Advances in Computer Science and Information Technology (ACSIT)
p-ISSN: 2393-9907; e-ISSN: 2393-9915; Volume 4, Issue 1; January-March, 2017, pp. 62-68

http://www.krishisanskriti.org/Publication.html

A Novel Reversible Color Image Covert Communication using SVM Classifier

Chaitra chandanaN[1], Anitha Devi M.D[2], Dr. M.Z. Kurian[3] and K.B. Shiva Kumar[4]

[1]PG Student, Department of ECE, SSIT, Tumkur, Karnataka, India
[2,3]Department of ECE, SSIT, Tumkur, Karnataka, India
[4]Department of TCE, SSIT, Tumkur, Karnataka, India

Abstract—*This paper proposes a novel reversible image data hiding scheme over encrypted domain. Data embedding is achieved through a public key modulation mechanism, in which access to the secret encryption key is not needed. At the decoder side, a powerful two-class SVM classifier is designed to distinguish encrypted and non-encrypted image patches, allowing us to jointly decode the embedded message and the original image signal. Compared with the state-of the-art methods, the proposed approach provides higher embedding capacity and is able to perfectly reconstruct the original image as well as the embedded message. Extensive experimental results are provided to validate the superior performance of our scheme.*

Index Terms*: Feature Extraction, Reversible Image Data Hiding (RIDH), Signal Processing Over Encrypted Domain, SVM.*

1. INTRODUCTION

Reversible image data hiding (RIDH) is a special category of data hiding technique, which ensures perfect reconstruction of the cover image upon the extraction of the embedded message. The reversibility makes such an image data hiding approach particularly attractive in the critical scenarios, e.g., military and remote sensing, medical image sharing, law forensics, and copyright authentication, where high fidelity of the reconstructed cover image is required. The majority of the existing RIDH algorithms are designed over the plaintext domain, namely, the message bits are embedded into the original unencrypted images. The early works mainly utilized the lossless compression algorithm to compress certain image features, to vacate room for message embedding. However, the embedding capacity of this type of method is rather limited and the incurred distortion on the watermarked image is severe. Histogram shifting based technique, initially designed by Ni *et al*, is another class of approach achieving better embedding performance through shifting of the histogram of some image features. The latest difference expansion-based schemes and the improved prediction error expansion-based strategies were shown to be able to offer the state-of-the-art capacity–distortion performance.

1.1. OBJECTIVE

In this project, 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 XOR operations, without the need of accessing the secret encryption key.

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.

1.2. METHODOLOGY

To differentiate encrypted and original unencrypted image blocks, we here design a feature vector $\rho = (H, \sigma, \mathbf{V})$, integrating the characteristics from multiple perspectives. Here, H is a tailored entropy indicator, σ is the SD of the block, and $\mathbf{V}$ represents the directional local complexities in four directions. The formation of the above feature elements will be detailed as follows.

1.3. PROBLEM STATEMENT

Reversible image data hiding (RIDH) is a special category of data hiding technique, which ensures perfect reconstruction of the cover image upon the extraction of the embedded message. The reversibility makes such an image data hiding approach particularly attractive in the critical scenarios, e.g., military and remote sensing, medical image sharing, law forensics, and copyright authentication, where high fidelity of the reconstructed cover image is required.

1.4. PROBLEM SOLUTION

The majority of the existing RIDH algorithms are designed over the plaintext domain, namely, the message bits are embedded into the original unencrypted images. The early works mainly utilized the lossless compression algorithm to compress certain image features, to vacate room for message embedding. However the problem arises because of the embedding capacity of this type of method is rather limited

and the incurred distortion on the watermarked image is severe.

2. LITERATURE SURVEY

W. Puech, M. Chaumont et.al [1] proposed some recent attempts which were made on embedding message bits into the encrypted images. They used a simple substitution method to insert additional bits into AES encrypted images. Local standard deviation (SD) was then exploited at the decoder side to reconstruct the original image.

X. Zhang [2] designed a method to embed additional message bits into stream cipher encrypted images by flipping three LSBs of half of the pixels in a block. The data extraction can be performed by utilizing the local smoothness inherent to natural images.

W. Hong, T.-S. Chen et.al [3] improved the above method through a side match technique. As local smoothness does not always hold for natural images, dataextraction errors can be observed in the high-activity regions.

X. Zhang [4] proposed a separable RIDH method such that the protection scopes of data hiding key and encryption key are gracefully separated.

X. Zhang, Z. Qian et.al [5] extended the lossless compression-based RIDH approach to the encrypted domain, namely, losslessly compress half of the fourth LSBs of the encrypted image via LDPC code to create space for data hiding.

K. Ma, W. Zhang et.al [6] suggested a new embedding method by reserving room before encryption with a traditional reversible image watermarking algorithm. Significant improvements on embedding performance can be achieved by shifting partial embedding operations to the encryption phase.

3. SYSTEM DESIGN

3.1. ENCODING

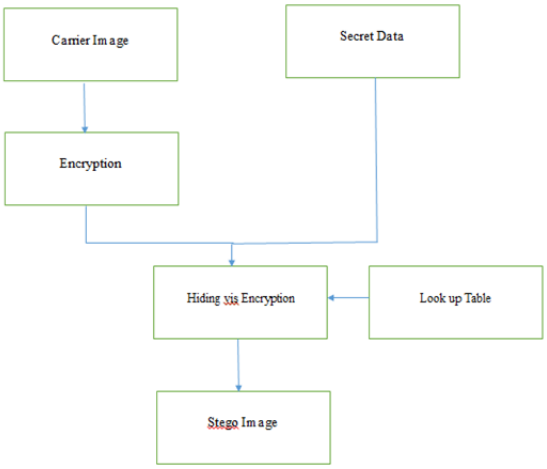

FIG 3.1: ENCODING

3.2. DECODING

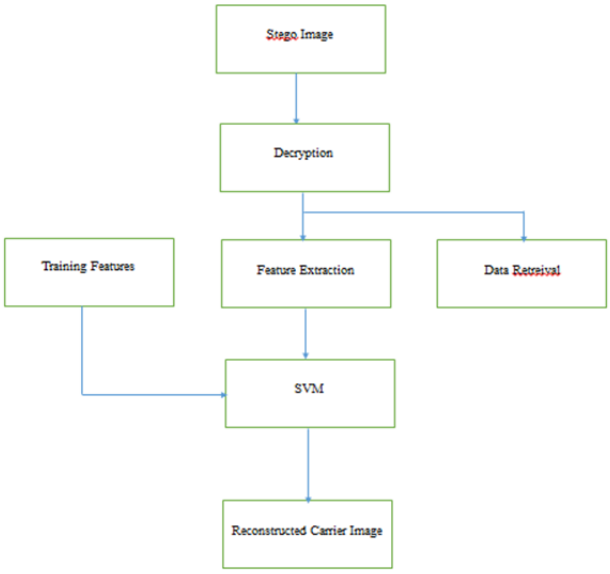

FIG 3.2: Decoding

3.3. PROPOSED RIDH SCHEME OVER ENCRYPTED DOMAIN

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 [26]. 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. When stream cipher is employed, the encrypted image is generated by

$$[[f]] = \mathrm{Enc}(f, K) = f \oplus K \quad (1)$$

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**.

Clearly, the original image can be obtained by performing the following decryption function:

$$\mathbf{f} = \mathrm{Dec}([[\mathbf{f}]], K) = [[\mathbf{f}]] \oplus \mathbf{K} \quad \textbf{(2)}$$

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 $M \times N$, 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 $n \cdot B$ bits. To enable efficient embedding, we propose to use $S = 2n$ binary *public* keys **Q**0,**Q**1, . . ., **Q**$S-1$, each of which is of length $L = M \times N \times 8$ bits. All **Q**j 's, for $0 \le j \le S - 1$, 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 [27] 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 [28] showed that an upper bound on the minimum Hamming distance can be given as follows. First, determine two integers $m1$ and $m2$ by

$$\sum_{i=0}^{m_1} \binom{L}{i} \le \frac{2^L}{S} < \sum_{i=0}^{m_1+1} \binom{L}{i} \quad (3)$$

$$\sum_{i=0}^{m_2} \binom{L-1}{i} \le \frac{2^{L-1}}{S} < \sum_{i=0}^{m_2+1} \binom{L-1}{i} \quad (4)$$

where $\_Li\_ = (L!/i\,!(L - i)!)$. It can be shown that both $m1$ and $m2$ are unique. Then, the minimum Hamming distance among all **Q**j 's satisfies

$$d\min \le \max\{2m1 + 1,\ 2m2 + 2 \quad \textbf{(5)}$$

The schematic diagram of the proposed message embedding algorithm over encrypted domain is shown in Fig. 3.3. 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.

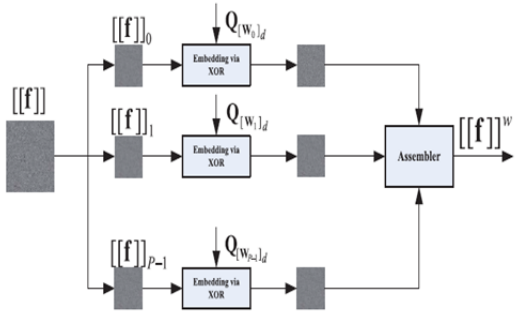

Fig 3.3: Schematic of data hiding over encrypted domain.

For simplicity, we assume that the number of message bits to be embedded is $n \cdot A$, where $A \le B$ and B is the number of blocks within the image. The steps for performing the message embedding are summarized as follows.

Step 1: Initialize block index $i = 1$.

Step 2*:* Extract n bits of message to be embedded, denoted by **W**i .

Step 3: 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 **Q**2.

Step 4: Embed the length-n message bits **W**i into the i th block via

$$[[\mathbf{f}]]\mathrm{wi} = [[\mathbf{f}]]\mathrm{i} \oplus \mathbf{Q}[\mathbf{W}\mathrm{i}\ ]\mathrm{d} \quad \textbf{(6)}$$

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 computation 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 [18], [19], upon trivial modifications, can still ensure embedding security even if the data hiding key is eliminated. In [18], if we fix the way of partitioning a block into $S0$ and $S1$ (namely, do not use data hiding key to randomize the block partitioning), then an attacker still cannot compute the fluctuation function [18, eq. (10)] so as to decode the embedded message. This is because an attacker does not access to the secret encryption key K. In other words, the protection mechanism 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 [15].

Though the possibility of removing the data hiding key holds for all non separable RIDH schemes over encrypted domain, it has never been pointed out in the existing work. It can be witnessed by the fact that all the existing RIDH schemes, including separable and non separable ones, involve a data hiding key that has to be shared and managed between the data hider and the recipient. In addition to identifying this property, we, in Section VI, will exploit the message indistinguishability to prove that the removal of data hiding key will not hurt the embedding security.

Before presenting the data extraction and image decryption methods, let us first investigate the features that can be used to discriminate encrypted and non encrypted image blocks. The classifier designed according to these features will be shown to be crucial in the proposed joint data extraction and image decryption approach.

3.4. FEATURE SELECTION FOR DISCRIMINATING ENCRYPTED AND NON ENCRYPTED IMAGE BLOCKS

To differentiate encrypted and original unencrypted image blocks, we here design a feature vector $\rho = (H, \sigma, \mathbf{V})'$, integrating the characteristics from multiple perspectives. Here, H is a tailored entropy indicator, σ is the SD of the block, and $\mathbf{V}$ 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 available 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.

$$\hat{f} = \left\lfloor \frac{MN \cdot f}{256} \right\rfloor \qquad (7)$$

where f and $\hat{f}$ denote the original and the quantized pixel values, respectively. Certainly, $\hat{f}$ falls into the range $[0, MN-1]$. The entropy indicator H based on quantized samples is then given by

$$H = -\sum_{j=0}^{MN-1} p(j) \log p(j) \qquad (8)$$

where $p(j)$ is the empirical probability of j in the quantized block.

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

$$\sigma = \sqrt{\frac{1}{MN} \sum_{j} (\mathbf{f}(j) - \mu)^2} \qquad (9)$$

where $\mathbf{f}(j)$ is the jth pixel in the block and $\mu = (1/MN) \Sigma j \, \mathbf{f}(j)$ 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 dispersiveness and denseness can be better reflected.

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 $\mathbf{V} = (v1, v2, v3, v4)'$, where

$$\begin{aligned} v_1 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{ne})| \\ v_2 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_e)| \\ v_3 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_{se})| \\ v_4 &= \sum_j |\mathbf{f}(j) - \mathbf{f}(j_s)| \end{aligned} \qquad (10)$$

where $\mathbf{f}(j_{ne})$, $\mathbf{f}(j_e)$, $\mathbf{f}(j_{se})$, and $\mathbf{f}(j_s)$ represent the neighbors in the 45° (northeast), 0° (east), −45° (southeast), and −90° (south) directions, relative to $\mathbf{f}(j)$, as shown in Fig. 3.4.

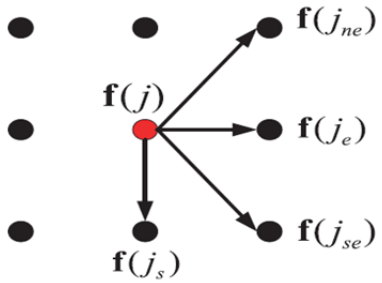

Fig 3.4: Illustration of the neighbors of f(j).

Upon the determination of the feature vector ρ, we train a two-class SVM classifier with RBF (Gaussian) kernel [29] taking the form

$$\mathrm{Ker}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|}. \quad (11)$$

The 0-class and 1-class correspond to the unencrypted and encrypted image blocks, respectively. Here, the training image set consists of 100 images of size 512×512, with a wide variety of characteristics including natural scenes, artificial images, synthetic images, and textual images. The *offline* trained SVM classifier will be used to discriminate the encrypted and non encrypted image patches in the process of data extraction and image decryption.

3.5. JOINT DATA EXTRACTION AND IMAGE DECRYPTION

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 $[[\mathbf{f}]]^w$, 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 $[[\mathbf{f}]]w$ with the encryption key stream $\mathbf{K}$ and obtains

$$\mathbf{f}^w = [[\mathbf{f}]]^w \oplus \mathbf{K}. \quad (12)$$

The resulting $\mathbf{f}w$ is then partitioned into a series of non overlapping blocks $\mathbf{f}_i^w$'s of size $M \times N$, similar to the operation conducted at the embedding stage. From (6), we have

$$\mathbf{f}_i^w = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d}. \quad (13)$$

The joint data extraction and image decryption now becomes a blind signal separation problem as both $\mathbf{W}i$ and $\mathbf{f}i$ are unknowns. Our strategy of solving this problem is based on the following observation: $\mathbf{f}i$, as the original image block, very likely exhibits certain image structure, conveying semantic information. Note that $\mathbf{Q}[\mathbf{W}i\ ]_d$ must match one of the elements in $\boldsymbol{Q} = \{\mathbf{Q0}, \mathbf{Q1}, \ldots, \mathbf{QS-1}\}$. Then, if we XOR $\mathbf{f}_i^w$ with all $\mathbf{Q}j$'s, one of the results must be $\mathbf{f}i$, 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 $\mathbf{f}i$.

More specifically, we first create S decoding candidates by XORing f_i^w with all the S possible public keys $\mathbf{Q}0, \mathbf{Q}1, \ldots, \mathbf{Q}S-1$

$$\begin{aligned}
\mathbf{f}_i^{(0)} &= \mathbf{f}_i^w \oplus \mathbf{Q}_0 = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_0 \\
\mathbf{f}_i^{(1)} &= \mathbf{f}_i^w \oplus \mathbf{Q}_1 = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_1 \\
&\vdots \\
\mathbf{f}_i^{(S-1)} &= \mathbf{f}_i^w \oplus \mathbf{Q}_{S-1} = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_{S-1}. \quad (14)
\end{aligned}$$

As mentioned earlier, one of the above S candidates must be $\mathbf{f}i$, while the others can be written in the form

$$\mathbf{f}_i^{(t)} = \mathbf{f}_i \oplus \mathbf{Q}_{[\mathbf{W}_i]_d} \oplus \mathbf{Q}_t \quad (15)$$

where $t \neq [\mathbf{W}i\ ]d$.

The result $\mathrm{fi}^{(t)} = \mathrm{Enc}(\mathbf{f}i, \mathbf{Q}[\mathbf{W}i\ ]d \oplus \mathbf{Q}t)$ corresponds to an encrypted version of $\mathbf{f}i$ with equivalent key stream being $\mathbf{Q}[\mathbf{W}i\ ]d \oplus \mathbf{Q}t$. Note that all the public keys $\mathbf{Q}j$'s, for $0 \leq j \leq S-1$, are designed to have maximized minimum Hamming distance, and the upper bound is given in (5). Hence, $\mathbf{fi}^{(t)}$ tends to lose the image structural information, making it appear random.

To identify which candidate corresponds to $\mathbf{f}i$, we apply the designed two-class SVM classifier to these S candidates. Let $\mathbf{r} = (r0, r1, \ldots, rS-1)$ 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

$$\mathbf{W}_i = [j]_2 \quad (16)$$

where $[\,j\,]2$ denotes the length-n binary representation of j and $n = \log 2\ S$. For example, if $n = 3$ and $j = 7$, then $[\,j\,]_2 = 111$. Upon determining $\mathbf{W}_i$, the original image block can be easily recovered by

$$\mathbf{f}_i = \mathbf{f}_i^w \oplus \mathbf{Q}_{[\mathbf{W}_i]_d}. \quad (17)$$

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:* $\mathbf{f}^{(j)}{}_I = \mathbf{f}i$, while $r\,j = 1$.

2) *Type II Error:* $\mathbf{f}^{(j)}{}_I \neq \mathbf{f}i$, 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:

$$\sum_j r_j < 2^n - 1 \qquad (18)$$

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 [30], as also shown in Fig. 3.5. 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.

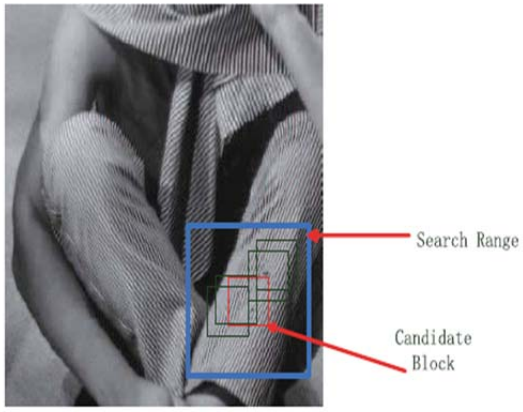

Fig 3.5: Illustration of the error correction mechanism based on image self similarity.

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

$$\mathcal{C} = \begin{cases} \{\mathbf{f}_i^{(j)} | 0 \le j \le S-1\} & \text{Type I error detected} \\ \{\mathbf{f}_i^{(j)} | r_j = 0\} & \text{Type II error detected.} \end{cases} \qquad (19)$$

For any candidate block $\mathbf{f}_i^{(j)}$ in C, we calculate its 2 distances from all the other blocks in a search range $D\backslash\{\mathbf{f}(j)i\}$, where D shares the same center as $\mathbf{f}(ji$ and its size is experimentally determined as $5M \times 5N$. We then can compute the minimum patch distance within the search window

$$d_i^{(j)} = \min_{\mathbf{D} \in \mathcal{D}\backslash\{\mathbf{f}_i^{(j)}\}} \left\| \mathbf{f}_i^{(j)} - \mathbf{D} \right\|_F^2 \qquad (20)$$

where **D** is an arbitrary block of size $M \times N$ within $D\backslash\{\mathbf{f}_i^{(j)}\}$.

Here, 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 $\mathbf{f}(j)\ i$ 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 (16) and (17). The above joint data extraction and image decryption procedures can also be summarized in Fig. 3.6.

Our proposed RIDH scheme over encrypted domain may also be extended to handle compressed and encrypted images, namely, embed watermark into the compressed and encrypted bit stream.

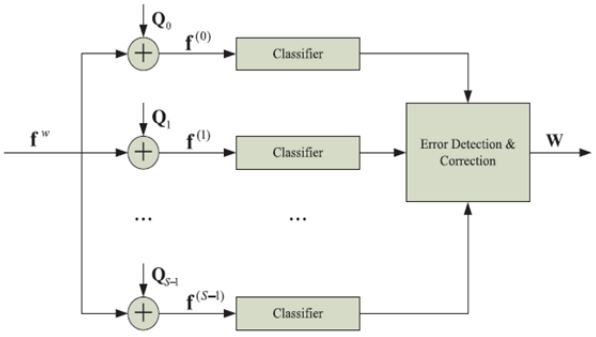

Fig 3.6: Schematic of the data extraction.

Take the JPEG for example. Assume that the encryption is conducted without destroying the structure of JPEG bit stream. For instance, the encryption scheme proposed in [22] can be used to this end. We can XOR the encrypted parts with one of the designed S binary public keys, according to the message bits to be embedded. At the extraction stage, we try all the S possibilities and identify the one that generates structured

image patches in the pixel domain. The embedded message can then be extracted based on the index of the identified public key.

4. APPLICATIONS

- It is used for Secured communication
- It is used for Military Application
- It is also used for Internet application

5. CONCLUSION

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 XOR operations, without the need of accessing the secret encryption key. 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.

REFERENCES

[1] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," *Proc. SPIE*, vol. 6819, pp. 1–9, Feb. 2008.

[2] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Process. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.

[3] W. Hong, T.-S. Chen, and H.-Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Process. Lett.*, vol. 19, no. 4, pp. 199–202, Apr. 2012.

[4] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[5] X. Zhang, Z. Qian, G. Feng, and Y. Ren, "Efficient reversible data hiding in encrypted images," *J. Vis. Commun. Image Represent.* vol. 25, no. 2, pp. 322–328, Feb. 2014.

[6] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[7] Z. Qian, X. Zhang, and S. Wang, "Reversible data hiding in encrypted JPEG bitstream," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1486–1491, Aug. 2014.