IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 20, 2019, accepted January 2, 2020, date of publication January 6, 2020, date of current version January 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964264

Color Image Reversible Data Hiding With Double-Layer Embedding

ZHENJUN TANG[1], (Member, IEEE), HEWANG NIE[1], CHI-MAN PUN[2], (Senior Member, IEEE), HENG YAO[3], (Member, IEEE), CHUNQIANG YU[1], AND XIANQUAN ZHANG[1]

[1]Guangxi Key Laboratory of Multi-Source Information Mining and Security, Department of Computer Science, Guangxi Normal University, Guilin 541004, China
[2]Department of Computer and Information Science, University of Macau, Macau 999078, China
[3]School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Corresponding author: Zhenjun Tang (tangzj230@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61562007, Grant 61962008, Grant 61762017, and Grant 61702332, in part by the Guangxi "Bagui Scholar" Team for Innovation and Research, in part by the Project of Guangxi Science and Technology under Grant GuiKeAD17195062, in part by the Guangxi 1000-Plan of Training Middle-aged/Young Teachers in Higher Education Institutions, and in part by the Guangxi Collaborative Innovation Center of Multi-source Information Integration and Intelligent Processing.

ABSTRACT Reversible data hiding (RDH) in color image is an important topic of data hiding. This paper presents an efficient RDH algorithm for color image via double-layer embedding. The key contribution is the proposed double-layer embedding technique based on histogram shifting (HS). This technique exploits image interpolation to generate prediction error matrices for HS in the first-layer embedding and uses local pixel similarity to calculate difference matrices for HS in the second-layer embedding. It inherits reversibility from HS and makes high embedding capacity due to the use of double layers in data embedding. In addition, inter-channel correlation is incorporated into the first-layer embedding and the second-layer embedding for generating histograms with high peaks, so as to improve embedding capacity. Experiments with open standard datasets are done to validate performance of the proposed RDH algorithm. Comparison results show that the proposed RDH algorithm outperforms some state-of-the-art RDH algorithms in terms of embedding capacity and visual quality.

INDEX TERMS Reversible data hiding, histogram shifting, double-layer embedding, image interpolation, inter-channel correlation.

I. INTRODUCTION

Secure transmission of private information via public channels [1], [2] has attracted much attention in the past decade. Reversible data hiding (RDH) [3]–[6] is an efficient technology of information security, and has been used in many fields, such as military communication, medical application and judicial imaging. In general, at the sender side, RDH embeds secret data into a cover media, such as image, audio or video, in an imperceptible manner. At the receiver side, it not only can exactly extract the embedded data, but also completely recovers original content of the cover media.

In recent years, researchers have exploited many techniques to develop useful RDH algorithms. According to the used embedding strategy, these RDH algorithms are mainly based on the following techniques, such as lossless compression (LC) [7]–[11], difference expansion (DE) [12]–[14], histogram shifting (HS) [15]–[18], prediction-error expansion (PEE) [19]–[23] and integer transform (IT) [24]–[27]. Some typical algorithms of each category are briefly introduced as follows.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiju Poovvancheri.

A. LC-BASED RDH ALGORITHMS

An early work using LC is proposed by Fridrich *et al.* [7]. This method compresses redundant information of the data without causing any distortion during image recovery. But it cannot reach high capacity and low distortion performance due to weak correlation in bitplane. In [8], Zhang *et al.* exploited a general decompression algorithm to generalize the recursive code construction for data embedding. In another work, Zhang *et al.* [9] used the decompression and compression processes of an entropy coder to embed data. Both the techniques [8], [9] can reach rate–distortion bound. For vector quantization (VQ) compressed images, Qin *et al.* [10]

exploited index mapping mechanism to achieve RDH. Recently, Tang *et al.* [11] designed a novel RDH algorithm for encrypted image. This algorithm uses a technique called differential compression to vacate room for data embedding and reaches high capacity.

B. DE-BASED RDH ALGORITHMS

A well-known DE-based algorithm is given by Tian [12]. In general, classical DE is used to embed one bit by expanding difference between two pixels. DE has a large embedding capacity and a low distortion rate, but it requires a location map for overcoming overflow problems when processing those pixels beyond the gray scale range after expansion. In [13], Alattar designed a high-capacity reversible watermarking algorithm for color images. This algorithm can embed several secret bits by exploiting vector DE of adjacent pixels. In another work, Kamstra and Heijmans [14] developed an improved algorithm of Tian's DE technique [12] by incorporating LC with an adaptive arithmetic coder. This algorithm outperforms Tian's algorithm in embedding capacity.

C. HS-BASED RDH ALGORITHMS

Ni *et al.* [15] gave the first work of HS-based RDH. This RDH algorithm uses the zero or minimum point of image histogram and modifies pixel values slightly to conduct data embedding. In [16], Li *et al.* jointly used two-dimensional difference HS and difference-pair-mapping to design a novel RDH algorithm. This algorithm can exploit image redundancy well and reach good embedding performance. In another work, Qin *et al.* [17] proposed a novel HS-based RDH algorithm based on prediction and image inpainting. Experiments have shown that this algorithm is better than some reported algorithms in embedding rate and visual quality. Recently, Tang *et al.* [18] applied HS to the design of RDH in encrypted image, and achieved data embedding by shifting block histogram of pixel differences in homomorphic encrypted domain. Ying *et al.* [19] proposed a novel RDH algorithm based on contrast enhancement. This algorithm used the HS technique based on prediction error to achieve high embedding payload.

D. PEE-BASED RDH ALGORITHMS

Thodi and Rodriguez [20] first proposed to use PEE to embed data. Compared with DE technique, PEE can well exploit correlation inherent in the neighborhood of a pixel and then doubles the maximum embedding capacity. Hong *et al.* [21] modified prediction errors of histogram to vacate room for data embedding. This scheme can produce stego image with good visual quality in terms of PSNR (Peak Signal to Noise Ratio). In [22], Li *et al.* proposed an adaptive PEE-based RDH algorithm by incorporating pixel selection and adaptive embedding. This algorithm is better than conventional PEE in capacity. To exploit correlations among prediction-errors, Ou *et al.* [23] proposed a pairwise PEE based RDH scheme, which can jointly consider every two adjacent prediction-errors. Compared with the conventional PEE, this scheme can also reach an improved performance. In another work, Ou *et al.* [24] extended the work [23] to make a high-fidelity RDH by using geodesic path. This algorithm is better the conventional pairwise PEE [23] in visual quality, but it is slower than [23].

E. IT-BASED RDH ALGORITHMS

Coltuc and Chassery [25] exploited an IT technique called reversible contrast mapping to construct a fast RDH algorithm. This algorithm has low complexity and is robust against cropping. Weng *et al.* [26] jointly used IT and PDA (pairwise difference adjustment) to design a novel RDH scheme for watermarking. This scheme is better than the algorithm [25] in embedding capacity. In another work, Wang *et al.* [27] exploited a generalized IT and a payload-dependent location map to extend the DE scheme given by Tian [12]. This method shows better embedding rate than the DE scheme [12]. Peng *et al.* [28] presented a new RDH algorithm using the techniques of IT and adaptive embedding. Since this RDH algorithm can adaptively embed data into blocks by tuning capacity parameter in IT, it can reach high capacity with good image quality.

Most of the above RDH algorithms are designed for grayscale images. Recently, some researchers have presented several useful color image RDH algorithms. For example, Yang *et al.* [29] utilized prediction errors to design an RDH algorithm for CFA (color filter array) mosaic images, in which spectral–spatial correlation in the color difference domain is exploited to achieve small prediction errors. Li *et al.* [30] proposed a color image RDH algorithm by using PEE and cross-channel correlation. In another work, Ou *et al.* [31] exploited inter-channel correlation and adaptive embedding to construct an efficient color image RDH algorithm. Recently, Yao *et al.* [32] used a guided filtering predictor and an adaptive PEE scheme to make a color image RDH algorithm, where the inter-channel correlation is exploited in the PEE scheme. Wang and Pan [33] exploited best neighboring coding to construct an improved RDH algorithm for compressed color images based on vector quantization. Xu and Li [34] used 3D PEE mapping to improve embedding performance. This RDH algorithm outperforms conventional PEE-based algorithm. In another study, Hou *et al.* [35] introduced a novel color image RDH algorithm with grayscale invariance. This algorithm can embed data into color image without changing gray version of the color host image and thus the further use of the marked image is not affected.

Although researchers have proposed some useful RDH algorithms, there are still some limitations, especially in making high embedding capacity for color images. In this paper, we present a novel efficient color image RDH algorithm with double-layer embedding. Compared with the existing techniques, our RDH algorithm has the below contributions.

(1) We propose a double-layer embedding technique based on HS. In the first-layer embedding, this technique exploits

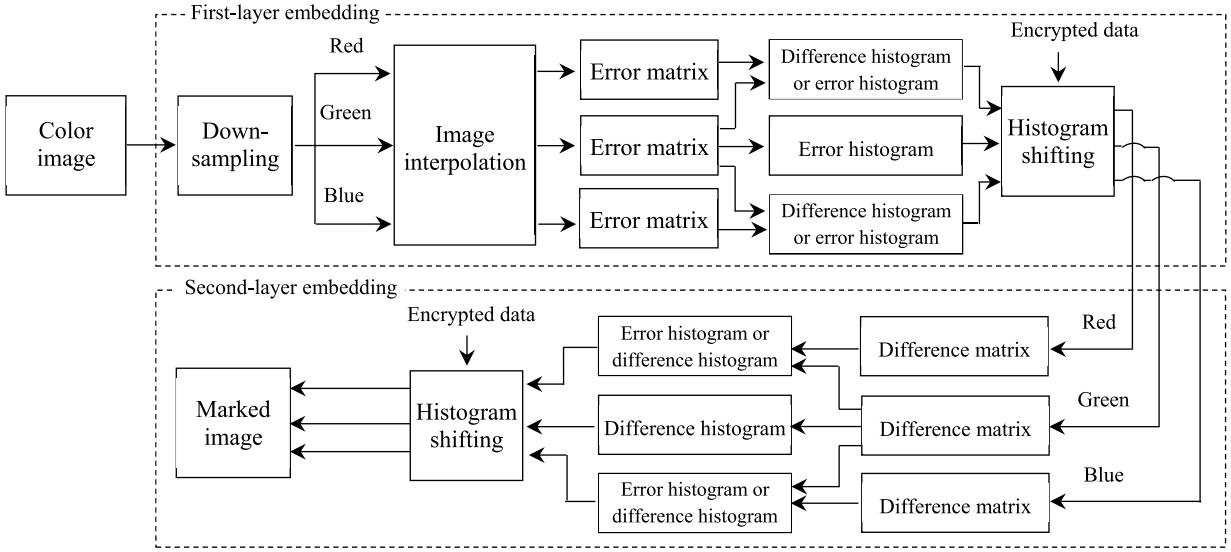


FIGURE 1. Framework of our RDH algorithm.

image interpolation to predict error matrices of the red, green and blue channels, which are used to construct histograms for HS. In the second-layer embedding, this technique uses local pixel similarity to calculate difference matrices, which are also used to construct histograms for HS. The use of double-layer embedding technique can ensure reversibility of our RDH algorithm and improve our embedding capacity.

(2) We incorporate inter-channel correlation into the first-layer embedding and the second-layer embedding. The inter-channel correlation is exploited to calculate errors/differences between corresponding elements of color channels. The use of inter-channel correlation can generate error/difference histograms with high peaks, which can provide high embedding capacity of our RDH algorithm.

Extensive experiments on the benchmark image databases are carried out and the results show that our RDH algorithm has better performance than some state-of-the-art RDH algorithms in terms of embedding capacity and visual quality. The rest of the paper is organized as follows. Section II describes our RDH algorithm. Section III presents experimental results and Section IV discusses performance comparison. Finally, Section V summarizes this paper.

II. OUR RDH ALGORITHM

Our RDH algorithm is a two-stage scheme, including the stages of the first-layer embedding and the second-layer embedding. Figure 1 presents the framework of our RDH algorithm. In the stage of the first-layer embedding, color image is firstly down-sampled and then the red, green and blue channels are resized to the original size of the cover image by image interpolation. Next, errors matrices between the red, green and blue channels of cover image and their corresponding channels of interpolated image are calculated. These errors matrices are used to construct error/difference histograms for embedding data with HS by using inter-channel correlation. In the stage of the second-layer embedding, difference matrices of the red, green and blue channels are firstly calculated by using local pixel similarity. These difference matrices are also used to construct error/difference histograms for embedding data with HS by using inter-channel correlation. Marked image is finally obtained by consolidating the red, green and blue channels. Section II A presents the down-sampling and image interpolation, and Section II B explains histogram shifting. Sections II C and II D introduce the first-layer embedding and the second-layer embedding, respectively. Section II E describes data extraction and image recovery.

1	2
3	4

FIGURE 2. Pixel order in a 2 × 2 block.

A. DOWN-SAMPLING AND IMAGE INTERPOLATION

Suppose that the size of cover image is $m \times n$. For simplicity, let m and n be even numbers. Thus, the down-sampling will produce an image sized $(m/2) \times (n/2)$ as follows. The cover image is firstly divided into non-overlapping blocks sized 2×2. For each block, the pixels are numbered with 1 to 4 from left to right and top to bottom. Figure 2 presents the pixel order in a 2×2 block. Consequently, the down-sampled image is generated by picking out the fourth pixels of all blocks.

To exploit prediction errors for data embedding, the down-sampled image is resized to $m \times n$ by image interpolation and then prediction error matrix between cover image and interpolated image can be determined. Here, the well-known

algorithm called bicubic interpolation is taken to achieve image interpolation. Let $f(i, j)$ be an original pixel and $g(i, j)$ be an interpolated pixel. Thus, the mathematical expression of bicubic interpolation [36] is as follows.

$$g(i+v, j+u) = \mathbf{A}_1\mathbf{A}_2\mathbf{A}_3 \tag{1}$$

where v stands for row number deviation, u stands for column number deviation, and the matrices $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{A}_3$ are defined as follows.

$$\mathbf{A}_1 = [S(1+v) \quad S(v) \quad S(1-v) \quad S(2-v)] \tag{2}$$

$$\mathbf{A}_2 = \begin{bmatrix} f(i-1,j-1) & f(i-1,j) & f(i-1,j+1) & f(i-1,j+2) \\ f(i,j-1) & f(i,j) & f(i,j+1) & f(i,j+2) \\ f(i+1,j-1) & f(i+1,j) & f(i+1,j+1) & f(i+1,j+2) \\ f(i+2,j-1) & f(i+2,j) & f(i+2,j+1) & f(i+2,j+2) \end{bmatrix} \tag{3}$$

$$\mathbf{A}_3 = [S(1+u)\, S(u)\, S(1-u)\, S(2-u)]^{\mathrm{T}} \tag{4}$$

in which the function $S(w)$ is determined by the below formula.

$$S(w) = \begin{cases} 1 - 2|w|^2 + |w|^3 & |w| < 1 \\ 4 - 8|w| + 5|w|^2 - |w|^3 & 1 < |w| < 2 \\ 0 & 2 < |w|. \end{cases} \tag{5}$$

In fact, the equation (1) can be written as follows.

$$g(i+v, j+u) = \sum_{r=-1}^{2} \sum_{c=-1}^{2} f(i+r, j+c) S(r-v) S(c-u) \tag{6}$$

Note that some pixels of the interpolated image are directly filled by the original pixels. Specifically, $g(2i, 2j) = f(i, j)$ is firstly calculated, where $1 \le i \le m$ and $1 \le j \le n$. Other pixels of the interpolated image are determined by the bicubic interpolation.

B. HISTOGRAM SHIFTING

In the proposed algorithm, we calculate error matrices and difference matrices by prediction techniques via image interpolation or local pixel similarity, respectively, and generate error histograms and difference histograms for data embedding by HS. The generations of error histogram and difference histogram are similar. Their details are as follows. Let $p_{i,j}$ be a pixel and $q_{i,j}$ be its reference pixel determined by interpolation or local pixel. Thus, the error/difference value $d_{i,j}$ is determined by the below equation.

$$d_{i,j} = p_{i,j} - q_{i,j} \tag{7}$$

After the error/difference values of available pixels are calculated, the corresponding error histogram or difference histogram is then available.

Once histogram is generated, data embedding can be done by HS. In this work, the bin 0 and bin 1 are used to embed data. To do so, other bins are first shifted to create embedding space. Specifically, the bins in the left side of the bin 0 and the bins in the right side of bin 1 are all shifted with one step along the left direction and the right direction, respectively. In other words, If $d_{i,j} \le 0$, it is shifted by -1. And if $d_{i,j} \ge 1$, it is shifted by 1. The shifting operation is defined as follows.

$$d'_{i,j} = \begin{cases} d_{i,j} - 1 & \text{if } d_{i,j} \le 0 \\ d_{i,j} + 1 & \text{if } d_{i,j} \ge 1. \end{cases} \tag{8}$$

As the reference pixel $q_{i,j}$ is unchanged, the above shifting operation is equivalent to the below equation.

$$p'_{i,j} = \begin{cases} p_{i,j} - 1 & \text{if } d_{i,j} \le 0 \\ p_{i,j} + 1 & \text{if } d_{i,j} \ge 1 \end{cases} \tag{9}$$

where $p'_{i,j}$ is the new pixel after shifting operation. According to the equation (7), the new pixel and the new difference have the following relation.

$$d'_{i,j} = p'_{i,j} - q_{i,j} \tag{10}$$

After all new differences $d'_{i,j}$ are calculated, our data embedding can be done by the below rules.

$$d''_{i,j} = \begin{cases} d'_{i,j} & \text{if } d'_{i,j} = -1 \text{ and } b = 0 \\ d'_{i,j} + 1 & \text{if } d'_{i,j} = -1 \text{ and } b = 1 \\ d'_{i,j} & \text{if } d'_{i,j} = 2 \text{ and } b = 0 \\ d'_{i,j} - 1 & \text{if } d'_{i,j} = 2 \text{ and } b = 1 \end{cases} \tag{11}$$

where $b \in \{0, 1\}$ represents the to-be-embedded bit. Therefore, according to the equation (10), our data embedding can be achieved by directly modifying the new pixels as follows.

$$p''_{i,j} = \begin{cases} p'_{i,j} & \text{if } d'_{i,j} = -1 \text{ and } b = 0 \\ p'_{i,j} + 1 & \text{if } d'_{i,j} = -1 \text{ and } b = 1 \\ p'_{i,j} & \text{if } d'_{i,j} = 2 \text{ and } b = 0 \\ p'_{i,j} - 1 & \text{if } d'_{i,j} = 2 \text{ and } b = 1 \end{cases} \tag{12}$$

where $p''_{i,j}$ is the marked pixel. Figure 3 illustrates schematic diagram of our data embedding with HS. Figure 3 (a) is an original histogram, Figure 3 (b) is the shifted histogram calculated by the equation (8), and Figure 3 (c) is the shifted histogram after data embedding with the equation (11).

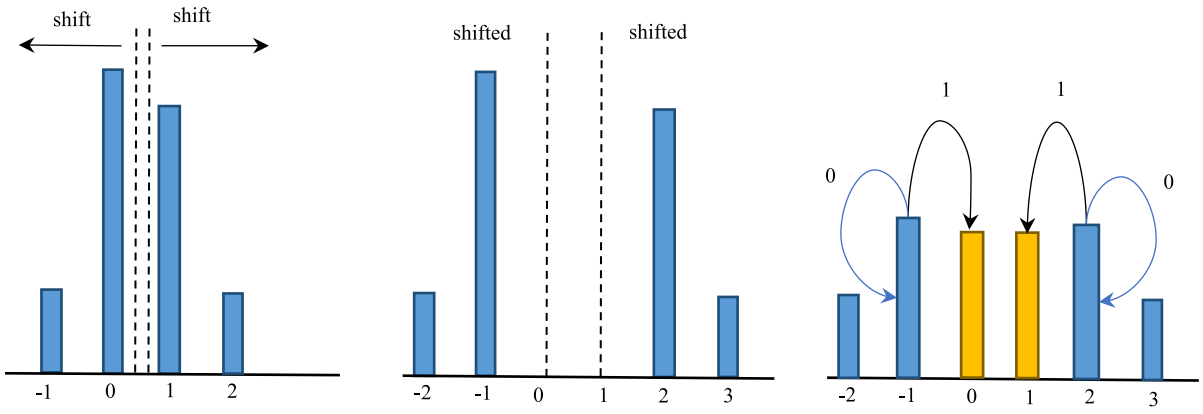
To extract the embedded bit, the error/difference value between the marked pixel and its reference pixel is first calculated as follows.

$$d''_{i,j} = p''_{i,j} - q_{i,j} \tag{13}$$

Then, the embedded bit can be determined by the below rule.

$$b = \begin{cases} 0 & \text{if } d''_{i,j} = -1 \text{ or } d''_{i,j} = 2 \\ 1 & \text{if } d''_{i,j} = 0 \text{ or } d''_{i,j} = 1 \end{cases} \tag{14}$$

To ensure correct data extraction and image recovery, some auxiliary information should be recorded and self-embedded into image. The required auxiliary information includes the number of embedded bits, the number of the saturated pixels equaling 0 or 255 which may overflow during histogram shifting, and the positions of all saturated pixels. Clearly, the number of embedded bits reaches the maximum value when all pixels can be used to conceal secret bit. Therefore, the maximum number of embedded bits is mn, which only



(a) Original histogram (b) Shifted histogram (c) Shifted histogram after embedding

FIGURE 3. Schematic diagram of data embedding by shifting histogram.

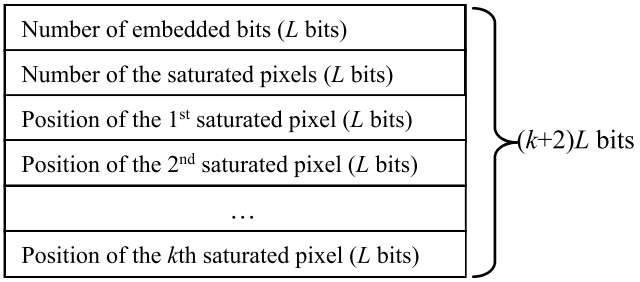


FIGURE 4. Structure of auxiliary information.

requires $L = \lfloor \log_2 mn \rfloor$ bits for storage where $\lfloor \cdot \rfloor$ is the upward rounding. Similarly, the maximum number of the saturated pixels is also mn. In addition, the position of the saturated pixel in the i-th row and j-th column can be recorded by its order number, i.e., $j + (i-1)n$, whose maximum number is also mn. Therefore, the storages of the number of the saturated pixels and the position of a saturated pixel are both $L = \lfloor \log_2 mn \rfloor$ bits. Suppose that the number of the saturated pixels is k. Thus, $(k+2)L$ bits are needed to store auxiliary information. Figure 4 presents structure of auxiliary information. In this work, auxiliary information is embedded into the LSBs of the pixels in the first and second rows by the well-known LSB replacement. Since the first and second rows provide $2n$ bits for data concealment. If $(k+2)L > 2n$, the current image cannot be used as cover image. In this case, data-hider should select another cover image. Moreover, the LSBs of the pixels in the first and second rows are picked out before the operation of LSB replacement. These extracted LSBs and the secret data are concatenated and embedded into image by our HS technique (pixels in the first and second rows are not used to calculate histogram).

Note that auxiliary information is firstly extracted during data extraction and image recovery. According to the auxiliary information, the embedded bits are then extracted by using the equation (14). Next, the LSBs of the used pixels in the first two rows are separated from the extracted bits and are used to recover their corresponding pixels. Finally, the restored image can be determined by re-shifting all histogram bins, except the bin 0 and the bin 1. It is equivalent to modifying the marked pixels as follows.

$$p_{i,j} = \begin{cases} p''_{i,j} + 1 & \text{if } d''_{i,j} \le -1 \\ p''_{i,j} - 1 & \text{if } d''_{i,j} \ge 2 \end{cases} \tag{15}$$

It is worth noting that the saturated pixels recorded in auxiliary information are not needed to perform the equation (15). From the above analysis, it can be seen that the restored image is perfectly the same with its original image.

C. FIRST-LAYER EMBEDDING

The first-layer embedding is done by the following steps.

STEP 1: The $m \times n$ color image is resized to an image sized $(m/2) \times (n/2)$ by the down-sampling operation as described in Section II A. The red, green and blue channels of the down-sampled image are then resized to $m \times n$ by the image interpolation explained in Section II A.

STEP 2: Let $\mathbf{R}_0$, $\mathbf{G}_0$ and $\mathbf{B}_0$ be the red, green and blue channels of the original $m \times n$ color image. Also, let $\mathbf{R}_1$, $\mathbf{G}_1$ and $\mathbf{B}_1$ be the red, green and blue channels of the interpolated image. Thus, error matrices between the original channels and the interpolated channels can be calculated by the following equations.

$$\mathbf{E}_1 = \mathbf{R}_0 - \mathbf{R}_1 \tag{16}$$
$$\mathbf{E}_2 = \mathbf{G}_0 - \mathbf{G}_1 \tag{17}$$
$$\mathbf{E}_3 = \mathbf{B}_0 - \mathbf{B}_1 \tag{18}$$

where the above minus represents subtraction between the corresponding elements of two matrices.

STEP 3: The correlation between color channels is then exploited to create two difference matrices as follows.

$$\mathbf{D}_1 = \mathbf{E}_1 - \mathbf{E}_2 \tag{19}$$
$$\mathbf{D}_2 = \mathbf{E}_3 - \mathbf{E}_2 \tag{20}$$

STEP 4: To embed data into the green channel, the histogram of the error matrix $\mathbf{E}_2$ is calculated, where only the error values in the positions of the interpolated pixels are used (value in the position of the pixel 4 in 2×2 block is not used). Then, additional data is embedded into the histogram of $\mathbf{E}_2$ by our HS technique as described in Section II B. As to the red channel, if the number of the elements equaling 0 and 1 in $\mathbf{D}_1$ is bigger than the number of the elements equaling 0 and 1 in $\mathbf{E}_1$ (value in the position of the pixel 4 in 2×2 block is not calculated), data embedding is conducted on the histogram of $\mathbf{D}_1$ by our HS technique. Otherwise, the histogram of $\mathbf{E}_1$ is used to embed additional data. Similarly, for the blue channel, if the number of the elements equaling 0 and 1 in $\mathbf{D}_2$ is bigger than the number of the elements equaling 0 and 1 in $\mathbf{E}_3$ (value in the position of the pixel 4 in 2×2 block is not calculated), data embedding is conducted on the histogram of $\mathbf{D}_2$ by our HS technique. Otherwise, the histogram of $\mathbf{E}_3$ is used to embed additional data.

Note that the data embedded into the channels is formed by the LSBs of pixels in their first two rows and the secret data. To enhance security, the data is encrypted by bitwise XOR operation with a pseudorandom bit stream generated by the famous RC4 algorithm [37] under the control of a secret key. Consequently, the practical embedded data is the encrypted version of the LSBs and the secret data.

D. SECOND-LAYER EMBEDDING

After the marked image generated by the first-layer embedding is obtained, our second-layer embedding is then performed to improve embedding capacity. Detailed steps of the second-layer embedding are explained as follows.

STEP 1: Difference matrices of the three channels of the marked image generated by the first-layer embedding are calculated. The red channel is first divided into non-overlapping blocks sized 2×2. For each block, pixel differences between the pixel 4 and other pixels are calculated. Let $b_{i,1}$, $b_{i,2}$, $b_{i,3}$ and $b_{i,4}$ be the pixel 1, the pixel 2, the pixel 3 and the pixel 4 of the i-th block, respectively. Thus, the calculation is done by the following rules. The pixel 1 is updated by subtracting the pixel 4 from the pixel 1, the pixel 2 is updated by subtracting the pixel 4 from the pixel 2, and the pixel 3 is updated by subtracting the pixel 4 from the pixel 3, i.e., $b_{i,1} \leftarrow b_{i,1} - b_{i,4}$, $b_{i,2} \leftarrow b_{i,2} - b_{i,4}$, and $b_{i,3} \leftarrow b_{i,3} - b_{i,4}$. The difference matrix of the red channel is finally extracted after all blocks are processed, except those blocks in the first two rows (pixels in these rows are used for embedding auxiliary information). Similarly, the difference matrices of the green channel and the blue channel can be determined by the same operations with the extraction of the difference matrix of the red channel.

STEP 2: Let $\mathbf{D}_3$, $\mathbf{D}_4$ and $\mathbf{D}_5$ be the difference matrices of the green, red and blue channels, respectively. Thus, the error matrices between the red/blue channel and the green channel are calculated by the below equations.

$$\mathbf{E}_4 = \mathbf{D}_4 - \mathbf{D}_3 \tag{21}$$

$$\mathbf{E}_5 = \mathbf{D}_5 - \mathbf{D}_3 \tag{22}$$

STEP 3: Additional data is embedded into the green, red and blue channels as follows. The histogram of $\mathbf{D}_3$ is first extracted, where the pixels in the first two rows are not used and the pixel 4 of every block is also not used. Then, additional data is embedded into the histogram of $\mathbf{D}_3$ by our HS technique. As to the red channel, if the number of the elements equaling 0 and 1 in $\mathbf{D}_4$ is bigger than the number of the elements equaling 0 and 1 in $\mathbf{E}_4$ (values in the position of 2×2 blocks are not calculated), data embedding is conducted on the histogram of $\mathbf{D}_4$ by our HS technique. Otherwise, the histogram of $\mathbf{E}_4$ is used to embed additional data. Similarly, for the blue channel, if the number of the elements equaling 0 and 1 in $\mathbf{D}_5$ is bigger than the number of the elements equaling 0 and 1 in $\mathbf{E}_5$ (values in the position of 2×2 blocks are not calculated), data embedding is conducted on the histogram of $\mathbf{D}_5$ by our HS technique. Otherwise, the histogram of $\mathbf{E}_5$ is used to embed additional data.

STEP 4: The final marked image is obtained by stacking the marked red, green and blue channels.

E. DATA EXTRACTION AND IMAGE RECOVERY

As our RDH algorithm exploits double-layer embedding to conduct data hiding, additional data embedded by the second-layer embedding is firstly extracted and the restored image is then calculated by LSB replacement and re-shifting histogram. Next, additional data embedded by the first-layer embedding is also extracted based on the restored image and the final restored image is lastly recovered by LSB replacement and re-shifting histogram. Detailed steps are as follows.

STEP 1: Embedded data in the green channel of the marked image is extracted and the green channel is then restored. To do so, auxiliary information hidden in the first two rows of the green channel is firstly extracted. The green channel is divided into non-overlapping 2×2 blocks, and its difference matrix is determined by the same operations as mentioned in the **STEP 1** of Section II D. According to the auxiliary information, the embedded bits are determined by the extraction rule as mentioned in the equation (14). As the embedded bits are encrypted before data embedding, they are decrypted by bitwise XOR operation with a pseudorandom bit stream generated by the RC4 algorithm under the control of a secret key. Once the decrypted bits are available, the LSBs of the used pixels in the first two rows and the secret data can be separated according to the bit length of auxiliary information. Next, the extracted LSBs are used to recover their original pixels by LSB replacement. Finally, the green channel after the first-layer embedding can be recovered by re-shifting histogram as mentioned in the equation (15).

STEP 2: Embedded data in the red channel of the marked image is extracted and the red channel is then restored. Similarly, auxiliary information hidden in the first two rows of the red channel is firstly extracted, and the difference matrix of the red channel is determined by the same operations as mentioned in the **STEP 1** of Section II D. The error matrix of the red channel is obtained by subtracting difference matrix of the green channel from the difference matrix of the red

channel, as described in the equation (21). If the number of the elements equaling 0 and 1 in the difference matrix of the red channel is bigger than the number of the elements equaling 0 and 1 in the error matrix of the red channel (values in the position of 2×2 blocks are not calculated), data extraction and the red channel recovery are done based on the difference matrix of the red channel by the similar operations as described in the **STEP 1**. Otherwise, data extraction and the red channel recovery are performed in terms of the error matrix of the red channel.

STEP 3: Embedded data extraction and recovery of the blue channel are done by using the similar operations of the data extraction and recovery of the red channel as described in the **STEP 2**.

STEP 4: After the above three steps, the red, green and blue channels contained additional data embedded by the first-layer embedding are obtained. To recover the green channel, auxiliary information hidden in its first two rows is firstly extracted. Then, the obtained green channel is down-sampled and interpolated, as described in the **STEP 1** of Section II C. The error matrix of the green channel is determined by subtracting the interpolated green channel from the obtained green channel, as described in the **STEP 2** of Section II C. Then, data extraction and the green channel recovery are conducted in terms of the error matrix of the green channel.

STEP 5: For the obtained red channel, auxiliary information hidden in its first two rows is also firstly extracted. Then, it is down-sampled and interpolated, as described in the **STEP 1** of Section II C. The error matrix of the red channel is determined by subtracting the interpolated red channel from the obtained red channel, as described in the **STEP 2** of Section II C. The difference matrix between the red channel and the green channel is determined by subtracting the error matrix of the green channel from the error matrix of the red channel. If the number of the elements equaling 0 and 1 in the error matrix of the red channel is bigger than the number of the elements equaling 0 and 1 in the difference matrix of the red channel (values in the position of 2×2 blocks are not calculated), data extraction and the red channel recovery are done based on the error matrix of the red channel. Otherwise, data extraction and the red channel recovery are performed in terms of the difference matrix of the red channel.

STEP 6: Embedded data extraction and recovery of the blue channel are done by using the similar operations of the data extraction and recovery of the red channel as described in the **STEP 5**.

III. EXPERIMENTAL RESULTS

In the experiments, performances of our RDH algorithm are evaluated with embedding capacity and visual quality. The embedding capacity is measured by the number of the embedded bits, and the visual quality is evaluated by the well-known metric called PSNR. Section III A presents our data-hiding results. Section III B discusses effect of channel selection on the embedding capacity.

FIGURE 5. Standard color images.

FIGURE 6. Marked images.

A. DATA-HIDING RESULTS

Eight standard color images from the open dataset called USC-SIPI Image Database [38] are selected as test images. These test images are shown in Figure 5. We embed secret bits into these images under maximum embedding capacity (i.e., available pixels are all used) and obtain the marked images as shown in Figure 6. The embedding capacities under different color images and PSNRs between the original images and marked images are calculated. In addition, the embedding capacities of the first-layer embedding and the second-layer embedding are also recorded. From Figures 5 and 6, it can be observed that the marked images are visually the same with their original images. Figure 7 demonstrates our embedding capacities under different color images, where the y-axis is the embedding capacity, and the x-axis is the first-layer embedding, the second-layer embedding, and the total result. Table 1 illustrates the detailed capacities contributed by different layers. It is observed that the first-layer embedding and the second-layer embedding both contribute large capacities. This validates effectiveness of the double-layer embedding technique. Table 2 lists the detailed results, where the second, third and fourth columns are the capacities contributed by the red channel, the green channel and the blue channel, respectively, and the fifth column is the sum of the capacities of the red, green and blue channels, and the last column is the PSNR. It can be seen that the total capacities of these color images are all bigger than 1.0×10^5 bits, except that of Baboon.

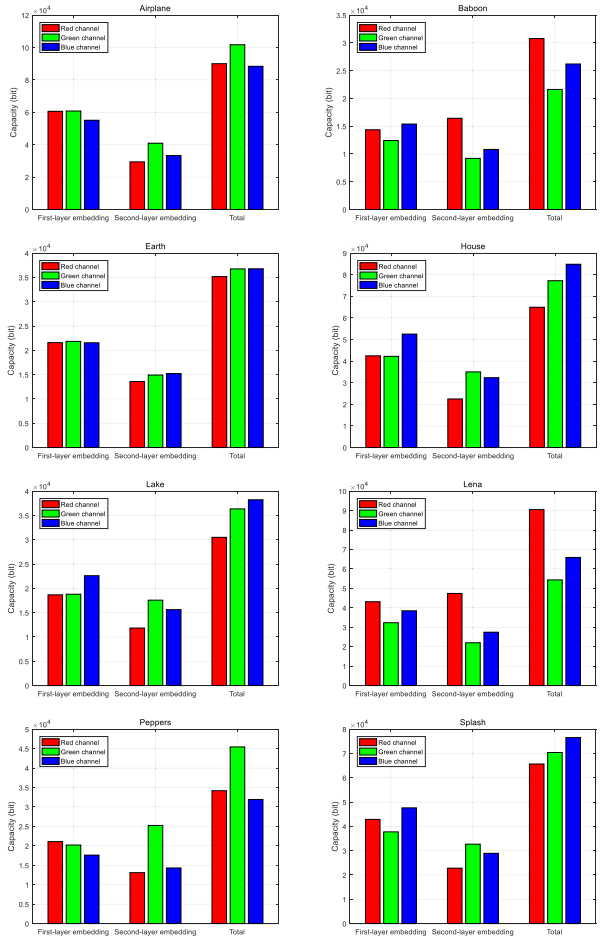


FIGURE 7. Capacities of our RDH algorithm based on eight test images.

TABLE 1. Capacities contributed by different layers (unit: bit).

Image	First-layer embedding	Second-layer embedding	Total
Airplane	176495	103643	280138
Baboon	42158	36446	78604
Earth	64991	43722	108713
House	137199	89847	227046
Lake	60067	45027	105094
Lena	113933	96836	210769
Peppers	58894	52645	111539
Splash	128245	84383	212628
Average	97747.75	69068.63	166816.38

The average total capacity of these images reaches 166816 bits. Moreover, the PSNRs between the original images and the marked images are all bigger than 50.00 dB, and the average PSNR is 52.30 dB. A high capacity and a big PSNR illustrate efficiency of our RDH algorithm.

B. EFFECT OF CHANNEL SELECTION

In this work, inter-channel correlation is exploited to conduct data embedding. Specifically, in the **STEP 3** of the first-layer embedding, difference matrix is calculated by subtracting the error matrix of green channel from the error matrix of red/blue channel. Similarly, in the **STEP 2** of the second-layer

TABLE 2. Our capacities and PSNRs under different test images.

Image	Capacity (bit)				PSNR (dB)
	Red	Green	Blue	Total	
Airplane	90029	101717	88392	280138	52.56
Baboon	30783	21619	26202	78604	52.72
Earth	35169	36755	36789	108713	52.02
House	64949	77240	84857	227046	52.57
Lake	30499	36359	38236	105094	51.84
Lena	90518	54348	65903	210769	52.67
Peppers	34179	45440	31920	111539	50.91
Splash	65675	70408	76545	212628	53.10
Average	55225.13	55485.75	56105.5	166816.38	52.30

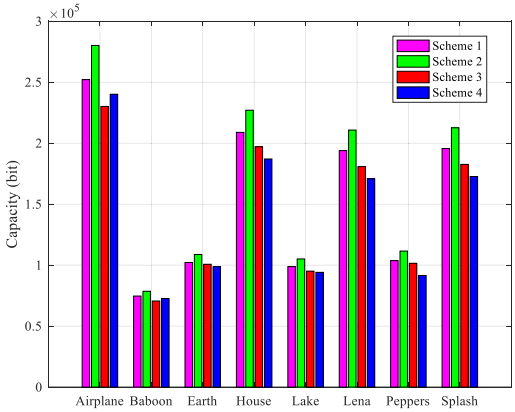


FIGURE 8. Capacity comparison among different schemes.

embedding, error matrix is determined by subtracting the difference matrix of green channel from the difference matrix of red/blue channel. Clearly, different uses of these channels will lead to different schemes. In this section, we compare the embedding capacities of four schemes as follows.

Scheme 1: This scheme does not use inter-channel correlation to embed data. Specifically, the **STEP 3** of the first-layer embedding is not needed. For the red channel and blue channel, it exploits their error matrices to embed data by similar operations in the error matrix of the green channel. Similarly, calculations of the equations (21) and (22) in the **STEP 2** of the second-layer embedding is not needed. For the red channel and blue channel, it exploits their difference matrices to embed data by similar operations in the difference matrix of the green channel.

Scheme 2: The proposed RDH algorithm is denoted as the Scheme 2.

Scheme 3: This scheme uses inter-channel correlation to embed data. Different from the Scheme 2, it subtracts the red channel from the green/blue channel in the first-layer embedding and the second-layer embedding. Other operations are similar with those of the Scheme 2.

Scheme 4: This scheme also uses inter-channel correlation to embed data. But it subtracts the blue channel from the red/green channel in the first-layer embedding and the

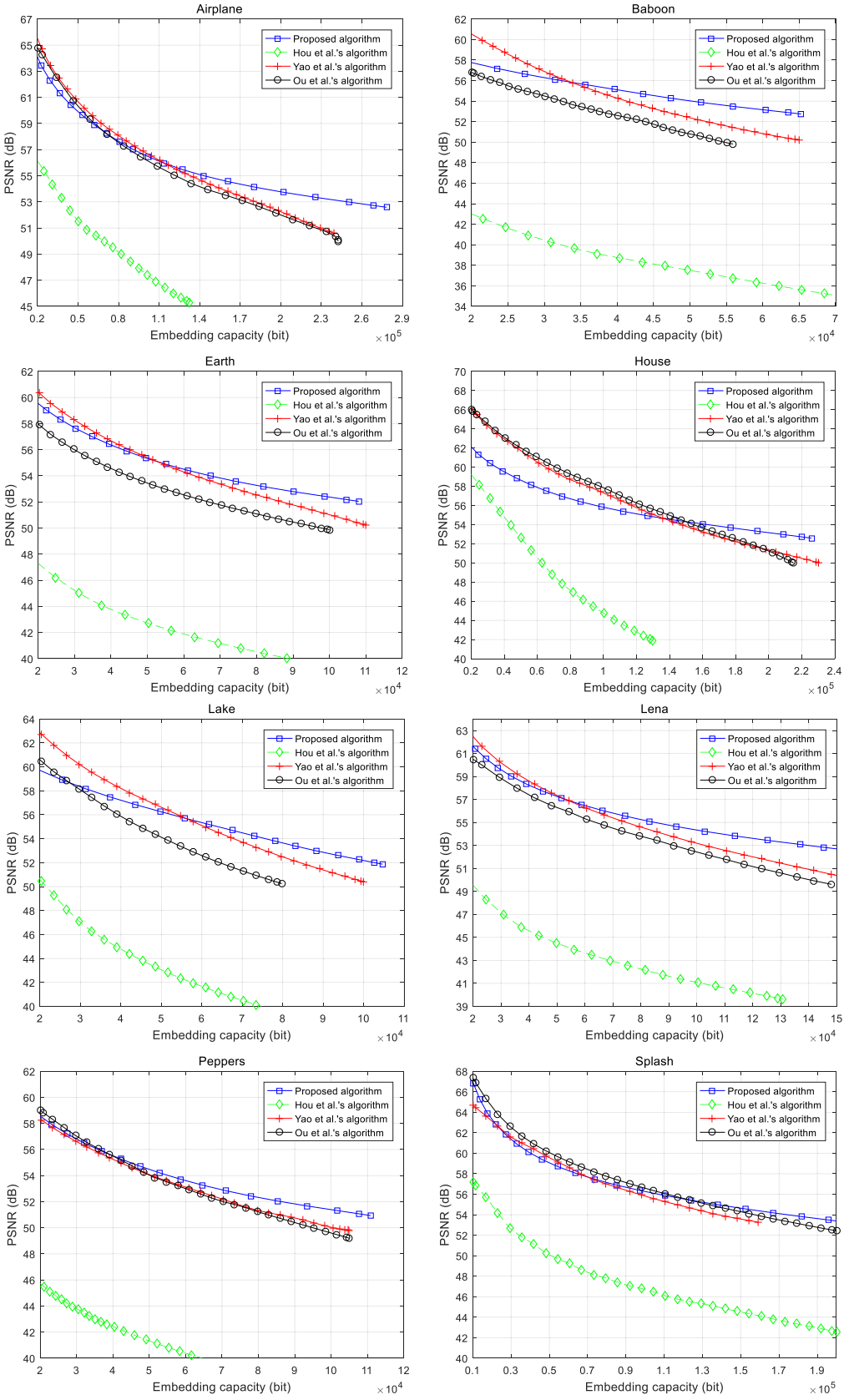


FIGURE 9. Performance comparison with respect to PSNR and embedding capacity.

TABLE 3. PSNR comparison under a fixed capacity of 100000 bits (unit: dB).

Image	Ou et al.'s algorithm	Yao et al.'s algorithm	Hou et al.'s algorithm	Proposed algorithm
Kodim01	52.08	55.46	39.00	**56.12**
Kodim02	54.91	55.19	45.69	**56.07**
Kodim03	54.65	55.59	48.53	**56.92**
Kodim04	55.21	55.38	45.27	**55.98**
Kodim05	53.21	54.84	40.75	**55.27**
Kodim06	54.26	54.27	41.53	**54.33**
Kodim07	54.23	55.72	47.78	**56.97**
Kodim08	–	54.35	39.15	**55.11**
Kodim09	54.77	55.64	45.89	**56.67**
Kodim10	54.75	55.55	45.68	**56.10**
Kodim11	55.70	55.56	44.38	**57.36**
Kodim12	55.56	55.22	46.57	**56.80**
Kodim13	–	54.23	36.67	**55.94**
Kodim14	52.79	55.40	41.35	**56.95**
Kodim15	53.69	53.95	46.25	**55.80**
Kodim16	54.83	55.64	46.51	**56.43**
Kodim17	55.15	55.68	46.10	**57.54**
Kodim18	51.03	54.71	41.55	**55.31**
Kodim19	54.05	55.31	44.94	**56.02**
Kodim20	**55.90**	52.41	41.45	53.35
Kodim21	53.78	55.31	45.02	**54.89**
Kodim22	53.62	54.99	43.31	**56.03**
Kodim23	55.70	55.44	46.95	**56.07**
Kodim24	54.69	54.51	43.45	**56.58**
Average	54.30	55.01	43.91	**56.03**

second-layer embedding. Other operations are similar with those of the Scheme 2.

Figure 8 presents capacity comparison among different schemes. Clearly, for all test images, the capacities of the Scheme 2 are bigger than those of other three schemes. It means that the capacity performance of the Scheme 2 is better than other three schemes. This illustrates the effectiveness of the use of inter-channel correlation in our double-layer embedding.

IV. PERFORMANCE COMPARISONS

To demonstrate advantages, our proposed RDH algorithm is compared with some state-of-the-art RDH algorithms. The selected RDH algorithms are Ou *et al.*'s algorithm [31], Yao *et al.*'s algorithm [32], and Hou *et al.*'s algorithm [35], which are all designed for color images and recently published in famous journals. Figure 9 demonstrates performance comparison with respect to PSNR and embedding capacity based on eight standard benchmark color images sized 512×512. In this comparison, we embed different secret bits into the color images by the evaluated algorithms and calculated the PSNRs between the original images and the marked images. It is observed that our PSNRs are bigger than those of Hou *et al.*'s algorithm under different embedding capacities. Moreover, most PSNRs of our algorithm are smaller than those of Ou *et al.*'s algorithm and Yao *et al.*'s algorithm when the embedding capacity is small. Note that our algorithm is not better than Ou *et al.*'s algorithm and Yao *et al.*'s algorithm in small embedding capacity. This is because our algorithm uses the double-layer embedding technique to conduct data embedding. When the embedding capacity is small, only the first-layer embedding is used and the second-layer embedding is not exploited. Take the Airplane for example. Our PSNRs are smaller than those of Ou *et al.*'s algorithm and Yao *et al.*'s algorithm when the embedding capacity is smaller than 1.3×10^5 bits, which is much smaller than the capacity 1.76×10^5 bits contributed by the first-layer embedding (See Table 1).

In addition, some image pixels are changed twice by the double-layer embedding technique. Specifically, the first time is changed by the first-layer embedding and the second time is altered by the second-layer embedding. This means that some pixels will be re-changed to their original values in the second-layer embedding. This property helps to make a high capacity and preserve good visual image quality. Therefore, when the embedding capacity increases to a large value, our PSNRs are bigger than those of Ou *et al.*'s algorithm and Yao *et al.*'s algorithm. This illustrates that our RDH algorithm is better than the compared RDH algorithms in large embedding capacity. Our RDH algorithm can make good visual quality of marked image under large embedding capacity. This is contributed by our double-layer embedding technique and the use of inter-channel correlation in data embedding.

Performance comparison on the Kodak image database [39] is also conducted, which contains 24 color images sized 512×768 or 768×512. In the experiments, 100000 bits are embedded into every image of the Kodak database by using the evaluated RDH algorithms, and the PSNRs between the original color images and their marked images are then calculated. Table 3 lists the PSNR comparisons under the fixed capacity of 100000 bits, where the numbers in bold are the best results (biggest values), and the symbol '–' denotes that the algorithm cannot conduct data embedding

TABLE 4. Time comparison among different RDH algorithms.

Algorithm	Time of an image (T)	Capacity of an image (N)	Time of embedding one bit (T/N)
Ou et al.'s	2.0106 s	107337 bits	0.000019 s
Yao et al.'s	57.2589 s	123757 bits	0.000463 s
Hou et al.'s	1.2539 s	76189 bits	0.000016 s
Proposed	10.2543 s	139655 bits	0.000073 s

under this capacity. It can be seen that the PSNRs of the proposed algorithm reach the best results for all images, except the image named Kodim20. For the Kodim20, the PSNR of Ou *et al.*'s algorithm reaches the biggest value and the PSNR of the proposed algorithm is bigger than those of Yao *et al.*'s algorithm and Hou *et al.*'s algorithm. As to the average PSNR, the values of Ou *et al.*'s algorithm, Yao *et al.*'s algorithm, Hou *et al.*'s algorithm and the proposed algorithm are 54.30, 55.01, 43.91 and 56.03 dB, respectively. Clearly, our average PSNR is the biggest one. This also validates that the proposed algorithm is better than the compared algorithm in terms of PSNR and embedding capacity.

Time of data embedding is also calculated. All algorithms are coded with MATLAB R2016a and run on a personal computer with a 3.40 GHz CPU (Intel Core i7-6700) and 8 GB RAM. The operating system installed in the computer is Windows10 Professional. The 1338 color images taken from the UCID [40] are taken as test images. We exploit the assessed RDH algorithms to embed secret data into these test images under their full capacities and compute the average time of embedding one bit. Table 4 demonstrates time comparison among the assessed RDH algorithms, where the second column is the average time of embedding data in an image, the third column is the average capacity of an image, and the fourth column is the average time of embedding one bit. It can be seen that the proposed algorithm is faster than Yao *et al.*'s algorithm, but it is slower than Ou *et al.*'s algorithm and Hou *et al.*'s algorithm.

V. CONCLUSION

In this paper, we have proposed a novel RDH algorithm for color image based on double-layer embedding. An important contribution is the double-layer embedding technique with histogram shifting, which not only can ensure reversible data hiding, but also reaches high embedding capacity. Moreover, inter-channel correlation is exploited during the first-layer embedding and the second-layer embedding, and thus improves embedding capacity. Many experiments have been conducted to validate performances of the proposed algorithm. Comparisons with some recent state-of-the-art RDH algorithm have shown that the proposed RDH algorithm outperforms the compared algorithms in terms of PSNR and embedding capacity.

ACKNOWLEDGMENT

The authors would like to thank Dr. Bo Ou [31] for sharing their code for performance comparisons. They would also like to thank the anonymous referees for the valuable comments and suggestions.

REFERENCES

[1] J. Tao, S. Li, X. Zhang, and Z. Wang, "Towards robust image steganography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 2, pp. 594–600, Feb. 2019.
[2] S. Li and X. Zhang, "Toward construction-based data hiding: From secrets to fingerprint images," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1482–1497, Mar. 2019.
[3] X. Zhang, "Reversible data hiding with optimal value transfer," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 316–325, Feb. 2013.
[4] C. Yu, X. Zhang, Z. Tang, and X. Xie, "Separable and error-free reversible data hiding in encrypted image based on two-layer pixel errors," *IEEE Access*, vol. 6, pp. 76956–76969, 2018.
[5] F. Peng, Z.-X. Lin, X. Zhang, and M. Long, "Reversible data hiding in encrypted 2D vector graphics based on reversible mapping model for real numbers," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 9, pp. 2400–2411, Sep. 2019.
[6] H. Yao, C. Qin, Z. Tang, and Y. Tian, "Improved dual-image reversible data hiding method using the selection strategy of shiftable pixels' coordinates with minimum distortion," *Signal Process.*, vol. 135, pp. 26–35, Jun. 2017.
[7] J. Fridrich, M. Goljan, and D. Rui, "Invertible authentication," *Proc. SPIE*, vol. 4314, pp. 197–208, Aug. 001.
[8] W. Zhang, B. Chen, and N. Yu, "Improving various reversible data hiding schemes via optimal codes for binary covers," *IEEE Trans. Image Process.*, vol. 21, no. 6, pp. 2991–3003, Jun. 2012.
[9] W. Zhang, X. Hu, X. Li, and N. Yu, "Recursive histogram modification: Establishing equivalency between reversible data hiding and lossless data compression," *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2775–2785, Jul. 2013.
[10] C. Qin, C.-C. Chang, and Y.-C. Chen, "Efficient reversible data hiding for VQ-compressed images based on index mapping mechanism," *Signal Process.*, vol. 93, no. 9, pp. 2687–2695, Sep. 2013.
[11] Z. Tang, S. Xu, H. Yao, C. Qin, and X. Zhang, "Reversible data hiding with differential compression in encrypted image," *Multimed Tools Appl.*, vol. 78, no. 8, pp. 9691–9715, Apr. 2019.
[12] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.
[13] 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.
[14] 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.
[15] Z. Ni, Y. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits. Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar. 2006.
[16] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 7, pp. 1091–1100, Jul. 2013.
[17] C. Qin, C.-C. Chang, Y.-H. Huang, and L.-T. Liao, "An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1109–1118, Jul. 2013.
[18] Z. Tang, S. Xu, D. Ye, J. Wang, X. Zhang, and C. Yu, "Real-time reversible data hiding with shifting block histogram of pixel differences in encrypted image," *J Real-Time Image Process.*, vol. 16, no. 3, pp. 709–724, Jun. 2019.
[19] Q. Ying, Z. Qian, X. Zhang, and D. Ye, "Reversible data hiding with image enhancement using histogram shifting," *IEEE Access*, vol. 7, pp. 46506–46521, 2019.
[20] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.
[21] W. Hong, T.-S. Chen, and C.-W. Shiu, "Reversible data hiding for high quality images using modification of prediction errors," *J. Syst. Softw.*, vol. 82, no. 11, pp. 1833–1842, Nov. 2009.
[22] 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.
[23] 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. 2013.

[24] B. Ou, X. Li, J. Wang, and F. Peng, "High-fidelity reversible data hiding based on geodesic path and pairwise prediction-error expansion," *Neurocomputing*, vol. 226, pp. 23–34, Feb. 2017.
[25] D. Coltuc and J.-M. Chassery, "Very fast watermarking by reversible contrast mapping," *IEEE Signal Process. Lett.*, vol. 14, no. 4, pp. 255–258, Apr. 2007.
[26] S. Weng, Y. Zhao, J.-S. Pan, and R. Ni, "Reversible watermarking based on invariability and adjustment on pixel pairs," *IEEE Signal Process. Lett.*, vol. 15, pp. 721–724, 2008.
[27] X. Wang, X. Li, B. Yang, and Z. Guo, "Efficient generalized integer transform for reversible watermarking," *IEEE Signal Process. Lett.*, vol. 17, no. 6, pp. 567–570, Jun. 2010.
[28] F. Peng, X. Li, and B. Yang, "Adaptive reversible data hiding scheme based on integer transform," *Signal Process.*, vol. 92, no. 1, pp. 54–62, Jan. 2012.
[29] W.-J. Yang, K.-L. Chung, and H.-Y.-M. Liao, "Efficient reversible data hiding for color filter array images," *Inf. Sci.*, vol. 190, pp. 208–226, May 2012.
[30] J. Li, X. Li, and B. Yang, "Reversible data hiding scheme for color image based on prediction-error expansion and cross-channel correlation," *Signal Process.*, vol. 93, no. 9, pp. 2748–2758, Sep. 2013.
[31] B. Ou, X. Li, Y. Zhao, and R. Ni, "Efficient color image reversible data hiding based on channel-dependent payload partition and adaptive embedding," *Signal Process.*, vol. 108, pp. 642–657, Mar. 2015.
[32] H. Yao, C. Qin, Z. Tang, and Y. Tian, "Guided filtering based color image reversible data hiding," *J. Vis. Commun. Image Represent.*, vol. 43, pp. 152–163, Feb. 2017.
[33] L. Wang and Z. Pan, "An improved reversible data hiding scheme using best neighboring coding based on color images," *Multimed Tools Appl.*, vol. 77, no. 20, pp. 26711–26739, Oct. 2018.
[34] M. Xu and J. Li, "3D PEE mapping based reversible data hiding for color images," *Multimed Tools Appl.*, vol. 78, no. 7, pp. 8003–8016, Apr. 2019.
[35] D. Hou, W. Zhang, K. Chen, S.-J. Lin, and N. Yu, "Reversible data hiding in color image with grayscale invariance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 2, pp. 363–374, Feb. 2019.
[36] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-29, no. 6, pp. 1153–1160, Dec. 1981.
[37] R. L. Rivest, *The RC4 Encryption Algorithm*. Bedford, MA, USA: RSA Data Security Inc., 1992.
[38] *USC-SIPI Image Database*. Accessed: Feb. 10, 2007. [Online]. Available: http://sipi.usc.edu/database/
[39] *Kodak Lossless True Color Image Suite*. Accessed: Apr. 15, 2017. [Online]. Available: http://r0k.us/graphics/kodak/
[40] G. Schaefer and M. Stich, "UCID: An uncompressed color image database," *Proc. SPIE*, vol. 5307, pp. 472–480, Dec. 2004.

ZHENJUN TANG (Member, IEEE) received the B.S. and M.Eng. degrees from Guangxi Normal University, Guilin, China, in 2003 and 2006, respectively, and the Ph.D. degree from Shanghai University, Shanghai, China, in 2010. He is currently a Professor with the School of Computer Science and Information Technology, Guangxi Normal University. He has contributed more than 60 articles in international journals, such as IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Information Forensics and Security, *The Computer Journal*, *Computers & Security, Signal Processing*, *Digital Signal Processing*, *IET Image Processing*, *Multimedia Tools and Applications*, and *Fundamenta Informaticae*. His research interests include image processing and multimedia security. He is a Senior Member of the China Computer Federation (CCF) and a Reviewer of more than 30 SCI journals, such as the IEEE journals, IET journals, Elsevier journals, Springer journals, and Taylor & Francis journals.

HEWANG NIE received the B.S. degree in computer science and technology from the Guangxi University of Science and Technology, Liuzhou, China. He is currently pursuing the M.Eng. degree with the School of Computer Science and Information Technology, Guangxi Normal University, Guilin, China. His research interests include information hiding and image processing.

CHI-MAN PUN (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in software engineering from the University of Macau, in 1995 and 1998, respectively, and the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, in 2002. He is currently an Associate Professor and the Head of the Department of Computer and Information Science, University of Macau. He has investigated several funded research projects. He has authored or coauthored more than 100 refereed scientific articles in international journals, books, and conference proceedings. His research interests include digital image processing, multimedia forensics and watermarking, pattern recognition, and computer vision. He is a Professional Member of the ACM. He has served as an Editorial Member/Referee for many international journals, such as the IEEE Transactions on Pattern Analysis and Machine Intelligence, the IEEE Transactions on Image Processing, and the IEEE Transactions on Information Forensics and Security.

HENG YAO (Member, IEEE) received the B.S. degree from the Hefei University of Technology, China, in 2004, the M.S. degree from Shanghai Normal University, China, in 2008, and the Ph.D. degree from Shanghai University, China, in 2012. He is currently an Associate Professor with the School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, China. His research interests include digital forensics, data hiding, image processing, and pattern recognition.

CHUNQIANG YU received the B.S. degree from Jiangxi Normal University, in 2010, and the M.Eng. degree from Guangxi Normal University, in 2013. His research interests include image processing and information hiding.

XIANQUAN ZHANG received the M.Eng. degree from Chongqing University, Chongqing, China. He is currently a Professor with the School of Computer Science and Information Technology, Guangxi Normal University. He has contributed more than 100 articles. His research interests include image processing and information hiding.