IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

NEW FRAMEWORK OF REVERSIBLE DATA HIDING IN ENCRYPTED JPEG BIT STREAMS

[1] Mrs.S.Yoga,M.Sc(CS&IT).,M.Sc.,(Maths).,M.Phil(CS).,M.Phil.,(Maths), Assistant Professor,

[2] K. Dharshini, M.Sc (Computer Science),

[1] [2] Department Of Computer Science, Sakthi College of Arts and Science for Women, Oddanchatram.

ABSTRACT

This paper proposes a novel framework of reversible data hiding in encrypted JPEG bit stream. We first provide a JPEG encryption algorithm to encipher a JPEG image to a smaller size and keep the format compliant to JPEG decoders. After an image owner uploads the encrypted JPEG bit streams to cloud storage, the server embeds additional messages into the cipher text to construct a marked encrypted JPEG bit stream. During data hiding, we propose a combined embedding algorithm including two stages, the Huffman code mapping and the ordered histogram shifting. The embedding procedure is reversible. When an authorized user requires a downloading operation, the server extracts additional messages from the marked encrypted JPEG bit stream and recovers the original encrypted bit-stream losslessly. After downloading, the user obtains the original JPEG bit stream by a direct decryption. The proposed framework out-performs previous works on RDH-EI. First, since the tasks of data embedding/extraction and bit stream recovery are all accomplished by the server, the image owner and the authorized user are required to implement no extra operations except JPEG encryption or decryption. Second, the embedding payload is larger than state-of-the-art works.

I INTRODUCTION

Reversible data hiding in encrypted images (RDH-EI) is an emerging technique originated from reversible data hiding in plaintext images [1-2], which has been investigated by many researchers [3-21]. In the cloud storage scenario, this technique is realized by a protocol that contains three parties, an image owner, a cloud server and an authorized user. The image owner encrypts the contents before uploading images onto a cloud storage server. The server hides additional messages into the encrypted images. The RDH-EI protocol

guarantees that the hidden message can be exactly extracted by the server, and the original content of the images can be losslessly recovered by the authorized user.

RDH-EI is especially useful for labeling the cipher text in cloud storage, as shown in Fig. 1. When image owners hope to protect their privacy, RDH-EI first provides a secure encryption algorithm for the owners to encrypt their images before up-loading. On the cloud side, RDH-EI allows the server to label an encrypted image by data hiding, e.g., hiding the identities, timestamps, and remarks into the cipher text to generate a marked encrypted copy. Therefore, the labels are attached inside the cipher text, providing a better management for ad-ministrations. Meanwhile, storage overheads can also be saved. On the other hand, when an authorized user downloads the encrypted image from the cloud, the original content can be losslessly recovered after image decryption. In traditional systems of file management, the server constructs a metadata file to record the information of the uploaded images. The RDH-EI technique provides an alternative way, which accommodates additional information of the image inside the encrypted bit-stream. Therefore, no metadata files are needed anymore for labeling the uploaded images.

Many RDH-EI methods were proposed in the past five years [3-21]. Most methods were designed for uncompressed images [3-17], while some works were for JPEG bit streams [18-21]. Since JPEG images are widely used on Internet, this paper focuses on RDH-EI in JPEG bit streams. We propose a novel RDH-EI framework to make this technique more practical. Compared with previous works [18-21], two achievements are achieved in the proposed method. First, the tasks of data embedding, data extraction and bit stream recovery are all done by the server. There are no extra computation burdens for owner/user except encryption/decryption.

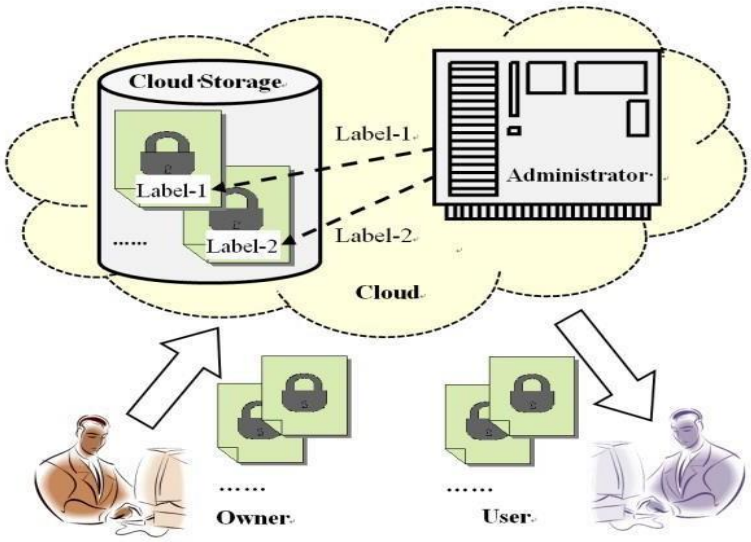


Fig.1. RDH-EI for Cloud Storage

1.1 RDH-EI FOR UNCOMPRESSED IMAGES

RDH-EI was first realized in uncompressed nature images [3]. The content owner encrypts the original image using a stream cipher algorithm and then uploads the encrypted image onto the cloud. The cloud server embeds additional bits into cipher text by flipping three least significant bits (LSB) of half pixels in each block. On the recipient end, the authorized user decrypts the marked encrypted image and generates two candidates for each block by flipping LSBs again. Since the original block of a nature image is smoother than the interfered block, one hidden bit can be extracted and the original block can be recovered. This method was improved in [4] using a side match algorithm to investigate the spatial correlation between neighboring blocks. The flipping based approaches in [3] and [4] were improved in [5] to reduce errors by comparing more neighbor pixels. However, when the pixels in a block have identical values, data extraction and image recovery in [3-5] may fail. A swapping and shifting based RDH-EI method was proposed to overcome this drawback [6]. Machine learning was also used in RDH-EI to improve the embedding capacity [7].

Although the methods in [2-7] have good embedding and recovery capabilities, data extraction must be done after image decryption. This limitation makes the technique less useful in cloud storage. Separable algorithms were proposed to resolve the problem [8]. With a pseudo-randomly generated matrix, the server compresses some LSB-planes of the encrypted pixels to fewer bits to generate spare rooms for hiding additional messages. Therefore, the hidden bits can be extracted directly from the marked encrypted image. On the recipient side, the original contents are recovered by estimating LSBs using MSBs of neighboring pixels. This method was improved in [9] by selecting appropriate bit planes from the encrypted image. Distributed source coding (DSC) is used to design RDH-EI in [10]. The server compresses some selected bits in the encrypted image to hide additional messages. A much higher capacity can be achieved using a Low Density-Parity Check (LDPC) based Slepian Wolf encoder. This method was further improved in [11] using a progressive recovery algorithm, which achieves a better embedding rate. In [12], an RDH-EI with higher em-bedding capacity was proposed by maintaining some redundancies during image encryption.

Another type of RDH-EI is realized by preprocessing the original image. Some works require the image owner to vacate spare rooms in the plaintext image before image encryption. We name this additional operation of vacating spare rooms in plaintext as preprocessing. After that, the image owner en-crypts the processed image and uploads it onto the server. Image encryption should not be accounted as preprocessing, since encryption is required in all RDH-EI methods. In [13], LSBs of some pixels are embedded into other pixels using traditional RDH for plaintext images. Therefore, these LSBs are vacated as spare rooms. The processed image is then encrypted and uploaded to the cloud. On cloud, the server embeds additional bits into the encrypted image using the specified spare rooms, which provides a very high embedding rate. In [14], some pixels are used to estimate the rest before encryption. After encrypting the pixels and estimation errors, a final version of the encrypted image is formulated. The server realizes data hiding by modifying the

estimation errors. In [15], patch-level sparse representation is used to explore the correlations of neighbor pixels. Another RDH-EI approach was realized by histogram shifting in the spatial permuted images [16]. In [17], image transformation was proposed to encrypt one image to another. Additional bits are embedded by RDH in plaintext images. The preprocessing based methods in [8-17] can achieve much better embedding rates, but require extra RDH operations before image encryption.

1.2 RDH-EI FOR JPEG BIT STREAMS

The aforementioned RDH-EI is for uncompressed images. However, those methods are not useful in many applications because most images transmitted over Internet are compressed, e.g., the popular JPEG. As a consequence, some RDH-EI works were proposed for JPEG bit streams [18-21].

The first RDH-EI for JPEG bit streams was proposed in [18]. This scheme begins with a JPEG encryption algorithm, in which the appended bits of Huffman codes are encrypted by a stream cipher, and all Huffman codes are kept unchanged. After encryption, the JPEG file size is preserved and the format is compliant to JPEG decoders. On cloud, the server selects the encrypted bit streams of some blocks as candidates. Additional bits are encoded by LDPC-based error correction codes (ECC) and embedded into the useful candidate bit stream by flipping the LSBs of the encrypted appended bits of the AC coefficients in each candidate block. After the authorized user downloads and decrypts the marked encrypted bit stream, LSBs of the appended bits of useful candidates are flipped again to estimate the additional bits using a predefined blocking artifact function and an ECC decoder. Meanwhile, the original bit stream is losslessly recovered according to the extracted bits. This method is improved in [19], in which the embedding room was reserved before bit stream encryption. Although the embedding capacity is larger, the preprocessing requires the image owner to do an additional computation. Another solution of reversibly hiding data in encrypted JPEG bit stream was proposed in [20], in which image encryption and data embedding are combined together. By scrambling the JPEG structure, the image is encrypted and the additional bits are embedded. The method in [20] cannot be used in cloud storage since the server is unable to embed bits into the encrypted bit stream.

To enhance the security of JPEG encryption and improve the embedding capability, another RDH-EI for JPEG bit stream was proposed in [21]. During JPEG bit stream encryption, a new JPEG bit stream is constructed by selecting a portion of blocks from the whole image. Bit streams of the rest blocks are encrypted and hidden in the JEPG header. With a compression algorithm, some appended bits of the encrypted JPEG bit stream are compressed to accommodate additional bits. On the recipient side, the authorized user uses an iterative algorithm to recovery the original JPEG bit stream according to the variation of blocking artifacts. Compared with [18], this method has larger capacity and better security, and the embedded bits can be directly extracted by the server.

Although RDH-EI methods for JPEG in [18-21] have good capabilities in data hiding and image recovery, there exists one problem that the recipient must do a recovery task after de-cryption. This recovery burden on the recipient side limits the real application of RDH-EI techniques, since users always hope to do nothing except image encryption and decryption. To this end, this paper proposes a new JPEG RDH-EI framework, in which no recovery task is required for the owner or the user.

1.3 NEW FRAMEWORK

The proposed RDH-EI framework focuses on labeling the encrypted JPEG images on cloud storage. There are three par-ties, including the image owner, the cloud server and the authorized user. The owner encrypts a JPEG bit stream and uploads it to the cloud. The cloud server embeds additional messages into the encrypted bit stream to generate a marked encrypted bit stream. The hidden messages can be extracted from the marked encrypted bit stream. When an authorized requires a downloading operation, the server recovers the original encrypted bit stream. After decryption, the user obtains the original JPEG image. The procedure is shown in Fig. 2.

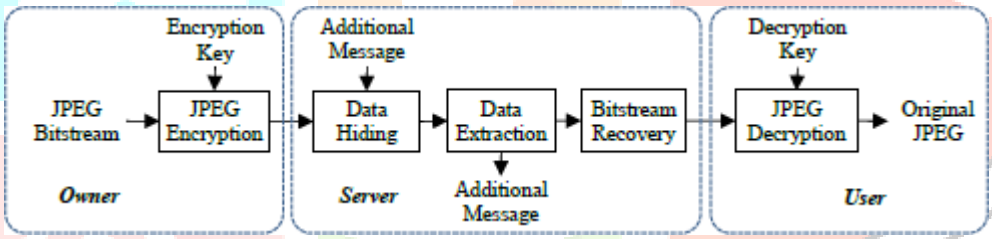


Fig.2. New Framework of RDH-EI for JPEG

Fig. 3 shows the traditional framework used in previous works. The user has to implement the JPEG recovery after decryption [18] [21]. Sometimes the owner is also required to do preprocessing [19], i.e., generating spare rooms before encryption. In Fig. 2, we propose to accomplish all computation tasks by the cloud server. Therefore, data extraction and bit-stream recovery are transparent to the owner or the user.

The traditional framework keeps the length of encrypted bit stream unchanged after data hiding, which leads to a limited embedding capacity. To embed more bits into the encrypted bit stream, the proposed framework allows the increment of bit stream length. With a condition that the amount of embedded bits must be larger than the bit stream increment [22], the pro-posed framework uses a two-phase combined embedding algorithm to increase the embedding payload and eliminate the bit stream increment.

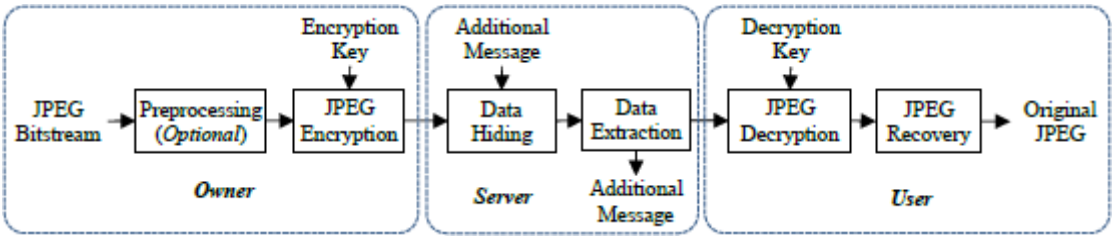


Fig. 3. Traditional Framework of RDH-EI for JPEG

II LITERATURE SURVAY

OVERVIEW

Reversible image data hiding (RIDH) is a special category of data hiding technique, which ensures perfect reconstruction of the cover image upon the extraction of the embedded message. The reversibility makes such image data hiding approach particularly attractive in the critical scenarios, e.g., military and remote sensing, medical images sharing, law forensics and copyright authentication, where high fidelity of the reconstructed cover image is required. The majority of the existing RIDH algorithms are designed over the plaintext domain, namely, the message bits are embedded into the original, un-encrypted images. The early works mainly utilized the lossless compression algorithm to compress certain image features, in order to vacate room for message embedding.

A NOVEL REVERSIBLE DATA HIDING SCHEME BASED ON TWODIMENSIONAL DIFFERENCEHISTOGRAM MODIFICATION

Author Name: Swapna Kumari and B Dasari Subbarao

Problem Definition

In this thesis, based on two-dimensional difference histogram modification, a novel reversible data hiding (RDH) scheme is proposed by using difference-pair mapping (DPM). First, by considering each pixel-pair and its context, a sequence consisting of pairs of difference values is computed. Then, a two-dimensional difference-histogram is generated by counting the frequency of the resulting difference-pairs. Finally, reversible data embedding is implemented according to a specifically designed DPM. Here, the DPM is an injective mapping defined on difference pairs.

Findings

It is a natural extension of expansion embedding and shifting techniques used in current histogram-based RDH methods. By the proposed approach, compared with the conventional one-dimensional difference-histogram and one dimensional prediction-error-histogram-based RDH methods, the image redundancy can be better exploited and an improved embedding performance is achieved. Moreover, a pixel pair-selection strategy is also adapted to priority use the pixel-pairs located in smooth image regions to embed data. This can further enhance the embedding performance.

Conclusion

Experimental results demonstrate that the proposed scheme outperforms some state-of-the-art RDH works

AN INPAINTING-ASSISTED REVERSIBLE STEGANOGRAPHIC SCHEME USING A HISTOGRAM SHIFTING MECHANISM

Author Name: Chuan Qin

Problem Definition

In this thesis, we propose a novel prediction-based reversible steganography scheme based on image in painting. First, reference pixels are chosen adaptively according to the distribution characteristics of the image content. Then, the image in painting technique based on partial differential equations is introduced to generate a prediction image that has similar structural and geometric information as the cover image.

Finding

Through the use of the adaptive strategy for choosing reference pixels and the in painting predictor, the prediction accuracy is high, and more embeddable pixels are acquired. Finally, by using the two selected groups of peak points and zero points, the histogram of the prediction error is shifted to embed the secret bits reversibly. Since the same reference pixels can be exploited in the extraction procedure, the embedded secret bits can be extracted from the stego image correctly, and the cover image can be restored lossless.

Conclusion

The proposed scheme provides a greater embedding rate and better visual quality compared with recently reported methods.

REVERSIBLE DATA HIDING WITH OPTIMAL VALUE TRANSFER

Author Name: Mrs. A. Niranjana Devi

Problem Definition

In reversible data hiding techniques, the values of host data are modified according to some particular rules and the original host content can be perfectly restored after extraction of the hidden data on receiver side. In this paper, the optimal rule of value modification under a payload-distortion criterion is found by using an iterative procedure, and a practical reversible data hiding scheme is proposed.

Finding

The secret data, as well as the auxiliary information used for content recovery, are carried by the differences between the original pixel-values and the corresponding values estimated from the neighbors. Here, the estimation errors are modified according to the optimal value transfer rule. Also, the host image is divided into a number of pixel subsets and the auxiliary information of a subset is always embedded into the estimation errors in the next subset.

Conclusion

A receiver can successfully extract the embedded secret data and recover the original content in the subsets with an inverse order. This way, a good reversible data hiding performance is achieved.

III THEORETICAL BACKGROUND

3.1 PROBLEM IDENTIFICATION

- Many RDH-EI methods were proposed in the existing. Most methods were designed for uncompressed images, while some works were for JPEG bit streams.
- Existing scheme begins with a JPEG encryption algorithm, in which the appended bits of Huffman codes are encrypted by a stream cipher, and all Huffman codes are kept unchanged. After encryption, the JPEG file size is preserved and the format is compliant to JPEG decoders. On cloud, the server selects the encrypted bit streams of some blocks as candidates.
- Additional bits are encoded by LDPC-based error correction codes (ECC) and embedded into the useful candidate bit stream by flipping the LSBs of the encrypted appended bits of the AC coefficients in each candidate block.
- After the authorized user downloads and decrypts the marked encrypted bit stream, LSBs of the appended bits of useful candidates are flipped again to estimate the additional bits using a predefined blocking artifact function and an ECC decoder.

DISADVANTAGES OF EXISTING SYSTEM

- RDH-EI is for uncompressed images. However, those methods are not useful ϖ in many applications because most images transmitted over Internet are compressed, e.g., the popular JPEG.
- The preprocessing based methods can achieve much better embedding rates, but require extra RDH operations before image encryption.
- There exists one problem that the recipient must do a recovery task after decryption. This recovery burden on the recipient side limits the real application of RDH-EI techniques, since users always hope to do nothing except image encryption and decryption

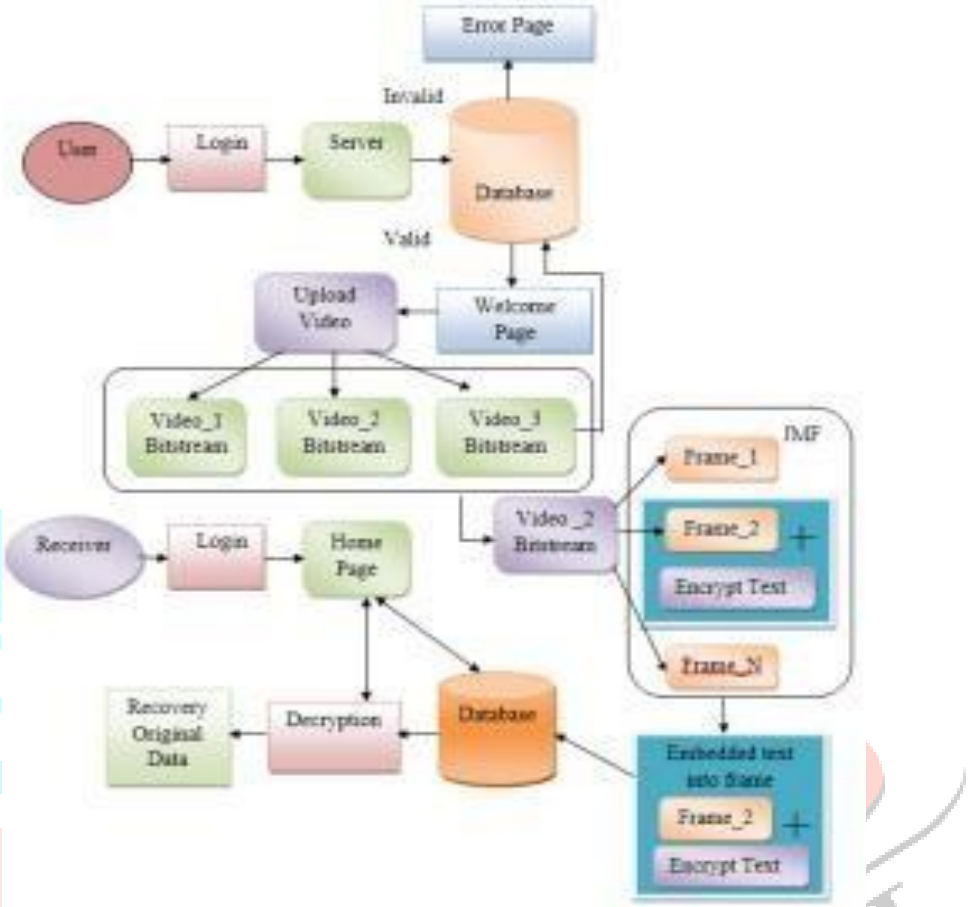
3.2 PROBLEM SOLVING

- This paper focuses on RDH-EI in JPEG bit streams. We propose a novel RDH-EI framework to make this technique more practical.
- The proposed RDH-EI framework focuses on labeling the encrypted JPEG images on cloud storage. There are three par-ties, including the image owner, the cloud server and the authorized user. The owner encrypts a JPEG bit stream and up-loads it to the cloud.
- The cloud server embeds additional messages into the encrypted bit stream to generate a marked encrypted bit stream. The hidden messages can be extracted from the marked encrypted bit stream.
- When an authorized requires a downloading operation, the server recovers the original encrypted bit stream. After decryption, the user obtains the original JPEG image.

ADVANTAGES OF PROPOSED SYSTEM

- Compared with previous works, two achievements are achieved in the proposed method. First, the tasks of data em-bedding, data extraction and bit stream recovery are all done by the server.
- There are no extra computation burdens for owner/user except encryption/decryption. Second, we achieve a larger embedding payload than state-of-the-art works on JPEG RDH-EI.
- This paper proposes a new JPEG RDH-EI framework, in which no recovery task is required for the owner or the user.

3.3 SYSTEM ARCHITECTURE



IV SYSTEM IMPLEMETATION

4.1. USER INTERFACE DESIGN

The User Interface Design plays an important role for the user to move login the Application. This module has created for the security purpose. In this login page we have to enter user name and password, it will check username and password, if valid means directly go to home page, invalid username or password means show the error message and redirect to registration page. So we are preventing from unauthorized user entering into the login page to user page. It will provide a good security for our project.

4.2. UPLOAD VIDEOS

In this module user is going to access the application after successful login, he will give the input for video and text message. In this module we will try to split the video in to number of different parts, and as for our convenience we will consider the 1st part of the splitted video and extract its STREAMS. If it is transmitted like this technique with side information then it is easy to retrieve the video with same quality.

4.3. SPLIT THE VIDEO INTO FRAMES

In this module we receive the splitting video from the database. Receive the video for extracting frames. Extracting frames by using Java Media Framework (JMF). And then every video must be converted in to frames by using the JMF. Every frame from the splitting videos must be stored in database for frame indexing. Thus, the module used for extracting videos in to frames.

4.4. WATER MARKING

This is fourth module of our project in this module first we encrypt plain text into Encrypted (cipher) format using the RDH-EI technique, after encryption is done the cipher text is embedded in to the user selected frame. After embedding encrypted text into frame reconstruct the all frame into single video and send to the destination.

4.5 DECRYPT THE FRAME

In this module we going to decrypt the text and showing the original message to the receiver. First we receive the video, to split the video into frame and chose the encrypted frame and extract the encrypted text into frame. After extract the encrypted text we decrypt the text into original format using the Reversible data hiding in encrypted images (RDHEI).

V CONCLUSION & FUTURE WORK

5.1 CONCLUSION

This paper proposes a new framework of reversible data hiding in encrypted JPEG bit streams. A JPEG encryption and decryption algorithm is developed to hide the content of the original image. The encryption is format-compliant to the popular JPEG decoders. On the cloud side, the servers can em-bed a large amount of additional bits into the encrypted bit-stream. We propose to do the embedding using a combination of code mapping and ordered embedding. Once an authorized user requires a downloading operation, the server extracts the additional message and recovers the original encrypted bit-stream. Since the recovery has been done before downloading, the user obtains an image exactly the same as the original. The proposed framework outperforms the previous JPEG RDH-EI frameworks on three aspects. First, the proposed em-bedding method provides a much larger payload than the other methods. Second, the proposed framework frees the computation burdens on both the owner and the user sides. While pre-processing or post-processing is required in the traditional works, the propose framework requires the owner or the user to no extra tasks except encryption or decryption. Finally, the proposed JPEG encryption algorithm is also secure against the cipher text-only attack.

5.2 FUTURE ENHANCEMENT

The proposed framework outperforms the previous JPEG RDH-EI frameworks on three aspects. First, the proposed embedding method provides a much larger payload than the other methods. Second, the proposed framework frees the computation burdens on both the owner and the user sides. While pre-processing or post-processing is required in the traditional works, the propose framework requires the owner or the user to no extra tasks except encryption or decryption. Finally, the proposed JPEG encryption algorithm is also secure against the cipher text-only attack.

REFERENCE

[1] 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.

[2] W. -L. Tai, C. -M. Yeh, and C. -C. Chang, "Reversible data hiding based on histogram modification of pixel differences," IEEE Transactions on Circuits and Systems for Video Tech-nology, vol. 19, no. 6, pp. 906–910, 2009.

[3] X. Zhang, "Reversible data hiding in encrypted images," IEEE Signal Processing Letters, vol. 18, no. 4, pp. 255–258, 2011.

[4] W. Hong, T. Chen, and H. Wu, "An improved reversible data hiding in encrypted images using side match," IEEE Signal Processing Letters, vol. 19, no. 4, pp. 199– 202, 2012.

[5] X. Liao, and C. Shu, "Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels," Journal of Visual Communication and Image Repre sentation, vol. 28, pp. 21–27, 2015.

[6] Z. Qian, S. Dai, F. Jiang, an d X. Zhang, "Improved joint reversible data hiding in encrypted images", Journal of Visual Communication and Image Representation, vol. 40, pp. 732-738, 2016.

[7] J. Zhou, W. Sun, L. Dong, et al. "Secure reversible image data hiding over encrypted domain via key modulation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 26, no. 3, pp. 441-452, 2016.

[8] X. Zhang, "Separable reversible data hiding in encrypted image," IEEE Transactions on Information Forensics Security, vol. 7, no. 2, pp. 826–832, 2012.

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

[10] X. Zhang,"Reversible data hiding with optimal value transfer" IEEE Trans. Multimedia, vol. 15, no. 2, pp. 316-325, 2013.

[11] T. Bianchi, A. Piva, and M. Barni," On the implementation of the discrete Fourier transform in the encrypted domain" IEEE Trans. Inf. Forensics Secur., vol. 4, no. 1, pp. 86-97, 2009.

[12] F. Huang, J. Huang, and Y. Q. Shi, "New framework for reversible data hiding in encrypted domain". IEEE Transactions on Information Forensics and Security, vol. 11, no. 12, pp. 2777-2789, 2016.

[13] K. Ma, W. Zhang, X. Zhao, et al. "Reversible data hiding in encrypted images by reserving room before encryption," IEEE Transactions on Information Forensics Security, vol. 8, no. 3, pp. 553-562, 2013.

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

[15] X. Cao, L. Du, X. Wei, et al. "High capacity reversible data hiding in encrypted images by patch-level sparse representation," IEEE Transactions on Cybernetics, vol. 46, no. 5, pp. 1132-1143, 2016.

[16] M. Fujiyoshi, "Separable reversible data hiding in encrypted images with histogram permutation," in Proceeding IEEE International Conference on Multimedia and Expo, pp. 1-4, 2013.

[17] W. Zhang, H. Wang, D. Hou, and N. Yu, "Reversible data hiding in encrypted images by reversible image transformation," IEEE Transactions on Multimedia, vol. 18, no. 8, pp. 1469-1479, 2016.

[18] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted JPEG bit stream," IEEE Transactions on Multimedia, vol. 16, no. 5, pp. 1486-1491, 2014.

[19] J. C. Chang, Y. Z. Lu, and H. L. Wu, "A separable reversible data hiding scheme for encrypted JPEG bit streams," Signal Processing, vol. 133, pp. 135-143, 2017.