CTU Journal of Innovation and Sustainable Development
ISSN 2588-1418 | e-ISSN 2815-6412

DOI:10.22144/ctujoisd.2026.049

Reversible data hiding in encrypted images based on overlapping block partitioning and shared-MSB encoding

Tram Hoang Nam[1], and Nguyen Thai Son[2*]

[1]College of Engineering and Technology, Tra Vinh University, Viet Nam

[2]Resource Development Institute, Tra Vinh University, Viet Nam

*Corresponding author (thaison@tvu.edu.vn)

Article info.

Received 30 Jun 2026
Revised 8 Sep 2026
Accepted 10 Oct 2026

Keywords

Data hiding, encrypted image, reversible, shared-MSB encoding

ABSTRACT

Reversible data hiding in encrypted images (RDHEI) allows secret data to be embedded into an encrypted image while enabling lossless recovery of the original image. Its practical payload, however, remains limited by the embedding room available in the encrypted domain. This paper proposes an RDHEI method based on overlapping sub-block partitioning and shared-MSB encoding to improve the net embedding capacity. Each encrypted image block is organized into four overlapping sub-blocks. A boundary pixel shared by two adjacent sub-blocks serves as the reference pixel of the next sub-block, thereby increasing pixel reuse. The shared most significant bits within each sub-block are then exploited to create embedding room. Experiments on six standard images show an average net embedding capacity of 408,730 bits, representing improvements of 5.61% and 17.88% compared with two existing methods. Additional experiments on 100 randomly selected images from each of the BOWS-2 and BOSSBase datasets also show higher average net embedding capacities. Moreover, the proposed method provides error-free data extraction, exact image recovery, and separable operation according to the available keys.

1. INTRODUCTION

The growing use of cloud platforms for image storage and processing has increased the need to protect visual content before it is transmitted to an external service provider. Image encryption provides content confidentiality, but it also prevents the cloud server from performing conventional data-hiding operations. Reversible data hiding in encrypted images (RDHEI) addresses this issue by allowing additional information to be embedded into an encrypted image. An authorized receiver can subsequently extract the hidden data and recover the original image without loss (Ankur et al., 2025a; Ren et al., 2026). This property is useful in applications where image privacy and content integrity must be maintained, such as medical data management, image authentication, and secure cloud storage.

Depending on when the embedding room is generated, RDHEI methods are divided into reserving room before encryption (RRBE) (Nguyen et al., 2022; Ankur et al., 2025b) and vacating room after encryption (VRAE) (Chai et al., 2024; Ren et al., 2025). In RRBE, the image owner preprocesses the original image to reserve room and then encrypts the processed image. Ma et al. (2013) proposed the RRBE-based method by dividing the original image into two regions. The LSBs of selected pixels in the first region were reversibly embedded into the second region before encryption, leaving the

corresponding LSB positions available for secret data. Zhang et al. (2014) later improved the reversibility of this method by reorganizing the image content before encryption. Other RRBE methods employed binary-block embedding (Yi & Zhou, 2017), MSB prediction (Puteaux & Puech, 2018), MSB-plane rearrangement (Chen & Chang, 2019), multi-MSB prediction with Huffman coding (Yin et al., 2020), and adaptive prediction-error labeling (Wu et al., 2021). Since these operations are performed on the plaintext image, RRBE methods can directly exploit spatial redundancy. However, they require the image owner to complete room reservation before encryption, increasing the preprocessing workload on the owner side.

In VRAE, the original image is encrypted before the embedding room is generated. In the early scheme proposed by Zhang (2011), the encrypted image was divided into blocks, and one secret bit was embedded by modifying selected encrypted pixels in each block. During decryption, spatial smoothness was used to determine the original form of the modified block. Zhang (2012) subsequently separated data extraction from image decryption, allowing the hidden data to be obtained without access to the image-encryption key. However, these early methods provided limited embedding capacity and could produce recovery errors in blocks containing complex textures. To preserve intra-block pixel correlations, Huang et al. (2016) combined block permutation with block-wise XOR encryption so that the correlations among pixels within each block were preserved after encryption. Fu et al. (2019) employed an adaptive encoding strategy to select suitable encrypted regions according to their local characteristics. Wang et al. (2019) used block selection and block-level encryption to distinguish blocks suitable for data embedding, while Yu et al. (2022) developed a hierarchical embedding procedure to exploit redundant space at multiple levels. These developments improved the embedding performance of VRAE methods, but their capacity still depends on how much intra-block redundancy is preserved by encryption and how efficiently that redundancy can be represented.

Among the redundancy preserved by block-wise encryption, equality at corresponding MSB positions is particularly useful. When all pixels within a block are XOR-encrypted using the same value, their original values are concealed while their shared-MSB relationships remain unchanged. This property has motivated several block-based MSB encoding methods. Wang et al. (2021) encoded the shared MSBs of encrypted blocks using an adaptive block-based strategy. Wang and He (2022) further employed adaptive MSB prediction within small blocks to increase the available embedding room.

Recently, Wang et al. (2026) proposed a VRAE method based on MSB-sharing encoding and overlapping sawtooth partitioning. In their method, the encrypted image is divided into overlapping sawtooth blocks. Each block contains one reference pixel and two neighboring pixels, and adjacent blocks share one pixel. However, the sawtooth partition generates a large number of blocks, each of which must be classified as embeddable or non-embeddable. As a result, the large number of classification results increases the location-map length and reduces the room available for secret data. To address this limitation, the proposed method divides the encrypted image into non-overlapping 4×4 blocks, each of which is further partitioned into four overlapping sub-blocks. Adjacent sub-blocks share one boundary pixel, which serves as the reference pixel of the next sub-block. This partition retains pixel reuse while reducing the location-map length, resulting in a higher net capacity.

The main contributions of this work are summarized as follows:

– An overlapping partition is introduced to organize each 4×4 encrypted image block into four sub-blocks. Each pair of adjacent sub-blocks shares one boundary pixel, which serves as the reference pixel of the next sub-block. This structure increases pixel reuse without enlarging the image block.

– A shared-MSB encoding and sequential recovery procedure is developed for the overlapping structure, allowing shared pixels to be reconstructed and reused as reference pixels during recovery.

– The proposed method provides a more favorable balance between embedding room and location-map overhead. Experiments on standard images and randomly selected images from BOWS-2 and BOSSBase demonstrate higher net embedding capacities than the compared methods while retaining exact image recovery, error-free data extraction, and separable operation.

2. PROPOSED METHOD

This section presents the proposed VRAE scheme, which combines a 4×4 block structure with overlapping sub-block partitioning and shared-MSB encoding. After image encryption, the encrypted

blocks are organized into overlapping sub-blocks to vacate room for data embedding. The location map is compressed and embedded together with the secret data. At the receiver, the location map is recovered before data extraction and image recovery. The framework of the proposed method is shown in Figure 1.

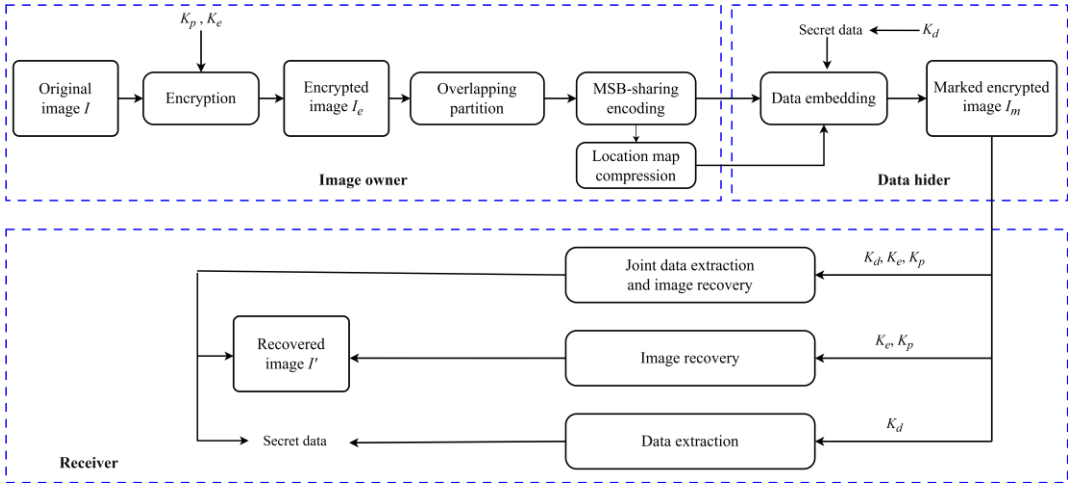


Figure 1. Framework of the proposed RDHEI method

2.1. Image encryption

To achieve content privacy while retaining intra-block pixel correlations, a hybrid encryption algorithm incorporating block permutation and bitwise exclusive-or (XOR) operations is employed. Given an original image I with dimensions $H \times W$, the image is first partitioned into non-overlapping blocks of size 4×4 pixels. The total number of blocks is denoted as $N = (H/4) \times (W/4)$.

Then, all 4×4 blocks are permuted using a pseudo-random sequence controlled by the block permutation key K_p. After block shuffling, bitwise XOR encryption is applied to every pixel within each permuted block. For the k-th block, an 8-bit pseudo-random value e_k is derived from the encryption key K_e. Each original pixel value $P_o(i,j)$ is converted into its 8-bit binary form and encrypted via $P_e(i,j) = P_o(i,j) \oplus e_k$, where $P_e(i,j)$ represents the encrypted pixel value. Because all pixels within the block undergo an XOR operation with an identical key e_k, equality at corresponding bit positions is preserved. Consequently, the shared-MSB length between neighboring pixels remains unchanged after encryption.

2.2. Overlapping partition

To increase pixel reuse, each 4×4 permuted block is further divided into 4 overlapping sub-blocks. The first three sub-blocks, B_1, B_2, and B_3, contain five pixels each, while the fourth B_4 contains four pixels. As depicted in Figure 2, each pair of overlapping sub-blocks shares a single boundary pixel, which serves as the reference pixel of the subsequent sub-block. Thus, the proposed partition produces 19 pixel usages from 16 distinct pixels, corresponding to a pixel utilization ratio of $(19/16) \times 100\% = 118.75\%$.

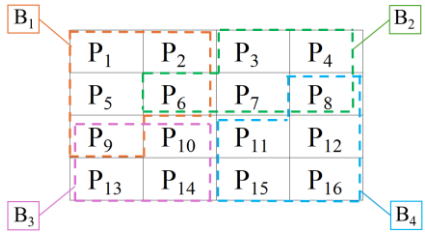


Figure 2. Proposed overlapping partition of a 4×4 image block

2.3. MSB-sharing encoding

2.3.1. Shared-MSB calculation and classification

Consider a sub-block $B_k = \{p_{k,1}, p_{k,2}, \ldots, p_{k,n_k}\}$, where $p_{k,1}$ represents the eight-bit reference pixel and $n_k \in \{4,5\}$ denotes the total number of pixels in the sub-block. Each pixel value is converted into an eight-bit binary sequence.

To evaluate the spatial redundancy within B_k, the number of shared MSBs between the reference pixel $p_{k,1}$ and each remaining pixel $p_{k,i}$ is determined by $s_{k,i}$. The shared-MSB length of the sub-block S_k is then calculated using Eq. (1):

$$S_k = \min_{2 \le i \le n_k} s_{k,i} \quad (1)$$

Thus, the first S_k bits are identical across all pixels in B_k. Since these shared bits can be fully recovered from $p_{k,1}$, they do not need to be stored repeatedly for the remaining regular pixels. A sub-block is considered embeddable when $S_k \geq 2$. Seven embeddable cases of S_k (ranging from 2 to 8) are represented by the three-bit codewords γ_k listed in Table 1. For a four-pixel sub-block, when $S_k < 2$, no vacant bit remains after storing the three-bit indicator. For a five-pixel sub-block, $S_k = 1$ theoretically provides one vacant bit; however, this case is excluded to maintain a common classification rule for both sub-block sizes. Therefore, a sub-block is classified as embeddable only when $S_k \geq 2$.

Table 1. Codeword assignment according to the shared-MSB length

Shared-MSB length S_k	Three-bit codeword γ_k	Available capacity C_k
2	001	$2(n_k - 1) - 3$
3	010	$3(n_k - 1) - 3$
4	011	$4(n_k - 1) - 3$
5	100	$5(n_k - 1) - 3$
6	101	$6(n_k - 1) - 3$
7	110	$7(n_k - 1) - 3$
8	111	$8(n_k - 1) - 3$

To distinguish between embeddable and non-embeddable sub-blocks, a location map LM is generated using Eq. (2):

$$LM_k = \begin{cases} 1, & S_k \geq 2, \\ 0, & S_k < 2. \end{cases} \quad (2)$$

2.3.2. Sub-block encoding and bit-stream construction

For an embeddable sub-block, the reference pixel $p_{k,1}$ is retained in full to serve as the decoding reference. For each remaining pixel, only its $8 - S_k$ non-shared bits are preserved. The remaining vacant space, providing an available capacity of $C_k = (n_k - 1)S_k - 3$ bits, is filled with encrypted secret bits. The complete encoded bit-stream Z_k for the sub-block is constructed via bit concatenation as defined in Eq. (3):

$$Z_k = p_{k,1} \,||\, \gamma_k \,||\, T_k \,||\, D_k \quad (3)$$

where: $p_{k,1}$ is the eight-bit binary representation of the reference pixel; γ_k is the three-bit indicator codeword corresponding to S_k; T_k contains the concatenated non-shared lower bits from the remaining pixels, with a total length of $|T_k| = (n_k - 1)(8 - S_k)$ bits; D_k represents the encrypted secret bits embedded into the vacant space, with a length of $|D_k| = C_k$ bits.

2.3.3. An illustrative example

Figure 3 illustrates the encoding of a five-pixel sub-block. The shared-MSB length is $S_k = 4$, corresponding to the codeword $\gamma_k = 011$. The four non-reference pixels require 16 non-shared bits, leaving 13 bits for data embedding. After the reference pixel, codeword, non-shared bits, and secret data are concatenated, the resulting 40-bit sequence is divided into five eight-bit segments, producing the marked values 115, 124, 125, 213, and 101.

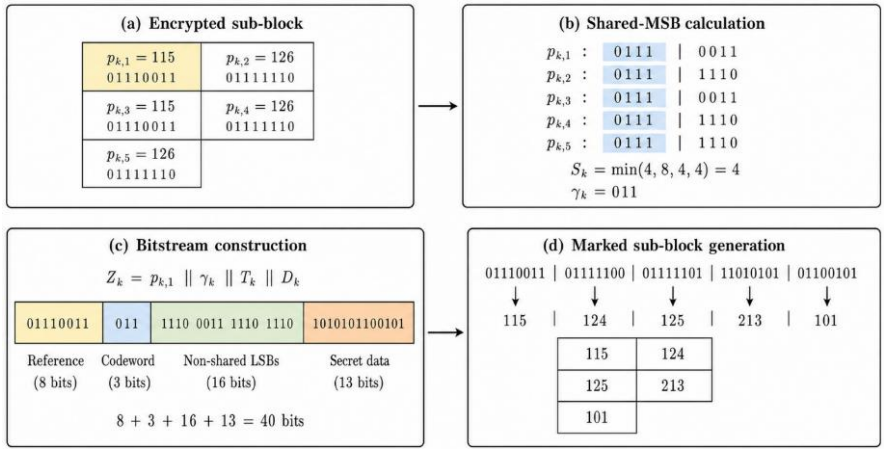


Figure 3. Illustrative example of MSB-sharing encoding for a five-pixel sub-block

(Notes: (a) encrypted sub-block, (b) shared-MSB calculation, (c) bitstream construction, and (d) marked sub-block generation)

2.4. Data embedding

To enable correct data extraction and image recovery, a predefined control region is reserved for storing the location map. The secret payload is embedded separately into the embeddable sub-blocks. The location map M is losslessly compressed using arithmetic coding to obtain M_{com}, whose length L_{map} is represented by an 18-bit header. Before data embedding, the 18-bit header and M_{com} are stored in the least significant bits (LSBs) of a predefined control region containing N_r pixels selected from several top image rows. These rows are excluded from overlapping sub-block encoding. To ensure complete reversibility, the original LSBs replaced in the control region, denoted as $D_{replaced}$, are collected and treated as part of the embedded payload.

The payload stream P_{stream} is formed by concatenating $D_{replaced}$ and the secret data D_{secret}, where only D_{secret} is encrypted using the data-hiding key K_d. For non-embeddable sub-blocks ($S_k < 2$), the pixel data remains unmodified. For embeddable sub-blocks ($S_k \geq 2$), a segment of C_k bits are taken from the payload stream to form D_k. Finally, the constructed sequence $Z_k = p_{k,1} \,||\, \gamma_k \,||\, T_k \,||\, D_k$ is segmented back into 8-bit marked pixels to update B_k, generating the final marked encrypted image I_m.

2.5. Data extraction and image recovery

Depending on the available keys, secret data extraction and original image recovery can be performed separately or jointly. A receiver possessing K_d can recover the secret data, whereas a receiver possessing K_e and K_p can reconstruct the original image. When all three keys are available, both operations can be performed to recover the secret data and the original image.

The decoding procedure begins by reading the LSBs of the N_r pixels in the predefined control region. The 18-bit header is first read to determine the length of the compressed location map. Then, the next L_{map} bits are retrieved to recover M_{com}. By applying arithmetic decoding to M_{com}, the location map M is reconstructed. Using the recovered location map, the receiver scans the sub-blocks in raster-scan order. If $LM_k = 1$, the codeword γ_k is used to determine S_k, and the last $C_k = (n_k - 1)S_k - 3$ bits are extracted from Z_k. The extracted segments are concatenated to reconstruct the payload stream P_{stream}, which contains $D_{replaced}$ followed by the encrypted secret data. This step does not require any cryptographic key.

2.5.1. Secret data extraction

A receiver possessing the data-hiding key K_d removes the first N_r bits corresponding to $D_{replaced}$ from the extracted payload stream. The remaining encrypted secret data are decrypted by performing a bitwise XOR operation with the pseudorandom bitstream generated using K_d, yielding the original secret data.

2.5.2. Original image reconstruction

A receiver possessing the image-encryption keys K_e and K_p reconstructs the original image through the following three stages:

(i) Sub-block restoration: Within each 4×4 image block, the sub-blocks are recovered sequentially according to the predetermined order shown in Fig. 2. For each embeddable sub-block, the reference pixel $p_{k,1}$, indicator codeword γ_k, and non-shared LSB sequence T_k are separated from Z_k. The shared-MSB length S_k is determined from γ_k, after which the non-reference pixels are reconstructed using $p_{k,1}$ and T_k. The recovered pixels are immediately used to update the partially recovered encrypted image, and the recovered shared pixels serve as references for recovering the subsequent sub-blocks. No sub-block restoration is required when $LM_k = 0$.

(ii) Control-region restoration: The LSBs of the N_r control-region pixels are restored using $D_{replaced}$, thereby recovering their original encrypted values.

(iii) Image decryption: After all sub-blocks and control-region pixels have been restored, block-wise XOR decryption is performed using K_e, followed by inverse block permutation using K_p, to obtain the reconstructed image I'.

3. EXPERIMENTAL RESULTS

We conducted the experiments in MATLAB R2023b on a laptop with an Intel Core i5-8250U CPU at 1.60 GHz and 8 GB of RAM under a 64-bit Windows operating system. Six standard 512×512 grayscale images were used: Lena, Baboon, House, Boat, Sailboat, and Barbara. These images represent different image characteristics, ranging from smooth regions to highly textured content. Each image was partitioned into 4×4 blocks, and each block was further organized into four overlapping sub-blocks containing 5 or 4 pixels.

Random binary sequences were used as secret payloads.

The embedding capacity, denoted by EC, represents the total number of bits available in all embeddable sub-blocks. Since part of this capacity is required to store the location map, the capacity available for secret data is calculated using Eq. (4):

$$EC_{net} = EC - (18 + L_{map}) \quad (4)$$

3.1. Net embedding capacity

Table 2 compares the compressed location-map length and net EC of the proposed method with those of Wang et al.'s (2026) method and Wang and He's (2022) method. The six test images were selected to cover different levels of spatial correlation. House contains relatively large smooth regions, whereas Baboon and Barbara contain considerably more texture. Lena, Boat, and Sailboat represent images with intermediate content characteristics.

Table 2. Comparison of net EC and location-map overhead

Images	Proposed method		Wang et al. (2026)		Wang and He (2022)	
	LM (bits)	Net EC (bits)	LM (bits)	Net EC (bits)	LM (bits)	Net EC (bits)
Lena	48,658	514,890	75,095	508,277	28,867	434,492
Baboon	65,143	159,836	126,479	126,121	60,033	135,849
House	55,722	514,982	91,085	500,479	35,142	440,715
Boat	46,224	514,613	76,381	492,879	29,268	427,497
Sailboat	58,248	385,680	96,641	361,668	28,219	336,287
Barbara	60,952	362,380	102,589	332,678	43,777	305,597
Average	**55,825**	**408,730**	**94,712**	**387,017**	**37,551**	**346,740**

The net EC represents the actual room available for secret data after deducting the location-map overhead. As shown in Table 2, the proposed method produces a shorter compressed location map than Wang et al.'s (2026) method for all six images, thereby leaving more room for secret data. The largest improvements over Wang et al.'s (2026) method are obtained for Baboon and Barbara, with gains of 33,715 and 29,702 bits, respectively. The average net EC increases from 387,017 to 408,730 bits, corresponding to an improvement of approximately 5.61%. Wang and He's (2022) method produces a shorter location map than the proposed method. However, it also reserves less embedding room initially, resulting in lower net EC for all six images. Compared with Wang and He's (2022) method, the proposed method provides net EC gains ranging from 23,987 bits for Baboon to 87,116 bits for Boat. On average, the net EC increases from 346,740 to 408,730 bits, corresponding to an improvement of approximately 17.88%.

To evaluate the proposed method on a broader range of image content, 100 images were randomly selected from each of the BOWS-2 and BOSSBase datasets. The same images were used to evaluate all three methods. For each image, the net EC was obtained after deducting the location-map overhead, and the average value over the 100 images was then calculated. Table 3 reports the mean and standard deviation of the net EC over the 100 images in each dataset.

Table 3. Mean and standard deviation of net EC on BOWS-2 and BOSSBase

Dataset	Proposed method	Wang et al. (2026)	Wang and He (2022)
BOWS-2	647,240 ± 197,781	638,857 ± 205,622	548,784 ± 156,476
BOSSBase	723,863 ± 256,814	711,087 ± 265,082	606,889 ± 201,287

As shown in Table 3, the proposed method achieves the highest average net EC on both datasets. On BOWS-2, the proposed method achieves an average net EC of 647,240 bits, exceeding Wang et al.'s (2026) method by 8,383 bits. On BOSSBase, the proposed method achieves an average net EC of 723,863 bits, which is 12,776 bits higher than that of Wang et al.'s (2026) method. Compared with Wang and He's (2022) method, the proposed method provides improvements of approximately 17.94% and 19.27% on BOWS-2 and BOSSBase, respectively. Although the improvements over Wang et al.'s (2026) method are moderate, the proposed method consistently achieves a higher average net EC on both datasets. The average net EC obtained on the two datasets is presented in Figure 4.

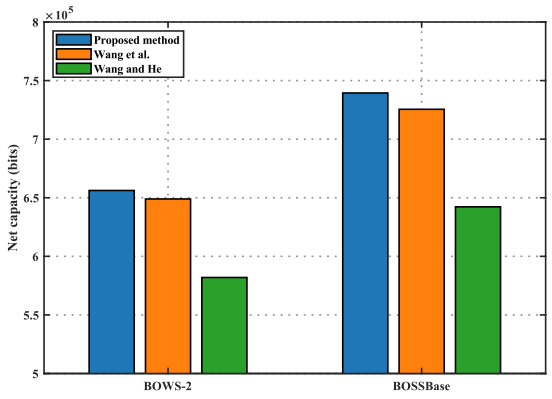


Figure 4. Average net EC obtained on 100 randomly selected images from each of the BOWS-2 and BOSSBase datasets

3.2. Statistical security analysis

We evaluated the statistical characteristics of the encrypted and marked encrypted images using the peak signal-to-noise ratio (PSNR), information entropy, and image correlation coefficient. The PSNR between the original image I and the marked encrypted image I_m is calculated using Eq. (5):

$$PSNR(I, I_m) = 10 \log_{10} \frac{255^2}{\text{MSE}(I, I_m)}, \quad (5)$$

where $MSE(I, I_m) = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} [I(i,j) - I_m(i,j)]^2$.

Since I_m remains in the encrypted domain, a low PSNR indicates a large visual difference from the original image rather than image degradation. For an 8-bit grayscale image I, its information entropy is calculated using Eq. (6):

$$H(I) = -\sum_{v=0}^{255} p_I(v) \log_2 p_I(v). \quad (6)$$

where $p_I(v)$ denotes the probability of gray level v. The maximum entropy of an 8-bit image is 8, corresponding to an ideally uniform gray-level distribution.

To measure the linear similarity between two images X and Y, the image correlation coefficient is computed using Eq. (7):

$$\rho(X, Y) = \frac{\sum_{i=1}^{H} \sum_{j=1}^{W} (X(i,j) - \mu_X)(Y(i,j) - \mu_Y)}{\sqrt{\sum_{i=1}^{H} \sum_{j=1}^{W} (X(i,j) - \mu_X)^2} \sqrt{\sum_{i=1}^{H} \sum_{j=1}^{W} (Y(i,j) - \mu_Y)^2}} \quad (7)$$

where μ_X and μ_Y are the mean pixel values of X and Y, respectively. A correlation coefficient close to zero indicates little linear similarity between the two images.

Table 4. Statistical characteristics of the encrypted and marked encrypted images

Images	PSNR(I, I_m) *(dB)*	$H(I_e)$	$H(I_m)$	$\rho(I, I_e)$	$\rho(I, I_m)$
Lena	9.6228	7.9987	7.9743	0.0053	0.0002
Baboon	9.6171	7.9992	7.9835	-0.0014	-0.0044
Barbara	9.1131	7.9983	7.9815	-0.0014	0.0036
Boat	9.2217	7.9985	7.9764	0.0007	0.0038
House	8.7978	7.9981	7.9879	-0.0077	-0.0025
Sailboat	8.5879	7.9988	7.9757	-0.0108	-0.0003

As shown in Table 4, the PSNR values between the original and marked encrypted images range from 8.5879 to 9.6228 dB. These low values indicate that the marked encrypted images differ substantially from their original counterparts. The entropy values of the encrypted images range from 7.9981 to 7.9992, approaching the maximum value of 8 for an 8-bit grayscale image. This suggests that the gray-level distributions of the encrypted images are approximately uniform. The entropy values of the marked encrypted images range from 7.9743 to 7.9879. Although these values are slightly lower than those of the encrypted images because of data embedding, they remain close to the maximum value of 8. In addition, the correlation coefficients between the original and encrypted images range from -0.0108 to 0.0053. After data embedding, the correlation coefficients between the original and marked encrypted images remain between -0.0044 and 0.0038. Therefore, data embedding does not introduce substantial linear similarity between the marked encrypted image and the original image. These results show that the encrypted and marked encrypted images reveal little statistical similarity to the original images.

Figure 5 presents the visual appearances and gray-level histograms of the original, encrypted, and marked encrypted Lena images. The original image contains clearly recognizable visual content and exhibits a non-uniform histogram concentrated in several gray-level ranges. After encryption, the image becomes noise-like, and its gray values are distributed more evenly. This indicates that block permutation and block-wise XOR encryption

effectively obscure the visual content of the original image. Data embedding changes the gray-level distribution of the encrypted image because part of its bitstream is replaced by auxiliary information and secret data. Consequently, the histogram of the marked encrypted image is not as uniform as that of the encrypted image. Nevertheless, the marked encrypted image remains noise-like, and no recognizable information about the original image is visually revealed.

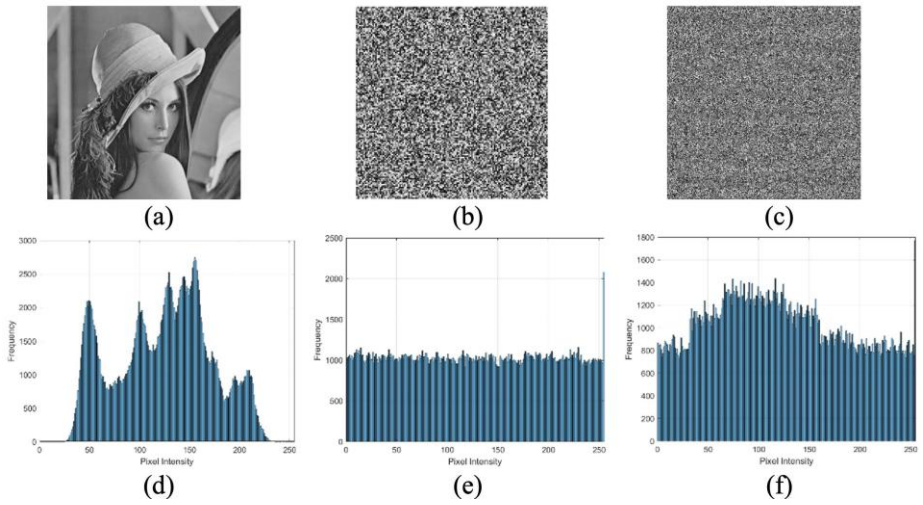


Figure 5. Visual appearances and gray-level histograms of Lena image

(Notes: (a) original, (b) encrypted, (c) marked encrypted, and (d)-(f) their corresponding histograms)

3.3. Reversibility and separability

The reversibility of the proposed method was evaluated by directly comparing the extracted secret data with the embedded data and the recovered image with the original image. The bit error rate (BER) is calculated using Eq. (8):

$$BER = \frac{N_e}{L_D} \times 100\% \quad (8)$$

where N_e is the number of mismatched bits between the original and recovered secret bitstreams, and L_D is the secret-payload length. For all six test images, the extracted secret data were identical to the corresponding embedded data, resulting in $BER = 0\%$. A pixel-by-pixel comparison also confirmed that each recovered image was identical to its original image. Therefore, $MSE(I, I') = 0$ and the corresponding *PSNR* was $+\infty$.

The separability of the proposed method was evaluated under three key settings. With only the data-hiding key K_d, the receiver can recover the secret data while the image remains encrypted. With the image-encryption keys K_e and K_p, the original image can be reconstructed without recovering the secret data. When all three keys are available, secret data extraction and original image recovery can be performed jointly. The results are summarized in Table 5.

Table 5. Reversibility and separability under different key settings.

Available keys	Secret data recovery	Image recovery	$\boldsymbol{BER}$ (%)	$\boldsymbol{PSNR(I, I')}$(dB)
K_d	Available	Unavailable	0	N/A
K_e, K_p	Unavailable	Available	N/A	$+\infty$
K_d, K_e, K_p	Available	Available	0	$+\infty$

Here, N/A indicates that the corresponding evaluation is not applicable under the given key setting. The results show that the proposed method supports separate and joint operations according to the available keys while maintaining error-free data extraction and exact image recovery.

3.4. Computational complexity and runtime

For an image of size $H \times W$, the proposed method divides the image into $HW/16$ non-overlapping blocks, each containing four sub-blocks of fixed size. Image encryption, overlapping partitioning, shared-MSB encoding, data embedding, and image

recovery process each pixel or sub-block a constant number of times. Arithmetic coding processes a location map whose length is proportional to the number of sub-blocks. Therefore, the overall computational complexity of the proposed method is $O(HW)$.

Table 6. Average runtime of the proposed method over 200 images from BOSSBase and BOWS-2

Stage	Average runtime (s)
Image encryption	0.0754
Room reservation and data embedding	2.3213
Data extraction and image recovery	2.0506
Total	**4.4473**

Table 6 reports the average runtime of the main processing stages over 200 images, comprising 100 randomly selected images from each of the BOSSBase and BOWS-2 datasets. The proposed method requires an average total runtime of 4.4473 s per image. Room reservation and data embedding take the most time on average (2.3213 s) because this stage includes overlapping partitioning, shared-MSB analysis, location-map compression, and data embedding. Data extraction and image recovery require 2.0506 s, while image encryption requires only 0.0754 s. The reported results provide an implementation-level reference for the computational cost of the proposed method.

4. CONCLUSION

This paper presented a reversible data hiding method for encrypted images that combines overlapping sub-block partitioning with shared-MSB encoding. Experimental results demonstrate higher net embedding capacities than the compared methods on both standard test images and randomly selected BOWS-2 and BOSSBase images. The encrypted and marked encrypted images have entropy values close to 8 and low correlation with the original images, indicating limited statistical similarity. Moreover, exact image recovery and error-free data extraction are achieved for all tested images.

ACKNOWLEDGMENT

This research was fully funded by Tra Vinh University (TVU) under grant contract number 37/2026/HĐ. HĐKH&ĐT.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

REFERENCES

Ankur, Kumar, R., & Sharma, A. K. (2025a). Reversible data hiding in encrypted image using bit-plane based label-map encoding with optimal block size. *Journal of Information Security and Applications*, *90*, 104005. https://doi.org/10.1016/j.jisa.2025.104005

Ankur, Kumar, R., Sharma, A. K., & Ranjan, P. (2025b). High-capacity reversible data hiding in encrypted images based on difference image transfiguration. *Signal, Image and Video Processing*, *19*(5), 365. https://doi.org/10.1007/s11760-025-03911-4

Chai, X., Cao, G., Fu, Z., Gan, Z., Wang, B., & Zhang, Y. (2024). High-capacity reversible data hiding in encrypted medical images using adaptive pixel-modulation and HBP-RMC. *Biomedical Signal Processing and Control*, *95*, 106424. https://doi.org/10.1016/j.bspc.2024.106424

Chen, K., & Chang, C.-C. (2019). High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. *Journal of Visual Communication and Image Representation*, *58*, 334–344. https://doi.org/10.1016/j.jvcir.2018.12.023

Fu, Y., Kong, P., Yao, H., Tang, Z., & Qin, C. (2019). Effective reversible data hiding in encrypted image with adaptive encoding strategy. *Information Sciences*, *494*, 21–36. https://doi.org/10.1016/j.ins.2019.04.043

Huang, F., Huang, J., & Shi, Y.-Q. (2016). New framework for reversible data hiding in encrypted domain. *IEEE Transactions on Information Forensics and Security*, *11*(12), 2777–2789. https://doi.org/10.1109/TIFS.2016.2598528

Ma, K., Zhang, W., Zhao, X., Yu, N., & Li, F. (2013). Reversible data hiding in encrypted images by reserving room before encryption. *IEEE Transactions on Information Forensics and Security*, *8*(3), 553–562. https://doi.org/10.1109/TIFS.2013.2248725

Nguyen, T.-S., Chang, C.-C., & Lin, C.-C. (2022). High capacity reversible data hiding scheme based on AMBTC for encrypted images. *Journal of Internet Technology*, *23*(2), 255–266. https://doi.org/10.53106/160792642022032302006

Puteaux, P., & Puech, W. (2018). An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images. *IEEE Transactions on Information Forensics and Security*, *13*(7), 1670–1681. https://doi.org/10.1109/TIFS.2018.2799381

Ren, H., Chen, T., Li, M., Yue, Z., Han, D., & Bai, G. (2025). High-capacity reversible data hiding in

encrypted images based on multi-predictions and efficient parametric binary tree labeling. *Digital Signal Processing*, *161*, 105096. https://doi.org/10.1016/j.dsp.2025.105096

Ren, H., Yue, Z., & Li, M. (2026). Reversible data hiding in encrypted images using adaptive block-level pixel difference encoding. *Journal of Information Security and Applications*, *96*, 104316. https://doi.org/10.1016/j.jisa.2025.104316

Wang, X., Chang, C.-C., & Lin, C.-C. (2021). Reversible data hiding in encrypted images with block-based adaptive MSB encoding. *Information Sciences*, *567*, 375–394. https://doi.org/10.1016/j.ins.2021.02.079

Wang, X., Li, L., Wang, M., & Chang, C.-C. (2026). High-capacity reversible data hiding in encrypted images based on MSB sharing and overlapping sawtooth partition. *Journal of King Saud University Computer and Information Sciences*, *38*(2), 24. https://doi.org/10.1007/s44443-025-00238-5

Wang, Y., Cai, Z., & He, W. (2019). A new high capacity separable reversible data hiding in encrypted images based on block selection and block-level encryption. *IEEE Access*, *7*, 175671–175680. https://doi.org/10.1109/ACCESS.2019.2957143

Wang, Y., & He, W. (2022). High capacity reversible data hiding in encrypted image based on adaptive MSB prediction. *IEEE Transactions on Multimedia*, *24*, 1288–1298. https://doi.org/10.1109/TMM.2021.3062699

Wu, X., Qiao, T., Xu, M., & Zheng, N. (2021). Secure reversible data hiding in encrypted images based on adaptive prediction-error labeling. *Signal Processing*, *188*, 108200. https://doi.org/10.1016/j.sigpro.2021.108200

Yi, S., & Zhou, Y. (2017). Binary-block embedding for reversible data hiding in encrypted images. *Signal Processing*, *133*, 40–51. https://doi.org/10.1016/j.sigpro.2016.10.017

Yin, Z., Xiang, Y., & Zhang, X. (2020). reversible data hiding in encrypted images based on multi-MSB prediction and huffman coding. *IEEE Transactions on Multimedia*, *22*(4), 874–884. https://doi.org/10.1109/TMM.2019.2936314

Yu, C., Zhang, X., Zhang, X., Li, G., & Tang, Z. (2022). Reversible data hiding with hierarchical embedding for encrypted images. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(2), 451–466. https://doi.org/10.1109/TCSVT.2021.3062947

Zhang, W., Ma, K., & Yu, N. (2014). Reversibility improved data hiding in encrypted images. *Signal Processing*, *94*, 118–127. https://doi.org/10.1016/j.sigpro.2013.06.023

Zhang, X. (2011). Reversible data hiding in encrypted image. *IEEE Signal Processing Letters*, *18*(4), 255–258. https://doi.org/10.1109/LSP.2011.2114651

Zhang, X. (2012). Separable reversible data hiding in encrypted image. *IEEE Transactions on Information Forensics and Security*, *7*(2), 826–832. https://doi.org/10.1109/TIFS.2011.2176120