IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 18, 2021, accepted October 1, 2021, date of publication October 4, 2021, date of current version October 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3117961

Exploring More Capacity for Grayscale-Invariance Reversible Data Hiding

CHUN-LIANG JHONG AND HSIN-LUNG WU
Department of Computer Science and Information Engineering, National Taipei University, New Taipei City 237, Taiwan

Corresponding author: Hsin-Lung Wu (hsinlung@mail.ntpu.edu.tw)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST-110-2221-E-305-006-MY2; and in part by the National Taipei University under Grant 110-NTPU-ORDA-F-004.

ABSTRACT Grayscale-invariant reversible data hiding (RDH) in color images was developed recently in which the grayscales of the marked color image are required to be the same as that of the host color image. Recently, some grayscale-invariant RDH schemes were proposed but their performance on embedding capacity and distortion is not satisfactory due to the large number of correction bits generated for losslessly recovering the host green scales. In this work, we propose a grayscale-invariant RDH method which reduces the need for correction bits. We design a self-correcting mechanism which modifies the green scales for keeping grayscales unchanged while remembering the information of the host green scales. To obtain this task, we observe that all previous methods generated correction bits in a functional approach which results in many correction bits. Instead of the functional approach, we use a relational approach to preserve the grayscales. By this approach, the proposed algorithm only has to generate a small number of correction bits for recovering green scales. Moreover, this relational approach allows the algorithm to embed more message bits into some green scales. The experimental results show that the proposed method significantly improves the embedding capacity and the rate-distortion performance compared to previous works.

INDEX TERMS Color images, difference expansion, grayscale-invariance, reversible data hiding, self-correcting ability.

I. INTRODUCTION

Reversible data hiding (RDH) is a technique which embeds a secret message into an image such that, from the embedded image, one can not only extract the embedded secret message but also recover the original image without any distortion. This novel technique has found many applications in the area of medical image processing, military, and forensics where it is required to restore the original image without any distortion. There are many reversible data hiding techniques developed in the past decade. These popular methods are constructed based on two major strategies: the difference-expansion-based (DE-based) approach [1] and the histogram-shifting-based (HS-based) approach [2]. In order to generate marked image with small distortion, RDH methods usually apply these strategies to the residuals of the host images such as prediction errors [3]–[6].

Generally speaking, most RDH algorithms are designed for gray images. Some RDH algorithms for color images are proposed [7]–[10] and their goals aim to explore more correlations within three color channels in order to generate a sharper histogram of prediction errors and minimize the distortion of the marked images. When these RDH algorithms cause some modification for the host image, they may not preserve some important properties of the host color image such as its grayscale. There are many applications for the gray version of color images including black and white printing, black and white eBook reader, and so on. Thus, it is meaningful to develop RDH schemes for color images while keeping their gray version unchanged. For this purpose, Hou *et al.* [11] proposed the first grayscale-invariance RDH scheme for color images which embeds a secret message into a color image without changing the grayscale of the host image recently. Moreover, their result inspired several grayscale-invariant RDH works such as grayscale-invariant RDH in the domain of AMBTC compression [12]. Some RDH schemes for color images which keep different subjective visual perception invariant are also proposed [13].

In the grayscale-invariant RDH framework suggested in [11], it usually yields a correction bit when modifying

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

a pixel for data embedding. For the purpose of lossless recovery, the generated correction bits should be kept in the host image. In the RDH scheme of Hou *et al.* [11], the correction bits are embedded in the blue channel while the secret message is embedded in the red channel and the green scale is adjusted to keep the grayscale unchanged. Thus each pixel only takes at most one message bit. In order to enlarge the embedding capacity, Gao *et al.* [14] used the technique of adaptive embedding pattern to find many pixels in which one can embed more secret bits into their red scales. However, the embedding distortion of their scheme is still unsatisfactory. Recently, Zhou *et al.* [15] proposed a grayscale-invariant RDH scheme based on recursive code constructions (RCC). First, they approximated the modification-independent rate-distortion model for grayscale-invariant RDH and estimated optimal transition probability matrices. Based on the optimal transition probability matrices, they used the method of recursive histogram modification developed in [16], [17] to achieve lossless data hiding. The RCC-based RDH scheme proposed by Zhou *et al.* [15] has better performance than previous works. However, its embedding distortion is still not satisfactory when large embedding rate is required. Generally speaking, the main bottleneck is the large number of generated correction bits which are required to be accommodated in blue scales for image recovery.

In this paper, we propose a novel grayscale-invariant RDH algorithm which significantly reduces the need for correcting bits in the framework of Hou *et al.* [11]. Our key contribution is to design a self-correcting mechanism which is able to modify the green scales for making corresponding grayscales unchanged while keeping the original green scales in mind. To obtain this task, we first observe that the previous methods [11], [14], [15] generated correction bits in a functional approach which results in a large number of correction bits. Instead of the functional approach, we adopt a relational approach to keep the grayscales unchanged when embedding message bits. Following the relational approach, the proposed algorithm only needs to generate a small number of correction bits for lossless recovery. In addition, by the relational approach, the proposed algorithm also can embed message bits into some selected green scales. Experimental results show that, based on the above two advantages, our proposed grayscale-invariant RDH scheme obtains larger embedding capacity and has much better rate-distortion performance than previous works.

The rest of this paper is organized as follows. In Section II, the definitions of grayscale-invariant RDH schemes are given. Moreover, some related works are introduced there. In Section III, the self-correcting grayscale-invariant RDH scheme is proposed. Experimental results are given in Section IV. Finally, the conclusion and the future works are given in Section V.

II. GRAYSCALE-INVARIANT REVERSIBLE DATA HIDING SCHEMES AND RELATED WORKS

In this section, we briefly introduce reversible data hiding in color image with grayscale invariance. Let $X = \{X_{i,j}\} = \{\{R_{i,j}\}, \{B_{i,j}\}, \{G_{i,j}\}\} = (R, B, G)$ denote a color image. Among many transformations from a color image $X = (R, B, G)$ into a gray image $V = \{V_{i,j}\}$, the most widely used technique is the well-known formula from [18]: for each pair i, j,

$$V_{i,j} = f_v(R_{i,j}, B_{i,j}, G_{i,j}), \tag{1}$$

where f_v is defined by

$$f_v(r, b, g) = \lfloor 0.299r + 0.114b + 0.587g \rceil \tag{2}$$

and $\lfloor x \rceil$ denotes the nearest integer to x. For convenience, let us denote the generated grayscale image by $V = \{V_{i,j}\} = f_v(R, B, G)$.

Based on these notations, in the reversible data hiding framework for color images with gray invariance, the data hider losslessly embeds a secret message into a given color image $X = (R, B, G)$ and generates the marked image $X' = (R', B', G')$ such that $f_v(R, B, G) = f_v(R', B', G')$ and the distortion between X and X' is minimized.

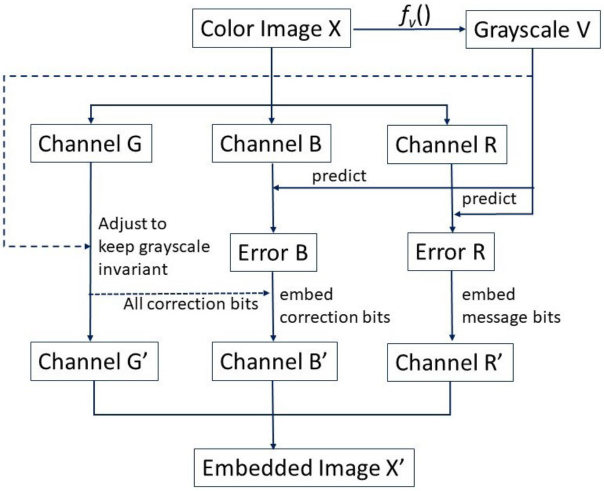


FIGURE 1. The reversible data hiding scheme of Hou *et al.* [11].

Hou *et al.* [11] proposed the first RDH scheme for solving the above problem. The flowchart of their proposed scheme is shown in Fig. 1. Their proposed scheme consists of two parts: a polynomial predictor to generate prediction errors and a data hiding scheme which embeds message bits into red-scales and correction bits into blue-scales, respectively.

A. THE POLYNOMIAL PREDICTOR BASED ON GRAY-SCALES

An image $X = (R, B, G)$ is given. The prediction errors of R and B are generated in a raster order. The prediction value of $R_{i,j}$ denoted as $\widehat{R}_{i,j}$ is generated as follows. The predictor computes the coefficient vector $[a, b, c]^T$ such that

$$\widehat{R}_{i,j} = a + bV_{i,j} + cV_{i,j}^2 \tag{3}$$

where

$$\begin{bmatrix} R_{i+1,j} \\ R_{i,j+1} \\ R_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} 1 & V_{i+1,j} & V_{i+1,j}^2 \\ 1 & V_{i,j+1} & V_{i,j+1}^2 \\ 1 & V_{i+1,j+1} & V_{i+1,j+1}^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}. \tag{4}$$

The prediction error of $R_{i,j}$ is defined by $e_{i,j}^R = R_{i,j} - \widehat{R}_{i,j}$. The prediction value $\widehat{B}_{i,j}$ and error $e_{i,j}^B$ of $B_{i,j}$ can be obtained in the same way.

B. REVERSIBLY ACCOMMODATING MESSAGES BY DIFFERENCE EXPANSION

One of the well-known RDH techniques is the so-called difference expansion developed by Tian [1]. The method reversibly accommodates message bits by expanding the generated prediction errors and embedding the message bit into the least significant bit of the expanded error. Specifically, the method embeds a message bit $b \in \{0, 1\}$ into the prediction error e by modifying it to e' as

$$e' = 2e + b. \tag{5}$$

From e', it is easy to recover e and b by computing $b = e' \mod 2$ and $e = \lfloor e'/2 \rfloor$.

C. A REVERSIBLE EMBEDDING ALGORITHM FOR MESSAGE AND CORRECTION BITS

Given v, r, b, let us define

$$f_g(r, b, v) = \lfloor (v - 0.299r - 0.114b)/0.587 \rceil. \tag{6}$$

Note that, if $g' = f_g(r, b, v)$, then $f_v(r, b, g') = v$. For each pixel $X_{i,j} = \{R_{i,j}, B_{i,j}, G_{i,j}\}$, the prediction errors $e_{i,j}^R$ and $e_{i,j}^B$ are generated firstly. To accommodate message bits, the algorithm uses Eq. 5 to expand $e_{i,j}^R$ and $e_{i,j}^B$ to $e'^R_{i,j}$ and $e'^B_{i,j}$, respectively. Next, $\{R_{i,j}, B_{i,j}\}$ are modified to $\{R'_{i,j}, B'_{i,j}\}$ where

$$\begin{cases} R'_{i,j} = \widehat{R}_{i,j} + e'^R_{i,j} \\ B'_{i,j} = \widehat{B}_{i,j} + e'^B_{i,j}. \end{cases} \tag{7}$$

The algorithm keeps the gray-scale $V_{i,j}$ unchanged in a functional approach. Precisely, the algorithm modifies the green-scale $G_{i,j}$ as

$$G'_{i,j} = f_g(R'_{i,j}, B'_{i,j}, V_{i,j}). \tag{8}$$

Finally the marked pixel $X'_{i,j} = \{R'_{i,j}, B'_{i,j}, G'_{i,j}\}$ is obtained.

From $X'_{i,j}$, the recipient calculates $\{\widehat{R}_{i,j}, \widehat{B}_{i,j}\}$ by using the polynomial predictor and gets $\{e'^R_{i,j}, e'^B_{i,j}\}$. Moreover, one can restore the embedded the messages bit, recover $\{e_{i,j}^R, e_{i,j}^B\}$, and then restore $\{R_{i,j}, B_{i,j}\}$ by calculating

$$\begin{cases} R_{i,j} = \widehat{R}_{i,j} + e_{i,j}^R \\ B_{i,j} = \widehat{B}_{i,j} + e_{i,j}^B. \end{cases} \tag{9}$$

To recover $G_{i,j}$, the recipient calculates

$$G_{i,j}^c = f_g(R_{i,j}, B_{i,j}, V_{i,j}) \tag{10}$$

where $V_{i,j}$ is calculated by $V_{i,j} = f_v(R'_{i,j}, B'_{i,j}, G'_{i,j})$.

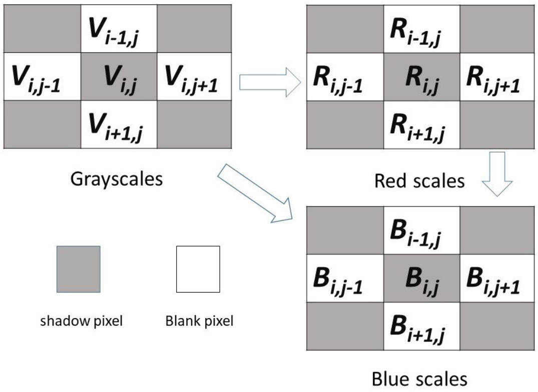


FIGURE 2. Rhombus predictor based on grayscales.

However, in general, $G_{i,j}^c$ is not equal to $G_{i,j}$ although their absolute difference $|G_{i,j}^c - G_{i,j}|$ is at most 1. For losslessly restoring $G_{i,j}$, the algorithm records the correction bit

$$c_{i,j} = |G_{i,j}^c - G_{i,j}| \tag{11}$$

and recursively embeds $c_{i,j}$ into the subsequent host pixels. In the scheme of Hou *et al.* [11], the correction bits $\{c_{i,j}\}$ are embedded into the prediction errors $\{e_{i,j}^B\}$ as shown in Fig. 1 while the secret message bits are embedded into $\{e_{i,j}^R\}$.

D. THE RHOMBUS PREDICTOR BASED ON GRAYSCALES

To generate a sharper host histogram of prediction errors, we use the rhombus prediction method of Zhou *et al.* [15] instead of the method of Hou *et al.* [11]. The rhombus prediction method was originally developed by Sachnev *et al.* [3]. However, their method is not suitable for grayscale-invariant setting. Zhou *et al.* [15] extended the rhombus method for the grayscale-invariant setting as follows. The whole cover image is divided into two groups denoted as "shadow" and "blank" as illustrated in Fig. 2. The first (second) half of the secret message will be embedded into the shadow (blank) pixels. Since the embedding procedures in these two layers are the same, we only explain the embedding procedure in the shadow part. The predicted value $\widehat{R}_{i,j}$ of the red scale $R_{i,j}$ is computed based on the linear regression method. Precisely, the predictor computes the coefficient vector $[a, b]^T$ such that

$$\widehat{R}_{i,j} = a + bV_{i,j} \tag{12}$$

where $[a, b]^T$ is a least-squares solution of the following linear system

$$\begin{bmatrix} R_{i-1,j} \\ R_{i,j-1} \\ R_{i,j+1} \\ R_{i+1,j} \end{bmatrix} = \begin{bmatrix} 1 & V_{i-1,j} \\ 1 & V_{i,j-1} \\ 1 & V_{i,j+1} \\ 1 & V_{i+1,j} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \tag{13}$$

The prediction error of $R_{i,j}$ is defined by $e_{i,j}^R = R_{i,j} - \widehat{R}_{i,j}$. Following the same way, the predicted value $\widehat{B}_{i,j}$ of the blue scale $B_{i,j}$ is also computed with only a few minor modifications described as follows. Let us define

$$V_{i,j}^- = \lfloor V_{i,j} - 0.299R_{i,j} \rceil. \tag{14}$$

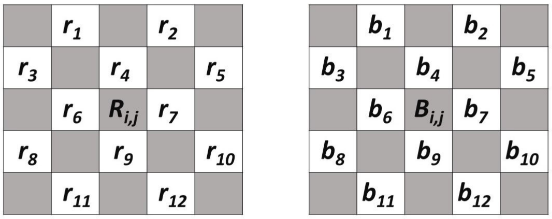


FIGURE 3. Locations of red and blue scales used for computing local variance.

The predicted value $\widehat{B}_{i,j}$ of the blue scale $B_{i,j}$ is computed by the following way. The predictor computes the coefficient vector $[a, b]^T$ such that $\widehat{B}_{i,j} = a + bV^-_{i,j}$ where $[a, b]^T$ is a least-squares solution of the following linear system

$$\begin{bmatrix} B_{i-1,j} \\ B_{i,j-1} \\ B_{i,j+1} \\ B_{i+1,j} \end{bmatrix} = \begin{bmatrix} 1 & V^-_{i-1,j} \\ 1 & V^-_{i,j-1} \\ 1 & V^-_{i,j+1} \\ 1 & V^-_{i+1,j} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \tag{15}$$

The prediction error of $B_{i,j}$ is defined by $e^B_{i,j} = B_{i,j} - \widehat{B}_{i,j}$.

Next, we determine the order of pixels for message embedding. There are several ways such as [19]–[21] to determine the order. In this paper, we propose the following method to determine the order of pixels for message embedding. The proposed method computes the local variance $\mathrm{LV}_{i,j}$ of the (i, j)-pixel in order to determine the order. As shown in Fig. 3, the local variance $\mathrm{LV}_{i,j}$ is computed by the following formula:

$$\mathrm{LV}_{i,j} = \sum_{k=1}^{12} |r_k - \widehat{R}_{i,j}| + |b_k - \widehat{B}_{i,j}|. \tag{16}$$

The embedding order is then determined according to the sorting order of the local variance $\mathrm{LV}_{i,j}$.

III. PROPOSED METHODS

In this section, we propose a novel embedding algorithm for reversible data hiding in color image with grayscale invariance. The proposed RDH scheme reduces a huge number of correction bits generated in the framework of Hou *et al.* [11]. To obtain this task, the main technique is a self-correcting mechanism which is able to modify the green scales for preserving the grayscales while remembering the information of the original green scales. This could be viewed a procedure to embed many correction bits into selected green scales as shown in Fig. 4. Now we introduce our proposed RDH algorithm in the following subsections.

A. A RELATIONAL APPROACH TO KEEP GRAYSCALES INVARIANT

Suppose that $X_{i,j} = (R_{i,j}, B_{i,j}, G_{i,j})$ is the currently selected pixel. We compute prediction errors $\{e^R_{i,j}, e^B_{i,j}\}$ and expand them to $\{e'^R_{i,j}, e'^B_{i,j}\}$ by using Eq. 5 in order to accommodate message bits. Then $\{R_{i,j}, B_{i,j}\}$ are modified to $\{R'_{i,j}, B'_{i,j}\}$

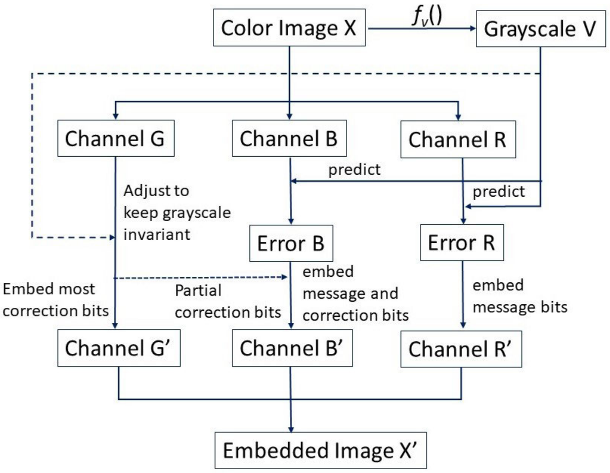


FIGURE 4. Flowchart of the proposed grayscale-invariant reversible data hiding scheme.

where

$$\begin{cases} R'_{i,j} = \widehat{R}_{i,j} + e'^R_{i,j} \\ B'_{i,j} = \widehat{B}_{i,j} + e'^B_{i,j}. \end{cases} \tag{17}$$

In order to keep the grayscale $V_{i,j}$ unchanged, we adopt a relational approach to modify the green scale instead of the functional approach used in previous works [11], [14], [15]. Precisely, we find a suitable green scale $\tilde{g}$ to satisfy the relation $f_v(R'_{i,j}, B'_{i,j}, \tilde{g}) = f_v(R_{i,j}, B_{i,j}, G_{i,j})$ and set $G'_{i,j} = \tilde{g}$ instead of directly computing $G'_{i,j} = f_g(R'_{i,j}, B'_{i,j}, V_{i,j})$. According to the relational approach, we consider the following procedure to determine $G'_{i,j}$.

Determining $G'_{i,j}$:

Input: $V_{i,j}, R_{i,j}, B_{i,j}, G_{i,j}, R'_{i,j}, B'_{i,j}$

$G'_{i,j}$ is obtained as follows.

1) (1-to-1 case) In this case, there is only one pair (g, g') such that

$$V_{i,j} = f_v(R_{i,j}, B_{i,j}, g) \tag{18}$$
$$= f_v(R'_{i,j}, B'_{i,j}, g'). \tag{19}$$

Clearly $g = G_{i,j}$. Now we set $G'_{i,j} = g'$.

2) (1-to-2 case) In this case, there are only two pairs (g, g'_0) and (g, g'_1) with $g'_0 < g'_1$ such that

$$V_{i,j} = f_v(R_{i,j}, B_{i,j}, g) \tag{20}$$
$$= f_v(R'_{i,j}, B'_{i,j}, g'_0) \tag{21}$$
$$= f_v(R'_{i,j}, B'_{i,j}, g'_1). \tag{22}$$

Clearly $g = G_{i,j}$. Now we can set $G'_{i,j}$ as either g'_0 or g'_1. Moreover, we can accommodate an additional secret bit b here by setting

$$G'_{i,j} = \begin{cases} g'_0 & \text{if } b = 0 \\ g'_1 & \text{if } b = 1 \end{cases} \tag{23}$$

3) (2-to-1 case) In this case, there are only two pairs (g_0, g') and (g_1, g') with $g_0 < g_1$ such that

$$V_{i,j} = f_v(R_{i,j}, B_{i,j}, g_0) \tag{24}$$

$$= f_v(R_{i,j}, B_{i,j}, g_1) \tag{25}$$

$$= f_v(R'_{i,j}, B'_{i,j}, g'). \tag{26}$$

We immediately set $G'_{i,j} = g'$. For lossless recovery, we need a correction bit $c_{i,j}$ to record the original $G_{i,j}$ by setting

$$c_{i,j} = \begin{cases} 0 & \text{if } g_0 = G_{i,j} \\ 1 & \text{if } g_1 = G_{i,j} \end{cases} \tag{27}$$

4) (2-to-2 case) In this case, there are four pairs (g_0, g'_0), (g_1, g'_0), (g_1, g'_0), and (g_1, g'_1) with $g_0 < g_1$ and $g'_0 < g'_1$ such that

$$V_{i,j} = f_v(R_{i,j}, B_{i,j}, g_s) \tag{28}$$

$$= f_v(R'_{i,j}, B'_{i,j}, g'_t) \tag{29}$$

for any $s, t \in \{0, 1\}$. For lossless recovery, we set $G'_{i,j}$ as

$$G'_{i,j} = \begin{cases} g'_0 & \text{if } g_0 = G_{i,j} \\ g'_1 & \text{if } g_1 = G_{i,j} \end{cases} \tag{30}$$

For better understanding the above procedure to determine $G'_{i,j}$, we illustrate the algorithm with some examples shown in Fig. 5.

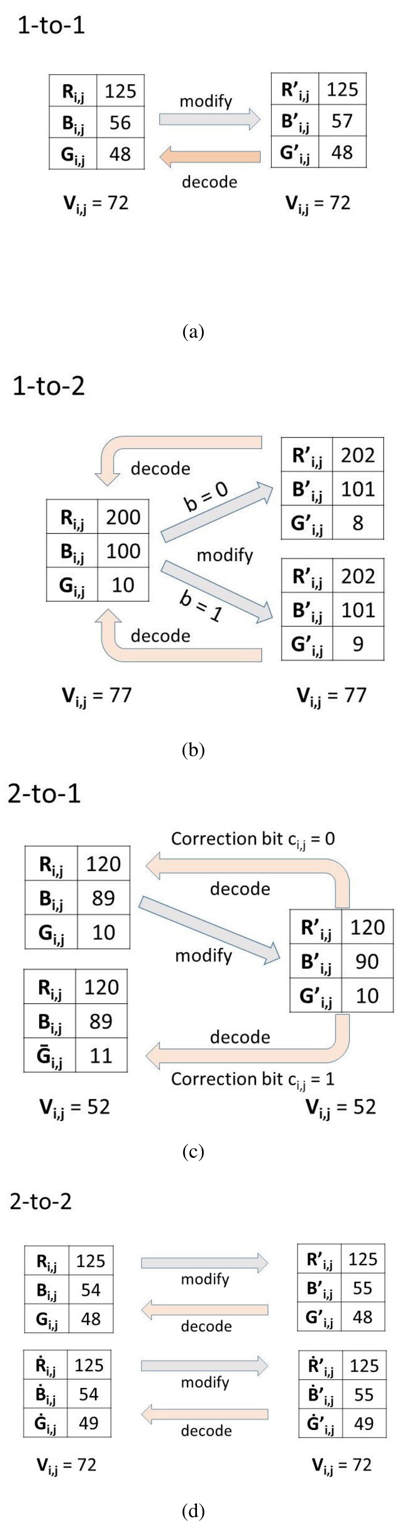
In Fig. 5(a), assume that $V_{i,j} = 72$, $R_{i,j} = 125$, $B_{i,j} = 56$, $R'_{i,j} = 125$, $B'_{i,j} = 57$. There is only one pair $(g, g') = (48, 48)$ such that $72 = f_v(125, 56, 48) = f_v(125, 57, 48)$. Clearly $G_{i,j} = 48$. Thus, $G'_{i,j} = 48$.

In Fig. 5(b), assume that $V_{i,j} = 77$, $R_{i,j} = 200$, $B_{i,j} = 100$, $R'_{i,j} = 202$, $B'_{i,j} = 101$. There are two pairs $(g, g'_0) = (10, 8)$ and $(g, g'_1) = (10, 9)$ such that $77 = f_v(200, 100, 10) = f_v(202, 101, 8) = f_v(202, 101, 9)$. Clearly $G_{i,j} = 10$. Then, $G'_{i,j} = 8$ if $b = 0$ and 9 if $b = 1$ where b is a message bit.

In Fig. 5(c), assume that $V_{i,j} = 52$, $R_{i,j} = 120$, $B_{i,j} = 89$, $R'_{i,j} = 120$, $B'_{i,j} = 90$. There are two pairs $(g_0, g') = (10, 10)$ and $(g_1, g') = (11, 10)$ such that $52 = f_v(120, 89, 10) = f_v(120, 89, 11) = f_v(120, 90, 10)$. In this case, we immediately set $G'_{i,j} = 10$. In addition, we need a correction bit $c_{i,j}$ to indicate which one is the original green scale. If the original green scales are 10 and 11 respectively, then $c_{i,j} = 0$ and 1 respectively.

In Fig. 5(d), assume that $V_{i,j} = 72$, $R_{i,j} = 125$, $B_{i,j} = 54$, $R'_{i,j} = 125$, $B'_{i,j} = 55$. There is only one quaternary vector $(g_0, g_1, g'_0, g'_1) = (48, 49, 48, 49)$ such that $72 = f_v(125, 54, 48) = f_v(125, 54, 49) = f_v(125, 55, 48) = f_v(125, 55, 49)$. For lossless recovery, we set $G'_{i,j} = 48$ if $G_{i,j} = 48$ and set $G'_{i,j} = 49$ if $G_{i,j} = 49$.

Note that some pixels for modification may yield overflow/underflow scales when embedding message bits. We skip them for modification and record them in the list *tag*. Precisely, if a pixel X_i causes an overflow/underflow scale after modification, then it is labelled with "$tag_i = 0$" otherwise it is labelled with "$tag_i = 1$".



(a)

(b)

(c)

(d)

FIGURE 5. Some examples for determining $G'_{i,j}$ when $V_{i,j}$, $R_{i,j}$, $B_{i,j}$, $G_{i,j}$, $R'_{i,j}$, and $B'_{i,j}$ are given. (a) 1-to-1 case, (b) 1-to-2 case, (c) 2-to-1 case, and (d) 2-to-2 case.

B. RESERVING SPACE FOR AUXILIARY PARAMETERS AND THE FINAL EMBEDDING ALGORITHM

First of all, we give an overview of how we store the auxiliary parameters. For lossless recovery, some particular pixels are reserved for recording auxiliary parameters. As in [11], we adopt the so-called invariant pixels to record the information. A pixel $X = (r, b, g)$ is called an invariant pixel if

$$f_v(r, 2\lfloor\frac{b}{2}\rfloor, g) = f_v(r, 2\lfloor\frac{b}{2}\rfloor + 1, g). \tag{31}$$

The auxiliary parameters will be embedded into the LSBs (least significant bits) of blue scales of selected invariant pixels. Note that the grayscales of the selected invariant pixels are kept unchanged after LSB substitutions of their blue scales.

Next, we embed the auxiliary parameters in the following way. For convenience, we introduce some notations. Let S_i be a fixed region and B_i^{in} be the ordered set of blue scales of the invariant pixels in S_i. Let S_0 be the initial fixed border region which provides enough space for storing some necessary information defined later. The auxiliary parameters are embedded as follows. First, we embed the pure message $\{m_1, m_2, \ldots, m_p\}$ in the non-border region (that is, the complement of S_0) according to the method described in Section III-A. Then we collect the pixels for modification during message embedding as a subset E. Let ρ be a fixed positive integer. Next, starting from $i = 0$ to $\lfloor |E|/\rho \rfloor$, we do the following:

1) We embed the LSBs of the blue scales in B_i^{in} into the pixels outside of $S_i \cup E$.
2) We collect these pixels for modification during embedding the LSBs as a subset L_i. Now we compress the bit string *tag* for the pixels in $E \cup L_i$ and concatenate some necessary side information including $|S_i|$, $|E \cup L_i|$, ending bits as the auxiliary information bits.
3) If the length of the auxiliary information bits is less than or equal to the length of LSBs of B_i^{in}, then we put them in the LSBs of blue scales in B_i^{in} and the embedding procedure is done. Otherwise, we extend S_i to a set S_{i+1} with $S_0 \subset S_{i+1} \subset S_0 \cup E$, $|S_{i+1} \backslash S_i| = \rho$, and redo Step 1 to Step 3 for S_{i+1}.

As a requirement for S_0, the border region S_0 should be large enough to accommodate the auxiliary parameters $|S_i|$ and $N = |E \cup L_i|$ for later extraction.

Now we give the detail of the final embedding algorithm. we assume that there are N_0 pixels $\{X_1, X_2, \ldots, X_{N_0}\}$ in the non-border region which are selected for accommodating message bits $\{m_1, m_2, \ldots, m_p\}$ in the sorting order according to their local variances. We illustrate the embedding process in Fig. 6. For the current pixel $X_i = (R_i, B_i, G_i)$, we embed message bits $\{m_1, m_2, \ldots, m_p\}$ according to the four cases described in Section III-A. First, we embed two message bits denoted as m_j and m_{j+1} into R_i and B_i respectively. By Eq. 5, we expand $\{e_i^R, e_i^B\}$ to $\{e'^R_i, e'^B_i\}$. After embedding m_j and m_{j+1}, we modify $\{R_i, B_i\}$ to $\{R'_i, B'_i\}$. According to the algorithm for determining G'_i in Section III-A, the embedding procedure is executed in one of the following four ways:

1) (1-to-1 case) There is a unique pair (G_i, G'_i) to make the grayscale V_i unchanged and thus no correcting bit is required for recovering G_i in this case.

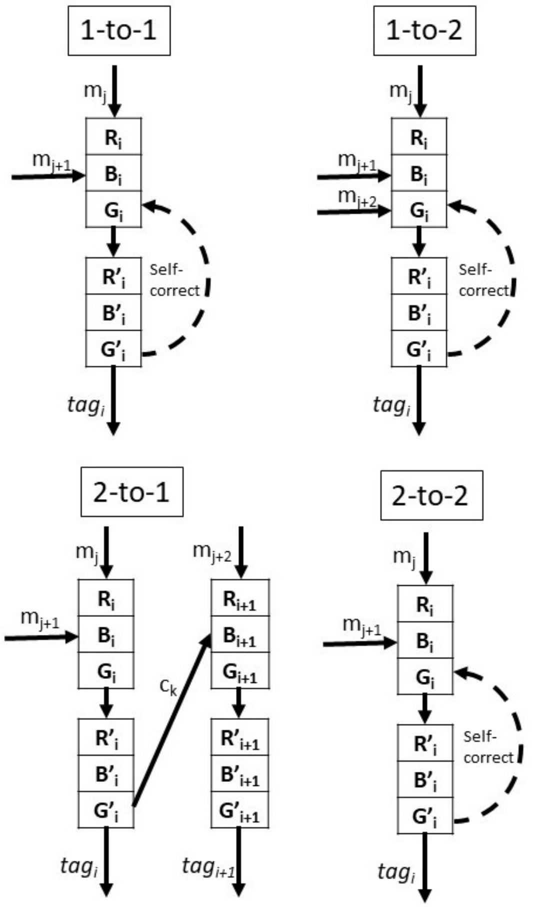


FIGURE 6. The embedding process of the proposed method.

2) (1-to-2 case) There are only two pairs (G_i, g'_0) and (G_i, g'_1) with $g'_0 < g'_1$ which make the grayscale V_i unchanged. Thus no correction bit is required for recovering G_i in this case. In addition, we can accommodate another message bit m_{j+2} by setting $G'_i = g'_{m_{j+2}}$.
3) (2-to-1 case) There are only two pairs (g_0, g') and (g_1, g') with $g_0 < g_1$ which make the grayscale V_i unchanged. One of g_0 and g_1 is G_i in this case. For lossless recovery, we need a correction bit c_k to indicate the original G_i by setting c_k to satisfy $G_i = g_{c_k}$. The correction bit c_k will be embedded into another pixel X_{i+1} as shown in Fig. 6.
4) (2-to-2 case) There are four pairs (g_0, g'_0), (g_1, g'_0), (g_1, g'_0), and (g_1, g'_1) with $g_0 < g_1$ and $g'_0 < g'_1$ which make the grayscale V_i unchanged. One of g_0 and g_1 is G_i in this case. For lossless recovery without correction bits, we set G'_i as

$$G'_i = \begin{cases} g'_0 & \text{if } g_0 = G_i \\ g'_1 & \text{if } g_1 = G_i \end{cases} \tag{32}$$

Next, we will check whether the modified pixel $X'_i = (R'_i, B'_i, G'_i)$ is overflow/underflow or not. If X'_i is overflow/underflow, then it is labelled by $tag_i = 0$ and skipped for modification. If X'_i is not overflow/underflow, then we label it with $tag_i = 1$. After embedding message bits $\{m_1, m_2, \ldots, m_p\}$ into the non-border region, we collect these pixels for modification as the set E. Note that $N_0 = |E|$.

Next, we find a set S_i with $S_0 \subset S_i \subset S_0 \cup E$ for some i which is described in the beginning of this subsection. Then we use the same embedding procedure to accommodate the LSBs of blue scales in B_i^{in}. Recall that L_i is the set of pixels which are used for embedding these LSBs. Finally, we get the location map tag for pixels in $E \cup L_i$ and compress it as a part of auxiliary parameters. The auxiliary parameters consist of the compressed string of tag, the cardinality $|S_i|$, and the number $N = |E \cup L_i|$ of pixels used for modification outside of the pixel border region S_0, and the two ending bits. The auxiliary parameters are embedded into the LSBs of the blue scales in B_i^{in}. This finishes the embedding procedure.

C. THE EXTRACTION ALGORITHM

The data extraction and image restoration are described as follows. When receiving the marked image X', the extracting algorithm locates the invariant pixels in the fixed region S_0, reads the LSBs of the blue scale of these pixels, and extracts the part of the auxiliary parameters including $|S_i|$, $N = |E \cup L_i|$, and the ending bits. Since $|S_i|$ is known, we can find the pixels in the set S_i and get the string tag. Next, since N is known, we list the first N pixels $\{X'_1, X'_2, \ldots, X'_N\}$ according to the sorting order of their local variances. Then we extract the embedded bits of the LSBs of blue scales in B_i^{in} in the reverse order.

Starting from the pixel $X'_N = \{R'_N, B'_N, G'_N\}$, we compute $\{e'^R_N, e'^B_N\}$, extract message bits, and restore $\{R_N, B_N\}$. In addition, we restore G_N directly since $(R_N, B_N, R'_N, B'_N, G'_N)$ does not belong to the 2-to-1 case. As a result, we restore the pixel X_N. In general, based on the location map tag_i, we can check if X'_i was modified in the embedding phase. If $tag_i = 0$, then we skip the pixel X'_i for restoring since $X'_i = X_i$. If $tag_i = 1$, then we compute $\{e'^R_i, e'^B_i\}$, extract the LSBs of pixels in B_i^{in}, and recover $\{R_i, B_i\}$. In addition, we recover G_i from the previously recovered embedded bits according to the following conditions:

1) (1-to-1 case) There is a unique pair (G_i, G'_i) to preserve the grayscale V_i. We just restore G_i directly.
2) (1-to-2 case) There are only two pairs (G_i, g'_0) and (G_i, g'_1) with $g'_0 < g'_1$ which can preserve the grayscale V_i. We just restore G_i directly. In addition, we extract the embedded bit b by calculating $g'_b = G'_i$.
3) (2-to-1 case) There are only two pairs (g_0, G'_i) and (g_1, G'_i) with $g_0 < g_1$ which can preserve the grayscale V_i. From the previous extracted correction bit c, we can restore G_i by computing $G_i = g_c$.
4) (2-to-2 case) There are four pairs (g_0, g'_0), (g_1, g'_0), (g_1, g'_0), and (g_1, g'_1) with $g_0 < g_1$ and $g'_0 < g'_1$ which can preserve the grayscale V_i. One of g'_0 and g'_1 is G'_i in this case. We restore G_i by

$$G_i = \begin{cases} g_0 & \text{if } g'_0 = G'_i \\ g_1 & \text{if } g'_1 = G'_i \end{cases} \tag{33}$$

Based on the above extraction step and the knowledge of S_i, we can retrieve the LSBs of blue scales in B_i^{in} and thus restore the original marked pixels in the set E. Now we execute the above extraction procedure for the marked pixels in E in the reverse order to get the message bits $\{m_1, m_2, \ldots, m_p\}$ and restore the host image X.

IV. EXPERIMENTAL RESULTS

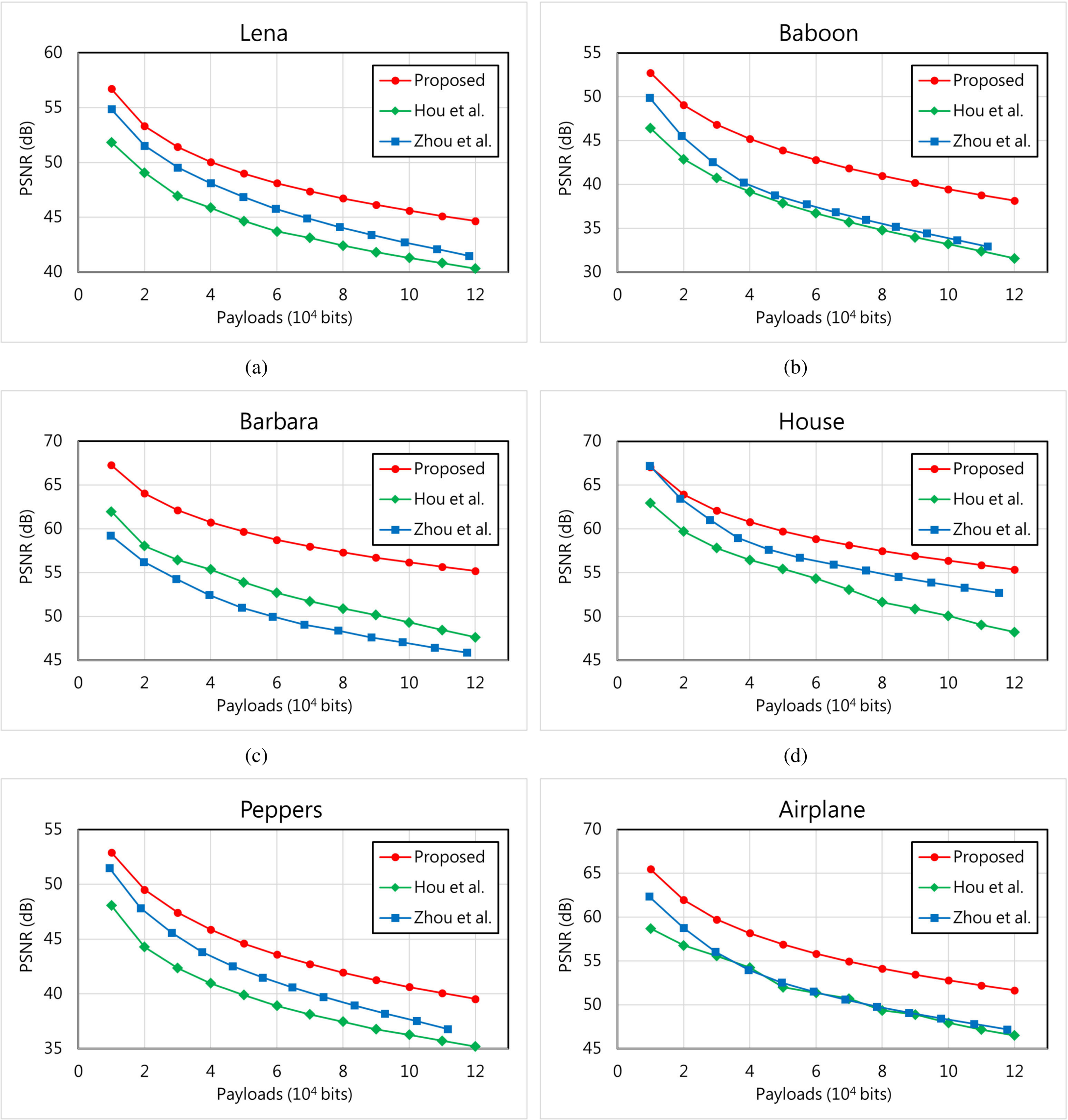
A. SELF-CORRECTING ABILITY AND ADDITIONAL EMBEDDING CAPACITY OF THE PROPOSED RDH SCHEME

First, we demonstrate the self-correcting ability of the proposed method with the following experimental results. Six standard 512 × 512 color images including Lena, Baboon, Airplane, Barbara, Peppers and House selected from [22] are used in our experiment. The self-correcting ability of a given algorithm could be quantified by the number of the correction bits generated by the given algorithm. The proposed algorithm adopts a relational approach to reduce the number of correction bits. In fact, it only generates correction bits for the pixels in the 2-to-1 case described in Section III-A.

In Table 1, the rate of each case with respect to the number of total pixels for modification with embedding capacity 50000 bits and 100000 bits is given. Each rate is obtained by executing the embedding procedure ten times and averaging their results. On average, 23 percent of the pixels for modification belongs to the 2-to-1 case. Thus the proposed method reduces 77 percentages on the number of correction bits. In addition, the proposed method also increases embedding capacity by accommodating message bits in the green scales of the pixels in the 1-to-2 case. On the average, 14 percent of the pixels for modification belongs to the 1-to-2 case. Therefore, the proposed method increases additional embedding capacity by 14 percent in this case.

We also statistically demonstrate the self-correcting ability by using 300 color images from the dataset in [23]. The average result is shown in Table 2. The average rate and the variance of each case with respect to the number of total pixels for modification with embedding capacity 50000 bits and 100000 bits are given. On average, 24.2 percent of the pixels for modification belongs to the 2-to-1 case. Thus, 75.8 percentage on the number of correction bits is reduced by the proposed algorithm. Moreover, on the average, 12.5 percent of the pixels for modification belongs to the 1-to-2 case. Therefore, the proposed algorithm increases additional embedding capacity by 12.5 percent on average. Since the variance of each case is small, this means that the proposed algorithm has nice self-correcting ability for most images from the dataset in [23].

In short, the proposed algorithm not only decreases the number of generated correction bits for the modified green scales but also increases additional space of green scales for



(a)

(b)

(c)

(d)

(e)

(f)

FIGURE 7. Rate-distortion performance comparison between the method of Hou *et al.* [11], Zhou *et al.* [15], and the proposed method.

further message embedding. Note that reducing the number of correction bits implies that one can accommodate more message bits in the blue scales. Thus, based on the statistical result on the dataset in [23], the proposed RDH scheme has effect on increasing embedding capacity by 88.3 percent on the average compared to the framework of Hou *et al.* [11].

B. RATE-DISTORTION PERFORMANCE

In this subsection, we compare the proposed scheme with the state-of-the-art grayscale-invariant RDH schemes including the method of Hou *et al.* [11] and the method of Zhou *et al.* [15]. Hou *et al.*'s method [11] used the polynomial predictor and a functional approach to keep grayscales unchanged. A lot of correction bits are generated by their method and the distortion is not well controlled because their algorithm embeds message bits by using the difference expansion method. Zhou *et al.*'s method [15] uses the rhombus predictor to sharpen the host histogram of prediction errors and minimizes the total distortion by using the rate-distortion optimization method. However, their scheme still generates a large number of correction bits since their scheme also uses the functional approach to keep grayscales invariant. By using a relational approach to make gray-scales invariant, the proposed method reduces quite a lot of correction bits and finds out more space of green scales for embedding message bits.

As shown in Fig. 7, the rate-distortion performance of the proposed method is much better than that of Hou *et al.*'s method [11] and Zhou *et al.*'s method [15]. In particular, the proposed method achieves high image quality when the

TABLE 1. Self-correcting ability of the proposed method with capacity of 50000 bits and 100000 bits, respectively.

Image	EC=50000 bits		EC=100000 bits	
	Case	Rate	Case	Rate
Lena	1-to-1	5.7%	1-to-1	5.6%
	1-to-2	12.5%	1-to-2	12.3%
	2-to-1	23.0%	2-to-1	23.1%
	2-to-2	58.8%	2-to-2	59.0%
Baboon	1-to-1	5.2%	1-to-1	5.2%
	1-to-2	12.3%	1-to-2	12.3%
	2-to-1	24.3%	2-to-1	24.3%
	2-to-2	58.2%	2-to-2	58.2%
Airplane	1-to-1	7.0%	1-to-1	6.2%
	1-to-2	15.7%	1-to-2	14.2%
	2-to-1	22.8%	2-to-1	23.3%
	2-to-2	54.5%	2-to-2	56.3%
Peppers	1-to-1	6.1%	1-to-1	6.2%
	1-to-2	12.1%	1-to-2	12.1%
	2-to-1	24.3%	2-to-1	24.5%
	2-to-2	57.5%	2-to-2	57.2%
Barbara	1-to-1	10.1%	1-to-1	10.1%
	1-to-2	20.6%	1-to-2	21.4%
	2-to-1	20.2%	2-to-1	19.9%
	2-to-2	49.1%	2-to-2	48.6%
House	1-to-1	5.1%	1-to-1	5.1%
	1-to-2	11.6%	1-to-2	11.7%
	2-to-1	23.4%	2-to-1	23.4%
	2-to-2	59.9%	2-to-2	59.8%
Average	1-to-1	6.53%	1-to-1	6.4%
	1-to-2	14.13%	1-to-2	14.0%
	2-to-1	23.0%	2-to-1	23.08%
	2-to-2	56.33%	2-to-2	56.52%

TABLE 2. The average result of self-correcting ability on the dataset in [23] with capacity of 50000 bits and 100000 bits, respectively.

Images from the dataset in [23]	EC=50000 bits		EC=100000 bits	
	Case	Rate	Case	Rate
Mean	1-to-1	5.4%	1-to-1	5.3%
	1-to-2	12.5%	1-to-2	12.5%
	2-to-1	24.2%	2-to-1	24.2%
	2-to-2	57.9%	2-to-2	58.0%
Variance	1-to-1	5.235×10^{-5}	1-to-1	2.141×10^{-5}
	1-to-2	3.037×10^{-4}	1-to-2	1.049×10^{-4}
	2-to-1	5.998×10^{-5}	2-to-1	2.428×10^{-5}
	2-to-2	3.106×10^{-4}	2-to-2	1.142×10^{-4}

embedding capacity is large. Therefore, our proposed algorithm not only has good self-correcting ability but also obtains nice rate-distortion performance.

C. EMBEDDING TIME COMPARISON

TABLE 3. Embedding time comparison between the method of Hou *et al.* [11] and the proposed method.

Image	Lena	Baboon	Barbara	House	Peppers	Airplane
[11]	0.78	5.22	0.42	0.41	4.96	0.42
Proposed	2.26	4.43	0.98	0.98	7.06	0.98

For embedding time complexity, we compare the proposed algorithm with the method of Hou *et al.* [11] in order to understand the difference between the functional and relational approaches for grayscale-invariant RDH. Two methods are implemented in Visual Studio 2019, and the test machine is a Lenovo personal computer with an i5-6400 CPU @ 2.70 GHz and 16.00 GB of RAM. Both methods embed 50000 bits into six images and their embedding times (measured in seconds) are shown in Table 3. From Table 3, we observe that the proposed algorithm has larger embedding time than the method of Hou *et al.* [11] for most images since the proposed method requires more time to determine the modified green-scale than the method of Hou *et al.* [11]. Moreover, for images with rich texture such as Baboon and Peppers, we observe that both algorithms are time-consuming since the prediction errors generated by two algorithms are so large that the algorithms require a lot of time to handle pixel value overflow and underflow.

V. CONCLUSION AND FUTURE WORKS

In this paper, we study grayscale-invariant RDH schemes for color images. Previous methods used a functional approach to keep grayscales unchanged. However, a large number of correction bits are generated by their embedding algorithms. To address this problem, the proposed grayscale-invariant RDH algorithm adopts a relational approach to keep grayscales unchanged. The proposed RDH scheme not only significantly reduces the number of correction bits but also increases additional space for embedding message bits. The experimental result shows that the proposed RDH scheme significantly outperforms previous methods.

We have two future works. Following Zhou *et al.*'s approach [15], our first future work is to apply the rate-distortion optimization method to further minimize the total embedding distortion of the grayscale-invariant RDH schemes which adopt the relational approach to preserve grayscales. Our second future work is to study whether the histogram-shifting technique can be used for grayscale-invariant RDH.

REFERENCES

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

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] X. Hu, W. Zhang, X. Li, and N. Yu, "Minimum rate prediction and optimized histograms modification for reversible data hiding," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 3, pp. 653–664, Mar. 2015.

[7] C.-C. Chang, C.-Y. Lin, and Y.-H. Fan, "Lossless data hiding for color images based on block truncation coding," *Pattern Recognit.*, vol. 41, no. 7, pp. 2347–2357, 2008.

[8] W.-J. Yang, K.-L. Chung, and H.-Y. M. Liao, "Efficient reversible data hiding for color filter array images," *Inf. Sci.*, vol. 172, pp. 1–11, Jul. 2020.

[9] 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, 2013.

[10] 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.

[11] 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.

[12] R. Kumar, N. Kumar, and K.-H. Jung, "Color image steganography scheme using gray invariant in AMBTC compression domain," *Multidimensional Syst. Signal Process.*, vol. 31, no. 3, pp. 1145–1162, Jul. 2020.

[13] Y. Yang, T. Zou, G. Huang, and W. Zhang, "A high visual quality color image reversible data hiding scheme based on B-R-G embedding principle and CIEDE2000 assessment metric," *IEEE Trans. Circuits Syst. Video Technol.*, early access, May 28, 2021, doi: 10.1109/TCSVT.2021.3084676.

[14] E. Gao, Z. Pan, and X. Gao, "Adaptive embedding pattern for grayscale-invariance reversible data hiding," 2019, *arXiv:1908.05965*. [Online]. Available: http://arxiv.org/abs/1908.05965

[15] S. Zhou, W. Zhang, and C. Shen, "Rate-distortion model for grayscale-invariance reversible data hiding," *Signal Process.*, vol. 172, pp. 1–11, Jul. 2020.

[16] 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.

[17] W. Zhang, X. Hu, X. Li, and N. Yu, "Optimal transition probability of reversible data hiding for general distortion metrics and its applications," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 294–304, Jan. 2015.

[18] (Dec. 2017). *ITU-R Recommendation BT.601*. [Online]. Available: https://en.wikipedia.org/wiki/Rec._601

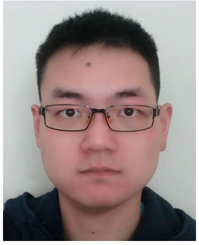
[19] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 7, pp. 1091–1100, Jul. 2013.

[20] 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, Jun. 2015.

[21] A. Mohammad and M. Nakhkash, "Sorting methods and adaptive thresholding for histogram based reversible data hiding," *Multimedia Tools Appl.*, vol. 80, no. 3, pp. 3307–3325, 2021.

[22] *The USC-SIPI Image Database*. Accessed: Apr. 10, 2021. [Online]. Available: http://sipi.usc.edu/database/

[23] M. Ponomarenko, N. Gapon, V. Voronin, and K. Egiazarian, "Blind estimation of white Gaussian noise variance in highly textured images," 2017, *arXiv:1711.10792*. [Online]. Available: http://arxiv.org/abs/1711.10792



CHUN-LIANG JHONG received the B.S. degree from National Taipei University, New Taipei City, Taiwan, in 2019, where he is currently pursuing the master's degree with the Department of Computer Science and Information Engineering. His research interests include reversible data hiding, algorithm design, and machine learning.

HSIN-LUNG WU received the Ph.D. degree in computer science and information engineering from National Chiao Tung University, Taiwan, in 2008. He is currently an Assistant Professor with the Department of Computer Science and Information Engineering, National Taipei University, Taiwan. His main research interests include design and analysis of algorithms, computational complexity, theory of machine learning, and deep learning.