Multimed Tools Appl (2018) 77:20917–20935
https://doi.org/10.1007/s11042-017-5498-8

Reversible data hiding in encrypted images with high capacity by bitplane operations and adaptive embedding

Fuqiang Di[1] · Fangjun Huang[2,3] · Minqing Zhang[1] · Jia Liu[1] · Xiaoyuan Yang[1]

Received: 28 March 2017 / Revised: 29 November 2017 / Accepted: 5 December 2017 /
Published online: 21 December 2017

Abstract Reversible data hiding in encrypted images (RDHEI) is a technique that makes contributions to cloud data management in privacy preservation and data security. A novel framework of RDHEI with high embedding capacity based on bitplane operations and adaptive embedding is proposed. Three parties constitute the proposed system: the content owner, the data hider and the receiver. First, the content owner encrypts the original image for privacy protection. A data hider partitions the encrypted image into two sub images by bitplane-level operations and embeds additional data with an adaptive embedding strategy. With the encrypted image containing the embedded data, the receiver can extract the embedded data without any error and losslessly recover the original image according to specific requirements. The proposed framework can not only work for many different specific image encryption methods but also accomplish hundreds of reversible data hiding (RDH) algorithms directly in the encrypted domain. Extensive experiments demonstrate that the proposed framework can significantly increase the embedding capacity of some existing RDHEI frameworks, although it may reduce the *PSNR* value.

Keywords Reversible data hiding · Encrypted image · Bitplane operations · Adaptive embedding

1 Introduction

Reversible data hiding (RDH) [7, 20, 25] is a multimedia security technique to embed additional data (e.g., authentication information) into a carrier image, and recover the original

✉ Minqing Zhang
1054165690@qq.com

1 Department of Electronic technology, Engineering University of Chinese People's Armed Police, Xi'an 710086, China

2 School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China

3 State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

image after data extraction. This emerging technique has been extensively studied and is widely used in many fields such as image authentication [4, 27] and activity recognition [2, 15–17, 19]. However, in many big data scenarios, the image is first encrypted before being uploaded to the server for secure communication. Recently, reversible data hiding in encrypted images (RDHEI) has attracted much attention and has many important applications in medical, military and other fields [18, 21, 32]. For example, medical images of patients are usually encrypted first to protect the privacy of patients and then uploaded to the hospital servers. On the one hand, the server manager may hope to embed some additional information, such as integrity authentication; on the other hand, the original medical images must be recovered without error.

The classic RDH algorithms for bmp images include three major approaches: difference expansion [26], histogram shifting [3] and lossless compression [14]. The lossless-compression-based methods release some place by losslessly compressing the cover image and generally have low embedding capacity. The difference-expansion-based methods usually perform a higher embedding capacity, while keeping the image distortion low. Compared with other methods, the histogram-shifting-based methods have the better capacity-distortion curve and have been extensively investigated [10–12].

The classic RDH algorithms for un-encrypted images generally cannot be applied to encrypted images directly. Zhang proposed the first RDHEI algorithm with a new pixel flipping strategy in [29]. The method embedded additional data into an image that is encrypted by stream cipher, and the original image can be recovered via the correlation of pixels. Some improved versions of Zhang's method were proposed in [1, 5, 13, 24, 34]. Since the embedded data can only be extracted after image decryption, i.e., a receiver having data-hiding key but no content owner key cannot extract the embedded information, the type of algorithms in [1, 5, 13, 24, 29, 34] are referred to as non-separable methods.

To overcome this problem, Zhang proposed a separable reversible data hiding scheme in [30]. In this new scheme, the legal receiver can extract the additional data with the use of a data hiding key directly without the image decryption. Qian et al. embedded a secret message into an encrypted image using a histogram modification and *n*-nary data hiding scheme [22]. This method improved the embedding capacity and image quality, but the leakage of image histogram reduced the safety of the image. Zhang et al. compressed a part of encrypted data in the cipher-text image using a Low Density Parity Check Code (LDPC) and inserted the compressed data into part of encrypted data [31]. Zheng et al. applied a chaotic sequence to encrypt the image and compressed the least significant bits of pixels using the Hamming distance [33]. Instead of stream encryption, Qian et al. proposed an alternative algorithm suitable for block encrypted images and improved the image security and quality [23].

However, all the previously mentioned RDHEI methods are specifically designed. To apply those numerous RDH methods designed in the plain domain to the encrypted domain, some new image encryption strategies, which can preserve the correlations of the neighboring pixels, have been proposed. Huang et al. proposed a specific image encryption method for data hiding in encrypted images, which includes image block partition, stream encryption, and block permutation [6]. Yin et al. partitioned the cover image into non-overlapping blocks and applied multi-granularity encryption to obtain an encrypted image [28]. In this paper, we propose a new RDHEI framework based on bitplane operations and adaptive embedding. Instead of considering each image pixel as an integer, we transform each pixel into two components based on the bitplane parameter and then adopt adaptive embedding in the proposed paper.

In brief, there are two parts in the framework of the most existing RDHEI methods: image encryption and reversible data hiding. Figure 1a gives the overview of the most existing RDHEI methods. The mainstream RDH methods used in RDHEI methods include difference histogram shifting (DHS) [8] and prediction-error histogram shifting (PEHS) [9]. The overview of the proposed RDHEI method is shown in Fig. 1b. It can be seen that the overall connection between this paper and the existing works is the added “bitplane operations” process. However, the proposed method is only applicable for some specific encryption methods, such as the method in [6] or [28]. The only requirement is that the correlation between the neighboring pixels in partial regions (such as in a block) should be well preserved after encryption. Experimental results show that the proposed method can effectively improve the embedding capacity of these kinds of RDHEI frameworks, although reduce the PSNR value.

The rest of this paper is organized as follows. In Section 2, the detailed procedures of the proposed RDHEI scheme are described. Section 3 elaborates the experimental results and the performance comparison. The conclusions of our work are given in Section 4.

2 The proposed framework

The proposed separable RDHEI framework is illustrated in Fig. 2 and includes three parties: the content owner, the data hider and the receiver. First, the content owner encrypts the original image I to produce an encrypted image E. Then, the data hider, without knowing the actual contents of I, transforms E into two new images E_h and E_l, which respectively consist of the high bitplanes and the low bitplanes. Additional data M is evenly divided into M_h and M_l. Data M_h and M_l are embedded into E_h and E_l respectively, and marked-encrypted sub images E_h^* and E_l^* are obtained. After that, the marked-encrypted image E^* is generated by bitplane combination and sent to the receiver. At the receiver side, there are three options for legal receivers. In Case I, the receiver transforms E^* into sub image E_h^* and sub image E_l^*, extracts the data in both sub images, and recovers the original image. In Case II, only the data is extracted. In Case III, an approximate image I^* is obtained by direct decryption. Note that separable RDHEI means that the data extraction process can be separately carried out before image decryption. From Case II in Fig. 2, the embedded data can be extracted without image decryption in our scheme, and thus our proposed framework is “separable”. As is seen in Fig. 1b, the proposed method can adopt different encryption methods and RDH methods. To simplify the discussion, we select the encryption method in [6] and the DHS method in the discussion below.

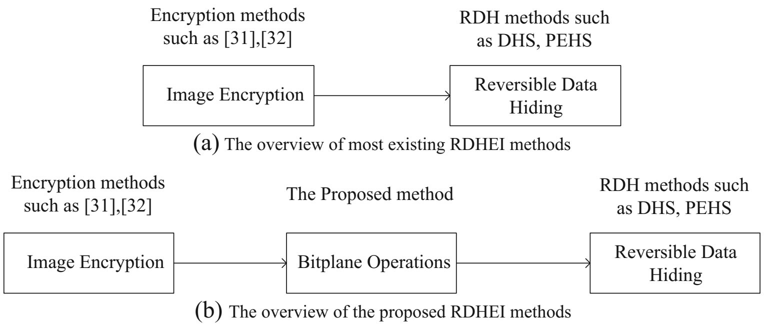

Encryption methods such as [31],[32]
RDH methods such as DHS, PEHS
Image Encryption
Reversible Data Hiding
(a) The overview of most existing RDHEI methods
Encryption methods such as [31],[32]
The Proposed method
RDH methods such as DHS, PEHS
Image Encryption
Bitplane Operations
Reversible Data Hiding
(b) The overview of the proposed RDHEI methods

Fig. 1 The overall contribution of this paper

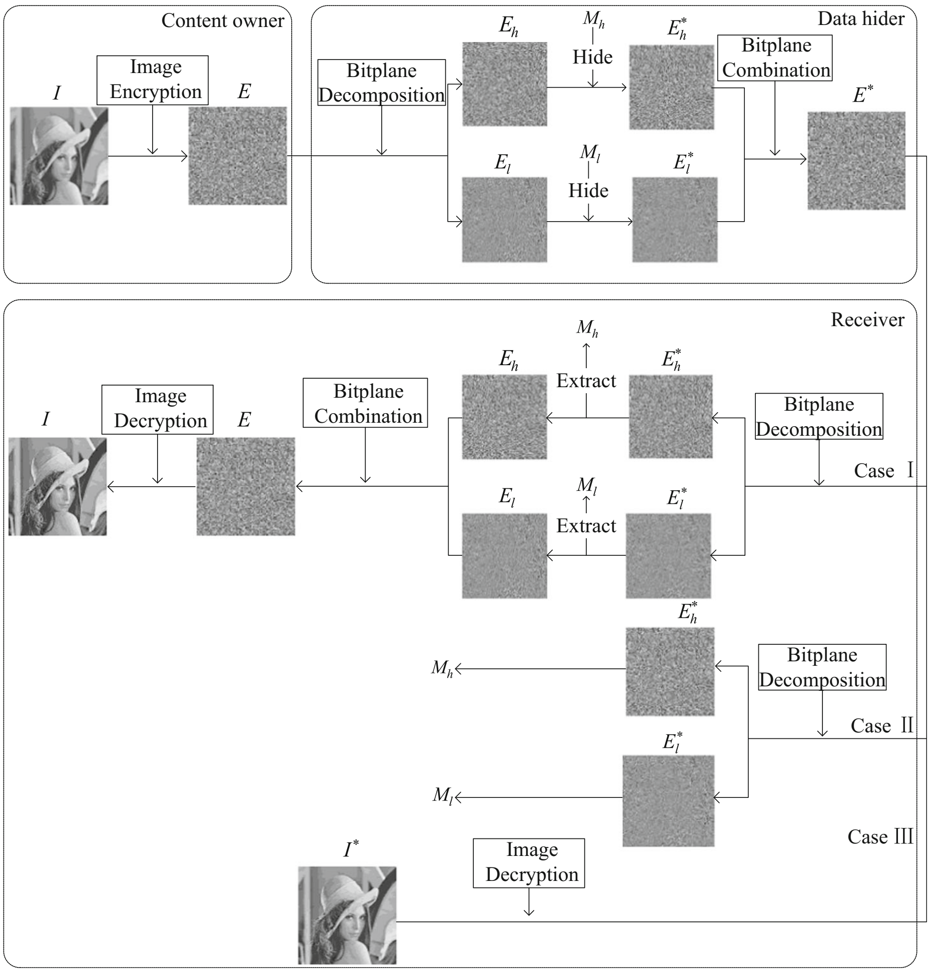


Fig. 2 Sketch of the proposed RDHEI scheme

2.1 Image encryption

Here, we adopt Huang's [6] image encryption algorithm. We assume the original image I is an 8-bit gray-scale image. The procedures of image encryption include the following steps.

Step 1: Divide the original image I into T non-overlapping blocks $\{B_1, B_2, \ldots, B_T\}$. The blocks are with the size of $m \times n$. Let $I(i,j)(1 \le i \le T, 1 \le j \le m \times n)$ denotes one of the pixels in block B_i, where i denotes the index of the block, and j denotes the index of the pixel in block B_i.

Step 2: Generate an encryption key stream E. For each block B_i, run the stream cipher E to generate a key stream $R_i (1 \le i \le T)$ with a length of 8 bits. For each pixel $I(i,j)$ in the block B_i, perform the bitwise exclusive-or (XOR) operation between $I(i,j)$ and R_i as follows,

$$E(i,j) = I(i,j) \wedge R_i \tag{1}$$

where $E(i,j)$ represents the encrypted pixel and $\wedge$ represents the bitwise XOR operation.

Step 3: Encrypt all blocks by repeating Step 2. Note that the pixels in the same block are encrypted with the same key stream, and different key streams are used in different blocks.

Step 4: Permute all the encrypted blocks with permutation key and the encrypted image E is generated. In this step, only the order of the blocks is disrupted, and the order of the pixels within each block is still preserved.

2.2 Data hiding in encrypted image

Let $E(i,j)(1 \le i \le T, 1 \le j \le m \times n)$ be a pixel in the encrypted image E, where i denotes the block index, and j denotes the pixel index in block B_i. Decompose $E(i,j)$ into 8 bits using

$$E(i,j,k) = \lfloor E(i,j)/2^k \rfloor \bmod 2, \qquad k = 0, 1, 2, \ldots 7 \tag{2}$$

where $\lfloor \cdot \rfloor$ represents the floor function. Instead of considering $E(i,j)$ of 8 bits as an integer within the interval of [0,255], the data hider transforms $E(i,j)$ into $E_h(i,j)$ and $E_l(i,j)$ as in Eq. (3) and Eq. (4). How to select the value of e $(1 \le e \le 7)$ will be discussed in Section 3.

$$E_h(i,j) = \sum_{k=0}^{e-1} E(i,j,k+8-e) \times 2^k \tag{3}$$

$$E_l(i,j) = \sum_{k=0}^{7-e} E(i,j,k) \times 2^k \tag{4}$$

After processing all pixels in image E by Eqs. (3) and (4), the data hider obtains the sub images E_h and E_l, which consist of e high bitplanes and $8-e$ low bitplanes, respectively. How to conduct the bitplane operations is depicted in Fig. 3.

Next, the sub images are further divided into blocks of the same size as that in image encryption phase. Note that, if $e = 1$, the data will only be embedded in E_l because the sub-image E_h has only one bitplane, which is not suitable to embed data. If $e = 7$, the data will only be embedded in E_h because the sub-image E_l has only one bitplane, which is not suitable to embed data. Since the data hiding procedure in E_l is the same as that in E_h, how to embed information into E_h will be illustrated here for brevity. The data hiding procedure in sub images E_h is as follows.

Step 1: Generate the difference histogram. The difference value in each block is computed as

$$D_h(i,j) = E_h(i,j) - E_h(i,1) \qquad i \in [1,T], j \in [1, m \times n], j \ne 1 \tag{5}$$

Then, the difference values in all blocks make up a difference vector D with a length of $(m \times n - 1)T$ can be obtained. We use H to denote the difference histogram so that H_k

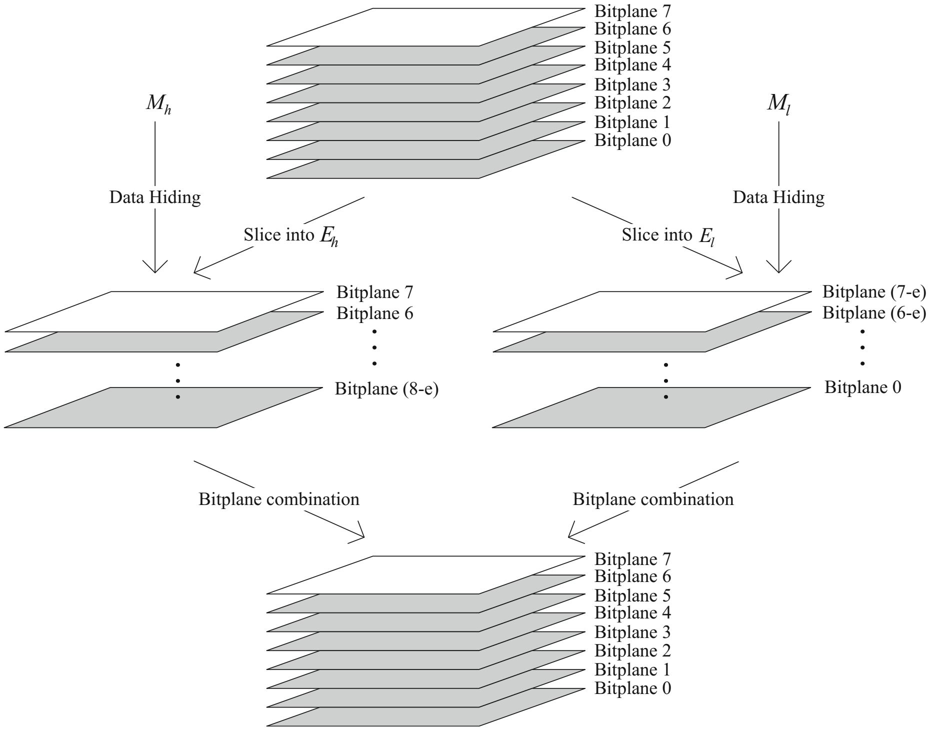


Fig. 3 Illustration of bitplane operations

represents the number of elements in the vector D that are equal to k. Note that $E_h(i,j) \in [0, 2^e - 1]$, $D_h(i,j) \in [1-2^e, 2^e-1]$ and $k \in [1-2^e, 2^e-1]$. Suppose $D_h(i,j)$ consists of $2^{e+1}-1$ different difference values. Thus, there are $2^{e+1}-1$ bins in H, from which the two peak bins (the highest two bins) are chosen. The left peak and right peak bins are represented by L and R respectively. Note that H_L and H_R represent the number of pixels associated with the peak bins L and R, respectively.

Step 2: Select peak bins for embedding. Similar to [6], to avoid the saturation (i.e., the underflow or overflow), the underflow pixels (pixels with value 0) and the overflow pixels (pixels with value $2^e - 1$) have to be preprocessed by modifying one gray-scale unit, and noted respectively in a left location map F_L for avoiding underflow and a right location map F_R for avoiding overflow. L_L and L_R denote respectively the length of F_L and the length of F_R. To embed the data in Step 3, L_L is set to the number of pixels with value 0 or 1 in $E_h(i,j)$, and L_R is set to the number of pixels with value $2^e - 1$ or $2^e - 2$ in $E_h(i,j)$. Now, the embedding location can be determined as follows.

(1) If $H_L - L_L > 0$, the left peak bin L will be used for data embedding; otherwise, it will not.
(2) If $H_R - L_R > 0$, the right peak bin R will be used for data embedding; otherwise, it will not.

Step 3: Embed data adaptively.

(1) When the left peak bin L is used for data embedding: visit all the pixels in E_h sequentially and append a bit "0" to F_L when $E_h(i,j) = 1$. If $E_h(i,j) = 0$, append a bit "1" to F_L and make $E_h(i,j)$ from 0 to 1 simultaneously. The reversible data hiding is conducted as follows.

$$E_h^*(i,j) = \begin{cases} E_h(i,j)-1 & \text{if } D_h(i,j) < L \\ E_h(i,j)-b & \text{if } D_h(i,j) = L \\ E_h(i,j) & \text{if } D_h(i,j) > L \end{cases} \tag{6}$$

where $b \in \{0, 1\}$ represents a bit in the additional data M_h, which is to be embedded.

(2) When R is used for data embedding: visit all the pixels in E_h sequentially and append a bit "0" to F_R when $E_h(i,j) = 2^e - 2$. If $E_h(i,j) = 2^e - 1$, append a bit "1" to F_R and change $E_h(i,j)$ from $2^e - 1$ to $2^e - 2$ simultaneously. The reversible data hiding is conducted as follows.

$$E_h^*(i,j) = \begin{cases} E_h(i,j) & \text{if } D_h(i,j) < R \\ E_h(i,j)+b & \text{if } D_h(i,j) = R \\ E_h(i,j)+1 & \text{if } D_h(i,j) > R \end{cases} \tag{7}$$

By a similar method, the sub image E_l is converted to E_l^* and the marked-encrypted sub images E_h^* and E_l^* can be obtained. Finally, the marked-encrypted image E^* is generated by bitplane combination which is depicted in Fig. 2 and sent to the receiver.

2.3 Data extraction and image recovery

With the marked-encrypted image E^*, there are three cases where the receiver has the following: (1) both the data hiding key and the encryption key; (2) only the data hiding key; (3) only the encryption key. After receiving the marked-encrypted image E^*, the sub images E_h^* and E_l^* can be generated. We only introduce the data extraction and image recovery procedure in E_h^* here for brevity.

Case I: both the data hiding key and the encryption key. The receiver first transforms E^* into E_h^* and E_l^*, then extracts the additional data by

$$b = \begin{cases} 0 & \text{if } E_h^*(i,j)-E_h^*(i,1) = L \text{ or } R \\ 1 & \text{if } E_h^*(i,j)-E_h^*(i,1) = L-1 \text{ or } R+1 \end{cases} \tag{8}$$

The recovery operations are carried out by processing all the pixels in E_h^* using

$$E_h(i,j) = \begin{cases} E_h^*(i,j)+1 & \text{if } E_h^*(i,j)-E_h^*(i,1) \le L-1 \\ E_h^*(i,j) & \text{if } E_h^*(i,j)-E_h^*(i,1) = L \text{ or } R \\ E_h^*(i,j)\text{-}1 & \text{if } E_h^*(i,j)-E_h^*(i,1) \ge R+1 \end{cases} \tag{9}$$

where $E_h(i,j)$ represents the recovery pixel, and $j \in [2, m \times n]$, $E_h(i,1) = E_h^*(i,1)$. Next, the preprocessing pixels are recovered via using the location maps F_L and F_R:

$$E_h(i,j) = \begin{cases} 0 & \textit{if } E_h^*(i,j) = 1 \\ 2^e - 1 & \textit{if } E_h^*(i,j) = 2^e - 2 \end{cases} \tag{10}$$

Note that E_l can be recovered in a similar way and the original image I can be recovered after bitplane combination and image decryption.

Case II: only the data hiding key. With the data hiding key, the receiver can extract the data by applying Eqs. (8) and Eqs. (10), but the original image cannot be obtained.
Case III: only the encryption key. An approximate image I^* is acquired by direct decryption using decryption key. The receiver can roughly get the original image information, but cannot obtain the embedded data.

2.4 Capacity analysis

According to Eqs. (5)–(7), there is a connection between the embedding capacity and the difference value of adjacent pixels. The smaller the difference value $D_h(i,j)$ in Eqs. (5), the greater the value H_L and H_R, and the greater the embedding capacity. As is shown in Fig. 3, instead of considering each image pixel as an integer, the proposed algorithm transforms the original image into a sub-image with high bitplanes and a sub-image with low bitplanes, respectively. In general, the difference value of the adjacent pixels in a sub-image with high bitplanes is smaller than the one in original image. Due to the bitplane operations and adaptive embedding, the proposed algorithm can better explore the correlation between neighbor pixels. Higher correlation between neighbor pixels means a lower difference value of adjacent pixels and higher embedding capacity. Thus, the proposed algorithm achieves a higher embedding capacity.

3 Experimental results

In this section, we conduct several experiments to evaluate the proposed framework. Twelve standard test images (all images are downloaded from the USC-SIPI database (http://sipi.usc.edu/database)) with size 512×512 are shown in Fig. 4. Without loss of generality, we adopt a 3×3 mode as the block size and random bit stream as the additional data in all our experiments. First, the changes of the classic image "Lena" in different phases corresponding to Section 2 are shown in Fig. 5, in which (a) and (b) are the original image and its encrypted version. Fig. 5c shows the marked encrypted images containing additional data. With the marked image, the receiver owning the data hiding key can extract the additional data. If the receiver has both the encryption key and the data hiding key, the recovered image given in Fig. 5d can be perfectly obtained, which is the same as (a). If the receiver only has the encryption key, he or she can obtain an approximate image. The directly decrypted images with different parameters are exhibited in Fig. 5e–j. It is clear that the larger the parameter value e is, the better image quality the approximate image is. This is because the larger parameter means the more bitplanes in the sub-image with high bitplanes, which is critical to the embedding capacity.

As stated in the previous section, the contribution of the proposed paper is adding a "bitplane operations" process for existing methods, and improving the embedding

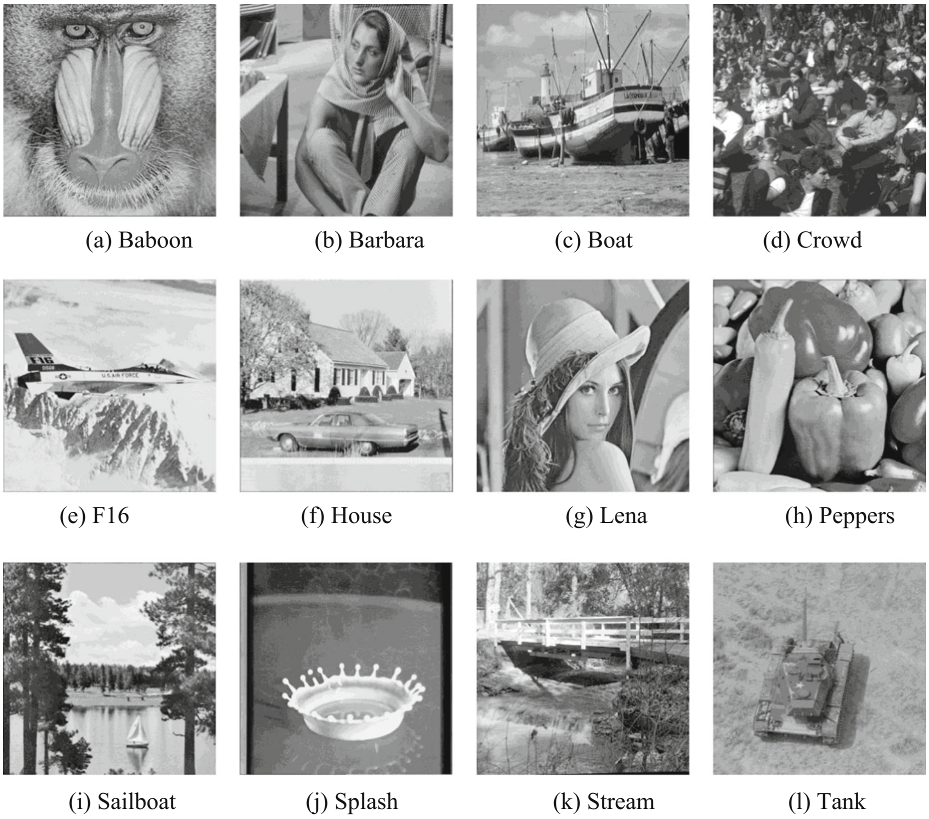


Fig. 4 The test images

performance of existing methods. To evaluate the proposed method, we choose algorithms in [6, 28] as two typical examples. However, the methods in [6] or [28] can apply different RDH methods, such as DHS and PEHS. Thus, we use the following four existing methods: [6] with

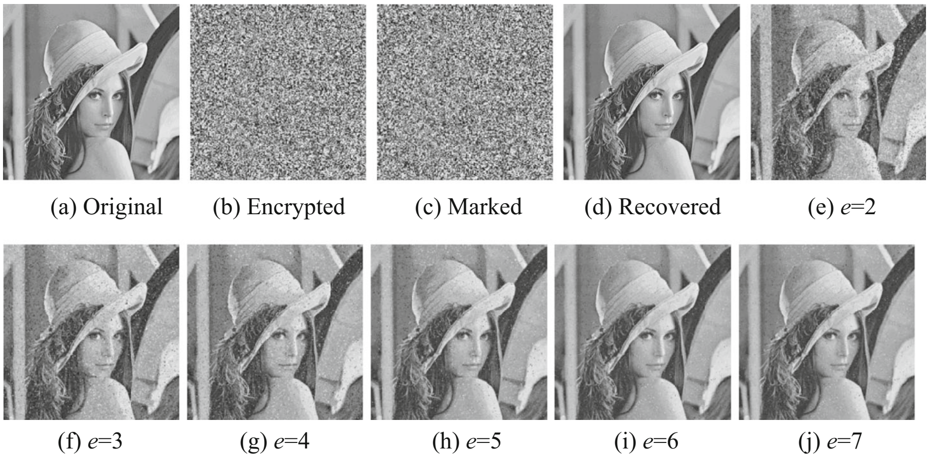


Fig. 5 The changes of "Lena" in different phases

Table 1 Performance comparison between before and after using the proposed method ([6] with DHS)

	Before	After											
		$e = 2$		$e = 3$		$e = 4$		$e = 5$		$e = 6$		$e = 7$	
	EC	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR
Baboon	7686	47,396	18.56	66,062	22.81	53,690	26.54	43,328	31.65	27,502	37.17	15,422	42.92
Barbara	24,231	88,936	18.50	103,596	22.82	98,484	28.17	89,189	32.24	68,382	37.63	43,490	43.21
Boat	18,803	98,331	18.49	115,672	22.79	101,368	27.30	87,186	32.21	59,064	37.52	34,616	43.11
Crowd	52,633	126,467	18.49	157,823	22.80	138,935	28.16	124,947	32.76	102,385	38.05	72,073	43.52
F16	44,484	124,810	18.55	157,806	22.79	139,113	28.17	124,253	32.76	102,102	38.05	72,084	43.52
House	42,765	116,639	18.50	135,131	22.80	121,261	28.17	102,870	32.44	85,626	37.84	63,261	43.43
Lena	31,672	112,390	18.56	140,530	22.79	128,200	28.17	115,181	32.61	87,187	37.86	56,089	43.34
Peppers	24,205	108,646	18.54	128,362	22.82	120,310	28.15	106,136	32.48	75,036	37.71	45,123	43.22
Sailboat	21,321	92,059	18.51	113,046	22.80	101,087	28.17	87,216	32.21	62,123	37.55	38,506	43.15
Splash	46,450	138,723	18.59	172,590	22.78	154,564	28.14	140,502	33.01	113,067	38.19	76,858	43.58
Stream	32,802	109,100	18.56	121,723	22.83	108,778	26.83	104,681	31.89	122,434	37.34	50,756	43.28
Tank	25,502	114,962	18.57	141,090	22.83	112,272	27.50	95,794	32.33	64,940	37.58	39,436	43.16

Table 2 Performance comparison between before and after using the proposed method ([6] with PEHS)

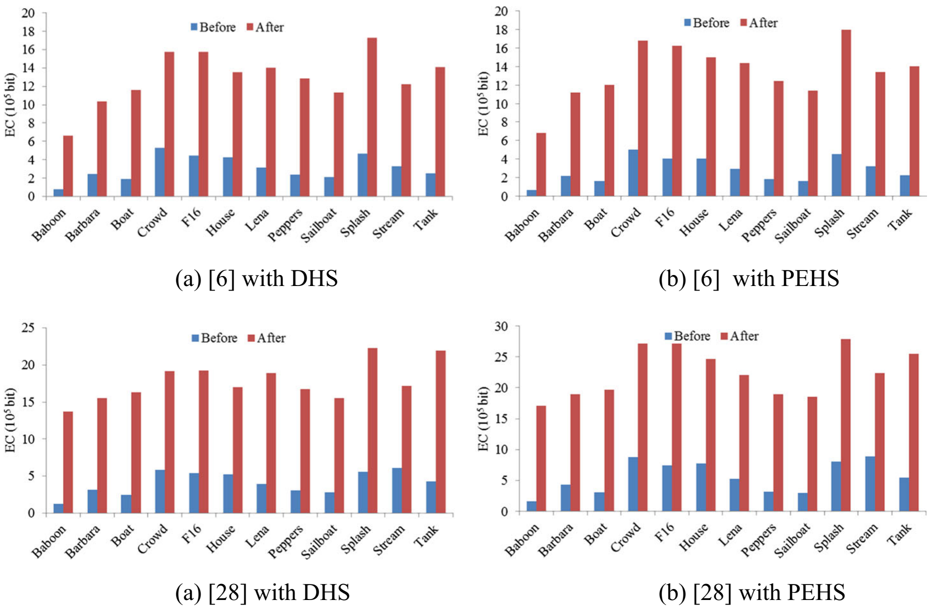
	Before	After											
		$e=2$		$e=3$		$e=4$		$e=5$		$e=6$		$e=7$	
	EC	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR
Baboon	6461	54,166	17.60	68,380	22.15	54,972	25.78	42,420	31.01	26,139	36.59	14,437	42.38
Barbara	21,906	99,001	17.56	111,701	22.14	102,264	27.59	91,284	31.55	68,507	37.01	42,959	42.64
Boat	16,421	107,184	17.56	120,246	22.16	102,883	26.45	85,383	31.49	56,205	36.88	32,160	42.54
Crowd	50,217	142,611	17.62	167,811	22.15	152,468	27.59	122,919	31.90	102,718	37.36	78,618	42.97
F16	40,743	133,516	17.59	162,342	22.15	142,352	27.58	126,217	31.97	101,798	37.36	71,081	42.91
House	40,671	129,345	17.63	150,234	22.13	133,199	27.60	111,809	31.81	92,618	37.26	68,463	42.88
Lena	29,339	119,193	17.60	144,022	22.14	128,383	27.59	113,864	31.82	84,759	37.18	53,935	42.74
Peppers	18,593	109,546	17.59	124,586	22.14	111,041	26.54	94,442	31.58	62,761	36.94	36,286	42.57
Sailboat	16,174	95,051	17.58	113,698	22.14	96,967	26.35	79,321	31.42	53,317	36.86	32,123	42.54
Splash	45,493	151,447	17.58	179,991	22.17	159,016	27.61	141,289	32.20	114,156	37.52	78,133	42.98
Stream	32,253	118,883	17.61	129,942	22.13	110,499	26.10	105,234	31.28	133,781	36.79	53,442	42.74
Tank	22,586	118,379	17.59	140,530	22.14	113,197	26.56	94,441	31.58	62,844	36.95	37,523	42.59

Table 3 Performance comparison between before and after using the proposed method ([28] with DHS)

	Before	After											
		$e = 2$		$e = 3$		$e = 4$		$e = 5$		$e = 6$		$e = 7$	
	EC	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR
Baboon	12,370	53,374	18.35	137,139	20.90	123,000	25.95	81,634	31.43	46,792	37.07	24,513	42.86
Barbara	31,210	66,932	19.62	134,861	21.78	155,539	26.47	129,044	32.00	93,838	37.58	57,702	43.20
Boat	24,439	59,734	20.54	163,024	21.77	157,412	26.75	127,224	32.05	83,383	37.47	46,484	43.08
Crowd	58,203	54,111	22.99	117,570	23.97	192,124	26.83	148,270	32.37	116,115	37.93	84,848	43.57
F16	54,044	59,569	22.87	92,898	25.05	192,591	26.89	164,077	32.47	128,003	38.00	88,638	43.54
House	52,178	53,320	22.38	121,642	22.96	170,061	26.80	142,955	32.18	110,158	37.78	78,254	43.43
Lena	39,229	105,591	18.96	179,397	21.99	189,039	26.84	157,755	32.38	115,753	37.85	71,641	43.35
Peppers	30,450	118,504	18.34	167,559	21.91	167,321	27.09	143,064	32.40	101,446	37.70	58,404	43.21
Sailboat	27,703	92,485	18.58	103,385	23.05	155,805	26.70	128,404	31.99	86,264	37.50	51,040	43.13
Splash	55,996	223,266	16.49	222,715	21.99	176,626	27.58	170,591	32.91	142,580	38.19	95,457	43.62
Stream	61,425	150,225	17.50	161,628	21.82	169,773	26.42	157,367	31.71	171,831	37.27	62,172	43.28
Tank	42,947	77,481	20.93	219,299	21.29	181,849	26.73	138,627	32.12	88,580	37.52	51,241	43.13

Table 4 Performance comparison between before and after using the proposed method ([28] with PEHS)

	Before	After											
		$e = 2$		$e = 3$		$e = 4$		$e = 5$		$e = 6$		$e = 7$	
	EC	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR	EC	PSNR
Baboon	16,440	71,407	17.77	170,656	20.54	154,213	25.63	105,488	31.07	61,764	36.65	32,496	42.41
Barbara	43,609	161,954	16.90	179,556	21.36	190,047	26.16	161,492	31.69	122,690	37.27	78,761	42.84
Boat	31,026	183,824	16.47	197,025	21.27	191,544	26.42	158,907	31.73	105,259	37.09	59,078	42.65
Crowd	87,772	263,990	15.80	271,287	21.31	249,067	26.61	195,766	32.22	161,476	37.79	124,331	43.37
F16	73,847	255,358	15.83	271,293	21.15	239,774	26.56	200,012	32.18	161,496	37.71	116,436	43.22
House	77,847	246,793	15.93	240,923	21.28	225,305	26.60	188,968	32.04	153,590	37.62	114,728	43.20
Lena	52,723	147,398	17.93	220,507	21.45	220,519	26.46	189,607	32.05	144,430	37.51	93,531	42.98
Peppers	31,464	145,447	17.41	189,134	21.37	190,069	26.55	159,284	31.86	109,091	37.15	61,249	42.67
Sailboat	30,049	124,571	17.59	153,999	21.87	185,140	26.30	149,451	31.55	97,384	37.00	56,055	42.62
Splash	80,999	279,008	15.87	277,330	21.41	224,865	27.14	204,392	32.56	179,908	37.94	129,171	43.36
Stream	88,545	141,867	18.65	216,534	21.49	224,325	26.16	191,444	31.48	169,752	36.99	90,545	42.99
Tank	54,054	214,019	16.37	255,391	20.82	212,092	26.34	171,435	31.81	116,515	37.20	69,484	42.75



(a) [6] with DHS

(b) [6] with PEHS

(a) [28] with DHS

(b) [28] with PEHS

Fig. 6 Comparison on maximum EC between before and after using the proposed method

DHS, [6] with PEHS, [28] with DHS, and [28] with PEHS. Detailed performance comparisons between before and after adding the proposed method for four existing methods are provided in Tables 1, 2, 3 and 4. EC and PSNR respectively mean the embedding capacity and peak signal-to-noise ratio, which are two criteria to evaluate RDHEI. "Before" represents the existing method which without adding the proposed "bitplane operations" process, and "After" represents the existing method after adding the proposed "bitplane operations" process. One obvious conclusion is, after using the proposed "bitplane operations" process under any *e* value, both algorithms achieve higher embedding capacity, while the PSNR value between the directly decrypted image and the original image decrease. It is generally known that in the RDHEI field, EC is relatively more important, and PSNR can be alleviated or even neglected [6]. Therefore, the proposed method has great practical significance. Another conclusion from Tables 1, 2, 3 and 4 is that, the parameter *e* has great influence over the embedding performance. EC is first increased and then decreased along with the increase of the value of parameter *e*. Due to the embedding process, both the sub-image with high bitplanes and the sub-image with low bitplanes can be used to embed data when the value of *e* is low, while only sub-images with high bitplanes can be used to embed data when the value of *e* is high. In general, the maximum EC can be achieved when the value of *e* is 3 or 4.

To better demonstrate the advantage of the proposed method in high capacity, we compared the maximum EC between before and after using the proposed method according to Tables 1, 2, 3 and 4.

Table 5 Average run time comparison (in embedding stage)

Method	DHS		PEHS	
	[6]	Proposed	[6]	Proposed
Run time (s) 48.76	1.06	6.92	2.32	19.83

For each image and each algorithm, we choose the e value which corresponding to maximum EC values. Figures 6 has shown the comparison results. Four sub-images correspond to the previous tables. From Fig. 6, the maximum EC of both algorithms can be improved significantly after introducing the proposed method. For example, for image "Lena", the embedding capacity of algorithm in [6] with DHS method is equal approximately to 30,000 bits, while the maximum embedding capacity of algorithm adding the proposed method rises to more than 140,000 bits (when $e = 3$). In some practical application that we are more concerned with EC rather than PSNR, we can select the e value which corresponding to maximum EC values. Thus, the proposed method has great practical significance.

The main factor that contributes to high capacity is bitplane operation, rather than the data hiding method or encryption method. As described in Subsection 2.4, due to the bitplane operations, the proposed algorithm can better explore the correlation between neighbor pixels and achieve better embedding capacity. The experimental results have shown that the embedding capacity can be increased significantly after introducing the proposed "bitplane operation", no matter which data hiding method is used.

The comparison of average run time in embedding phase between the method in [6] and the proposed time are shown in Table 5. The computation times mentioned in this table were measured on an Intel CPU (i7-7500 U, 3.5 GHz), Windows 7 PC with 4.00 GB RAM. From the table, the run times in [6] are approximately 1.06 and 2.32 s, when DHS and PEHS are used respectively. Note that the PEHS method costs more time due to the prediction process. When the "bitplane operation" is added, the times rise to 6.92 and 19.83 s. It is clear that the better performance in embedding capacity in the proposed method engenders more computational complexity, because of the added bitplane operations and parameter optimization. However, the impact of this is not serious in practical application.

4 Conclusion

In this paper, we propose a novel RDHEI method with high embedding capacity based on bitplane operations and adaptive embedding. Instead of considering the image pixel of 8 bits as an integer within the interval of [0,255], the data hider transforms the pixel into two components according to the bitplane parameter. Our experimental results show that high embedding capacity can be achieved using the bitplane-level operations and adaptive embedding, although the PSNR values between the original image and the directly decrypted image are lower compared with the existing algorithms. This is very useful for RDHEI, in which field PSNR is less important than embedding capacity. Future work aims at increasing the PSNR values of directly decrypted images in our method on the basis of keeping the embedding capacity as high as possible.

Acknowledgements This work is partially supported by National Natural Science Foundation of China (No. 61379152, 61772572 and 61403417), Natural Science Foundation of Guangdong Province of China (2017A030313366), and Fundamental Research Funds for the Central Universities (17lgjc45).

References

1. Chen YC, Shiu CW, Horng G (2014) Encrypted signal-based reversible data hiding with public key cryptosystem. J Vis Commun Image R 25(5):1164–1170

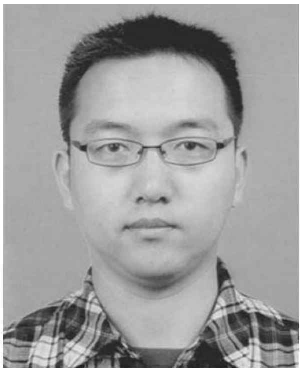
2. Cui J, Liu Y, Xu Y et al (2013) Tracking Generic Human Motion via Fusion of Low- and High-Dimensional Approaches. IEEE Trans Syst Man Cybern Syst 43(4):996–1002
3. Dragoi L, Coltuc D (2014) Local-prediction-based difference expansion reversible watermaking. IEEE Trans Image Process 23(4):1779–1790
4. Haouzia A, Noumeir R (2008) Methods for image authentication: a survey. Multimed Tools Appl 39(1):1–46
5. Hong W, Chen TS, Wu HY (2012) An improved Reversible data hiding in encrypted images using side match. IEEE Signal Process Lett 19(4):199–202
6. Huang F, Huang J, Shi YQ (2016) New framework for reversible data hiding in encrypted domain. IEEE Trans Inf Forensics Secur 11(12):2777–2789
7. Kumar R, Chand S (2017) A novel high capacity reversible data hiding scheme based on pixel intensity segmentation. Multimed Tools Appl 76(1):979–996
8. Lee SK, Suh YH, Ho YS (2006) Reversible image authentication based on watermarking. In: Proceeding IEEE Int. Conf. Multimedia Expo (ICME), Toronto, Canada, pp 1321–1324
9. Li X, Yang B, Zeng T (2011) Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection. IEEE Trans Image Process 20(12):3524–3533
10. Li X, Zhang W, Gui X et al (2013) A novel reversible data hiding scheme based on two-dimensional difference-histogram modification. IEEE Trans Inf Forensics Secur 8:1091–1100
11. Li X, Li B, Yang B et al (2013) General framework to histogram-shifting-based reversible data hiding. IEEE Trans Image Process 22:2181–2191
12. Li XL, Zhang WM, Gui XL, Yang B (2015) Efficient Reversible Data Hiding Based on Multiple Histograms Modification. IEEE Trans Inf Forensics Secur 10(9):2016–2027
13. Liao X, Shu C (2015) Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels. J Vis Commun Image R 28:21–27
14. Lin CC, Liu XL, Tai WL, Yuan SM (2015) A novel reversible data hiding scheme based on ambtc compression technique. Multimed Tools Appl 74(11):3823–3842
15. Liu Y, Zhang X, Cui J et al (2010) Visual analysis of child-adult interactive behaviors in video sequences. In: Proceeding International Conference on Virtual Systems and Multimedia, pp 26–33
16. Liu Y, Cui J, Zhao H et al (2012) Fusion of low-and high-dimensional approaches by trackers sampling for generic human motion tracking. In: Proceeding of IEEE International Conference on Pattern Recognition, pp 898–901
17. Liu Y, Nie L, Liu L et al (2016) From action to activity: Sensor-based activity recognition. Neurocomputing 181:108–115
18. Liu Y, Qu X, Xin G (2016) A ROI-based reversible data hiding scheme in encrypted medical images. J Vis Commun Image R 39:51–57
19. Lu Y, Wei Y, Liu L et al (2016) Towards unsupervised physical activity recognition using smartphone accelerometers. Multimed Tools Appl 76(8):10701–10719
20. Ma B, Shi YQ (2016) A Reversible Data Hiding Scheme Based on Code Division Multiplexing. IEEE Trans Inf Forensics Secur 11(9):1914–1927
21. Qian ZX, Zhang XP (2016) Reversible data hiding in encrypted images with distributed source encoding. IEEE Trans Circuits Syst Video Technol 26(4):636–646
22. Qian Z, Han X, Zhang X (2013) Separable reversible data hiding in encrypted images by *n*-nary histogram modification. In: IEEE Proceeding International Conference on Multimedia Technology, pp 869–976
23. Qian Z, Zhang X, Ren Y, Feng G (2016) Block cipher based separable reversible data hiding in encrypted images. Multimed Tools Appl 75(21):13749–13763
24. Qin C, Zhang XP (2015) Effective reversible data hiding in encrypted image with privacy protection for image content. J Vis Commun Image R 31:154–164
25. Shi YQ, Li XL, Zhang XP, Wu HT, Ma B (2016) Reversible data hiding: Advances in the past two decades. IEEE Access 4:3210–3237
26. Tian J (2003) Reversible data embedding using a difference expansion. IEEE Trans Circuits Syst Video Technol 13(8):890–896
27. Yang H, Kot AC (2007) Pattern-based data hiding for binary image authentication by connectivity-preserving. IEEE Trans Multimedia 9(3):475–486
28. Yin Z, Luo B, Hong W (2014) Separable and error-free reversible data hiding in encrypted image with high payload. Sci World J. https://doi.org/10.1155/2014/604876
29. Zhang XP (2011) Reversible data hiding in encrypted image. IEEE Signal Process Lett 18(4):255–258
30. Zhang XP (2012) Separable data hiding in encrypted image. IEEE Trans Inf Forensics Secur 7(2):826–832
31. Zhang X, Qian Z, Feng G, Ren Y (2014) Efficient reversible data hiding in encrypted images. J Vis Commun Image R 25(2):322–328
32. Zhang W, Wang H, Hou D, Yu N (2016) Reversible data hiding in encrypted images by reversible image transformation. IEEE Trans Multimedia 18(8):1469–1479

33. Zheng S, Li D, Hu D, Ye D, Wang L, Wang J (2016) Lossless data hiding algorithm for encrypted images with high capacity. Multimed Tools Appl 75(21):13765–13778
34. Zhou J, Sun W, Dong L, Liu X (2016) Secure Reversible Image Data Hiding over Encrypted Domain via Key Modulation. IEEE Trans Circuits Syst Video Technol 26(3):441–452

Di Fuqiang received the B.S. and the M.S. degrees from the Engineering University of Chinese People's Armed Police, China, in 2015. He is currently working toward the Ph.D. degree in multimedia security at Department of Electronic Technology, Engineering University of Chinese People's Armed Police, Xi'an, China. His research interests include multimedia security, image processing, and information hiding.

Huang Fangjun received the B.S. degree from the Nanjing University of Science and Technology, China, in 1995, and the M.S. and Ph.D. degrees from the Huazhong University of Science and Technology, China, in 2002 and 2005, respectively. He has been with Sun Yat-Sen University since 2005, where he is currently an Associate Professor. From 2009 to 2010, he was a Postdoctoral Researcher with the New Jersey Institute of Technology, NJ, USA. From 2013 to 2014, he was a Korea Foundation for Advanced Studies Scholar with Korea University, Seoul, South Korea. His research interests include reversible data hiding, steganography, steganalysis, and digital forensics.

Zhang Minqing received the B.S. degree from the Engineering University of Chinese People's Armed Police, China, in 1989, and the M.S. and Ph.D. degrees from the Northwestern Polytechnical University, China, in 2001 and 2016, respectively. She has been with the Engineering University of Chinese People's Armed Police, China, since 2016, where she is currently a Professor. Her research interests include information hiding, steganography, steganalysis, and multimedia security.



Liu Jia received the B.S. and the M.S. degrees from the Engineering University of Chinese People's Armed Police, China, in 2004 and 2007, respectively, and the Ph.D. degree from the Shanghai Jiao Tong University, China, in 2012. He has been with the Engineering University of Chinese People's Armed Police, China, since 2012, where he is currently an Associate Professor. His research interests include Machine Learning, information hiding, steganography and steganalysis.

Yang Xiaoyuan received the B.S. degree from the Xidian University, China, in 1982, and the M.S. degree from the Xidian University, China, in 1991. He has been with the Engineering University of Chinese People's Armed Police, China, since 2001, where he is currently a Professor. His research interests include information hiding, steganography, steganalysis, and multimedia security.