Reversible Data Hiding in Encrypted Images Using Global Compression of Zero-Valued High Bit-Planes and Block Rearrangement

Ye Yao, Ke Wang, Qi Chang, and Shaowei Weng, *Member, IEEE*

***Abstract*—Recently, reversible data hiding in encrypted images (RDHEI) has received widespread attention from researchers. To embed high payload into encrypted images while maintaining sufficient security, a novel RDHEI algorithm in combination with consecutive zero-valued high bit-planes compression, bit-plane swapping as well as block rearrangement is proposed in this article. The proposed method is the first work to compress global zero-valued high bit-planes in a block-wise manner and adaptively allocate different Huffman indicators based on the occurrence frequency of zero-valued bit-planes so that a higher embedded payload is greatly provided. Unlike existing RDHEI methods embedded with unencrypted auxiliary information, resulting in low security, the bit-plane swapping and block rearrangement are subtly designed to cluster together all embeddable bit-planes, which enables most auxiliary information to be encrypted, largely enhancing the security and facilitating data embedding and data extraction. The experiment results demonstrate that the proposed method outperforms some state-of-the-art RDHEI methods in terms of security and payload. The average payload of the proposed method for two publicly-used datasets including BOSSbase and BOWS-2, are 3.793 bpp and 3.705 bpp, respectively.**

***Index Terms*—Block rearrangement, encrypted image, privacy preservation, reversible data hiding, zero-valued high bit-planes compression.**

I. Introduction

REVERSIBLE data hiding (RDH) is a technique that hides secret data in images and is able to recover the original images without any distortion after extracting embedded data when needed. RDH is primarily split into four types, i.e., difference expansion [1], [2], [3], histogram shifting (HS) [4], [5], prediction-error (PE) expansion [6], [7], [8], and multiple histograms modification [9], [10], [11], [12]. All these methods aim to improve the embedding capacity, reduce the embedding distortion and recover the image losslessly.

RDH plays a vital role in medical image processing [13], military multimedia archive management [14], and image transcoding [15], etc., thus attracting the extensive attention of researchers. In cloud applications, after the cloud server gets the user's transmitted encrypted image files, it needs to embed some additional information, such as timestamp, copyright, etc., into images. RDH is an efficient solution and RDH in encrypted images (RDHEI) has gradually attracted widespread research interest.

Current RDHEI methods are mainly divided into three categories, including reserving room before encryption (RRBE) [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], vacating room by encryption (VRBE) [30], [31], [32], [33] and vacating room after encryption (VRAE) [34], [35], [36]. The RRBE methods require the image owner to vacate enough redundant room for the plaintext image by making use of the correlation between pixels so that the data hider can embed secret data. The VRBE methods encrypt the image while performing room-vacation operations, and the correlation between image pixels will be partially maintained. Different from RRBE and VRBE methods, VRAE methods encrypt a given cover image and then embed additional data by modifying the encrypted pixels. The correlation between the encrypted image pixels will be destroyed, so it is challenging to realize data embedding and image recovery. This article focuses on proposing an RRBE method for large payloads and high security.

Since Ma et al. [16] proposed a HS-based RRBE, in which the released room in the original image is vacated and used by the data hider, a large number of related methods have been proposed. Among them, Zhang et al. [17] proposed to use an estimation technique that estimates a fraction of pixels via a large portion of pixels in the original image and secret data can be embedded by modifying the estimation errors. Cao et al. [18] vacated room for embedding data by using the patch-level representation. The original image is represented by sparse coefficients and the generated residual errors are encoded to reserve a large amount of embedding room. Then a new RRBE-based method which marks the most significant bit (MSB) of the original image was proposed by Puteaux et al. [19], the non-embeddable pixels are recorded by the error label map, and the additional data is embedded into the MSB of embeddable pixels. On the

Manuscript received 17 May 2023; revised 29 July 2023 and 1 September 2023; accepted 5 September 2023. Date of publication 13 September 2023; date of current version 14 February 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 62302135, 62262062, and 72374058 and in part by the Fujian Science Fund for Distinguished Young Scholars under Grant 2020J06043. The Associate Editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaochun Cao. *(Corresponding author: Qi Chang.)*

Ye Yao, Ke Wang, and Qi Chang are with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, Zhejiang 310018, China (e-mail: yaoye@hdu.edu.cn; wang_ke517@163.com; qichang@hdu.edu.cn).

Shaowei Weng is with the School of Electronic, Electrical Engineering and Physics, Fujian University of Technology, Fuzhou, Fujian 350108, China (e-mail: wswweiwei@126.com).

Digital Object Identifier 10.1109/TMM.2023.3314975

basis of [19], Puyang et al. [20] proposed an improved method by using two MSBs to mark the pixels of the original image. Another method based on a parametric binary tree labeling was proposed by Yi et al. [21], the parameter is used to determine which pixels in the encrypted image can be embedded data. To further increase the capacity of cover images, Chen et al. [22] designed a joint scheme of extended run-length coding. The bit-planes are rearranged and the vacated room is used to store the additional data over multi least significant bit (LSB) substitution. By exploiting the overall spatial correlation of adjacent pixels in the original image, Wu et al. [23] marked the PEs in the entire original image to vacate more room. Soon after, Yin et al. [24] compressed the successive same high bit-planes between the original pixels and their prediction values by Huffman coding. The generated compressed auxiliary information is embedded into the encrypted image. In [25], Mohammadi et al. divided the original image into non-overlapping blocks, each with a leader pixel used to calculate PEs with other pixels. This approach achieves a high embedding payload by determining the embedding capacity of blocks one by one and vacating the room before encryption. Moreover, a new method based on bit-plane compression was proposed by Yin et al. [26], the original image is divided into equal-sized blocks and the 8 b-planes of PE are executed bit-stream compression to vacate the room. In [27], Yu et al. introduced a new RDHEI method with hierarchical embedding, where PEs are divided into small-magnitude, medium-magnitude, and large-magnitude, all used to accommodate secret bits, contributing to high embedded capacity. Xu et al. [28] proposed a new hierarchical block variable length coding technique in their scheme, the bit-plane into PE blocks can be adaptively decompressed to different hierarchical levels based on local smoothness, and the block will be encoded with a variable length coding method. In this way, the carrier image can be effectively compressed to provide more embedding space for data embedding. To further solve dissatisfied payload and insecurity in Yin et al.'s scheme [24], Gao et al. [29] proposed an adaptive Huffman encoding algorithm to shorten the total length of auxiliary information. Besides, the way of image scrambling is used to improve the security of final marked images.

However, many of the RRBE-based methods mentioned above achieve high payloads but tend to neglect image security. Although Gao et al.'s scheme [29] discussed the security, it increases tediousness and is not conducive to data extraction and image recovery. To address the need for both high payloads and image security while ensuring reversible image recovery, this article proposes a novel RRBE method that combines global compression of zero-valued high bit-planes and block rearrangement. Unlike previous schemes such as Yin et al.'s [24] and Gao et al.'s [29], which involve comparing multi-MSB of original pixel values and predicted values, and consequently requiring each pixel to be recorded with corresponding tag value information, our approach focuses on comparing the adjacent PEs of non-overlapping blocks. By compressing multi-MSB zero-valued bit-planes using Huffman coding, we effectively reduce the length of auxiliary information, freeing up more embedding space and achieving higher payload capacity. Additionally, we analyze the relationship between the number of '0' and '1' in the uncompressed bit-planes to further optimize the payload. To ensure sufficient security, our proposed method incorporates a flexible approach to bit-plane swapping and block rearrangement. This design allows almost all auxiliary information to be encrypted, thus mitigating the risk of content leakage from the cover image.

The main contributions of this article are the following:

1) *High Zero-Valued Bit-Plane Compression in the Block-Wise Manner:* Different from existing RRBE methods compressing bit-planes of PEs in a pixel-wise manner, we propose to compress global zero-valued high bit-planes of PEs in a block-wise manner for largely improving the payload.
2) *Bit-Plane Swapping and Block Rearrangement:* Clustering available bit-planes in a block together by carrying out the bit-plane swapping and rearranging all the blocks in the descending of their respective payload are proposed so that only a small portion of auxiliary information is needed for data embedding while a large portion is encrypted to enhance the security while facilitating data embedding and extraction.
3) *Huffman Indicators:* We adaptively allocate different Huffman indicators based on the occurrence frequency of high zero-valued bit-planes so as to further reduce the length of auxiliary information.

The rest of this article is organized as follows. Section II introduces three related works based on pixel prediction. The proposed RRBE-based method using global compression of zero-valued high bit-planes and block rearrangement is described in Section III. Then, the experimental results including security analysis and embedding performance comparison are given in Section IV. Finally, Section V concludes this article.

II. Related Works

In this Section, three related works are introduced in detail, including Yin et al.'s scheme [24], Mohammadi et al.'s scheme [25] and Yu et al.'s scheme [27]. Yin et al. [24] marked and embedded data on the same MSBs between original pixels and predicted pixels. Mohammadi et al. [25] considered the center pixel in the block as the predicted value of other pixels and calculated the embedding amount based on the PE range. Yu et al. [27] marked embeddable bits of the pixel as labels according to PE, and the bit labels are compressed to be embedded into the encrypted images for assisting data extraction and image recovery.

A. Yin et al.'s Scheme [24]

For an original image I with a size of $m \times n$, the predicted value $p(i,j)$ of each pixel $x(i,j)$ is calculated by using the median edge detector (MED) predictor [37], in which $1 \leq i \leq m$ and $1 \leq j \leq n$. Convert $x(i,j)$ and $p(i,j)$ into 8-bit binary sequences $\{x^k(i,j)\}_{k=1}^{8}$ and $\{p^k(i,j)\}_{k=1}^{8}$ by (1).

$$x^k(i,j) = \left\lfloor \frac{|x(i,j)| \bmod 2^{9-k}}{2^{8-k}} \right\rfloor \tag{1}$$

Compare each layer of bits of $x^k(i,j)$ and $p^k(i,j)$ in order from MSB to LSB until the two compared bits are not the same, setting the label l of the current pixel equal to the length of the compared identical bits. The label l has nine cases of values from 0 to 8 and its values show that the front l MSBs of the current pixel are the same as the predicted pixel while the $(l+1)^{th}$ MSB are different, so only the remaining $(8-1-l)$ LSBs sequence T and label l need to be stored as auxiliary information. The original pixel value $x(i,j)$ can be recovered from the label l, sequence T, and the predicted value $p(i,j)$. In order to reduce the storage cost of the all label l effectively, the image owner defines 9 kinds of Huffman codes to replace the label, namely $\{00, 01, 100, 101, 1100, 1101, 1110, 11110, 11111\}$. The more frequently the label l appears, the shorter Huffman code is selected to represent it. After the original image I is encrypted by exclusive-or (XOR) operation using an encryption key K_e which is used to generate a pseudo-random matrix, the Huffman encoding rules, the length of the binary sequence of label map, and the label map will first be embedded into the encrypted image. Before data embedding, the data is encrypted in advance by the data hiding key K_d to ensure security. After obtaining all the auxiliary information and restoring the label l based on Huffman encoding rules, the encrypted data is embedded in the remaining room.

In the data extraction and image recovery phase, the Huffman encoding rules and label l are extracted from the marked image by the receiver first. Then the encrypted data can be extracted based on the label l of every pixel. The receiver will decrypt the extracted data and reconstruct the original image through K_e and K_d, respectively.

B. Mohammadi et al.'s Scheme [25]

An original image is divided into non-overlapping blocks and each block is the same size. The center pixel in each block is the leader pixel and the other pixels are the follower pixels. The absolute difference value between every follower pixel and the leader pixel is computed, and the minimum value $|e|$ of each block is obtained. Each block will free up embedding space if the condition of $|e|$ satisfies

$$|e| < 2^c, \quad 0 \le c \le 7 \tag{2}$$

The embedded capacity of every follower pixel in the block denoted as q bits is given by

$$q = \begin{cases} 8-c-1, & c \neq 0 \\ 8, & c = 0 \end{cases} \tag{3}$$

Assuming that the value of the follower pixel is defined as $p = (p_7 p_6 p_5 p_4 p_3 p_2 p_1 p_0)_2$ with 8 bits from MSB (p_7) to LSB (p_0) and the data of q bits to be embedded is $D = (d_0, d_1, \ldots, d_{q-1})_2$, the marked pixel p' containing data D is constructed by the following:

$$p' = \sum_{i'=1}^{q} \left(2^{8-i'} \times d_{i'-1}\right) + \sum_{i'=q+1}^{8} \left(p_{8-i'} \times 2^{8-i'}\right) \tag{4}$$

In order to extract the correct data and recover the original image losslessly, $|e|$ of each block needs to be embedded in the encrypted image as auxiliary information. In the data extraction and image recovery stage, when $|e|$ as well as q of every block are known in advance, the receiver can extract data D correctly or restore the original follower pixel p by different keys.

C. Yu et al.'s Scheme [27]

In *Yu et al.*'s scheme, the pixel p and its predicted value $\hat{p}$ generated by MED predictor [37] are used to embed secret data. Similar to Mohammadi et al.'s scheme [25], convert p and $\hat{p}$ into 8-bit binary sequences $(p_8 p_7 p_6 p_5 p_4 p_3 p_2 p_1)_2$ and $(\hat{p}_8 \hat{p}_7 \hat{p}_6 \hat{p}_5 \hat{p}_4 \hat{p}_3 \hat{p}_2 \hat{p}_1)_2$ from MSB to LSB, respectively and the absolute value of the difference between p and $\hat{p}$ is defined as $|e|$. If $0 \le |e| \le 2^{q-1}-1$, the $(q+1)^{th} \sim 8^{th}$ bits of p can be marked by $(8-q)$ labels '$0, 0, \ldots, 0$' and embedded in $(8-q)$ secret bits. If $|e| = 2^{q-1}$, the $(q+1)^{th} \sim 8^{th}$ bits of p are marked by $(8-q)$ labels '$2, 0, \ldots, 0$' and substituted with $(8-q-1)$ secret bits, the label '2' remains unchanged and cannot store a secret bit. Each embeddable pixel will have its labels and the sign of e, which will be compressed and stored in the encrypted image for data extraction and image recovery. For reversibility, the labels are first extracted from the marked image. If there are $(8-q)$ labels '$0, \ldots, 0$', the original pixel p can be restored according $0 \le |e| \le 2^{q-1}-1$. Similarly, the labels '$2, 0, \ldots, 0$' mean that $|e| = 2^{q-1}$, and p can be easily recovered by $\hat{p}$ and the sign of e.

III. Proposed Method

In this article, we propose a novel RRBE-based method with high embedded capacity and enough security by means of global zero-valued high bit-planes compression, bit-plane swapping and block rearrangement. It is obvious in Fig. 1 that our proposed method contains three phases: 1) Vacating room and image encryption are done by the content-owner; 2) The data hider embeds secret data into the encrypted image; 3) Data extraction and image recovery are performed by the receiver. In the first stage, to vacate the room, the consecutive zero-valued bit-planes and remaining available bit-planes in PE blocks are fully compressed in a block-wise manner. Then, to enhance the security and facilitate data embedding or extraction, we apply the proposed way of bit-plane swapping and block rearrangement to cluster embeddable bit-planes, and most auxiliary information will be encrypted. We also adaptively use Huffman encoding to improve the payload. In the second stage, the encrypted data is embedded in the vacated room generated in the first stage. The receiver extracts data and reconstructs the original image with different keys in the last stage. Modules of the same color in the diagram represent reversible processes.

A. Bit-Plane Compression

1) PE Calculation: For an 8-bit cover image $\mathcal{I}$ of size $m \times n$, the pixel located at the ith row and jth column is denoted as $\mathcal{I}(i,j)$ ($\mathcal{I}(i,j) \in$ [0,255]). For the pixel $\mathcal{I}(i,j)$ with $1 \le i \le m$ and $1 \le j \le n$, the predicted value $p(i,j)$ is constructed using

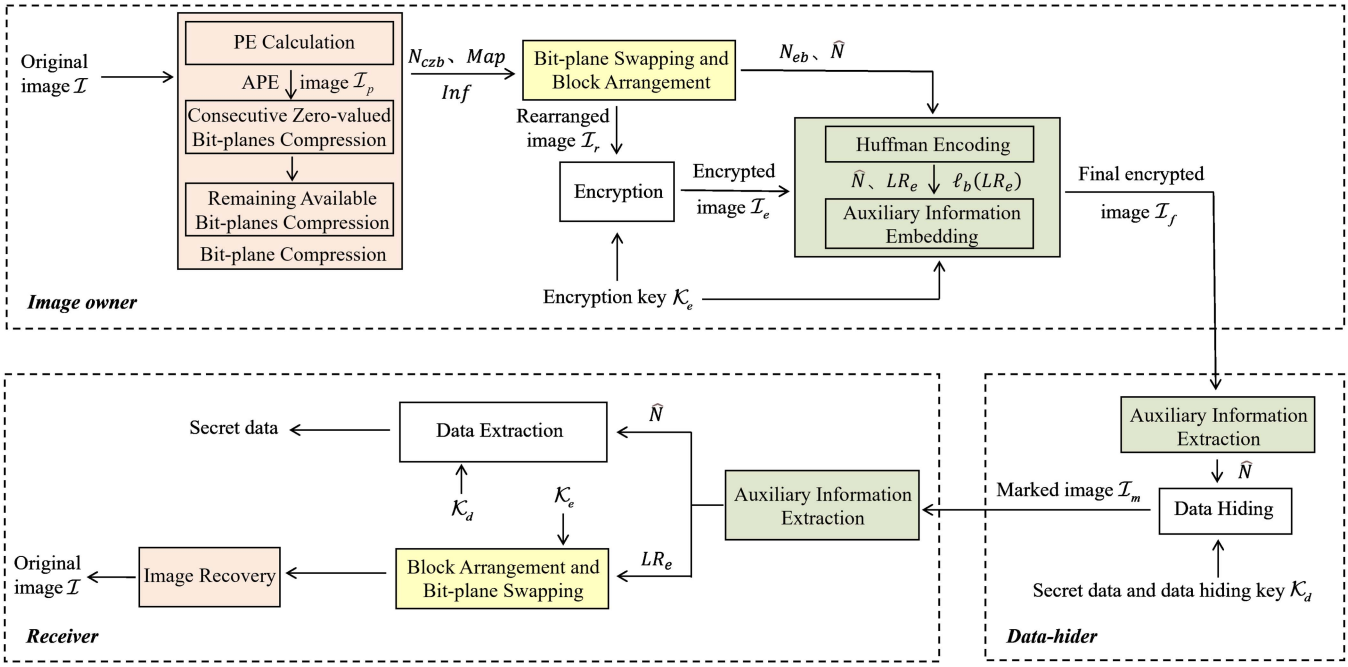

Fig. 1. Framework of the proposed method.

the following rule:

$$p(i,j)=\begin{cases}\mathcal{I}(1,1), & \text{if } i=1,j=1\\ x_1, & \text{if } i=1,2\le j\le n\\ x_2, & \text{if } 2\le i\le m,j=1\\ \max(x_1,x_2), & \text{if } i\neq 1,j\neq 1,x_3\le\min(x_1,x_2)\\ \min(x_1,x_2), & \text{if } i\neq 1,j\neq 1,x_3\ge\max(x_1,x_2)\\ x_1+x_2-x_3, & \text{otherwise}\end{cases} \tag{5}$$

where $x_1=\mathcal{I}(i,j-1)$, $x_2=\mathcal{I}(i-1,j)$ and $x_3=\mathcal{I}(i-1,j-1)$. The pixels with $i\neq 1$ and $j\neq 1$ are predicted by the median edge detector (MED) predictor [37].

Then the PE $e(i,j)$ ($e(i,j)\in[-255,255]$) of $\mathcal{I}(i,j)$ is computed as:

$$e(i,j)=\mathcal{I}(i,j)-p(i,j) \tag{6}$$

To guarantee that all the PEs are larger than or equal to 0, we have to create an APE image $\mathcal{I}_p$ to contain the absolute values of all the PEs in $\mathcal{I}$, and a sign map L_{sign} to record the sign of each PE $e(i,j)$. The $L_{sign}(i,j)$ located at the ith row and jth column is calculated below:

$$L_{sign}(i,j)=\begin{cases}0, & \text{if } e(i,j)\ge 0\\ 1, & \text{otherwise}\end{cases} \tag{7}$$

From (7), it can be clearly observed that original image $\mathcal{I}$ can be completely recovered by means of APE image $\mathcal{I}_p$, sign map L_{sign}, and pixel $\mathcal{I}(1,1)$. Each pixel in $\mathcal{I}_p$ is converted into the 8-bit binary digits $\{e^k(i,j)\}_{k=1}^8$ using (1).

2) Compressing Consecutive Zero-Valued Bit-Planes in the Block-Wise Manner: For many natural images, most PEs are very small or even close to 0, and therefore, almost all 8 bits of each PE are 0. Based on this property, we attempt to provide a considerable amount of embeddable space by compressing consecutive high zero-valued bit-planes in a block-wise manner. To do so, we partition the APE image $\mathcal{I}_p$ into non-overlapping blocks of size $t\times t$, thereby yielding a total of $\lfloor m/t\rfloor\times\lfloor n/t\rfloor$ blocks. Suppose that $\mathcal{B}(a,b)$ represents each divided block in $\mathcal{I}_p$, which contains $t\times t$ pixels and can be formulated by:

$$\mathcal{B}(a,b)=\begin{bmatrix}\mathcal{I}_p(at-t+1,bt-t+1) & \cdots & \mathcal{I}_p(at-t+1,bt)\\ \vdots & \ddots & \vdots\\ \mathcal{I}_p(at,bt-t+1) & \cdots & \mathcal{I}_p(at,bt)\end{bmatrix} \tag{8}$$

where $1\le a\le\lfloor m/t\rfloor$ and $1\le b\le\lfloor n/t\rfloor$.

Each pixel of $\mathcal{B}(a,b)$ is converted to 8-bit binary representation via (1), and the kth bit of all the pixels in $\mathcal{B}(a,b)$ is collected to construct the kth bit-plane of $\mathcal{B}(a,b)$. Suppose that the notation $N_{czb}(a,b)$ is used to represent the number of consecutive zero-valued high bit-planes in $\mathcal{B}(a,b)$, and it has 9 possible values varying from 0 to 8. Storing $N_{czb}(a,b)$ occupies very few bits while compressing the first $N_{czb}(a,b)$ zero-valued bit-planes of $\mathcal{B}(a,b)$ can provide a large amount of embeddable space, thereby greatly enhancing the payload.

3) Compressing Remaining Available Bit-Planes: For one of the remaining $(8-N_{czb}(a,b))$ bit-planes in $\mathcal{B}(a,b)$, if it satisfies the following condition: the number of '0' (short for num(0)) is significantly greater than the number of '1' (short for num(1)), and vice versa, then this bit-plane can also be compressed for providing embeddable space, that is, this bit-plane is termed the available bit-plane. When num(1) $<$ num(0), $\mathcal{G}$ is used to represent '1' and $|\mathcal{G}|$ is considered as num(1); otherwise $\mathcal{G}$ is '0' and $|\mathcal{G}|$ is num(0). We use $Inf(a,b)$ to record

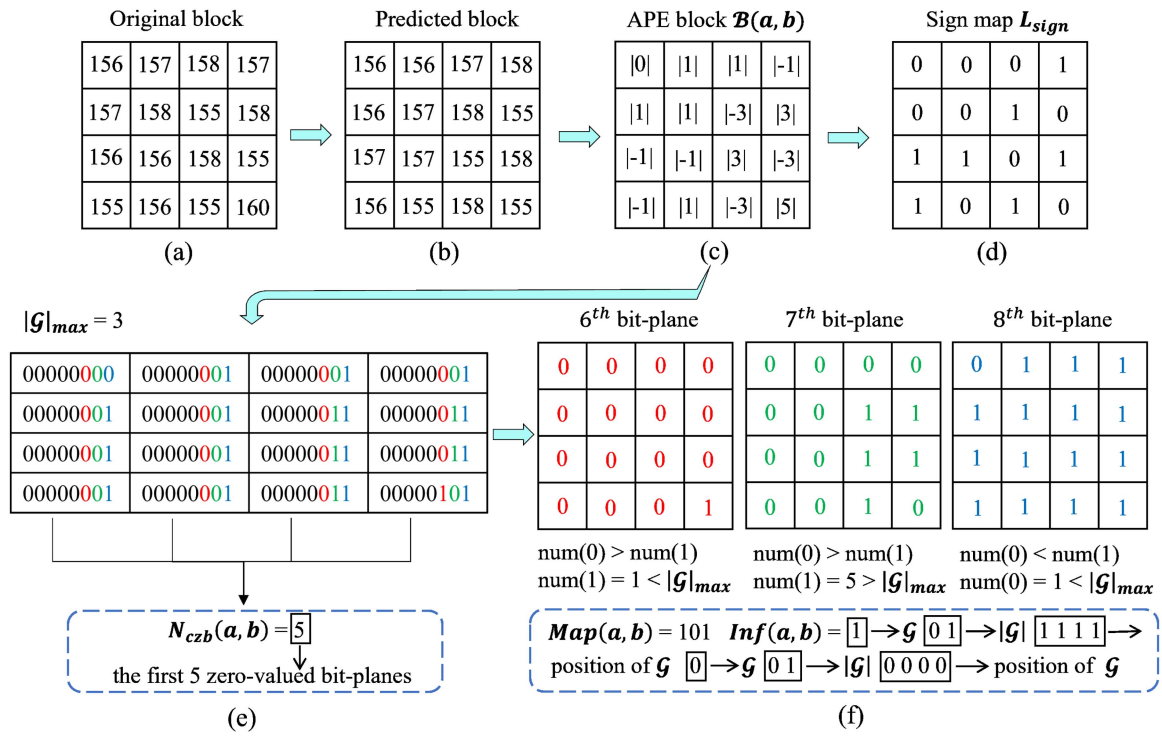

Fig. 2. Example of PE calculation and bit-plane compression in a block.

$\mathcal{G}$ in all available bit-planes of $\mathcal{B}(a,b)$ and the recorded information about each available bit-plane consists of three parts: $\mathcal{G}$ (1 b), $|\mathcal{G}|$ ($\max(1, \lceil \log_2(|\mathcal{G}|+1) \rceil)$ bits) and the position of $\mathcal{G}$ ($2\lceil \log_2 t \rceil |\mathcal{G}|$ bits). We also need to measure the bit cost $\mathcal{C}$ of recording information for an available bit-plane. $\mathcal{C}$ can be obtained by the following formula:

$$\mathcal{C} = 1 + \max(1, \lceil \log_2(|\mathcal{G}|+1) \rceil) + 2\lceil \log_2 t \rceil |\mathcal{G}| \quad (9)$$

A bit-plane contains a total of t^2 bits, therefore, the remaining embeddable room is calculated as $\mathcal{F} = t^2 - \mathcal{C}$. If $\mathcal{F} > 0$, the bit-plane is available and the corresponding value in $Map(a,b)$ is set to 1; otherwise, the value is set to 0, where $Map(a,b)$ is employed to indicate which of the remaining $(8 - N_{czb}(a,b))$ bit-planes is available, whose length is $8 - N_{czb}(a,b)$.

Since $|\mathcal{G}|$ is uncertain, the maximum value $|\mathcal{G}|_{\max}$ needs to be determined. For a $t \times t$-sized bit-plane, the maximum value $|\mathcal{G}|_{\max}$ can be obtained by the following equation:

$$|\mathcal{G}|_{\max} = \arg\min_{|\mathcal{G}|}(\mathcal{F}), \quad \text{s.t.} \quad \mathcal{F} > 0 \quad (10)$$

where $|\mathcal{G}| \in \{0, 1, \ldots, \lfloor t^2/2 \rfloor\}$.

Fig. 2 is a simple example of the PE calculation and bit-plane compression in a 4×4-sized block. Each PE is generated based on (5) and (6). The APE block $\mathcal{B}(a,b)$ and sign map denoted as L_{sign} contain the absolute values and the sign of the total 4×4 PEs, respectively. Since the block size is 4×4, the maximum value $|\mathcal{G}|_{\max}$ is 3 obtained via (10). Each pixel in the APE block is converted to the 8-bit binary sequence through (1). It can be seen that the first 5 bits of all 4×4 sequences in this block are '00000', which implies that tag $N_{czb}(a,b)$ is 5. For the 6th bit-plane and 8th bit-plane, the fewest bits are '1' and '0', respectively, and they are both less than $|\mathcal{G}|_{\max}$. Therefore, these two bit-planes are available and the corresponding bits in $Map(a,b)$ are all marked by '1'. The num(1) in 7th bit-plane is 7, which is less than num(0), but greater than $|\mathcal{G}|_{\max}$, so this bit-plane is not available, and the corresponding bit in $Map(a,b)$ is '0'. $Inf(a,b)$ is composed of '1' ($\mathcal{G}$, 1 b), '01' ($|\mathcal{G}|$, 2 bits), '1111' (the position of '1', 4 bits), '0' ($\mathcal{G}$, 1 b), '01' ($|\mathcal{G}|$, 2 bits) and '0000' (the position of '0', 4 bits) (i.e., '1' is located at the 4th row and 4th column of the 6th bit-plane and '0' is located at the 1^{st} row and 1^{st} column of the 8th bit-plane).

B. Bit-Plane Swapping and Block Rearrangement

To facilitate data embedding and extraction while improving security, we need to cluster embeddable bit-planes together and place unavailable bit-planes together by performing bit-plane swapping and block rearrangement.

For each individual block $\mathcal{B}(a,b)$ in the APE image $\mathcal{I}_p$, we use $N_{czb}(a,b)$ and $Map(a,b)$ to change the order of the last $(8 - N_{czb}(a,b))$ bit-planes, so that the bit-planes corresponding to '1' in $Map(a,b)$ are clustered together and placed in the front, while the bit-planes corresponding to '0' in $Map(a,b)$ are clustered together and placed in the back. Then, a new value $N_{eb}(a,b)$ which indicates the number of all embeddable bit-planes in block $\mathcal{B}(a,b)$ is calculated according to the following formula:

$$N_{eb}(a,b) = N_{czb}(a,b) + \ell(Map(a,b) = 1) \quad (11)$$

where $\ell(Map(a,b) = 1)$ denotes the number of '1's in the current $Map(a,b)$.

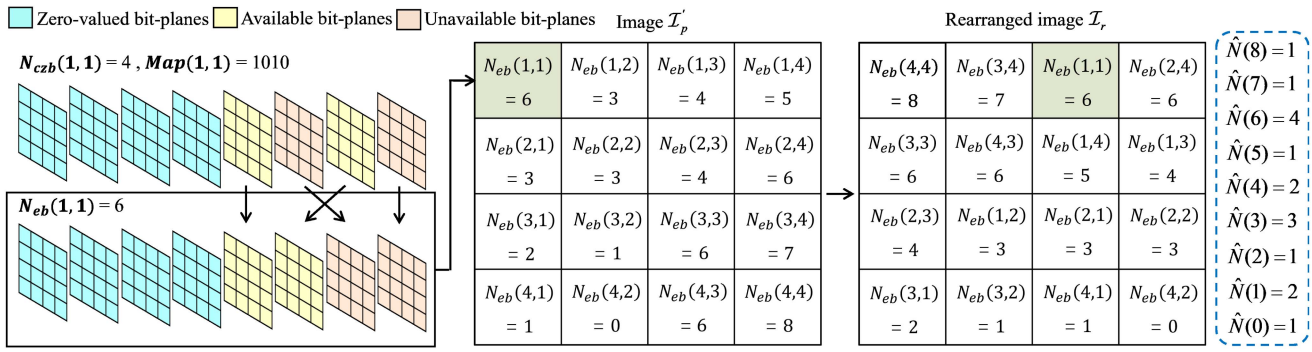

Fig. 3. Illustration of bit-plane swapping and block rearrangement.

After applying the above operation to all PE blocks in the image $\mathcal{I}_p$, the changed image $\mathcal{I}'_p$ is generated. Next, to facilitate data embedding, the blocks in the image $\mathcal{I}'_p$ need to be rearranged in descending order of $N_{eb}(a,b)$ to yield a new rearranged image $\mathcal{I}_r$. Additionally, we need to use $\hat{N}(d)$ to record the number of blocks with $N_{eb}(a,b) = d$, where d ranges from 0 to 8.

For better illustration, Fig. 3 gives a simple example to illustrate the process of bit-plane swapping and block rearrangement. Taking a 4×4 block $\mathcal{B}(1,1)$ with $N_{czb}(1,1) = 4$ and $Map(1,1) =$ '1010' as an example, it is split into 8 b-planes via (1). Based on the value of $Map(1,1)$, it is known that the 5th and 7th bit-planes are available. To this end, we swap the positions of the 7th and 5th bit-planes to collect all available bit-planes together and place them in the front. Afterwards, the $N_{eb}(1,1)$ of the changed block is 6 and the first $N_{eb}(1,1)$ bit-planes of the changed block can be used to embed data. When a new image $\mathcal{I}'_p$ is completely generated, we rearrange each block in the image $\mathcal{I}'_p$ to obtain the rearranged image $\mathcal{I}_r$ as well as $\hat{N}(d)$, where $0 \leq d \leq 8$. As we traverse the blocks of image $\mathcal{I}_r$ from left to right and top to bottom, the block with the largest $N_{eb}(a,b)$ value is visited first, enabling to vacate the maximum room to embed data. For a receiver who obtains $N_{czb}(a,b)$, $Map(a,b)$ and $\hat{N}(d)$ in advance, the $N_{eb}(a,b)$ can first be calculated via (11). Then based on generated $N_{eb}(a,b)$ and $\{\hat{N}(d)\}_{d=8}^{0}$, the original order of all rearranged blocks in the image $\mathcal{I}_r$ is obtained and all rearranged blocks are reordered in the retrieved order to generate the image $\mathcal{I}'_p$. Finally, in virtue of $N_{czb}(a,b)$ and $Map(a,b)$, the bit-planes in each block are kept unaltered or swapped according to the reverse of bit-plane swapping to restore the original APE image $\mathcal{I}_p$.

C. Image Encryption

In this step, the image $\mathcal{I}_r$ is encrypted using an encryption key $\mathcal{K}_e$. To begin with, a pseudo-random matrix $\mathcal{R}$ of size $m \times n$ is generated using the encryption key $\mathcal{K}_e$. Suppose $\mathcal{I}_r^k(i,j)$ and $\mathcal{R}^k(i,j)$ are the the 8-bit binary sequences generated using (1) for the pixel $\mathcal{I}_r(i,j)$ and its corresponding $\mathcal{R}(i,j)$, respectively. Next, the operation in (12) is applied to combine $\mathcal{I}_r^k(i,j)$ with $\mathcal{R}^k(i,j)$, resulting in the encrypted 8-bit binary sequence $\mathcal{I}_e^k(i,j)$.

$$\mathcal{I}_e^k(i,j) = \mathcal{I}_r^k(i,j) \bigoplus \mathcal{R}^k(i,j) \tag{12}$$

TABLE I
NUMBER OF EACH POSSIBLE VALUE OF $N_{czb}(a,b)$ FOR FIVE TEST IMAGES

Image	$N_{czb}(a,b)$								
	0	1	2	3	4	5	6	7	8
Lena	0	67	768	2468	6237	6271	569	4	0
Baboon	3	2443	5895	4760	3019	263	1	0	0
Jetplane	2	54	1075	2155	3316	5716	3585	481	0
Tiffany	4	31	886	2469	4624	7386	836	136	12
Man	55	569	4315	16041	35643	7889	912	105	7

where $\bigoplus$ is the XOR operation and $k \in \{1,2,\ldots,8\}$. Finally, the encrypted pixel value $\mathcal{I}_e(i,j)$ is generated via (13), the inverse of (1), and furthermore, we get the encrypted image $\mathcal{I}_e$.

$$\mathcal{I}_e(i,j) = \sum_{k=1}^{8} \mathcal{I}_e^k(i,j) \times 2^{8-k} \tag{13}$$

D. Huffman Encoding Rule and Auxiliary Information

To ensure the original image can be completely reconstructed, some certain auxiliary information must be embedded in the encrypted image $\mathcal{I}_e$ in advance. After obtaining the tag $N_{czb}(a,b)$ of all blocks in the APE image $\mathcal{I}_p$ through Section III-A, each tag $N_{czb}(a,b)$ needs to be converted into the 8-bit binary sequence and embed as a part of the auxiliary information into the encrypted image $\mathcal{I}_e$, where $N_{czb} = \{N_{czb}(a,b) | \ 1 \leq a \leq \lfloor m/t \rfloor, 1 \leq b \leq \lfloor n/t \rfloor\}$. As shown in Table I, each tag $N_{czb}(a,b)$ has 9 possible values, and the occurrence frequency of each possible value varies from image to image. To minimize the length of auxiliary information as much as possible for leaving more data embedding space, we record each tag $N_{czb}(a,b)$ through Huffman coding. A detailed explanation of the Huffman encoding rule and the composition of the auxiliary information will be provided below.

1) Huffman Encoding Rule: The proposed method allocates a longer indicator to a less frequently case and vice versa. To this end, the 9 Huffman indicators, namely $\{00, 01, 100, 101, 1100, 1101, 1110, 11110, \ 11111\}$, are defined to represent the 9 possible values, more specifically, a shorter indicator is allocated to a possible value of $N_{czb}(a,b)$

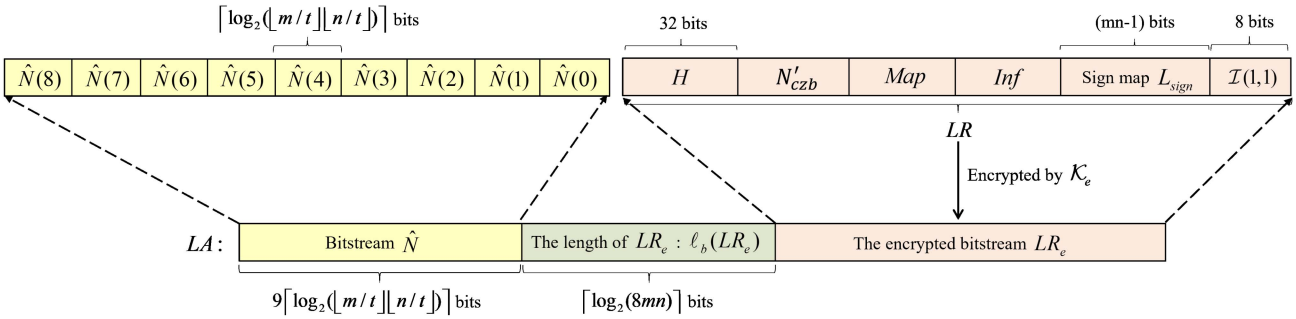

Fig. 4. Auxiliary information LA.

TABLE II
HUFFMAN ENCODING INDICATORS TAKING LENA AS AN EXAMPLE

$N_{czb}(a,b)$	0	1	2	3	4	5	6	7	8
Indicators	11110	1101	101	100	01	00	1100	1110	11111

with the most frequently occurrence frequency. Taking Lena as an example, Table II depicts that $N_{czb}(a,b) = 5$ with the most frequent occurrence are assigned the shortest indicator (namely '00'). The 9 Huffman indicators corresponding to 9 possible values (i.e., 0–8) of $N_{czb}(a,b)$ are concatenated to form the encoding rule H, i.e., $H =$'111101101101100010011001110111111'. Based on the encoding rule in Table II, each tag $N_{czb}(a,b)$ is assigned a Huffman indicator, and all the indicators according to the order from top to bottom and left to right are concatenated to form a binary bitstream N'_{czb}. With the exception of N_{czb}, it is also necessary to embed the Huffman encoding rule H into the encrypted image.

2) Auxiliary Information: All auxiliary information LA shown in Fig. 4 is mainly composed of the following three parts:

- $\hat{N}(8)$ to $\hat{N}(0)$ ($9\times \lceil \log_2(\lfloor m/t \rfloor \lfloor n/t \rfloor) \rceil$ bits, e.g., $\lceil \log_2(\lfloor m/t \rfloor \lfloor n/t \rfloor) \rceil = 14$ when splitting a 512×512-sized image into blocks with size 4×4);
- The encrypted auxiliary information LR_e. The original LR is further split into the following six sub-parts:

-Huffman encoding rule H (32 bits);
-N'_{czb};
-$Map(a,b)$ of all $\lfloor m/t \rfloor \times \lfloor n/t \rfloor$ blocks;
-$Inf(a,b)$ of all $\lfloor m/t \rfloor \times \lfloor n/t \rfloor$ blocks;
-The sign map L_{sign} ($m \times n - 1$ bits);
-The first original pixel $\mathcal{I}(1,1)$ (8 bits);

- The binary sequence $\ell_b(LR_e)$ of length $\lceil \log_2(8mn) \rceil$ bits used for recording the length of bitstream LR_e.

To ensure sufficient security, LR needs to be encrypted using the encryption key $\mathcal{K}_e$ mentioned above so as to form the encrypted bitstream LR_e. According to the bitstream $\hat{N}$, the image owner first traverses each block of the encrypted image $\mathcal{I}_e$ in the top-to-bottom and left-to-right order, the number of embeddable bit-planes in the currently traversed block can be obtained in advance by $\hat{N}(d)$, where $0 \le d \le 8$. Next, for each block, the auxiliary information will be embedded into embeddable bit-planes in the order of 1^{st} bit-plane to 8th bit-plane and every bit-plane is substituted with bits in raster-scanning order. When a level of bit-plane is full, the next embeddable bit-plane will be replaced until all embeddable bit-planes are used up. Eventually, the final encrypted image $\mathcal{I}_f$ containing all the auxiliary information is generated and sent by the image owner to the data hider.

E. Data Hiding

When a data hider receives the encrypted image $\mathcal{I}_f$ only embedded with the auxiliary information, all $m \times n$ blocks are visited in the top-to-bottom and left-to-right order. Before embedding the secret data into the image, it is necessary to know the starting position. Therefore, the first two steps are used to obtain the position of data embedding and the final step is carried out to achieve data embedding.

1) From the first block which can provide the largest embedding capacity, $\hat{N}(d)$ of length $\lceil \log_2(\lfloor m/t \rfloor \lfloor n/t \rfloor) \rceil$ bits is extracted in order from $d = 8$ to $d = 0$ until extracted $\hat{N}(d)$ is not 0. By the value of non-zero $\hat{N}(d)$, the data hider can know the first d high bit-planes of the $1^{st} \sim \hat{N}(d)$th blocks are available, and thus, the remaining $\hat{N}(d)$ are extracted by these known embeddable space. The known embeddable space also increases with the later continuously extracted $\hat{N}(d)$, so that the full $\hat{N}$ is finally obtained and all embedding space of the entire image will be known.
2) The data hider continues to extract the $\ell_b(LR_e)$ of length $\lceil \log_2(8mn) \rceil$ bits, then the LR_e can also be obtained by its length.
3) The payload is further encrypted by a data hiding key $\mathcal{K}_d$ for increasing the security and then embedded into the image $\mathcal{I}_f$ after LR_e in the same way as embedding auxiliary information in Section III-D.

In this way, the data hider can hide any desired data in the image $\mathcal{I}_f$. When the last block in the image is traversed and all the embeddable space has been utilized, an encrypted marked image $\mathcal{I}_m$ is finally generated.

F. Data Extraction and Image Recovery

When a receiver receives the encrypted marked image $\mathcal{I}_m$, the entire bitstream $\hat{N}$, the length $\ell_b(LR_e)$, the encrypted bitstream LR_e and secret data can be extracted by the same way as the data hider. The above process does not require any keys, but the

receiver's access to the embedded secret data and recovery of the original image depends on different keys.

If the receiver only has the data hiding key $\mathcal{K}_d$, and no encryption key $\mathcal{K}_e$, the embedded data can be obtained by decrypting the extracted encrypted secret data, but the original image cannot be recovered.

If the receiver only has encryption key $\mathcal{K}_e$ and no data hiding key $\mathcal{K}_d$, the original image $\mathcal{I}$ can be recovered without distortion. The detailed image recovery process consists of the following steps:

1) *Decryption using encryption key $\mathcal{K}_e$:* The extracted encrypted bitstream LR_e and encrypted marked image $\mathcal{I}_m$ can be decrypted into the original auxiliary information LR and new image $\mathcal{I}'_m$ by $\mathcal{K}_e$ respectively.
2) *Extract Huffman encoding rule H and all tags N_{czb}:* The first 32 bits in the bitstream LR are the Huffman encoding rule H. With exception to H, the remaining bits of LR are extracted according to the encoding rule H to obtain $\lfloor m/t \rfloor \times \lfloor n/t \rfloor$ Huffman indicators. Afterwards, all tags N_{czb} can be obtained.
3) *Determine the length $\ell(Map)$ and extract Map:* With extracted N_{czb}, the length $\ell(Map)$ of bitstream Map can be determined using the following rule:

$$\ell(Map) = 8\lfloor m/t \rfloor \lfloor n/t \rfloor - \sum_{i=1}^{\lfloor m/t \rfloor} \sum_{j=1}^{\lfloor n/t \rfloor} N_{czb}(a,b) \quad (14)$$

Defending on the length $\ell(Map)$, the bitstream Map in LR be easily extracted.
4) *Extract the Inf:* To extract Inf, the number of available bit-planes (short for N_a) in all blocks of the APE image $\mathcal{I}_p$ needs to be determined using the following formula:

$$N_a = \sum_{i=1}^{\lfloor m/t \rfloor} \sum_{j=1}^{\lfloor n/t \rfloor} \ell(Map(a,b) = 1) \quad (15)$$

where $\ell(Map(a,b) = 1)$ denotes the number of '1's in the current bitstream $Map(a,b)$. Since the length of $Inf(a,b)$ depends on the extracted $|\mathcal{G}|$ from available bit-planes in block $\mathcal{B}(a,b)$, the $Inf(a,b)$ is extracted from LR one by one until the sum of extracted $|\mathcal{G}|$ in all blocks is equal to N_a.
5) *Extract the sign map L_{sign} and $\mathcal{I}(1,1)$:* The sign map L_{sign} and pixel $\mathcal{I}(1,1)$ in LR can be easily obtained according to their respective length.
6) *Recover the position of blocks and bit-planes:* The $N_{eb}(a,b)$ is calculated via (11). Based on $N_{eb}(a,b)$ and $\{\hat{N}(d)\}_{d=8}^{0}$, the original order of all rearranged blocks in the image $\mathcal{I}'$ are obtained. All rearranged blocks are reordered in the retrieved order. In virtue of $N_{czb}(a,b)$ and $Map(a,b)$, the bit-planes in each block are kept unaltered or swapped according to the reverse of bit-plane swapping in Section III-B.
7) *Recover the original APE image $\mathcal{I}_p$:* The original bits in available bit-planes are restored using $Map(a,b)$ and $Inf(a,b)$. In contrast, the bits in zero-valued bit-planes are all set to '0'. Suppose that each block of the current image is set to $\mathcal{B}'(a,b)$, the original PE block $\mathcal{B}(a,b)$ is reconstructed through . (16):

$$\mathcal{B}(a,b) = \mathcal{B}'(a,b) \bmod 2^{8-N_{czb}(a,b)} \quad (16)$$

and the original APE image $\mathcal{I}_p$ is generated.
8) *Recover the original image $\mathcal{I}$:* With the values of sign map L_{sign} and pixel $\mathcal{I}(1,1)$, the original image $\mathcal{I}$ can be recovered without distortion.

If the receiver possesses both the data hiding key $\mathcal{K}_d$ and the encryption key $\mathcal{K}_e$, the embedded secret data can be correctly extracted, and the original image can be recovered losslessly.

IV. Experimental Results and Analysis

In this section, several experiments are conducted to evaluate the effectiveness and superiority of the proposed method. The proposed method is compared with several state-of-the-art RDHEI schemes in terms of security and payload. Five 512×512-sized grayscale images including 'Lena', 'Baboon', 'Jetplane', 'Tiffany', and 'Man' are used as test images. In addition, two commonly used image datasets, including the BOSSbase [38] and BOWS-2 [39] datasets, are used in the experiments. Each of BOSSbase and BOWS-2 includes 10,000 grayscale images of size 512×512. In the experiments, two metrics, namely PSNR and SSIM, are adopted to test the reversibility of the proposed method.

A. Security Analysis

The security of images is mainly measured from two aspects, i.e., the invisibility of the content and the undetectability of the embedded data. In general, the statistical analysis can be used to measure the visual security level. Therefore, the security of the proposed method is measured from statistics analysis and different attacks. In addition, the weakness of some existing state-of-the-art methods in terms of security is discussed. From the discussion, it is proved that the proposed method is sufficiently secure in comparison with other state-of-the-art schemes.

1) Statistical Analysis: The visual security metrics, including the horizontal and vertical correlation, Shannon entropy (SE), mean absolute error (MAE), χ^2, the number of changing pixel rate (NPCR), and unified averaged changed intensity (UACI) are used in statistical analysis.

Table III depicts the comparison of security among the original image $\mathcal{I}$, the final encrypted image $\mathcal{I}_f$ and the encrypted marked $\mathcal{I}_m$ under different statistical analysis metrics for the four test images, each metric has its corresponding boundary value. From Table III, it can be observed that for all four test images, the vertical and horizontal correlation between two adjacent pixels of the original image $\mathcal{I}$ are approximately 1, while the vertical and horizontal correlation of both $\mathcal{I}_f$ and $\mathcal{I}_m$ are very close to 0, which implies that there exists strong correlation between the pixels in image $\mathcal{I}$ while weak correlation between the pixels in image $\mathcal{I}_f$ or $\mathcal{I}_m$.

The SE for $\mathcal{I}$ varies from image to image. In contrast, the SE for $\mathcal{I}_f$ or $\mathcal{I}_m$ is approximately 8 for all four images. In addition, χ^2 for $\mathcal{I}$ is much larger than those of $\mathcal{I}_f$ and $\mathcal{I}_m$, which implies that image $\mathcal{I}_f$ and image $\mathcal{I}_m$ offer a more uniform distribution

TABLE III
STATISTICAL ANALYSIS OF TEST IMAGES

Test images		Correlation (0 ∼ 1) Horizontal↓	Correlation (0 ∼ 1) Vertical↓	SE↑ (0 ∼ 8)	MAE↑ (0 ∼ 255)	χ^2↓ (0 ∼ +∞)	NPCR↑ (0% ∼ 100%)	UACI↑ (0% ∼ 100%)
Lena	$\mathcal{I}$	0.9698	0.9843	7.4451	-	158349.3555	-	-
	$\mathcal{I}_f$	0.0025	0.0019	7.9993	72.7720	250.7949	99.61%	28.54%
	$\mathcal{I}_m$	0.0024	0.0012	7.9993	72.9633	254.2246	99.60%	28.61%
Baboon	$\mathcal{I}$	0.8652	0.7515	7.3583	-	187356.5723	-	-
	$\mathcal{I}_f$	0.0025	0.0030	7.9994	70.9180	234.1953	99.61%	27.81%
	$\mathcal{I}_m$	0.0012	0.0022	7.9993	71.0073	249.0313	99.60%	27.85%
Jetplane	$\mathcal{I}$	0.9605	0.9607	6.7025	-	717779.3359	-	-
	$\mathcal{I}_f$	0.0002	0.0004	7.9993	82.8043	260.5098	99.61%	32.47%
	$\mathcal{I}_m$	0.0023	0.0006	7.9994	82.7691	234.8242	99.60%	32.46%
Tiffany	$\mathcal{I}$	0.9190	0.9246	6.6009	-	530208.9355	-	-
	$\mathcal{I}_f$	0.0002	0.0009	7.9992	94.5984	281.8965	99.62%	37.10%
	$\mathcal{I}_m$	0.0001	0.0029	7.9992	94.3221	273.3438	99.61%	36.99%

↑ denotes higher value is better, and vice versa.

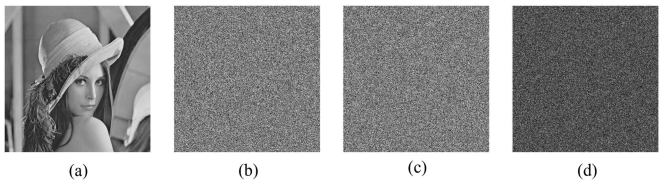

Fig. 5. Proposed method resisting the Brute-Force attack. (a) The original image; (b) The encrypted image via $\mathcal{K}_e$; (c) The encrypted image via $\mathcal{K}'_e$; (d) The difference between (b) and (c).

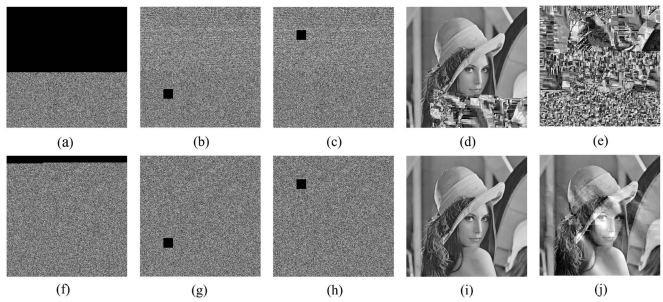

Fig. 6. Process of Yin et al.'s method [24] and proposed method suffering from patch removal attacks: (a) and (f) are the final encrypted images of Yin et al.'s method [24] and the proposed method, respectively, the necessary auxiliary information is marked in black; (b) and (c) represent the 40×40 patch removal attack on the lower-left region and upper-left region of (a), respectively; (g) and (h) represent the 40×40 patch removal attack on the lower-left region and upper-left region of (f), respectively; (d), (e), (i), and (j) are the recovered images produced by (b), (c), (g), (h), respectively.

compared to original image $\mathcal{I}$. For $\mathcal{I}_f$ and $\mathcal{I}_m$, the NPCR approaches to 1. In addition, higher MAE and UACI also indicate that both images significantly differ from the original image, and therefore, it is more difficult to obtain $\mathcal{I}$ from $\mathcal{I}_f$ or $\mathcal{I}_m$. The above statistical analysis effectively demonstrates the security of image $\mathcal{I}_f$ and $\mathcal{I}_m$.

2) Brute-Force Attack: Taking Lena of size 512×512 as an example, the probability that the original image can be obtained without the encryption key is evaluated. The image is split into 4×4-sized non-overlapping blocks and then encrypted using a randomly generated binary sequence of length $8 \times 512 \times 512$, where each bit valued at 0 or 1 has equal probability. Considering that $\hat{N}$ of length 126 bits, $\ell_b(LR_e)$ of length 21 bits, LR_e of length 516,856 bits, and the encrypted secret data of length 814,149 bits in the encrypted marked image, the number C_k of all possible values of the encryption key can be obtained by:

$$C_k = 2^{8\times512\times512-126-21-814,149}\left(2^{1,282,856}\right) \qquad (17)$$

C_k is so large that it is almost unlikely to obtain the correct encryption key from all possible values. Suppose another encryption key $\mathcal{K}'_e$ is generated by changing only one bit of $\mathcal{K}_e$. The two encryption keys $\mathcal{K}_e$ and $\mathcal{K}'_e$ are used to test the key sensitivity when confronting the brute-force attack, Lena is encrypted using $\mathcal{K}_e$ and $\mathcal{K}'_e$ to yield two encrypted images, respectively. The experimental results in Fig. 5 depict that although the difference between $\mathcal{K}_e$ and $\mathcal{K}'_e$ is only one bit, two resulting encrypted images are indeed completely different, implying that it is very difficult for an attacker to obtain the original image content by analyzing the encryption key. To sum up, the proposed method can efficiently resist the brute-force attack.

3) Patch Removal Attack: In the proposed method, although the positions of all the blocks as well as available bit-planes will be changed, these shifted blocks and bit-planes are eventually restored to their original positions by means of auxiliary information $\hat{N}$, N_{czb} and Map. Therefore, as long as the auxiliary information in the image can not be destroyed, the encrypted image can finally restore the approximate content of the original image when it is attacked by patch removal. The existing RDHEI methods need to embed the auxiliary information used to restore the original image into the encrypted image in order to ensure reversibility. The distribution range of auxiliary information in the image determines the image's ability to resist patch removal attacks, and when the extent of distribution is larger, the image is less resistant to attacks.

Taking Yin et al.'s method [24] as an example, Fig. 6(a), (b), (c), (d), and (e) show how the method suffers from a patch removal attack using Lena as a test image. Fig. 6(a) is the encrypted image containing auxiliary information, the locations in the image where the auxiliary information is stored are marked

in black, and it can be seen that the distribution range of the auxiliary information occupies almost half of the image. The 40×40 patch removal attack is applied to the lower-left and the upper-left region of the encrypted image, which is represented in Fig. 6(b) and (c), respectively. Fig. 6(d) and (e) are the final recovered images generated from Fig. 6(b) and (c), respectively. From the results, it can be seen that most of the regions in Fig. 6(d) can be recovered to their original appearance, while Fig. 6(e) fails to reflect the contents of the original image. This is because the method makes the distribution of the auxiliary information too large, and the patch in the upper-left corner of the image removes a part of the auxiliary information thus leading to the failure of image recovery. Therefore, the ability of this method to resist patch removal attacks is not strong. Figs. 6(f), (g), (h), (i), and (j) show the process of the proposed method using the same test image subjected to patch removal attack. From Fig. 6(f), it can be seen that the area occupied by the auxiliary information $\hat{N}$, N_{czb} and Map is very small compared with Yin et al.'s method [24]. Fig. 6(g) and (h) carry out the patch removal attack at the same location as Fig. 6(b) and (c). It can be seen from Fig. 6(i) and (j) that the attacks at both locations result in the successful recovery of the majority of the original image information. This is due to the fact that the proposed method compresses consecutive zero-valued bit-planes in a block-wise manner making the length of auxiliary information greatly reduced. In addition, the proposed bit-plane swapping and block rearrangement technique gathers the embeddable bit-planes in each block and makes the blocks with larger embedding capacity concentrated at the top of the image, so that the number of blocks for storing the auxiliary information is greatly reduced and the ability of the image to resist this type of attack is greatly enhanced.

4) Security Comparison: It is found that many existing RDHEI methods [20], [21], [23], [24] significantly change the distribution of encrypted pixels because they exploited the unencrypted auxiliary information to replace the encrypted bits, thereby destroying the random characteristics of encrypted pixels. Therefore, we can know whether an encrypted image contains the auxiliary information or not by judging the difference of pixel distribution between the encrypted image that is only embedded with the auxiliary information and the encrypted marked image. Although some methods [25], [29] try to use a new key to encrypt the auxiliary information to solve such security problems, the receiver needs two kinds of keys to extract the secret data or recover the original image. The lack of either key cannot complete the corresponding process, so this way is tedious and error-prone.

To remedy this problem, the proposed method uses an encryption key $\mathcal{K}_e$ to encrypt a large portion of the auxiliary information while retaining only a small portion, resulting in more uniform distribution of pixels in the resulting image. Four metrics, namely χ^2, SE, MAE and UACI, are exploited to evaluate the pixel distribution of the final encrypted image $\mathcal{I}_f$ and the marked image $\mathcal{I}_m$ produced using any compared method from [20], [21], [23], [24], [33]. The smaller χ^2 and larger SE indicate that the shape of pixels resembles a more uniform distribution, and a certain degree of high MAE or UACI means that the image pixels have better random variability. As shown in Table IV, the proposed method provides the smallest $\chi^2_{\mathcal{I}_f-\mathcal{I}_m}$ and the largest $\text{SE}_{\mathcal{I}_f-\mathcal{I}_m}$ compared with the other five methods, which means that our method has a higher degree of pixel uniformly distribution. Higher $\text{MAE}_{\mathcal{I}_f-\mathcal{I}_m}$ and $\text{UACI}_{\mathcal{I}_f-\mathcal{I}_m}$ also indicate that the image pixels generated by proposed method have better randomness than others. All four metrics in our methods have the lowest difference between final encrypted image $\mathcal{I}_f$ and marked image $\mathcal{I}_m$. These differences close to 0 prove that the images produced by the various processes of this method have the best unpredictability as well as safety, while the other compared methods have significant differences, which may increase the risk due to the traces of image pre-processing.

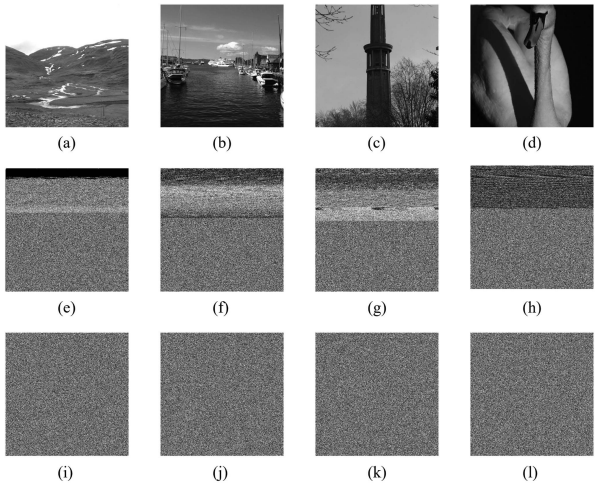

Fig. 7. Visual quality comparison of encrypted marked images generated through Yin et al.'s [24] and the proposed methods: (a)–(d) are four original images, (e)–(h) are four encrypted marked images generated using Yin et al.'s method [24], (i)–(l) are four encrypted marked images generated using the proposed method.

Given that the embedded auxiliary information in Yin et al.'s scheme [24] is not encrypted, there are some black or white blocks appearing in the marked images, directly resulting in damage to the invisibility of the encrypted image content, and thus seriously affecting the security. The same phenomenon occurs in Gao et al.'s scheme [29]. To solve this problem, Gao et al. reversibly scrambled the encrypted marked images, however, their method is not conducive to processing marked images by receivers with different keys, and increases the tediousness of image processing and data extraction, which may be prone to produce errors. Taking four test images from the BOSSbase dataset [38], i.e., Bossbase1, BOSSbase59, BOSSbase1002, and BOSSbase1280, as an example, Fig. 7 shows the comparison of marked images using Yin et al.'s [24] and the proposed methods, where Fig. 7(a), (b), (c), and (d) are four original images and Fig. 7(e), (f), (g), and (h) are the corresponding encrypted marked images using [24]. Fig. 7(e), (f), (g), and (h) suffer from a considerable amount of black or white points, which inevitably lead to the insecurity of the images content.

TABLE IV
COMPARISON OF $\chi^2_{\mathcal{I}_f-\mathcal{I}_m}$, $\mathrm{SE}_{\mathcal{I}_f-\mathcal{I}_m}$, $\mathrm{MAE}_{\mathcal{I}_f-\mathcal{I}_m}$, AND $\mathrm{UACI}_{\mathcal{I}_f-\mathcal{I}_m}$ AMONG SIX COMPARED WORKS

Metrics	Puyang *et al.* [20]		Yi *et al.* [21]		Wu *et al.* [23]		Yin *et al.* [24]		Yu *et al.* [33]		Proposed method	
	$\mathcal{I}_f$	$\mathcal{I}_m$	$\mathcal{I}_f$	$\mathcal{I}_m$	$\mathcal{I}_f$	$\mathcal{I}_m$	$\mathcal{I}_f$	$\mathcal{I}_m$	$\mathcal{I}_f$	$\mathcal{I}_m$	$\mathcal{I}_f$	$\mathcal{I}_m$
$\chi^2_{\mathcal{I}_f-\mathcal{I}_m}$	260	138,938(+138678)	389	118,641(+118254)	260	257,147(+256887)	260	7101(+6841)	456	6406(+5950)	251	**254(+3)**
$\mathrm{SE}_{\mathcal{I}_f-\mathcal{I}_m}$	7.9993	7.6766(-0.3227)	7.9989	7.7088(-0.2901)	7.9993	7.3810(-0.6183)	7.9993	7.9821(-0.0172)	7.9987	7.9845(-0.0142)	7.9993	**7.9993(-0)**
$\mathrm{MAE}_{\mathcal{I}_f-\mathcal{I}_m}$	73.3091	75.3866(+2.0775)	72.5177	74.8116(+2.2939)	72.5382	75.0908(+2.5526)	73.0668	74.9499(+1.8831)	73.7053	75.2111(+1.5058)	75.2976	**75.3915(+0.0939)**
$\mathrm{UACI}_{\mathcal{I}_f-\mathcal{I}_m}$	28.64%	29.45%(+0.81%)	28.33%	29.22%(+0.89%)	28.34%	29.33%(+0.99%)	28.54%	29.28%(+0.74%)	28.79%	29.38%(+0.59%)	29.41%	**29.44%(+0.03%)**

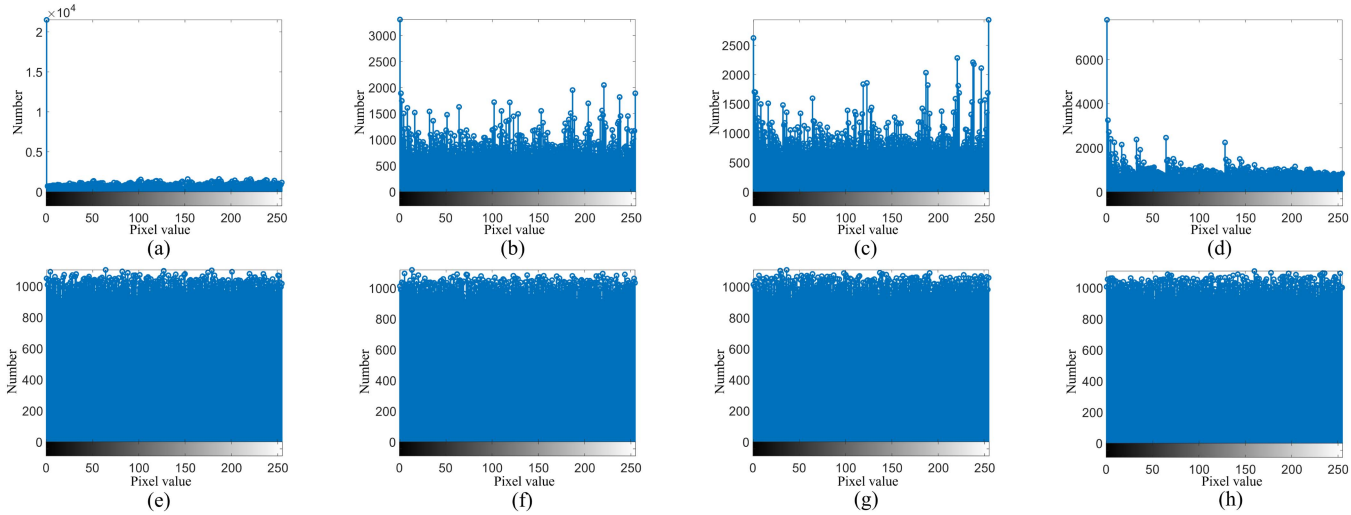

Fig. 8. Illustration of undetectability of the encrypted marked images: (a)–(d) are the pixel distributions of four encrypted marked images corresponding to Fig. 7(e)–(h), (e)–(h) are the pixel distribution of four encrypted marked image corresponding to Fig. 7(i)–(l).

Fig. 8(a), (b), (c), and (d) give the pixel histogram distribution of images in Fig. 7(e), (f), (g), and (h). From Fig. 8(a), (b), (c), and (d), it can be observed that the pixel distribution is not uniform and the pixels valued at 0 or 255 occupy a large proportion of all the pixels. Accordingly, Fig. 7(i), (j), (k), and (l) show the encrypted marked images generated using the proposed RRBE-based method. From Fig. 7(i), (j), (k), and (l), it is clearly observed that the image contents do not contain visible black or white points. Fig. 8(e), (f), (g), and (h) give the corresponding histograms, which show that the pixel distribution of all these images is uniform. Therefore, the proposed method does not produce black or white blocks and achieves satisfactory security and robustness, compared with two existing RRBE methods.

B. Embedding Performance Comparison

Table V illustrates the amount of the embedding capacity and auxiliary information of five test images when the block size is $m \times n$ is 3×3, 3×4, 4×4, 4×5 and 5×5, respectively. The 'EC' in Table V means the size of embedded bits including the auxiliary information and pure payload measured by bits. The bitstream $\hat{N}$, $\ell_b(LR_e)$ and LR_e together form the auxiliary information while 'P_{pure}' in Table V denotes the pure payload. The capacity gain from using Huffman coding is labeled after the P_{pure}. It can be observed that different block sizes can contribute to different payloads for each test image. The best payload for each image is marked in bold. Generally, the block size 4×4 can provide larger embedding capacity than the other four block sizes. As the block size decreases and the number of blocks increases, the capacity improvement caused by using Huffman coding is more obvious. When the block size is reduced from 4×4 to 3×4 or 3×3, $N_{czb}(a, b)$ of each block indeed becomes larger due to smaller number of pixels in the block; however, the number of blocks increases and the $8 - N_{czb}(a, b)$ bit-planes of each block are less likely to be available. When the block size is increased from 4×4 to 4×5 or 5×5, the number of blocks indeed decreases, and the probability of the $8 - N_{czb}(a, b)$ bit-planes in each block judged as available becomes greater; however, the value of $N_{czb}(a, b)$ tends to be smaller due to larger number of pixels in each block. Therefore, to make a balance between $N_{czb}(a, b)$, the number of blocks, and the probability of a bit-plane that is judged as available, the block size is set 4×4 in the experiments, and thus the proposed method achieves a larger payload. To demonstrate the generality of the proposed method, we conduct the experiments on the two datasets, i.e., BOSSbase [38] and BOWS-2 [39]. Table VI shows the experimental results of pure payload measured by bpp and reversibility of images on the two datasets. It can be seen that for the dataset BOSSbase, the maximum payload can reach 6.772 bpp and the minimum is only 0.651 bpp. Similarly, for the BOWS-2 dataset, the maximum is 6.384 bpp and the minimum is 0.582 bpp. In addition, for the two datasets, the proposed method has an average payload of 3.793 bpp and 3.705 bpp, respectively, and all marked images in the two datasets can be

TABLE V
PURE PAYLOAD AND AUXILIARY INFORMATION FOR FIVE TEST IMAGES (BITS)

Test images	Block size	EC	Auxiliary information			P_{pure}(Increased capacity using Huffman coding)
			$\hat{N}$	$\ell_b(LR_e)$	LR_e	
Lena	3×3	1291671	135	21	524347	767168(+49676)
	3×4	1330356	135	21	547009	783191(+37443)
	4×4	**1331152**	126	21	516856	**814149**(+28214)
	4×5	1265360	126	21	460427	804786(+22419)
	5×5	1220125	126	21	421409	798569(+17791)
Baboon	3×3	891837	135	21	591933	299748(+46009)
	3×4	**933012**	135	21	609919	322937(+35225)
	4×4	926064	126	21	565371	**360546**(+26737)
	4×5	848100	126	21	489907	358046(+21357)
	5×5	796325	126	21	440011	356167(+17017)
Jetplane	3×3	1398564	135	21	516045	882363(+43236)
	3×4	1439028	135	21	541055	897817(+32285)
	4×4	**1443568**	126	21	516833	**926588**(+23983)
	4×5	1379600	126	21	464685	914768(+18907)
	5×5	1329775	126	21	424626	905002(+15006)
Tiffany	3×3	1324503	135	21	521881	802466(+47751)
	3×4	1364796	135	21	547166	817474(+36192)
	4×4	**1366752**	126	21	519299	**847306**(+27290)
	4×5	1300640	126	21	463598	836895(+21907)
	5×5	1247600	126	21	419281	828172(+17426)
Man	3×3	4785696	153	23	2183546	2601974(+195846)
	3×4	**4938372**	153	23	2279885	2658311(+150371)
	4×4	4926864	144	23	2151918	**2774779**(+115364)
	4×5	4634780	144	23	1896683	2737930(+92911)
	5×5	4424425	144	23	1713260	2710998(+74406)

The bold entities represent the best payload for each image.

TABLE VI
EXPERIMENTAL RESULTS OF PAYLOAD (BPP) AND REVERSIBILITY ON TWO IMAGE DATASETS

Dataset	Payload			PSNR	SSIM
	Maximum	Minimum	Average		
BOSSbase	6.772	0.651	3.793	$+\infty$	1
BOWS-2	6.384	0.582	3.705	$+\infty$	1

restored to the original states without the distortion from PSNR $= +\infty$ and SSIM $= 1$.

To further illustrate the superiority, the proposed method is compared with several state-of-the-art works [22], [23], [24], [25], [26], [27], [28], [29], [33]. In the experimental settings, the length of fixed-length codewords and block size in [22] are set as 3 and 4×4 respectively. In [23], the parameters α and β are set as 5 and 2. The block sizes in [25] and [33] are set to be 3×3 and 4×4 respectively to achieve maximum embedded payloads and the 8 b-planes are all compressed in [26]. In [28], we use the HBVLC-based scheme, which yields a larger embedding capacity. First, we compare the ER of five test images. Fig. 9 provides the performance comparison of ER among the proposed scheme and methods in [22], [23], [24], [25], [26], [27], [29], [33], it is clear that our proposed method outperforms the other methods on almost every image. Table VII shows the detailed ER of different methods on test images. It can be seen that the ER of the proposed method is higher than other methods on all four test images and only 0.086 bpp lower than Yu et al.'s

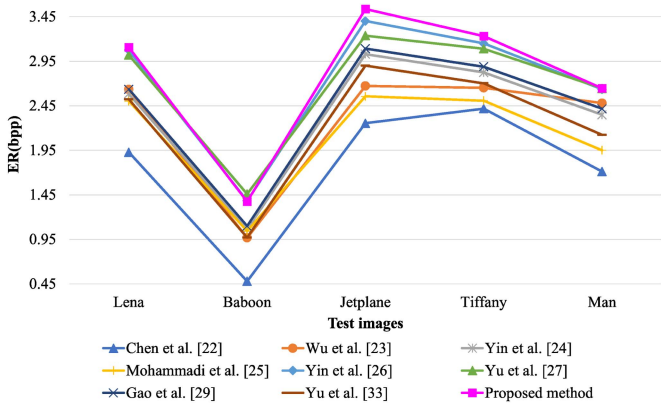

Fig. 9. Comparison of maximum payload (bpp) among several compared methods for five test images.

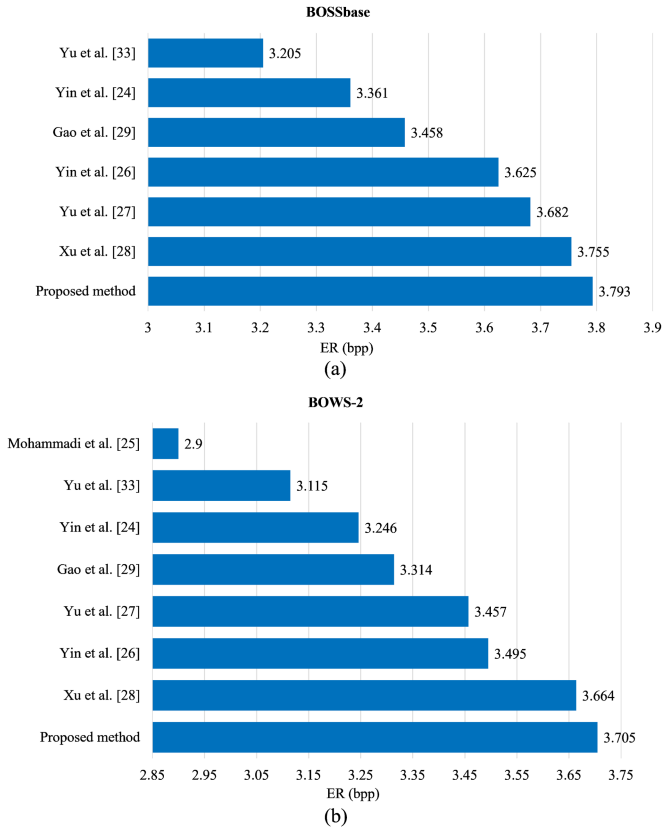

Fig. 10. Comparison of average payload (bpp) among several compared methods on two datasets.

method [27] on the Baboon image. This is because the Baboon image has abundant textures and the correlation between pixels weakens, which can lead to limited secret bits embedding. In addition, the proposed method also has higher average ER on five tested images than other methods.

Likewise, the proposed method is compared with several related schemes [24], [25], [26], [27], [28], [29], [33] on two datasets. As can be seen from Fig. 10, the proposed method still performs the best on the BOSSbase and BOWS-2 datasets.

TABLE VII
MAXIMUM ER (BPP) COMPARISON OF DIFFERENT METHODS ON FIVE TEST IMAGES

Test images	Maximum payload (bpp)								
	Chen *et al.* [22]	Wu *et al.* [23]	Yin *et al.* [24]	Mohammadi *et al.* [25]	Yin *et al.* [26]	Yu *et al.* [27]	Gao *et al.* [29]	Yu *et al.* [33]	Proposed method
Lena	1.928	2.635	2.583	2.503	3.075	3.019	2.634	2.521	**3.106**
Baboon	0.480	0.969	1.066	1.044	1.383	**1.460**	1.099	0.970	1.375
Jetplane	2.254	2.673	3.030	2.556	3.402	3.237	3.092	2.900	**3.535**
Tiffany	2.418	2.652	2.824	2.507	3.149	3.090	2.888	2.701	**3.232**
Man	1.712	2.479	2.349	1.950	2.635	2.645	2.417	2.123	**2.647**
Average	1.758	2.282	2.370	2.112	2.729	2.690	2.426	2.243	**2.779**

The bold entities represent the largest payload in each item compared.

TABLE VIII
COMPARISON OF TIME COMPLEXITY AND SPACE COMPLEXITY

Method	Time complexity	Space complexity
Puteaux *et al.*'s method [19]	$O(n^3)$	$O(n^2)$
Yi *et al.*'s method [21]	$O(n^2)$	$O(n^2)$
Chen *et al.*'s method [22]	$O(n^5)$	$O(n^2)$
Yin *et al.*'s method [24]	$O(n^2)$	$O(n^2)$
Mohammadi *et al.*'s method [25]	$O(n^2)$	$O(n^2)$
Yu *et al.*'s method [27]	$O(n^2)$	$O(n^2)$
Gao *et al.*'s method [29]	$O(n^2)$	$O(n^2)$
Proposed method	$O(n^2)$	$O(n^2)$

The proposed method compresses consecutive high zero-valued bit-planes in a block-wise manner for providing a large amount of payload. In addition, we take full advantage of available bit-planes for further improving the payload. In a word, the payload of the proposed method is generally higher than the other compared methods.

C. Time and Space Complexity Analysis

In this section, time and space complexity are measured and compared by theoretical analysis. In the proposed scheme, four main steps can be considered: 1) Bit-plane compression; 2) Bit-plane swapping and block rearrangement; 3) Image encryption/decryption; 4) Data hiding/extraction and image recovery. The bit-plane compression is done for bit-planes in $\lfloor m/t \rfloor \times \lfloor n/t \rfloor$ blocks, thus the time complexity would be $O(8\lfloor m/t \rfloor \times \lfloor n/t \rfloor)$. Since bit-plane swapping can be performed in every block along with bit-plane compression, it is only necessary to consider the time complexity of block rearrangement as $O(\lfloor m/t \rfloor \times \lfloor n/t \rfloor)$. In our method, image encryption/decryption is an exclusive or (XOR) operation between the image and a pseudo-random matrix that needs constant time. The data hiding and data extraction are inverse processes of each other and the worst-case time complexity is $O(8\lfloor m/t \rfloor \times \lfloor n/t \rfloor)$. The image recovery is equivalent to the inverse process of (1) and (2), so its time complexity is $O(9\lfloor m/t \rfloor \times \lfloor n/t \rfloor)$. The proposed method defines and stores some auxiliary information throughout the process, and the worst-case space complexity due to the capacity of the image is $O(8mn)$. To facilitate comparison with other methods, n is used to denote the size of the input data scale, the results are shown in Table VIII. It is obvious that the time complexities of [19], [22] are higher than other methods and the space complexities are the same.

V. CONCLUSION

In this article, a secure and high-capacity RRBE method is proposed by compressing global zero-valued high bit-planes and adaptively allocating different Huffman indicators. The bit-plane swapping and block rearrangement are utilized to cluster all embeddable bit-planes together so that most auxiliary information can be encrypted so as to enhance security while facilitating data embedding and data extraction. The experimental results show that not only the original image can be restored losslessly and the embedded data can be correctly extracted for the receiver, respectively, but also the security and payload of the proposed method outperform that of the state-of-the-art methods.

In the future, we will aim to improve the accuracy of predicted value using other methods such as deep learning, and we will try to seek new ways to effectively compress the length of auxiliary information so as to increase the payload while guaranteeing security.

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] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Process.*, vol. 13, no. 8, pp. 1147–1156, May 2004.
[3] Y. Qiu, Z. Qian, and L. Yu, "Adaptive reversible data hiding by extending the generalized integer transformation," *IEEE Signal Process. Lett.*, vol. 23, no. 1, pp. 130–134, Jan. 2016.
[4] 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.
[5] G. Coatrieux, W. Pan, N. Cuppens-Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 1, pp. 111–120, Jan. 2013.
[6] D. M. Thodi and J. J. Rodríguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, May 2007.
[7] 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.
[8] Q. Chang, X. Li, and Y. Zhao, "Reversible data hiding for color images based on adaptive three-dimensional histogram modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 9, pp. 5725–5735, Sep. 2022.
[9] Q. Chang, X. Li, Y. Zhao, and R. Ni, "Adaptive pairwise prediction-error expansion and multiple histograms modification for reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4850–4863, Dec. 2021.
[10] T. Zhang et al., "Adaptive reversible data hiding with contrast enhancement based on multi-histogram modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 8, pp. 5041–5054, Aug. 2022.

[11] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "General framework to reversible data hiding for JPEG images with multiple two-dimensional histograms," *IEEE Trans. Multimedia*, early access, Aug. 18, 2022, doi: 10.1109/TMM.2022.3198877.
[12] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "Reversible data hiding for JPEG images with adaptive multiple two-dimensional histogram and mapping generation," *IEEE Trans. Multimedia*, early access, Feb. 01, 2023, doi: 10.1109/TMM.2023.3241541.
[13] G. Coatrieux, C. Le Guillou, J.-M. Cauvin, and C. Roux, "Reversible watermarking for knowledge digest embedding and reliability control in medical images," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 2, pp. 158–165, Mar. 2009.
[14] S. Lee, C. D. Yoo, and T. Kalker, "Reversible image watermarking based on integer-to-integer wavelet transform," *IEEE Trans. Inf. Forensic Secur.*, vol. 2, no. 3, pp. 321–330, Sep. 2007.
[15] R. Y. M. Li, O. C. Au, C. K. M. Yuk, S.-K. Yip, and T.-W. Chan, "Enhanced image trans-coding using reversible data hiding," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2007, pp. 1273–1276.
[16] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Trans. Inf. Forensic Secur.*, vol. 8, no. 3, pp. 553–562, Mar. 2013.
[17] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Process.*, vol. 94, pp. 118–127, 2014.
[18] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1132–1143, May 2016.
[19] P. Puteaux and W. Puech, "EPE-based huge-capacity reversible data hiding in encrypted images," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, 2018, pp. 1–7.
[20] Y. Puyang, Z. Yin, and Z. Qian, "Reversible data hiding in encrypted images with two-MSB prediction," in *Proc. IEEE Int. Workshop Inf. Forensics Secur.*, 2018, pp. 1–7.
[21] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multim.*, vol. 21, no. 1, pp. 51–64, Sep. 2019.
[22] K. Chen and C.-C. Chang, "High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement," *J. Vis. Commun. Image Representation*, vol. 58, pp. 334–344, 2019.
[23] Y. Wu, Y. Xiang, Y. Guo, J. Tang, and Z. Yin, "An improved reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multimedia*, vol. 22, no. 8, pp. 1929–1938, Aug. 2020.
[24] Z. Yin, Y. Xiang, and X. Zhang, "Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding," *IEEE Trans. Multimedia*, vol. 22, no. 4, pp. 874–884, Apr. 2020.
[25] A. Mohammadi, M. Nakhkash, and M. A. Akhaee, "A high-capacity reversible data hiding in encrypted images employing local difference predictor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2366–2376, Aug. 2020.
[26] Z. Yin, Y. Peng, and Y. Xiang, "Reversible data hiding in encrypted images based on pixel prediction and bit-plane compression," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 992–1002, Mar. 2022.
[27] C. Yu, X. Zhang, X. Zhang, G. Li, and Z. Tang, "Reversible data hiding with hierarchical embedding for encrypted images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 451–466, Feb. 2022.
[28] S. Xu, J.-H. Horng, C.-C. Chang, and C.-C. Chang, "Reversible data hiding with hierarchical block variable length coding for cloud security," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 5, pp. 4199–4213, Sep./Oct. 2023.
[29] G. Gao, L. Zhang, Y. Lin, S. Tong, and C. Yuan, "High-performance reversible data hiding in encrypted images with adaptive Huffman code," *Digit. Signal Process.*, vol. 133, 2023, Art. no. 103870.
[30] L. Qu, F. Chen, S. Zhang, and H. He, "Cryptanalysis of reversible data hiding in encrypted images by block permutation and co-modulation," *IEEE Trans. Multimedia*, vol. 24, pp. 2924–2937, 2022.
[31] Y. Wang, Z. Cai, and W. He, "High capacity reversible data hiding in encrypted image based on intra-block lossless compression," *IEEE Trans. Multimedia*, vol. 23, pp. 1466–1473, 2021.
[32] Y. Wang and W. He, "High capacity reversible data hiding in encrypted image based on adaptive MSB prediction," *IEEE Trans. Multimedia*, vol. 24, pp. 1288–1298, 2022.
[33] C. Yu, X. Zhang, G. Li, S. Zhan, and Z. Tang, "Reversible data hiding with adaptive difference recovery for encrypted images," *Inf. Sci.*, vol. 584, pp. 89–110, 2022.
[34] W. Hong, T.-S. Chen, and H.-Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Process. Lett.*, vol. 19, no. 4, pp. 199–202, Apr. 2012.
[35] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Trans. Inf. Forensic Secur.*, vol. 7, no. 2, pp. 826–832, Apr. 2012.
[36] X. Liao and C. Shu, "Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels," *J. Vis. Commun. Image Representation*, vol. 28, pp. 21–27, 2015.
[37] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000.
[38] P. Bas, T. Filler, and T. Pevný, "Break our steganographic system the ins and outs of organizing BOSS," in *Proc. 13th Int. Conf. Inf. Hiding*, 2011, pp. 59–70.
[39] P. Bas and T. Furon, "Image database of BOWS-2," 2017. Accessed: Jun. 20, 2017. [Online]. Available: http://bows2.ec-lille.fr

Ye Yao received the M.S. degree in computer science and the Ph.D. degree in communication and information systems from Wuhan University, Wuhan, China, in 2005 and 2008, respectively. From 2016 to 2017, he was a Visiting Scholar with the New Jersey Institute of Technology, Newark, NJ, USA. He is currently an Associate Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou. His research interests include multimedia forensics and information security.

Ke Wang received the B.S. degree from Southwest Minzu University, Chengdu, China, in 2022. He is currently working toward the master's degree with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His research interests include image processing and information hiding.

Qi Chang received the Ph.D. degree from Beijing Jiaotong University, Beijing, China, in 2022. In 2018, she was a Visiting Ph.D. student with the National Institute of Informatics, Tokyo, Japan. She is currently a Lecturer with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. Her research interests include image processing and information hiding.

Shaowei Weng (Member, IEEE) received the Ph.D. degree from the Institute of Information Science, Beijing Jiaotong University, Beijing, China, in 2009. She is currently a Professor with the School of Electronic, Electrical Engineering and Physics, Fujian University of Technology, Fuzhou, China. She is also in charge of two Natural Science Foundation of China Projects. She has authored or coauthored more than 60 articles and applies six national patents. Her research interests include image processing, data hiding and digital watermarking, pattern recognition, and computer vision.