, and S. A. Malik, "A recent survey of reversible watermarking techniques," *Inf. Sci.*, vol. 279, pp. 251–272, Sep. 2014.

[5] H.-T. Wu, J.-L. Dugelay, and Y.-Q. Shi, "Reversible image data hiding with contrast enhancement," *IEEE Signal Process. Lett.*, vol. 22, no. 1, pp. 81–85, Jan. 2015.

[6] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 9, pp. 2016–2027, Sep. 2015.

[7] Y. Qiu, Z. Qian, and L. Yu, "Adaptive reversible data hiding by extending the generalized integer transformation," *IEEE Signal Process. Lett.*, vol. 23, no. 1, pp. 130–134, Jan. 2016.

[8] R. Kumar and S. Chand, "A reversible high capacity data hiding scheme using pixel value adjusting feature," *Multimedia Tools Appl.*, vol. 75, no. 1, pp. 241–259, Jan. 2016.

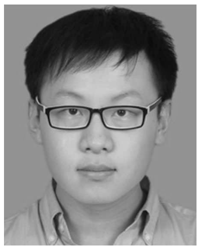
[9] Y.-H. Chen, H.-C. Huang, and C.-C. Lin, "Block-based reversible data hiding with multi-round estimation and difference alteration," *Multimedia Tools Appl.*, vol. 75, no. 21, pp. 13679–13704, Nov. 2016.

[10] T. Luo, G. Jiang, M. Yu, H. Xu, and W. Gao, "Sparse recovery based reversible data hiding method using the human visual system," *Multimedia Tools Appl.*, vol. 77, no. 15, pp. 19027–19050, Aug. 2018.

[11] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Process.*, vol. 93, pp. 198–205, 2013.

[12] C.-F. Lee, J.-J. Shen, Y.-C. Kao, and S. Agrawal, "Overlapping pixel value ordering predictor for high-capacity reversible data hiding," *J. Real-Time Image Process.*, vol. 16, no. 4, pp. 835–855, Aug. 2019.
[13] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Trans. Image Process.*, vol. 22, no. 12, pp. 5010–5021, Dec. 2018.
[14] B. Ou, X. Li, W. Zhang, and Y. Zhao, "Improving pairwise PEE via hybrid-dimensional histogram generation and adaptive mapping selection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 7, pp. 2176–2190, Jul. 2019.
[15] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Process.*, vol. 94, pp. 118–127, Jan. 2014.
[16] M. Li, D. Xiao, and Y. Zhang, "Reversible data hiding in block compressed sensing images," *ETRI J.*, vol. 38, no. 1, pp. 159–163, Feb. 2016.
[17] Z. Qian and X. Zhang, "Reversible data hiding in encrypted images with distributed source encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 4, pp. 636–646, Apr. 2016.
[18] F. Huang, J. Huang, and Y.-Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2777–2789, Dec. 2016.
[19] A. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147–1156, Aug. 2004.
[20] Z. Yin, B. Luo, and W. Hong, "Separable and error-free reversible data hiding in encrypted image with high payload," *Sci. World J.*, vol. 2014, pp. 1–8, Apr. 2014.
[21] L. Kamstra and H. Heijmans, "Reversible data embedding into images using wavelet techniques and sorting," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2082–2090, Dec. 2005.
[22] S.-K. Lee, Y.-H. Suh, and Y.-S. Ho, "Reversiblee image authentication based on watermarking," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2006, pp. 1321–1324.
[23] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3524–3533, Dec. 2011.
[24] X. Xiong, "Novel scheme of reversible watermarking with a complementary embedding strategy," *IEEE Access*, vol. 7, pp. 136592–136603, 2019.
[25] X. Duan, K. Jia, B. Li, D. Guo, E. Zhang, and C. Qin, "Reversible image steganography scheme based on a U-Net structure," *IEEE Access*, vol. 7, pp. 9314–9323, 2019.
[26] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 989–999, Jul. 2009.
[27] J. Wang, J. Ni, X. Zhang, and Y.-Q. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 315–326, Feb. 2017.
[28] I.-C. Dragoi and D. Coltuc, "Adaptive pairing reversible watermarking," *IEEE Trans. Image Process.*, vol. 25, no. 5, pp. 2420–2422, May 2016.

TENGFEI LI was born in 1990. He received the B.S. degree from the College of Software Engineering, Jilin University (JLU), where he is currently pursuing the Ph.D. degree from the College of Computer Science and Technology. His current research interests include network security, cryptography, data mining, image processing and data hiding, machine learning, and NLP.



HUIFENG LI was born in July 1993. He received the B.S. degree from the College of Computer Science and Technology, Jilin University, where he is currently pursuing the master's degree. His research interests are privacy and information security.

LIANG HU was born in 1968. He received the B.S. degree from the Harbin Institute of Technology (HIT), Harbin, and the M.S. and Ph.D. degrees from the College of Computer Science and Technology, Jilin University (JLU), Changchun, China. He has been a Professor, since 2002 and has been a Ph.D. supervisor, since 2003 with the School of Jilin University. His current research interests include distributed computing, network computing and security, data security and privacy, and so on.

HONGTU LI was born in 1984. He received the Ph.D. degree from the College of Computer Science and Technology, Jilin University (JLU), Changchun, China. He has been a Lecturer with the School of Jilin University, since 2012. His current research interests include cryptography, data integrity protection, wireless network security, and so on.