iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 4 **Issue:** VIII **Month of publication:** August 2016

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Data Hiding in Encrypted Images by using Huffman coding algorithm

R. Kalaivani[1], E. Mythili[2], G. Pathma Prabha[3], K.M. Niveditha.[4], Project Guide: Ms.V.Parameshwari[5] M.E.,M.B.A,(AP/ECE)
Department of Electronics And Communication Engineering, Nandha Engineering College, Erode.

Abstract: ***It is used for privacy, security, and protection. As the entropy of encrypted images is maximized, it is difficult to losslessly vacate room after encryption using the existing methods. In the proposed work, security and authentication can be ensured using the Huffman Coding Algorithm In this proposed system the original image is encrypted using encryption key and the data is embedded using data hiding key then the image & data is decrypted in the receiver side using respective keys. we propose to consider the patch-level sparse representation when hiding the secret data. a large vacated room can be achieved, and thus the data hider can embed more secret messages in the encrypted image.***

Index Terms—Image encryption, reversible data hiding (RDH),Huffman coding

I. INTRODUCTION

REVERSIBLE data hiding (RDH) in images aims to exactly recover both the embedded secret information and the original cover image. It has attracted intensive research interests. Military, medical and legal scenarios are its typical examples, in which even a slight distortion is not tolerable. Many RDH algorithms have already been developed, such as image compression-based [1], [2], difference expansion- based [3]–[7], histogram shift (HS)-based [8]–[11], image pixel pair based [12], [13], and dual/multi-image [14], [15] hiding methods. Recently, due to the requirement of privacy protection [16], [17], the cover owner usually encrypts the original content before transferring it to the data manager. Meanwhile, the data manager may want to embed addi- tional messages into the encrypted image for authentication or steganography [18], even though the content of the orig- inal image is unknown to him. In this situation, hiding data in the encrypted image is an intuitive and effective way to meet such requirement. To hide data in encrypted domains, some digital watermarking [19], [20]-based schemes are pro- posed. Besides, the commutative watermarking and ciphering schemes for digital images are introduced in [21] and [22]. Although the methods mentioned above have provided promis- ing encrypted domains, they are insufficient for more sensitive military and medical scenarios, where the image content should be not only kept secret strictly, but also be losslessly recovered after data extraction. Therefore, RDH in encrypted images (RDHEIs) is desirable. To this end, many RDHEI schemes have been proposed in past years. One of the common techniques is based on manipulating the least-significant-bit (LSB) planes by directly replacing the three LSBs of the cover-image with the message bits, which is kind of the pixel-level compressive methods essentially. In [23], the encrypted image is segmented into a number of nonoverlapped blocks, while each block is divided into two sets. Each block carries one bit by flipping three LSBs of a set for predefined pixels. Hong et al. [24] gave an improved version based on [23]. Specifically, they fully har- ness the pixels in calculating the smoothness of each block and consider the pixel correlations in the border of neigh- boring blocks. The resulting error rate of extracted-bits is thereby decreased. In [25], the proposed method creates a sparse space to accommodate some additional data by com- pressing the LSBs of the encrypted image. It is hard to squeeze room by only considering three LSBs of the encrypted images. Instead, Zhang et alThis article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination. Reversible data hiding (RDH) in images aims to exactly recover both the embedded secret information and the original cover image. Due to the requirement of privacy protection the cover owner usually encrypts the original content before transferring it to the data manager. Meanwhile, the data manager may want to embed additional messages into the encrypted image for authentication or steganography, even though the content of the original image is unknown to him.The Huffman Coding algorithm is used to embedded the encryption image and hiding the data in order to secure the information.Based on the key the owner have they can extract the encrypted image alone or Hide data alone or both encrypted image and hide data.

©IJRASET: All Rights are Reserved

Volume 4 Issue VIII, August 2016
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

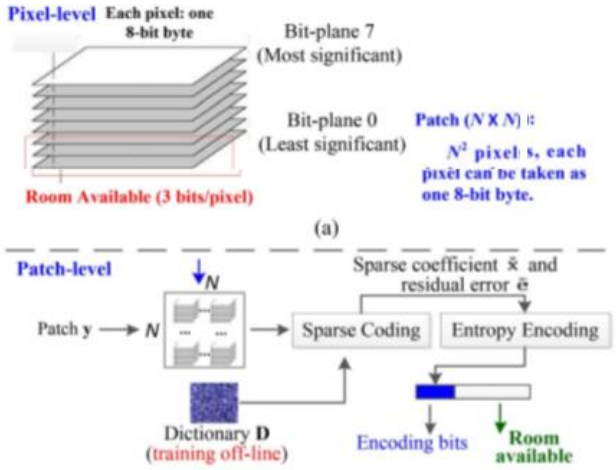
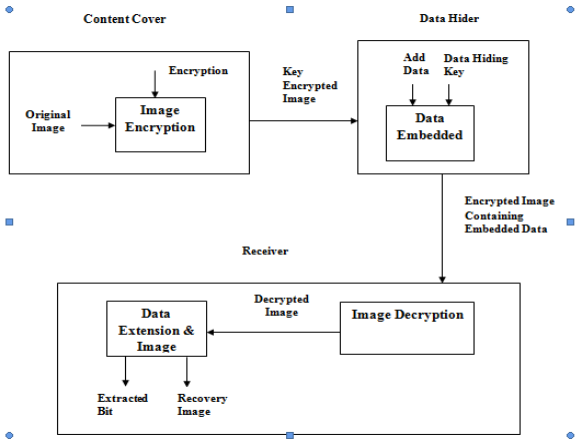


Fig. 1. Comparison between pixel-level and patch-level representation for room preserving. Take one patch with size N×N for example. (a) For pixel level representation, the number of LSB for one pixel can be set as 3 for better imperceptibility. Therefore, one patch with size N ×N can hide 3N2 bits (3 bits per pixel). In contrast, (b) for patch level representation, by using an over-complete dictionary D (training offline for sparse representation), one patch y can be represented as a sparse combination of several atoms in the dictionary. That is, y = round(D˜ x)+˜ e. Thanks to the representation power of sparse coding, only a small number of coefficients ˜ x and residual error ˜ e require space to record. Thus, a higher capacity room is available. For more details, please see our proposed method described in Section III. the space to carry the data. To further improve the compres- sion ratio, Yin et al.[27] selected the smooth blocks in the encrypted image, and embed the additional data into the blocks in a sorted order with respect to block smoothness by using local HS. Although the methods in [23]–[27] divide the image into patches or groups, the preserved spaces are all acquired by using the LSB modification or compression. As the entropy of encrypted images is maximized, it is difficult to loss- lessly vacate room after encryption (VRAE) using the above methods. To overcome this drawback, the methods of reserv- ing room before encryption (RRBE) are proposed [28], [29]. In [28], a large portion of pixels are utilized to estimate the rest before encryption, the additional data is embedded in the encrypted image by operating the estimating errors. In [29], the reserving room is obtained by embedding LSBs of some pixels into other pixels. The spare space emptied out is three LSBs of the selected pixels. Compared with the VRAE methods, RRBE methods have shown a superior performance. The success of the above RDHEI methods has verified that the data hiding can be accomplished by exploiting the redundancy within the image. However, for better impercep- tibility, only three LSBs can be used for data hiding. one patch with size N × N can hide 3N2 bits (3 bits per pixel). Actually, for numerous computer vision applications, the image can be analyzed at the patch level rather than at the individual pixel level. Patches contain contextual information and have advantages in terms of com- putation and generalization. Specifically , because the pixels in certain ranges (like patches or regions) are of strong similar- ity, the information in any image is correlated in a certain way, especially within a limited local searching range [30]. They can be heavily compressed, and thus can result in a large hiding room. Considering the two aspects, to better exploit the correlations of neighbor pixels, we propose a novel method for high capacity separable reversible data hiding in encrypted images (HC_SRDHEI). As shown in Fig. 1(b), we follow the framework of RRBE and propose to consider the mid-level visual representation with image patches. Using an overcom- plete dictionary D that contains prototype signal atoms, the image patch y is described by sparse linear combinations of these atoms. That is, only a small number of coefficients ˜ x and the corresponding residual error ˜ e caused by sparse rep- resentation, require space to record. Thus, a higher capacity room is available. shows the flowchart of the proposed HC_SRDHEI method. For the content owner, the given cover image is rep- resented according to an over-complete dictionary by sparse coefficients. After that, for the given selected patches, the corresponding coefficients and reconstructed residual errors are encoded directly without quantization. For most of the patches, the data size is well reduced in the basis of coeffi- cient representation, thus the vacated room is preserved for high capacity data hiding after image encryption. Note that, for losslessly recovering the cover image, the residual errors are self-embedded into the nonselected patches. The learned dictionary, is also embedded into the encrypted image for further use. At the receiver side, when the receivers accept an encrypted image containing additional data, the processing procedure depends on the role of receiver. If the receiver is a data hider and only has the data hiding key, he can extract the data without knowing the image content. If the receiver is an image owner and only has the encryption key, he can decrypt the image with a better quality. If the receiver is both the

©IJRASET: All Rights are Reserved

image owner and data hider, he has both of two keys in such case. Thus, the data extraction and content recovery are all done, and both results are free of error. In summary, for our proposed framework, the data extraction and image recovery are separable and reversible. The contributions of this paper are summarized as follows. 1) We present a patch-based RDHEI scheme. Although some other methods also divide the cover image into patches to perform RDHEI, their definition is mainly to consider the correlation of pixels within the patch. Therefore, they are kind of the pixel-level compressive methods essentially. In contrast, we regard the patch as a whole, and represent them using a small number of coefficients, which is beyond the traditional pixel-level case. And thus a high capacity room is available. 2) Our scheme obtains a significant performance improve- ment over the traditional RDHEI methods. For a cer- tain embedding rate, the visual quality of the directly decrypted image is improved. Meanwhile, for an accept- able peak signal to noise ratio (PSNR), for instance PSNR = 40 dB, the range of embedding rates is greatly enlarged. Experimental results demonstrate that our average maximum embedding rate (MER) reached 1.7 times as large as that of the previous best alternative method conducts. The remainder of this paper is organized as follows. Section II briefly reviews the related work.



This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

In the receiver side, we consider three cases depending on whether the receiver has data hiding and/or encryption keys in our framework. Case 1: if a receiver has the data-hiding key, he can extract the data without knowing the image content. Case 2: if the receiver has both of two keys, the data extraction and content recovery are free of error. Case 3: if the receiver has the encryption key, he can decrypt the image with a better quality. Note that there are two keys in our framework. The encryption key is used for encryption, which can be taken as a security measure that turns image into an unreadable cipher. The data hiding key is used to encrypt the hidden data. Anyone who does not possess the data hiding key could not extract the embedded data.

method is detailed in Section III. Section IV reports the extensive experimental results and comparisons to the state-of- the-art methods. Finally, this paper is concluded in Section V.

II. RELATED WORK

Recently, a series of RDHEI schemes have been designed. Generally, these RDHEI methods can be grouped into two categories. The first one is to VRAE, and the second one is to RRBE. For VRAE, some classic methods can be found in [23]–[27]. For RRBE, two major proposed methods are [28] and [29]. Next, we will give a detailed introduction about them. In [23], the original image cover is first encrypted, and then secret data are embedded by modifying a small proportion of the encrypted image. The receiver first decrypts the encrypted image, and then extracts the embedded data and recovers the original cover image based on the decrypted version. The limitation of this method is that, each block is embedded only one bit payload. Moreover, if the block size is relatively small, the error bits of data extraction increase. To over- come this shortcoming, an improved RDHEI method using side match is proposed by Hong et al. [24]. First, it fully exploits pixels when evaluating the smoothness by summing vertical and horizontal differences in image blocks, which leads to a better estimation of the smoothness of blocks for data extraction and image recovery. Second, it adopts the side match technique to concatenate the borders of the recovered blocks to the unrecovered blocks. The error rate obtained by [24] is much lower than [23]. However, another This article has been accepted for inclusion in a future

©IJRASET: All Rights are Reserved

issue of this journal. Content is final as presented, with the exception of pagination.

III. IEEE TRANSACTIONS ON CYBERNETICS

property for RDHEI, i.e., separability, is not taken into account [23], [24]. That is, the data extraction should be separable from the content decryption. From this point of view, Zhang [25] proposed a novel scheme for separable RDHEIs. The content owner encrypts the original uncom- pressed image using an encryption key, and then the data hider compresses the LSBs of the encrypted image using a data hiding key to create a sparse space to accommodate some additional data. In the receiver side, three cases that the receiver has encryption and/or data hiding keys are consid- ered. Zhang et al.[26] proposed a novel scheme of RDHEIs based on lossless compression of encrypted data by using LDPC code. The data hider compresses a half of the fourth LSB in the cipher-text image, and inserts the compressed data and the additional data into the half of the fourth LSB using efficient embedding method. Moreover, Yin et al. [27] proposed a RDHEI scheme which also offers error-free data extraction. The cover image is partitioned into nonover- lapping blocks and multigranularity encryption is applied. The data hider selects the several smoother blocks for data embedding. Since the entropy of the image has been maximized due to encryption, the room vacating is more difficult than the cover image. Thus the schemes in [23]–[27] can only achieve small payloads even with the advance of compressing encrypted images [31], [32]. The MER in [23]–[27] are all less than 0.2 bits per second. Since losslessly vacating room from encrypted images is relatively difficult and sometimes inef- ficient, the RRBE methods [28], [29] are proposed. Instead of embedding data in encrypted images directly, Zhang et al.[28] proposed to estimate some pixels before encryption, and then the additional data are embedded in the estimating errors. Moreover, Ma et al. [29] designed an effective scheme by RRBE. They first empty out room by embedding LSBs of some pixels in one region into another region via a traditional RDH method, and then encrypt the image. As a result, the positions of these LSBs in the first region of encrypted image can be used to hide data. This method can separately extract hidden data and decrypt the image. Moreover, they can embed more than ten times as large payloads as those of previous methods discussed above. However, the spare space emptied out is limited to at most three LSB-planes per pixel. Therefore, the MER is only about 0.5 bits per second in [29]. The proposed HC_RDHEI method inherits the merits of RRBE based on patch level sparse representation. Not only does the proposed method separate the data extraction from image decryption but also achieve excellent performance. Moreover, different from the above methods mainly consid- ering pixel-level compressive property, our scheme takes the patch as a whole, and represents them using sparse coding. As a result, a high capacity is achieved.

IV. PROPOSED METHOD

In this section, we give a detailed introduction about our HC_SRDHEI in the following three aspects: 1) encrypted image generation; 2) data hiding in the encrypted image; and 3) data extraction and image recovery. For simplicity, we use the grayscale images with 8 bits per pixel. The extension from gray images to color images is straightforward.

A. Encrypted Image Generation For the image owner, to generate encrypted images, three phases: 1) sparse representation; 2) self-reversible embedding; and 3) stream encryption, are involved. Given a cover image, we first divide it into patches that are then represented accord- ing to an overcomplete dictionary via sparse coding. Then, the smoother patches with lower residual errors are selected for room reserving. These selected patches are represented by the sparse coefficients, and the corresponding residual errors are encoded and reversibly embedded into the other nonselected pathes with a standard RDH algorithm. Finally, the room pre-served and self-embedded image is encrypted to generate the finally version. 1) Sparse Representation: For reserving room to hide data, we train the dictionary based on K-means singular value decomposition (K-SVD) algorithm [33], [34], which is widely used for designing over-complete dictionaries that lead to sparse signal representation. Note that, the K-SVD training is an offline procedure, and the corresponding dictionary pro- duced by training is then considered fixed for the whole RDH procedure. Given a cover image I with size N1 × N2, we first divide it into a bunch of nonoverlapped N ×N patches. Unless stated, the patch size N is set to 4 as default in our algorithm. Denote S as the number of patches of I, and S = N1 ×N2/N ×N. For each patch, the pixel values are vec- torized as yi ∈

R n×1 (i = 1,2,...,S, and n = N2). Therefore,the image I ∈ R n×S, which contains S column vectors {yi}S i=1.Using an overcomplete dictionary matrix D ∈R n×K(K > n) that contains K prototype signal atoms for columns, {dj}K j=1,every image patch yi can be represented as a sparse linear combination of these atomsmin xi ||yi −Dxi||2 2 subject to ||xi||0 ≤ L (1) where ||·||0 is the l0 norm, counting the nonzero entries of a vector. L is a predetermined number of nonzero entries. The coefficient vector xi ∈R K×1 contains the representation coef- ficients of yi, and is expected to be sparse. Note that once the well trained dictionary D is obtained, it can be used for any cover image. The computational complexity analysis for training and the training time will be discussed in Section IV.

©IJRASET: All Rights are Reserved

Actually, the approximation of yi using Dxi needs not to be exact, and could absorb a moderate error. In other words, the representation of yi may be either exact yi = Dxi or approx- imate, yi ≈ Dxi. This suggests an approximation that trades off accuracy of representation with its simplicity. Therefore, we make an error correction step for lossless image recovery. Moreover, the sparse coefficients xi are adjusted to integers ˜ xi = round(xi) for the convenience of encoding. Therefore, yi can be reconstructed by yi = roundD˜ xi————————————————+˜ ei (2) where i = 1,2,...,S, ˜ xi ∈
Z K×1, and ˜ ei ∈
Z n×1. Here, ˜ ei is considered as the residual error, which contains two parts: 1) the reconstructed error caused by sparse coding and 2) theThis article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination

V. HUFFMAN CODING

The idea behind Huffman coding is to find a way to compress the storage of data using variable length codes. Our standard model of storing data uses fixed length codes. For example, each character in a text file is stored using 8 bits. There are certain advantages to this system. When reading a file, we know to ALWAYS read 8 bits at a time to read a single character. But as you might imagine, this coding scheme is inefficient. The reason for this is that some characters are more frequently used than other characters. Let's say that the character 'e' is used 10 times more frequently than the character 'q'. It would then be advantageous for us to use a 7 bit code for e and a 9 bit code for q instead because that could shorten our overall message length.Huffman coding finds the optimal way to take advantage of varying character frequencies in a particular file. On average, using Huffman coding on standard files can shrink them anywhere from 10% to 30% depending to the character distribution. (The more skewed the distribution, the better Huffman coding will do.)The idea behind the coding is to give less frequent characters and groups of characters longer codes. Also, the coding is constructed in such a way that no two constructed codes are prefixes of each other. This property about the code is crucial with respect to easily deciphering the code. The algorithm used in this process for providing security and authentication is Huffma Coding Algorithm.The idea behind Huffman coding is to find a way to compress the storage of data using variable length codes.The idea behind the coding is to give less frequent characters and groups of characters longer codes. Also, the coding is constructed in such a way that no two constructed codes are prefixes of each other. This property about the code is crucial with respect to easily deciphering the code.For Huffman coding we need to create a binary tree for each character that also stores the frequency with which it occurs.

VI. BUILDING A HUFFMAN TREE

The easiest way to see how this algorithm works is to work through an example. Let's assume that after scanning a file we find the following character frequencies:

Character	Frequency
'a'	12
'b'	2
'c'	7
'd'	13
'e'	18

Now, create a binary tree for each character that also stores the frequency with which it occurs.The algorithm is as follows: Find the two binary trees in the list that store minimum frequencies at their nodes. Connect these two nodes at a newly created common node that will store NO character but will store the sum of the frequencies of all the nodes connected below it.

VII.DICTIONARY ENCODING SIZE (NA)WITH DIFFERENT PARAMETERS K AND L SETTING (BIT)

Finally, these selected patches {yk}C k=1 are represented by the sparse coefficients. The room for parameter bits, dictio- nary bits and hidden data are filled with random bits. The parameter bits and dictionary bits are set after encryption, and the vacated room is

©IJRASET: All Rights are Reserved

preserved for data hider. For lossless image recovery, the corresponding residual errors {˜ ek}C k=1 are reversibly embedded into the other nonselected pathes, which construct the area B, with a standard RDH algorithm [37]. For simplicity, the size of area B is computed by NB1×NB2, where NB1 = N ×floorS−C N2/N NB2 = N2. (10) Thus, the cover image is converted into its room preserved and self-embedded version Ic. The illustration of image partition and reversible self-embedding process is shown in Fig. 5. Note that, this step does not rely on any specific RDH algorithm. 3) Image Encryption: For the room preserved self- embedded image Ic, we generate the encrypted image Ie by a stream cipher, such as RC4 or data encryption standard in cipher feedback mode [38]. Denote the eight bits of the pixel pi,j(i = 1,2,...N1,j = 1,2,...N2) as bi,j,0, bi,j,1, bi,j,2, bi,j,3, bi,j,4, bi,j,5, bi,j,6, and bi,j,7. Thus bi,j,k =pi,j/2mmod 2, m = 0,1,...,7. (11)

Then, the encrypted bit stream can be expressed as b i,j,m = bi,j,m ⊕ri,j,m, m = 0,1,...,7 (12) where ri,j,m is a pseudo-random bit generated by the encryp- tion key Ke using the stream cipher. After encryption, we set the parameter bits in the selected patches {yk}C−1 k=1 to inform the data hider the positions of thenext patches {yk}C k=2 they can embed. As shown in Fig. 4, the data hiders are able to access each embedded patch by traversing this list structure. Note that, the end mark is set in the last selected patch yC with nb bits zeros. As for the well trained dictionary, its corresponding encoding bits na, are encrypted with Ke and also embedded into the corresponding reserving room. After that, the position of the first selected patch y1 and the vacated room size nd for data hiding, are also embedded by RDH algorithm proposed in [37]. Here, the position is encrypted and can be decrypted either by encryption key Ke or data hiding key Kd. In contrast, the data hiding size nd is encrypted and can only be decrypted by data hiding key Kd. Finally, the encrypted image Ie is obtained.

B. Data Hiding in Encrypted Images Once the encrypted image is received, the data hider can embed secret data for management or authentication requirement. The embedding process starts with locating the encrypted version of area A. Since the image owner has embedded the position of the first room preserving patch and the room size for each patch in the encrypted image, it is effortless for the data hider to know where and how many bits they can modify. After that, the data hider scans each selected

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

MER =

C×8N2 −L(np +nv)−nb———————————————————————−na N1 ×N2

where na is the dictionary size and is fixed for our algorithm. After data hiding, the position of the first data hiding patch and the hiding room size for each patch are also embedded into the encrypted image containing additional embedded data with RDH algorithm. Note that, the secret data is encrypted according to the data hiding key Kd before hiding.

C. Data Extraction and Image Recovery With the encrypted image containing additional embedded data, the receiver faces three situations depending on whether the receiver has data hiding and/or encryption keys. The data extraction and image decryption can be processed separately. 1) Data Extraction With Only Data Hiding Key: For the receiver who only has data hiding key Kd, he first extracts and computes the starting position and the hiding room size for each patch and divides the received image into nonoverlapped N ×N patches. Then, data extraction is finished by checking the last nd bits for the selected patches in the received image. After that, all original hidden data are extracted and recovered with the data hiding key Kd. The extracted data is lossless. Moreover, the receiver does not access to image content in the entire extraction process. 2) Image Decryption With Only Encryption Key: In this case, the receiver has the encryption key Ke only. After extrac- tion the position of the first selected patch by RDH algorithm, all the selected patches are identified one by one. Moreover, the dictionary D is also obtained by extraction. After patch segmentation of the received image, the decryption procedure is performed and it includes two cases: 1) unselected patch decryption and 2) selected patch decryption. For unselected patch, the content can be directly decrypted according to the encryption key bi,j,m = b i,j,m ⊕ri,j,m, m = 0,1,...,7 (14) where ri,j,m is the pseudo-random bit generated by the encryp- tion key Ke. b i,j,m and bi,j,m are the encrypted bit and the decrypted bit for the pixel pi,j, respectively. Consequently, the unselected patch decryption is losslessly achieved. For the selected patch, we first decrypt the encoded bits by (14) based on encrypting Ke. Then, the position and value of the sparse coefficients for each selected patch {yk}C k=1 are determined. After that, the corresponding coefficient, denoted as ˜ xk, are obtained. Then, the decrypted patch yd k is computed via yd k = roundD˜ xk——————————————————— (15) where k = 1,2,...,C, D is the trained dictionary. Since both the unselected and selected patches are decrypted, the image decryption in our proposed method is completed.

©IJRASET: All Rights are Reserved

VIII. TRAINING TIME WITH DIFFERENT PARAMETERS K AND L SETTING (HOURS)

3) Data Extraction and Image Recovery With Both Data Hiding and Encryption Keys: If the receiver has both the data hiding key Kd and encryption key Ke, the data extraction and image recovery can be achieved On the one hand, with the data hiding key K one can extract the hidden secret data without any error. On the other hand, with the encryp- tion key Ke, they first perform directly image decryption, then the corresponding coefficient for selected patches{yk}C k=1,denoted as ˜ xk, are obtained. After that, the residual errors ˜ ek, are extracted from the nonselected patches (corresponding to area B). Therefore, the recovery patch yr k is computed as yr k = roundD˜ xk————————————————+˜ ek (16) where k = 1,2,...,C. As the patch recovery is based on the lossless coefficients and residual errors, there exists no errors for the selected patches. Moreover, thanks to the RDH algorithm, the nonselected patches are also recovered loss- lessly after residual errors extraction. That is to say, the image recovery in our proposed method is free of any error.

IX. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we conduct several experiments to evaluate the proposed algorithm, which include: choice of dictionary parameters, image encoding, and performance analysis on public available standard images. After that, the particular comparisons between our method and the most state-of-the-art competitor [29] on Kodak,1 BOSSBase [39], PASCAL [40], and Holidays [41] are described. Finally, we make the corre- sponding computational complexity analysis.

A. Choice of Dictionary Parameters Our dictionary training is based on 786432 patches with size 4 × 4 taken from 48 standard 8-bit grayscale images in the University of Southern California, Signal and Image Processing Institute image database. For the training process, we adopt K-SVD as the trainer. In our implementation, the maximal number of iterations, T, of K-SVD is set to 50. The experiment PC configuration is: Intel i7-2600, CPU 3.40 GHz, 10 GB RAM. The training time for different K and L is shown in Table II. The output dictionary has the size of 16×K. The coefficients are computed using orthogonal match- ing pursuit (OMP) with a fixed maximal number of nonzero coefficient L. K and L are selected by referring to the perfor- mance comparison of our algorithm. For making a best choice of dictionary parameters which represent the higher embed- ding rate, we compute the average position bits (¯ np), value bits (¯ nv), and error bits (¯ ne) for all the patches in the training set. The results is shown Note that, to reduce the effect of random samples,

1http://www.r0k.us/graphics/kadak

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

The textures for these four images (Airplane, Man, Lena, andBaboon) range from smooth to complex.aspects, for the acceptable PSNR, such as PSNR 40 dB, the range of ER is much wider. As for theThis article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

X. IEEE TRANSACTIONS ON CYBERNETICS

ER comparison with the closest competitor [29] (PSNR = 40 dB) on 100 images randomly selected from three image datasets, i.e., BOSSBase [39], PASCAL [40], and Holidays [41], respectively. The image indices are obtained based on sorting according to image names.PSNR comparison with the closest competitor [29] (ER = 0.35 bits per second) on 100 images randomly selected from three image datasets, i.e., BOSSBase [39], PASCAL [40], and Holidays [41], respectively. The image indices are obtained based on sorting according to image names.four images Lena, Airplane, Man, and Crowd, the MERs of our method are 0.61, 1.0, 0.63, and 1.01 bits per second, while the closest competitor [29] has the MERs of 0.49, 0.63, 0.46, and 0.58 bits per second. The performance analysis implies that our proposed method has a very good hiding capacity. That is mainly because that, in the RRBE method [29], the spare space emptied out is limited to at most three LSB-planes (3N2 bits for each patch). For our scheme, if an image with size 512×512, patch size 4×4, and dictionary parameters K = 64 and L = 2, the hiding bits per patch for this image can give more than 55 bits according to experiment results, while only 48 bits can be preserved in [29].

D. Performance Analysis on Datasets To evaluate the average performance of our proposed method, the performance analysis on image datasets, i.e., Kodak, BOSSBase, PASCAL, and Holidays are involved. Kodak image database contains 25 color images sized

©IJRASET: All Rights are Reserved

512× 768 or 768 × 512. For testing, the color images are trans- formed into gray-level sized 512×512. The comparisons of PSNR in directly decrypted images with different ER are shown in Table IV.Justasshown,theresultsaremostly better than the most state-of-the-art competitor [29]. For the other three database, in order to clearly demonstrate the per- formance, we randomly select 100 images from each dataset as the test sets. The images in BOSSBase are gray-levels with size 512×512. For comparison without loss of generality, we also transform these color images in PASCAL and Holidays into gray levels with the same size. The experimental results of comparison with the state-of-the-art method proposed in [29] are shown in Figs. 11 and 12.Actually,forRDHEImethod in [29] and ours, the ERs of both methods vary for differ- ent images. Given PSNR = 40 dB, as shown in Fig. 11,our algorithm has a much wider range of the embedding rate. The average ER of our method for these three datasets are 1.2458, 0.9873, and 0.9258 bits per second, respectively. As for the method proposed in [29], the average ER are 0.6916, 0.5930, and 0.5925 bits per second for the acceptable directly decrypted image quality (PSNR = 40 dB). The experimental results show that our proposed method reaches about 1.7 times as large payloads as the method in [29]. Meanwhile, given ER = 0.35 bits per second, our method obtains better perfor- mance without surprise, as shown in Fig. 12.Theaverage gains of PSNR for three dataset are 5.2963, 3.4312, and 4.2360 dB, respectively. Therefore, we can conclude the sum- marization that the proposed scheme is more suitable for the RDH applications in encrypted images thanks to its higher hiding capacity and better image quality for direct decryption.

E. Computational Complexity Analysis For the computational complexity analysis, we mainly consider the three steps: 1) smooth area selection; 2) self- reversible embedding; and 3) image encryption, for our proposed algorithm and the most state-of-the-art competi- tor [29]. Let the gray-scale image with its size N1 × N2, ˜ N = max(N1,N2). For smooth area selection, the compu- tational complexity for [29] can be denoted by O(˜ N2) +O (˜ N log ˜ N). The first item represents pixels error computing, the second item represents sorting for smooth area selec- tion. In assessing computational complexities for smooth area selection of our method, we should separately discuss the dictionary training and our proposed data hiding algorithmThis article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination. Practically, the training pro- cedure has been done with a nonoptimized MATLAB code on a regular PC. This procedure can be preprocessed by any user on the PC with common configurations. As for our pro- posed scheme, the computational complexity mainly depends on sparse representation (encoding) and selected patch recov- ery used for data hiding (decoding). The encoding adopts the OMP algorithm, the complexity of which is O(P/NNKL) = O(˜ N2KL). Moreover, we also need sort function for final smooth area selection, Therefore, the computational complex- ity of our method is O(˜ N2KL)+O(2˜ N2 log ˜ N). For the other two steps: self-reversible embedding and image encryption, both of [29] and ours adopt traditional RDH algorithm [37] and stream encryption. The computational complexity is nearly the same, which are O(R˜ N2) and O(˜ N2) practically, where R is the number of embedding rounds. Therefore, the overall computational complexity mainly depend on smooth area selection, O(˜ N2) + O(˜ N log ˜ N) for the method in [29], and O(˜ N2KL)+O(2˜ N2 log ˜ N) for ours.

XI. CONCLUSION

Compared to state-of-the-art alternatives, the room vacated for data hiding by our method is much larger used. The data hider simply adopts the pixel replacement to substitute the avail- able room with additional secret data. The data extraction and cover image recovery are separable, and are free of any error. Experimental results on three datasets have demonstrated that our average MER can reach 1.7 times as large as the previous best alternative method provides. The performance analysis implies that our proposed method has a very good potential for practical applications.

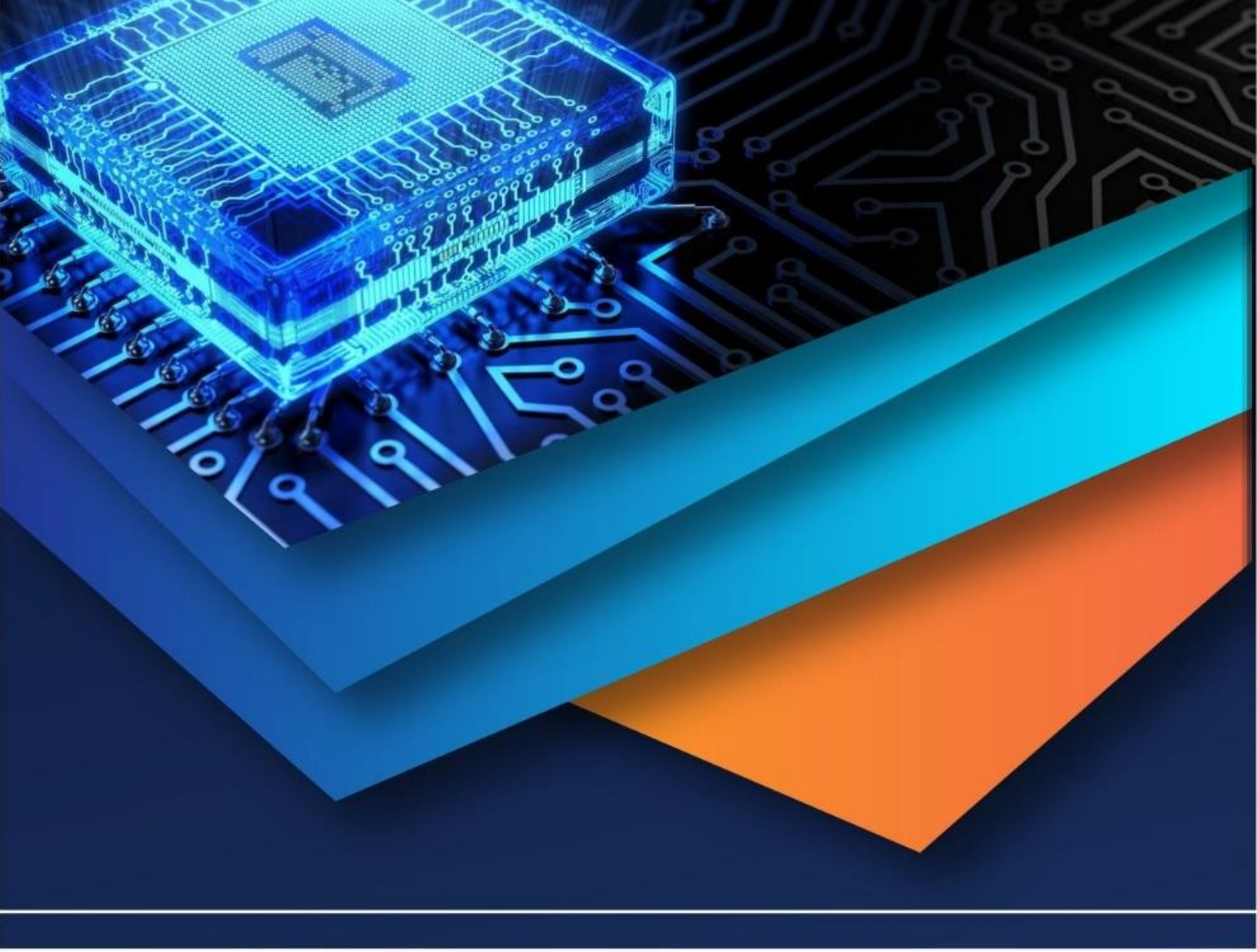
REFERENCES

[1] M. U. Celik, G. Sharma, and A. M. Tekalp, “Lossless generalized- LSB data embedding,” IEEE Trans. Image Process., vol. 14, no. 2, pp. 253–266, Feb. 2005.
[2] J. Fridrich, M. Goljan, and R. Du, “Invertible authentication watermark for JPEG images,” in Proc. Inf. Technol. Coding Comput., Las Vegas, NV, USA, Apr. 2001, pp. 223–227
[3] H. J. Kim, V. Sachnev, Y. Q. Shi, J. Nam, and H. G. Choo, “A novel dif- ference expansion transform for reversible data embedding,” IEEE Trans. Inf. Forensics Security, vol. 3, no. 3, pp. 456–465, Sep. 2008.
[4] D. Coltuc, “Improved embedding for prediction based reversible watermarking,” IEEE Trans. Inf. Forensics Security, vol. 6, no. 3, pp. 873–882, Sep. 2011.
[5] X. Li, B. Ying, and T. 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.
[6] Y. Hu, H. K. Lee, K. Chen, and J. Li, “Difference expansion based reversible data hiding using two embedding directions,” IEEE Trans. Multimedia, vol. 10, no. 8, pp. 1500–1511, Dec. 2008.
[7] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, “Pairwise prediction- error expansion for efficient reversible data hiding,” IEEE Trans. Image Process., vol. 22, no. 12, pp. 5010–5021, Dec. 2013.
[8] W. L. Tai, C. M. Yeh, and C. C. Chang, “Reversible data hiding based on histogram modification of pixel differences,” IEEE Trans. Circuits Syst. Video

©IJRASET: All Rights are Reserved

Technol., vol. 19, no. 6, pp. 906–910, Jun. 2009.
[9] C. C. Lin, W. L. Tai, and C. C. Chang, "Multilevel reversible data hiding based on histogram modification of difference images," Pattern Recognit., vol. 41, no. 12, pp. 3582–3591, Dec. 2008.
[10] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," Signal Process., vol. 89, no. 6, pp. 1129–1143, Jun. 2009.
[11] S. L. Lin, C. F. Huang, M. H. Liou, and C. Y. Chen, "Improving histogram-based reversible information hiding by an optimal weight- based prediction scheme," J. Inf. Hiding Multimedia Signal Process., vol. 4, no. 1, pp. 19–33, Jan. 2013.
[12] S. W. Weng, Y. Zhao, R. R. Ni, and J. S. Pan, "Parity-invariability- based reversible watermarking," Electron. Lett., vol. 45, no. 20, pp. 1022–1023, Sep. 2009.
[13] S. Weng, Y. Zhao, J. S. Pan, and R. Ni, "Reversible watermarking based on invariability and adjustment on pixel pairs," IEEE Signal Process. Lett., vol. 15, no. 20, pp. 721–724, Dec. 2008.
[14] C. F. Lee and Y. L. Huang, "Reversible data hiding scheme based on dual stegano-images using orientation combinations," J. Telecommun. Syst., vol. 52, no. 4, pp. 2237–2247, 2013.
[15] G. Horng, Y. H. Huang, C. C. Chang, and Y. Liu, "(k, n)-image reversible data hiding," J. Inf. Hiding Multimedia Signal Process., vol. 5, no. 2, pp. 152–164, Apr. 2014.
[16] Y. Wang and K. N. Plataniotis, "An analysis of random projection for changeable and privacy-preserving biometric verification," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 40, no. 5, pp. 1280–1293, Oct. 2010.
[17] A. Dabrowski, E. R. Weippl, and I. Echizen, "Framework based on pri- vacy policy hiding for preventing unauthorized face image processing," in Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC), Manchester, U.K., Oct. 2013, pp. 455–461.
[18] Y. T. Wu and F. Y. Shih, "Genetic algorithm based methodology for breaking the steganalytic systems," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 36, no. 1, pp. 24–31, Feb. 2006.
[19] X. Gao, C. Deng, X. Li, and D. Tao, "Geometric distortion insensitive image watermarking in affine covariant regions," IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 40, no. 3, pp. 278–286, May 2010.
[20] M. S. Hsieh and D. C. Tseng, "Image subband coding using fuzzy infer- ence and adaptive quantization," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 33, no. 3, pp. 509–513, Jun. 2003.

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)