Reversible Data Hiding With Hierarchical Embedding for Encrypted Images

Chunqiang Yu, *Member, IEEE*, Xianquan Zhang, Xinpeng Zhang, *Member, IEEE*, Guoxiang Li, and Zhenjun Tang, *Member, IEEE*

***Abstract*— Reversible data hiding in encrypted images (RDHEI) is an effective technique of data security. Most state-of-the-art RDHEI methods do not achieve desirable payload yet. To address this problem, we propose a new RDHEI method with hierarchical embedding. Our contributions are twofold. (1) A novel technique of hierarchical label map generation is proposed for the bit-planes of plaintext image. The hierarchical label map is calculated by using prediction technique, and it is compressed and embedded into the encrypted image. (2) Hierarchical embedding is designed to achieve a high embedding payload. This embedding technique hierarchically divides prediction errors into three kinds: small-magnitude, medium-magnitude, and large-magnitude, which are marked by different labels. Different from the conventional techniques, pixels with small-magnitude/large-magnitude prediction errors are both used to accommodate secret bits in the hierarchical embedding technique, and therefore contribute a high embedding payload. Experiments on two standard datasets are discussed to validate the proposed RDHEI method. The results demonstrate that the proposed RDHEI method outperforms some state-of-the-art RDHEI methods in payload. The average payloads of the proposed RDHEI method are 3.4568 bpp and 3.6823 bpp for BOWS-2 dataset and BOSSbase dataset, respectively.**

***Index Terms*— Reversible data hiding, encrypted images, lossless compression, hierarchical embedding, high payload.**

I. Introduction

With the rapid development of cloud computing and cloud storage, the amount of multimedia data uploaded to cloud server has increased rapidly. Meanwhile, multimedia data is confronted with security issues, such as confidentiality, authentication and integrity. Therefore, protection of multimedia data has become an important task [1]–[5]. Currently, data hiding and data encryption are two effective techniques adopted to address the protection of multimedia data. This work mainly focuses on data hiding technique.

Data hiding system consists of two entities, namely cover image and secret data. The conventional data hiding aims to embed secret data into cover image with small distortion. It focuses on correct extraction of secret data, but ignores recovery of original cover image. However, in some applications, such as medical image processing and law enforcement, the original cover image is required to be recovered without error. For this demand, reversible data hiding (RDH) is proposed to recover both secret data and cover image losslessly. Most existing RDH algorithms are mainly based on lossless compression [6], [7], difference expansion (DE) [8], [9], histogram shifting (HS) [10]–[12] and prediction error expansion (PEE) [13]–[19]. These RDH algorithms usually exploit spatial correlation among local pixels to embed secret data, such as the statistical or prediction errors of pixel pairs. These algorithms can be used in plaintext images, but they are not suitable for encrypted images because pixels of encrypted image are uncorrelated.

In recent years, researchers proposed RDH algorithms for the application of encrypted images. Generally, RDH in encrypted image (RDHEI) aims to protect both the original images and secret data simultaneously. There are three users in RDHEI: content-owner, data-hider and receiver. A content owner encrypts original image according to encryption key and uploads it to the server. The data-hider embeds secret data into the encrypted image by using data-hiding key and he/she cannot access original image content without encryption key. The receiver is able to access original image, secret data, or both under specific authority. There are mainly two RDHEI frameworks, namely reserving room before encryption (RRBE) [20]–[27], vacating room after encryption (VRAE) [28]–[42]. A good RDHEI is expected to make a trade-off among three performances, i.e., payload, error rate of the extracted bits, and visual quality of the reconstructed image. However, most previous methods have limitations in these performances, such as low payload, errors in data extraction or image reconstruction, unsatisfied image quality under high embedding payload.

To address the above problems, we propose a new RDHEI method based on hierarchical embedding. The proposed RDHEI method is error-free and can reach a high payload. The main contributions of our work are summarized as follows.

Manuscript received December 26, 2020; revised February 9, 2021 and February 24, 2021; accepted February 25, 2021. Date of publication March 1, 2021; date of current version February 4, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62062013, Grant 61962008, Grant U1936214, Grant 61762017, and Grant 61562007; in part by the Guangxi "Bagui Scholar" Team for Innovation and Research; in part by the Guangxi Talent Highland Project of Big Data Intelligence and Application; and in part by the Natural Science Project of Guangxi Universities under Grant 2021KY0051. This article was recommended by Associate Editor X. Cao. *(Corresponding authors: Xianquan Zhang; Zhenjun Tang.)*

Chunqiang Yu, Xianquan Zhang, Guoxiang Li, and Zhenjun Tang are with the Guangxi Key Laboratory of Multi-Source Information Mining & Security, Guangxi Normal University, Guilin 541004, China, and also with the Department of Computer Science, Guangxi Normal University, Guilin 541004, China (e-mail: yu_chunqiang@126.com; zxq6622@163.com; masterlgx@163.com; tangzj230@163.com).

Xinpeng Zhang is with the School of Computer Science, Fudan University, Shanghai 200433, China (e-mail: zhangxinpeng@fudan.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2021.3062947.

Digital Object Identifier 10.1109/TCSVT.2021.3062947

(1) A novel technique of hierarchical label map generation is proposed for the bit-planes of plaintext image. The hierarchical label map is calculated before image encryption by using prediction technique. During data embedding, it is firstly compressed and then embedded into the encrypted image. In addition, it is also used to assist exact data extraction and image recovery.

(2) Hierarchical embedding is designed to achieve a high embedding payload meanwhile full reversibility is guaranteed. The presented embedding technique hierarchically divides prediction errors into three kinds: small-magnitude, medium-magnitude, and large-magnitude, which are marked by different labels. Conventional methods usually exploit those pixels with small-magnitude prediction errors to conduct data hiding and discard the pixels with medium-magnitude or large-magnitude prediction errors. Different from the previous techniques, the pixels with small-magnitude prediction errors and the pixels with large-magnitude prediction errors are both used to accommodate secret bits in our hierarchical embedding technique, and therefore contribute a high embedding payload.

(3) Many experiments on two standard datasets are discussed to validate the proposed RDHEI method. The results demonstrate that the proposed RDHEI method outperforms some state-of-the-art RDHEI methods in payload. The average payloads of the proposed RDHEI method are 3.4568 bpp and 3.6823 bpp for BOWS-2 dataset and BOSSbase dataset, respectively.

The rest of this paper is organized as follows. Section II reviews the related work of current popular RDHEI methods. Section III explains the proposed RDHEI method in detail. Section IV discusses the experimental results and performance comparisons. Section V concludes this paper.

II. Related Work

State-of-the-art RDHEI methods can be divided into two categories. They are RRBE (Reserving Room Before Encryption) and VRAE (Vacating Room After Encryption). RRBE based methods take advantage of redundant information in plaintext domain to obtain embedding room before image encryption. Prior to encryption, the original image is pre-processed by the content owner to release some room to accommodate data. For example, a pioneer RRBE method is proposed by Ma *et al.* [20]. In their method, the image is first divided into two parts and then the LSB-planes of one part is embedded into the other part by using a standard RDH method to achieve image self-embedding before encryption. The released room of LSB-planes can accommodate secret data. Zhang *et al.* [21] used an estimation technique to reserve room before encryption. A small portion of the pixels is estimated via a large portion of pixels in the original image. Then secret data can be embedded by modifying the estimation errors. Cao *et al.* [22] adopted patch-level representation to reserve room for embedding data. They used sparse coefficients to represent the original image according to an over-complete dictionary. In addition, the generated residual errors are encoded and then self-embedded into the cover image. The dictionary is required to be embedded into the encrypted image. Due to the powerful ability of sparse coding representation, a large amount of embedding room can be reserved. Zhang *et al.* [23] took advantage of a public-key cryptosystem with probabilistic and homomorphic properties to design a lossless data hiding method and an RDH method. In the reversible method, the histogram of the original image is firstly shrunk before encryption and then the original image is encrypted with an additive homomorphic cryptosystem. The encrypted image is marked with the bits of the secret message and error-correction codes. In the lossless method, the data embedding is then performed using multilayer wet paper coding. In [24], a pair of adjacent original pixels is transformed into a pair of odd or even pixels using DE technique [8], and then the transformed pixel pair is encrypted by Paillier encryption with additive homomorphism. One secret bit is added on one encrypted pixel of pair. During extraction, if the pair of pixels are both even or odd, a bit "0" can be extracted. Otherwise, a bit "1" can be obtained. Their payload is closed to 0.5 bpp. However, the use of Paillier encryption suffers from data expansion and brings about large storage cost. In [25], the original image is pre-processed by prediction technique before encryption. Secret data is embedded into the encrypted image with additive homomorphism. Data extraction and image recovery can be performed according to additive homomorphism. To further improve embedding capacity, some researchers introduced compression technique into RDHEI. For example, Yi and Zhou [26] proposed a binary-block embedding (BBE) method for binary image data hiding which efficiently encodes sparse blocks to reserve room. And they applied BBE to image with 8 bit-planes. The low bit-planes are embedded into the high bit-planes so that the room of low bit-planes is released to accommodate secret data. In [27], a block-based MSB plane rearrangement (BMPR) scheme is designed to transform the MSB planes of the original image into high compressible bit streams before encryption, and then an extended run-length coding is used to compress the transformed bit streams with high compression ratio so as to vacate a large amount of room. The RRBE-based methods can achieve good embedding performance, but they require extra pre-processing before image encryption, which increases computational cost for the content owner.

VARE based methods use the standard image encryption algorithm, such as AES, RC4 encryption or specific encryption algorithms, to directly encrypt the original image and do not require extra pre-processing before image encryption. After encryption, data hider modifies the encrypted pixels to embed secret data. Early VARE based methods directly encrypt the original image with stream cipher. For example, in [28], [29], the original image is firstly encrypted by using the stream cipher according to the encryption key and then the encrypted image is divided into several blocks. Three least significant bits (LSB) of a half of pixels of each block are flipped to embed one secret bit. The receiver can extract secret data and recover image by using a fluctuation function. These algorithms may suffer from incorrect data extraction and image recovery, especially for non-smoothness blocks when the block size is relatively small. To reduce extraction error rate, Liao and Shu [30] exploited the locations of different pixels of

a block to design a new measure of block complexity for data extraction and image restoration. Qin and Zhang [31] flipped the LSBs of fewer pixels by an elaborate selection mechanism so as to improve visual quality of decrypted image. In the above methods, encryption key and data-hiding key must be both known during data extraction and image recovery. In order to separate data extraction and image recovery, Zhang [32] first introduced a separable RDHEI method. An original image is encrypted by using the stream cipher. The LSBs of the encrypted image are compressed to vacate room for data embedding. Image decryption and data extraction can be performed independently. To increase the size of embedding room, distributed source coding is employed to obtain compression room of encrypted image for data embedding [33]. Wu and Sun [34] embedded secret bits into the encrypted image with most significant bit (MSB) replacement technique. At the receivers side, prediction technique is utilized to extract data and recover original image.

Although stream cipher is able to provide confidentiality for original image, it disorganizes spatial pixel correlation and there is almost no redundant information in the encrypted image. Consequently, it is quite difficult to vacate room from the encrypted image created by stream cipher. To make high embedding capacity, some researchers are inclined to design specific encryption which can both provide confidentiality for original image and remain the redundancy within the encrypted image. Since spatial redundancy exists in the encrypted image, specific encryption based method can achieve better embedding performance than the stream cipher based VARE method. For example, Huang *et al.* [35] proposed a new framework for RDHEI. The encryption in this framework consists of specific stream encryption and block permutation. The original image is divided into several blocks and the blocks are scrambled to change pixel position. Then all pixels of block are encrypted by stream encryption using the same key to change pixel values so that the redundancy within each block is available. By this encryption, difference histogram or prediction error histogram can be generated for data hiding. Di *et al.* [36] adopted the encryption framework [35] to encrypt original image. Bit-planes of encrypted pixels are divided into two components and the HS is adopted for data hiding in these two components according to the bit-plane parameter. In [37], the original image is divided into blocks sized 2×2 and all pixels of block are encrypted by mod 256 with the same key. The pixel value ordering (PVO) strategy is exploited for data hiding in each block. Tang *et al.* [38] adopt block-based encryption to transfer spatial correlation between neighboring pixels of plaintext image into the encrypted image. The differences among encrypted pixels are compressed to vacate room for data hiding. In [39], block permutation and encryption with stream cipher are used to encrypt original image by content owner. With redundant information of MSB bit-planes, adaptive block encoding is then conducted to vacate room according to occurrence frequency of MSB and Huffman coding. In [40], [41], redundant space of the original image is transferred to the encrypted image by disordering bit-planes while pixel values are changed for confidentiality. Then block permutation is utilized to enhance security further. Due to redundant space within the encrypted image, image blocks can be compressed by efficient sparse code to vacate room for data hiding. Yi and Zhou [42] introduced a data embedding method by using parametric binary tree labeling scheme (PBTL-DE) and applied the PBTL-DE to the encrypted image generated by block permutation and block encryption with same key for achieving high payload.

Recently, researchers pay much attention to RDHEI methods with high payload [43]–[47]. These methods aim to achieve perfect reversibility and a lossless message extraction when embedding data with high payload. For example, Puteaux and Puech [43] used MSB instead of LSB to embed a secret message. The MSB of original pixel is flipped to generate its inverse value. The absolute difference between prediction value and original pixel value is calculated. And the absolute difference between prediction value and its inverse value is also calculated. If the former is less than the later, there is no prediction error and the MSB can accommodate one secret bit. Otherwise, the prediction is wrong and the pixel needs to be marked by location map. Since only MSB is available, their payload is only close to 1 bpp. Yin *et al.* [44] compared MSBs between prediction value and original pixel value, and determined the number of the consecutive MSBs with the same value. This number is regarded as the label of the current pixel. Huffman coding is then used to compress label map, and secret bits can be embedded into the MSBs with specific label for encrypted pixel. In fact, for some scenarios, the prediction value and original pixel value are very closed, but the number of consecutive same MSBs between is limited. Although their EC can reach 3.361 bpp on the BOSSBase dataset, they do not make use of those pixels close to their prediction values. Wu *et al.* [45] extended the block based PBTL-DE method [42] to the entire image to achieve high payload. Puteaux and Puech [46] recursively processed each bit-plane of an image from MSB to LSB by combining error prediction, reversible adaptation, encryption and embedding. Correspondingly, each bit-plane must be recursively reconstructed from the LSB-plane to the MSB-plane. Mohammadi *et al.* [47] proposed a smart RDHEI method with high payload. They divided the original image into blocks and calculated prediction errors between the reference pixel and the other pixels in each block. According to the range of prediction error, the payload of each pixel can be determined. The minimum payload of each block is regarded as it label. With the assistance of block labels, data extraction and image recovery can be conducted perfectly.

Table I summarizes payload performance of some typical RDHEI methods. It can be seen that the methods [42], [44]–[47] can achieve high payload. But their bit-plane utilizations are still not good enough. To better take advantage of bit-planes, this paper elaborates hierarchical label map generation for bit-planes so as to increase payload and ensure perfect reversibility.

III. Proposed Method

In this paper, we propose a new RDHEI method with a high payload based on hierarchical embedding which is able to

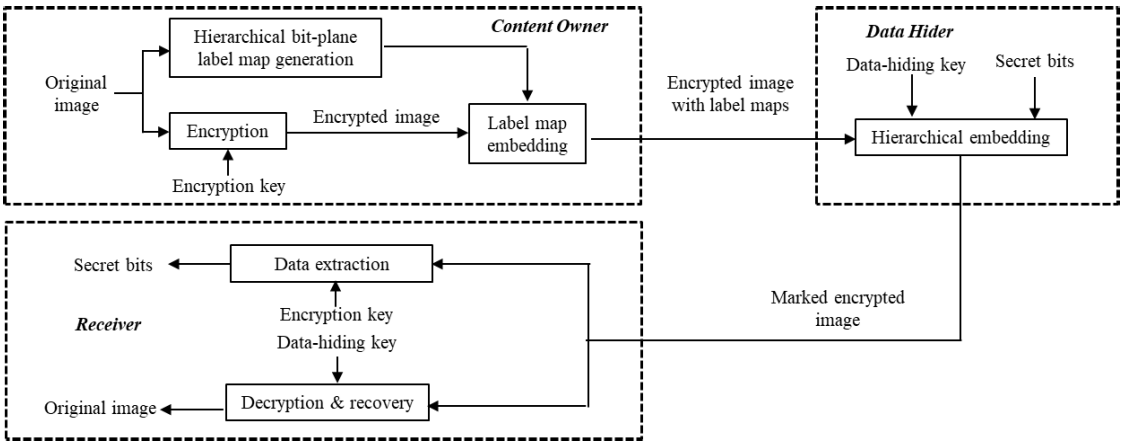

Fig. 1. Block diagram of the proposed method.

TABLE I
PAYLOAD PERFORMANCE

Payload	Method
Low	[28], [29], [30], [31], [32]
Moderate	[20], [21], [22], [23], [24], [26]
High	[27], [42], [44], [45], [46], [47]

perform data extraction and image recovery without any error. As shown in Fig.1, the proposed method contains three stages: 1) The content-owner conducts image encryption with encryption key; 2) The data-hider performs data hiding according to data-hiding key; 3) Data extraction and image decryption are done by the receiver. Previous to image encryption, a label map for each bit-plane of the original image is calculated from 8^{th} bit-plane to 2^{th} bit-plane hierarchically and this label map is also called bit-plane label map. After encryption, each bit-plane label map is compressed by using arithmetic coding to reduce its size and then the compressed version is embedded into its corresponding bit-plane of the encrypted image. In the embedding stage, secret bits are hierarchically embedded into bit-planes of the encrypted image from 8^{th} bit-plane to 2^{th} bit-plane by bit replacement technique according to bit-plane label maps. Finally, data extraction and image recovery can be performed separately according to the knowledge of data-hiding key and encryption key, and the extracted data and the restored image are both error-free.

A. Hierarchical Bit-Plane Label Map Generation

In this section, we first elaborate a hierarchical prediction error magnitude division strategy. In each layer of prediction error division, the corresponding bit-plane is marked by a label according to magnitude of prediction error. With the proposed division strategy, we design a hierarchical bit-plane label map mechanism for multiple bit-plane modification. According to hierarchical bit-plane label maps, the multiple modified bit-planes of the original image can be recovered so as to recover image losslessly.

Let $\mathbf{I}$ be an 8-bit grayscale uncompressed image with $H \times W$ size and $p_{i,j}$ denote its pixel located at the coordinates (i, j), where $0 \le p_{i,j} \le 255$, $1 \le i \le H$, and $1 \le i \le W$. It is clear that a pixel consists of 8 bit-planes. Suppose that 8 bit-planes of $p_{i,j}$ are denoted by $b_{i,j,1}, b_{i,j,2}, \ldots, b_{i,j,8}$, where $b_{i,j,8}$ is the most significant bit and $b_{i,j,1}$ is the least significant bit. These bit values can be derived as follows.

$$b_{i,j,k} = \lfloor \frac{p_{i,j}}{2^{k-1}} \rfloor \bmod 2, \quad k = 1, 2, \ldots, 8 \tag{1}$$

where mod and $\lfloor . \rfloor$ denote the modulo operation and the rounding down operation, respectively. Correspondingly, $p_{i,j}$ can be calculated as

$$p_{i,j} = \sum_{k=1}^{8} b_{i,j,k} \times 2^{k-1} \tag{2}$$

Since there is high correlation in local pixels of the original image, pixels can be predicted by their context pixels. Here, we exploit Median Edge Detection (MED) predictor [48] to conduct prediction due to the reasons that the MED predictor has been successfully used in the well-known compression standard of JPEG-LS and it can make a good balance between predictor performance and computational cost. We take $p_{i,j}$ to illustrate pixel prediction. As shown in Fig. 2, the prediction value of $p_{i,j}$ is derived as

$$\hat{p}_{i,j} = \begin{cases} \max\left(p_{i-1,j}, p_{i,j-1}\right), \\ \qquad p_{i-1,j-1} \le \min\left(p_{i-1,j}, p_{i,j-1}\right) \\ \min\left(p_{i-1,j}, p_{i,j-1}\right), \\ \qquad p_{i-1,j-1} \ge \max\left(p_{i-1,j}, p_{i,j-1}\right) \\ p_{i-1,j} + p_{i,j-1} - p_{i-1,j-1}, \\ \qquad \text{Otherwise} \end{cases} \tag{3}$$

where $2 \le i \le H$ and $2 \le j \le W$. Next, the absolute value of prediction error is calculated by the below equation.

$$|e_{i,j}| = |p_{i,j} - \hat{p}_{i,j}| \tag{4}$$

After obtaining $|e_{i,j}|$, we divide $|e_{i,j}|$ with coarse-magnitude $0 \le |e_{i,j}| \le 2^r - 1$ into fine-magnitudes, where $7 \ge r \ge 1$. According to the fine-magnitudes, we mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ with the corresponding label and determine whether or not it can accommodate secret bit.

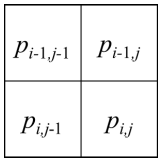

Fig. 2. The pixel $p_{i,j}$ and its context pixels.

Specifically, $|e_{i,j}|$ is divided finely into three magnitudes as follows.

Case 1: A small magnitude, i.e., $0 \le |e_{i,j}| \le 2^{r-1} - 1$. In this case, a label $s_{i,j,r} = 0$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane can accommodate secret bit.

Case 2: A medium magnitude, i.e., $|e_{i,j}| = 2^{r-1}$. In this case, a label $s_{i,j,r} = 2$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane cannot accommodate secret bit due to error recovery.

Case 3: A large magnitude, i.e., $2^{r-1} + 1 \le |e_{i,j}| \le 2^r - 1$. In this case, a label $s_{i,j,r} = 1$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane can accommodate secret bit.

Note that the range expression of prediction error in [47] is similar with our expression. In [47], the range expression of prediction error is exploited to determine the payload of a pixel, and only one label is used to mark a pixel. Different from [47], we further divide the prediction error into three cases: small magnitude, medium magnitude and large magnitude, and use several labels to mark a pixel according to its specific case. Compared with [47], our strategy of several labels for a pixel can provide a large embedding rate. For example, [47] reported that its average embedding rate on BOWS-2 database is 2.9 bpp when the block size is 3×3. But the average embedding rate of the proposed method on BOWS-2 database reaches 3.4568 bpp (See Fig. 9 (b)).

Next we explain why the $(r+1)^{th}$ bit-plane of $p_{i,j}$ with label "0" or "1" can be used for data hiding while the one with label "2" cannot be used. Without loss of generality, bit-plane replacement technique is adopted to embed secret bits which will modify bit-planes of the pixel. Suppose that the $8^{th} \sim (r+1)^{th}$ bit-planes of $p_{i,j}$ are replaced by $8-r$ secret bits and $r^{th} \sim 1^{st}$ bit-planes keep unchanged. Clearly, there are 2^{8-r} modification combinations for $8^{th} \sim (r+1)^{th}$ bit-planes. Suppose that the modified $8^{th} \sim (r+1)^{th}$ bit-planes and the pixel after modification are denoted as $b'_{i,j,k}\,(r+1 \le k \le 8)$ and $p'_{i,j}$, respectively. We have

$$\begin{aligned} p'_{i,j} &= \sum_{k=1}^{r} \left(b_{i,j,k} \times 2^{k-1} \right) + \sum_{k=r+1}^{8} \left(b'_{i,j,k} \times 2^{k-1} \right) \\ &= p_{i,j} + \sum_{k=r+1}^{8} \left(a_{i,j,k} \times 2^{k-1} \right) \end{aligned} \tag{5}$$

where $a_{i,j,k}$ indicates whether or not the bit-plane $b_{i,j,k}$ is modified. If $b'_{i,j,k} = b_{i,j,k}$, it means that this bit-plane remains unchanged and let $a_{i,j,k} = 0$. If $\begin{cases} b_{i,j,k} = 1 \\ b'_{i,j,k} = 0 \end{cases}$, $b_{i,j,k} \times 2^{k-1}$ is subtracted from $p_{i,j}$ and let $a_{i,j,k} = -1$. If $\begin{cases} b_{i,j,k} = 0 \\ b'_{i,j,k} = 1 \end{cases}$, $b'_{i,j,k} \times 2^{k-1}$ is added to $p_{i,j}$ and let $a_{i,j,k} = 1$. For clarity, let $m = p'_{i,j} - p_{i,j} = \sum_{k=r+1}^{8} \left(a_{i,j,k} \times 2^{k-1} \right)$. After dealing with the $8^{th} \sim (r+1)^{th}$ bit-planes, $p'_{i,j}$ is generated according to the Eq. (5). The absolute value of prediction error after bit-plane modification is derived as

$$\begin{aligned} \left| e'_{i,j} \right| &= \left| p'_{i,j} - \hat{p}_{i,j} \right| \\ &= \left| p_{i,j} + \sum_{k=r+1}^{8} \left(a_{i,j,k} \times 2^{k-1} \right) - \hat{p}_{i,j} \right| \\ &= \left| e_{i,j} + m \right| \end{aligned} \tag{6}$$

If $\sum_{k=r+1}^{8} \left| a_{i,j,k} \right| = 0$, it means that all bit-planes are not modified. Otherwise, there is at least one modified bit-plane and the original pixel value is changed. Next, we only analyze $\left| e'_{i,j} \right|$ with $\sum_{k=r+1}^{8} \left| a_{i,j,k} \right| > 0$. Namely, at least one bit-plane among $8^{th} \sim (r+1)^{th}$ bit-planes of $p_{i,j}$ is modified. Clearly, there are two kinds of m values, i.e., positive value and negative value. For the positive m value, its maximum value is $m = \sum_{k=r+1}^{8} 2^{k-1}$ when all bit-planes are changed from 0 to 1 and its minimum value is $m = 2^r$ when the $(r+1)^{th}$ bit-plane is changed from 0 to 1. Therefore, for positive value, the range of m is $[2^r, \sum_{k=r+1}^{8} 2^{k-1}]$. Similarly, for the negative m value, its maximum value is $m = -2^r$ when the $(r+1)^{th}$ bit-plane is changed from 1 to 0 and its minimum value is $m = -\sum_{k=r+1}^{8} 2^{k-1}$ when all bit-planes are changed from 1 to 0. Therefore, the range of the negative m is $[-\sum_{k=r+1}^{8} 2^{k-1}, -2^r]$. Consequently, $|m| \ge 2^r > \left| e_{i,j} \right|$ due to $0 \le \left| e_{i,j} \right| \le 2^r - 1$. Thus, the Eq. (6) can be revised as

$$\begin{aligned} e'_{i,j} &= \left| e_{i,j} + m \right| \\ &= \begin{cases} |m| + \left| e_{i,j} \right|, & \text{if } e_{i,j} \ge 0 \text{ and } m \ge 2^r \\ |m| + \left| e_{i,j} \right|, & \text{if } e_{i,j} < 0 \text{ and } m \le -2^r \\ |m| - \left| e_{i,j} \right|, & \text{if } e_{i,j} \ge 0 \text{ and } m \le -2^r \\ |m| - \left| e_{i,j} \right|, & \text{if } e_{i,j} < 0 \text{ and } m \ge -2^r \end{cases} \end{aligned} \tag{7}$$

According to the above equation, we have two possible values of $e'_{i,j}$, namely, $e'_{i,j} = |m| + \left| e_{i,j} \right|$ or $e'_{i,j} = |m| - \left| e_{i,j} \right|$. Next, we discuss label generation of $(r+1)^{th}$ bit-plane for different cases.

Case 1: $0 \le \left| e_{i,j} \right| \le 2^{r-1} - 1$

If $e'_{i,j} = |m| + \left| e_{i,j} \right|$, we have

$$\begin{aligned} \left| e'_{i,j} \right| &= \left| e_{i,j} \right| + |m| \\ &\ge \left| e_{i,j} \right| + 2^r \\ &\ge 2^r \\ &> 2^{r-1} - 1 \end{aligned} \tag{8}$$

If $e'_{i,j} = |m| - \left| e_{i,j} \right|$, we have

$$\begin{aligned} \left| e'_{i,j} \right| &= |m| - \left| e_{i,j} \right| \\ &\ge 2^r - \left| e_{i,j} \right| \\ &\ge 2^r - (2^{r-1} - 1) \end{aligned}$$

$$\begin{aligned} &\geq 2^{r-1}+1 \\ &> 2^{r-1}-1 \end{aligned} \tag{9}$$

Clearly, when $e'_{i,j} = |m| + |e_{i,j}|$ or $e'_{i,j} = |m| - |e_{i,j}|$, we both have $|e'_{i,j}| > 2^{r-1}-1$. As long as one bit-plane is modified at least, the absolute value of the prediction error generated by Eq. (6), namely $|e'_{i,j}|$, will be not within the range $\left[0, 2^{r-1}-1\right]$. Consequently, the original pixel can be recovered by searching a unique combination for $8^{th} \sim (r+1)^{th}$ bit-planes such that the prediction error is within the range $\left[0, 2^{r-1}-1\right]$. Consequently, the $(r+1)^{th}$ bit-plane of $p_{i,j}$ with $0 \leq |e_{i,j}| \leq 2^{r-1}-1$ can be modified for data hiding and this bit-plane is marked with the label $s_{i,j,r} = 0$.

Case 2: $\left|e_{i,j}\right| = 2^{r-1}$

If $e'_{i,j} = |m| + \left|e_{i,j}\right|$, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= \left|e_{i,j}\right| + |m| \\ &\geq \left|e_{i,j}\right| + 2^r \\ &\geq 2^{r-1} + 2^r \\ &> 2^{r-1} \end{aligned} \tag{10}$$

However, if $e'_{i,j} = |m| - \left|e_{i,j}\right|$, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= |m| - \left|e_{i,j}\right| \\ &\geq 2^r - 2^{r-1} \\ &\geq 2^{r-1} \end{aligned} \tag{11}$$

Clearly, when $e_{i,j} = 2^{r-1}$ and $e'_{i,j} = 2^{r-1}$, the $(r+1)^{th}$ bit-plane cannot be recovered losslessly if it is modified. Consequently, the $(r+1)^{th}$ bit-plane of $p_{i,j}$ with $e_{i,j} = 2^{r-1}$ cannot accommodate secret bit and this bit-plane is marked with the label $s_{i,j,r} = 2$.

Case 3: $2^{r-1}+1 \leq \left|e_{i,j}\right| \leq 2^r - 1$

If $e'_{i,j} = |m| + \left|e_{i,j}\right|$, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= |m| + \left|e_{i,j}\right| \\ &\geq 2^r + \left|e_{i,j}\right| \\ &\geq 2^r + 2^{r-1} + 1 \\ &> 2^r - 1 \end{aligned} \tag{12}$$

If $e'_{i,j} = |m| - \left|e_{i,j}\right|$, we classify $|m|$ into two types, namely, $|m| = 2^r$ and $|m| \geq 2^{r+1}$. When $|m| = 2^r$, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= |m| - \left|e_{i,j}\right| \\ &\leq 2^r - (2^{r-1}+1) \\ &\leq 2^{r-1} - 1 \\ &< 2^{r-1} + 1 \end{aligned} \tag{13}$$

When $|m| \geq 2^{r+1}$, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= |m| - \left|e_{i,j}\right| \\ &\geq 2^{r+1} - \left|e_{i,j}\right| \\ &\geq 2^{r+1} - \left(2^r - 1\right) \\ &\geq 2^r + 1 \\ &> 2^r - 1 \end{aligned} \tag{14}$$

According to the Eqs. (12-14), we always have $e'_{i,j} > 2^r - 1$ or $e'_{i,j} < 2^{r-1} + 1$ when $e'_{i,j} = |m| + \left|e_{i,j}\right|$ or $e'_{i,j} = |m| - \left|e_{i,j}\right|$. To sum up, as long as one bit-plane is modified at least, the absolute value of the prediction error generated by Eq. (6), namely $e'_{i,j}$, will be not within the range $\left[2^{r-1}+1, 2^r - 1\right]$. Consequently, the original pixel can be recovered by finding a unique combination for $8^{th} \sim (r+1)^{th}$ bit-planes so that the prediction error is within the range $\left[2^{r-1}+1, 2^r - 1\right]$. Thus, the $(r+1)^{th}$ bit-plane of $p_{i,j}$ with large-magnitude $2^{r-1}+1 \leq \left|e_{i,j}\right| \leq 2^r - 1$ can be modified for data hiding and this bit-plane is marked with the label $s_{i,j,r} = 1$. More bit-planes used in the proposed method contribute to a high embedding capacity.

Note that if the $(r+1)^{th}$ bit-plane is marked with "1" or "2", we do not deal with the remaining $r^{th} \sim 1^{st}$ bit-planes and the labels of these bit-planes are "void". And these bit-planes cannot accommodate secret bits. As soon as these bit-planes are modified, they cannot be recovered losslessly. Next, we use the proof by contradiction to explain why the remaining $r^{th} \sim 1^{th}$ bit-planes cannot be used for data hiding when the $(r+1)^{th}$ bit-plane is marked with "1" or "2".

As previous analysis, if the $(r+1)^{th}$ bit-plane is marked with the label $s_{i,j,r} = 1$, namely, **Case 3**, it means that $2^{r-1}+1 \leq \left|e_{i,j}\right| \leq 2^r - 1$ and the $8^{th} \sim (r+1)^{th}$ bit-planes can be replaced by secret bits. Assume that the r^{th} bit-plane can be also modified for data hiding, we have $|e'_{i,j}| = |e_{i,j} + m'|$ according to the Eqs. (5-6), where $m' = \sum_{k=r}^{8} \left(a_{i,j,k} \times 2^{k-1}\right)$. We only consider that at least one bit-plane among these $8^{th} \sim r^{th}$ bit-planes is modified. Thus, we have

$$\begin{aligned} \left|e'_{i,j}\right| &= \left|e_{i,j} + m'\right| \\ &= \begin{cases} \left|m'\right| + \left|e_{i,j}\right|, & \text{if } e_{i,j} \times m' > 0 \\ \left|m'\right| - \left|e_{i,j}\right|, & \text{if } e_{i,j} \times m' < 0 \text{ and } \left|m'\right| \geq \left|e_{i,j}\right| \\ \left|e_{i,j}\right| - \left|m'\right|, & \text{if } e_{i,j} \times m' < 0 \text{ and } \left|m'\right| < \left|e_{i,j}\right| \end{cases} \end{aligned} \tag{15}$$

Bit-plane replace technique will generate different bit-plane modification combinations. Suppose that one of bit-plane modification combinations of $p_{i,j}$ is $a_{i,j,r} = -1, a_{i,j,r+1} = -1, a_{i,j,r+2} = 0, \ldots, a_{i,j,8} = 0$, it is clear that $m' = -(2^{r-1} + 2^r)$. If $e_{i,j} = 2^{r-1} + 1$, according to the Eq. (15), we have

$$\begin{aligned} |e'_{i,j}| &= |m'| - |e_{i,j}| \\ &= 2^{r-1} + 2^r - (2^{r-1}+1) \\ &= 2^r - 1 \end{aligned} \tag{16}$$

Clearly, the values of $|e'_{i,j}|$ and $|e_{i,j}|$ are both within the range $[2^{r-1}+1, 2^r - 1]$. Different from the modification combination of the $8^{th} \sim (r+1)^{th}$ bit-planes, there may be multiple modification combinations for the $8^{th} \sim r^{th}$ bit-planes so that the generated prediction errors are within the range $[2^{r-1}+1, 2^r - 1]$. This bit-plane modification will result in error recovery. Consequently, the r^{th} bit-plane cannot be used for data hiding. Similarly, the other $(r-1)^{th} \sim 1^{st}$ bit-planes are not used.

If the $(r+1)^{th}$ bit-plane is marked with the label $s_{i,j,r} = 2$, namely, **Case 2**, it means that $|e_{i,j}| = 2^r - 1$ and the $(r+1)^{th}$

bit-plane cannot accommodate secret bits. Assume that the remainder $r^{th} \sim 1^{st}$ bit-planes are the same with the secret bits for replacing them. According to the analysis of **Case 2**, it will result in error recovery. Thus, the remaining $(r-1)^{th} \sim 1^{st}$ bit-planes cannot accommodate secret bits.

Based on the above analysis, we design a hierarchical bit-plane label map generation mechanism for multiple bit-planes. Starting from $r = 7$ to $r = 1$, namely, starting from the 8^{th} bit-plane to the 2^{nd} bit-plane hierarchically, we divide the coarse-magnitude $0 \le |e_{i,j}| \le 2^r - 1$ into three fine-magnitudes, namely, $0 \le |e_{i,j}| \le 2^{r-1} - 1$, $|e_{i,j}| = 2^{r-1}$ and $2^{r-1} + 1 \le |e_{i,j}| \le 2^r - 1$. If $0 \le |e_{i,j}| \le 2^{r-1} - 1$, a label $s_{i,j,r} = 0$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane can be used for data hiding by bit replacement. If $|e_{i,j}| = 2^{r-1}$, a label $s_{i,j,r} = 2$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane remains unchanged. If $2^{r-1} + 1 \le |e_{i,j}| \le 2^r - 1$, a label $s_{i,j,r} = 1$ is used to mark the $(r+1)^{th}$ bit-plane of $p_{i,j}$ and this bit-plane can be also used for data hiding by bit replacement. After marking the $(r+1)^{th}$ bit-plane, we update $r \leftarrow r - 1$ and repeat the above processing until $r = 1$. Note that if the $(r+1)^{th}$ bit-plane is marked by "1" or "2", we do not deal with the remaining $r^{th} \sim 1^{st}$ bit-planes.

For the image with 8 bit-planes, it is clear that $0 \le |e_{i,j}| \le 255$. We first divide $0 \le |e_{i,j}| \le 255$ into $0 \le |e_{i,j}| \le 127$ and $128 \le |e_{i,j}| \le 255$. For a natural image, the prediction errors with relatively large magnitudes $128 \le |e_{i,j}| \le 255$ occupy a small proportion and all bit-planes of these corresponding pixels cannot accommodate secret bits. The 8^{th} bit-planes of these pixels with $128 \le |e_{i,j}| \le 255$ are marked with labels "3" and the labels of the other bit-planes of these pixels are "void". Next, we only illustrate the label generation for the prediction errors with relatively small magnitude $0 \le |e_{i,j}| \le 127$. Starting from $r = 7$, we divide $0 \le |e_{i,j}| \le 127$ into $0 \le |e_{i,j}| \le 63$, $|e_{i,j}| = 64$ and $65 \le |e_{i,j}| \le 127$. According to the above three cases, the 8^{th} bit-plane of $p_{i,j}$ with $0 \le |e_{i,j}| \le 63$ or $65 \le |e_{i,j}| \le 127$ is marked with "0" or "1" and it can be replaced by a secret bit. The 8^{th} bit-plane of $p_{i,j}$ with $|e_{i,j}| = 64$ is marked with "2" and cannot accommodate secret bit. Then, $r = 6$. We divide $0 \le |e_{i,j}| \le 63$ into $0 \le |e_{i,j}| \le 31$, $|e_{i,j}| = 32$ and $33 \le |e_{i,j}| \le 63$. The 7^{th} bit-plane of $p_{i,j}$ with $0 \le |e_{i,j}| \le 31$ or $33 \le |e_{i,j}| \le 63$ is marked with "0" or "1" and it can be replaced by a secret bit. The 7^{th} bit-plane of $p_{i,j}$ with $|e_{i,j}| = 32$ is marked with "2" and cannot accommodate secret bit. Repeat the above processing until $r = 1$. Note that when $r = 1$, we have $0 \le |e_{i,j}| \le 1$, which can be divided into $|e_{i,j}|$=0 and $|e_{i,j}| = 1$. Finally, the 2^{nd} bit-plane of $p_{i,j}$ is marked with "1" or "2" when $|e_{i,j}| = 0$ or $|e_{i,j}| = 1$. After all pixels are processed, we can get 7 bit-plane label maps.

To better illustrate hierarchical label map generation for $8^{th} \sim 2^{nd}$ bit-planes, we design an incomplete ternary tree as shown in the Fig. 3 according to prediction error magnitude division. The root node is $0 \le |e_{i,j}| \le 255$. Except for the node $128 \le |e_{i,j}| \le 255$, the other nodes are labeled by ternary codes. Only the predictor error of the left node in the current layer can be further divided. In the next layer, the left, middle, and right nodes are marked with the labels "0", "2" and "1", respectively. Table II illustrates the labels of the bit-planes of $p_{i,j}$ according to different magnitude divisions, where "-" denotes that the label is "void".

TABLE II
MAGNITUDE OF PREDICTION ERROR AND THE BIT-PLANE LABEL

Magnitude	Bit-plane label						
	8^{th}	7^{th}	6^{th}	5^{th}	4^{th}	3^{th}	2^{th}
$128 \le \lvert e_{i,j}\rvert \le 255$	3	-	-	-	-	-	-
$65 \le \lvert e_{i,j}\rvert \le 127$	1	-	-	-	-	-	-
$\lvert e_{i,j}\rvert = 64$	2	-	-	-	-	-	-
$\lvert e_{i,j}\rvert = 32$	0	2	-	-	-	-	-
$33 \le \lvert e_{i,j}\rvert \le 63$	0	1	-	-	-	-	-
$\lvert e_{i,j}\rvert = 16$	0	0	2	-	-	-	-
$17 \le \lvert e_{i,j}\rvert \le 31$	0	0	1	-	-	-	-
$\lvert e_{i,j}\rvert = 8$	0	0	0	2	-	-	-
$9 \le \lvert e_{i,j}\rvert \le 15$	0	0	0	1	-	-	-
$\lvert e_{i,j}\rvert = 4$	0	0	0	0	2	-	-
$5 \le \lvert e_{i,j}\rvert \le 7$	0	0	0	0	1	-	-
$\lvert e_{i,j}\rvert = 2$	0	0	0	0	0	2	-
$\lvert e_{i,j}\rvert = 3$	0	0	0	0	0	1	-
$\lvert e_{i,j}\rvert = 1$	0	0	0	0	0	0	2
$\lvert e_{i,j}\rvert = 0$	0	0	0	0	0	0	1

Next we take an example to illustrate pixel recovery with bit-plane labels. Suppose that $p_{i,j} = 183$ and its prediction value is $\hat{p}_{i,j} = 169$. The binary sequence of $p_{i,j}$ is $(10110111)_2$. It is clear that $|e_{i,j}| = |p_{i,j} - \hat{p}_{i,j}| = 14 \in [9, 15]$. According to Fig. 3 and Table II, the labels of the $8^{th} \sim 5^{th}$ bit-planes are "0001" and the four bit-planes can be replaced by secret bits to generate the marked pixel. For recovering $p_{i,j}$, we need to find a combination for the $8^{th} \sim 5^{th}$ bit-planes such that the absolute value of the generated prediction error is within the range $[9, 15]$. Since the $1^{st} \sim 4^{th}$ bit-planes remain unchanged, the binary sequences of $p_{i,j}$ with all modification combinations for the $8^{th} \sim 5^{th}$ bit-planes are $(00000111)_2$, $(00010111)_2$, $(00100111)_2$, $(00110111)_2$, $(01000111)_2$, $(01010111)_2$, $(01100111)_2$, $(01110111)_2$, $(10000111)_2$, $(10010111)_2$, $(10100111)_2$, $(10110111)_2$, $(11000111)_2$, $(11010111)_2$, $(11100111)_2$, $(11110111)_2$, respectively. And their corresponding decimal values are 7, 23, 39, 55, 71, 87, 103, 119, 135, 151, 167, 183, 199, 215, 231 and 247. The absolute values of prediction errors are 162, 146, 130, 114, 98, 82, 66, 50, 34, 18, 2, 14, 30, 46, 62 and 78. Among these values, only 14 generated by 183 is within the range [9, 15]. Consequently, $p_{i,j}$ can be recovered as 183.

Overall, these 7 bit-plane label maps indicate which bit-planes of original image can be replaced by secret bits. And with the assistance of bit-plane label maps, the original image can be recovered perfectly.

B. Image Encryption

The stream cipher (e.g., AES-CTR) is one of the most popular and reliable encryption tools due to its provable security and easy implementation in software and hardware. In addition, large amounts of data have already been encrypted by using stream cipher in a standard way. Consequently, in the proposed method, the original image is encrypted by stream cipher using an encryption key K_e. First, we generate a pseudo-random matrix $\mathbf{R}$ of size $H \times W$ through the key K_e. Next, the pixel $p_{i,j}$ and its corresponding $R_{i,j}$ are transformed

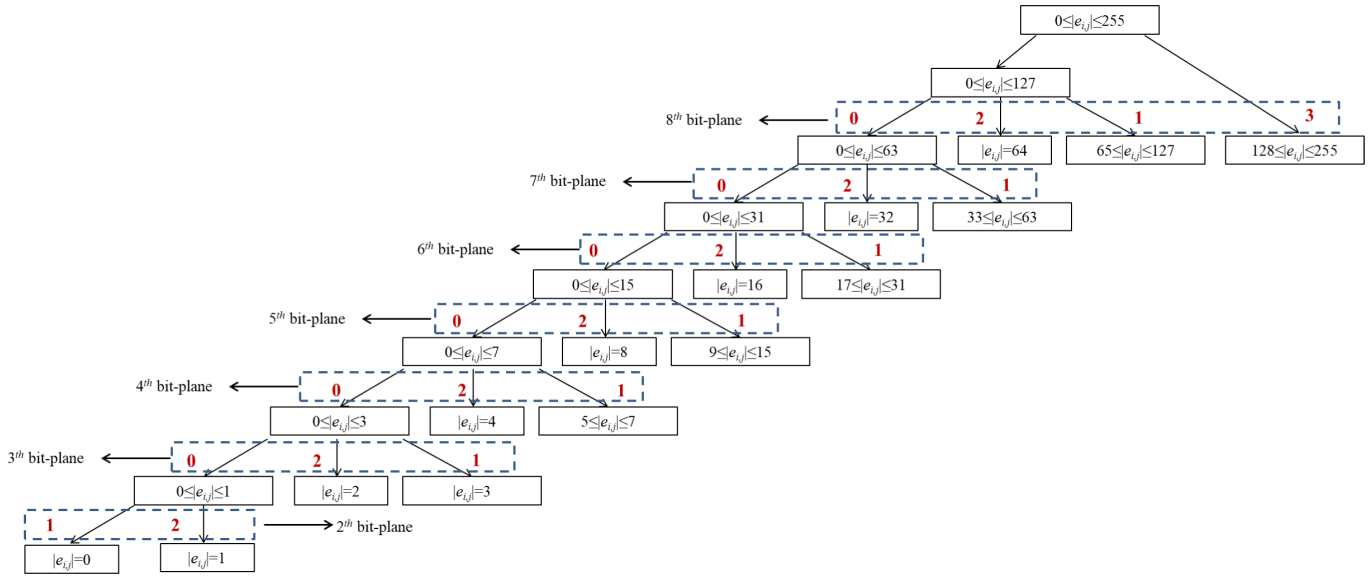

Fig. 3. Hierarchical label generation for prediction errors using incomplete ternary tree.

into the 8-bit binary sequence using the Eq. (1), denoted as $b_{i,j,1}, b_{i,j,2}, \ldots, b_{i,j,8}$ and $R_{i,j,1}, R_{i,j,2}, \ldots, R_{i,j,8}$. Then, each bit-plane of original pixel is encrypted as following.

$$E_{i,j,k} = b_{i,j,k} \oplus R_{i,j,k}, \quad k = 1, 2, \ldots, 8 \tag{17}$$

where $\oplus$ denotes the exclusive-or operation. The corresponding encrypted pixel $p^{(e)}_{i,j}$ can be obtained by using the Eq. (2). Finally, an encrypted image $\mathbf{I}_e$ is generated by encrypting all pixels.

C. Label Map Embedding

In order to generate room for embedding in the encrypted image as well as perform lossless recovery, hierarchical bit-plane label maps are required to be embedded into the encrypted image before data embedding operation. Firstly, hierarchical bit-plane label maps for multiple bit-planes are generated according to Section *A*. Note that the label for the 8^{th} bit-plane has four states, namely, "0", "1", "2" and "3" and the labels for the other bit-planes have four states, namely, "0", "1", "2" and "void" according to Fig. 3 and Table II.

Then we transform each bit-plane and its bit-plane label map into a one-dimensional sequence in raster-scanning order, respectively. Specifically, the 8^{th} bit-planes of all pixels are transformed to a sequence, the 7^{th} bit-planes of all pixels are transformed to a sequence, and so on. Note that when the labels are "void", they are skipped. Let the 7 bit-plane label map sequences be $\mathbf{m}_7, \mathbf{m}_6, \ldots, \mathbf{m}_1$, which corresponds to the $8^{th}, 7^{th}, \ldots, 2^{nd}$ bit-planes, respectively. And suppose that the sizes of these label map sequences are $l_7, l_6, \ldots, l_1$, respectively. Clearly, $l_7 = (H-1)\times(W-1)$ because the pixels in the first row and in the first column remain unchanged. We count the numbers of "0", "1", "2" and "3" in these 7 bit-plane label maps, which are denoted by $l_7^{(0)}, l_6^{(0)}, \ldots, l_1^{(0)}$, $l_7^{(1)}, l_6^{(1)}, \ldots, l_1^{(1)}, l_7^{(2)}, l_6^{(2)}, \ldots, l_1^{(2)}$ and $l_7^{(3)}$, respectively. Here, the label "3" is generated in the 8^{th} bit-plane. So only $l_7^{(3)}$ is exhibited for the label "3". Clearly, $l_r = l_r^{(0)} + l_r^{(1)} + l_r^{(2)} + l_r^{(3)}$ if $r = 7$. Otherwise, $l_r = l_r^{(0)} + l_r^{(1)} + l_r^{(2)}$ for $r \in \{6, 5, 4, 3, 2, 1\}$. According to Section A, the size of label map of the r^{th} bit-plane is equal with the number of "0" of its upper bit-plane, namely, the $(r+1)^{th}$ bit-plane, and then we have

$$\begin{cases} l_6 = l_7^{(0)} \\ l_5 = l_6^{(0)} \\ l_4 = l_5^{(0)} \\ \ldots \\ l_1 = l_2^{(0)} \end{cases} \tag{18}$$

To reduce the size of bit-plane label map, we use arithmetic coding to compress the bit-plane label maps $\mathbf{m}_7, \mathbf{m}_6, \ldots, \mathbf{m}_1$ to generate corresponding binary sequences $\mathbf{s}_7, \mathbf{s}_6, \ldots, \mathbf{s}_1$, whose sizes are $cl_7, cl_6, \ldots, cl_1$, respectively. It is known that local pixels are highly correlated. The distribution of the histogram of the predictor errors concentrates on the bin 0. Since the label "3" only occurs in the 8^{th} bit-plane and occupies a small proportion, it is not mentioned in the following sections for simplicity. According to the Section A, for the $(r+1)^{th}$ bit-plane, we have $0 \le |e_{i,j}| \le 2^{r-1} - 1$, $|e_{i,j}| = 2^{r-1}$ and $2^{r-1} + 1 \le |e_{i,j}| \le 2^r - 1$, which are marked with "0", "2" and "1", respectively. It is clear that the prediction errors with $0 \le |e_{i,j}| \le 2^{r-1} - 1$ occupy a large proportion. And the number of the prediction errors with $2^{r-1} + 1 \le |e_{i,j}| \le 2^r - 1$ is bigger than that of the prediction errors with $|e_{i,j}| = 2^{r-1}$. Correspondingly, in the $(r+1)^{th}$ bit-plane label map, the "0" occupies a large proportion and the proportion of "1" is bigger than that of "2". In addition, for MSB bit-planes, the proportion of "0" is significantly larger than the sum of proportions of "1" and "2". Consequently, we can obtain considerable compression ratios for MSB bit-plane label maps using arithmetic coding. However, as r decreases, the proportion of the prediction errors with $2^{r-1}+1 \le |e_{i,j}| \le 2^r - 1$ and $|e_{i,j}| = 2^{r-1}$ increases and

the compression ratio also decreases. For the $(r+1)^{th}$ bit-plane of encrypted image, $\lceil \log_2 l_r \rceil$ bits are sufficient to encode the size of the compressed bit-plane label map, namely, cl_r. Then, the compressed bit-plane label map and its size are concatenated to generate the auxiliary information sized $\lceil \log_2 l_r \rceil + cl_r$. Since the auxiliary information must be embedded into the first $\lceil \log_2 l_r \rceil + cl_r$ bits of the $(r+1)^{th}$ bit-plane by replacement technique for reversibility, we count the number of "0" and "1" from the $(\lceil \log_2 l_r \rceil + cl_r + 1)^{th}$ index to the end of the $(r+1)^{th}$ bit-plane label map, denoted by l'_r. The first replaced $\lceil \log_2 l_r \rceil + cl_r$ bits with labels "2" and "void" are required to be embedded into the remaining bits with labels "0" and "1" of the current bit-plane by bit replacement technique since the bits with labels "2" and "void" are irrecoverable. Suppose that the number of the first replaced $\lceil \log_2 l_r \rceil + cl_r$ bits with labels "2" and "void" is lu_r in the $(r+1)^{th}$ bit-plane of the encrypted image. If $l'_r > lu_r$, there is embedding room for secret bits. Otherwise, this bit-plane cannot accommodate its corresponding label map and secret bits cannot be embedded.

Based on the above analysis, the compressed bit-plane label maps are embedded into the corresponding bit-planes from the MSB bit-plane to the LSB bit-plane hierarchically. The generated auxiliary information is embedded starting from $r = 7$. If $l'_r > lu_r$, we update $r \leftarrow r-1$ and repeat the embedding. Otherwise, the embedding operation is finished and let $u = r+1$. After auxiliary information embedding, we get the final encrypted image containing the label maps. Note that the $8^{th} \sim u^{th}$ bit-planes have room for secret bits after embedding label maps.

D. Hierarchical Embedding

In this section, secret bits are encrypted by the data hiding key K_D to enhance security. Before data hiding in encrypted image, the auxiliary information of the $8^{th} \sim u^{th}$ bit-planes is hierarchically picked out from the obtained encrypted image. For the 8^{th} bit-plane of the encrypted image containing the label map, we first extract its first $\lceil \log_2 l_7 \rceil$ bits and decode these bits of data to obtain the size of compressed label map, namely, cl_7, where $l_7 = (H-1)\times(W-1)$. Then the following $(\lceil \log_2 l_7 \rceil + 1)^{th} \sim (\lceil \log_2 l_7 \rceil + cl_7)^{th}$ bits of data are extracted to obtain the compressed label map and the corresponding uncompressed version can be obtained by decompression. According to the bit-plane label map, we calculate the number of the labels "void" and "2" in the first $(\lceil \log_2 l_7 \rceil + cl_7)$ bits of the bit-plane label map, namely lu_7. Beginning with the $(\lceil \log_2 l_7 \rceil + cl_7 + 1)^{th}$ bit, we count the number of "0" and "1" until the number is equal to lu_7 and record the current bit index as ld_7. Obviously, the $(\lceil \log_2 l_7 \rceil + cl_7 + 1)^{th} \sim (ld_7)^{th}$ bits of data store the first $\lceil \log_2 l_7 \rceil + cl_7$ bits of the original bit-plane with the labels "void" or "2" . The remaining bits of the bit-plane can be replaced by secret bits. Starting from the $(ld_7 + 1)^{th}$ bit to the end of the bit-plane, if the label of the current bit is "0" or "1", a secret bit is embedded into this bit by replacement technique. If the label of the current bit is "2" or "void", the current bit is skipped. Then, we can get the marked 8^{th} bit-plane. For the $7^{th} \sim u^{th}$ bit-planes, according to the Eq. (18), we can obtain the size of the current bit-plane label map, namely, $l_r (r = 6, 5, \ldots, u-1)$. Then, the first $\lceil \log_2 l_r \rceil$ bits of data are extracted from the $(r+1)^{th}$ bit-plane and decoded to obtain the size of compressed bit-plane label map, cl_r. The following cl_r bits of data are decompressed to generate the uncompressed bit-plane label map. For each bit of the r^{th} bit-plane, if the label of the $(r+1)^{th}$ bit-plane is "0", we can get its label according to the extracted r^{th} bit-plane label map. Otherwise, its label is "void". Similarly, secret bits can be embedded into the $7^{th} \sim u^{th}$ bit-planes by the above calculations. Note that the remaining $(u-1)^{th} \sim 1^{st}$ bit-planes are kept unchanged. Finally, a marked encrypted image $\mathbf{I}_m$ is generated.

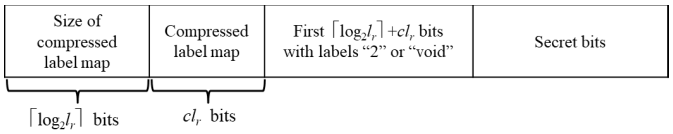

Fig. 4. Constitution of the marked bit-plane.

Fig. 4 illustrates constitution of the marked bit-plane. The first $\lceil \log_2 l_r \rceil$ bits represent the size of compressed label map and the following cl_r bits are the compressed label map. The $(\lceil \log_2 l_r \rceil + cl_r + 1)^{th} \sim (ld_r)^{th}$ bits store the first $\lceil \log_2 l_r \rceil + cl_r$ bits with labels "2" or "void" of the original bit-plane, where the definition of ld_r can be referred to ld_7. The remaining part is used to store secret bits, where the bits with labels "0" or "1" are replaced by secret bits.

E. Data Extraction and Image Recovery

On the receiver side, three scenarios are considered due to the separation characteristic of the proposed method. The first scenario is that the receiver only has the data-hiding key K_D, he/she can accurately extract secret bits from the marked encrypted image according to the bit-plane label maps. Another scenario is that the receiver only has the encryption key K_E, he/she can perfectly reconstruct the original image according to pixel prediction and bit-plane label maps. For the third scenario, the receiver has both keys. So he/she can perfectly extract secret bits and recover the original image.

Clearly, the receiver can hierarchically extract the $8^{th} \sim u^{th}$ bit-plane label maps from the marked encrypted image. Then, data extraction and image recovery can be performed based on the label maps. Although the label map extraction can be done without keys, the next process will conduct different operations according to the receiver's keys.

If the receiver only has the data-hiding key K_D, he/she extracts the encrypted secret bits from the $(ld_r + 1)^{th}$ bit to the end of each of the $8^{th} \sim u^{th}$ bit-planes hierarchically according to bit-plane label maps, where $r = 7, 6, \ldots, u-1$. If the current bit label is "0" or "1", he/she can extract the current bit as one secret bit. Otherwise, the current bit is skipped. After data extraction, the receiver has to concatenate the extracted bits from each marked bit-plane from the 8^{th} bit-plane to the u^{th} bit-plane. Finally, the encrypted secret bits are obtained and further decrypted by using the data-hiding key K_D. Note that there is no error in the whole process.

If the receiver only has the encryption key K_E, the original image can be constructed without loss. This is because the bit-plane label maps indicate that which bit-planes can be modified to accommodate secret bits and the modified bit-planes can be recovered with assistance of these label maps according to the Section A. First, the receiver extracts bit-plane label maps from the 8^{th} bit-plane to the u^{th} bit-plane. Then the first $\lceil \log_2 l_r \rceil + cl_r$ bits with labels "2" or "void" in the $(r+1)^{th}$ bit-plane are extracted from the $(\lceil \log_2 l_r \rceil + cl_r + 1)^{th}$ bit to the $(ld_r)^{th}$ bit of the bit-plane, where $r = 7, 6, \ldots, u-1$. In this way, the first $\lceil \log_2 l_r \rceil + cl_r$ bits with labels "2" or "void" of the current encrypted bit-plane are recovered. According to K_E, all pixels can be directly decrypted by exclusive-or operation. Note that the pixels in the first row and the first column are recovered losslessly after decryption and these pixels can be used for predicting and recovering the other pixels. The prediction and recovery of other pixels are conducted in the raster scanning order. For a current to-be-recovered pixel $p'_{i,j}$, suppose that its decrypted 8 bit-planes are denoted as $b'_{i,j,8}, b'_{i,j,7}, \ldots, b'_{i,j,1}$. According to the Sections C and D, $b'_{i,j,u-1}, b'_{i,j,7}, \ldots, b'_{i,j,1}$ are kept unchanged. Thus, we can recover the $(u-1)^{th} \sim 1^{st}$ bit-planes as

$$b_{i,j,k} = b'_{i,j,k}, \quad k = u-1, \ldots, 1 \tag{19}$$

Since the $8^{th} \sim u^{th}$ bit-planes of the current pixel are modified, we calculate its prediction value $\hat{p}_{i,j}$ according to the Eq. (3) and then recover them by combining the prediction value with bit-plane label maps. The labels of the $8^{th} \sim u^{th}$ bit-planes of $p'_{i,j}$ are denoted by $s_{i,j,7}, s_{i,j,6}, \ldots, s_{i,j,u-1}$. According the generation of the label map in Section A, we have the following cases.

(1) If $\begin{cases} s_{i,j,7} = 0 \\ s_{i,j,6} = 0 \\ \ldots \\ s_{i,j,u-1} = 0 \end{cases}$, we have that the original prediction error $e_{i,j}$ is subject to $0 \le |e_{i,j}| \le 2^{u-1-1} - 1$ according to the hierarchical label in the Fig. 3. Since every bit-plane has two possibilities, namely, "0" and "1", the $8^{th} \sim u^{th}$ bit-planes can be recovered by finding a unique bit-plane combination as follows.

$$\begin{cases} b_{i,j,8} = b^*_{i,j,8} \\ b_{i,j,7} = b^*_{i,j,7} \\ \ldots \\ b_{i,j,u} = b^*_{i,j,u} \end{cases}$$

$$s.t.\ 0 \le |\sum_{k=u}^{8} \left(b^*_{i,j,k} \times 2^{k-1} \right) + \sum_{k=1}^{u-1} \left(b_{i,j,k} \times 2^{k-1} \right) - \hat{p}_{i,j}| \le 2^{u-1-1} - 1 \tag{20}$$

where $b^*_{i,j,k} \in \{0, 1\}$.

(2) Otherwise, we search the label $s_{i,j,v} = 1$ or $s_{i,j,v} = 2$, starting from $s_{i,j,7}$ to $s_{i,j,u-1}$. According to the Section A, the remaining labels $s_{i,j,v-1}, s_{i,j,v-2}, \ldots, s_{i,j,u-1}$ are "void" and the corresponding bit-planes remain unchanged. So, we have

$$b_{i,j,k} = b'_{i,j,k}, k = u, \ldots, v \quad and\ v \ge u \tag{21}$$

Then we recover the $8^{th} \sim (v+1)^{th}$ bit-planes in terms of $s_{i,j,v}$. If $s_{i,j,v} = 1$, the original prediction error $e_{i,j}$ is subject to $2^{v-1} + 1 \le |e_{i,j}| \le 2^v - 1$ according to the hierarchical label in the Fig. 3. Thus, the $8^{th} \sim (v+1)^{th}$ bit-planes can be recovered by finding a unique bit-plane combination as follows.

$$\begin{cases} b_{i,j,8} = b^*_{i,j,8} \\ b_{i,j,7} = b^*_{i,j,7} \\ \ldots \\ b_{i,j,v+1} = b^*_{i,j,v+1} \end{cases}$$

$$s.t.\ 2^{v-1} + 1 \le |\sum_{k=v+1}^{8} \left(b^*_{i,j,k} \times 2^{k-1} \right) + \sum_{k=1}^{v} \left(b_{i,j,k} \times 2^{k-1} \right) - \hat{p}_{i,j}| \le 2^v - 1 \tag{22}$$

If $s_{i,j,v} = 2$, the $(v+1)^{th}$ bit-plane also remains unchanged since it is skipped. So it can be recovered as $b_{i,j,v+1} = b'_{i,j,v+1}$. According to the Fig. 3, we have that the original prediction error $e_{i,j}$ is subject to $|e_{i,j}| = 2^{v-1}$. Consequently, $8^{th} \sim (v+2)^{th}$ bit-planes can be recovered by finding a unique bit-plane combination as follows.

$$\begin{cases} b_{i,j,8} = b^*_{i,j,8} \\ b_{i,j,7} = b^*_{i,j,7} \\ \ldots \\ b_{i,j,v+2} = b^*_{i,j,v+2} \end{cases}$$

$$s.t.\ |\sum_{k=v+2}^{8} \left(b^*_{i,j,k} \times 2^{k-1} \right) + \sum_{k=1}^{v+1} \left(b_{i,j,k} \times 2^{k-1} \right) - \hat{p}_{i,j}| = 2^{v-1} \tag{23}$$

Finally, all the bit-planes $b_{i,j,1}, b_{i,j,2}, \ldots, b_{i,j,8}$ of the original pixel $p_{i,j}$ can be recovered losslessly. According to the Eq. (2), the original pixel $p_{i,j}$ can be recovered without loss so that the original image can be recovered losslessly.

IV. Experimental Results

In this section, we first analyze security of the proposed method, then show the experimental results and make comparison with some state-of-the-art methods. Six commonly used test images are exploited in the experiments, as shown in Fig.5. We also test performance of different methods with two entire image datasets: BOSSBase [49], and BOWS-2 [50]. The embedding rate (ER) represented by bpp (bits per pixel) is used as the metric of embedding capacity (EC), and the below-mentioned ER or EC is the pure ER or EC. In addition, other commonly-used metrics, i.e., PSNR (Peak signal-to-noise ratio) and SSIM (structural similarity), are also adopted to analyze performance.

A. Security Analysis

As an encryption domain based RDH method, the proposed method is able to provide the security for both the original image and secret data. In the proposed method, the security of secret data is provided by data encryption with K_D. It is worth

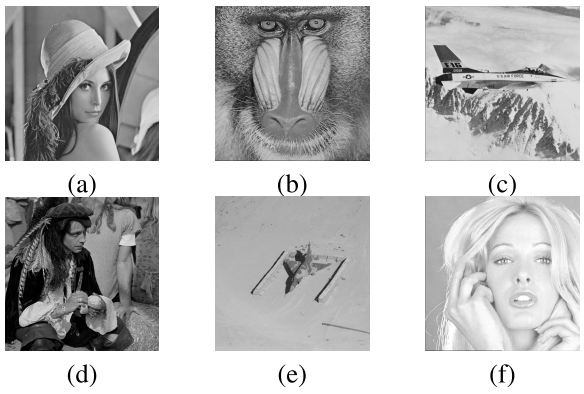

Fig. 5. Six test images. (a) Lena; (b) Baboon; (c) Jetplane; (d) Man; (e) Airplane; (f) Tiffany.

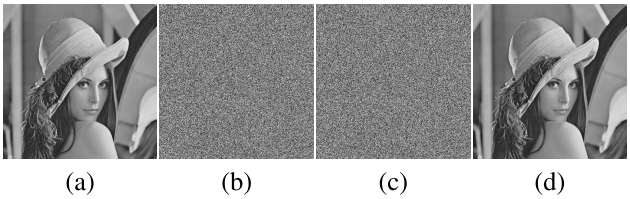

Fig. 6. Results of the proposed method on Lena image. (a) Original image; (b) Encrypted image; (c) Marked encrypted image (ER = 3.019 bpp); (d) Reconstructed image (PNSR $\rightarrow +\infty$ and SSIM = 1).

noting that secret data can be encrypted by any secure data encryption algorithm. Hence, it is almost impossible to leak the secret data without K_D. Therefore, we mainly pay attention to the perceptual security and statistical analysis of the encrypted images generated by using stream cipher. We select Lena image to illustrate security analysis. Fig. 6 exhibits visual results generated by the proposed method. It can be seen from Fig. 6 that both the encrypted image and the marked encrypted image are noise-like and completely incomprehensible. The subjective features of original image cannot be revealed from them, illustrating a high perceptual security level.

It is well-known that resisting statistical analysis on ciphertext is of crucial importance for a cryptosystem. In order to illustrate security of the encryption scheme used in the proposed method, five statistical indicators, namely, histogram, correlation coefficient, entropy, PSNR and SSIM, are calculated. Fig. 7 shows the histograms of the original image, the encrypted image and the marked encrypted image. It can be seen that the original image has rich histogram features, while the histograms of the encrypted image and the marked encrypted image are almost distributed uniformly. It is difficult to find the matched features between the histogram of original image and the histogram of encrypted image or the marked one. Therefore, the original image cannot be retrieved from the encrypted image or marked encrypted image by statistical analysis. This validates a good statistical property.

Our statistical results of correlation coefficient, entropy, PSNR and SSIM are given in the Table III. Generally, for most natural images, any two adjacent pixels are always highly correlated in three directions, namely horizontal direction, vertical direction and diagonal direction. Therefore, encryption

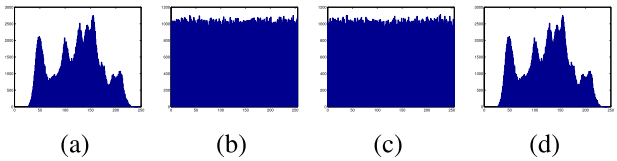

Fig. 7. Histograms of different images. (a) Original image (b) Encrypted image (c) Marked encrypted image (d) Reconstructed image.

scheme should disorganize the correlation among adjacent pixels so as to withstand statistical attack. The $2^{nd} \sim 4^{th}$ columns of the Table III present the correlation coefficients of two adjacent pixels in three directions of the original image, the encrypted image and the marked encrypted image. It can be seen that the correlation coefficients of the encrypted image and the marked encrypted image are all close to 0. This illustrates that our encryption scheme can efficiently destroy pixel correlation in three directions.

Information entropy is a useful metric for evaluating the randomness of the encrypted images. If the information entropy of an encrypted image is close to the theoretical value, the encrypted image has good randomness. For a grayscale image with 8 bit-planes, the theoretical value of information entropy is 8. As shown in Table III, the entropies of the encrypted image and the marked encrypted image are both close to the theoretical value 8. In addition, PSNR and SSIM between the original image and the encrypted/marked image are also calculated. It is found that the PSNR and SSIM results are very small values, meaning that the evaluated images are dissimilar. Therefore, it can be concluded that our encryption scheme is secure in terms of correlation coefficient, entropy, PSNR and SSIM.

B. Embedding Performance

In this section, the mentioned ER or EC (Embedding Capacity) is the pure ER or EC. Here we also select Lena image to illustrate the embedding performance in details. Bit-plane label maps are generated from MSB bit-plane to LSB bit-plane hierarchically according to Section III A. And the characteristic of each bit-plane label map is closely related with the numbers of prediction errors with different magnitudes. Since the number of the prediction errors with $128 \leq |e_{i,j}| \leq 255$ is 0 for image Lena, we only count the numbers of "0", "1" and "2" for each bit-plane label map according to prediction errors. As shown in Table IV, the sizes of the $8^{th} \sim 2^{th}$ bit-plane label maps are $l_7 = 511 \times 511$, $l_6 = l_7^{(0)} = 260998$, $l_5 = l_6^{(0)} = 259273$, $l_4 = l_5^{(0)} = 249935$, $l_3 = l_4^{(0)} = 221338$, $l_2 = l_3^{(0)} = 153911$, $l_1 = l_2^{(0)} = 78169$. For the 8^{th} bit-plane, the number of prediction errors with $0 \leq |e_{i,j}| \leq 63$ is 260998. Thus, the number of the corresponding labels with "0" is also 260998. Similarly, the numbers of corresponding labels with "1" and "2" are 116 and 7, respectively. It is clear that the labels with "1" and "2" occupy a small proportion, while the labels with "0" occupy a large proportion. Consequently, a considerable compression ratio of the 8^{th} bit-plane label map is obtained by arithmetic coding. And the size of its compressed version is only 1600 bits which can be encoded

TABLE III
STATISTICAL ANALYSIS ON LENA IMAGE

Image	Correlation coefficient			Entropy	PSNR between $\mathbf{I}$ and $\mathbf{I}_e/\mathbf{I}_m$	SSIM between $\mathbf{I}$ and $\mathbf{I}_e/\mathbf{I}_m$
	Horizontal	Vertical	Diagonal			
Original image $\mathbf{I}$	0.9711	0.9842	0.9576	7.4451	-	-
Encrypted image $\mathbf{I}_e$	0.0061	-0.0011	0.0251	7.9994	8.5667	0.0381
Marked image $\mathbf{I}_m$	0.0074	-0.0006	-0.0020	7.9985	8.7567	0.0376

TABLE IV
PERFORMANCE ANALYSIS ON LENA IMAGE

Bit-plane index	Magnitude	Label	Number	Entropy of bit-plane label map	Size of auxiliary information (bit)	Pure capacity (bit)
8^{th}	$0 \le \lvert e_{i,j}\rvert \le 63$	0	260998	0.0060	1618	259496
	$\lvert e_{i,j}\rvert = 64$	2	7			
	$64 \le \lvert e_{i,j}\rvert \le 127$	1	116			
	$128 \le \lvert e_{i,j}\rvert \le 255$	3	0			
7^{th}	$0 \le \lvert e_{i,j}\rvert \le 31$	0	259273	0.0607	15634	245164
	$\lvert e_{i,j}\rvert = 32$	2	200			
	$33 \le \lvert e_{i,j}\rvert \le 63$	1	1525			
6^{th}	$0 \le \lvert e_{i,j}\rvert \le 15$	0	249935	0.2446	62706	195275
	$\lvert e_{i,j}\rvert = 16$	2	1292			
	$17 \le \lvert e_{i,j}\rvert \le 31$	1	8046			
5^{th}	$0 \le \lvert e_{i,j}\rvert \le 7$	0	221338	0.6074	151074	91479
	$\lvert e_{i,j}\rvert = 8$	2	7382			
	$9 \le \lvert e_{i,j}\rvert \le 15$	1	21215			
4^{th}	$0 \le \lvert e_{i,j}\rvert \le 3$	0	153911	1.1772	260426	-
	$\lvert e_{i,j}\rvert = 4$	2	25159			
	$5 \le \lvert e_{i,j}\rvert \le 7$	1	42268			
3^{th}	$0 \le \lvert e_{i,j}\rvert \le 1$	0	78169	1.4877	228913	-
	$\lvert e_{i,j}\rvert = 2$	2	42014			
	$\lvert e_{i,j}\rvert = 3$	1	33728			
2^{nd}	$\lvert e_{i,j}\rvert = 0$	1	29089	0.9522	74352	-
	$\lvert e_{i,j}\rvert = 1$	2	49080			
Total pure EC/ER						791414 bits / 3.019 bpp

by l_7 bits. The auxiliary information size of the 8^{th} bit-plane is $\lceil \log_2 l_7 \rceil + 1600 = 1618$ bits. Similarly, those of the $7^{th} \sim 2^{nd}$ bit-planes are $\lceil \log_2 l_6 \rceil + 15616 = 15634$ bits, $\lceil \log_2 l_5 \rceil + 62688 = 62706$ bits, $\lceil \log_2 l_4 \rceil + 151056 = 151074$ bits, $\lceil \log_2 l_3 \rceil + 260408 = 260426$ bits, $\lceil \log_2 l_2 \rceil + 228896 = 228913$ bits, $\lceil \log_2 l_1 \rceil + 74336 = 74352$ bits, respectively. As to entropy, it can be seen from Table IV that the entropy of the 8^{th} bit-plane label map is minimum. In addition, the entropy of the label map will increase when its bit-plane index decreases. After encryption, the auxiliary information of each bit-plane is embedded into the corresponding encrypted bit-plane from MSB to LSB hierarchically. First, for the 8^{th} bit-plane, its first 1618 bits of data is extracted and then the auxiliary information is embedded into its first 1618 bits by replacement technique. We count the number of "0" and "1" from the 1619^{th} bit to the end of the 8^{th} bit-plane label map. Obviously, this number is the EC of the current bit-plane. The EC of the 8^{th} bit-plane is 259496 bits in the experiment. Since the bits with labels "2" or "void" in the first 1618 bits cannot be recovered, they are also required to be embedded into the bit-plane. For Lena, there is no bit with label "2" or "void" in the first 1618 bits. Consequently, the pure EC of the 8^{th} bit-plane is 259496 bits. By the same way, the pure EC of the following bit-planes can be obtained. As shown in the Table IV, the pure ECs of the $7^{th} \sim 5^{th}$ bit-planes are 245164, 195275 and 91479 bits, respectively. The remaining bit-planes are not used as their vacated room cannot accommodate auxiliary information and secret data. Thus, the total pure EC is $259496 + 245164 + 195275 + 91479 = 791414$ bits and the corresponding ER is 3.019 bpp.

We also calculate the ECs for other test images, and the results are shown in the Table V. It can be observed from the Table V that the EC of an image is closely related with its content. In these test images, Baboon image contains more textures, while Airplane image has more smooth regions. Table V shows that only three bit-planes of Baboon can accommodate secret bits and the ER of Baboon is only 1.4596 bpp, which is the smallest ER. In addition, the EC of its every bit-plane is also smaller than those of other images. The Airplane image has the maximum EC among the test images, which is 3.9872 bpp. Compared with Baboon image, Airplane has more bit-planes for data hiding. it can be seen that five bit-planes of Airplane, namely, the $8^{th} \sim 4^{th}$ bit-planes can be used for data hiding. Since there are rich textures in Baboon, the pixels are not easy to be accurately predicted. Therefore, more prediction errors with large magnitudes will be generated. Consequently, a few bit-planes of the corresponding pixels can accommodate secret bits. On the contrary, more prediction errors with small magnitudes can be generated from the images with few textures. Overall, the smooth images have bigger ECs than the texture images.

Furthermore, we apply the proposed method to the entire image datasets, namely, BOSSBase and BOWS-2. The two databases both have 10000 gray-scale images sized 512×512.

TABLE V
OUR PURE EMBEDDING CAPACITIES OF DIFFERENT TEST IMAGES

Image	EC of bit-plane (bit)					EC	ER
	8^{th}	7^{th}	6^{th}	5^{th}	4^{th}		
Lena	259496	245164	195275	91479	-	791414	3.019
Baboon	231185	125335	26104	-	-	382624	1.4596
Jetplane	260120	243413	200945	133285	10962	848725	3.2376
Man	258933	239917	176601	19526	-	694977	2.6511
Airplane	256568	247345	234049	193480	113788	1045230	3.9872
Tiffany	260424	244784	195306	109539	-	810053	3.0901

TABLE VI
RESULTS ON TWO DATABASES

Performance	BOSSbase	BOWS-2
Best Case	6.2354	5.9833
Worst Case	0.9731	0.9252
Average	3.6823	3.4568
PSNR	$+\infty$	$+\infty$
SSIM	1	1

Experimental results on two entire image datasets are shown in the Table VI. It can be seen that, for the BOSSbase database, the ERs of the best case and the worst case are 6.2354 bpp and 0.9734 bpp, respectively. The average ER of the BOSSbase database is 3.6823 bpp. For the BOWS-2 database, the ERs of the best case and the worst case are 5.9833 bpp and 0.9252 bpp, respectively. The average ER of the BOWS-2 database is 3.4568 bpp. Since the original images can be recovered losslessly, we have PNSR $\rightarrow +\infty$ and SSIM=1. From the above analysis, it can be concluded that the proposed method has good performance in terms of EC. This means that the proposed method can be applied to some applications with demands of large ECs.

C. Performance Comparisons

To demonstrate superiority of the proposed method, we compare the proposed method with six state-of-the-art methods, including Yi and Zhou's method [26], Chen *et al.*'s method [27], Yi and Zhou's method [42], Puteaux and Puech 's method [43], Yin *et al.*'s method [44], and Wu *et al.*'s method [45]. As the EC of Yi and Zhou's method [42] is affected by the parameters of α, β and block size, we select the maximum EC of each image by choosing the optimum parameters in the following experiments. Here, we do not compare the quality of the recovered image since the original image can be losslessly recovered during the recovery phase (PSNR $\rightarrow +\infty$ and SSIM $= 1$ for all evaluated methods). Therefore, comparison is only evaluated with ER in this section. To compare ER performance, we first apply the proposed method and six existing RDHEI methods to several standard images, as shown in Fig. 5. The comparison results are illustrated in the Fig. 8. It can be seen that the proposed method outperforms the compared methods [26], [27], [42]–[45] for all images.

Our maximal ER among these test images is 3.9872 bpp contributed by Airplane. For Airplane, the maximal ER among the compared methods is 3.725 bpp obtained by Yin *et al.*'s method [44]. Compared with [44], the proposed method gains 0.2622 bpp. Our minimum ER among these test images is 1.4596 bpp generated by Baboon, while the maximal ER of Baboon among the compared methods is 1.066 bpp also obtained by Yin *et al.*'s method [44]. The proposed method gains 0.3936 bpp for Baboon compared with Yin *et al.*'s method [44]. In [43], to guarantee reversibility, the unpredicted pixels must be detected and their locations are embedded by MSB substitution. Since only those predictable pixels conceal secret bits by MSB replacement technique, the ER of [43] is determined by the number of the predictable pixels and does not exceed 1.0 bpp. Yi and Zhou [26] designed a binary-block embedding (BEE) scheme which divides binary-blocks into good and bad blocks. They applied this scheme to bit-planes of image. The LSB planes of the original image are embedded into its MSB planes by using BBE in order to reserve the LSB planes for embedding secret data. As to Yi and Zhou's method [26], for Lena, the ECs of the $8^{th} \sim 5^{th}$ bit-planes are 182952 bits, 133516 bits, 90019 bits and 3138 bits, respectively. It can be seen from Table V that their ECs of bit-planes are all smaller than those of the proposed method. In addition, for Baboon, their ER is only 0.4838 bpp due to many bad blocks in textural image. Yi and Zhou's method [42] and Wu *et al.*'s method [45] are both parametric binary tree labeling (PBTL) based on RDHEI methods and Chen *et al.* [27] compressed MSB bit-planes to generate room for data embedding. It can be seen that, the ERs of the three methods don not exceed 3.0 bpp for Lena, Jetplane, Airplane, Tiffany, while the ERs of the proposed method are all larger than 3.0 bpp. For Baboon, the ERs of these methods do not exceed 1.0 bpp, while the proposed method can reach 1.459 bpp. For Man, the ERs of three methods are all smaller than 2.5 bpp, while the proposed method can reach 2.6511 bpp. Yin *et al.* [44] only considered the same MSB bit-planes between pixel value and its prediction value for data hiding. In fact, pixel value and its prediction value are very closed in some cases. But their same MSB bit-planes are few such that the ER of [44] is limited. For example, the pixel value and prediction value are 160 and 159, which can be encoded by binary sequences "10100000" and "10011111". Clearly, only two MSB bit-planes are same. Therefore, only two MSB bit-planes can be used for data hiding in [44]. However, the difference between 160 and 159 is 1 and the redundancy generated by prediction is not fully exploited. For the proposed method, by using hierarchical bit-plane label maps, more than two MSBs of the pixel value 160 and the prediction value 159 are likely to be used to accommodate secret bits. For Lena and Jetplane, their $8^{th} \sim 5^{th}$ and $8^{th} \sim 4^{th}$ bit-planes can be used for data hiding as illustrated in the Table V. It is clear that the proposed method can take full advantage of redundancy information. This validates that the proposed method outperforms the method [44].

To further demonstrate the superiority of the proposed method, BOSSBase [49] and BOWS-2 [50] datasets are used for comparison. The average ERs of different methods are compared and the results are shown in the Fig. 9. It can be seen that the ER results on the two databases also illustrate that the ER of the proposed method reaches the best performance. The average ERs of Puteaux and Puech's method [43] do not exceed 1.0 bpp for both BOSSbase and BOWS-2. The

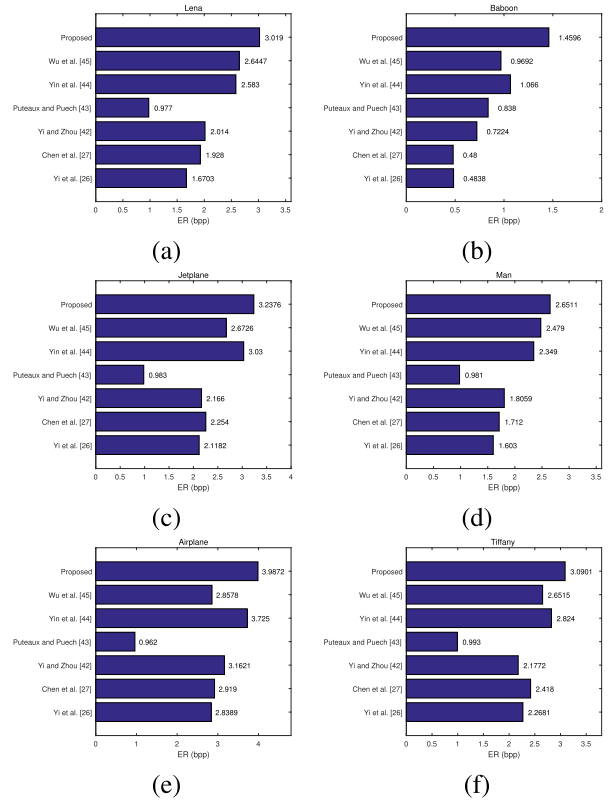

Fig. 8. ER comparison on standard images (a) Lena; (b) Baboon; (c) Jetplane; (d) Man; (e) Airplane; (f) Tiffany.

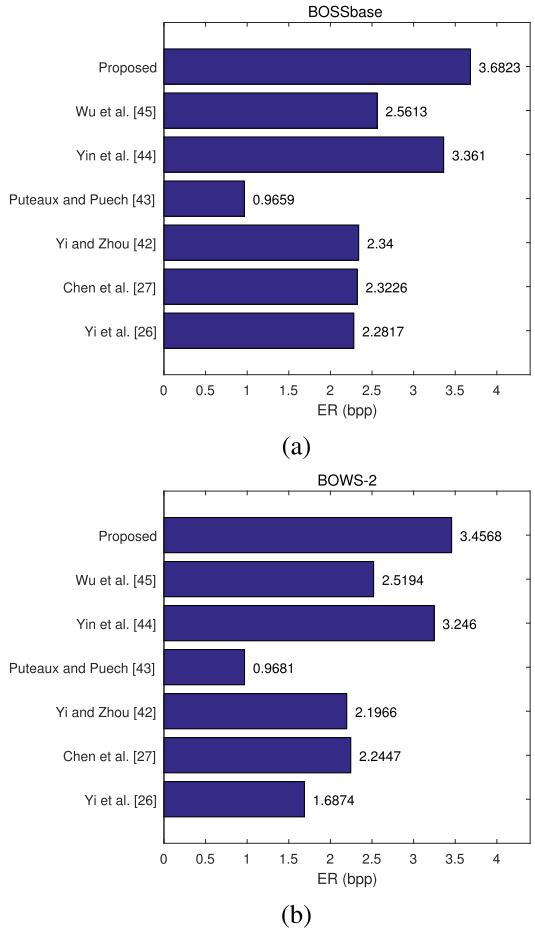

Fig. 9. Average ER comparison on two open datasets (a) BOSSbase; (b) BOWS-2.

average ERs of the methods [26], [27], [42], [45] do not exceed 3.0 bpp for BOSSbase and BOWS-2. Although the ERs of the method [44] can reach 3.361 bpp and 3.246 bpp for BOSSbase and BOWS-2, they do not take full advantage of redundancy information. From the above analysis, it can be found that the proposed method has better ER performance and the compared methods.

Moreover, computational time is evaluated. The assessed methods are implemented with MATLAB language and run on a computer with configuration that the processors are two dual-core CPUs with 2.10 GHz, the RAM capacity is 64 GB, and the used operating system is Windows 10. We exploit our method and the compared methods [26], [27], [42]–[45] to embed secret bits with respective full capacity into six images as shown in Fig. 5 and then calculate their average time of embedding one bit. Table VII presents computational time comparison of data embedding, where the first column lists the assessed methods, the second column is the total time of six images, the third column is the total EC of six images, and the final column is the average time of embedding one bit. It can be seen that all methods run very fast. For example, the slowest method is [45] which just needs 0.2571×10^{-4} seconds to embed one bit, meaning that its total time of embedding 10^5 bits is only 2.571 seconds. The proposed method is faster than [45], but it is slower than other compared methods [26], [27], [42]–[44]. In general, an effective programming language (e.g., C language or C++ language) is used in practice and will provide a much faster speed than the MATLAB language.

TABLE VII
TIME COMPARISON AMONG DIFFERENT METHODS

Method	Total time of six images (second)	Total EC of six images (bit)	Average time of embedding one bit (second)
[26]	29.155	2878944	0.1013×10^{-4}
[27]	34.5	3069968	0.1124×10^{-4}
[42]	49.69	2848718	0.1744×10^{-4}
[43]	21.88	1503133	0.1456×10^{-4}
[44]	67.48	4083417	0.1653×10^{-4}
[45]	96.21	3742053	0.2571×10^{-4}
Proposed	105.68	4573023	0.2311×10^{-4}

V. CONCLUSION

In this paper, we have proposed a new RDHEI method based on hierarchical embedding, which is error-free and achieves a very high payload. An important contribution is the new technique of hierarchical label map generation. The label map is determined by prediction technique and it is compressed and embedded into the encrypted image. In addition, the hierarchical embedding is proposed to divide prediction errors into small-magnitude errors, medium-magnitude errors and large-magnitude errors, and those pixels with small-magnitude/large-magnitude prediction errors are both used to accommodate secret bits. Many experiments have been done.

The results have shown that the proposed RDHEI method outperforms some state-of-the-art RDHEI methods in payload. Our average payloads on the BOWS-2 and BOSSbase datasets reach 3.4568 bpp and 3.6823 bpp, respectively.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their valuable comments and suggestions.

REFERENCES

[1] Z. Tang, X. Zhang, X. Li, and S. Zhang, "Robust image hashing with ring partition and invariant vector distance," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 1, pp. 200–214, Jan. 2016.

[2] C. Qin, P. Ji, X. Zhang, J. Dong, and J. Wang, "Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy," *Signal Process.*, vol. 138, pp. 280–293, Sep. 2017.

[3] Z. Tang, L. Chen, X. Zhang, and S. Zhang, "Robust image hashing with tensor decomposition," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 3, pp. 549–560, Mar. 2019.

[4] S. Li and X. Zhang, "Toward construction-based data hiding: From secrets to fingerprint images," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1482–1497, Mar. 2019.

[5] J. Tao, S. Li, X. Zhang, and Z. Wang, "Towards robust image steganography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 2, pp. 594–600, Feb. 2019.

[6] J. Fridrich, M. Goljan, and R. Du, "Lossless data embedding—New paradigm in digital watermarking," *EURASIP J. Adv. Signal Process.*, vol. 2002, no. 2, 2002, Art. no. 986842.

[7] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 253–266, Feb. 2005.

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

[9] 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, Aug. 2004.

[10] 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.

[11] C. Qin, C.-C. Chang, Y.-H. Huang, and L.-T. Liao, "An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1109–1118, Jul. 2013.

[12] Y. Jia, Z. Yin, X. Zhang, and Y. Luo, "Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting," *Signal Process.*, vol. 163, pp. 238–246, Oct. 2019.

[13] B. Ou, X. Li, Y. Zhao, and R. Ni, "Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion," *Signal Process., Image Commun.*, vol. 29, no. 7, pp. 760–772, Aug. 2014.

[14] F. Peng, X. Li, and B. Yang, "Improved PVO-based reversible data hiding," *Digit. Signal Process.*, vol. 25, pp. 255–265, Feb. 2014.

[15] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Process.*, vol. 93, no. 1, pp. 198–205, Jan. 2013.

[16] D. Wang, X. Zhang, C. Yu, and Z. Tang, "Reversible data hiding by using adaptive pixel value prediction and adaptive embedding bin selection," *IEEE Signal Process. Lett.*, vol. 26, no. 11, pp. 1713–1717, Nov. 2019.

[17] J. Qin and F. Huang, "Reversible data hiding based on multiple two-dimensional histograms modification," *IEEE Signal Process. Lett.*, vol. 26, no. 6, pp. 843–847, Jun. 2019.

[18] 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.

[19] W. Qi, X. Li, T. Zhang, and Z. Guo, "Optimal reversible data hiding scheme based on multiple histograms modification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2300–2312, Aug. 2020.

[20] 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. Forensics Security*, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[21] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Process.*, vol. 94, pp. 118–127, Jan. 2014.

[22] 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.

[23] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 9, pp. 1622–1631, Sep. 2016.

[24] C.-W. Shiu, Y.-C. Chen, and W. Hong, "Encrypted image-based reversible data hiding with public key cryptography from difference expansion," *Signal Process., Image Commun.*, vol. 39, pp. 226–233, Nov. 2015.

[25] C. Yu, X. Zhang, Z. Tang, Y. Chen, and J. Huang, "Reversible data hiding with pixel prediction and additive homomorphism for encrypted image," *Secur. Commun. Netw.*, vol. 2018, pp. 1–13, Sep. 2018.

[26] S. Yi and Y. Zhou, "Binary-block embedding for reversible data hiding in encrypted images," *Signal Process.*, vol. 133, pp. 40–51, Apr. 2017.

[27] 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 Represent.*, vol. 58, pp. 334–344, Jan. 2019.

[28] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Process. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.

[29] 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.

[30] X. Liao and C. Shu, "Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels," *J. Vis. Commun. Image Represent.*, vol. 28, pp. 21–27, Apr. 2015.

[31] C. Qin and X. Zhang, "Effective reversible data hiding in encrypted image with privacy protection for image content," *J. Vis. Commun. Image Represent.*, vol. 31, pp. 154–164, Aug. 2015.

[32] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[33] Z. Qian and X. Zhang, "Reversible data hiding in encrypted images with distributed source encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 4, pp. 636–646, Apr. 2016.

[34] X. Wu and W. Sun, "High-capacity reversible data hiding in encrypted images by prediction error," *Signal Process.*, vol. 104, pp. 387–400, Nov. 2014.

[35] F. Huang, J. Huang, and Y.-Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 12, pp. 2777–2789, Dec. 2016.

[36] F. Di, F. Huang, M. Zhang, J. Liu, and X. Yang, "Reversible data hiding in encrypted images with high capacity by bitplane operations and adaptive embedding," *Multimedia Tools Appl.*, vol. 77, no. 16, pp. 20917–20935, Aug. 2018.

[37] D. Xiao, Y. Xiang, H. Zheng, and Y. Wang, "Separable reversible data hiding in encrypted image based on pixel value ordering and additive homomorphism," *J. Vis. Commun. Image Represent.*, vol. 45, pp. 1–10, May 2017.

[38] Z. Tang, S. Xu, H. Yao, C. Qin, and X. Zhang, "Reversible data hiding with differential compression in encrypted image," *Multimedia Tools Appl.*, vol. 78, no. 8, pp. 9691–9715, Apr. 2019.

[39] Y. Fu, P. Kong, H. Yao, Z. Tang, and C. Qin, "Effective reversible data hiding in encrypted image with adaptive encoding strategy," *Inf. Sci.*, vol. 494, pp. 21–36, Aug. 2019.

[40] Z.-L. Liu and C.-M. Pun, "Reversible data-hiding in encrypted images by redundant space transfer," *Inf. Sci.*, vols. 433–434, pp. 188–203, Apr. 2018.

[41] C. Qin, X. Qian, W. Hong, and X. Zhang, "An efficient coding scheme for reversible data hiding in encrypted image with redundancy transfer," *Inf. Sci.*, vol. 487, pp. 176–192, Jun. 2019.

[42] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multimedia*, vol. 21, no. 1, pp. 51–64, Jan. 2019.

[43] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 7, pp. 1670–1681, Jul. 2018.

[44] 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.

[45] 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.

[46] P. Puteaux and W. Puech, "A recursive reversible data hiding in encrypted images method with a very high payload," *IEEE Trans. Multimedia*, vol. 23, pp. 636–650, 2021.

[47] 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.
[48] 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.
[49] P. Bas, T. Filler, and T. Pevn, "Break our steganographic system': The ins and outs of organizing BOSS," in *Information Hiding*. Prague, Czech Republic: Springer, 2011.
[50] P. Bas and T. Furon. *Image Database of Bows-2*. Accessed: Jun. 20, 2017. [Online]. Available: http://bows2.ec-lille.fr./

Chunqiang Yu (Member, IEEE) received the B.S. degree from Jiangxi Normal University, Nanchang, China, in 2010, and the M.Eng. degree from Guangxi Normal University, Guilin, China, in 2013, where he is currently pursuing the Ph.D. degree. His research interests include image processing and data hiding.

Xianquan Zhang received the M.Eng. degree from Chongqing University, Chongqing, China, in 1996. He is currently a Professor with the Department of Computer Science, Guangxi Normal University. He has contributed more than 100 articles. His research interests include image processing and multimedia security.

Xinpeng Zhang (Member, IEEE) received the B.S. degree in computational mathematics from Jilin University, China, in 1995, and the M.E. and Ph.D. degrees in communication and information system from Shanghai University, China, in 2001 and 2004, respectively. Since 2004, he was with the faculty of the School of Communication and Information Engineering, Shanghai University, where he is currently a Professor. He is also with the faculty of the School of Computer Science, Fudan University. He was with The State University of New York at Binghamton, as a Visiting Scholar, from 2010 to 2011, and also an experienced Researcher with Konstanz University, sponsored by the Alexander von Humboldt Foundation, from 2011 to 2012. His research interests include multimedia security, image processing, and digital forensics. He has published over 200 articles in these areas. He was an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY from 2014 to 2017.

Guoxiang Li received the M.Eng. degree from Guangxi Normal University, Guilin, China, in 2010, where he is currently pursuing the Ph.D. degree. He is also an Associate Professor with the Faculty of Information and Statistics, Guangxi University of Finance and Economics. He has contributed more than 20 articles. His research interests include image processing and data mining.

Zhenjun Tang (Member, IEEE) received the B.S. and M.Eng. degrees from Guangxi Normal University, Guilin, China, in 2003 and 2006, respectively, and the Ph.D. degree from Shanghai University, Shanghai, China, in 2010. He is currently a Professor with the Department of Computer Science, Guangxi Normal University. He has contributed more than 70 international journal articles. His research interests include image processing, video processing, and multimedia security. He is a reviewer of more than 30 SCI journals, such as IEEE, ACM, IET, Elsevier, and Springer journals.