Signal Processing 150 (2018) 171–182

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Parametric reversible data hiding in encrypted images using adaptive bit-level data embedding and checkerboard based prediction

Shuang Yi, Yicong Zhou*

Department of Computer and Information Science, University of Macau, Macau 999078, China

ARTICLE INFO

Article history:
Received 14 October 2017
Revised 29 March 2018
Accepted 17 April 2018
Available online 19 April 2018

Keywords:
Reversible data hiding
Encrypted images
High capacity
Privacy protection
Adaptive bit-level data embedding
Adaptive checkerboard based prediction

ABSTRACT

In this paper, we first propose an adaptive bit-level data embedding (ABDE) method to embed secret data into a cover image and an adaptive checkerboard based prediction (ACBP) method to predict 3/4 of the pixels in an image using its remaining 1/4 of the pixels. Based on ABDE and ACBP, we further propose a parametric reversible data hiding method in encrypted images (PRDHEI). When parameter $\varepsilon = 1$ ($\varepsilon \in [1, 255]$), PRDHEI is a full reversible method that both the original image and secret data can be completely recovered. The embedding rate is much higher than state-of-the-art RDHEI methods. Moreover, when $\varepsilon > 1$, the embedding rate of PRDHEI increases significantly. The receiver can fully recover the secret data and reconstruct the original image with very high quality. When $\varepsilon = 255$, PRDHEI reaches its maximum embedding rate of 4.5 bpp while the recovered images are of an average peak signal-to-noise ratio larger than 32 dB.

1. Introduction

Image encryption and data hiding are two main techniques for privacy protection [1]. The former aims to change the meaningful image into a noise-like one for preventing unauthorized access, while the later conceals secret data into a cover image in an imperceptible way [2,3]. In image encryption, the original image is the one to be protected, while, in data hiding, the secret data is the information that should be undisclosed. Reversible data hiding (RDH) is a technique that require to perfectly recover the original image after extracting the secret data from the marked image. It can be used in many applications, such as law enforcement, medical imaging and remote sensing, where the original cover image is demanded to be lossless recovered for some legal consideration, medical diagnosis and high-precision nature requirement [4]. A lot of image encryption algorithms [5,6] and reversible data hiding methods [4,7–9] have been proposed in recent years. They both have good performances in their research fields.

Nowadays, reversible data hiding in the encrypted image (RDHEI) has received increasing attentions in the research community, where both the original image and secret data need to be protected [10,11]. In RDHEI, image encryption and data embedding are accomplished by different users separately. The content-owner first encrypts the original image to a noise-like one, then the data-hider embeds secret data into the encrypted image without knowing its original content. This can be used in many scenarios such as Cloud storage [12,13], medical image management system [14,15], law forensics, military applications [1] and secure remote sensing [16]. Taking the medical images as an example, due to the utilization telesurgery and telediagnosis, medical imaging plays an increasing important roles in real applications. The transmission and sharing of medical images must consider the confidentiality, which means that only authorized users (e.g., the attending doctor) are able to access the medical image. Thus, cryptography is needed. For ease of management and protecting the patient privacy, some additional information (hereafter, we call it the secret data) such as patient personal information and image source are added to the encrypted image by the database administrator, who has no right to access the medical image content. At the receiver side, the authorized users, such as doctor and database administrator, are required to reveal the original image or secret data separately. Another example for secure remote sensing are provided in [16], in which the original satellite images are encrypted and sent to the base station(s). Some secret data, such as base station ID, location, local temperature, wind speed, etc., are embedded into the encrypted image before transmission and storage. These images cannot be compressed due to their special utilization. For example, the most commonly used medical image such as the Digital Imaging and Communications in Medicine (DICOM) image is an uncompressed image, and after converting it into JPEG file, a lot of data will be lost [17]. Thus, like other RDHEI methods, we focus on the uncompressed images.

* Corresponding author.
E-mail address: yicongzhou@umac.mo (Y. Zhou).

https://doi.org/10.1016/j.sigpro.2018.04.016
0165-1684/

Many RDHEI methods have been proposed in recent years. Depending on whether a preprocessing procedure is needed to reserve room before image encryption, these methods can be divided into two categories, namely: vacating room after encryption (VRAE) and vacating room before encryption (VRBE) [1,18]. It is more efficient to reserve spare space in the original image than that in the encrypted image because many existing RDH [19–23] and sparse representation [24] techniques can be used for reserving room. Thus, VRBE methods can reach a larger embedding rate than VRAE methods in general. However, the content-owner may not be willing to perform such extra operation for reserving room because s/he may have difficulty to do such complicated operation due to the limited computation ability [25].

Therefore, VRAE seems to be more applicable in real applications. Hence, many researchers devote to develop efficient VRAE methods. A lot of VRAE methods encrypt the original image using the stream cipher [15,16,26–32]. These methods may cause incorrect extracted data or recovered image when the original image is textured. Because they recover the original image mainly by analyzing the spacial correlations of the directly decrypted image, and textured areas will result in inaccurate estimation results. Other methods encrypt the original image by permutation [33,34] and Paillier encryption [35–37]. The former is limited in embedding rate and the later may cause data expansion in the encrypted images. In addition, these existing VRAE methods endure low quality of the final recovered image.

In order to solve these problems, in this paper, we propose a parametric reversible data hiding method in encrypted image. It inherits the merits of VRAE which does not need a reserve room process at the content-owner side. In addition, it is a separable method that data extraction and image recovery can be performed independently. The contributions of this paper are summarized as follows:

(1) We propose an adaptive bit-level data embedding (ABDE) method to embed secret data into the cover image by bit replacement. Two embedding strategies are designed for data embedding, and ABDE automatically selects the one with a higher embedding rate.
(2) We propose an adaptive checkerboard based prediction (ACBP) to predict 3/4 of the pixels in an image using its remaining 1/4 of the pixels.
(3) Based on ABDE and ACBP, we further propose a parametric reversible data embedding method in encrypted images (PRDHEI), where ABDE is used for data embedding, ACBP is adopted for image recovery and parameter ε is to control the embedding rate. It is a full reversible method that both the secret data and original image can be perfectly recovered when parameter $\varepsilon = 1$. The embedding rate is much higher than state-of-the-art RDHEI methods.
(4) In addition, when $\varepsilon > 1$, PRDHEI slightly reduces the recovered image quality while significantly increases the embedding rate. When $\varepsilon = 255$, it is able to embed as large as 4.5 bpp of the secret data and obtain an average PSNR larger than 30 dB in the recovered images.

The rest of this paper is organized as follows: Section 2 reviews the state-of-the-art VRAE methods. Section 3 and Section 5 introduce the proposed adaptive bit-level data embedding and checkerboard based prediction methods. Section 5 describes the proposed parametric reversible data embedding method in encrypted images. Section 6 provides simulation results and comparisons with several related works. Section 7 discusses several issues of the proposed algorithm. Finally, Section 8 concludes this paper.

2. Related work

In this section, we review several state-of-the-art RDHEI VRAE methods, in which the content-owner requires no preprocessing before encrypting the original image.

Several VRAE methods [15,16,26–32] encrypt the original image using a stream-cipher. In [15,26], the encrypted image is first divided into a number of non-overlapped blocks, and pixels in each block are separated into two categories, namely s_0 and s_1, with equal size. The data-hider then embeds one bit of the secret data into one block by flipping the 3 least significant bits (LSBs) of pixels in s_0 (if the secret data is 0) or s_1 (if the secret data is 1). Data extraction and image recovery are accomplished by analyzing the spatial correlations of the directly decrypted image. In Yu et al.'s method [27], p% of the pixels in s_0 or s_1 are flipped for data embedding. Thus, [15,26] are the special case of [27] with $p = 100$. In order to decrease the error rate of the extracted data, Li et al. [28] duplicate the secret data before embedding into the encrypted image. Liao et al.[38] improved Zhang's method [15] by considering multiple neighboring pixels according to the locations of different pixels, so that the embedding rate is increased. In Zhou et al.'s method [16], $n(n \geq 1)$ bits of the secret data are embedded into one image block using the public key modulation mechanism, and the support vector machine technique is utilized for data extraction and image recovery. In previous methods [15,16,26–28], incorrect secret data may be extracted when the image block size is small. In addition, data extraction and image recovery should be done simultaneously.

In order to improve flexibility, several separable VRAE methods have been proposed, where the data extraction is completely independent from image recovery. In Zhang's method [29], 3 LSB planes of the encrypted image are compressed to accommodate the secret data. In [30], the low-density parity-check code is utilized to compress half of the bits in the 4th LSB plane of the encrypted image. Zheng et al. [32] selected several LSB planes of the encrypted image and compressed every 3 bits into 2 to accommodate the secret data. However, this method needs a large size of auxiliary information for image recovery, which is not practical in real application. In Zhou et al.'s method [16], all pixel values within the block are modified after data embedding due the utilization of bit-level XOR and public key modulation, while, in Zhang et al.'s method [31], only the 3 LSBs of pixels in each block are changed by using the pseudorandom sequence modulation mechanism. Different from the previous methods that encrypt the original image using stream-cipher, other algorithms encrypt the original image by permutation. Yin et al. [33] used two permutations to encrypt the original image: a coarse-grained permutation to scramble image blocks within the whole image, and a fine-grained permutation to scramble pixels within each block. Then, the histogram shifting method [4] is utilized to embed secret data in each block by randomly selecting two pixels to be the peak values. In this method, the histograms of the original image and encrypted image will keep the same, which is not secure enough. Later, Yin et al. [34] improved this method using a modulation function to change pixel values in each block. However, it requires that the original image contains no saturation values. This means the pixel values should be within the range of [1, 254]. These two methods [33,34] also have limited embedding rates. For method in [39], the stream cipher is adopt to encrypt several most significant bit (MSB) planes in the original image, and secret data is embedded into the rest LSB planes of the image by histogram shifting (HS) method. In [40], Wu and Sun proposed a joint method and a separable method for data embedding. The joint method embeds secret data by modifying the ith$(1 \leq i \leq 6)$ LSBs of the pixels in encrypted image, while the separable method embeds secret data by modifying its two MSBs. Huang et al. [41] use block per-

Table 1
Allocation of intervals for embedding Strategy-I and $1 \le \varepsilon \le 255$.

ε	0	100	101	110	111
$1 \le \varepsilon \le 4$	NP	$\min\{-\varepsilon, -2\} \leqslant e \leqslant -2$	$e = -1$	$e = 0$	$1 \le e \le \varepsilon$
$5 \le \varepsilon \le 13$	NP	$-\varepsilon \leqslant e \leqslant -3$	$-2 \leqslant e \leqslant -1$	$0 \le e \le 1$	$2 \le e \le \varepsilon$
$14 \le \varepsilon \le 22$	NP	$-\varepsilon \leqslant e \leqslant -4$	$-3 \leqslant e \leqslant -1$	$0 \le e \le 2$	$3 \le e \le \varepsilon$
$23 \le \varepsilon \le 32$	NP	$-\varepsilon \leqslant e \leqslant -5$	$-4 \leqslant e \leqslant -1$	$0 \le e \le 3$	$4 \le e \le \varepsilon$
$33 \le \varepsilon \le 255$	NP	$-\varepsilon \leqslant e \leqslant -6$	$-5 \leqslant e \leqslant -1$	$0 \le e \le 4$	$5 \le e \le \varepsilon$

mutation and bit-XOR to embed the original image, and the traditional RDH method such as difference histogram shifting (DHS) and prediction-error histogram shifting (PHS) are utilized to embed secret data. Another type of VRAE methods adopts the Paillier encryption to encrypt the original image [35–37]. However, these VRAE methods may cause data expansion and increase the communication bandwidth.

3. Adaptive bit-level data embedding

In this section, we propose an adaptive bit-level data embedding (ABDE) to embed secret data into the cover image $\mathbb{C}$ by pixel labeling and bit replacement. Here, we introduce the ABDE in two phases: (1) data embedding and (2) data extraction and image recovery.

3.1. Data embedding

3.1.1. Pixel grouping

Firstly, we divide an 8-bit depth $M \times N$ gray-scale cover image $\mathbb{C}$ into k non-overlapped blocks $\mathbb{C}_{(1)}, \mathbb{C}_{(2)}, \ldots, \mathbb{C}_{(k)}$ with a size of 2×2, where $k = MN/4$. We use $\mathbb{C}^j_{(i)} (i = 1, \ldots, k; j = 1, 2, 3, 4)$ to denote the jth pixel in the ith block in raster scanning order. Then, all pixels in the image are divided into four pixel sets, namely: (a) reference pixel (RP), (b) specific pixel (SP), (c) embeddable pixel (EP) and (d) non-embeddable pixel (NP), where RP $\cap$ SP $\cap$ EP $\cap$ NP $= \emptyset$. Here, RP denotes the set of pixels $\{\mathbb{C}^1_{(t)}\}^k_{t=1}$. All pixels in RP will be kept unchanged during the whole data embedding phase. SP indicates two specific pixels $\mathbb{C}^2_{(1)}$ and $\mathbb{C}^3_{(1)}$. They are utilized to store some parameters which will be discussed later. EP refers to a set of pixels whose e satisfies with the condition in Eq. (2), where e is calculated by Eq. (1), and ε is a predefined parameter. This means that pixels in EP can be approximately predicted by their reference pixel $\mathbb{C}^1_{(t)}$. NP denotes the pixel whose e fails to meet the condition in Eq. (2). Assume that there are n pixels in EP. The value of n will change when ε is different.

$$e = \mathbb{C}^j_{(t)} - \mathbb{C}^1_{(t)}, \quad j = 2, 3, 4 \tag{1}$$

$$-\varepsilon \leqslant e \leqslant \varepsilon, \quad \varepsilon \in \mathbf{N}^+ \tag{2}$$

3.1.2. Pixel labeling and data embedding

To embed secret data, firstly, pixels in EP and NP are labeled by some predefined labeling bits according to the difference value e and parameter ε. Here, we provide two strategies for pixel labeling as shown in Tables 1 and2. Given a parameter ε, for Strategy-I, we use one bit ('0') to label each pixel in NP and four sets of 3-bit labeling bits ('100', '101','110','111') to label pixels in EP under various ranges of e. Thus, for an 8-bit depth image, one bit of each pixel in NP is replaced by '0', and the remaining 7 bits will be kept unmodified. For each pixel in EP, 3 bits are replaced by the corresponding labeling bits and the remaining 5 bits are reserved to accommodate the secret data. Similarly, for Strategy-II, we use two bits ('00') to label each pixel in NP and three sets of 2-bit labeling bits ('01', '10', '11') to mark pixels in EP, respectively. Thus,

Table 2
Allocation of intervals for embedding Strategy-II and $1 \le \varepsilon \le 255$.

ε	00	01	10	11
$1 \le \varepsilon \le 4$	NP	$-\varepsilon \leqslant e \leqslant -1$	$e = 0$	$1 \le e \le \varepsilon$
$5 \le \varepsilon \le 13$	NP	$-\varepsilon \leqslant e \leqslant -2$	$-1 \leqslant e \leqslant 1$	$2 \le e \le \varepsilon$
$14 \le \varepsilon \le 22$	NP	$-\varepsilon \leqslant e \leqslant -3$	$-2 \leqslant e \leqslant 2$	$3 \le e \le \varepsilon$
$23 \le \varepsilon \le 32$	NP	$-\varepsilon \leqslant e \leqslant -4$	$-3 \leqslant e \leqslant 3$	$4 \le e \le \varepsilon$
$33 \le \varepsilon \le 255$	NP	$-\varepsilon \leqslant e \leqslant -5$	$-4 \leqslant e \leqslant 4$	$5 \le e \le \varepsilon$

Strategy-I and Strategy-II are able to embed totally $6n$ and $5n$ bits of the payload, respectively.

The embedding strategy type and parameter ε are important for data extraction and image recovery, thus, they will be stored as well. Because the strategy type can be stored by 1 bit (e.g., '0' for Strategy-I and '1' for Strategy-II), and ε can be stored by 8 bits, we use the two pixels in SP to store them by bit replacement. The original 9 bits in SP are preselected and stored with secret data. Thus, we can construct the payload $\mathbf{P}$, which consists of three parts: (1) 9 bits in SP; (2) the bit sequence $\mathbf{Q}$ that is formed by picking the been replaced bits in NP; (3) the secret data $\mathbf{M}$. Thus, the length of $\mathbf{Q}$ will be $(\frac{3}{4}MN - n - 2)$ or $2 \times (\frac{3}{4}MN - n - 2)$ bits when Strategy-I or Strategy-II is used. We then can calculate the effective embedding rate $r_{\mathbf{I}}$ and $r_{\mathbf{II}}$ by

$$r_{\mathbf{I}} = \frac{5n - (\frac{3}{4}MN - n - 2) - 9}{MN} \quad \text{(bpp)} \tag{3}$$

$$r_{\mathbf{II}} = \frac{6n - 2(\frac{3}{4}MN - n - 2) - 9}{MN} \quad \text{(bpp)}, \tag{4}$$

where $r_{\mathbf{I}}$ and $r_{\mathbf{II}}$ are the embedding rates using Strategy-I and Strategy-II, respectively. When $r_{\mathbf{I}} \geqslant r_{\mathbf{II}}$, we use Strategy-I to embed the secret data; otherwise, we use Strategy-II. Therefore, given a cover image and parameter ε, we can determine the embedding strategy (ES) by

$$\text{ES} = \begin{cases} \text{Strategy-I}, & \text{if } n \leqslant \frac{3}{8}MN - 1 \\ \text{Strategy-II}, & \text{otherwise} \end{cases} \tag{5}$$

Thus, the actual embedding rate r (bpp) is

$$r = \max\{r_{\mathbf{I}}, r_{\mathbf{II}}\} \tag{6}$$

As can be seen, when $\varepsilon = 255$, the difference e calculated by Eq. (1) will always fall into the range in Eq. (2). This means that no pixel will be classified into the set NP and thus $n = \frac{3}{4}MN - 2$. Therefore, ABDE is able to achieve the maximum embedding rate of

$$r_{\max} = \frac{(3MN/4 - 2) \times 6 - 9}{MN} \approx 4.5 \text{ bpp}. \tag{7}$$

The detailed procedures of ABDE are provided in Algorithm 1. We use a 1-bit parameter T to denote the embedding strategy type, for example, '0' for Strategy-I and '1' for Strategy-II.

3.2. Data extraction and image recovery

After obtaining the marked image, users are able to extract the secret data and recover the original image.

Algorithm 1 ABDE.

Input: Cover image $\mathbb{C}$, secret data, parameter ε.

Divide $\mathbb{C}$ into four categories: RP, SP, EP and NP.Determine the embedding strategy ES according to Eq. (5). Construct the payload **P**.
Embed the parameters ε and T into SP by bit replacement.
for Each pixel in EP **do**
  if ES = Strategy-I **then**
    Replace the first 3 bits by '100', '101', '110' or '111' according to Table 1.
    Replace the remaining 5 bits by payload bits.
  else if ES = Strategy-II **then**
    Replace the first 2 bits by '01', '10' or '11' according to Table 2.
    Replace the remaining 6 bits by payload bits.
  end if
end for
for Each pixel in NP **do**
  if ES = Strategy-I **then**
    Replace the first 1 bit by '0'.
  else if ES = Strategy-II **then**
    Replace the first 2 bits by '00'.
  end if
end for

Output: Marked image $\hat{\mathbb{C}}$.

3.2.1. Data extraction

To extract the secret data, firstly, we obtain the parameters ε and T from the pixels in SP. According to the embedding strategy type and ε, we can distinguish the pixels in EP and NP, respectively. We then extract 5 (or 6) bits of the payload from each pixel of EP sequentially when $T = 0$ (or 1). Finally we obtain the secret data from the extracted payload bits.

3.2.2. Image recovery

For image recovery, we first use the first 9 bits and the subsequent bit sequence **Q** extracted from the payload to recover pixels in SP and NP, respectively. By now, only the pixels in EP need to be recovered. For each of the pixel in EP, according to its labeling bits, we can determine the range $[v_l, v_r]$ in which e belongs to. For example, if the labeling bits equal to '01' and $\varepsilon = 7$, according to Table 2, $-7 \leqslant e \leqslant -2$. Thus, $[v_l, v_r] = [-7, -2]$. We then obtain the approximately difference value $\hat{e}$ by

$$\hat{e} = \begin{cases} \lceil (v_l + v_r)/2 \rceil, & \text{if } v_r < 0 \\ \lfloor (v_l + v_r)/2 \rfloor, & \text{otherwise,} \end{cases} \tag{8}$$

where $\hat{e}$ is the median value of the difference interval $[v_l, v_r]$. Next, for each pixel in EP, we recover it by

$$\tilde{\mathbb{C}}^j_{(i)} = \hat{e} + \hat{\mathbb{C}}^1_{(i)}. \tag{9}$$

The reference pixels in RP are not modified during the data embedding phase, thus, $\hat{\mathbb{C}}^1_{(i)} = \mathbb{C}^1_{(i)}$. When $\hat{e} = e$, the recovered pixel $\tilde{\mathbb{C}}^j_{(i)}$ equals to the original pixel $\mathbb{C}^j_{(i)}$; otherwise, we obtain $\tilde{\mathbb{C}}^j_{(i)} \approx \mathbb{C}^j_{(i)}$, thus result in an approximately recovered image. As we can see from Tables 1 and 2, when $\varepsilon = 1$, the labeling bits are one-to-one mapped to e. Therefore, $\hat{e} = e$, and the original image can be perfectly recovered.

3.3. Example

An illustrative example of ABDE is shown in Fig. 1, in which the original image is with size of 4×2 and $\varepsilon = 2$. According to ε, we calculate $n = 3 < \frac{3}{8} * 8 - 1 = 2$. Thus, the embedding Strategy-II is applied. Parameters ε and T are converted into an 8-bit binary sequence '00000010' and 1-bit '1' and embedded into SP by bit replacement, respectively. The effective embedding rate is $r = \frac{3\times6-2-9}{8} = 0.875$ bpp.

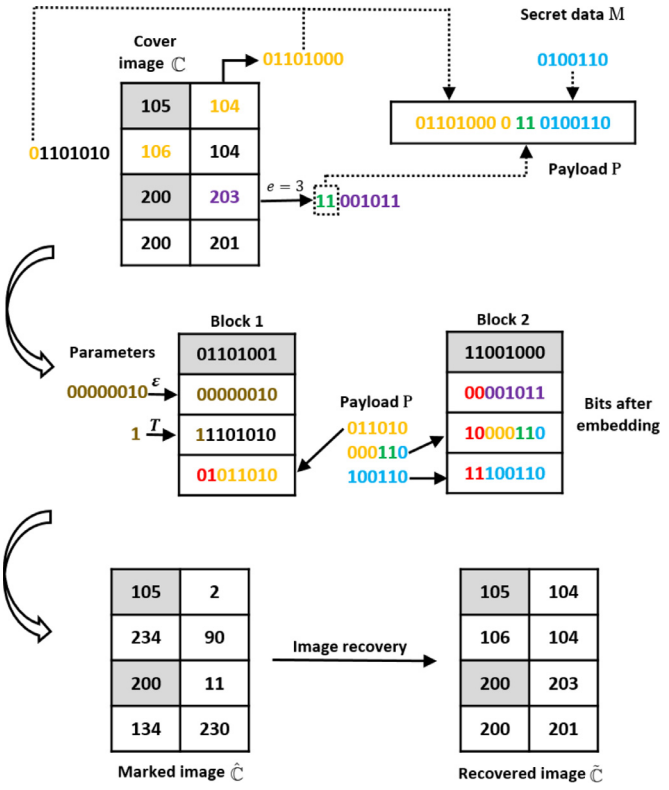

Fig. 1. Illustrative example of ABDE using Strategy-II, the effective embedding rate $r = 0.875$ bpp.

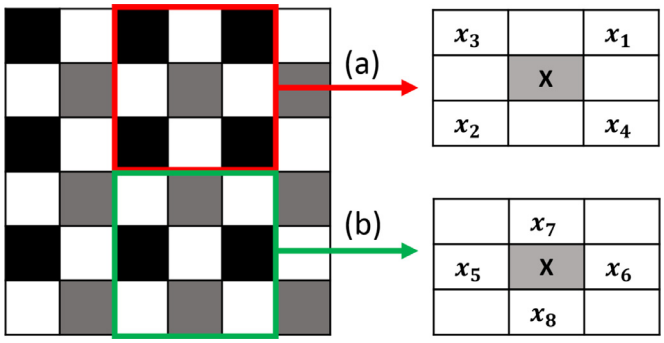

Fig. 2. Illustration of ACBP.

4. Adaptive checkerboard based prediction

In this section, we propose an adaptive checkerboard based prediction (ACBP) to predict 3/4 of the pixels in an image by its remaining 1/4 pixels. It is an improved version of checkerboard based prediction (CBP) proposed in [8] by considering image structure features within small image blocks. As shown in Fig. 2, we use the pixels marked in black to predict the pixels marked in gray and white. ACBP consists of two steps as shown in Fig. 2(a) and (b). In the first step, pixels marked in gray are predicted by their four diagonal pixels, and the results can be calculated by Eq. (10). In the second step, pixels in white are predicted by their four neighboring pixels in horizontal and vertical directions using Eq. (11). Here,

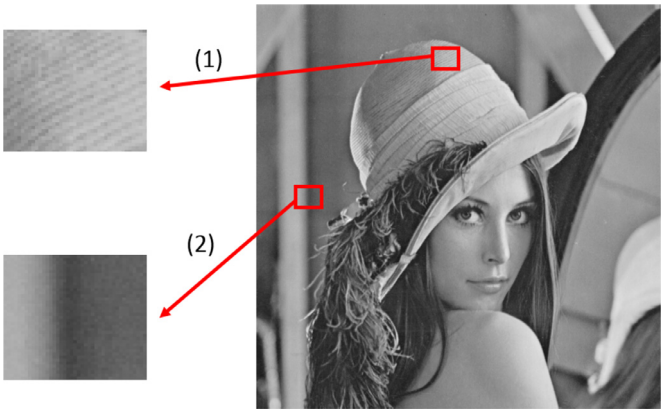

Fig. 3. Example of texture features of small regions in *Lena* image.

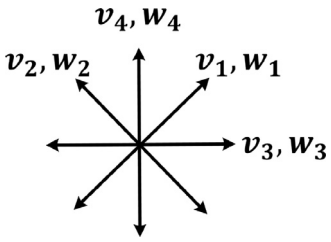

Fig. 4. Four directions of image smoothness and their corresponding weight coefficients.

w_1, w_2, w_3 and w_4 are weight coefficients that are satisfied with $\{w_1, w_2, w_3, w_4\} \geq 0$, $w_1 + w_2 = 0.5$ and $w_3 + w_4 = 0.5$.

$$X = \text{round}(w_1 * (x_1 + x_2) + w_2 * (x_3 + x_4)) \tag{10}$$

$$X = \text{round}(w_3 * (x_5 + x_6) + w_4 * (x_7 + x_8)), \tag{11}$$

where round(.) converts the value to its nearest integer.

In general, a small region in a normal image has similar texture features. As can be seen from Fig. 3, the regions (1) and (2) contain oblique and vertical stripes, respectively. Thus, we divide the image into small blocks with a size of $s_1 \times s_2$, and apply ACBP in each block to predict the pixels. Weight coefficients w_1, w_2, w_3 and w_4 are set by analyzing the local texture feature within a block. Thus, blocks with different texture features will use different values of weight coefficients. Next, we discuss the settings of weight coefficients w_1, w_2, w_3 and w_4 used in ACBP. Given an image block $\mathbb{A}$ with size of $s_1 \times s_2$, we first calculate its four texture features v_1, v_2, v_3 and v_4 by

$$v_1 = \frac{\sum_{i=1}^{s_1-1} \sum_{j=2}^{s_2} \|\mathbb{A}(i,j) - \mathbb{A}(i+1, j-1)\|}{(s_1-1)(s_2-1)} \tag{12}$$

$$v_2 = \frac{\sum_{i=1}^{s_1-1} \sum_{j=1}^{s_2-1} \|\mathbb{A}(i,j) - \mathbb{A}(i+1, j+1)\|}{(s_1-1)(s_2-1)} \tag{13}$$

$$v_3 = \frac{\sum_{i=1}^{s_1} \sum_{j=1}^{s_2-1} \|\mathbb{A}(i,j) - \mathbb{A}(i, j+1)\|}{s_1(s_2-1)} \tag{14}$$

$$v_4 = \frac{\sum_{i=1}^{s_1-1} \sum_{j=1}^{s_2} \|\mathbb{A}(i,j) - \mathbb{A}(i+1, j)\|}{s_2(s_1-1)}, \tag{15}$$

where v_1, v_2, v_3 and v_4 are smoothness measurement functions of $\mathbb{A}$ at four directions shown in Fig. 4. The values of v_1, v_2, v_3 and v_4 are used to deduce the settings of w_1, w_2, w_3 and w_4. Next, we calculate three minimum values $G = \min\{v_1, v_2, v_3, v_4\}$, $G_1 = \min\{v_1, v_2\}$ and $G_2 = \min\{v_3, v_4\}$. According to G, G_1 and G_2, the detail settings of weight coefficients are listed in Table 3. As can be seen from this table, two groups of weights {0.45, 0.05} and {0.35, 0.15} are utilized to set the coefficients based on the values of G, G_1 and G_2, where the larger weight is corresponding to the smaller smoothness measure value. For example, when $G = v_3$, the image block $\mathbb{A}$ is smoother at the horizontal direction than that at the vertical direction. Thus, w_3 is set to the largest value of 0.45, while w_4 is set to the smallest value of 0.05. Then, we compare the smoothness in two diagonal directions and set the weights to 0.35 and 0.15 in the corresponding directions, respectively.

Table 3
Settings of weight coefficients w_1, w_2, w_3 and w_4 based on different values of texture features.

		G	w_1	w_2	w_3	w_4
G_1	v_1	v_3	0.35	0.15	0.45	0.05
		v_4	0.35	0.15	0.05	0.45
	v_2	v_3	0.15	0.35	0.45	0.05
		v_4	0.15	0.35	0.05	0.45
G_2	v_3	v_1	0.45	0.05	0.35	0.15
		v_2	0.05	0.45	0.35	0.15
	v_4	v_1	0.45	0.05	0.15	0.35
		v_2	0.05	0.45	0.15	0.35

5. Parametric reversible data hiding in encrypted images

In this section, we propose a parametric reversible data hiding in encrypted images (PRDHEI) using ABDE and ACBP. The framework of the proposed algorithm is shown in Fig. 5. It consists three phases: image encryption, data embedding, and data extraction/image recovery. These three phases are accomplished by content-owner, data-hider and receiver, respectively. In Phase I, the content-owner encrypts the original image by block-based permutation and modulation using K_e. In Phase II, the data-hider encrypts the secret data using the hiding key K_h and embeds the encrypted secret data into the stream-deciphered image according to the parameter ε. Finally, the stream encipher is utilized to further protect the secret data and encrypted image. In Phase III, using K_s and K_h, the receiver is able to extract the secret data losslessly. To recover the original image, ACBP is applied. Based on different settings of parameter in data embedding phase, the proposed algorithm is able to perfectly recover the original image or obtain an approximately recovered image with high visual quality.

5.1. Image encryption

Without loss of generality, we assume that an original image $\mathbb{O}$ is with size of $M \times N$ and its pixel values are within the range of [0, 255]. First, the content-owner divides $\mathbb{O}$ into a number of 2×2 non-overlapped blocks and permutates these blocks within image $\mathbb{O}$ using $\dot{K}_e$, where $\dot{K}_e = \mathcal{H}(\mathbb{O}) \oplus K_e$, $\oplus$ is the bit-level XOR operation, $\mathcal{H}(*)$ is the secure hash function to generate a hash sequence with respect to the input content. Thus, totally $k = MN/4$ blocks will be obtained. We denote the obtained block-permutated image as $\mathbb{I}$, and its blocks as $\mathbb{I}_{(1)}, \mathbb{I}_{(2)}, \ldots, \mathbb{I}_{(k)}$. Then, each pixel value in $\mathbb{I}$ is modified by

$$\mathbb{E}^i_{(t)} = (\mathbb{I}^i_{(t)} + \mathbf{R}_t) \mod 256, \quad \text{for} \quad t = 1, 2, \ldots, k, \tag{16}$$

where $\mathbb{I}^i_{(t)} (i = 1, 2, 3, 4)$ indicates the ith pixel of the tth block $\mathbb{I}_{(t)}$ in the raster scanning order, $\mathbf{R}_t \in [0, 255]$ is a random integer generated by $\dot{K}_e$. In this way, the encrypted image $\mathbb{E}$ is generated. Note that, any existing pixel permutation and random number generation methods can be used here to encrypt the original image.

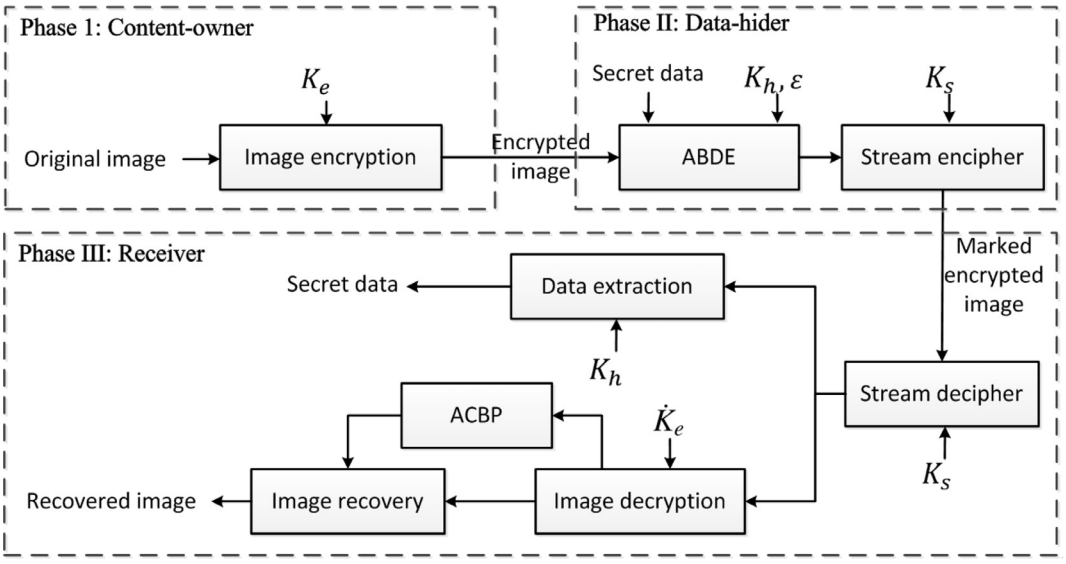

Fig. 5. Framework of the proposed PRDHEI.

5.2. Data embedding

After obtaining the encrypted image, the data hider encrypts the secret data using data hiding key K_h and embeds it into the encrypted image using ABDE and parameter ε. According to the key K_s, a stream encipher is then applied to the data embedded image to further protect the secret data from being illegally extracted. Any secure stream cipher encryption method can be used here to protect the image and secret data. Finally, the obtained image is denoted as the marked encrypted image.

5.3. Data extraction and image recovery

At the receiver side, using different security keys, one can extract the secret data or obtain the recovered image. Next, we describe data extraction and image recovery in detail.

5.3.1. Data extraction

When holding the security keys K_s and K_h, the receiver is able to extract the secret data successfully. First, the stream decipher is applied to the marked encrypted image using K_s, and the obtained image is denoted as $\mathbb{D}$. We then extract the payload which contains the encrypted secret data from $\mathbb{D}$ as described in Section 5.3. Finally, we decrypt the secret data using K_h to obtain the plaintext secret data.

5.3.2. Image recovery

To recover the original image, firstly, we obtain the stream deciphered image $\mathbb{D}$ as in data extraction phase. We then divide $\mathbb{D}$ into a series of 2×2 non-overlapped blocks. According to the parameter ε and embedding strategy type T extracted from SP, we classify the rest non-reference pixels into categories EP and NP by checking their labeling bits. Next, we replace the pixels in SP and NP by extracted payload bits as mentioned in data extraction phase. Thus, all pixels in RP, SP and NP are the same as in the encrypted image $\mathbb{E}$.

To recover the pixels in EP, we first classify them into two groups as $\dot{\mathbf{Z}} = \{\dot{\mathbf{Z}}_1, \dot{\mathbf{Z}}_2, \ldots, \dot{\mathbf{Z}}_u\}$ and $\ddot{\mathbf{Z}} = \{\ddot{\mathbf{Z}}_1, \ddot{\mathbf{Z}}_2, \ldots, \ddot{\mathbf{Z}}_v\}$, where $u + v = n$. $\dot{\mathbf{Z}}$ contains the pixels when $\|\hat{e}\| \leqslant H$ and $\ddot{\mathbf{Z}}$ consists of rest pixels in EP. Here, $H \geq 1$ is a predefined threshold that will be discussed in Section 7, and $\hat{e}$ is calculated by Eq. (8). Next, pixels in $\dot{\mathbf{Z}}$ are recovered by Eq. (17) while pixels in $\ddot{\mathbf{Z}}$ are kept unchanged.

$$\dot{\mathbf{E}}_i = \dot{\mathbf{Z}}_i^r + \hat{e}_i, \quad i = 1, 2, \ldots, u, \tag{17}$$

where $\dot{\mathbf{Z}}_i^r$ is the reference pixel of $\dot{\mathbf{Z}}_i$. Thus, if $\hat{e}_i$ equals to its original value e_i, the current pixel will be perfectly recovered. Otherwise, it will be approximately recovered.

Up to now, all pixels are recovered except for $\ddot{\mathbf{Z}}$. We denote the obtained image as $\hat{\mathbb{E}}$ and decrypt it by

$$\tilde{\mathbb{O}}^i_{(t)} = (\hat{\mathbb{E}}^i_{(t)} - \mathbf{R}_{(t)}) \mod 256 \tag{18}$$

Using $\dot{K}_e$, the receiver then inversely permutates all blocks within the image $\tilde{\mathbb{O}}$ to obtain a recovered image $\check{\mathbb{O}}$.

Next, we recover the pixels in the position of $\ddot{\mathbf{Z}}$. Firstly, we divide $\check{\mathbb{O}}$ into 2×2 non-overlapped blocks and select pixels in RP to form a sub-image $\mathbb{S}$. Using the ACBP proposed in Section 4, we obtain an approximately recovered image from $\mathbb{S}$ and denote the resulted image as $\mathbb{W}$. Next, according to the pixel positions in $\ddot{\mathbf{Z}}$, we pick the pixels in $\mathbb{W}$ with the same positions and replace them to image $\check{\mathbb{O}}$ to obtain the final recovered image $\hat{\mathbb{O}}$.

As analyzed in Section 5.3, pixels in RP, SP and NP will always be perfectly recovered. When parameter $\varepsilon = 1$, all pixels in EP will be classified in $\dot{\mathbf{Z}}$. The obtained difference value $\hat{e}$ will always equals to its original value e. Thus, the recovered image $\hat{\mathbb{O}}$ is exactly the same with the original image $\mathbb{O}$. When parameter $\varepsilon > 1$, pixels in EP will be approximately recovered, which results in a recovered image $\hat{\mathbb{O}}$ that is similar to the original image $\mathbb{O}$.

6. Experiments and comparisons

In this section, we show the experimental results of PRDHEI and comparisons with several existing related works. All test images used in our experiments are selected from the Miscelaneous[1] database as shown in Fig. 6 and with size of 512×512. For all the simulation results of the proposed algorithm, we set the block size to 4×4 in ACBP for image recovery.

6.1. Simulation results

Fig. 7shows the simulation results of the proposed algorithm for *Lena* image when $\varepsilon = 1$. From the results we can observe that, it is able to embed 0.7 bpp of the secret data and the original image can be perfectly recovered.

$$PSNR = 10 \times \log_{10} \frac{255^2}{MSE}, \tag{19}$$

[1] http://decsai.ugr.es/cvg/dbimagenes/g512.php.

Fig. 6. Test images used in this paper with size of 512×512. (a) *Lena*; (b) *Airplane*; (c) *Barbara*; (d) *Peppers*; (e) *Boat* and (f) *Lake*.

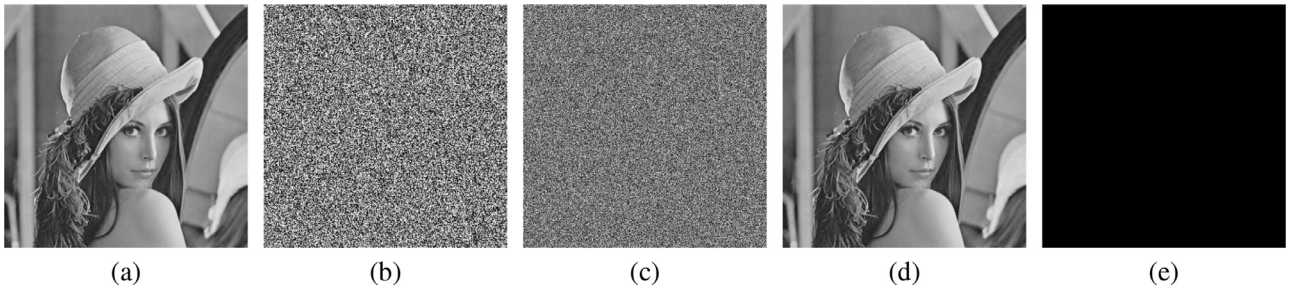

Fig. 7. Simulation results of the proposed algorithm in *Lena* image: (a) the original image; (b) encrypted image; (c) marked encrypted image with $\varepsilon = 1$, $r = 0.7$ bpp; (d) recovered image, PSNR $= +\infty$ dB and (e) difference between (a) and (d).

Table 4
Maximum embedding rate comparisons of different images applied by PRDHEI and several VRAE algorithms when the original image can be perfectly recovered.

r	Zhang	Hong	Zhang	Li	Zhang	Wu	Yin	Yin	Yin	Zhou	Huang[41]		PRDHEI
(bpp)	[15]	[26]	[31]	[28]	[29]	[40]	[33]	[34]	[39]	[16]	DHS	PHS	
Lena	0.004	0.004	0.005	0.010	0.036	0.060	0.12	0.13	0.07	0.15	0.103	0.087	0.70
Airplane	0.001	0.001	0.005	0.013	0.036	0.039	0.19	0.21	0.15	0.19	0.147	0.121	1.27
Barbara	0.001	0.001	0.001	0.005	0.036	0.001	0.09	0.10	0.05	0.15	0.079	0.064	0.35
Peppers	0.004	0.004	0.005	0.006	0.036	0.009	0.11	0.12	0.06	0.19	0.077	0.056	0.54
Boat	0.001	0.004	0.005	0.009	0.036	0.009	0.14	0.15	0.10	0.15	0.061	0.049	0.77
Lake	0.001	0.004	0.005	0.005	0.036	0.007	0.09	0.10	0.05	0.01	0.067	0.047	0.30

where

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(\mathbb{O}(i,j) - \hat{\mathbb{O}}(i,j))^2 \tag{20}$$

$\mathbb{O}$ and $\hat{\mathbb{O}}$ are the original and recovered images, respectively.

6.2. Comparisons

Some applications may hope to lossless recover the original image at the receiver side. For example, the law forensics, military applications, etc. For this purpose, we compare the maximum embedding rate of PRDHEI with several related work under the situation that the secret data can be successfully extracted and the original image can be lossless recovered. Because the proposed PRDHEI is a VRAE method, for a fair comparison, we select several existing VRAE methods to compare the performances with it. The results are listed in Table 4. In our method, secret data can always be successfully extracted under various ε, and the original image can be perfectly recovered when $\varepsilon = 1$. For other VRAE methods, in order to achieve a better performance, we set the block size in [15,26] to 8×8, and [33,34] to 4×4 for demonstrations. In Wu and Sun's method [40], we embed 4 bits of the secret data into each group of the pixels for experiments. In [39], the 2 MSB planes of the original image are encrypted and secret data is embedded into the rest 6 LSB planes. For Huang et al.'s method [41], we set the block size to 2×2 for demonstration. For Zhou et al.'s [16] method, we embed 3 bits of secret data into each image block, different embedding rates can be achieved by setting different sizes to the block.

Table 5
Performance comparison between ACBP and CBP from the aspect of MAE and PSNR.

	MAE			PSNR (dB)		
	CBP	ACBP	Improved	CBP	ACBP	Improved
Lena	3.98	3.77	0.21	32.90	33.56	0.66
Airplane	4.28	4.07	0.21	29.41	29.89	0.48
Babara	9.81	8.58	1.23	23.70	25.10	1.40
Peppers	4.63	4.49	0.14	31.34	31.85	0.51
Boat	5.40	5.09	0.31	29.86	30.52	0.66
Lake	6.95	6.57	0.38	28.36	29.15	0.79
Average	5.83	5.43	0.40	29.26	30.01	0.75

As can be seen from the results, our proposed algorithm significantly improved the maximum embedding rate, especially when the original image is with less texture. For example, for the test image *Airplane*, it can reach the maximum embedding rate as large as 1.27 bpp, which is more than 5 times larger than the existing methods.

7. Discussion

In this section, we discuss the proposed PRDHEI from the following nine aspects: (1) experimental effects of using different embedding strategies; (2) embedding rate comparison; (3) performance comparison between ACBP and CBP; (4) analysis on parameter setting of H for image recovery; (5) rate-distortion analysis; (6) simulation results when $\varepsilon > 1$; (7) PSNR results comparisons under various embedding rates; (8) Difference between PRDHEI

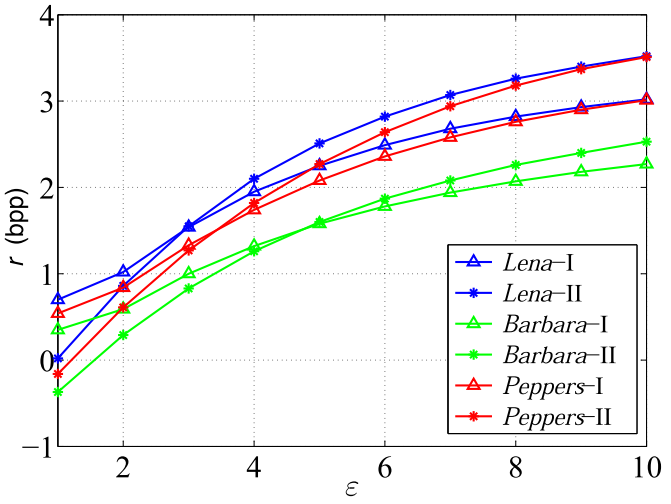

Fig. 8. Effective embedding rate of three images under different embedding strategies and parameter ε.

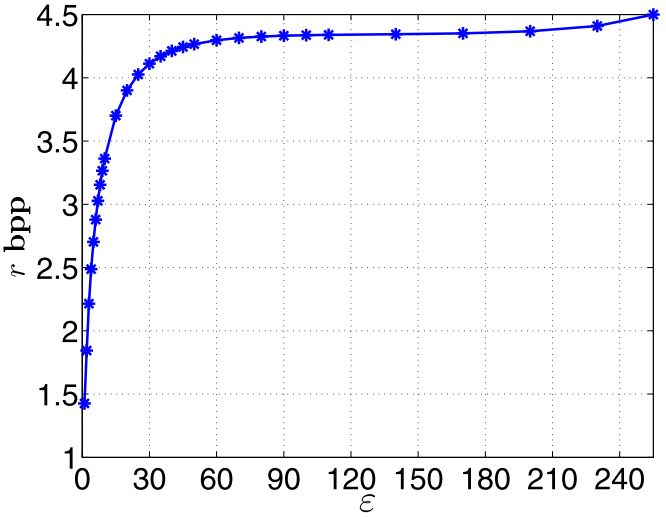

Fig. 9. Average embedding rates of 200 images under various ε.

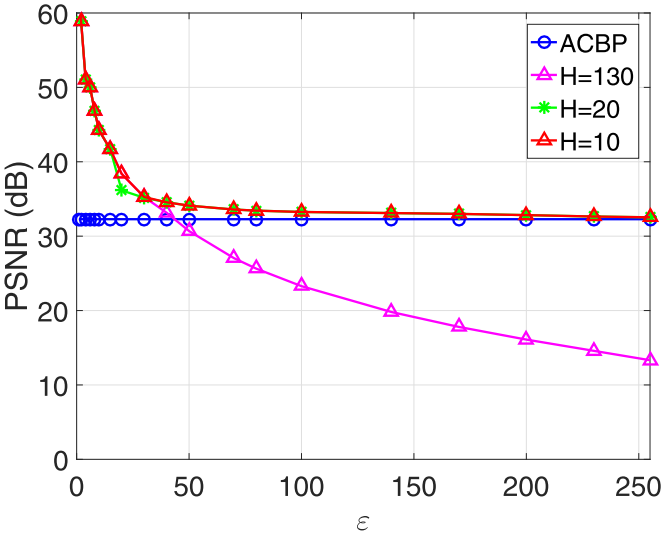

Fig. 10. Average PSNR values of 10000 images recovered by different schemes under various ε. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

and the unified data embedding and scrambling (USE) method and (9) security analysis.

7.1. Embedding strategy comparison

In the proposed algorithm, we provide two strategies for data embedding. Here, we compare the embedding rates of several test images selected from Fig. 6 using different embedding strategies and parameter ε. The results are plotted in Fig. 8, where $\mathbb{A}$-I/II means image $\mathbb{A}$ is embedded by Strategy-I/II. From the results, we can observe that, when ε is small, Strategy-I is able to reach a higher embedding rate; when ε is larger, Strategy-II performs better than Strategy-I. The negative value means it cannot embed any secret data when the corresponding epsilon and embedding strategy are used. In our proposed algorithm, we automatically select the embedding strategy using Eq. (5), so that it can always achieve an optimal embedding rate.

7.2. Embedding rate comparison

Here, we select 200 images from the BowsBase[2] database to show the average embedding rates under various ε. The results are plotted in Fig. 9. As can be seen from this figure, the average embedding rate rises sharply when $\varepsilon \le 30$, while increases slowly when $\varepsilon > 30$. Thus, by setting $\varepsilon = 30$, the proposed algorithm can reach an average embedding rate larger than 4 bpp, which is close to the maximum embedding rate of 4.5 bpp.

7.3. Performance comparison between ACBP and CBP

Here, we use six images in Fig. 6 to compare the performance between ACBP and CBP from the aspect of mean absolute error (MAE) and PSNR. The results are listed in Table 5, where the block size is set to 4×4 in ACBP. The MAE is calculated by

$$MAE = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} (|p_i - p_i'|), \tag{21}$$

where p_i and p_i' are the original and predicted pixel values, respectively. Because we use 1/4 of the pixels in the image to predict the remaining 3/4 of them, $\mathcal{N} = 0.75MN$. The smaller MAE values indicate that the pixels can be better predicted. The PSNR is to compare the visual quality between the original image and predicted image. From the results we can observe that the proposed ACBP has better performance than CBP.

7.4. Analysis on parameter setting of H for image recovery

We use the 10000 images from the BowsBase database to show the average PSNR values of recovered images under various ε and H. The results are plotted in Fig. 10, where the blue line indicates the average PSNR of recovered images using ACBP only. Because only the reference pixels are utilized for image recovery in ACBP, no matter how many secret data is embedded, the PSNR of the recovered image will keep the same value around 32 dB. When $H = 130$, the PSNR of recovered image seriously reduces as the increasing of ε. When $\varepsilon > 50$, the PSNR of recovered image will be less than that of using ACBP only. This is because all pixels in EP belong to $\dot{\mathbf{Z}}$ and will be recovered by considering their labeling bits only. As can be seen from Eq. (17), when $|\hat{e}_i - e|$ is large, the recovered pixel has a significant difference to its original one. This leads to a low PSNR value in the recovered image. Thus, when setting H to a small value, a portion of pixels in EP will be recovered by Eq. (17), while the others with large $|\hat{e}_i - e|$ will be recovered by

[2] http://bows2.ec-lille.fr/.

Table 6
Effective embedding rate r (.bpp) and PSNR (.dB) results of recovered images under various ε.

		ε												
		2	3	4	5	6	7	8	10	15	20	25	30	255
Lena	r	1.02	1.55	2.10	2.51	2.82	3.07	3.26	3.52	3.87	4.04	4.14	4.21	4.50
	PNSR	60.92	54.98	50.61	50.45	48.93	46.75	45.47	42.79	39.45	36.54	36.18	35.74	34.07
Airplane	r	1.64	2.26	2.67	2.94	3.14	3.30	3.42	3.59	3.85	3.99	4.08	4.14	4.50
	PNSR	56.66	54.19	50.00	50.36	48.97	46.94	45.67	44.15	40.32	36.42	35.98	35.53	30.15
Barbara	r	0.59	1.00	1.32	1.60	1.87	2.08	2.26	2.53	2.97	3.25	3.45	3.60	4.50
	PNSR	62.06	57.32	54.09	51.48	49.92	47.71	46.39	43.66	40.17	31.49	30.97	30.36	25.41
Peppers	r	0.84	1.33	1.82	2.27	2.64	2.94	3.18	3.51	3.91	4.07	4.15	4.20	4.50
	PNSR	61.27	56.53	50.85	50.59	48.99	46.77	45.47	42.83	39.42	35.61	35.39	35.12	32.3
Boat	r	1.09	1.58	2.04	2.35	2.59	2.77	2.92	3.16	3.56	3.79	3.94	4.04	4.50
	PNSR	60.83	54.93	50.65	50.65	49.14	47.04	45.69	42.98	39.91	34.72	34.17	33.65	30.92
Lake	r	0.53	0.93	1.26	1.58	1.91	2.18	2.42	2.81	3.38	3.67	3.85	3.96	4.50
	PNSR	62.31	57.47	54.03	51.43	49.81	47.56	46.21	43.45	39.91	33.09	32.63	32.19	29.58

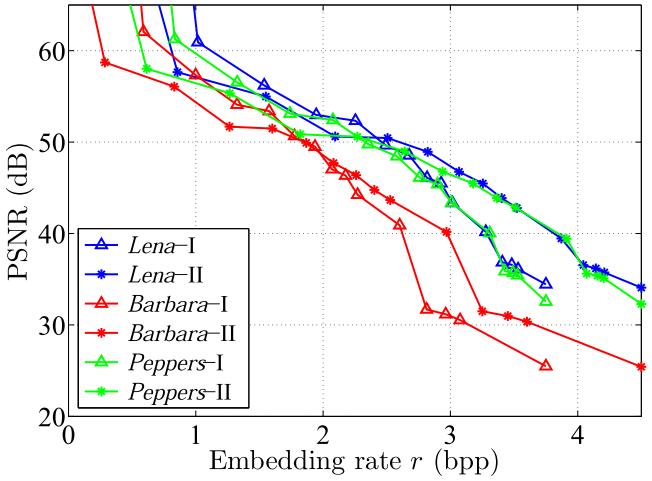

Fig. 11. PSNR comparison of three test images under various embedding rates and using different embedding strategies.

ACBP. Therefore, a higher visual quality of the recovered image can be obtained. As can be seen, when H is small, the recovered images using the proposed algorithm will always achieve better visual quality than that of using ACBP only. The average PSNR value is larger than 30 dB. For simplicity, we set $H = 10$ for experiments in the rest of this paper.

7.5. Rate-distortion comparison

Here, we compare the distortion of the recovered images applied by different embedding strategies. The results are plotted in Fig. 11, where $\mathbb{A}$-I/II means image $\mathbb{A}$ is embedded by Strategy-I/II. From the results we can observe that, using Strategy-I and Strategy-II will obtain a maximum embedding rate close to 3.75 bpp and 4.5 bpp, respectively. Given an embedding rate with small value, embedding Strategy-I will obtain a higher PSNR result; otherwise, Strategy-II will perform a better visual quality in the recovered image.

7.6. Simulation results when parameter $\varepsilon > 1$

Table 6shows the effective embedding rate r and PSNR values of recovered images under different values of ε. As can be seen from this table, the embedding rate r increases as the increasing of parameter ε. When ε is small, images with less texture are able to embed more secret data than that with more texture. On the other hand, the recovered image quality is dependent on the settings of parameter ε. When $\varepsilon = 2$, the PSNR of recovered image is as high as 60 dB. When $\varepsilon = 30$, all images can achieve an embedding rate r larger than 3.6 bpp. The final recovered images have satisfied visual quality with PSNR values all larger than 30 dB. When $\varepsilon = 255$, it achieves the maximum embedding rate of 4.5 bpp, and the recovered images are all with PSNR larger than 25 dB. Thus, the proposed algorithm is able to embed a large number of secret data while keeping high image quality in the final recovered images.

7.7. PSNR Results comparisons under various embedding rates

For some applications, for example, the Cloud storage and other image management systems, they may allow the original image to be slightly modified while embedding as large secret data as possible. In this scenario, we compare the PSNR results of recovered images under various embedding rates. Six test images in Fig. 6 are utilized for demonstration and the results are plotted in Fig. 12. In our proposed algorithm, the secret data can always be lossless extracted under various embedding rates. For other VRAE methods whose secret data cannot be perfectly extracted, the comparison is under the case that the secret data has high probability to be lossless extracted. Thus, for those methods, the pure embedding rate r becomes to $r(1 - H(\rho))$, where $H(\rho)$ is the binary entropy function with the error rate ρ. From the results we can observe that, our proposed PRDHEI has significantly larger PSNR values than the existing VRBE methods under various embedding rates. Although be embedded with as large as 3 bpp of secret data, it is able to obtain a recovered image with PSNR larger than 40 dB. However, other VRAE methods have limited embedding rates that are almost less than 1 bpp and with low PSNR values of the recovered images. These show the superiority of our proposed algorithm.

7.8. Difference between PRDHEI and the unified data embedding and scrambling (USE) method

PRDHEI is an encrypted domain based data embedding method. It encrypts the original image using block based permutation and modular operation, and exploits the spatial redundancy within small encrypted image blocks for data embedding. USE [8] uses CBP to predict pixels in an original image and embeds secret data by degrading the image quality. Here, we summarize the difference between PRDHEI and USE from six following aspects:

(1) The image for embedding secret data. PRDHEI embeds the secret data on the encrypted image while USE performs data embedding in a normal image (i.e. the original image).
(2) The way of performing image encryption and data embedding. PRDHEI performs image encryption and data embedding by different users, separately and independently. In PRDHEI, the original image is first encrypted by the content

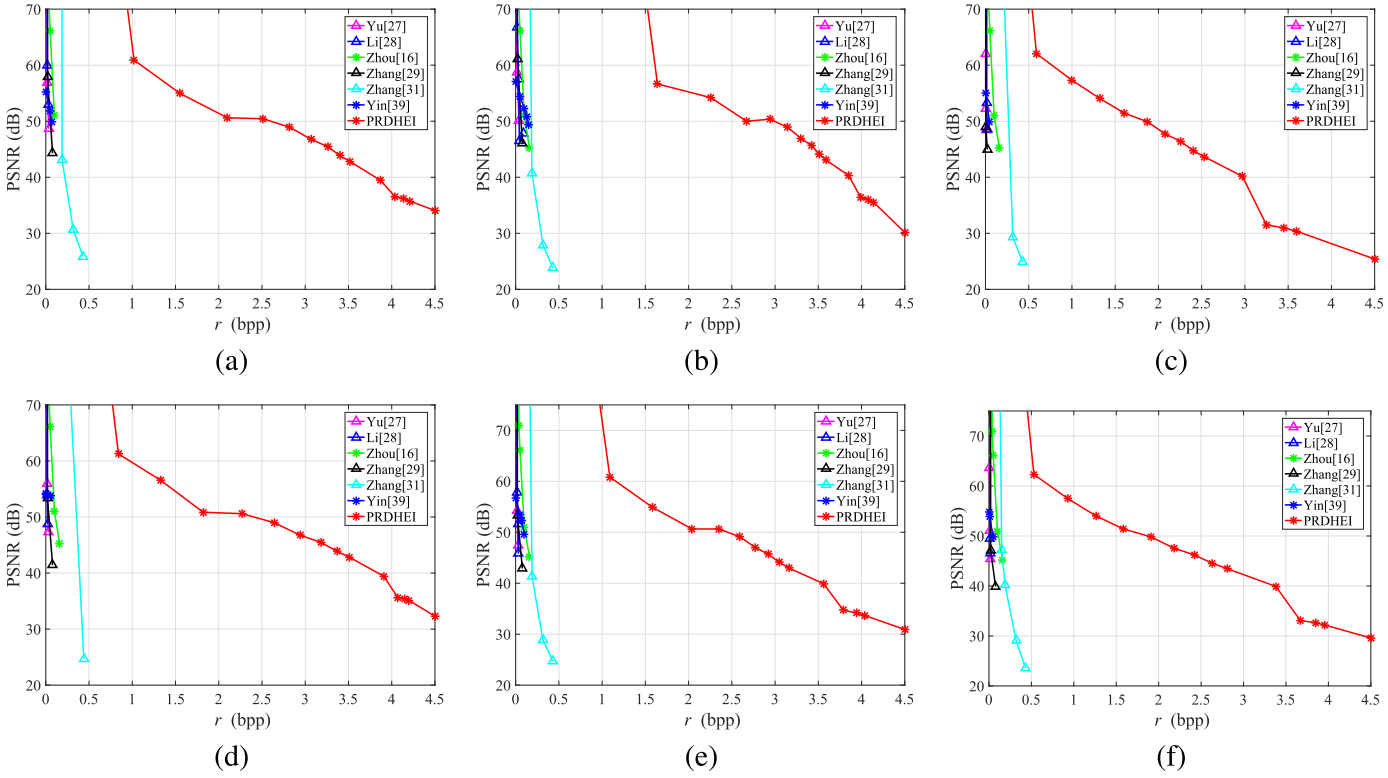

Fig. 12. Comparisons of PSNR results of the recovered image after data extraction under various embedding rates. (a) *Lena*; (b) *Airplane*; (c) *Barbara*; (d) *Peppers*; (e) *Boat* and (f) *Lake*.

owner, the data hider then embeds secret data into the encrypted image without knowing the original image content. However, USE embeds secret data into the original image for image scrambling. It performs data embedding and image scrambling in a single step.

(3) The calculation method of the prediction-error value. PRDHEI calculates the prediction-error value according to the differences between the reference pixel and other three pixels within a 2×2 encrypted image block. However, USE obtains the prediction-error value after using CBP to predict a pixel via its four/eight surrounding pixels in a normal image.

(4) The strategy of data embedding. PRDHEI uses two strategies (Strategy-I and Strategy-II) for data embedding while USE uses only one embedding strategy (similar to Strategy-II in PRDHEI). From the experiment results in Fig. 8, Strategy-I significantly improves the effective embedding rate when ε is small.

(5) The prediction method. PRDHEI uses ACBP only for image recovery. However, USE uses CBP to predict pixels for both data embedding and image recovery. Our ACBP is an improved version of CBP. Experiment results in Table 5 show that our ACBP achieves a better performance than CBP.

(6) The data embedding rounds. PRDHEI performs only a single round of data embedding while USE performs multiple rounds for data embedding by exploring the spatial correlations of the unchanged pixels in the original image.

7.9. Security analysis

For the encrypted image based data hiding, both the original image and the secret data need to be protected. To protect the secret data, we first encrypt it using the data hiding key K_h before data embedding, and then apply a stream cipher. Note that users can choose any secure data encryption algorithm to ensure the secret data security. Therefore, we mainly focus on analyzing the security of the image encryption method used in PRDHEI.

7.9.1. Ciphertext-only attack

Ciphertext-only attack is an attack model that the attackers attempt to reveal the information of the plaintext, or even the security key, from the ciphertext only. To protect the original image, firstly, we divide it into k blocks and apply a block permutate to change the pixel positions. Thus, totally $k!$ possible permutations. The pixel modulation operation is then adopted to change the pixel values using random numbers. Thus, for an image with 8-bit pixel depth, the possibility of successful decrypting it without the correct encryption key is as small as $1/(k!256^k)$.

Next, we experimentally analyze the encryption effect of the proposed algorithm. Fig. 13 provides the histogram result of the original and encrypted *Lena* image. As can be seen, after image encryption, the pixel values are uniform distributed, so that it will not reveal the original image information by analyzing its histogram. In our algorithm, each image block is treated as a single unit for permutation and modular operation, the encryption effect also depends on the correlation between image blocks. Thus, the encryption algorithm should break the block correlations. The mathematical data correlation is defined by

$$Corr = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}, \tag{22}$$

where X and Y are two data sequence, μ is the mean value and σ is the standard deviation. When X and Y are high correlated, the correlation value is close to 1; otherwise, it is close to 0. In this experiments, we randomly select 3000 image blocks and their corresponding neighboring blocks along with the horizontal, vertical and diagonal directions from the original and encrypted images separately. Here, X and Y are obtained by calculating the mean value of each selected block and its neighboring block. The adjacent block correlation results are listed in Table 7. As can be seen,

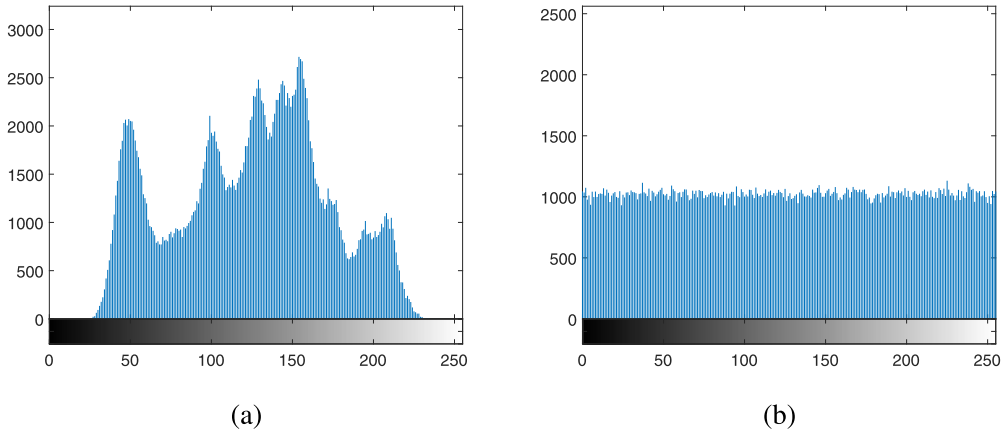

Fig. 13. Histogram of *Lena* image. (a) the original image and (b) encrypted image.

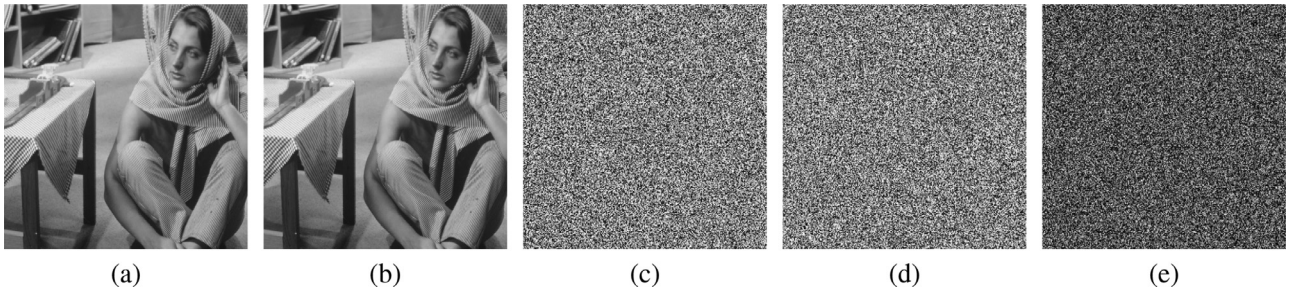

Fig. 14. Simulation results of differential analysis on *Barbara* image: (a) the original image $\mathbb{O}_1$; (b) the original image $\mathbb{O}_2$, where $\mathbb{O}_1$ and $\mathbb{O}_2$ are with only one bit difference; (c) encrypted results of $\mathbb{O}_1$, (d) encrypted results of $\mathbb{O}_2$ and (e) the difference between (c) and (d).

Table 7
Image block correlations of *Lena* image and its encrypted image.

Images	Horizontal	Vertical	Diagonal
Original image	0.9867	0.9731	0.9601
Encrypted image	0.0240	−0.0190	−0.0039

the original image has high block correlations. However, after image encryption, the spatial correlations between image blocks are disturbed. Thus, one has difficulty to obtain any information from the encrypted image by analyzing the correlations between image blocks.

7.9.2. Known/chosen-plaintext attack

Known/Chosen-plaintext attack is a cryptanalysis model that the attackers have the plaintexts and their corresponding ciphertexts and try to reveal the information of the secret key. Here, we use the differential analysis to show the robustness of the proposed encryption algorithm against the known/chosen-plaintext attack. The results are shown in Fig. 14. From the results, we can observe that even with one pixel difference in the original images, the obtained encrypted images are totally different. Therefore, it is difficult for attackers to obtain any information by analyzing the correlations between the ciphertexts and their plaintexts. This verifies that the proposed algorithm is able to withstand the known/chosen-plaintext attack.

8. Conclusion

In this paper, we proposed a parametric reversible data hiding method in encrypted images. It uses the adaptive bit-level data embedding method to embed the secret data and combines the adaptive checkerboard based prediction and labeling bits information to recover the original image. In the proposed algorithm, data extraction and image recovery are totally independent and secret data can always be completely extracted under various embedding rates. The original image can also be perfectly recovered when parameter $\varepsilon = 1$. Experiments have shown that the embedding rate is much larger than the state-of-the-art methods. In addition, when $\varepsilon > 1$, it significantly increased the embedding rate. Simulations and comparisons have shown that our proposed algorithm has a maximum embedding rate of 4.5 bpp. Meanwhile, the recovered images have significantly higher quality than existing methods under various embedding rates.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valued comments which helped to improve this paper. This work was supported in part by the Macau Science and Technology Development Fund under Grant FDCT/016/2015/A1 and by the Research Committee at University of Macau under Grant MYRG2016-00123-FST.

References

[1] Y.Q. Shi, X. Li, X. Zhang, H. Wu, M. B., Reversible data hiding: advances in the past two decades, IEEE Access 4 (99) (2016) 3210–3237.
[2] X. Liao, Z. Qin, L. Ding, Data embedding in digital images using critical functions, Signal Process. Image Commun. 58 (Supplement C) (2017) 146–156.
[3] X. Liao, J. Yin, S. Guo, X. Li, A.K. Sangaiah, Medical JPEG image steganography based on preserving inter-block dependencies, Comput. Electr. Eng. (2017), doi:10.1016/j.compeleceng.2017.08.020.
[4] Z. Ni, Y.-Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 16 (3) (2006) 354–362.
[5] Z. Hua, Y. Zhou, C.-M. Pun, C.L.P. Chen, 2D Sine logistic modulation map for image encryption, Inf. Sci. (Ny) 297 (0) (2015) 80–94.
[6] Z. Hua, Y. Zhou, Image encryption using 2D logistic-adjusted-Sine map, Inf. Sci. (Ny) 339 (2016) 237–253.
[7] J. Tian, Reversible data embedding using a difference expansion, IEEE Trans. Circuits Syst. Video Technol. 13 (8) (2003) 890–896.

[8] R.M. Rad, K. Wong, J.-M. Guo, A unified data embedding and scrambling method, IEEE Trans. Image Process. 23 (4) (2014) 1463–1475.
[9] X. Li, B. Yang, T. Zeng, Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection, IEEE Trans. Image Process. 20 (12) (2011) 3524–3533.
[10] X. Liao, K. Li, J. Yin, Separable data hiding in encrypted image based on compressive sensing and discrete fourier transform, Multimed. Tools Appl. 76 (20) (2017) 20739–20753.
[11] M. Li, D. Xiao, Y. Zhang, H. Nan, Reversible data hiding in encrypted images using cross division and additive homomorphism, Signal Process. Image Commun. 39 (Part A) (2015) 234–248.
[12] Z. Qian, H. Zhou, X. Zhang, W. Zhang, Separable reversible data hiding in encrypted JPEG bitstreams, IEEE Trans. Dependable Secur. Comput. PP (99) (2016). 1–1
[13] M. Li, D. Xiao, A. Kulsoom, Y. Zhang, Improved reversible data hiding for encrypted images using full embedding strategy, Electron. Lett. 51 (9) (2015) 690–691.
[14] D. Bouslimi, G. Coatrieux, M. Cozic, C. Roux, A joint encryption/watermarking system for verifying the reliability of medical images, IEEE Trans. Inf. Technol. Biomed. 16 (5) (2012) 891–899.
[15] X. Zhang, Reversible data hiding in encrypted image, IEEE Signal Process. Lett. 18 (4) (2011) 255–258.
[16] J. Zhou, W. Sun, L. Dong, X. Liu, O.C. Au, Y.Y. Tang, Secure reversible image data hiding over encrypted domain via key modulation, IEEE Trans. Circuits Syst. Video Technol. 26 (3) (2016) 441–452.
[17] G. Coatrieux, C.L. Guillou, J.M. Cauvin, C. Roux, Reversible watermarking for knowledge digest embedding and reliability control in medical images, IEEE Trans. Inf. Technol. Biomed. 13 (2) (2009) 158–165.
[18] K. Ma, W. Zhang, X. Zhao, N. Yu, F. Li, Reversible data hiding in encrypted images by reserving room before encryption, IEEE Trans. Inf. Forensics Secur. 8 (3) (2013) 553–562.
[19] T. Mathew, M. Wilscy, Reversible data hiding in encrypted images by active block exchange and room reservation, in: Proceedings of the International Conference on Contemporary Computing and Informatics (IC3I), pp. 839–844.
[20] W. Zhang, K. Ma, N. Yu, Reversibility improved data hiding in encrypted images, Signal Process. 94 (0) (2014) 118–127.
[21] S. Yi, Y. Zhou, An improved reversible data hiding in encrypted images, in: Proceedings of the IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP), 2015, pp. 225–229.
[22] D. Xu, R. Wang, Separable and error-free reversible data hiding in encrypted images, Signal Process. 123 (2016) 9–21.
[23] T.-S. Nguyen, C.-C. Chang, W.-C. Chang, High capacity reversible data hiding scheme for encrypted images, Signal Process. Image Commun. 44 (2016) 84–91.
[24] X. Cao, L. Du, X. Wei, D. Meng, X. Guo, High capacity reversible data hiding in encrypted images by patch-level sparse representation, IEEE Trans. Cybern. 46 (5) (2016) 1132–1143.
[25] Z. Qian, X. Zhang, Reversible data hiding in encrypted images with distributed source encoding, IEEE Trans. Circuits Syst. Video Technol. 26 (4) (2016) 636–646.
[26] W. Hong, T.-S. Chen, H.-Y. Wu, An improved reversible data hiding in encrypted images using side match, IEEE Signal Process. Lett. 19 (4) (2012) 199–202.
[27] J. Yu, G. Zhu, X. Li, J. Yang, An improved algorithm for reversible data hiding in encrypted image, Digit. Forensics Watermarking 7809 (2013) 384–394.
[28] M. Li, D. Xiao, Z. Peng, H. Nan, A modified reversible data hiding in encrypted images using random diffusion and accurate prediction, ETRI J. 36 (2) (2014) 325–328.
[29] X. Zhang, Separable reversible data hiding in encrypted image, IEEE Trans. Inf. Forensics Secur. 7 (2) (2012) 826–832.
[30] X. Zhang, Z. Qian, G. Feng, Y. Ren, Efficient reversible data hiding in encrypted images, J. Vis. Commun. Image Represent. 25 (2) (2014) 322–328.
[31] X. Zhang, C. Qin, G. Sun, Reversible data hiding in encrypted images using pseudorandom sequence modulation, Digit. Forensics Watermaking 7809 (2013) 358–367.
[32] S. Zheng, D. Li, D. Hu, D. Ye, L. Wang, J. Wang, Lossless data hiding algorithm for encrypted images with high capacity, Multimed. Tools Appl. (2015) 1–14.
[33] Z. Yin, B. Luo, W. Hong, Separable and error-free reversible data hiding in encrypted image with high payload, Sci. World J. 2014 (2014) 8.
[34] Z. Yin, H. Wang, H. Zhao, B. Luo, X. Zhang, Complete separable reversible data hiding in encrypted image, in: Proceedings of the Cloud Computing and Security: First International Conference, ICCCS 2015, 2015, pp. 101–110.
[35] Y.-C. Chen, C.-W. Shiu, G. Horng, Encrypted signal-based reversible data hiding with public key cryptosystem, J. Vis. Commun. Image Represent. 25 (5) (2014) 1164–1170.
[36] C.-W. Shiu, Y.-C. Chen, W. Hong, Encrypted image-based reversible data hiding with public key cryptography from difference expansion, Signal Process. Image Commun. 39, Part A (2015) 226–233.
[37] X. Zhang, J. Wang, Z. Wang, H. Cheng, Lossless and reversible data hiding in encrypted images with public key cryptography, IEEE Trans. Circuits Syst. Video Technol. 26 (9) (2016) 1622–1631.
[38] X. Liao, C. Shu, Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels, J. Vis. Commun. Image Represent. 28 (0) (2015) 21–27.
[39] Z. Yin, A. Abel, J. Tang, X. Zhang, B. Luo, Reversible data hiding in encrypted images based on multi-level encryption and block histogram modification, Multimed. Tools Appl. 76 (3) (2017) 3899–3920.
[40] X. Wu, W. Sun, High-capacity reversible data hiding in encrypted images by prediction error, Signal Process. 104 (2014) 387–400.
[41] F. Huang, J. Huang, Y.Q. Shi, New framework for reversible data hiding in encrypted domain, IEEE Trans. Inf. Forensics Secur. 11 (12) (2016) 2777–2789.