qrAs per UGC guidelines an electronic bar code is provided to seure your paper

International Journal for Modern Trends in Science and Technology
Volume 10, Issue 03, pages 108-120.
ISSN: 2455-3778 online
Available online at: http://www.ijmtst.com/vol10issue03.html
DOI: https://doi.org/10.46501/IJMTST1003018

Enhancing Privacy-preserving Image Sharing - A Supervised Machine Learning Approach for Reversible Data Hiding in Encrypted Images with Multiple Data Embedders

D Prabhakar, Dasari Mahathi, Matangi Bala Joseph, Kanala Pavan, Kalluri Venkata Saibabu

Department of Electronics and Communication Engineering, Chalapathi Institute of Technology, Guntur, India.

To Cite this Article
D Prabhakar, Dasari Mahathi, Matangi Bala Joseph, Kanala Pavan, Kalluri Venkata Saibabu,Enhancing Privacy-preserving Image Sharing - A Supervised Machine Learning Approach for Reversible Data Hiding in Encrypted Images with Multiple Data Embedders, International Journal for Modern Trends in Science and Technology, 2024, 10(03), pages. 108-120.https://doi.org/10.46501/IJMTST1003018

Article Info
Received: 30 January 2024; Accepted: 22 February 2024; Published: 01 March 2024.

ABSTRACT

Reversible Data Hiding in Encrypted Image (RDHEI) is a technology for embedding secret information in an encrypted image. It allows the extraction of secret information and lossless decryption and the reconstruction of the original image. This paper proposes an RDHEI technique based on Shamir's Secret Sharing technique and multi-project construction technique. Our approach is to let the image owner hide the pixel values in the coefficients of the polynomial by grouping the pixels and constructing a polynomial. Then, we substitute the secret key into the polynomial through Shamir's Secret Sharing technology. It enables the Galois Field calculation to generate the shared pixels. Finally, we divide the shared pixels into 8 bits and allocate them to the pixels of the shared image. Thus, the embedded space is vacated, and the generated shared image is hidden in the secret message. The experimental results demonstrate that our approach has a multi-hider mechanism and each shared image has a fixed embedding rate, which does not decrease as more images are shared. Additionally, the embedding rate is improved compared with the previous approach.

Keywords:: encrypted image; reversible data hiding; secret sharing; multi-hider mechanism

1. INTRODUCTION

Multimedia security technology is used to prevent unauthorized users from copying, sharing, and modifying media content. To prevent this problem, encryption and information hiding are often used to protect media content. As far as information hiding technology is concerned, traditional information hiding technology will destroy the content of the cover image. However, in some exceptional cases, such as military, medical, and legal document images, the slight distortion of the image is entirely unacceptable. Therefore, whether these images can be completely restored is very important. Reversible data hiding scheme (RDH) can correspond with the requirement of being lossless. RDH methods applied the methodology of changing context to hide the secret data in cover media. After data extracting, the changing context will be fully recovered to the cover media. On the other hand, RDHEI (Reversible Data Hiding in Encrypted Images) technology can combine encryption technology with RDH technology, which can not only hide secret information in the image, but can also encrypt the image to protect the image content. RDH techniques can be broadly classified into three types: (1) Difference Expansion [1–3]: Difference expansion is performed by expanding the difference between adjacent pixels, and then secret information is embedded into the difference. Since the difference will be expanded after the secret information is embedded, this technique will inevitably produce a larger distortion; (2) Histogram Shift [4–6]: through Histogram Shift, the histogram is shifted by the original image or the histogram of the predicted discrepancy, and the empty position after the shift is used to embed the secret information; (3) Lossless Compression [7,8]: The secret information is hidden in the extra space after compressing the original image. Since lossless compression may lead to significant degradation of visual quality, it has received less attention. By and large, the RDH approach is based on the above with some other added strategies.

In this study, we propose a new RDHEI technique where we hide the pixel values into the polynomial coefficients by dividing multiple pixels into a group. First, the image owner confidentially shares the image, generating a shared image and leaving space in each shared image for information embedding. Then, the information recipient can embed a confidential message after receiving the shared image. Finally, the recipient can decrypt the image and remove the hidden information. The recipient needs to obtain at least a Threshold before the original image can be recovered. Our approach enables multiple information hiders and has a more suitable fixed embedding rate. Furthermore, it solves the problem that the embedding rate becomes smaller as the number of shared images becomes larger and successfully improves the embedding rate compared with the previous method.

2. LITERATURE REVIEW

In 2019, Zhao et al. proposed amethod of histogram displacement [9]. First, the secret information is converted into a message with only −1, 0, and 1 by encoding. Then, the middle segment bin is selected for embedding on the histogram displacement. These bins can all be used to embed a −1, 0, or 1 message by shifting them left and right. Finally, the band size is adjusted using the Threshold, and the histogram is created using the prediction error values generated by the Chess Board Prediction method. In 2021, Gao et al. proposed a histogram displacement method for embedding medical images [10]. By dividing medical images into ROI and NROI, the embedding process will stretch the histogram created by the pixel values of ROI to the left and right, which expands the embedding capacity of ROI and enhance the contrast of the image. RDHEI techniques can be divided into two categories: (1) Vacating Room before Encryption, VRBE [11–17]: the original image is first vacated and then encrypted; and (2) Vacating Room after Encryption, VRAE [18–21]: The image is first encrypted, and because the encrypted image retains certain properties, the encrypted image can be hidden. VRBE technology mainly focuses on reversible information hiding or compression of the original image to free up space. It encrypts the image, so there is space to hide information in the encrypted image. For example, in 2013, Ma et al.'s RDHEI method belongs to the technology of Vacating Room before Encryption [11]. They performed reversible information hiding in the original image beforehand and embedded the LSB part of a block into the image. As a result, the encrypted image can use the LSB part of this block to hide data. In 2018, Puteaux et al. performed MSB prediction and compression on the original image so that the MSB part could vacate most of the space, and the encrypted image could use this MSB part to embed the data [13]. In 2021, Yin et al. proposed a method based on pixel prediction and multi-MSB plane, which belongs to Vacating Room before Encryption [17]. First, it used the MED predictor to obtain the predicted value, and to calculate the predicted error (PE) between the predicted value and original value. Next, the sign bit of PE is stored in bit-plane 8, and the absolute value of PE is represented in bit-plane 7 (bit-plane 7) and bit-plane 1 (bit-plane 1). The bit plane is then divided into Uniform Blocks and Non-Uniform Blocks, and these blocks are rearranged. Since prediction errors are usually concentrated around zero, these bit planes, which have a large number of uniform blocks, can be encoded and compressed to free up space. On the other hand, VRAE technology mainly uses a specific encryption technique to maintain the dependency of the neighboring pixels in the encrypted image. Therefore, this feature can be used to hide the information in the encrypted image. Commonly used encryption techniques include Block-Level Scrambling [18] or Block-Level Encryption [19], in which pixels in the same block use the same random value for XOR (Exclusively-OR). For example, in 2019, Qin et al. proposed a VRAE method [18] that first encrypts the image by stirring the image between bit-planes, between blocks, and within blocks so that the pixels within the blocks retain their dependency. Then, block classification and encoding compression are used to generate the embedding space when embedding information. In 2019, Wang et al. proposed a VRAE approach, where their encryption uses inter- and intra-block stirring so that pixels within a block retain dependency [19]. Then, for the information embedding, they use the 3-LSB of all pixels in the flip block to divide the block into EB (Embeddable Block), and NEB (Non-Embeddable Block), and only the secret information is embedded in the EB. The location map of the block is embedded using the RDH method to hide 5-MSB of all pixels. Recently, some scholars have proposed the RDHEI approach based on Secret Sharing [20–22]. In 2018, Wu et al. proposed an RDHEI method based on Secret Sharing [20]. However, their encrypted image size was more than twice the size of the original image. In 2019, Chen and other scholars proposed a secret sharing-based RDHEI method that converts original images into encrypted images via the sharing technique of Shamir's Secret Sharing and distributes the encrypted images to information hiders for information hiding [21]. Since the size of the generated encrypted image is the same as that of the original image, no information inflation occurs. All of the above methods are targeted at a single information supporter. In 2020, Chen and other scholars proposed a new secret-sharingRDHEI approach [22], which expands the original single information hider into multiple information hiders. The total embedding volume of Chen et al.'s method is fixed, so the embedding rate decreases as the number of shared images increases.

3.SUPERVISED MACHINE LEARNING APPROACH FOR REVERSIBLE DATA HIDING IN ENCRYPTED IMAGES

We Instead of considering dedicated encryption algorithms tailored to the scenario of encrypted-domain data hiding, we here stick to the conventional stream cipher applied in the standard format. That is, the cipher text is generated by bitwise XORing the plaintext with the key stream. If not otherwise specified, the widely used stream cipher AES in the CTR mode (AES-CTR) is assumed. The resulting data hiding paradigm over encrypted domain could be more practically useful because of two reasons:

1. Stream cipher used in the standard format (e.g.,AES-CTR) is still one of the most popular and reliable encryption tools, due to its provable security and high software/hardware implementation efficiency. It may not be easy, or even infeasible, to persuade customers to adopt new encryption algorithms that have not been thoroughly evaluated.
2. Large amounts of data have already been encrypted using stream cipher in a standard way.

Firstly, in the Reversible image data-hiding scheme the cover image that acts as the media to embed the data must be encrypted. This is accomplished using the standard stream cipher algorithm the AES-CTR. The main advantage of using the stream cipher algorithm over the existing algorithm used in the existing method as XOR is:

- ➢ AES-CTR is block cipher mode algorithms which converts block cipher to stream cipher.
- ➢ AES-CTR provides the advantage of inducing confusion and diffusion in the encrypted image produced.
- ➢ It overcomes the linear cryptanalysis attack in which the action of cipher values produced can be predicted.
- ➢ It contains non-linear elements which makes to difficult for the attacker to decode the image.

Hence, due the implementation of the AES-CTR algorithm it can be told the RIDH technique takes place over an encrypted domain.

3.1 Encryption Block Diagram:

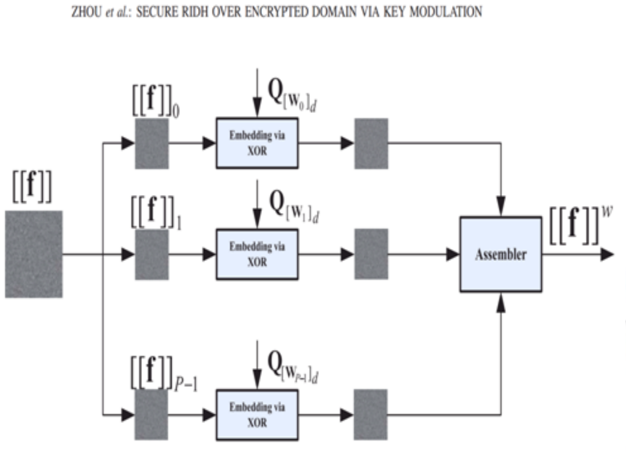


Figure 1: Schematic of data hiding over encrypted domain

Modules at the Transmitter Side:

1. Input image initialization.
2. Image encryption.
3. Image segmentation.
4. Key-modulation.
5. Assembler.

1. Input Image Initialization:

In this module, we initialize the given image (i.e.) get the input image from user by using the keyword 'uigetfile'. This contains only the pathname and filename. To read the image filename, we used 'imread' command. This read image was store in a variable as a matrix. Then we estimate the size of the given imageusing 'size' command. This give information of size of given image to estimate whether the given text was within the size of input image.

2. Image Encryption:

Assume the original image with a size of N1XN2 is in uncompressed format and each pixel with gray value falling into [0, 255] is represented by 8 bits. Denote the bits of a pixel as $b_{i,j,0}$,$b_{i,j,1}$,…,$b_{i,j,7}$ where 1<=i<=N1 and1<=j<=N2, the gray value as, and the number of pixels as N(N=N1XN2).That implies In encryption phase, the exclusive-or results of the original bits and pseudo-random bits are calculated. When stream cipher is employed, the encrypted image is generated by

$$[[\mathbf{f}]] = \mathrm{Enc}(\mathbf{f}, K) = \mathbf{f} \oplus \mathbf{K}$$

Where **f** and [[**f**]] denote the original and the encrypted images, respectively. Here, **K** denotes the key stream generated using the secret encryption key *K* . In this paper, without loss of generality, all the images are assumed to be 8 bits. Throughout this paper, we use [[**x**]] to represent the encrypted version of **x**.

As mentioned earlier, the encrypted image [[**f**]] now serves as the cover to accommodate message to be hidden. We first divide [[**f**]] into a series of non-overlapping blocks [[**f**]]$_i$'s of size *M*×*N*, where *i* is the block index. Each block is designed to carry *n* bits of message. Letting the number of blocks within the image be *B*, the embedding capacity of our proposed scheme becomes *n·B* bits. To enable efficient embedding, we propose to use *S*= 2^n binary *public* keys $\mathbf{Q}_0$,$\mathbf{Q}_1$, . . . ,$\mathbf{Q}_{S-1}$, each of which is of length *L*=*M*×*N*× 8 bits. All $\mathbf{Q}_j$'s, for 0 ≤*j*≤*S*− 1, are made publicly accessible, which implies that even the attacker knows them These public keys are preselected prior to the message embedding, according to a criterion of maximizing the minimum Hamming distance among all keys. The algorithm developed by MacDonald can be used to this end. Note that all the public keys are built into the data hider and the recipient when the whole system is set up, and hence, it is not necessary to transmit them during the data embedding stage. Also, for fixed *S* and *L*, Hamming showed that an upper bound on the minimum Hamming distance can be given as follows. First, determine two integersm_1 and m_2 by

$$\sum_{i=0}^{m_1} \binom{L}{i} \leq \frac{2^L}{S} < \sum_{i=0}^{m_1+1} \binom{L}{i}$$

$$\sum_{i=0}^{m_2} \binom{L-1}{i} \leq \frac{2^{L-1}}{S} < \sum_{i=0}^{m_2+1} \binom{L-1}{i}$$

Where _*Li*_ = (*L*!/*i* !(*L* − *i*)!). It can be shown that both *m*1 and *m*2 are unique. Then, the minimum Hamming distance among all **Q***j*'s satisfies

$$d_{\min} \leq \max\{2m_1 + 1, 2m_2 + 2\}$$

3. Image Segmentation:

The term image segmentation refers to the partition of an image into a set of regions that cover it. The goal in many tasks is for the regions to represent meaningful areas of the image, such as the crops, urban areas, and forests of a satellite image. In other analysis tasks, the regions might be sets of border pixels grouped into such structures as line segments and circular arc segments in images of 3D industrial objects. Regions may also be denied as groups of pixels having both a border and a particular shape such as a circle or ellipse or polygon. When the interesting regions do not cover the whole image, we can still talk about segmentation, into foreground regions of interest and background regions to be ignored.

Segmentation has two objectives. The first objective is to decompose the image into parts for further analysis. In simple cases, the environment might be well enough controlled so that the segmentation process reliably extracts only the parts that need to be analyzed further. Forexample, in the chapter on color, an algorithm was presented for segmenting ahuman face from a color video image. The segmentation is reliable, provided that the person's clothing or room background does not have the same color components as a human face. In complex cases, such as extracting a complete road network from a grayscale aerial image, the segmentation problem can be very difficult and might require application of agreat deal of domain building knowledge.

The second objective of segmentation is to perform a change of representation. The pixels of the image must be organized into higher-level units that are either more meaningful or more efficient for further analysis (or both). A critical issue is whether or not segmentation can be performed for many different domains using general bottom-up methods that do not use any special domain knowledge.

4. Key Modulation:

EX-OR Gate truth table:

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Public key modulation mechanism is used to embed the data into the blocks of encrypted image. It is used to overcome the problems arise in the frequency domain of an image.The proposed technique embeds message through a public key modulation mechanism and performs data extraction by exploiting the statistical distinguish ability of encrypted and non-encrypted image blocks. Since the decoding of the message bits and the original image is tied together, our proposed technique belongs to the category of non-separable RIDH solutions Compared with the state-of-the-art methods, the proposed approach provides higher embedding capacity and is able to achieve perfect reconstruction of the original image as well as the embedded message bits. for all the existing RIDH schemes including both non separable as well as separable solutions, an extra data hiding key is introduced to ensure embedding security. Certainly, the data hiding key needs to be shared and managed between the date hider and the recipient. As mentioned earlier, the key management functions Instead of considering dedicated encryption algorithms tailored to the scenario of encrypted-domain data hiding, we here stick to the conventional stream cipher applied in the standard format. That is, the cipher text is generated by bitwise XOR the plaintext with the key stream. Find the public key $Q[W_i]_d$ associated with W_i , where the index $[W_i]_d$ is the decimal representation of W_i For instance, when $n = 3$ and $W_i = 010$, the corresponding public key is Q2. Embed the length-n message bits W_i into the i th block via

$$[[\mathbf{f}]]_i^{w} = [[\mathbf{f}]]_i \oplus \mathbf{Q}_{[\mathbf{w}_i]_d}$$

Data Embedding:

In LSB message bits are embed in least significant bit of cover image. LSB steganography can be classified by two methods LSB replacement and LSB matching was firstly discussed by T.Sharp. LSB replacement steganography replace the last bits of cover image with each bits of the message that needs to be hidden. Second method is LSB matching in which each pixel of the cover image is taken mainly in a pseudo-random order which is generated by a secret key, if the LSB of the cover pixel matches the bit of secret data no changes are done otherwise, one is added or subtracted from the cover pixel value, at random. If the length of secrete message contains fewer bits than the number of pixels in the cover image The schematic diagram of the proposed message embedding algorithm over encrypted domain is shown in Fig. In this paper, we do not consider the case of embedding multiple watermarks for one single block, meaning that each block is processed once at most. For simplicity, we assume that the number of message bits to be embedded is $n.A$, where $A \leq B$ and B is the number of blocks within the image. The steps for performing the message embedding are summarized as follows.

Embedding Algorithm:

- Step 1: Initialize block index i = 1.
- Step 2: Extract n bits of message to be embedded, denoted by $\mathbf{W}_i$

- Step 3: Find the public key $\mathbf{Q}_{[\mathbf{W}_i]_d}$ associated with $\mathbf{W}_i$, where the index $[\mathbf{W}_i]_d$ is the decimal representation of $\mathbf{W}_i$. For instance, when n = 3 and $\mathbf{W}_i$= 010, the corresponding public key is $\mathbf{Q}_2$.
- Embed the length-*n* message bits $\mathbf{W}i$ into the *i* th block via

$$[[\mathbf{f}]]_i^w = [[\mathbf{f}]]_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d}$$

- Step 5: Increment i = i + 1 and repeat Steps 2–4 until all the message bits are inserted.

The watermark length parameter *A* needs to be transmitted alone with the embedded message bits. There are many ways to solve this problem. For instance, we can reserve some blocks to embed *A*, or we can append an end-of-file symbol to the message to be embedded, such that the decoder can implicitly determine *A*. Both strategies can be readily implemented in practice with negligible effect on the actual embedding rate. For the sake of simpler presentation, we exclude the discussion of embedding *A* in the sequel.

From the above steps, it can be observed that the message embedding is performed without the aid of a secret data hiding key. As will be proved in Section VI, high level of embedding security can still be guaranteed, thanks to the protection offered by the encryption key *K* . In addition, the computations involved in message embedding are rather small (simple XOR operations), and all the block-by-block processing can be readily made parallel, achieving high throughput.

It is emphasized that the possibility of eliminating the data hiding key is not unique to our proposed method, but rather arguably applicable for all non-separable RIDH schemes over encrypted domain. For instance, the existing non separable RIDH schemes in the encrypted domain can be naturally extended to provide security for message embedding, eliminating the necessity of introducing an extra data hiding key. This could lead to significant reduction of the computational cost and potential risk of building up a secure KMS, which has been proved to be very challenging in the multiparty environment

Feature Selection for Discriminating Encrypted and Non-encrypted Image Blocks:

To differentiate encrypted and original unencrypted image blocks, we here design a feature vector *ρ*=(*H*, *σ*,**V**) ,integrating the characteristics from multiple perspectives. Here, *H* is a tailored entropy indicator, *σ* is the SD of the block, and **V** represents the directional local complexities in four directions. The formation of the above feature elements will be detailed as follows.Compared with the original unencrypted block, the pixels in the encrypted block tend to have a much more uniform distribution. This motivates us to introduce the local entropy into the feature vector to capture such distinctive characteristics. However, we need to be cautious when calculating the entropy values because the number of avail-able samples in a block would be quite limited, resulting in estimation bias, especially when the block size is small. For instance, in the case that *M*=*N*= 8, we only have 64-pixel samples, while the range of each sample is from 0 to 255. To reduce the negative effect of insufficient number of samples relative to the large range of each sample, we propose to compute the entropy quantity based on quantized samples, where the quantization step size is designed in accordance with the block size. Specifically, we first apply uniform scalar quantization to each pixel of the block

$$\hat{f} = \left\lfloor \frac{MN \cdot f}{256} \right\rfloor$$

Where *f* and ˆ *f* denote the original and the quantized pixel values, respectively. Certainly, *f*ˆ falls into the range [0, *MN*− 1]. The entropy indicator *H* based on quantized samples is then given by As a single first-order entropy quantity may not be sufficient to cover all the underlying characteristics of a block, we suggest augmenting the feature vector by introducing another element, i.e., the SD defined by

$$\sigma = \sqrt{\frac{1}{MN}\sum_j (\mathbf{f}(j) - \mu)^2}$$

Where **f**(*j*) is the *j* th pixel in the block and *μ* = (1/*MN*)_*j* **f**(*j*) is the sample mean over all the samples in the block. By including this feature element, we can improve the classification performance as the data depressiveness and denseness can be better reflected.

5. Assembler:

The decoder in the data center has the decryption key K and attempts to recover both the embedded message and the original image simultaneously from [[f]]w, which is assumed to be perfectly received without any distortions. Note that this assumption is made in almost all the existing RIDH methods. Due to the interchangeable property of XOR operations, thedecoder first XORs [[f]]w with the encryption key stream K and obtains fw =[[f]]w ⊕K.

The resulting fw is then partitioned into a series of non-overlapping blocks fwi 's of size M × N, similar to the operation conducted at the embedding stage. we have fw i = fi ⊕Q[Wi]d.

3.2 Decryption Block Diagram:

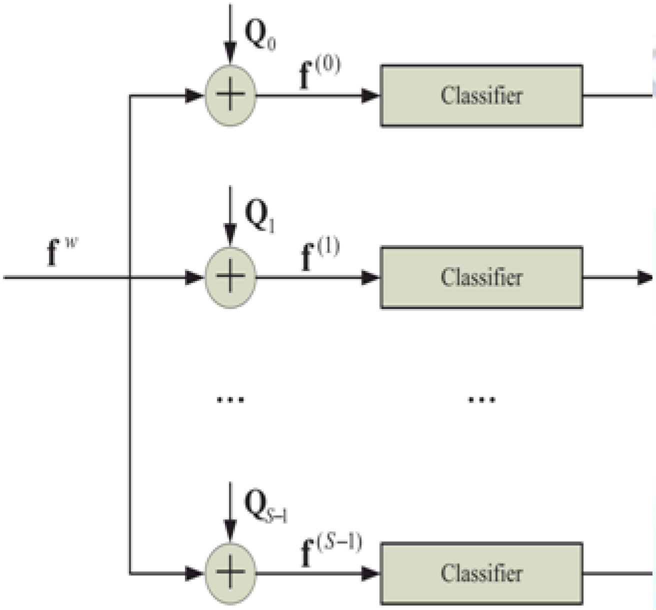


Figure 2: Schematic of data extraction

Modules at the Receiver Side:

1. Assembled image.
2. Segmentation.
3. Data extraction and Image Recovery.
4. SVM classifier.

1. Assembled Image

The decoder in the data center has the decryption key K and attempts to recover both the embedded message and the original image simultaneously from [[f]]w, which is assumed to be perfectly received without any distortions. Note that this assumption is made in almost all the existing RIDH methods. Due to the interchangeable property of XOR operations, the decoder first XORs [[f]]w with the encryption key stream K and obtains fw =[[f]]w ⊕K. The resulting fw is then partitioned into a series of non-overlapping blocks fwi 's of size M × N, similar to the operation conducted at the embedding stage. we have fw i = fi ⊕Q[Wi]d.

2. Segmentation

24-bit color image is best defined by RGB color model in which each color appears in its primary spectral component of red, green and blue. This model is based on Cartesian coordinate system shown in Figure. In which RGB primary value are at three corner, the secondary color cyan, magenta and yellow are at three other corner, black is ate the origin and white is at the corner farthest from the origin. Line joining the two corners has equal values for red, green and blue. This produces various shades of grey. The locus of all these points is called the grey line. In RGB model, each pixel is composed of RGB values and each of these colors requires 8- bit for its representation. Hence each pixel is represented by 24 bits. So total number of color possible with 24-bit RGB image.

3. Data Extraction and Image Recovery:

The decoder in the data center has the decryption key *K* and attempts to recover both the embedded message and the original image simultaneously from [[**f**]]*w*, which is assumed to be perfectly received without any distortions. Note that this assumption is made in almost all the existing RIDH methods. Due to the interchangeable property of XOR operations, the any attacker without the data-hiding key cannot obtain the parameter values and the pixel-groups, therefore cannot extract the embedded data. Furthermore, although the receiver having the data-hiding key can successfully extract the embedded data, he cannot get any information about the original image content. Decoder first XORs [[**f**]]*w* with the encryption key stream **K** and obtains

$$\mathbf{f}^w = [[\mathbf{f}]]^w \oplus \mathbf{K}$$

The resulting **f***wi*'s then partitioned into a series of non-overlapping Blocks fwi's of size *M* × *N*, similar to the operation conducted at the embedding stage. From (6), we have where [*j*]2 denotes the length-*n* binary representation of *j* and *n* = log2 *S*. For example, if *n* = 3 and *j* = 7, then [*j*]2 = 111. Upon determining **W***i* , the original image block can be easily recovered by

$$\mathbf{f}_i^w = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d}$$

The joint data extraction and image decryption now becomes a blind signal separation problem as both **W***i*and **f***i* are unknowns. Our strategy of solving this problem is based on the following observation: **f***i*, as the original image block, very likely exhibits certain image structure, conveying semantic information. Note that **Q**[**W***i*]*d* must match one of the elements in *Q* = {**Q**0,**Q**1, . . . ,**Q***S*−1}. Then, if we XOR **f***wi*with all **Q***j*'s, one of the results must be **f***i*, which would demonstrate structural information. As will become clear shortly, the other results correspond to randomized blocks, which can be distinguished from the original structured **f***i*. More specifically, we first create *S* decoding candidates by XORing **f***wi* with all the *S* possible public keys **Q**0, **Q**1, . . ., ***QS*–1**

$$
\begin{aligned}
\mathbf{f}_i^{(0)} &= \mathbf{f}_i^{w} \oplus \mathbf{Q}_0 = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_0 \\
\mathbf{f}_i^{(1)} &= \mathbf{f}_i^{w} \oplus \mathbf{Q}_1 = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_1 \\
&\vdots \\
\mathbf{f}_i^{(S-1)} &= \mathbf{f}_i^{w} \oplus \mathbf{Q}_{S-1} = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_{S-1}.
\end{aligned}
$$

The result **f**(*t*)*i*= Enc(**f***i* ,**Q**[**W***i*]*d***Q***t*) corresponds to an encrypted version of **f***i* with equivalent key stream being **Q**[**W***i*]*d***Q***t*. Note that all the public keys **Q***j*'s, for 0 ≤ *j* ≤ *S* −1, are designed to have maximized minimum Hamming distance, and the upper bound is given in (5). Hence, **f**(*t*)*i* tends to lose the image structural information, making it appear random To identify which candidate corresponds to **f***i* , we apply the designed two-class SVM classifier to these *S* candidates. Let **r** = (*r*0, *r*1, . . . , *rS*−1)_ be the vector recording the classification results, where *r j* = 0 and *r j* = 1 correspond to the original (structured) and randomized blocks, respectively. If there exists a unique *j* such that *r j* = 0, then we decode the embedded message bits as

$$\mathbf{W}_i = [j]_2$$

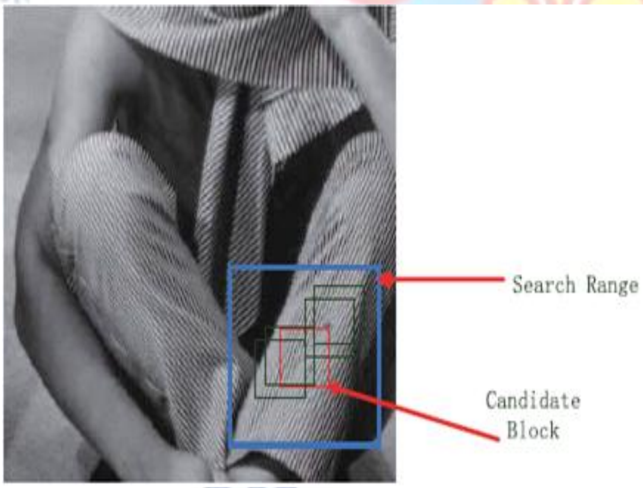


Figure 3: Illustration of the error correction mechanism based on image self-similarity

4.SVM Classifier

In addition to the above feature components, we also include directional complexity indicators that encode the local geometric information. To this end, we define a four-tuple vector **V** = (*v*1, *v*2, *v*3, *v*4), where

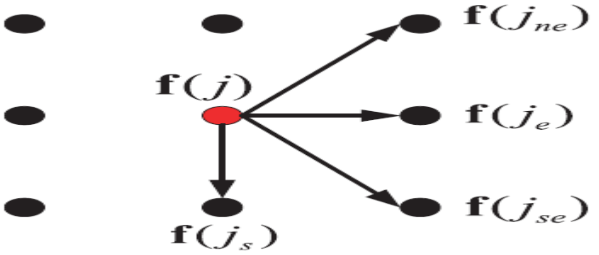


Figure 4: Illustration of the neighbors of f(*j*)

In addition to the above feature components, we also include directional complexity indicators that encode the local geometric information. To this end, we define a four-tuple vector **V** = (*v*1, *v*2, *v*3, *v*4)_, where

$$
\begin{aligned}
v_1 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{ne})| \\
v_2 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{e})| \\
v_3 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{se})| \\
v_4 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{s})|
\end{aligned}
$$

where **f**(*jne*), **f**(*je*), **f**(*jse*), and **f**(*js*) represent the neighbors in the 45° (northeast), 0° (east), −45° (southeast), and −90° (south) directions, relative to **f**(*j*), as shown in Fig. Upon the determination of the feature vector *ρ*, we train a two-class SVM classifier with RBF (Gaussian) kernel taking the form

$$\mathrm{Ker}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|}$$

The 0-class and 1-class correspond to the unencrypted and encrypted image blocks, respectively However, we do observe several cases where there exist multiple *j* 's or no *j* such that *r j* = 0. When any of these two cases happens, it indicates that some decoding errors appear. To formally analyze these errors and later suggest an effective error correction mechanism, we define two types of classification errors.

1. Type I Error: $\mathbf{f}_i^{(j)}=\mathbf{f}_i$, while r_j= 1.
2. Type II Error: $\mathbf{f}_i^{(j)}=\mathbf{f}_i$, while r_j= 0.

Type I error mainly occurs when the original block **f***i* is very complicated, e.g., from highly textured regions, behaving similarly as an encrypted block. Type II error usually arises when the block size is rather small, making an encrypted block mistakenly be classified as an original unencrypted one. As verified experimentally from 200 test images of size 512 × 512, for a specific block, we assume that at most one type of error will occur. Under this assumption, both Type I and Type II errors can be easily detected. When Type I error occurs, the classification result vector becomes **r** = **1**_. While when Type II error appears, the following inequality holds

$$\sum_j r_j < 2^n - 1$$

Where n= $\log_2 S$. In the rare cases that the above assumption does not hold (both types of errors appear simultaneously), these errors cannot be detected and will still be counted when calculating the extraction accuracy When classification errors are detected for some blocks, we need a mechanism to correct them. Though the classifier is carefully designed, it is still difficult to distinguish those highly textured original blocks from the encrypted ones, especially when the block size is small. To solve this challenging problem, we propose to exploit the self-similarity property inherent to natural images. Even for those highly textured images, it is observed that similar blocks could be found in a nonlocal window.

According to this phenomenon, the proposed error correction approach is based on the following key observation: if a block is correctly decoded, then with very high probability, there are some similar patches around it. Such a property of nonlocal image similarity motivates us to rank all the potential candidate blocks according to the minimum distance with the patches in a nonlocal search window. To this end, we first define a to-be-corrected set C by

$$\mathcal{C} = \begin{cases} \{\mathbf{f}_i^{(j)} \mid 0 \le j \le S-1\} & \text{Type I error detected} \\ \{\mathbf{f}_i^{(j)} \mid r_j = 0\} & \text{Type II error detected.} \end{cases}$$

For any candidate block **f**(j) i in C, we calculate its distances from all the other blocks in a search range where D shares the same center as experimentally determined as 5M × 5N. We then can compute the minimum patch distance within the search window

$$d_i^{(j)} = \min_{\mathbf{D} \in \mathcal{D} \setminus \{\mathbf{f}_i^{(j)}\}} \left\| \mathbf{f}_i^{(j)} - \mathbf{D} \right\|_F^2$$

We employ the simple MSE criterion when ranking the candidate blocks. By including the texture direction and scale into the above minimization framework, we could further improve the error correcting performance, but we find that the additional gain is rather limited and the incurred complexity is large. The candidate **f**(j)i that gives the smallest d(j)i is then selected as the decoded block. Upon determining the index j of the employed public key, the embedded message bits and the original image block can be straightforwardly recovered as in This nonlocal-based error correction strategy will be shown experimentally to be quite effective in The above joint data extraction and image decryption procedures can also be summarized.

According to the context of the attack, the attacker may have access to different amounts of information. Clearly, the attacker at least can access to watermarked signal, namely, $[[\mathbf{f}]]^w$. In some occasions, the embedded message or the cover signal can also be available to the attacker [31]. Therefore, the security level of the encrypted-domain RIDH scheme should be assessed for different contexts. Similar to the problem of evaluating the security for encryption primitives, Cayre et al defined three types of attacks.

1. The watermarked only attack (WOA), in which the attacker only has access to watermarked images.
2. The known message attack, in which the attacker has access to several pairs of *previously* watermarked images and the associated messages. Certainly, the currently transmitted message bits are not known to the attacker.
3. The known original attack, in which the attacker has access to several pairs of *previously* watermarked images and the corresponding cover image. Certainly, the current cover image is not known to the attacker.

As explained the purposes of the last two attacks are mainly to recover the data hiding key, so as to extract the future embedded messages or hack different pieces of content watermarked with the same key. In our proposed RIDH scheme, the data hiding key has been eliminated, and hence, these two attack models are not applicable.Under the WOA, the only attack type relevant to our scheme, the attacker attempts to extract the embedded message and/or recover the original image from the watermarked and encrypted image $[[\mathbf{f}]]^w$. Before evaluating the security under WOA, let us first give the definition of message in distinguish ability, which should hold for any secure encryption method. getting the watermarked and encrypted image $[[\mathbf{f}]]^w$, we can still partition it into non over-lapping blocks of size M×N. For each block, we can generate S decoding candidates in a similar fashion as

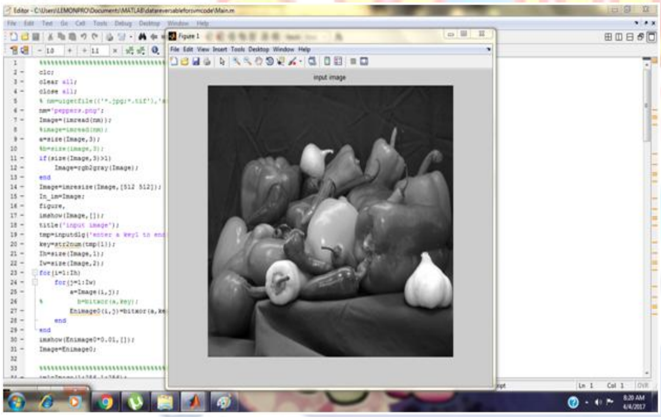
$$
\begin{aligned}
\mathbf{f}_i^{(0)} &= [[\mathbf{f}]]_i^{w} \oplus \mathbf{Q}_0 = \mathbf{f}_i^{w} \oplus \mathbf{Q}_0 \oplus \mathbf{K}_i \\
&= \mathrm{Enc}(\mathbf{f}_i^{w} \oplus \mathbf{Q}_0, \mathbf{K}_i) \\
\mathbf{f}_i^{(1)} &= [[\mathbf{f}]]_i^{w} \oplus \mathbf{Q}_1 = \mathbf{f}_i^{w} \oplus \mathbf{Q}_1 \oplus \mathbf{K}_i \\
&= \mathrm{Enc}(\mathbf{f}_i^{w} \oplus \mathbf{Q}_1, \mathbf{K}_i) \\
&\vdots \\
\mathbf{f}_i^{(S-1)} &= [[\mathbf{f}]]_i^{w} \oplus \mathbf{Q}_{S-1} = \mathbf{f}_i^{w} \oplus \mathbf{Q}_{S-1} \oplus \mathbf{K}_i \\
&= \mathrm{Enc}(\mathbf{f}_i^{w} \oplus \mathbf{Q}_{S-1}, \mathbf{K}_i)
\end{aligned}
$$

With any observed **f**(*j*)*i* , it is computationally infeasible to figure out, with probability significantly larger than 1/*S*, which one among is the message encrypted by **K***i* , due to the property of message indistinguishability described in . Therefore, the attacker attempting to extract the embedded message bits from [[**f**]]*w* should be able to do no better than random guessing. This proves the security of our proposed encrypted-domain RIDH strategy against WOA attack.

4. RESULTS& DISCUSSION

In this section, we perform experiments and analysis. All the tested images are gray 425 level sized by 512 × 512. Figures 5-6 presents results various processes in the proposed method.

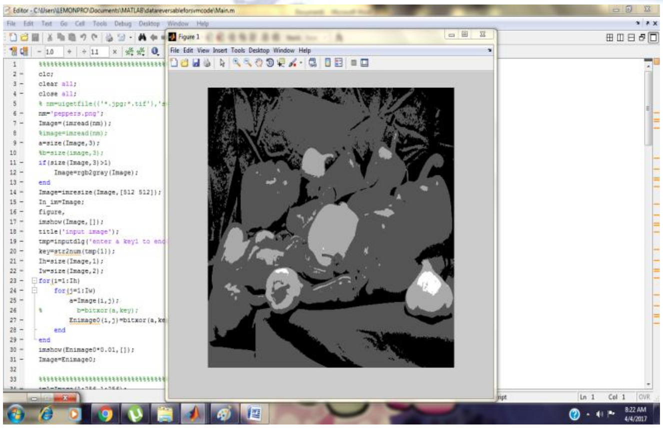
Encryption Process:



```
Editor - C:\Users\LEMONPRO\Documents\MATLAB\datareversableforsvmcode\Main.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc;
clear all;
close all;
% nm=uigetfile({'*.jpg;*.tif'},'s
nm='peppers.png';
Image=(imread(nm));
%image=imread(nm);
a=size(Image,3);
%b=size(image,3);
if(size(Image,3)>1)
    Image=rgb2gray(Image);
end
Image=imresize(Image,[512 512]);
In_im=Image;
figure,
imshow(Image,[]);
title('input image');
tmp=inputdlg('enter a key1 to enc
key=str2num(tmp(1));
Ih=size(Image,1);
Iw=size(Image,2);
for(i=1:Ih)
    for(j=1:Iw)
        a=Image(i,j);
%         b=bitxor(a,key);
        Enimage0(i,j)=bitxor(a,ke
    end
end
imshow(Enimage0*0.01,[]);
Image=Enimage0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

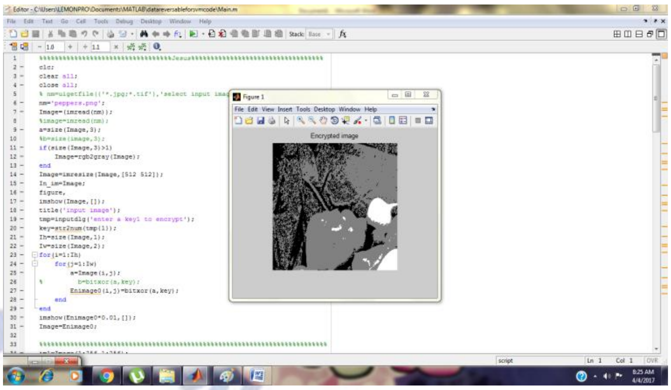
Figure 5: Referring to an input image of peppers with a size of 512x512 pixels.



```
Editor - C:\Users\LEMONPRO\Documents\MATLAB\datareversableforsvmcode\Main.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc;
clear all;
close all;
% nm=uigetfile({'*.jpg;*.tif'},'s
nm='peppers.png';
Image=(imread(nm));
%image=imread(nm);
a=size(Image,3);
%b=size(image,3);
if(size(Image,3)>1)
    Image=rgb2gray(Image);
end
Image=imresize(Image,[512 512]);
In_im=Image;
figure,
imshow(Image,[]);
title('input image');
tmp=inputdlg('enter a key1 to enc
key=str2num(tmp(1));
Ih=size(Image,1);
Iw=size(Image,2);
for(i=1:Ih)
    for(j=1:Iw)
        a=Image(i,j);
%         b=bitxor(a,key);
        Enimage0(i,j)=bitxor(a,ke
    end
end
imshow(Enimage0*0.01,[]);
Image=Enimage0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 6: Enter the public Key For Encryption

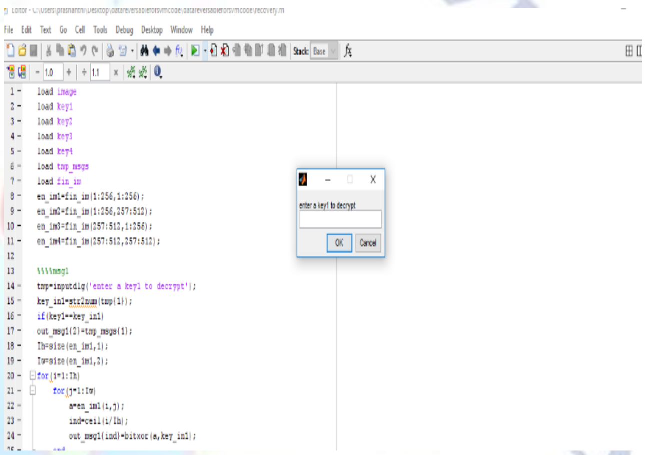


```
Editor - C:\Users\LEMONPRO\Documents\MATLAB\datareversableforsvmcode\Main.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Jesus%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc;
clear all;
close all;
% nm=uigetfile({'*.jpg;*.tif'},'select input imag
nm='peppers.png';
Image=(imread(nm));
%image=imread(nm);
a=size(Image,3);
%b=size(image,3);
if(size(Image,3)>1)
    Image=rgb2gray(Image);
end
Image=imresize(Image,[512 512]);
In_im=Image;
figure,
imshow(Image,[]);
title('input image');
tmp=inputdlg('enter a key1 to encrypt');
key=str2num(tmp(1));
Ih=size(Image,1);
Iw=size(Image,2);
for(i=1:Ih)
    for(j=1:Iw)
        a=Image(i,j);
%         b=bitxor(a,key);
        Enimage0(i,j)=bitxor(a,key);
    end
end
imshow(Enimage0*0.01,[]);
Image=Enimage0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 7: Enter the Messages for Data Embedding by using key modulation with help of xor operation

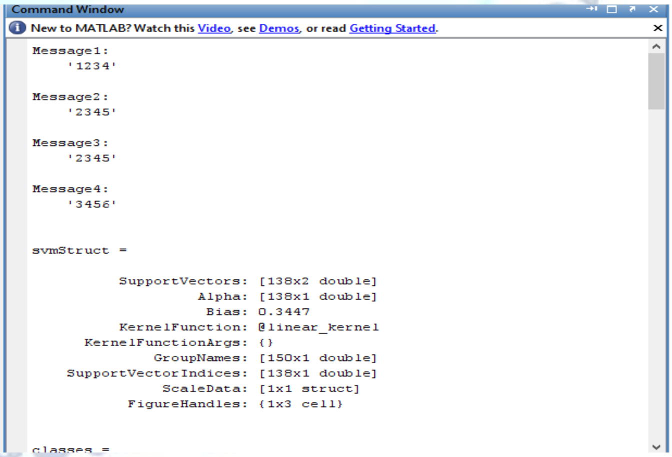
Decryption Process:



```
Editor - C:\Users\prashanthi\Desktop\datareversableforsvmcode\datareversableforsvmcode\recovery.m
load image
load key1
load key2
load key3
load key4
load tmp_msgs
load fin_im
en_im1=fin_im(1:256,1:256);
en_im2=fin_im(1:256,257:512);
en_im3=fin_im(257:512,1:256);
en_im4=fin_im(257:512,257:512);

%%%%msg1
tmp=inputdlg('enter a key1 to decrypt');
key_in1=str2num(tmp{1});
if(key1==key_in1)
out_msg1(2)=tmp_msgs(1);
Ih=size(en_im1,1);
Iw=size(en_im1,2);
for(i=1:Ih)
    for(j=1:Iw)
        a=en_im1(i,j);
        ind=ceil(i/Ih);
        out_msg1(ind)=bitxor(a,key_in1);
```

Figure 8: Enter the public Key For decryption



```
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Message1:
    '1234'

Message2:
    '2345'

Message3:
    '2345'

Message4:
    '3456'


svmStruct =

          SupportVectors: [138x2 double]
                   Alpha: [138x1 double]
                    Bias: 0.3447
          KernelFunction: @linear_kernel
      KernelFunctionArgs: {}
              GroupNames: [150x1 double]
    SupportVectorIndices: [138x1 double]
               ScaleData: [1x1 struct]
           FigureHandles: {1x3 cell}


classes =
```

Figure 9: Decrypted Messages in Command Window

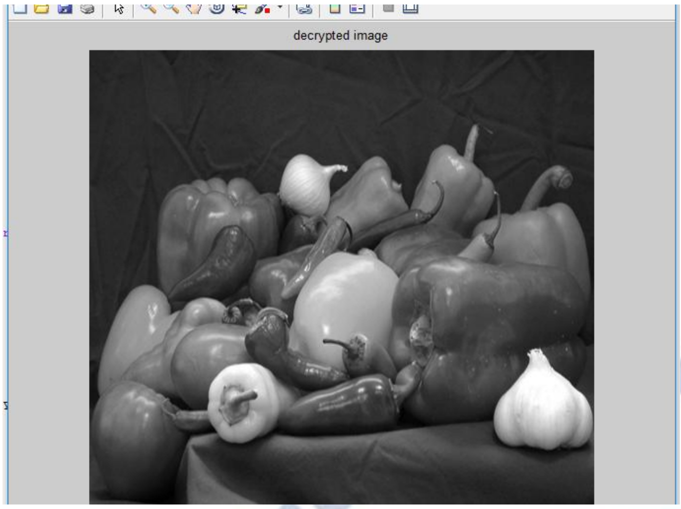
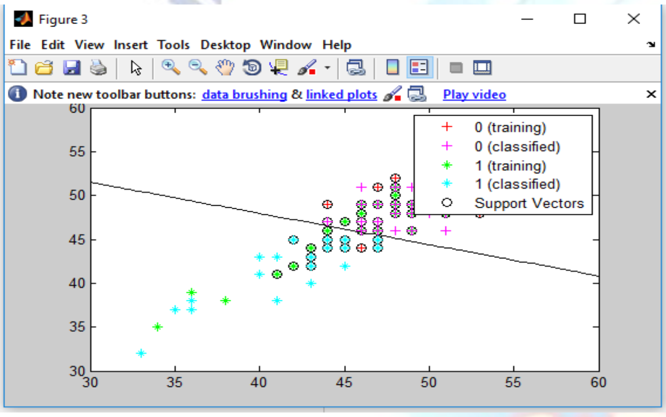


Figure 10: Decrypted image



Note new toolbar buttons: data brushing & linked plots Play video

Figure 11: SVM classifier used for to detect impressibility of information based on binary classification

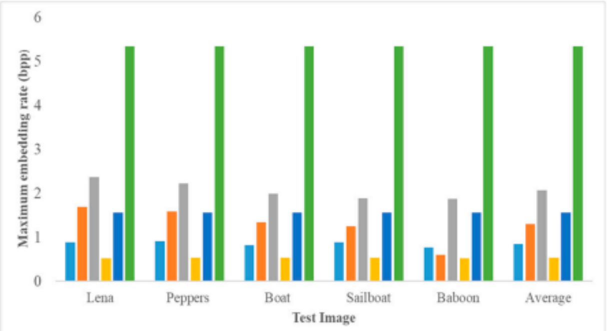


Figure 12: Maximum embedding rate comparison among the proposed method

The comparisons of embedding capacity (bits) and embedding rate (bpp), here the experimental data is used in a three-out-of-four threshold secret sharing approach, in our method, a pixel of a shared image can embed multiple data of secret information.

Advantages:

- Enabling us to jointly decode the embedded message and the original image perfectly.
- Security is improved.
- Copy forgery problem is reduced.

Applications:

- Military
- Secure remote sensing
- Cloud computing
- Medical image sharing
- Satellite communication

4.CONCLUSIONS

In this paper, we design a secure RIDH scheme operated over the encrypted domain. We suggest a public key modulation mechanism, which allows us to embed the data via simple BIT-XOR operations, without the need of accessing the secret encryption key. constructed shared image has B part of the embedding space, and each shared image is encrypted. Information hiders can hide secret information in Part B of the shared image. Compared with other methods, our method has a higher embedding rate, and the embedding rate does not decrease due to more shared images. At the decoder side, we propose to use a powerful two-class SVM classifier to discriminate encrypted and non-encrypted image patches, enabling us to jointly decode the embedded message and the original image signal perfectly. We have also performed extensive experiments to validate the superior embedding performance of our proposed RIDH method over encrypted domain.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] N. A. Saleh, H. N. Boghdad, S. I. Shaheen, A. M. Darwish, "High Capacity Lossless Data Embedding Technique for Palette Images Based on Histogram Analysis," Digital Signal Processing, 20, pp. 1629–1636, 2010.

[2] J. Tian, "Reversible Data Embedding Using a Difference Expansion," IEEE Trans. on Circuits and Systems for Video Technology, 13(8), pp. 890–896, 2003.

[3] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, “Reversible Data Hiding,” IEEE Trans. on Circuits and Systems for Video Technology, 16(3), pp. 354–362, 2006.

[4] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, “Lossless Generalized-LSB Data Embedding,” IEEE Trans. on Image Processing,

[5] X. Hu, W. Zhang, X. Li, and N. Yu, “Minimum Rate Prediction and Optimized Histograms Modification for Reversible Data Hiding,” IEEE Trans. on Information Forensics and Security, 10(3), pp. 653-664, 2015.

[6] X. Zhang, “Reversible Data Hiding with Optimal Value Transfer,” IEEE Trans. on Multimedia, 15(2), 316–325, 2013.

[7] W. Zhang, X. Hu, X. Li, and N. Yu, “Optimal Transition Probability of Reversible Data Hiding for General Distortion Metrics and Its Applications,” IEEE Trans. on Image Processing, 24(1), pp. 294-304, 2015.

[8] S. Lian, Z. Liu, Z. Ren, and H. Wang, “Commutative Encryption and Watermarking in Video Compression,” IEEE Trans. on Circuits and Systems for Video Technology, 17(6), pp. 774–778, 2007.

[9] M. Cancellaro, F. Battisti, M. Carli, G. Boato, F. G. B. Natale, and A. Neri, “A Commutative Digital Image Watermarking and Encryption Method in the Tree Structured Haar Transform Domain,” Signal Processing: Image Communication, 26(1), pp. 1–12, 2011.

[10]] X. Zhang, “Commutative Reversible Data Hiding and Encryption,” Security and Communication Networks, 6, pp. 1396–1403, 2013.

[11] X. Zhang, “Reversible Data Hiding in Encrypted Image,” IEEE Signal Processing Letters, 18(4), pp. 255–258, 2011.

[12] W. Hong, T.-S.Chen, and H.-Y. Wu, “An Improved Reversible Data Hiding in Encrypted Images Using Side Match,” IEEE Signal Processing Letters, 19(4), pp. 199–202, 2012.

[13] J. Yu, G. Zhu, X. Li, and J. Yang, “An Improved Algorithm for Reversible Data Hiding in Encrypted Image,” Proceeding of the 11th International Workshop on Digital-Forensics Watermark (IWDW 2012), Shanghai, China, Oct. 31-Nov. 02, 2012, Lecture Notes in Computer Science, 7809, pp. 358-367, 2013.

[14] W. Puech, M. Chaumont, and O. Strauss, “A Reversible Data Hiding Method for Encrypted Images,” Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, Proc. SPIE, 6819, 2008.

[15] X. Zhang, “Separable Reversible Data Hiding in Encrypted Image,” IEEE Trans. Information Forensics & Security, 7(2), pp. 526–532, 2012.

[16] Z. Qian, X. Zhang, and S. Wang, “Reversible Data Hiding in Encrypted JPEG Bitstream,” IEEE Trans. on Multimedia, 16(5), pp. 1486–1491, 2014.

[17] M. S. A. Karim, and K. Wong, “Universal Data Embedding in Encrypted Domain,” Signal Processing, 94, pp. 174-182, 2014.

[18] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, “Reversible Data Hiding in Encrypted Images by Reserving Room Before Encryption,” IEEE Trans. Information Forensics & Security, 8(3), pp. 553-562, 2013.

[19] W. Zhang, K. Ma, and N. Yu, “Reversibility Improved Data Hiding in Encrypted Images,” Signal Processing, 94, pp. 118-127, 2014.

[20] Y.-C. Chen, C.-W.Shiu, and G. Horng, “Encrypted Signal-Based Reversible Data Hiding with Public Key Cryptosystem,” Journal of Visual Communication and Image Representation, 25, pp. 1164-1170, 2014.

[21] P. Paillier, “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes,” Proceeding of the Advances Cryptology, EUROCRYPT’99, LNCS, 1592, pp. 223-238, 1999.

[22] T. Bianchi, A. Piva, and M. Barni, “On the Implementation of the Discrete Fourier Transform in the Encrypted Domain,” IEEE Trans. Information Forensics and Security, 4(1), pp. 86–97, 2009.

[23] T. Bianchi, A. Piva, and M. Barni, “Composite Signal Representation for Fast and Storage-Efficient Processing of Encrypted Signals,” IEEE Trans. Information Forensics and Security, 5(1), pp. 180–187, 2010.

[24] P. Zheng, and J. Huang, “Discrete Wavelet Transform and Data Expansion Reduction in Homomorphic Encrypted Domain,” IEEE Trans. Image Processing, 22(6), pp. 2455-2468, 2013.

[25] I. Damgård, and M. Jurik, “A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System,” Public Key Cryptography, pp. 119-136, 2001.

[26] J. Fridrich, M. Goljan, P. Lisonek, and D. Soukal, “Writing on Wet Paper,” IEEE Trans. Signal Processing, 53(10), pp. 3923-3935, 2005.

[27] Ravikiran, D. N., & Dethe, C. G. (2018). Improvements in Routing Algorithms to Enhance Lifetime of Wireless Sensor Networks. International Journal of Computer Networks & Communications (IJCNC), 10(2), 23-32.

[28] Ravikiran, D. N., & Dethe, C. G. Fuzzy Rule Selection using LEACH Algorithm to Enhance Life Time in Wireless Sensor Networks. Advances in Wireless and Mobile Communications. ISSN, 0973-6972.

[29] Rajesh, G., Thommandru, R., & Subhani, S. M. DESIGN AND IMPLEMENTATION OF 16-BIT HIGH SPEED CARRY SELECT PARALLEL PREFIX ADDER.

[30] Polanki, K., Purimetla, N. R., Roja, D., Thommandru, R., & Javvadi, S. Predictions of Tesla Stock Price based on Machine Learning Model.

[31] Thommandru, R. A PROSPECTIVE FORECAST OF BRAIN STROKE USING MACHINE LEARNING TECHNIQUES.

[32] Rajesh, G., Raja, A., & Thommandru, R. OPTIMIZATION OF MINIATURIZED MICROSTRIP PATCH ANTENNAS WITH GA.

[33] Vellela, S. S., & Balamanigandan, R. (2022, December). Design of Hybrid Authentication Protocol for High Secure Applications in Cloud Environments. In 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 408-414). IEEE.

[34] Vellela, S. S., & Balamanigandan, R. (2024). Optimized clustering routing framework to maintain the optimal energy status in the wsn mobile cloud environment. Multimedia Tools and Applications, 83(3), 7919-7938.

[35] Praveen, S. P., Sarala, P., Kumar, T. K. M., Manuri, S. G., Srinivas, V. S., & Swapna, D. (2022, November). An Adaptive Load Balancing Technique for Multi SDN Controllers. In 2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS) (pp. 1403-1409). IEEE.

[36] Priya, S. S., Vellela, S. S., Reddy, V., Javvadi, S., Sk, K. B., & Roja, D. (2023, June). Design And Implementation of An Integrated IOT Blockchain Framework for Drone Communication. In 2023 3rd

International Conference on Intelligent Technologies (CONIT) (pp. 1-5). IEEE.

[37] Vellela, S. S., & Balamanigandan, R. An intelligent sleep-awake energy management system for wireless sensor network. Peer-to-Peer Netw. Appl.(2023).

[38] Addepalli, T., Babu, K. J., Beno, A., Potti, B. M. K., Sundari, D. T., & Devana, V. K. R. (2022). Characteristic mode analysis of two port semi-circular arc-shaped multiple-input-multiple-output antenna with high isolation for 5G sub-6 GHz and wireless local area network applications. International Journal of Communication Systems, 35(14), e5257.

[39] Srija, V., & Krishna, P. B. M. (2015). Implementation of agricultural automation system using web & gsm technologies. International Journal of Research in Engineering and Technology, 04 (09), 385-389.

[40] Potti, D. B., MV, D. S., & Kodati, D. S. P. (2015). Hybrid genetic optimization to mitigate starvation in wireless mesh networks. Hybrid Genetic Optimization to Mitigate Starvation in Wireless Mesh Networks, Indian Journal of Science and Technology, 8(23).

[41] Potti, B., Subramanyam, M. V., & Prasad, K. S. (2013). A packet priority approach to mitigate starvation in wireless mesh network with multimedia traffic. International Journal of Computer Applications, 62(14).

[42] Potti, B., Subramanyam, M. V., & Satya Prasad, K. (2016). Adopting Multi-radio Channel Approach in TCP Congestion Control Mechanisms to Mitigate Starvation in Wireless Mesh Networks. In Information Science and Applications (ICISA) 2016 (pp. 85-95). Springer Singapore.

[43] S Phani Praveen, Sai Srinivas Vellela, Dr. R. Balamanigandan, "SmartIris ML: Harnessing Machine Learning for Enhanced Multi-Biometric Authentication", Journal of Next Generation Technology (ISSN: 2583-021X), 4(1), pp.25-36 . Jan 2024.