Improved Reversible Data Hiding in Encrypted Images using Histogram Modification

Shuang Yi, Yicong Zhou*
Department of Computer and Information Science
University of Macau, Macau 999078, China
Email: *yicongzhou@umac.mo.

***Abstract*—Inspired by Zhang *et al.*'s method that applies the integer discrete wavelet transform (DWT) to the original image, and embeds the secret data into the middle (LH, HL) and high (HH) frequency sub-bands of integer DWT coefficients with histogram modification based method, we propose a reversible data hiding method that embeds the secret data into the encrypted prediction error values. Compared with the LH, HL and HH integer DWT coefficients, the prediction error values generated by our proposed method are more concentrated to 0, and thus a high visual quality of the marked decrypted image can be achieved. Experimental results show that our proposed method has a better performance than Zhang's.**

I. Introduction

Reversible data hiding (RDH) in images aims to hide the secret data into an original image by slightly modifying the pixel values, and to perfectly recover the original image after the secret data have been extracted. Many RDH methods have been proposed such as difference expansion (DE) [1], histogram shifting (HS) [2], prediction error expansion [3], [4]. In recent years, RDH in encrypted images (RDHEI) has caught many researchers' attention. The original image is encrypted by content provider and the secret data is embedded by data hider, and the data hider has no knowledge about the encrypted image. The receiver can obtain different contents (secret data, original image or both) with different access rights. Due to the property of reversibility, RDHEI can be used for many applications such as cloud storage and medical image management system, where the original images are unwilling to disclose to the cloud provider or system administrator. For example, to prevent any unauthorized access, the content owner encrypts his personal images before store them to the Cloud. The Cloud provider who manage the resources may add some notations to the encrypted images without knowing the original image content. The receivers with different access rights are able to obtain different contents (original image, additional data or both).

Many RDHEI methods have been proposed in recent years. In Zhang [5] and Hong *et al.*'s [6] methods, the original image is encrypted by a stream cipher, then one bit of the secret data is embedded into an image block by flipping the 3 least significant bits (LSBs) of half pixels within the block. Data extraction and image recovering are accomplished by a smoothness measurement function of the recovered image. Li *et al.* [7] proposed a RDHEI method by embedding one bit of the secret data in one selected pixel by flipping the 3 LSBs. The four neighboring pixels of each selected pixel will keep unchanged to ensure that the selected pixels can be accurately predicted. In Zhou *et al.*'s method [8], $n(n \geqslant 1)$ bits of the secret data are embedded in one image block using the public key cryptography. And the support vector machine (SVM) is utilized to determine whether an image block is successfully decrypted. In [9] and [10], two separable RDHEI methods are proposed by compression the a number of the LSB planes and the 4^{th} LSB plane for reserving spare space to embed secret data. Yin *et al.* [11] encrypt the original image using a coarse-grained encryption to permute the blocks in a global image and a fine-grained encryption to permute the pixels in each block. Then two pixel are randomly selected form each block to indicate the peak points, the secret data is then embedded into the pixels that equals to the either of the peak points using the histogram modification method.

For the VRAE methods, the data extraction and image recovering procedures are mainly accomplished by measuring the smoothness of the recovered image, thus they may suffer from incorrectly extraction of secret data and/or original image, and the embedding rate is relatively low. In order to solve these problems, Ma *et al.* [12] first proposed a RRBE method, which suggests reserving spare space from the original image before encryption, and embedding the secret data into the reserved spare space. Mathew *et al.* [13] improved Ma's method [12] by using an active block exchange strategy so that the quality of the marked decrypted image can be improved. Zhang *et al.* [14] randomly select a number pixels and obtain their estimation-error and embed the secret data into the encrypted estimation-error values. In Zhang *et al.*'s method in [15], the integer discrete wavelet transform (DWT) is first applied to the original image, the secret data is then embedded into the middle (LH, HL) and high (HH) frequency sub-bands of integer DWT coefficients with histogram modification based method. Cao *et al.* [16] reserve the spare space by using the sparse representation method.

Inspired by the method in [15], in this paper, we propose a RDHEI method by using 1/4 of the pixels in original image to predict the remaining 3/4 pixels, and the secret data is then embedded into the encrypted prediction-error values. Compared with the LH, HL and HH of integer DWT

978-1-5090-1897-0/16/$31.00 ©2016 IEEE

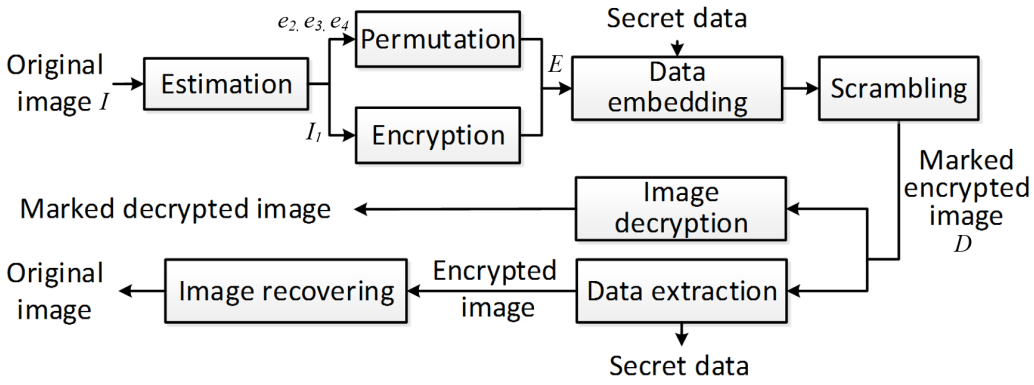


Fig. 1. The framework of proposed the algorithm.

coefficients, our proposed method can generate the prediction error values that are more concentrated to 0, and thus a high visual quality of the marked decrypted image can be achieved.

The rest of this paper is organized as follows: Section II will introduce our proposed algorithm, Section III will show some simulation results and the conclusion will be drawn in Section IV.

II. Proposed Algorithm

The framework of the proposed algorithm is shown in Fig. 1. It consists of three phases: image encryption, data embedding, and data extraction/image recovering. The content owner first uses 1/4 of the pixels in the original image to predict its remaining 3/4 pixels and obtains the prediction-error values, encrypts the unchanged pixels and the prediction-error values separately by using the encryption key K_E. Then the data hider embeds secret data into the prediction-error values using the data hiding key K_H and scrambles the whole image using the sharing key K_S. At the receiver side, one can obtain the secret data or marked decrypted image which is similar to the original image using (K_S, K_H) or (K_S, K_E), respectively. And if three keys (K_S, K_H, K_E) are held, the receiver can extract the secret data and perfectly recover the original image.

A. Image Encryption

Assume that an original image I is with a size of $M \times N$, and the pixel values are within a range of [0,255]. Firstly, decompose the image I to form four sub-images, namely I_1, I_2, I_3 and I_4, by

$$\begin{cases} I_1^{(i,j)} = I^{(2i-1,2j-1)} \\ I_2^{(i,j)} = I^{(2i-1,2j)} \\ I_3^{(i,j)} = I^{(2i,2j-1)} \\ I_4^{(i,j)} = I^{(2i,2j)} \end{cases} \tag{1}$$

where $1 \leqslant i \leqslant M/2$ and $1 \leqslant j \leqslant N/2$. The illustration of image decomposition is shown in Fig. 2. Then use the pixels in I_1 to predict the pixels in the rest three sub-images by

$$\hat{I}_2^{(i,j)} = \begin{cases} \lceil (I_1^{(i,j)} + I_1^{(i,j+1)})/2 \rceil, & \text{if } i < N/2 \\ I_1^{(i,j)}, & \text{if } i = N/2 \end{cases} \tag{2}$$

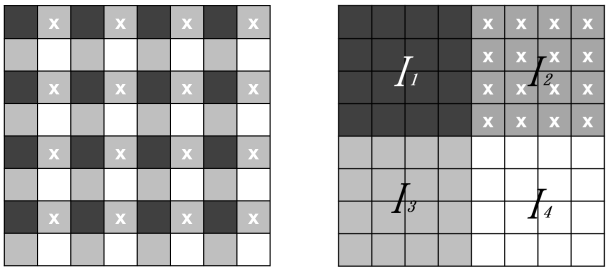


Fig. 2. Image decomposition. (a) The original image; (b) decomposed sub-images.

$$\hat{I}_3^{(i,j)} = \begin{cases} \lceil (I_1^{(i,j)} + I_1^{(i+1,j)})/2 \rceil, & \text{if } i < M/2 \\ I_1^{(i,j)}, & \text{if } i = M/2 \end{cases} \tag{3}$$

$$\hat{I}_4^{(i,j)} = \begin{cases} \lceil (I_1^{(i,j)} + I_1^{(i+1,j+1)})/2 \rceil, & \text{if } H_1^{(i,j)} < H_2^{(i,j)} \\ \lceil (I_1^{(i,j+1)} + I_1^{(i+1,j)})/2 \rceil, & \text{if } H_1^{(i,j)} \geqslant H_2^{(i,j)} \\ \lceil (I_1^{(i,j)} + I_1^{(i+1,j)})/2 \rceil, & \text{if } i < \frac{M}{2}, j = \frac{N}{2} \\ \lceil (I_1^{(i,j)} + I_1^{(i,j+1)})/2 \rceil, & \text{if } i = \frac{M}{2}, j < \frac{N}{2} \\ I_1^{(i,j)}, & \text{if } i = \frac{M}{2}, j = \frac{N}{2} \end{cases} \tag{4}$$

where

$$H_1^{(i,j)} = |(I_1^{(i,j)} - I_1^{(i+1,j+1)}|, \ (i < M/2, j < N/2) \tag{5}$$

and

$$H_2^{(i,j)} = |(I_1^{(i,j+1)} - I_1^{(i+1,j)}|, \ (i < M/2, j < N/2) \tag{6}$$

Thus we can obtain the prediction-error values of these three sub-images by

$$\begin{cases} e_2^{(i,j)} = I_2^{(i,j)} - \hat{I}_2^{(i,j)} \\ e_3^{(i,j)} = I_3^{(i,j)} - \hat{I}_3^{(i,j)} \\ e_4^{(i,j)} = I_4^{(i,j)} - \hat{I}_4^{(i,j)} \end{cases} \tag{7}$$

Obviously, the prediction-error can be positive or negative values. We use the most significant bit (MSB) to represent the sign bit (e.g., 0 for positive value and 1 for negative value)

and the rest 7 bits to store its absolute value. Thus, only the prediction-error values fall into the range of [-127,127] can be successfully stored with 8 bits. For the prediction error values out of this data range, we record them into a overflow map O_1 and embed it into the sub-image I_1 using the RDH method [4].

Next, we encrypt the four sub-images separately. For I_1, we encrypt it by

$$E_1 = I_1 \oplus R \tag{8}$$

where $'\oplus'$ is a bit-level XOR operation and R is a $\frac{M}{2} \times \frac{N}{2}$ random matrix generated by the encryption key K_E, and $R^{(i,j)} \in [0, 255]$. Here, R can be generated by secure chaotic systems such as [17], [18]. For e_2, e_3 and e_4, we permutate them to form the encrypted sub-images E_2, E_3 and E_4 using K_E. Finally we concatenate E_1, E_2, E_3 and E_4 to generate the final encrypted image E.

B. Data Embedding

After obtained the encrypted image, the data hider uses the data hiding key K_H to encrypt the secret data and embeds the encrypted data into the sub-images E_2, E_3 and E_4 using histogram modification method. The detailed histogram modification based data embedding procedures are list as follows:

Step 1: According to the data hiding key K_H, permutate pixels in sub-images E_2, E_3 and E_4 and form a pixel sequence. Keep the first 500 pixels in the sequence and denote the remaining $K = (3MN/4) - 500$ pixels as a sub-sequence $\{E_s^i\}_{i=1}^K (E_s^i \in [-127, 127])$. Let $h(k)$ be the histogram when prediction-error value in the sequence $\{E_s^i\}_{i=1}^K$ equals to k.

Step 2: Given a C-bits encrypted secret data, concatenate it with LSBs of preselected 500 pixels to form the payload P, thus the length of P is $(C+500)$ bits. Then we use the center p pairs of histograms of $\{E_s^i\}_{i=1}^K$ to embed the payload, where p is calculated by

$$p = \arg\min_k \{ \sum_{e=-k}^{k-1} h(e) \geqslant C + 500, k = 1, 2, ...\} \tag{9}$$

Next, divide $\{E_s^i\}_{i=1}^K$ into two sub-sequences: $\{E_s^u\}_{u=1}^U$ and $\{E_s^v\}_{v=U+1}^K$ that denote the first U elements and the remaining part of $\{E_s^i\}_{i=1}^K$, respectively. Here $\{E_s^u\}_{u=1}^U$ satisfies that the number of elements falling into the range of $[-2p, 2p)$ equals to $(C+500)$, which means that by only shifting or modifying the elements in $\{E_s^u\}_{u=1}^U$, we can embed the whole payload bits, and pixels in $\{E_s^v\}_{v=U+1}^K$ will keep unchanged. Thus we embed the payload into $\{E_s^u\}_{u=1}^U$ by

$$E_s^{'u} = \begin{cases} E_s^u + p, & \text{if } E_s^u \geqslant p \\ E_s^u - p, & \text{if } E_s^u < -p \\ 2 \times E_s^u + m, & \text{otherwise} \end{cases} \tag{10}$$

where $m \in \{0, 1\}$ is one bit of the secret data.

Step 3: After embedded the payload bits, $E_s^{'u}$ may out of the range of [-127,127]. Thus, we store these overflow information into an overflow map O_2 and embed it into the preselected 500 pixels by simply LSB substitution. The parameters p and U also need to store with O_2 since they will be utilized for secret data extraction at the receiver side.

Step 4: Concatenate $\{E_s^{'u}\}_{u=1}^U$ and $\{E_s^v\}_{v=U+1}^K$ to form the marked encrypted sequence $\{E_s^{'i}\}_{i=1}^K$ which contains payload bits. According to the data hiding key K_H, inversely permutate $E_s^{'i}$ and other 500 pixels and form 3 sub-images and put them to their original positions.

Finally, scramble all pixels in the image using the sharing key K_S to generate the final marked encrypted image D.

C. Data Extraction and Image Recovering

At the receiver side, with different combinations of security keys, the receiver can obtain different contents separately. With the keys $\{K_S, K_H\}$, one can extract the secret data from the marked encrypted image, and if K_E is also held, the original image can be further obtained. If the receiver holds only the keys $\{K_S, K_E\}$, he/she can obtain a marked encrypted image which is similar to the original one. Next, we present the data extraction and image recovering procedures separately.

1) Data extraction: Using the sharing key K_S, firstly, inversely scramble the marked encrypted image, and denote the obtained image as D_s. Then extract the encrypted sub-image E_1 from D_s, permutate the remaining $(3MN/4)$ pixels and form a pixel sequence based on the data hiding key K_H. Extract the overflow map O_2, if any, parameters p and U from the LSBs of the first 500 pixels of the sequence, and denote the remaining pixel sub-sequence as $\{E_s^{'i}\}_{i=1}^K$. Separate $\{E_s^{'i}\}_{i=1}^K$ into $\{E_s^{'u}\}_{u=1}^U$ and $\{E_s^{'v}\}_{v=U+1}^K$, respectively. According to the parameter p, extract the payload bits from $\{E_s^{'u}\}_{u=1}^U$ by

$$m = E_s^{'u} - 2\lfloor E_s^{'u}/2 \rfloor, \quad \text{for } -2p \leqslant E_s^{'u} < 2p \tag{11}$$

Finally, the receiver obtains the encrypted secret data from payload and decrypts it using the data hiding key K_H and obtains the plain secret data.

2) Image recovering: After obtained the secret data, if the receiver also holds K_E, he/she can further recover the original image. Firstly, recover the pixel values in $\{E_s^{'u}\}_{u=1}^U$ by

$$E_s^u = \begin{cases} E_s^{'u} + p, & \text{if } E_s^{'u} < -2p \\ E_s^{'u} - (p-1), & \text{if } E_s^{'u} \geqslant 2p \\ \lfloor E_s^{'u}/2 \rfloor, & \text{if } -2p \leqslant E_s^{'u} < 2p \end{cases} \tag{12}$$

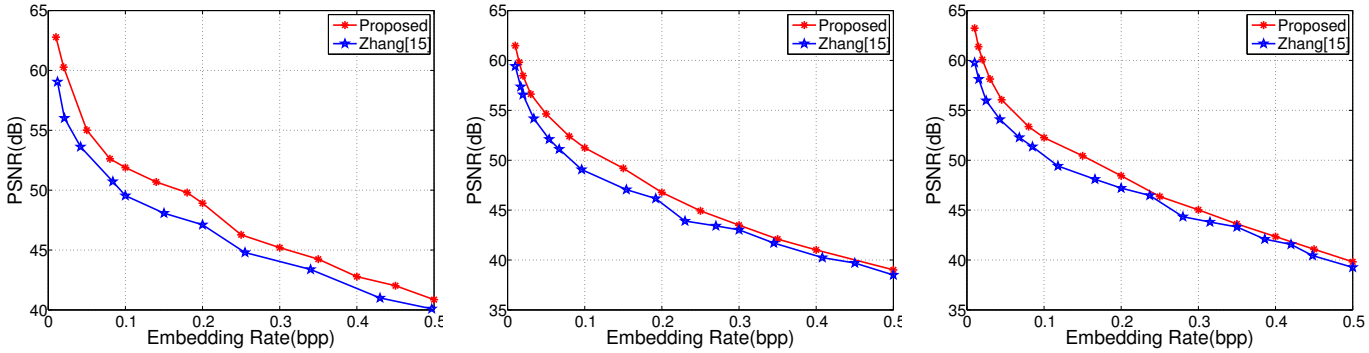
Next, replace the LSBs of the preselected 500 pixels by extracted payload bits. According to K_H and K_E, inversely permutate these 500 pixels, $\{E_s^u\}_{u=1}^U$ and $\{E_s^v\}_{v=1}^{K-U}$ to form three sub-images: e_2, e_3 and e_4. Then decrypt E_1 using Eq. (13) to obtain the decrypted sub-image I_1.

$$I_1 = E_1 \oplus R \tag{13}$$

where R is generated by the same way as in image encryption process. Next, extract overflow map O_1, if any, from I_1, and recover the overflow values in e_2, e_3 and e_3. According to Eqs. (2)-(4), obtain the prediction-error sub-images $\hat{I}_2, \hat{I}_3$ and

(a) (b) (c) (d)

Fig. 3. (a) The original image; (b) the marked encrypted image with embedding rate = 0.1777 bpp, $p = 1$; (c) the marked decrypted image with PSNR=49.929 dB and (d) the recovered image.



(a) (b) (c)

Fig. 4. PSNR comparisons of the marked decrypted images generated by the proposed and other RDHEI methods. (a) Lena; (b) Peppers; (c) Boat.

$\hat{I}_4$, recover the original sub-images I_2, I_3 and I_4 by

$$\begin{cases} I_2^{(i,j)} = e_2^{(i,j)} + \hat{I}_2^{(i,j)} \\ I_3^{(i,j)} = e_3^{(i,j)} + \hat{I}_3^{(i,j)} \\ I_4^{(i,j)} = e_4^{(i,j)} + \hat{I}_4^{(i,j)} \end{cases} \tag{14}$$

Finally, put the pixels in four sub-images to their original image pixel locations, thus, the recovered image is generated.

If only the keys $\{K_S, K_E\}$ are held, the receiver can obtain the marked decrypted image which is similar to the original one. Firstly, inversely permutate the marked encrypted image using the sharing key K_S, the obtained image is denoted as D_s. Then divide D_s into four parts, the encrypted sub-image E_1 and three encrypted prediction error sub-images E_2, E_3 and E_4. Decrypt these four sub-images separately. For E_1, decrypt it using Eq. (13), the decrypted sub-image is denoted as I_1. For other three sub-images, inversely permutate them using the image encryption key K_E and generate three sub-images: e_2, e_3 and e_4. Next, according to Eqs. (2)-(4), obtain the prediction-error sub-images $\hat{I}_2, \hat{I}_3$ and $\hat{I}_4$, and recover the original sub-images I_2, I_3 and I_4 by Eq. (14), Finally, put the pixels in four sub-images to their original locations, and the generated marked encrypted image is denoted by I_m. Note that the pixel values in the marked encrypted image I_m may exceed the data range of [0,255], we simply set the pixel value to 0 (255) if it is less (larger) than 0 (255).

III. Simulation Results

All images used in our simulation results are with the size of 512×512, and their pixel values fall into [0, 255].

Fig. 3 shows the experimental results of the proposed algorithm using the standard test image *Lena* which selected from the Miscelaneous database[1], the parameter $p = 1$ and the embedding rate r = 0.1777 bpp. From the results we can observe that the marked decrypted image has high visual quality (peak signal to noise ratio (PSNR)=49.929 dB) when ER=0.1777 bpp, the recovered image is exactly the same with the original image due to the reversibility of the proposed algorithm.

We select three commonly used standard test images, *Lena*, *Peppers* and *Boat* from the Miscelaneous database to show the quality of the marked decrypted images generated by proposed method and Zhang *et al.*'s [15] method. As can been seen from Fig. 4, the PSNR results of our proposed method outperform Zhang *et al.*'s [15] at different embedding rates.

We randomly selected 200 images in BOWSBase[2] to show the PNSR results of marked decrypted images generated by proposed and Zhang *et al.*'s [15] methods at different

[1]http://decsai.ugr.es/cvg/dbimagenes/g512.php.
[2]http://bows2.ec-lille.fr/.

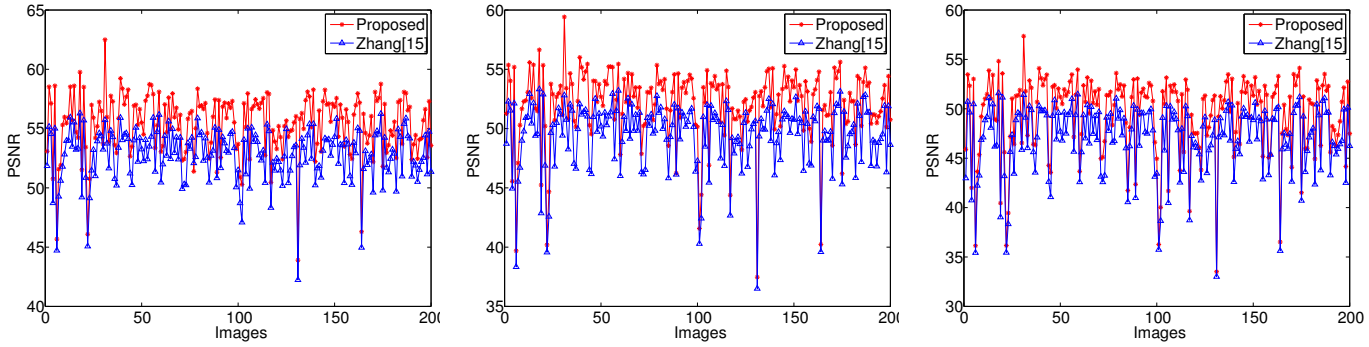


Fig. 5. PSNR comparisons of 200 marked decrypted images generated by the proposed and Zhang *et al.*'s [15] methods at different embedding rates. (a) 0.1 bpp; (b) 0.2 bpp; (c) 0.3 bpp.

embedding rates. The results are plotted in Fig. 5. From the results we can observe that our proposed method has higher PSNR results than Zhang *et al.*'s at different embedding rates. The average PSNR results of these 200 images at different embedding rates are listed in Table I. One can observe that, compared with Zhang *et al.*'s method, the average PSNRs of our proposed algorithm improved about 2.5 dB on average.

TABLE I
AVERAGE PSNRS OF 200 MARKED ENCRYPTED IMAGE GENERATED BY THE PROPOSED AND ZHANG *et al.*'S METHODS AT DIFFERENT EMBEDDING RATES (ER).

PSNR (.dB)	**0.1 bpp**	**0.2 bpp**	**0.3 bpp**
Proposed method	55.3674	52.0805	49.4231
Zhang *et al.*'s [15]	52.7828	49.4819	46.9825
Gain of PSNR	2.5846	2.5986	2.4406

IV. CONCLUSION

This paper proposed a new RDHEI method by using 1/4 of the pixels in original image to predict the remaining 3/4 of the pixels, and embedding the secret data bits into the encrypted prediction error values. It can achieve full reversibility, data extraction and image recovery procedures can be accomplished separately by using different combinations of security keys. Experimental results have shown that our proposed method has better performance than Zhang *et al.*'s [15].

ACKNOWLEDGMENT

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 Grants MYRG2014-00003-FST and MYRG2016-00123-FST.

REFERENCES

[1] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.
[2] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.
[3] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721–730, 2007.
[4] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3524–3533, 2011.
[5] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.
[6] W. Hong, T.-S. Chen, and H.-Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.
[7] M. Li, D. Xiao, Z. Peng, and H. Nan, "A modified reversible data hiding in encrypted images using random diffusion and accurate prediction," *ETRI Journal*, vol. 36, no. 2, pp. 325–328, 2014.
[8] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2015.
[9] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2012.
[10] X. Zhang, Z. Qian, G. Feng, and Y. Ren, "Efficient reversible data hiding in encrypted images," *Journal of Visual Communication and Image Representation*, vol. 25, no. 2, pp. 322–328, 2014.
[11] Z. Yin, B. Luo, and W. Hong, "Separable and error-free reversible data hiding in encrypted image with high payload," *The Scientific World Journal*, vol. 2014, p. 8, 2014.
[12] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.
[13] T. Mathew and M. Wilscy, "Reversible data hiding in encrypted images by active block exchange and room reservation," in *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, pp. 839–844.
[14] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, no. 0, pp. 118–127, 2014.
[15] S. Zhang, T. Gao, and G. Sheng, "A joint encryption and reversible data hiding scheme based on integer-DWT and arnold map permutation," *Journal of Applied Mathematics*, vol. 2014, p. 12, 2014.
[16] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–1, 2015.
[17] Z. Hua, Y. Zhou, C.-M. Pun, and C. L. P. Chen, "2D Sine Logistic modulation map for image encryption," *Information Sciences*, vol. 297, no. 0, pp. 80–94, 2015.
[18] Z. Hua and Y. Zhou, "Image encryption using 2D Logistic-adjusted-Sine map," *Information Sciences*, vol. 339, pp. 237–253, 2016.