REVERSIBLE DATA HIDING IN ENCRYPTED IMAGES BY RESERVING ROOM BEFORE ENCRYPTION USING INTERPOLATION TECHNIQUE

A.Nizamudeen[1] Dr.T.K.Shanthi [2]
[1] P.G. Student [2] Assistant Professor
[1, 2]Thanthai Periyar Govt Instituteof Technology, Vellore.

Abstract— Recently, more and more attention is paid to reversible data hiding (RDH) in encrypted images, since it maintains the excellent property that the original cover can be lossless recovered after embedded data is extracted while protecting the image content's confidentiality. The previous method embed data by reversibly reserving room before encryption, without number of times of hide. In this paper, we propose a novel method by reserving room before encryption with a traditional RDH algorithm, and thus it is easy for the data hider to reversibly embed data in the encrypted image. The proposed method can achieve real reversibility, that is, data extraction and image recovery are free of any error. Experiments show that this novel method can embed more than 15 times as large payloads for the image quality as the previous methods, such as for PSNR=60dB.

I. INTRODUCTION

REVERSIBLE data hiding (RDH) in images is a technique, by which the original cover can be losslessly recovered after the embedded message is extracted. This important technique is widely used in medical imagery, military imagery and law forensics, where no distortion of the original cover is allowed. Since first introduced, RDH has attracted considerable research interest.

In theoretical aspect, Kalker and Willems established a rate-distortion model for RDH, through which they proved the rate-distortion bounds of RDH for memoryless covers and proposed a recursive code construction which, however, does not approach the bound.

"Achieves excellent performance in two different prospects
"

- Real reversibility is realized, that is, data extraction andimage recovery are free of any error.
- For given embedding rates, the PSNRs of decrypted imagecontaining the embedded data are significantly improved;and for the acceptable PSNR, the range of embedding ratesis greatly enlarged.

II. INTERPOLATION TECHNIQUE

A. ADDITIVE INTERPOLATION-ERROR EXPANSION

Essentially,the data embedding approach of the proposed reversible watermarking scheme, Namely additive interpolation error expansion, is a kind of DE. But it is different from most DE approaches in two important aspects:

It uses interpolation-error, instead of interpixel difference or prediction error, to embed data.

It expands difference, which is interpolation-error here, by addition instead of bit shifting.

First, interpolation values of pixels are calculated Using interpolation technique, which works by guessing a pixel value from its surrounding pixels. Then interpolation-error are

Obtained via

$$e = x - x'$$

Where x' are the interpolation values of pixels x. let LM and RM denote the corresponding values of the two highest points of interpolation-errors histogram and be formulated as

$$\begin{cases} LM = \arg\max_{e \in E} \text{hist}(e) \\ RM = \arg\max_{e \in E - \{LM\}} \text{hist}(e) \end{cases}$$

Where hist(e) is number of occurrence when the interpolation-error is equal to e and E denotes the set of interpolation-errors. Without loss of generality, assume LM<RM. Then, we divide the interpolation-errors into two parts:

(1) Left interpolation-errors(LE): interpolation-error e satisfies e<=LM.
(2) Right interpolation-errors(RE): interpolation-error e satisfies e>=RM.

The additive interpolation-error expansion is formulated as

$$e' = \begin{cases} e + \text{sign}(e) \times b, & e = LM \text{ or } RM \\ e + \text{sign}(e) \times 1, & e \in (LN, LM) \cup (RM, RN) \\ e, & \text{otherwise} \end{cases}$$

Where e' is the expanded interpolation-error, b is the bit to be embedded, and sign (.) is a sign function defined as

$$\text{sign}(e) = \begin{cases} 1, & e \in RE \\ -1, & e \in LE. \end{cases}$$

In (3) the parameters LN and RN are defined as

$$\begin{cases} LN = \arg\min_{e \in LE} \text{hist}(e) \\ RN = \arg\min_{e \in RE} \text{hist}(e). \end{cases}$$

Usually, LM is a very small integer and in most cases 0,while LN is a smaller integer that with no interpolation-error satisfying e=LN. Similarly, in mostcases, RM is equal to 1 and RN is a larger integer with no interpolation-error satisfying e=RN. After expansion of interpolation-errors, the watermarked pixels x'' become

$$x'' = x' + e'.$$

During the extracting process, with the same interpolation algorithm, we can obtain the same interpolation values x' and the corresponding interpolation-errors via

$$e' = x'' - x'.$$

Note that (7) is the deformation of (6). Once the same LM,LN,RM and RN

Are known,embedded data can be extracted through

$$b = \begin{cases} 0, & e' = LM \text{ or } RM \\ 1, & e' = LM - 1 \text{ or } RM + 1. \end{cases}$$

Then the inverse function of additive interpolation-error expansion is applied to recover the original interpolation-errors

$$e = \begin{cases} e' - \text{sign}(e') \times b, & e' \in [LM-1, LM] \cup [RM, R \\ e' - \text{sign}(e') \times 1, & e' \in [LN, LM-1) \cup (RM+1, \\ e', & \text{otherwise.} \end{cases}$$

Finally,we can restore the original pixels through

$$x = x' + e.$$

Compared with previous DE the additive interpolation-error expansion is advantageous in three aspects: first, the distortion of additive expansion is smaller since each pixel is altered atmost by 1.

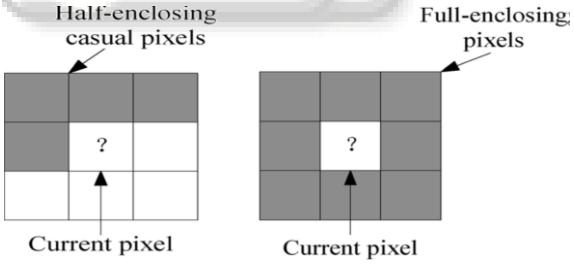


Fig. 1: interpretations of half-enclosing casual pixels and full-enclosing pixel

Fig 3.1 shows that Second, no location map is needed to tell between expanded interpolation-errors and nonexpanded ones since they are distinguishable with LM,LN,RM, and RN. Last, interpolation-errors are more expandable than interpixel in Fig 11.

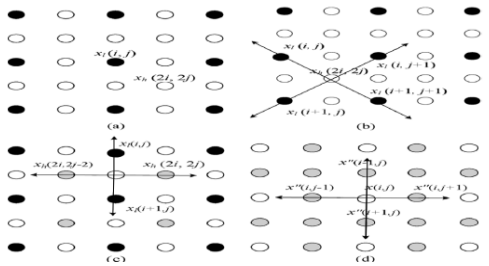


Fig. 2: (a) Low Resolution(b)High Resolution(c)Interpolation Of Residual Samples Of High Resolution (d)Interpolation Of Sample Pixels.

$$\begin{cases} S_{45} = \{x_l(i, j+1), x'_{45}, x_l(i+1, j)\} \\ S_{135} = \{x_l(i, j), x'_{135}, x_l(i+1, j+1)\}. \end{cases}$$

We can use to obtain the estimations of the missing high resolution pixels and finish in Fig 2(a).

III. IMPLEMENTATION OF PROPOSED SYSTEM

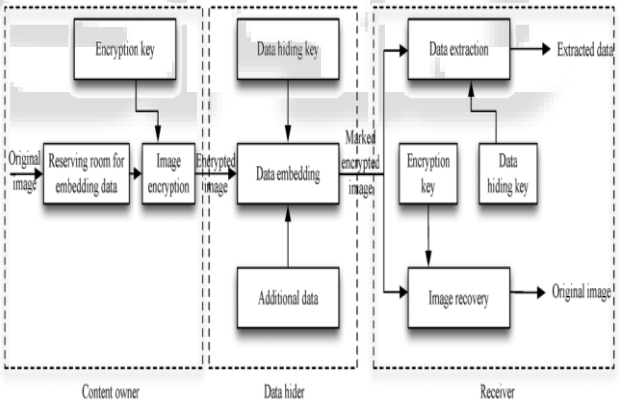
A. RRBE

Since losslessly vacating room from the encrypted images is relatively difficult and sometimes inefficient, why are we still so obsessed to find novel RDH techniques working directly for encrypted images? If we reverse the order of encryption and vacating room, i.e., reserving room prior to image encryption at content owner side, the RDH tasks in encrypted images would be more natural and much easier which leads us to the novel

Fig. 3: RRBE framework

framework fig3.shows, "reserving room before encryption (RRBE)".

The content owner first reserves enough space on original image and then converts the image into its encrypted version with the encryption key. Now, the data embedding process in encrypted images is inherently reversible for the data hider only needs to accommodate data



into the spare space previous emptied out. The data extraction and image recovery are identical to that of Framework VRAE. Obviously, standard RDH algorithms are the ideal operator for reserving room before encryption and can be easily applied to Framework RRBE to achieve better performance compared with techniques from Framework VRAE. This is because in this new framework, we follow the customary idea that first losslessly compresses the redundant image content (e.g., using excellent RDH techniques) and then encrypts it with respect to protecting privacy.

Next, we elaborate a practical method based on the Framework "RRBE", which primarily consists of four stages: generation of encrypted image, data hiding in encrypted image, data extraction and image recovery.Note that the reserving operation we adopt in the proposed method is a traditional RDH approach.

B. GENERATION OF ENCRYPTED IMAGE

Actually, to construct the encrypted image, the first stage can be divided into three steps: image partition, self reversible embedding followed by image encryption. At the beginning, image partition step divides original image into two parts A andB ; then, the LSBs of A are reversibly embedded into B with a standard RDH algorithm so that LSBs of Acan be used for accommodating messages; at last, encrypt the rearranged image to generate its final version.

C. IMAGE PARTITION

The operator here for reserving room before encryption is a standard RDH technique, so the goal of image partition is to construct a smoother area B, on which standard RDH algorithms can achieve better performance. To do that, without loss of generality, assume the original image Cis an 8 bits gray-scale image with its size and pixels . First, the content owner extracts from the original image, along the rows, several overlapping blocks whose number is determined by the size of to-be-embedded messages, denoted by l. In detail, every block consists of mrows, where m=[l/n], and the number of blocks can be computed through n=M-m+1 . An important point here is that each block is overlapped by pervious and/or subsequential blocks along the rows. For each block, define a function to measure its first-order smoothness.

$$f = \sum_{u=2}^{m} \sum_{v=2}^{N-1} \left| \mathbf{C}_{u,v} - \frac{\mathbf{C}_{u-1,v} + \mathbf{C}_{u+1,v} + \mathbf{C}_{u,v-1} + \mathbf{C}_{u,v}}{4} \right|$$

Higher f relates to blockswhich contain relativelymore complex textures. The content owner, therefore, selects the particular block with the highest f to be A, and puts it to the front of the image concatenated by the rest part B with fewer textured areas.The above discussion implicitly relies on the fact that only single LSB-planes of A is recorded. It is straight forward that the content owner can also

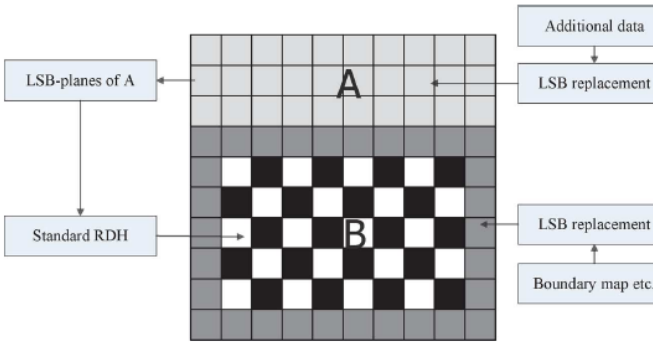


Fig. 4: illustration of image partition and embedding process

embed two or more LSB-planes A into B, which leads to half, or more than half, reduction in size of A . However, the performance of A , in terms of PSNR, after data embedding in the second stage decreases significantly with growing bit-planes exploited. Therefore, in this paper, we investigate situations that at most three LSB-planes of Aare employed and determine the number of bit-plane with regardFig 4 shows illustration of image partition and embedding processto different payloads experimentally in the next section.

D. SELF-REVERSIBLE EMBEDDING

The goal of self-reversible embedding is to embed the LSB-planes of A into B by employing traditional RDH algorithms. For illustration, we simplify the method for demonstrate the process of self-embedding. Note that this step does not rely on any specific RDH algorithm. Pixels in the rest of image are first categorized into two sets: white pixels with its indices I and j satisfying (i+j) mod 2 =0 and black pixels whose indices meet(i+j)mod 2 =1. Then, each white pixel is estimated by the interpolation value obtained with the four black pixels surrounding it as follows

$$\mathbf{B}_{\backslash}^{i,j} = w_1 \mathbf{B}^{i-1,j} + w_2 \mathbf{B}^{i+1,j} + w_3 \mathbf{B}^{i,j-1} + w_4 \mathbf{B}^{i,j+1}$$

where the weight is determined by the samemethod as proposed in them. The estimating error is calculated via and then some data can be embedded into the estimating error sequence with histogram shift, which will be described later. After that, we further calculate the estimating errors of black pixels with the help of surrounding white pixels that may have been modified.

E. IMAGE ENCRYPTION

After rearranged self-embedded image, denoted by x, is generated, we can encrypts to construct the encrypted image, denoted by e.With a stream cipher, the encryption version of x is easily obtained. For example, a gray value ranging from 0 to 255 can be represented by8 bits,

$$\mathbf{X}_{i,j}(0), \mathbf{X}_{i,j}(1), \ldots, \mathbf{X}_{i,j}(7)$$

$$\mathbf{X}_{i,j}(k) = \left\lfloor \frac{\mathbf{X}_{i,j}}{2^k} \right\rfloor \bmod 2, \quad k = 0, 1, \ldots, 7.$$

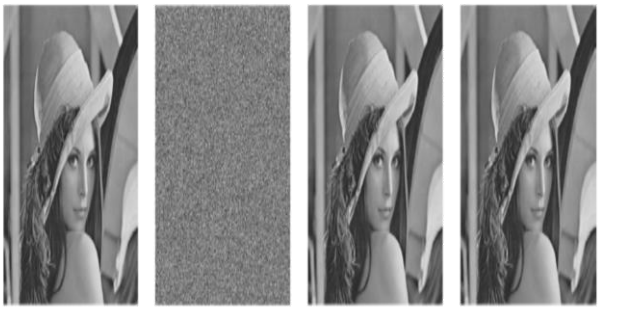


Fig. 5. (a) Original image, (b) encrypted image, (c) decrypted image containing messages (embedding rate 0.1 bpp), (d) recovery version.

The encrypted bits can be calculated through exclusive-or operation

$$\mathbf{E}_{i,j}(k) = \mathbf{X}_{i,j}(k) \oplus r_{i,j}(k),$$

Where is generated via a standard stream cipher determined by the encryption key. Finally, we embed 10 bits

information in to LSBs of first 10 pixels in encrypted version of A to tell data hider the number of rows and the number of bit-plane he can embed information into. Note that after image encryption, the data hider or a third party cannot access the content of original image without the encryption key, thus privacy of the content owner being protected.

Once the data hider acquires the encrypted image, he can embed some data into it, although he does not get access to the original image. The embedding process starts with locating the encrypted version of, denoted by. Since has been rearranged to the top of, it is effortless for the data hider to read 10 bits information in LSBs of first 10 encrypted pixels.

$$\mathbf{X}''_{i,j}(k) = \mathbf{E}'_{i,j}(k) \oplus r_{i,j}(k)$$

$$\mathbf{X}''_{i,j} = \sum_{k=0}^{7} \mathbf{X}''_{i,j}(k) \times 2^k,$$

After knowing how many bit-planes and rows of pixels he can modify, the data hider simply adopts LSB replacement to substitute the available bit-planes with additional data. Finally, the data hider sets a label following to point out the end position of embedding process and further encrypts according to the data hiding key to formulate marked encrypted image denoted. Anyone who does not possess the data hiding key could not extract the additional data.

IV. DATA HIDING IN ENCRYPTED IMAGE

Once the data hider acquires the encrypted image , he can embed some data into it, although he does not get access to the original image. The embedding process starts with locating the encrypted version of A, denoted by A. Since has been rearranged to the top of e, it is effortless for the data hider to read 10 bits information in LSBs of first 10 encrypted pixels. After knowing how many bit-planes and rows of pixels he can modify, the data hider simply adopts LSB replacement to substitute the available bit-planes with additional data . Finally, the data hider sets a label following to point out the end position of embedding process and further encrypts according to the data hiding key to formulate marked encrypted image. Anyone who does not possess the data hiding key could not extract the additional data.

A. DATA EXTRACTION AND IMAGE RECOVERY

Since data extraction is completely independent from image decryption, the order of them implies two different practical applications.

B. EXTRACTING DATA FROM ENCRYPTED IMAGES AND DECRYPTED IMAGES

To manage and update personal information of images which are encrypted for protecting clients' privacy, an inferior database manager may only get access to the data hiding key and have to manipulate data in encrypted domain. The order of data extraction before image decryption guarantees the feasibility of our work in this case. When the database manager gets the data hiding key, he can decrypt the LSB-planes of Aeand extract the additional data by directly reading the decrypted version.

When requesting for updating information of encrypted images, the database manager, then, updates information through LSB replacement and encrypts updated information according to the data hiding key all over again. As the whole process is entirely operated on encrypted domain, it avoids the leakage of original content. Fig 5 shows before and after encryption for LENA image.

In Case 1, both embedding and extraction of the data are manipulated in encrypted domain. On the other hand, there is a different situation that the user wants to decrypt the image first and extracts the data from the decrypted image when it is needed. The following example is an application for such scenario. Assume Alice outsourced her images to a cloud server, and the images are encrypted to protect their contents. Into the encrypted images, the cloud server marks the images by embedding some notation, including the identity of the images' owner, the identity of the cloud server and time stamps, to manage the encrypted images Fig 5 shows lenaimage, Theorder of image decryption before/without data extraction is perfectly suitable for this case. Some attempts on RDH in encrypted images have been made. In Zhang divided the encrypted image into several blocks. By flipping 3 LSBs of the half of pixels in each block, room can be vacated for the embedded bit. The data extraction and image recovery proceed by finding which part has been flipped in one block. This process can be realized with the help of spatial correlation in decrypted image. Hong ameliorated Zhang's method at the decoder side by further exploiting the spatial correlation using a different estimation equation and side match technique to achieve much lower error rate. These two methods mentioned above rely on spatial correlation of original image to extract data. That is, the encrypted image should be decrypted first before data extraction.

C. IMAGE DECRYPTION

After generating the marked decrypted image,the content owner can further extract the data and recover original image. The process is essentially similar to that of traditional RDH methods.

The following outlines the specific steps:

Step 1. Record and decrypt the LSB-planes of A''accordingto the data hiding key; extract the data until the end label is reached.

Step 2. Extract and boundary map from the LSB of marginal area of B''.Then, scan B'' to undertake the following steps.

Step 3.Ifblackpixels is equal to 0, whichmeans no black pixels participate in embedding process, go to Step 5.

Step 4. Calculate estimating errors of the black pixels.

If belongs to [1, 254], recover the estimating error and original pixel value in a reverse order and extract embedded bits when is equal topixels , (or) and .

Else, if, refer to the corresponding bit in boundary map. If skip this one, else operate like .

Repeat this step until the part of payload is extracted. If extracted bits are LSBs of pixels in marginal area, restore them immediately.

Step 5. Calculate estimating errors of the white pixels, and extract embedded bits and recover white pixels in the same manner with Step 4. If extracted bits are LSBs of pixels in marginal area, restore them immediately.

Step 6. Continue doing Step 2 to Step 5 rounds on and merge all extracted bits to form LSB-planes of A.Until now, we have perfectly recover B.

Step 7. Replace marked LSB-planes ofA'' with its original bits extracted B'' from get original cover image C .

V. CONCLUSION

Reversible data hiding in encrypted images is a new topicdrawing attention because of the privacy-preserving requirements from cloud data management. Previous methods implement RDH in encrypted images by vacating room after encryption,as opposed to which we proposed by reserving roombefore encryption. Thus the data hider can benefit from the extra space emptied out in previous stage tomake data hiding process effortless.The proposed method can take advantage of all traditional RDH techniques for plain images and achieve excellent performance without loss of perfect secrecy. Furthermore, this novel method can achieve real reversibility, separate data extraction and greatly improvement on the quality of marked decrypted images.

REFERENCES

[1] T. Kalker and F.M.Willems, "Capacity bounds and code constructions for reversible data-hiding," in *Proc. 14th Int. Conf. Digital Signal Processing (DSP2002)*, 2002, pp. 71–76.
[2] W. Zhang, B. Chen, and N. Yu, "Capacity-approaching codes for reversible data hiding," in *Proc 13th Information Hiding (IH'2011),LNCS 6958*, 2011, pp. 255–269, Springer-Verlag.
[3] W. Zhang, B. Chen, and N. Yu, "Improving various reversible data hiding schemes via optimal codes for binary covers," *IEEE Trans. Image Process.*, vol. 21, no. 6, pp. 2991–3003, Jun. 2012.
[4] J. Fridrich and M. Goljan, "Lossless data embedding for all image formats," in *Proc. SPIE Proc. Photonics West, Electronic Imaging, Security and Watermarking of Multimedia Contents*, San Jose, CA, USA, Jan. 2002, vol. 4675, pp. 572–583.
[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.
[6] Z. Ni, Y. Shi, N. Ansari, and S. Wei, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 3, pp. 354–362, Mar.2006.
[7] D.M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, Mar. 2007.
[8] X. L. Li, B. Yang, and T. Y. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3524–3533, Dec. 2011.
[9] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," *Signal Process.*, vol. 89, pp. 1129–1143, 2009.
[10] L. Luo*et al.*, "Reversible image watermarking using interpolation technique," *IEEE Trans. Inf. Forensics Security*, vol. 5, no. 1, pp. 187–193, Mar. 2010.

