[1]Neetha S. S

[2]Dr. Bhuvana J

Reversible Data hiding based on Three-Level Hierarchical Embedding with Histogram Mapping in Encrypted Images

Abstract: - This paper proposes a reversible data-hiding (RDH) approach based on a three-level hierarchical embedding with histogram mapping in encrypted images for a cloud environment. The proposed approach uses two-level encryption followed by the three-level hierarchical embedding process. The two-level encryption uses an intra-inter block permutation with an exclusive OR operation. In the three-level hierarchical embedding two predictors namely strong predictor and weak predictor are estimated. The strong predictor is used in level 1 embedding, while the weak predictor is used in level 2 embedding. A histogram mapping is then performed on the location map obtained during the level 1 and level 2 embedding process and is then compressed using the Huffman coding. The compressed location map information is then further embedded as the level 3 embedding process. In the extraction phase, the compressed location map information is first extracted, then the level 2 data followed by the level 1 data. The evaluation of the proposed data embedding approach is done using the Bossbase and Bows-2 datasets that result in an embedding rate of 3.806 bpp and 3.513 bpp respectively.

Keywords: Image encryption, Permutation, Reversible data hiding, Hierarchical embedding, Encrypted images

I. INTRODUCTION

Due to the availability of huge storage, cloud storage [1] has become popular and the multimedia content which is uploaded to the cloud storage is highly increased. However, there are several security issues such as integrity, authentication, and confidentiality. Therefore, it is mandatory to protect multimedia content against such security issues [2], [3]. To preserve the privacy content of the image, data encryption, and data hiding is an effective approaches. Several data-hiding approaches [4] aim in embedding secret data with small distortion on a cover image. This scheme fails to recover the cover image by only extracting the hidden data. It is necessary to recover the extracted cover image content along with the secret data in several applications [5] such as law enforcement and medical images. Thus the reversible data hiding provides a lossless recovery of the cover image and secret data. Lossless data embedding algorithms such as prediction error expansion [6]-[7], histogram shifting [8]-[9] difference expansion [10]-[11] and lossless compression [12] are commonly used schemes. These approaches provide a high embedding rate, for plain images, since the variation between the neighboring pixels is less. However, these approaches provide a lower performance on encrypted images since the variation between the neighboring pixels is random and abrupt.

In a cloud environment, the owner, hider, and receiver are the three users. Based on the encryption key, the original image is encrypted and uploaded to the cloud by the owner, the encryption may be done either in RRBE (Reserving room before encryption), or VRAE (Vacating room after encryption). The neighbor pixel correlation is used to estimate the prediction error, which is used to hide the secret data. The prediction error values will be higher in the case of an encrypted image, while they will be lower in the case of a plain image. A cumulative data-hiding approach was proposed by Zhang et al. [13] where the original image is fragmented into non-overlapping blocks.

[1] School of CS & IT Jain (Deemed-To-Be) University Bangalore, Karnataka, India

ssneetha.kala@gmail.com

[2]School of CS & IT Jain (Deemed-To-Be) University Bangalore, Karnataka, India

bhuvi.jayabalan@gmail.com

Two neighboring pixels are masked using a pseudo-random mask value. This encryption preserves spatial redundancy where the difference between the neighbor values remains the same before and after encryption. The encryption is not performed on the same pixel values due to the neighbor pixel differences. This non-encrypted pixel can lead to information leakages.

In the approach [14], a few of the prediction errors are estimated before the encryption and the prediction errors are used in data embedding. These prediction errors are not encrypted and the remaining pixel values are encrypted. The correlation of neighboring pixels is used to embed the data [15], this includes four major processes namely block subdividing the image, grouping the pixel, labeling the pixel, and the embedding process. Permutation of blocks and modulation of pixels are involved in the encryption process. The modulation of the pixel is performed by adding random integer values with the pixel values of the selected block. Due to the addition of a similar random integer, the pixel difference is also proportional to the plain image which leads to leakage of information. The data extraction accuracy [16] was improved using a two-class support vector machine (SVM) classifier. This approach uses different public keys in the encryption process, where the keys are chosen based on the maximum values of the minimum hamming distance. The SVM classifier present in the receiver classifies the pixels after embedding as non-encrypted and encrypted blocks from which the original image and hidden data are reconstructed.

A separable RDH [17] uses the LSBs of every encrypted block, whereas the scheme [18] used MSBs to embed the data. The shifting of pixel histogram is performed on a single level and is used in data embedding [19], where the embedding capacity is improved using a multi-level shifting of pixel histogram. A parametric binary tree labeling [20] is proposed by Yi et al. which exploits the spatial redundancy and preserves the spatial correlations. The author Yu et al. [21] proposed a hierarchal process of embedding that occurs in two stages. In the first stage hierarchal label map (HLM) is constructed using the bit planes of the cover images, where the prediction approach is used to estimate the HLM. In the second stage, three different kinds of prediction errors are found based on the magnitude namely large, medium, and small. The prediction errors with small or large magnitudes are used to hide the secret data. The author Yin et al. [22] used Huffman coding and multi-MSB prediction in data embedding. In this approach, an adaptive prediction is used to estimate the prediction from the multi-MSB of every pixel. The embedding was done on the original image using the Huffman coding.

The author Qiu et al. [23] used entropy encoding and pixel prediction in data embedding. The author presents two data embedding approaches, one of the approaches is suitable for reserving room before encryption (RRBE) model, where the embedding room formation process is used to generate an embedding room. The rest of the region is encrypted by the image owner. While embedding the data, the embedding room was identified by the data hider, and the data is embedded. The second approach uses the vacating room by encryption (VRBE). In this approach permutation encryption and improved block modulation are used to encrypt the image. The data hider then constructs the embedding room from the encrypted image and then hides the data. The author Ammar et al. [24] used a local difference predictor for reversible data embedding, where non-overlapping blocks are first constructed and the pixels are formed as leader pixels and followers. The local difference predictor is estimated using the intensity of leader and follower pixels. The non-overlapping blocks are then labeled based on the hiding capacity of the blocks.

Even though several schemes that are discussed above provide a high embedding rate, the embedding rate should be still improved due to the requirement of storing huge data in the images which are caused due to the availability of more images in the cloud storage. The paper has the following contributions.

(i) The paper proposes a two-level encryption/decryption which uses three different sub-keys for encrypting/decrypting the image.
(ii) The paper proposes a three-level hierarchical data embedding/ extraction algorithm, that embeds/extracts the data using a strong predictor and weak predictor in the level-1 and level-2 stages.
(iii) The paper also proposes a location map compression approach that uses a histogram mapping approach followed by Huffman compression.
(iv) Finally, the evaluation of the algorithm was done using the datasets namely Bossbase, and Bows-2 using the metrics namely, PSNR, embedding rate, and SSIM.

The following sections of the paper can be highlighted as follows. Section 2 shows the working of the proposed three-level hierarchical data embedding algorithm. Section 3 shows the experimental results and analysis of the proposed three-level hierarchical data embedding algorithm and section 4 concludes the work.

II. PROPOSED METHOD

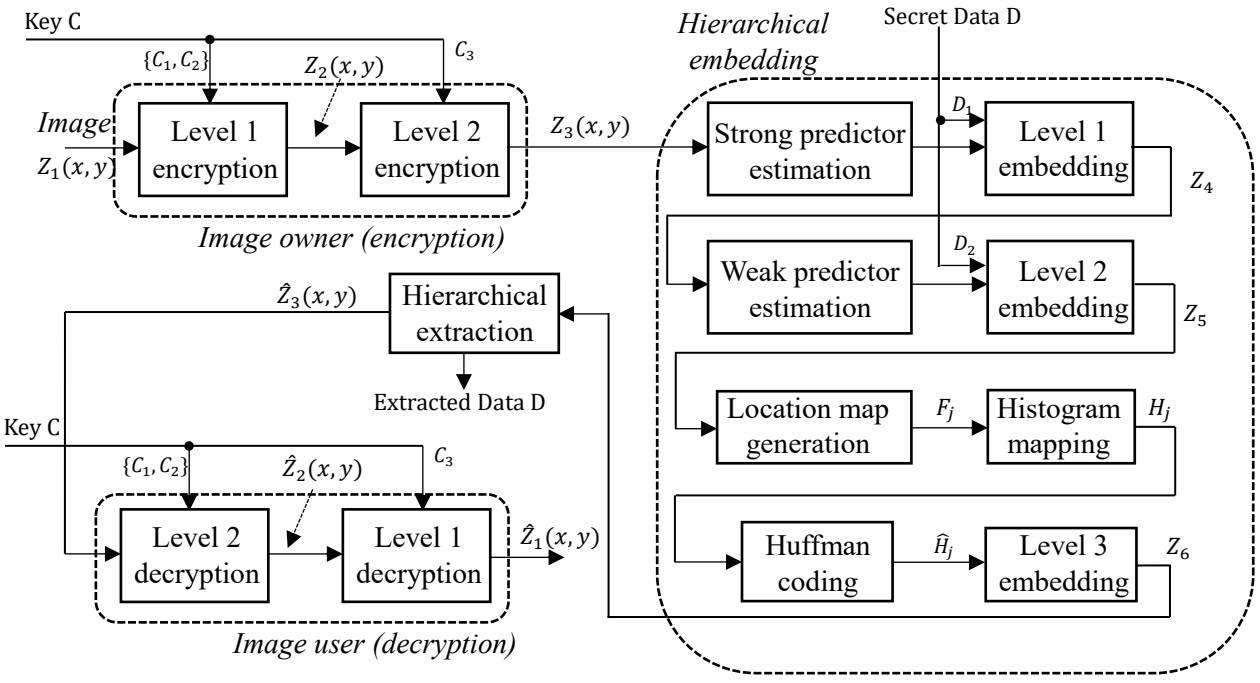


Figure 1: Flow diagram illustration of the proposed three-level hierarchical data embedding

The proposed encrypted image-based reversible data hiding includes three major processes such as (i) Image Encryption and Decryption (ii) Data hiding in encrypted images (iii) Data extraction in encrypted images. The flow diagram representation of the proposed algorithm is illustrated in Fig. 1.

2.1 Image encryption/ decryption

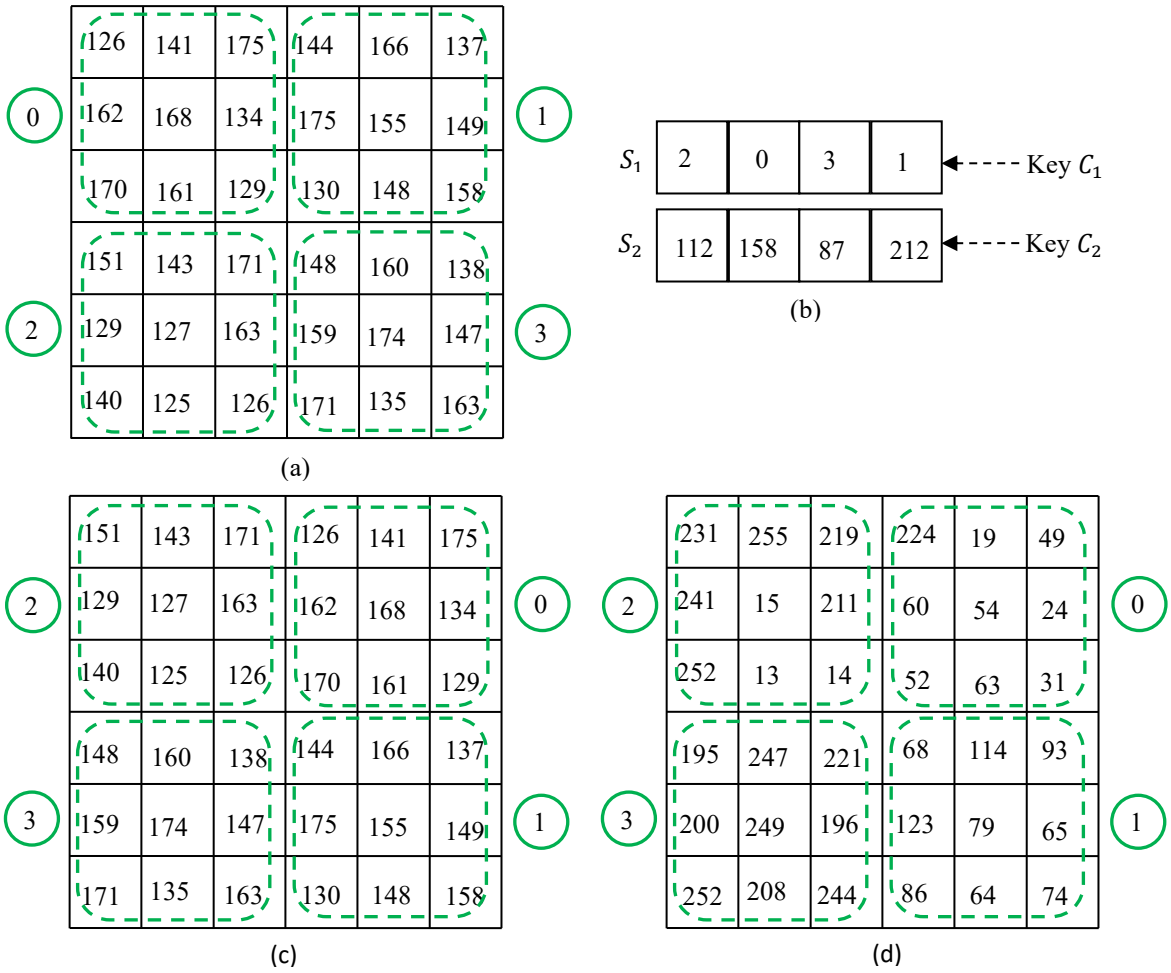


Figure 2: Level 1 encryption (a) Original image (b) Representation of sequence S_1 and S_2 (c) Block permutation using S_1 (d) EX-OR operation between S_2 and permuted block

Let $Z_1(x, y)$ be the plain image. The encryption of the image is performed in two levels namely level 1 and level 2 permutation. Initially, the plain image is subdivided into non-overlapping blocks having a size of 3×3. Let the size of the plain image be $U \times V$. Therefore the total number of the non-overlapping block is $N_b = \frac{(U \times V)}{9}$. Let the key $C = \{C_1, C_2, C_3\}$ has three sub-keys namely C_1 C_2 and C_3. Using the key C_1, N_b number of pseudo integers random sequence S_1 is generated. The position of the non-overlapping blocks is scrambled using the generated pseudo-random sequence S_1 . The key C_2 is used to generate a pseudo-random integer sequence S_2 whose intensities lie between 0 and 255, such that each group is assigned a single integer value. Exclusive OR operation (XOR) is performed between the pseudo-random integer value of the group with the scrambled block pixels values to obtain the level 1 encrypted image $Z_2(x, y)$. Fig.2 illustrates the level 1 encryption process, whereas Fig 2(a) shows an example of an image with four 3×3 blocks. Therefore using key C_1, four pseudo-random sequences S_1 are generated. Using the sequence S_1, the blocks are permuted as shown in Fig 2(c). Also using the key C_2 four random integers sequence S_2 are generated as shown in Fig 2(b). Using the random integers of the group S_2 and the scrambled group, the bit-XOR operation is performed as illustrated in Fig 2(d). In level 2 encryption the position of pixels in the 3×3 blocks of $Z_2(x, y)$ are rotated in a clockwise or counter clock-wise direction using

the key C_2. To attain this the N_b number of a pseudo-random integer sequence S_3 are generated whose magnitude is between 0 to 7. Let the pixel values of the block of $Z_2(x,y)$ be represented by the pixels values $\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix}$. Based on the key C_2, the neighborhood of the pixels is rotated in a clockwise direction as illustrated in Fig 3. As a result of level 2 encryption the spatial relationship of the pixel within the neighborhood is preserved.

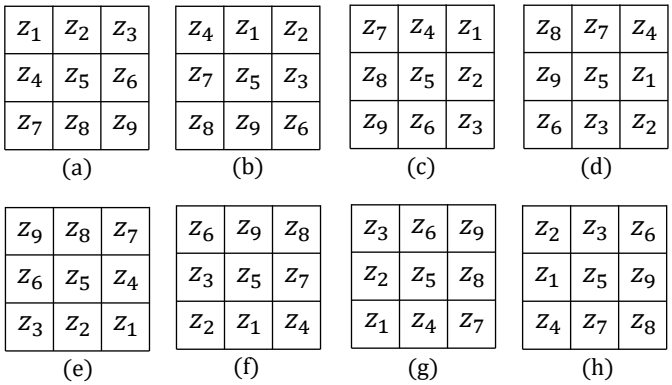


Figure 3: Representation of scrambling position in level 2 encryption for different values of S_3 (a) $S_3 = 0$ (b) $S_3 = 1$ (c) $S_3 = 2$ (d) $S_3 = 3$ (e) $S_3 = 4$ (f) $S_3 = 5$ (g) $S_3 = 6$ (h) $S_3 = 7$

For decrypting the image, level-2 decryption is first performed followed by level-2 decryption. The key used for decryption must be the same as that of encryption for obtaining the exact plain image $\hat{Z}_1(x,y) = Z_1(x,y)$.

2.2 Data hiding in encrypted images

The proposed hierarchical data hiding in the encrypted image is performed in three levels namely level 1, level 2, and level 3 data hiding. In level 1 data hiding a strong predictor is used to embed the data, while in level 2 data hiding a weak predictor is used to embed the data. The level 1 predictor is represented as a strong predictor since the predictor values are estimated completely from the actual pixel values. The level 2 predictor is termed a weak predictor since the predictor values are estimated from the newly generated pixel values after embedding the data in level 1. The weak predictor indirectly depends on the data that has been embedded in level 1. The level 3 data hiding is used to embed the location map information. Let the level 2 encrypted image be represented by the 3×3 matrix $Z_3(x,y) = \begin{bmatrix} v_1 & p_1 & v_2 \\ p_4 & p_0 & p_2 \\ v_4 & p_3 & v_3 \end{bmatrix}$ as illustrated in Fig 4(a). In level 1 the four data values of $d_1 = \{d_{1,1}, d_{1,2}, d_{1,3}, d_{1,4}\}$ is embedded in the pixel locations $v_i = \{v_1, v_2, v_3, v_4\}$ respectively.

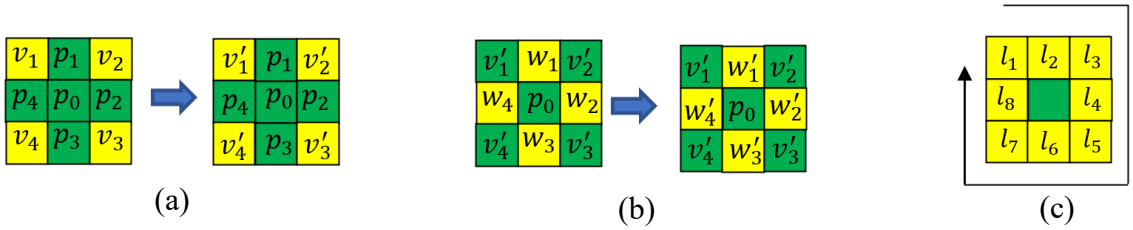


Fig 4: Representation of Level 1 embedding, Level 2 embedding, and location map formation (a) Level 1 embedding (b) Level 2 embedding (c) location map

The predictor [22] for the pixel v_1 can be estimated using the pixels values p_1, p_0 and p_4.

$$\hat{v}_1 = \begin{cases} \max(p_1, p_4) & if \quad p_0 \le \min(p_1, p_4) \\ \min(p_1, p_4) & if \quad p_0 \ge \max(p_1, p_4) \\ p_1 + p_4 - p_0 & otherwise \end{cases} \tag{1}$$

The predictor for the pixel v_2 can be estimated using the pixel values p_1, p_0 and p_2.

$$\hat{v}_2 = \begin{cases} \max(p_1, p_2) & if \quad p_0 \le \min(p_1, p_2) \\ \min(p_1, p_2) & if \quad p_0 \ge \max(p_1, p_2) \\ p_1 + p_2 - p_0 & otherwise \end{cases} \tag{2}$$

The predictor for the pixel v_3 can be estimated using the pixel values p_2, p_0 and p_3.

$$\hat{v}_1 = \begin{cases} \max(p_2, p_3) & if \quad p_0 \le \min(p_2, p_3) \\ \min(p_2, p_3) & if \quad p_0 \ge \max(p_2, p_3) \\ p_2 + p_3 - p_0 & otherwise \end{cases} \tag{3}$$

The predictor for the pixel v_4 can be estimated using the pixel values p_3, p_0 and p_4.

$$\hat{v}_1 = \begin{cases} \max(p_3, p_4) & if \quad p_0 \le \min(p_3, p_4) \\ \min(p_3, p_4) & if \quad p_0 \ge \max(p_3, p_4) \\ p_3 + p_4 - p_0 & otherwise \end{cases} \tag{4}$$

Since the above strong predictors are estimated from actual pixel values of the image, the predictors $\hat{v}_1$, $\hat{v}_2$, $\hat{v}_3$ and $\hat{v}_4$ are termed as strong predictors. Using the predictor $\hat{v}_i = \{\hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{v}_4\}$ and the prediction error that corresponds to the pixel values $v_i = \{v_1, v_2, v_3, v_4\}$, the prediction error can be estimated as

$$e_i = v_i - \hat{v}_i \tag{5}$$

where $i = \{1,2,3,4\}$. The prediction error e_i is expanded as

$$\hat{e}_i = L \times e_i + D_{1,i} \tag{6}$$

Where $D_{1,i}$ represent the decimal data equivalent to the L bit data $d_{1,i}$. (L is also termed as group length). The marked pixel that corresponds to the pixel values v_i can be estimated as

$$v_i' = \hat{v}_i + \hat{e}_i \tag{7}$$

The weak predictor $\hat{w}_i = \{\hat{w}_1, \hat{w}_2, \hat{w}_3, \hat{w}_4\}$ can be estimated using the marked pixel values values v_1', v_2', v_3', v_4' and pixel p_0

$$\hat{w}_1 = \begin{cases} \max(v_1', v_2') & if \quad p_0 \le \min(p_1, p_4) \\ \min(p_1, p_4) & if \quad p_0 \ge \max(p_1, p_4) \\ p_1 + p_4 - p_0 & otherwise \end{cases} \tag{8}$$

$$\hat{w}_1 = \begin{cases} \max(v_1', v_2') & if \quad \lfloor (v_1' + v_2' + p_0)/3 \rfloor \le \min(v_1', v_2') \\ \min(v_1', v_2') & if \quad \lfloor (v_1' + v_2' + p_0)/3 \rfloor \ge \max(v_1', v_2') \\ v_1' + v_2' - \lfloor (v_1' + v_2' + p_0)/3 \rfloor & otherwise \end{cases} \tag{9}$$

$$\widehat{w}_2 = \begin{cases} \max(v_3', v_2') & if \quad \lfloor (v_3' + v_2' + p_0)/3 \rfloor \leq \min(v_3', v_2') \\ \min(v_3', v_2') & if \quad \lfloor (v_3' + v_2' + p_0)/3 \rfloor \geq \max(v_3', v_2') \\ v_3' + v_2' - \lfloor (v_3' + v_2' + p_0)/3 \rfloor & otherwise \end{cases} \quad (10)$$

$$\widehat{w}_3 = \begin{cases} \max(v_3', v_4') & if \quad \lfloor (v_3' + v_4' + p_0)/3 \rfloor \leq \min(v_3', v_4') \\ \min(v_3', v_4') & if \quad \lfloor (v_3' + v_4' + p_0)/3 \rfloor \geq \max(v_3', v_4') \\ v_3' + v_4' - \lfloor (v_3' + v_4' + p_0)/3 \rfloor & otherwise \end{cases} \quad (11)$$

$$\widehat{w}_4 = \begin{cases} \max(v_1', v_4') & if \quad \lfloor (v_1' + v_4' + p_0)/3 \rfloor \leq \min(v_1', v_4') \\ \min(v_1', v_4') & if \quad \lfloor (v_1' + v_4' + p_0)/3 \rfloor \geq \max(v_1', v_4') \\ v_1' + v_4' - \lfloor (v_1' + v_4' + p_0)/3 \rfloor & otherwise \end{cases} \quad (12)$$

Since the above weak predictors $\widehat{w}_i = \{\widehat{w}_1, \widehat{w}_2, \widehat{w}_3, \widehat{w}_4\}$ are estimated from marked pixel values which also depend on the level 1 embedded data, the predictors $\widehat{w}_1$, $\widehat{w}_2$, $\widehat{w}_3$ and $\widehat{w}_4$ are termed weak predictors. Using the predictor $\widehat{w}_i = \{\widehat{w}_1, \widehat{w}_2, \widehat{w}_3, \widehat{w}_4\}$, the prediction error that corresponds to the pixel values $w_i = \{w_1, w_2, w_3, w_4\}$ can be estimated as

$$E_i = w_i - \widehat{w}_i \quad (13)$$

where $i = \{1,2,3,4\}$. The prediction error E_i is expanded as

$$\hat{E}_i = L \times E_i + D_{2,i} \quad (14)$$

Where $D_{2,i}$ represent the decimal data equivalent to the L bit data $d_{2,i}$. The marked pixel that corresponds to the pixel values w_i can be estimated as

$$w_i' = \widehat{w}_i + \hat{E}_i \quad (15)$$

The level 1 embedding and level 2 embedding processes are illustrated in Fig. 4.

(a) Histogram mapping and Huffman compression

Let $l = \{l_1, l_2 \ldots . . l_8\}$ be the eight location map values, where $l \in [0,1]$. Here $l = 0$, resembles that there is no overflow or underflow and the data has been embedded in the corresponding location, and $l = 1$, resembles that there is either overflow or underflow and the data was not embedded in the corresponding location. The location map for each neighborhood is converted to decimal values. Let the decimal location map that corresponds to the level 2 embedded image be $F_j = \{F_1, F_2 \ldots \ldots F_{N_b}\}$, Since F_j is the decimal equivalent of an 8-bit binary sequence, $F_j \in [0,255]$. The histogram is then estimated for the sequence F_j. The decimal value that has the highest histogram is mapped to the least intensity, while the decimal value that has the highest histogram is mapped to the least intensity. An example of histogram mapping [25] is illustrated in Fig. 5. The mapped histogram sequence H_j is then compressed using Huffman compression [26] to obtain the sequence $\widehat{H}_j$. The compressed mapped histogram sequence $\widehat{H}_j$ and sorted histogram index is embedded similarly to level 1 embedding which completes the level 3 embedding.

2	7	0	5	7	6	7
7	6	6	4	3	2	0
7	1	2	7	6	7	5
5	3	0	2	7	4	3
1	5	6	3	2	6	7
2	6	5	3	4	0	7
6	2	3	5	0	3	6

(a)

2	0	5	4	0	1	0
0	1	1	6	3	2	5
0	7	2	0	1	0	4
4	3	5	2	0	6	3
7	5	1	3	2	1	0
2		4	3	6	5	0
1	2	3	4	5	3	1

(b)

Location map values	0	1	2	3	4	5	6	7
Histogram	5	2	7	7	3	6	9	10
Sorted histogram	5	7	2	3	6	4	1	0

(c)

Fig 5: Example of histogram mapping (a) Location map before histogram mapping (b) Location map after histogram mapping (c) Sorting of histogram values

For attaining higher embedding rates, the 3-level data embedding can be recursively performed using the marked encrypted images estimated in each stage. Let R represent the recursion length.

1.3 Data Extraction

In the extraction phase, the compressed location map information along with the sorted histogram index is initially extracted. The location map information is decompressed using the Huffman decompression and then the histogram demapping is performed. Level 2 and Level 1 data extraction is then performed to extract the data and recover the actual pixel value. The extraction is repeated R times. The expanded error in level-2 embedding can be estimated from the marked pixel value w_i' and the predictor $\widehat{w}_i$ as

$$\hat{E}_i = w_i' - \widehat{w}_i \tag{16}$$

From the expanded error $\hat{E}_i$ the hidden data in level-2 can be estimated

$$D_{2,i} = \hat{E}_i - L\left\lfloor\frac{\hat{E}_i}{L}\right\rfloor \tag{17}$$

The prediction error can be estimated from the expanded error $\hat{E}_i$ and data $D_{2,i}$ as

$$E_i = \frac{\hat{E}_i - D_{2,i}}{L} \tag{18}$$

From the prediction error E_i and the predictor $\widehat{w}_i$ the actual cover pixel value can be estimated as

$$w_i = E_i + \widehat{w}_i \tag{19}$$

Similarly, the expanded error in level-1 embedding can be estimated from the marked pixel value v_i' and the predictor $\hat{v}_i$, the prediction error can be estimated as,

$$\hat{e}_i = v_i' - \hat{v}_i \tag{20}$$

The hidden data in level-1 can be estimated as

$$D_{1,i} = \hat{e}_i - L\left\lfloor\frac{\hat{e}_i}{L}\right\rfloor \tag{21}$$

The prediction error can be estimated from the expanded error $\hat{e}_i$ and data $D_{1,i}$ as .

$$e_i = \frac{\hat{e}_i - D_{1,i}}{L} \quad (22)$$

From the prediction error e_i and the predictor $\hat{v}_i$ the actual cover pixel value can be estimated as

$$v_i = e_i + \hat{v}_i \quad (23)$$

Using the data $D_{1,i}$ and $D_{2,i}$ the actual data D can be reconstructed.

III. EXPERIMENTAL RESULTS

The performance evaluation of the three-level hierarchical data embedding algorithm was evaluated with the images of Bows-2 [27], and BossBase [28] datasets. A few of the standard test images depicted in Fig. 6 are also used to evaluate the algorithm's performance. The evaluation metrics such as structural similarity index ($SSIM$), embedding rate (ER), and peak signal-to-noise ratio ($PSNR$) are used to evaluate the performance which can be estimated using the following relations,

$$SSIM = C_0(Q, \hat{Q}) s_a(Q, \hat{Q}) L_u(Q, \hat{Q}) \quad (24)$$

Here $C_0(Q, \hat{Q})$, $s_a(Q, \hat{Q})$, and $L_u(Q, \hat{Q})$ represents the comparison between $Q, \hat{Q}$ in terms of contrast, saturation, and luminance respectively. Here Q represents the image before embedding and $\hat{Q}$ represent the image after embedding the data.

$$ER = \frac{Capacity}{U \times V} \quad (25)$$

$$PSNR = 10 \log_{10} \frac{255^2}{\tau} \text{dB} \quad (26)$$

$$\text{Where } \tau = \frac{1}{U \times V} \sum_{u=0}^{U-1} \sum_{v=0}^{V-1} (Q(u, v - \hat{Q}(u, v)))^2 \quad (27)$$

Here τ represents the mean square error.

(a) (b) (c) (d) (e)

(e) (f) (g) (h) (i)

Figure 6: Standard test images (a) Lena (b) Jetplane (c) Pepper (d) Lake (e) Cameraman (f) Blonde (g) Mandril (h) Living room (i) House (j) Bridge

Figure 7: Sample test images from BossBase and Bows-2 dataset (row-1) BossBase dataset (row-2) Bow-2 dataset

A few of the sample images from BossBase and Bows-2 datasets are illustrated in Fig.7. Table 1 shows the embedding rate comparison between the proposed method and the traditional data embedding schemes for the standard test images. The traditional schemes namely Puteax et al. [18], Ren et al. [29], Yin et al. [22], Wu et al. [30], Luo et al. [31], Zhang et al. [21], and Wang et al. [32] are used to compare the performance of the proposed scheme. The evaluation results provided in Table 1 were obtained with the group length $L = 2$ and the recursion length $R = 2$. For the standard test images, the proposed data-hiding approach provides a higher embedding rate than the traditional schemes.

Table 1: Comparison of embedding rate between the proposed and traditional schemes

Test image	Puteax et al. [18]	Ren et al. [29]	Yin et al. [22]	Wu et al. [30]	Luo et al. [31]	Zhang et al. [21]	Wang et al. [32]	Proposed
Lena	0.977	1.712	2.583	2.645	2.87	3.019	2.83	3.216
Jetplane	0.983	2.097	3.03	2.673	3.232	3.237	3.154	3.317
Pepper	0.871	1.824	2.348	1.963	1.581	2.087	2.347	2.532
Lake	0.954	2.054	2.705	2.546	1.51	1.923	1.893	2.203
Cameraman	0.981	1.577	2.349	2.479	2.49	2.651	2.629	2.804
Blonde	0.993	2.158	2.824	2.652	2.943	3.091	3.017	3.207
Mandril	0.838	0.456	1.006	0.969	1.321	1.4596	1.452	2.283
Living room	0.825	2.11	2.251	1.087	2.557	2.672	2.617	2.932
House	0.846	1.457	3.08	2.861	2.532	3.149	2.896	3.201
Bridge	0.957	2.024	1.054	2.406	2.54	2.851	2.768	3.001

Table 2. shows the comparison of the average embedding rate when evaluated using the Bows-2 and BossBase datasets. The proposed three-level hierarchical data embedding scheme provides an average embedding rate of $3.513bpp$ and $3.806bpp$ for Bows-2 and BossBase datasets respectively with the group length $L = 2$ and recursion length $R = 2$. The embedding rate of the proposed method is 0.057bpp and 0.124bpp greater than Bows-2 and Bossbase datasets respectively than the scheme proposed by Zhang et al. [21].

Table 2: Comparison of embedding rate on Bow-2 and BossBase dataset

Schemes	BOWS-2	BOSSbase
Puteax et al. [18]	0.968	0.966
Ren et al. [29]	2.519	2.561
Yin et al. [22]	3.393	3.498
Wu et al. [30]	3.246	3.361
Luo et al. [31]	3.421	3.517
Zhang et al. [21]	3.456	3.682
Wang et al. [32]	3.442	3.587
Proposed	3.513	3.806

The graphical comparison between the embedding rate of proposed and traditional schemes when evaluated using Bows-2 and Bossbase is illustrated in Fig. 8.

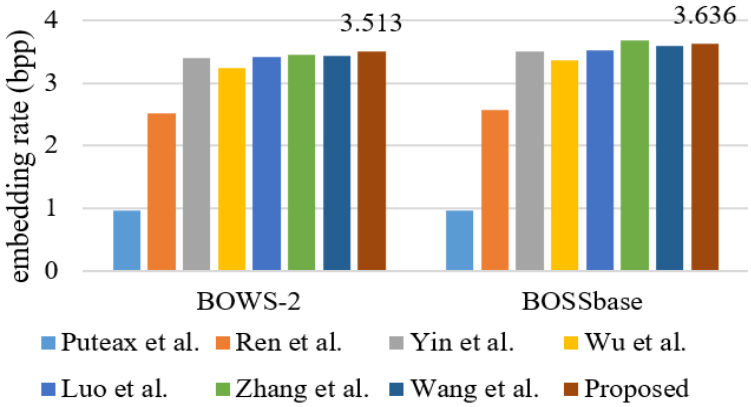
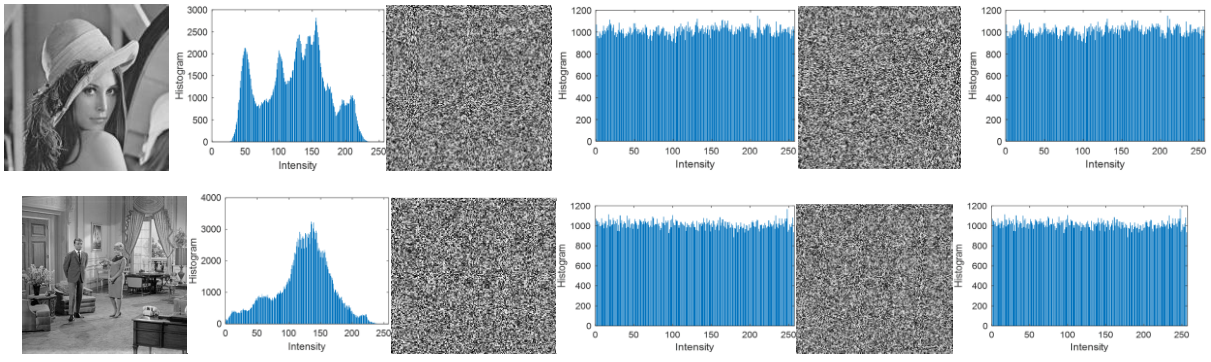


Figure 8: Graphical comparison of embedding rate between the proposed and traditional schemes

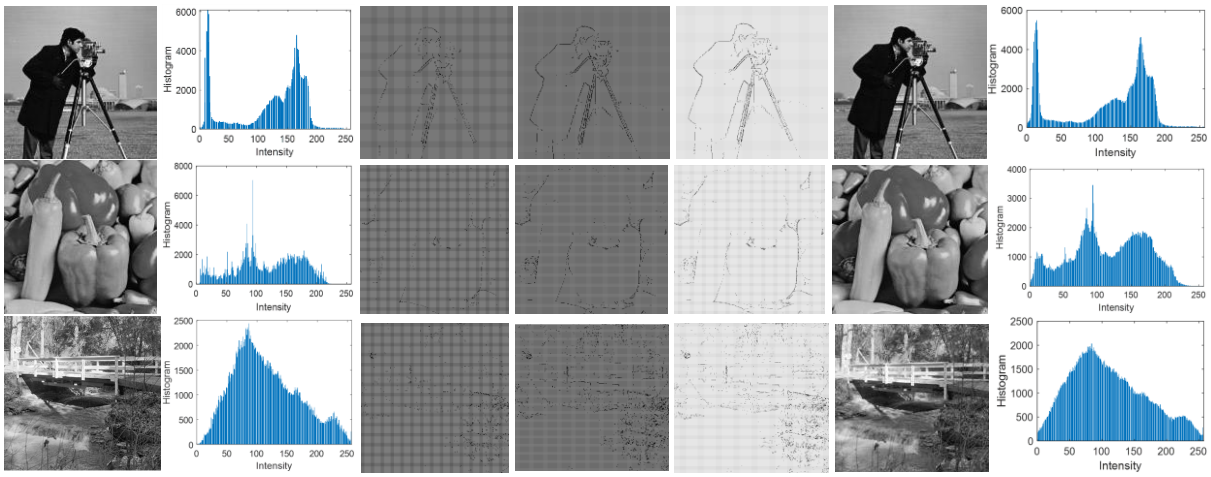
Fig.9 illustrates the experimental result obtained using the two-level encryption. In level-1 encryption, the plain original image information is completely made cipher. The histogram of the level 1 encrypted image is uniformly distributed. This shows that the proposed method completely encrypts the information present in the original image. The histogram of the encrypted image obtained in level-2 is the same as that of the level-1 encryption. However, the position of boundary pixels in the 3×3 encrypted image of level-1 encryption gets scrambled within the blocks.



(a) (b) (c) (d) (e) (f)

Fig 9: Results obtained in two-level encryption (a) Original image (b) Histogram of original image (c) Level-1 encrypted image (d) Histogram of Level 1 encrypted image (e) Level 2 encrypted image (f) Histogram of Level 2 encrypted image

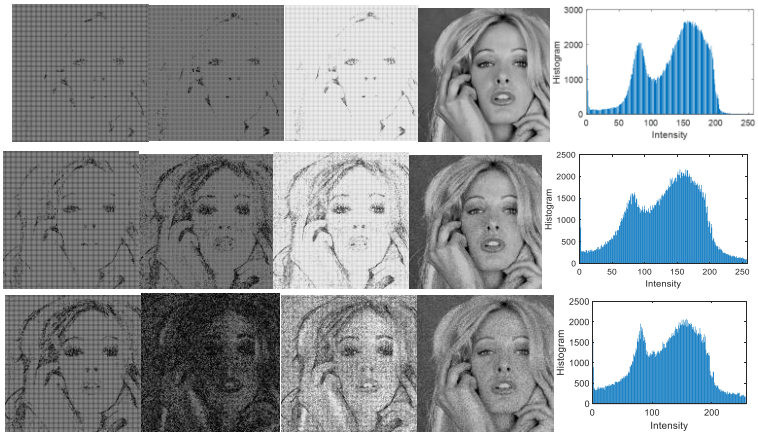
Fig.10 depicts the experimental results obtained in the embedding of data on the original image without encryption with the group length $L = 1$ and recursion length $R = 1$. Fig10(a) and Fig. 10(b) illustrate the original image with its histogram respectively. Fig 10(c), Fig 10(d), and Fig. 10(e) show the embedding pixel locations in Level-1, Level-2, and Level-3 embedding respectively. In these binary images, intensity 1 corresponds to embedded pixel locations while intensity 0 corresponds to the location where the data is not embedded. Fig 10(f) and Fig 10(g) show the marked image and its histogram respectively.



(a) (b) (c) (d) (e) (f) (g)

Figure 10: Results obtained in three-level data embedding (a) Original image (b) Histogram of the original image (c) Embedded pixel locations in level-1 embedding (d) Embedded pixel locations in level 2 embedding (e) Embedded pixel locations after three-level embedding (f) Marked image (g) Histogram of marked image

Fig.11 depicts the experimental results obtained in the embedding of data on the original 'Blonde' image without encryption with the group length $L = 1,2,3$ and recursion length $R = 1$. As the group length, L increases the embedding rate increases while the visual quality $PSNR$ reduces. As the value of group length L increases the number of non-embeddable pixels increases and this is due to the increase in overflow and underflow pixels.



(a) (b) (c) (d) (e)

Figure 11: Results obtained in three-level embedding (row1) $L = 1$ (row2) $L = 2$ (row3) $L = 3$ (a) Embedded pixel locations in level-1 embedding (b) Embedded pixel locations in level-2 embedding (c) Embedded pixel locations after three level embedding (d) Marked image (e) Histogram of marked image

Table 3 shows the comparison of PSNR and embedding rate for different values of group length L and recursion length R for different test images. With $R = 1$, as the value of L increases the embedding rate increases reducing the $PSNR$. However with $R = 2$, as the value of L increases the embedding rate increases till $L = 2$, and increasing the L value to 3 reduces the embedding rate. This is due to the increase in the number of overflow and underflow pixels. The maximum embedding rate is obtained for $L = 2$ and $R = 2$.

Table 3: Comparison of $PSNR$ and ER for different values of group length L and recursive length R

	R=1						R=2					
Test image	L=1		L=2		L=3		L=1		L=2		L=3	
	PSNR	ER	PSNR	ER	PSNR	ER	PSNR	ER	PSNR	ER	PSNR	ER
Lena	67.40	0.767	61.79	1.701	59.21	2.131	63.80	1.367	52.70	3.216	53.352	3.108
Jetplane	67.30	0.783	61.27	1.764	58.58	2.236	61.24	1.793	52.10	3.317	53.232	3.128
Pepper	67.57	0.711	62.78	1.536	59.39	2.096	65.52	1.132	56.97	2.532	57.907	2.348
Lake	67.84	0.694	63.73	1.409	60.52	1.913	64.30	1.284	58.78	2.203	59.772	2.038
Cameraman	67.55	0.741	62.48	1.587	59.44	2.094	63.89	1.351	55.18	2.804	56.142	2.643
Blonde	67.38	0.759	61.90	1.683	58.89	2.193	63.83	1.361	52.69	3.207	53.594	3.096
Mandril	67.76	0.706	63.25	1.413	60.44	1.926	64.05	1.314	58.30	2.283	59.022	2.163
Living room	67.51	0.749	62.42	1.596	59.44	2.093	63.80	1.367	54.22	2.932	56.322	2.613
House	67.67	0.761	61.90	1.683	58.95	2.198	63.76	1.373	52.79	3.201	54.218	2.961
Bridge	67.46	0.757	62.36	1.607	59.37	2.105	63.36	1.392	53.99	3.001	54.984	2.836

Table 4 shows the $SSIM$ comparison for different values of L and R. Similar to PSNR, higher SSIM values are obtained for $L = 1$ and $R = 1$. Lower values of SSIM are obtained for $L = 2$ and $R = 2$.

Table 4: SSIM comparison for different values of group length L and recursion length R

	R=1			R=2		
Test image	L=1	L=2	L=3	L=1	L=2	L=3
Lena	0.9800	0.9521	0.9393	0.9621	0.9069	0.9101
Jetplane	0.9795	0.9502	0.9362	0.9494	0.9039	0.9095
Pepper	0.9817	0.9570	0.9403	0.9691	0.9273	0.9328
Lake	0.9822	0.9608	0.9458	0.9646	0.9371	0.9421
Cameraman	0.9808	0.9555	0.9404	0.9626	0.9192	0.9240
Blonde	0.9802	0.9527	0.9374	0.9623	0.9072	0.9105
Mandril	0.9818	0.9607	0.9454	0.9637	0.9348	0.9383
Living room	0.9805	0.9553	0.9404	0.9621	0.9154	0.9249
House	0.9802	0.9527	0.9373	0.9619	0.9074	0.9145
Bridge	0.9803	0.9549	0.9401	0.9613	0.9133	0.9183

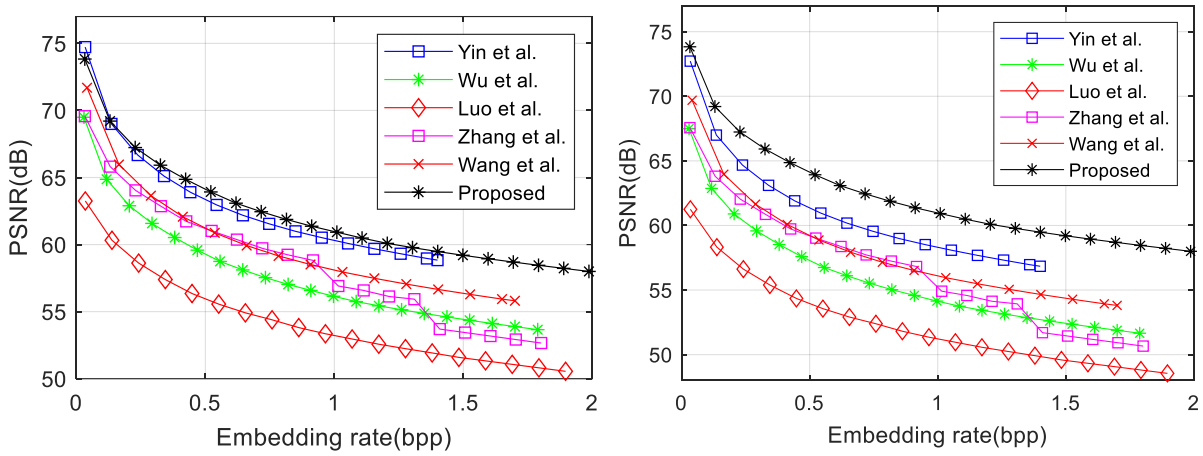


Figure 12: Comparison of *PSNR* for different embedding rates (a) BossBase dataset (b) Bows-2 dataset

Fig 12 shows the variation of $PSNR$ with a different embedding rate between $0.01bpp$ and $2bpp$. For the embedding rate of $0.01bpp$, the proposed method provides a $PSNR$ of $73dB$ in both the BossBase and Bows-2 datasets. As the embedding rate is increased from $0.01dB$, the $PSNR$ gradually reduces. With an embedding rate of $2bpp$, the proposed scheme provides a $PSNR$ of around $58dB$. In both the BossBase and Bows-2 datasets, the $PSNR$ of the proposed method is higher than the traditional schemes for different embedding rates between $0.01bpp$ and $2bpp$.

Let $T_{e,1}$, $T_{e,2}$, $T_{d,1}$, and $T_{d,2}$ represents the time of level-1 encryption, level-2 encryption, level-1 decryption, and level-3 decryption respectively. Similarly, $T_{E,1}$, $T_{E,2}$, and $T_{E,3}$ represent the time of level 1, level 2, and level 3 embedding respectively. The time complexity of level-1 embedding includes the estimation of a strong predictor followed by actual level-1 embedding. The time complexity of level 2 embedding includes the time complexity of weak predictor estimation followed by actual level 2 data embedding. The time complexity of level 3 embedding includes the time complexity of location map generation histogram mapping, Huffman compression, and actual level 3 data embedding represented as

$$T_E = T_{E,1} + T_{E,2} + T_{E,3} \tag{20}$$

Similarly, $T_{X,1}$, $T_{X,2}$, and $T_{X,3}$ represent the time of level-1, level-2, and level-3 data extraction process respectively. The computational complexity has been evaluated in a PC with Windows 10, a 16-bit operating system with an INTEL core i5 processor $(3.00GHz)$ using MATLAB 2018a.

Table 5: Computational complexity of the proposed 3-level data embedding algorithm

R	L	Time consumption (sec)							
		T_e	$T_{E,1}$	$T_{E,2}$	$T_{E,3}$	$T_{X,3}$	$T_{X,2}$	$T_{X,1}$	T_d
1	1	0.236	0.456	0.438	0.896	0.912	0.461	0.461	0.381
	2	0.236	0.487	0.456	0.931	0.936	0.489	0.478	0.381
	3	0.236	0.501	0.471	0.957	0.968	0.501	0.492	0.381
2	1	0.236	0.832	0.812	1.286	1.563	0.932	0.863	0.381
	2	0.236	0.867	0.836	1.354	1.768	0.958	0.881	0.381
	3	0.236	0.897	0.857	1.457	1.812	0.987	0.935	0.381

Table 5 shows the computational complexity comparison for the proposed method for different values of R and L. The proposed method has time consumption in encryption and decryption of $0.236s$ and $0.381s$ respectively. For any value of R, as the value of L increases the embedding time $T_{E,1}$, $T_{E,2}$, $T_{E,3}$ and data extraction time $T_{X,1}$, $T_{X,2}$, and $T_{X,3}$ increases. Fig. 13 shows the graphical comparison of time complexity for different values of $R = 2$ and $L = 2$.

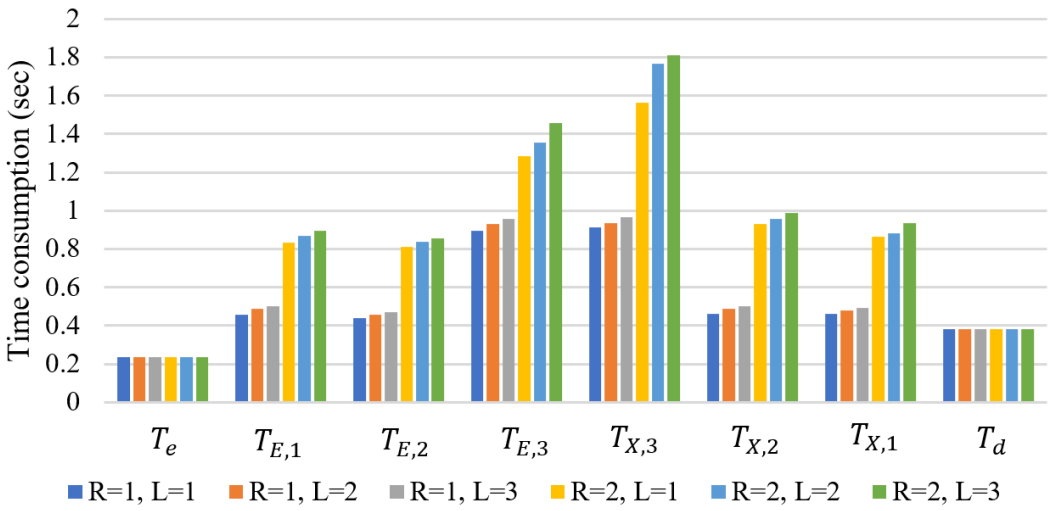


Figure 13: Comparison of computation time for different values of R and L

IV. CONCLUSION

The paper proposed a three-level hierarchical embedding along with histogram mapping in encrypted images to achieve reversible data hiding. The image owner initially encrypts the image using the two-level permutation-based encryption after subdividing the image into non-overlapping blocks. In the data embedding phase two predictors: weak and strong predictors are estimated which is used in level 1 and level 2 of hierarchical data embedding. The location map for level 1 and level 2 embedding is then constructed and mapped using a histogram mapping approach. The location map is then compressed using the Huffman coding and is embedded as level 3 of the hierarchical embedding process. Datasets namely Bossbase and Bows-2 are used for the evaluation of the proposed data embedding algorithm with the evaluation metrics such as embedding rate, PSNR, and SSIM. With a recursion length R=2 and group length $L = 2$, the proposed approach provides an embedding rate of 3.513 bpp and 3.806 bpp respectively.

REFERENCES

[1] Akbar, Hussain, Muhammad Zubair, and Muhammad Shairoze Malik. "The Security Issues and challenges in Cloud Computing." International Journal for Electronic Crime Investigation 7, no. 1 (2023): 13-32.

[2] Z. Tang, X. Zhang, X. Li, and S. Zhang, “Robust image hashing with ring partition and invariant vector distance,” IEEE Trans. Inf. Forensics Security, vol. 11, no. 1, pp. 200–214, Jan. 2016.

[3] C. Qin, P. Ji, X. Zhang, J. Dong, and J. Wang, “Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy,” Signal Process., vol. 138, pp. 280–293, Sep. 2017.

[4] Ge, Sulong, Jianwei Fei, Zhihua Xia, Yao Tong, Jian Weng, and Jianan Liu. "A screen-shooting resilient document image watermarking scheme using deep neural network." IET Image Processing 17, no. 2 (2023): 323-336.

[5] Bhardwaj, Rupali. "Hiding patient information in medical images: an enhanced dual image separable reversible data hiding algorithm for E-healthcare." Journal of Ambient Intelligence and Humanized Computing 14, no. 1 (2023): 321-337.

[6] B. Ou, X. Li, Y. Zhao, and R. Ni, “Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion,” Signal Process., Image Commun., vol. 29, no. 7, pp. 760–772, Aug. 2014.
[7] F. Peng, X. Li, and B. Yang, “Improved PVO-based reversible data hiding,” Digit. Signal Process., vol. 25, pp. 255–265, Feb. 2014
[8] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, “Reversible data hiding,” IEEE Trans. Circuits Syst. Video Technol., vol. 16, no. 3, pp. 354–362, Mar. 2006.
[9] C. Qin, C.-C. Chang, Y.-H. Huang, and L.-T. Liao, “An inpaintingassisted reversible steganographic scheme using a histogram shifting mechanism,” IEEE Trans. Circuits Syst. Video Technol., vol. 23, no. 7, pp. 1109–1118, Jul. 2013.
[10] J. Tian, “Reversible data embedding using a difference expansion,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 8, pp. 890–896, Aug. 2003.
[11] M. Alattar, “Reversible watermark using the difference expansion of a generalized integer transform,” IEEE Trans. Image Process., vol. 13, no. 8, pp. 1147–1156, Aug. 2004.
[12] J. Fridrich, M. Goljan, and R. Du, “Lossless data embedding—New paradigm in digital watermarking,” EURASIP J. Adv. Signal Process., vol. 2002, no. 2, 2002, Art. no. 986842.
[13] X. Zhang, “Commutative reversible data hiding and encryption,” Secur. Commun. Netw., vol. 6, no. 11, pp. 1396–1403, Nov. 2013.
[14] W. Zhang, K. Ma, and N. Yu, “Reversibility improved data hiding in encrypted images,” Signal Process., vol. 94, pp. 118–127, Jan. 2014.
[15] S. Yi and Y. Zhou, “Separable and reversible data hiding in encrypted images using parametric binary tree labeling,” IEEE Trans. Multimedia, vol. 21, no. 1, pp. 51–64, Jan. 2019
[16] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, “Secure reversible image data hiding over encrypted domain via key modulation,” IEEE Trans. Circuits Syst. Video Technol., vol. 26, no. 3, pp. 441–452, Mar. 2016.
[17] X. Zhang, “Separable reversible data hiding in encrypted image,” IEEE Trans. Inf. Forensics Security, vol. 7, no. 2, pp. 826–832, Apr. 2012.
[18] P. Puteaux and W. Puech, “An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images,” IEEE Trans. Inf. Forensics Security, vol. 13, no. 7, pp. 1670–1681, Jul. 2018
[19] H. Ge, Y. Chen, Z. Qian, and J. Wang, “A high capacity multi-level approach for reversible data hiding in encrypted images,” IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 8, pp. 2285–2295, Aug. 2019.
[20] S .Yi and Y. Zhou, “Separable and reversible data hiding in encrypted images using parametric binary tree labeling,” IEEE Trans. Multimedia, vol. 21, no. 1, pp. 51–64, Jan. 2019
[21] Yu, Chunqiang, et al. "Reversible data hiding with hierarchical embedding for encrypted images." IEEE Transactions on Circuits and Systems for Video Technology 32.2 (2021): 451-466.
[22] Yin, Zhaoxia, Youzhi Xiang, and Xinpeng Zhang. "Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding." IEEE Transactions on Multimedia 22.4 (2019): 874-884.
[23] Qiu, Yingqiang, et al. "High-capacity framework for reversible data hiding in encrypted image using pixel prediction and entropy encoding." IEEE Transactions on Circuits and Systems for Video Technology 32.9 (2022): 5874-5887.
[24] Mohammadi, Ammar, Mansor Nakhkash, and Mohammad Ali Akhaee. "A high-capacity reversible data hiding in encrypted images employing local difference predictor." IEEE Transactions on Circuits and Systems for Video Technology 30.8 (2020): 2366-2376.
[25] Shaji, C., and I. Shatheesh Sam. "A new data encoding based on maximum to minimum histogram in reversible data hiding." The Imaging Science Journal 67, no. 4 (2019): 202-214.
[26] Satir, E. and Isik, H., 2014. A Huffman compression based text steganography method. Multimedia tools and applications, 70, pp.2085-2110.
[27] P. Bas and T. Furon, “Image database of bows-2,” Accessed: Jun, vol. 20, 2017.
[28] P. Bas, T. Filler, and T. Pevny, “Break our steganographic system The ins and outs of organizing BOSS,” in Proc. 13th Int. Workshop Inf. Hiding, May 2011, pp. 59–70. Available: http://dde.binghamton.edu/download/
[29] H. Ren, W. Lu, B. Chen, Reversible data hiding in encrypted binary images by pixel prediction, Signal Process. 165 (2019) 268–277.
[30] Y. Wu, Y. Xiang, Y. Guo, J. Tang, and Z. Yin, “An improved reversible data hiding in encrypted images using parametric binary tree labeling,” IEEE Transactions on Multimedia, vol. 22, no. 8, pp. 1929–1938, 2020.
[31] Yin, Zhaoxia, et al. "Reversible data hiding in encrypted images based on pixel prediction and multi-MSB planes rearrangement." Signal Processing 187 (2021): 108146.
[32] Wang, Yaomin, and Wenguang He. "High capacity reversible data hiding in encrypted image based on adaptive msb prediction." IEEE Transactions on Multimedia 24 (2021): 1288-1298.