Image Steganography Using Complex Blocks and an Enhanced Pixel Indicator Technique With LSBMR

Original Scientific Paper

Sami Ghoul*
Universiti Kebangsaan Malaysia (UKM),
Faculty of Information Science and Technology,
Center for Cyber Security
Bangi, Selangor, Malaysia
P114465@siswa.ukm.edu.my
Department of Computer Engineering
Faculty of Engineering,
University of Zawia, Libya

Rossilawati Sulaiman
Universiti Kebangsaan Malaysia (UKM),
Faculty of Information Science and Technology,
Center for Cyber Security
Bangi, Selangor, Malaysia
rossilawati@ukm.edu.my

Zarina Shukur
Universiti Kebangsaan Malaysia (UKM),
Faculty of Information Science and Technology,
Center for Cyber Security
Bangi, Selangor, Malaysia
zarinashukur@ukm.edu.my

Faizan Qamar
Universiti Kebangsaan Malaysia (UKM),
Faculty of Information Science and Technology,
Center for Cyber Security
Bangi, Selangor, Malaysia
faizanqamar@ukm.edu.my

*Corresponding author

Abstract *– Region-based steganography conceals data in visually complex areas of an image to minimize perceptual distortion. Traditional approaches typically rely on statistical metrics such as standard deviation or entropy to identify these regions; however, these measures often fail to capture spatial relationships among pixels and can be significantly affected by outliers, leading to unreliable complexity estimations. This paper introduces a novel block complexity measure based on local pixel similarity, which effectively captures spatial structure and enables accurate selection of embedding regions. The method segments the image into N×N blocks, quantizes pixel intensities into L levels, and calculates a similarity-based complexity value. A threshold is then applied to select the most complex blocks for data embedding. Additionally, an enhanced Pixel Indicator Technique (EPIT) is proposed to integrate repetition-based encoding with least significant bit matching revisited (LSBMR), reducing visual distortion and increasing robustness. By embedding data only in highly complex blocks, the proposed approach achieves strong imperceptibility and robustness against statistical steganalysis. Experimental evaluations on grayscale and RGB images confirm its effectiveness, achieving peak signal-to-noise ratio (PSNR) values up to 56.57 and weighted PSNR up to 86.60 for grayscale images, and exceeding 52 dB PSNR and 0.98 SSIM at 8 KB payloads for RGB images. These results demonstrate strong visual fidelity and resilience against RS and Chi-square attacks, as well as moderate resistance to convolutional neural network-based steganalysis (XuNet) and ensemble classifier-based steganalysis.*

Keywords: *Image steganography, Region-based steganography, block complexity, Pixel Indicator Technique*

Received: August 23, 2025; Received in revised form: January 19, 2026; Accepted: January 21, 2026

1. INTRODUCTION

With the widespread use of computers and smart devices in the Internet age, secure transmission of sensitive information has become a growing concern. While cryptography protects data through encryption, it often signals the presence of secret content. Steganography, in contrast, conceals the existence of secret data by embedding it within cover media such as images [1], videos [2], audio [3], or text [4], making detection more challenging [5]. A typical steganographic system is illustrated in Fig. 1 [6] between sender and receiver. The cover image is used to embed the secret data, producing the stego image, which is then transmitted over an insecure medium, such as the Internet. At the receiver's end, the extraction algorithm operates in reverse order to recover the embedded data. The secret data may be compressed and encrypted to reduce size and enhance security.

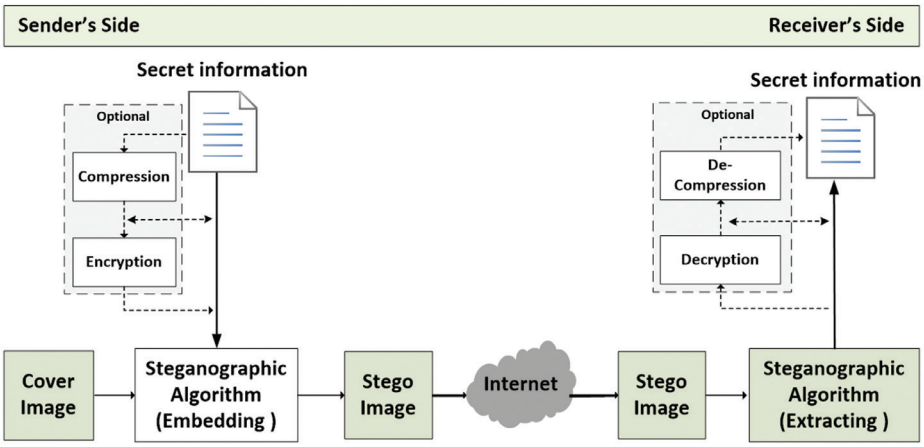


Fig. 1. General image steganographic system

Digital images are particularly popular in steganographic applications due to their widespread use and inherent redundancy. Effective steganographic algorithms aim to balance three key requirements: embedding capacity (payload), imperceptibility (visual quality), and security (undetectability). The payload is measured in bits or bits per pixel (bpp), while imperceptibility refers to how closely the stego image resembles the original. Security ensures resistance against visual and statistical steganalysis attacks [5, 6]. Among image steganography techniques, region-based steganography embeds data only in edge or highly textured (i.e., complex) regions, where the human visual system is less sensitive to changes and statistical modeling is more difficult due to abrupt intensity variations [7, 8]. Edge-based schemes identify such regions using edge detectors (e.g., Sobel, Canny) and embed data in the resulting edge pixels, thereby improving imperceptibility and resistance to steganalysis. However, their embedding capacity is limited, especially in images with few edges [6, 7]. To address this limitation, some methods locate highly textured blocks using complexity metrics such as standard deviation or entropy [9]. These methods, however, overlook the spatial distribution of pixel values and are often biased by outliers, leading to an inaccurate selection of embedding region. Additionally, Pixel Indicator Techniques (PITs) enhance data hiding in RGB images by designating one channel as an indicator and others as carriers. Nevertheless, since most existing PITs use sequential embedding with fixed patterns, they limit flexibility and increase pixel modifications, which can reduce imperceptibility and make them more vulnerable to steganalysis.

This paper addresses such limitations through two main innovations. First, we introduce a new pixel similarity-based block complexity measure that evaluates local pixel relationships, providing a more accurate and outlier-resistant method for identifying embedding regions. Second, we propose an enhanced PIT (EPIT), which integrates value-repetition encoding and the least significant bit matching revisited (LSBMR) algorithm. This EPIT-LSBMR integration not only improves embedding efficiency but also reduces visual distortion by minimizing pixel modifications. By restricting data embedding to blocks identified as highly complex using our proposed metric, the method achieves a superior balance between imperceptibility, payload, and security. This work offers three main contributions to the field:

- a novel pixel similarity-based block complexity metric with an integrated thresholding scheme for accurately identifying highly complex embedding regions, thereby improving imperceptibility and security;
- the design of EPIT-LSBMR for RGB images, which combines PITs with LSBMR to minimize pixel modifications and enhance embedding efficiency; and
- the introduction of EPIT-LSBMR-Gray, a grayscale-compatible variant that extends the applicability of the proposed method beyond RGB images.

The paper is organized as follows: Section 2 elaborates on related work, followed by Section 3, which details the core principles underlying the proposed method. Section 4 presents the overall framework and the implementation of the proposed method on both the sender's and receiver's sides. Section 5 demonstrates the experimental results and compares the efficiency of the proposed methods to state-of-the-art methods using selected evaluation metrics. The concluding remarks are then summarized in Section 6.

2. RELATED WORK

The literature can be broadly categorized into two major groups: region-based embedding techniques and PITs.

2.1. REGION-BASED EMBEDDING TECHNIQUES

Region-based methods hide secret data in visually complex, textured, or edge-rich areas where modifications are less perceptible to the human eye. The Block-wise Edge Adaptive Steganography Scheme (BEASS), explained in [9], employs a fuzzy edge detector to iden-

tify edges and utilizes adjacent pixels to conceal data. To assess local complexity, the method divides the cover image into blocks of size 64x64 and calculates their standard deviation. Following a sorting process based on the calculated standard deviation, these blocks are further divided into 3x3 blocks. Three message bits are hidden in edge pixels within each block based on minimum mean square error (MSE) considerations. This technique minimizes image distortion and enhances embedding capacity. Nevertheless, relying solely on standard deviation to assess block complexity may be inaccurate, as it disregards the spatial arrangement of pixel blocks. Dividing the image into 64x64 blocks and subsequently into 3x3 blocks may result in substantial bits of pixels being skipped. Furthermore, the fuzzy edge identification process requires significant processing resources.

Similarly, the approach presented in [8] utilizes a pixel-level complexity measure based on the variation between a central pixel and its neighboring pixels. The cover image is divided into overlapping blocks of 3x3, and a multidirectional high-pass filter bank is used to find eight residual responses. These responses are then utilized to apply a proposed Complex Block Prior (CBP) criterion to determine complexity. Subsequently, the blocks are sorted in descending order of complexity, and those with the same complexity level are grouped. Finally, a single bit or multiple bits are adaptively embedded in the central pixel. Although CBP offers advantages in complexity estimation and adaptive embedding, it is a pixel-oriented approach that may not integrate easily with block inversion methods. Additionally, the pixel-oriented approach can introduce significant computational overhead, and using the entire image for embedding may impact both imperceptibility and security.

An image steganography technique that adapts to an image's content is proposed in [10]. This method employs complex, busy pixels to achieve a high embedding capacity, ensuring that confidential data is embedded in regions less likely to be detected. To identify areas with high busyness, the cover image is divided into 3×3 blocks, and to calculate the energy of each pixel, an energy function based on Ising's lattice spin glass model is used. The energy function is computed as the sum of the differences between the central pixel and its neighbors. However, because energy estimation relies on the sum of the differences between nearby pixels, outliers can affect the technique and lead to inaccurate results. Additionally, the use of an energy function based on Ising's lattice spin-glass model adds computational complexity to the process.

The method in [11] combines Hamming codes, zig-zag block scanning, and canonical Gray code decomposition to better capture directional variations, thus improving accuracy over traditional row/column estimation. Although this approach enhances embedding capacity and imperceptibility, it faces misclassification in some smooth regions and higher distortion at large payloads. The Contrast Channel Embedding (CCE) framework proposed by [12] introduces mid- and extreme-contrast embedding (MCCE and ECCE), using LSB substitution to achieve high peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) performance with low complexity. However, its static thresholding can cause inconsistent embedding capacity and visible artifacts in visually sensitive areas. A steganography technique based on inverted LSB substitution is proposed in [13]. The message bits are encoded within the color channels of each pixel, utilizing the intensity levels. For every pixel, two bits are inserted into the dominant color channel, and three bits are inserted into each recessive color channel. The embedding capacity is maximized by concealing eight bits within a single pixel. This technique improves security by rotating the bits within and across each color channel for every pixel. Imperceptibility is enhanced by inverting the bits of the message per channel. However, the embedding process is applied uniformly to all image regions without considering their specific characteristics, which may lead to noticeable signs of the embedding.

The method described in [14] employs graded fuzzy edge detection and pixel classification to guide adaptive LSB embedding, enhancing imperceptibility and resistance to pixel difference histogram (PDH) and RS steganalysis attacks. However, it introduces computational overhead and performs poorly on low-contrast images. The approach based on the difference of Gaussians (DoG) [15] embeds data in edge regions to improve capacity and visual quality. The preprocessing utilizes DoG, contrast level adaptive histogram equalization (CLAHE), and Otsu thresholding to enhance edge clarity. However, these additional steps increase complexity, and performance declines on images with few edges. Similarly, the authors in [16] suggested employing the LSB approach along with dilation in edge regions for data concealment. Data are hidden by performing an XOR operation with the value of the pixel's MSB. It is essential to save and conceal the image edge information for the extraction process, as edge detection utilizes the entire pixel value. By combining the LSB technique with dilation, this method improves imperceptibility, ensuring that the embedded data remains undetectable to human observers. However, this approach reduces embedding capacity since only the LSB is used in edge pixels. Additionally, the computational burden increases due to the dilation process and the need to preserve edge information. A steganographic technique called the Strong Edge Detection Algorithm (SEDA) was introduced in [17]. This technique identifies edges using Canny edge detection and selects only strong (thin) edges, defined as pixels surrounded by edge pixels, for data embedding. It randomly selects strong pixels using the Chirikov map and employs two LSBs for embedding. This approach balances embedding capacity and imperceptibility while the randomness generated by the Chirikov map enhances security and prevents predictability. However, images with few edges may not support larger data payloads, and the

use of dual LSBs introduces relatively high distortion, potentially compromising imperceptibility.

The approach in [18] combines Otsu's thresholding, image segmentation, and Canny edge detection to identify high-frequency texture regions. A PRNG is utilized to randomize edge-pixel selection, improving resistance to simple statistical steganalysis and to achieve high imperceptibility. However, in some cases, it may miss smooth, visually important regions, and because embedding is limited to edge regions, capacity decreases in smoother images. Furthermore, the evaluation does not comprehensively address robustness against CNN-based steganalysis, compression, or geometric attacks. In [19], data are embedded along Maximum Energy Seams (MES) defined by high-gradient paths, leveraging the masking effect of the human visual system to conceal changes. Although this improves quality and capacity, the method may alter perceptually important regions and exhibit limited adaptability under compression. Similarly, the modified MAT-based scheme proposed by [20] embeds data only in highly textured blocks determined by pixel deviation from the block median. While this improves imperceptibility, it displays sensitivity to noise and outliers that can cause misclassification.

The adaptive block-based method proposed by [21] employs linear hash functions with chessboard-partitioned 8×8 blocks, classifying regions by local variance to balance payload and imperceptibility. Although efficient and statistically robust, reliance on variance alone makes the system sensitive to noise, leading to inconsistent embedding decisions. Similarly, [22] proposed an adaptive LSB method that adjusts embedded bits based on local standard deviation within 5×5 blocks, achieving reasonable PSNR values, though the standard deviation measure remains sensitive to noise and compression. The deep learning–based steganography system (DLSS) for region selection suggested by [1] uses hierarchical neural networks to extract pixel-level features such as intensity, contrast, and spatial proximity, allowing adaptive embedding in low-visibility regions. The method achieves superior imperceptibility and robustness by dividing images into sub-images processed by separate neural networks; however, this comes at the cost of high computational complexity and training overhead.

2.2. PIXEL INDICATOR TECHNIQUES (PITS)

PITs represent another major class of steganography algorithms that enhance control and adaptability by designating one color channel as an indicator and others as data carriers. These methods determine embedding locations based on predefined sequences, pixel intensities, or parity checks, offering flexible embedding with minimal distortion. In [23], the technique uses the two LSBs of one channel to indicate the presence of secret data in other data channels. Moreover, the approach considers the pixel intensity while concealing data in channels and enhances security by utilizing the length of the secret message as a selection criterion. Although it considers pixel intensity when embedding data, resulting in minimal visual distortion, its capacity depends on the specific indicator channel and the characteristics of the cover image.

A PIT presented in [24] uses indicator channel criteria to determine the data channel order based on combinations of the MSB of the RGB channels. The selection criteria are determined by the length of the message and the specific type of parity bit used to generate a randomized sequence of RGB channels. By utilizing 24 bits per pixel, the PIT significantly increases data embedding capacity. The PIT eliminates the need for a separate encryption key during the embedding and extraction processes, enhancing usability. However, the efficacy of the PIT depends on the characteristics of the selected indicator channel. Embedding data in multiple LSBs increases the risk of visual distortion.

To strengthen security and payload capacity, [25] proposed an enhanced LSB indicator method that embeds data in random bit positions across RGB channels. The red channel serves as an indicator, while the green and blue channels carry hidden data using variable LSB depths (2–4 bits). This approach improves embedding flexibility and achieves reasonable PSNR values for lower bit depths. However, its reliance on pixel intensity and random placement makes it vulnerable to noise and compression artifacts, thus reducing robustness in lossy formats. Similarly, [26] proposed a multi-layer LSB exchange method (MLLSBEM) that encrypts and compresses the secret message before embedding. The embedding is done by expanding the LSB substitution method to exchange the secret message bits with the pixels of the cover image using a shared generated key between the sender and the receiver. The two LSBs are utilized as indicators for pixel pair mapping. The compression increases capacity, while the encryption potentially improves security. However, changes to the LSBs can introduce distortions, especially if the secret message bits misalign with the LSBs. The technique proposed for data hiding in [27] uses the LSB of the green channel to preserve the channel bits. To enhance security, the secret bits are not directly embedded in the LSBs but are instead encoded using binary encoding. The technique uses the green channel as an indicator because the human eye is more sensitive to green light [28]. To enhance security, the technique encodes secret bits before embedding, thus adding a layer of complexity that makes it harder for steganalysis algorithms to detect the hidden message. Nevertheless, the technique's efficacy heavily relies on the properties of the green channel, which may limit the achievable payload. To improve imperceptibility and capacity, the technique presented in [29] combines Deflate compression (LZ77 + Huffman coding) with LSB embedding. The second bit of each of the three RGB pixel channels is used to determine the embedding channel. While this approach offers potential benefits in terms of capacity and imperceptibility, its effectiveness

depends on the compression ratio. Additionally, utilizing only one channel for embedding significantly affects embedding capacity.

In summary, although region-based and content-adaptive methods have achieved strong imperceptibility and improved embedding accuracy, they continue to face challenges such as reduced capacity in low-edge images, noise sensitivity, and inaccurate region selection caused by limited spatial analysis. Similarly, PIT methods offer simplicity and adaptability but suffer from predictable embedding patterns, sequential data placement, and vulnerability to noise and detection. These limitations highlight the need for a more adaptive embedding framework that integrates pixel similarity and textured block analysis while employing non-sequential, region-restricted embedding and secret-bit manipulation to better balance imperceptibility, capacity, and robustness.

3. CORE PRINCIPLES

This section will thoroughly explain the proposed method, which introduces two key innovations:

- a pixel similarity-based block complexity measure that improves region selection for data embedding, and
- an EPIT-LSBMR to improve imperceptibility and security.

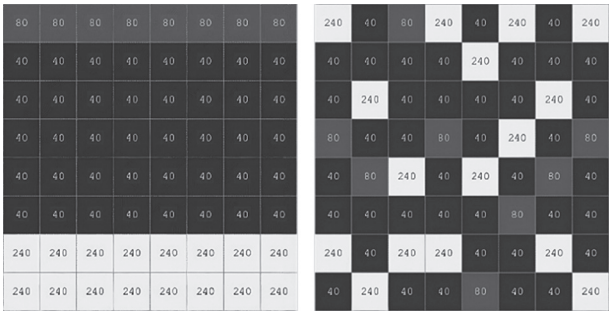
3.1. LIMITATIONS OF TRADITIONAL STATISTICAL MEASURES

Common complexity measures in image steganography, such as standard deviation and entropy, quantify pixel intensity variation but neglect the spatial arrangement of pixels. This often leads to incorrect classification of regions, as blocks with similar statistical values may have vastly different structures (e.g., smooth vs. textured).

Let B be a block of size $N \times N$, with mean μ and standard deviation σ. The standard deviation is given by:

$$\sigma = \sqrt{\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(I(i,j)\mu)^2} \quad (1)$$

Here, $I(i, j)$ is the pixel's intensity at position (i, j). While this metric captures intensity dispersion, it treats pixels as independent values and ignores local texture, leading to false positives in some cases when identifying complex regions. As shown in Figs. 2 (a) and (b), two blocks with the same standard deviation can exhibit entirely different spatial structures, one representing a complex block and the other a smooth block. This clearly demonstrates that standard deviation, similar to other metrics such as variance and entropy, is insufficient as it fails to distinguish between such cases. This limitation highlights the need for a more robust complexity measure that considers both intensity variation and spatial arrangements.



(a) (b)

Fig. 2. Two 8×8 images with identical standard deviations of 85.375: (**a**) non complex block with smooth areas, and (**b**) complex block with intensity variations

3.2. PROPOSED PIXEL SIMILARITY-BASED COMPLEXITY MEASURE

To address the above limitation, we propose a local similarity-based measure that evaluates how similar each pixel is to its neighbors. This approach captures spatial patterns and reduces sensitivity to outliers.

Let f_i denote the number of neighbors of pixel I whose values are sufficiently close to be treated as equal under the quantization operation, which maps similar intensity levels to a single representative value. The total similarity for a block is defined as:

$$f_i = \sum_{j \in \omega(i)} \delta((I(i)), I(j)) \quad (2)$$

Here, $\omega(i)$ is the set of neighboring pixels (using either the 4-neighbor concept, pixels adjacent horizontally and vertically, or the 8-neighbor concept, which includes all surrounding pixels). In this work, the similarity function $\delta(I(i), I(j))$ evaluates equality after quantization. Thus, two pixels are considered equal if their intensities fall within the same quantized interval. This ensures that visually similar pixels are treated consistently, improving robustness to minor variations and yielding a more reliable complexity assessment.

The block complexity (normalized block complexity), C_B, is then defined as:

$$C_B = 1 - F/F_{max} \quad (3)$$

where:

$$F = \sum_{i=1}^{2^N x 2^N} f_i \quad (4)$$

Here, F denotes the total sum of quantized pixel similarities within the block, and F_{max} represents the maximum possible sum of quantized pixel similarities within a block. This occurs when every quantized pixel in the block has the same intensity value as all of its neighbors, meaning that all pixel pairs in the neighborhood comparisons are identical. In this case, for each pixel i, the similarity f_i to its neighbors reaches its highest value because $\delta(I(i), I(j)) = 1$ for all neighboring pixels $j \in \omega(i)$. When this condition holds for all pixels in the block, the total similarity F reaches its theoretical maximum F_{max}.

The value of F_{max} depends on the block size NxN and the selected neighborhood model (4- or 8-neighbor). It is obtained by summing the valid neighbor connections of all pixels, which fall into three categories—corners, edges, and interiors—as follows:

For the 4-neighbor concept:

$$F_{\max} = 4 \times 2\ (corenrs) + 4(N-2) \times 3(edges) + (N-2)^2 \times 4\ (interiors)$$

which simplifies to:

$$F_{\max} = 4N^2 - 4N \quad (5)$$

For the 8-neighbor concept:

$$F_{\max} = 4 \times 3\ (corenrs) + 4(N-2) \times 5(edges) + (N-2)^2 \times 8\ (interiors)$$

which simplifies to:

$$F_{\max} = 8N^2 - 12N + 4 \quad (6)$$

In both cases, the first term corresponds to the four corner pixels with limited neighbors, the second to edge pixels with partial neighborhoods, and the third to interior pixels with full neighborhoods. Boundary pixels are evaluated only over existing neighbors (i.e., without padding). These closed-form expressions ensure that $C_B = 1 - F/F_{max}$ remains properly normalized between 0 and 1, where 0 denotes maximum similarity (a smooth block) and values approaching 1 indicate higher complexity and greater spatial variation.

Unlike traditional statistical measures, the proposed metric captures the spatial arrangement of pixels by evaluating the local relationships between neighboring pixels. This enables more accurate identification of complex regions. Furthermore, the metric is robust to outliers because it relies on local pixel similarities rather than statistical descriptors, which tend to be more sensitive to anomalies.

3.3. THRESHOLD-BASED EMBEDDING REGION SELECTION

To automate the selection of suitable blocks for embedding, we use the mean complexity across all blocks as a threshold T:

$$T = \frac{1}{NB}\sum_{i=1}^{NB} C_B^{(i)} \quad (7)$$

Here, NB denotes the total number of blocks. Blocks with $C_B > T$ are selected for embedding. Since T is derived from the statistical distribution of block complexities within each image, it naturally adapts to variations in content. This approach ensures that the embedding process dynamically targets regions whose complexity exceeds the average complexity of the image, thereby maximizing embedding effectiveness.

3.4. INTEGRATION OF EPIT WITH LSBMR

Traditional PIT methods embed data sequentially, rely on fixed indicator rules, and do not consider visual sensitivity or encoding robustness. To address these shortcomings, we propose an EPIT approach that incorporates the following components:

- Repetition-based encoding: Secret bits are grouped and encoded based on their repetition counts.
- Indicator alignment: Indicator bits are adjusted so that they match the bit type of the corresponding encoded values.
- LSBMR embedding: Two bits are embedded into pixel pairs using the LSBMR algorithm, which reduces the average number of pixel modifications compared with conventional LSB.

By integrating EPIT with LSBMR (EPIT-LSBMR), the proposed method reduces visual distortion, enhances embedding efficiency, and offers stronger resistance to statistical steganalysis.

4. THE PROPOSED METHOD

The proposed method enhances image steganography by embedding data in visually complex regions using a pixel similarity-based complexity measure and a novel EPIT-LSBMR scheme. The embedding process consists of three main components:

1. identification of complex image blocks using pixel similarity,
2. pre-processing of the secret data using repetition-based encoding, and
3. embedding using the EPIT and LSBMR techniques in the selected regions.

A high-level overview of the embedding process is presented in Fig. 3. The secret data are first pre-processed, and the cover image is partitioned into non-overlapping blocks for complex block identification. This stage involves grayscale conversion, LSB resetting, block complexity calculation, threshold estimation, and the selection of high-complexity blocks. The pre-processed secret data is then embedded into the selected blocks using EPIT-LSBMR, with the red and blue channel LSBs inverted when it reduces distortion, resulting in the final stego image.

4.1. COMPLEX BLOCKS IDENTIFICATION

The method for identifying complex blocks is detailed in the steps below:

1. The RGB image is converted to grayscale.
2. The LSB of each pixel is set to zero to ensure consistent block complexity computation during both embedding and extraction.

The grayscale image is quantized into L intensity levels (e.g., $L = 6$) using a step size $D = 255/L$. Each interval r_i ranges from $((i-1) \times D + 1)$ to $i \times D$, with r_1 beginning at 0 and r_L ending at 255.

The parameter L determines the number of discrete intensity intervals used during quantization. It must be an integer $L \geq 2$, with an upper bound of $L = 256$, which corresponds to the original 8-bit pixel precision (i.e., no quantization). Very small values ($L = 2$–4) produce a heavily degraded representation by merging distinct textures, whereas very large values ($L \geq 128$) closely approximate the original image and cause the complexity measure to depend on minor intensity variations. Thus, moderate choices of L values (e.g., 6–32) provide a suitable balance between sensitivity, robustness, and computational efficiency.

The quantization step $D = 255/L$ defines the width of each intensity interval and is inversely proportional to L. Smaller D values increase sensitivity to fine texture details, while larger ones improve noise tolerance and stability.

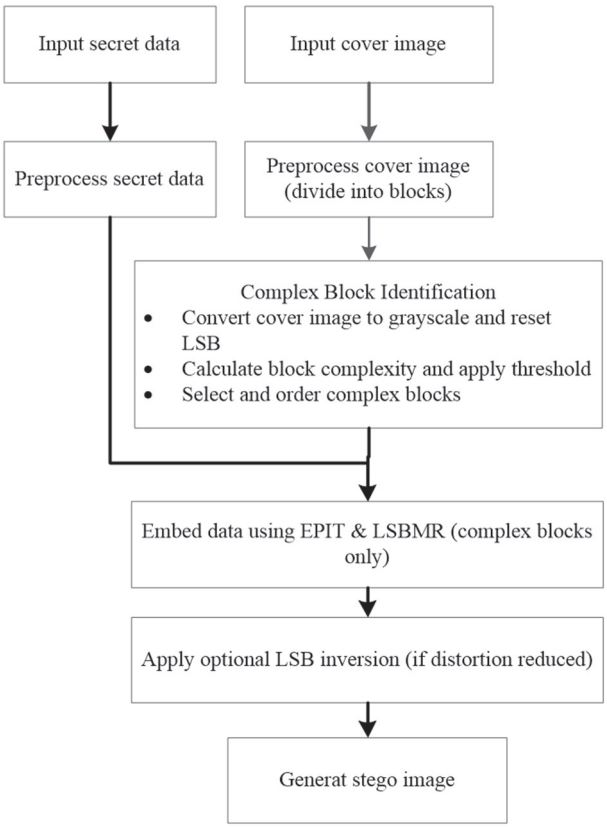


Fig. 3. Proposed steganographic technique: embedding phase

3. The image is divided into non-overlapping blocks of size 2^Nx2^N, where N is chosen to ensure the blocks are neither too small nor excessively large (e.g., $N = 3$, 4, or 5).
4. For each pixel $P(x, y)$ within a block, its similarity f_i is computed with neighboring pixels using Eq. (2), based on either the 4-neighbor model $(x \pm 1, y)$, $(x, y \pm 1)$ or the 8-neighbor model (all surrounding pixels). The block complexity C_B is computed using Eq. (3). Fig. 4 illustrates the stepwise computation of block complexity under the 8-neighbor concept.
5. The threshold T is determined as the average block complexity using Eq. (7). The threshold T represents the overall texture level across the image and effectively separates complex (edge-rich or high-texture) and smooth (low-texture) regions.
6. Only blocks with complexity $C_B > T$ are selected for embedding, as these blocks are highly textured and therefore suitable for concealing secret data. The threshold T can be adjusted to balance imperceptibility and embedding capacity using $T' = \alpha \cdot T$, where $\alpha \in [0.8, 1.2]$. providing a tunable balance between embedding capacity and imperceptibility. This flexible scaling allows precise control over both visual quality and payload.

Original Block

$$\begin{bmatrix} 10 & 12 & 60 & 58 \\ 15 & 14 & 65 & 70 \\ 200 & 198 & 190 & 185 \\ 205 & 202 & 180 & 175 \end{bmatrix}$$

Quantized Block $Q = (I/D)$, $D = 255/6 \approx 42.5$

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 5 & 5 & 4 & 4 \\ 5 & 5 & 4 & 4 \end{bmatrix}$$

Similarity Counts f_i(8-neighbor)

$$\begin{bmatrix} 3 & 3 & 1 & 1 \\ 3 & 3 & 1 & 1 \\ 3 & 3 & 3 & 3 \\ 3 & 3 & 3 & 3 \end{bmatrix}$$

$$F = \sum f_i = 40$$
$$F_{\max} = 8N^2 - 12N + 4 = 84$$
$$C_B = 1 - F/F_{\max} = 1 - 40/84 \approx 0.52$$

Fig. 4. Example of block complexity computation using a 4×4 block and 8-neighbor similarity

The average complexity across all blocks is used as the adaptive threshold because it provides a simple, reproducible, and data-driven criterion that naturally separates smooth and complex regions; typically, approximately half of the blocks fall below this value and half above it, depending on the distribution [30]. However, zero-complexity blocks can bias the mean, so excluding them yields a more realistic threshold, which practically corresponds to multiplying the mean by $\alpha = 1.2$. Varying α between 0.8 and 1.2 enables sensitivity control: lower values increase embedding capacity by including more moderately smooth blocks, while higher values enhance imperceptibility and robustness by restricting embedding to highly textured regions. For example, in the Pepper image (8x8 blocks, $T = 0.143$), the embedding capacities were approximately 18 KB, 14 KB, and 12 KB for $\alpha = 0.8$, 1.0, and 1.2, respectively, illustrating the trade-off among payload, imperceptibility, and detectability.

Similarly, using an 8-neighbor concept and a block size of 16×16 pixels, the yellow squares in Figs. 5(a–c) represent complex blocks whose complexity threshold is determined by the average complexity of all image blocks. Fig. 5(d) shows the highest-complexity blocks, which can accommodate smaller payloads. The complexity values of these blocks range from 0.6387 to 0.6968, making them suitable for embedding a 1 KB payload.

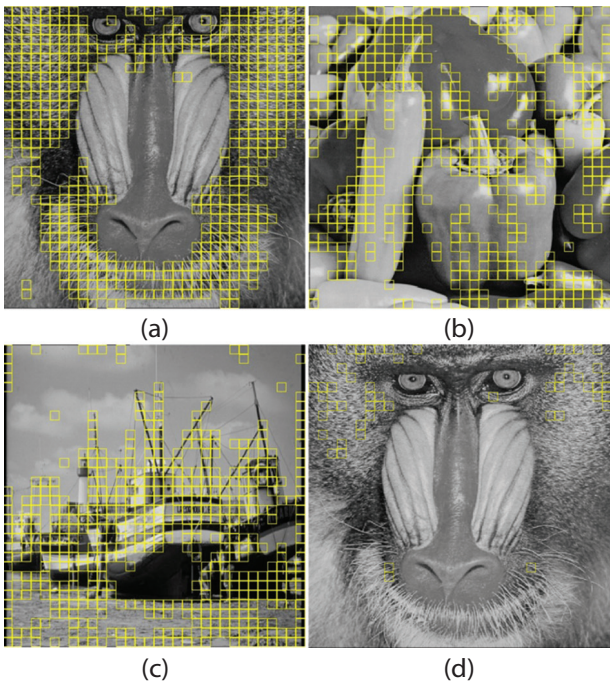


Fig. 5. Complex-block selection using the 8-neighbor concept and 16×16 blocks. Yellow squares in (**a**–**c**) indicate blocks exceeding the threshold, while (**d**) shows the highest-complexity blocks (0.6387–0.6968)

4.2. EMBEDDING PROCESS

All algorithm parameters—including data size, quantization level, block size, and the neighbor concept—are deterministically derived from a single shared secret key using a hash-based key derivation function (e.g., HKDF). The embedding process then hides the secret data within complex blocks using the EPIT-LSBMR method. To minimize distortion introduced during embedding, the LSBs of a block may be inverted when this reduces the MSE. The LSB of the top-left pixel in grayscale images, or of the blue channel in RGB images, serves as a flag indicating inversion (1 = inversion of LSBs, 0 = no inversion). This mechanism ensures minimal visual distortion while maintaining accurate data recovery.

Blocks are selected non-linearly based on their complexity to enhance security and randomness. This study extends and improves upon the PIT method in [27] by refining the handling of indicator bits and integrating the LSBMR embedding scheme. The proposed EPIT-LSBMR-X algorithm embeds data exclusively within complex blocks, where "X" denotes embedding restricted to complex regions. Before detailing the embedding process, the message preprocessing steps and the LSBMR technique are introduced.

4.2.1. Secret message pre-processing

The secret message, provided as a text file, is first converted into a binary stream. This stream is then processed to generate two separate streams: one containing the bit values (0 or 1) and the other containing the corresponding number of repetitions for each bit value. In our design, the repetition count is encoded using two bits, which support exactly four possible states. Consequently, the maximum number of repetitions for any bit value is limited to four. If a sequence of identical bits exceeds this limit, the count is reset and a new run is recorded. For example, the binary stream 1101111100 is encoded as follows:

- Bit values: 1 0 1 1 0
- Repetitions: 2 1 4 1 2

This encoding reduces redundancy and enables more compact embedding.

4.2.2. LSBMR

LSBMR is an improved LSB-based steganography method that embeds two secret bits using only a single modification across a pair of pixels, thereby enhancing imperceptibility compared to standard LSB techniques [31]. The method consists of two phases: embedding and extraction. During embedding, two secret bits, m_i and m_i+1, are concealed within a pixel pair (x_i, x_i+1). The method follows the diagram shown in Fig. 6, which illustrates the different matching cases used to determine the appropriate embedding strategy. The LSB of x_i is compared with m_1, and the value of the decision function defined in Eq. (8) is compared with m_2 to select the appropriate embedding case.

$$F(x_i, x_{i+1}) = LSB(\left\lfloor \frac{x_i}{2} \right\rfloor + x_{i+1}) \tag{8}$$

In this work, m_i and m_i+1 denote two consecutive secret bits from the message stream, specifically m_i = MSB(f) and m_i+1 = LSB(f), derived directly from the bit-repetition factor f. These bits are then jointly embedded into the corresponding pixel pair (x_i, x_i+1).

For example, let x_i=28, x_i+1= 30, and the secret bits be m_1=0 and m_2=1. First, we compute LSB(x_i)=0 and compare it with m_1. Since LSB(x_i)=m_1, the modified pixel value becomes x'_i=28, according to Fig. 6. Next, we compute F(28, 30) and compare it with m_2 to obtain x'_i+1.

$$F(28,30) = LSB\left(\left\lfloor \frac{28}{2} \right\rfloor + 30\right) = LSB(16 + 30) = 0 \neq m2$$

Therefore, according to the case shown in Fig. 6, the updated value is x'_i+1 = 30 + 1 = 31. The modified pixel pair becomes x'_i = 28, and x'_i+1 = 31. The equation for extracting the two embedded bits, m_1 and m_2, is as follows:

$$m_1 = LSB(x'_i),\ m_2 = LSB(\left\lfloor \frac{x'_i}{2} \right\rfloor + x'_{i+1}) \tag{9}$$

Using the modified pixels x'_i=28 and x'_i+1=31, the extracted bits are:

$$m_1 = LSB(28) = 0,\ m_2 = LSB\left(\left\lfloor \frac{28}{2} \right\rfloor + 31\right) = 1$$

Thus, m_1 =0 and m_2 =1.

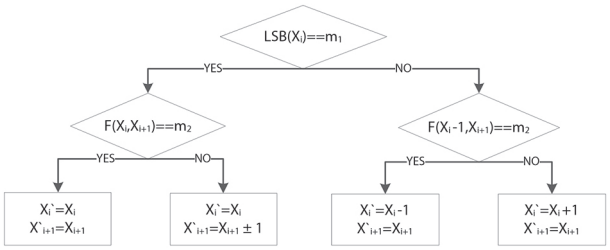


Fig. 6. LSBMR algorithm diagram showing the different cases of the matching process

4.2.3. Embedding techniques

The embedding process differs slightly for RGB and grayscale images. Secret bits are first encoded using EPIT and then embedded using the LSBMR algorithm, with the green channel serving as the indicator. The following subsections describe each procedure in detail.

i. Embedding process in RGB images

In this method, the LSB of the green channel (G_1) encodes the type of the current secret bit (0 or 1), while the LSBs of the red and blue channels (R_1, B_1) encode its repetition count. The triplet ($G_1R_1B_1$) encodes both the bit value and its frequency according to the mapping in Table 1. For example, a bit value of 1 with one repetition is encoded as $(1\ 01)_2$; one with two repetitions; and so on. To reduce pixel alterations, the encoded bits and the repetition are embedded into the R and B channels using the LSBMR technique, yielding the enhanced method EPIT-LSBMR-X. For each pixel, G_1 is modified only when it differs from the current bit value, while R and B are adjusted using LSBMR according to the repetition encoding.

Table 1. Indicator and encoded values

	Indicators G1	Embedded secret bits	Encoded value R1 B1
Zeros indicator	0	0 0	4 zeros
		0 1	1 zero
		1 0	2 zeros
		1 1	3 zeros
Ones indicator	1	0 0	4 ones
		0 1	1 one
		1 0	2 ones
		1 1	3 ones

As an example, consider the secret bitstream 10001110, which is first pre-processed into bit-repetition pairs: (1,1), (0,3), (1,3), and (0,1). For the first pair (1,1), the LSB of the green channel (G_1) is set to 1 to match the bit value. The repetition count (1) is then encoded as $(01)_2$, representing the target bits to be embedded. These bits are embedded into the red and blue channels (R and B) using the LSBMR technique, which minimally modifies pixel values to preserve visual quality. Using the cover pixel values R = 80, B = 81 from Table 2, and the secret bits, $m_1 = 0$, $m_2 = 1$ (the repetition count), LSBMR adjusts B to 80, resulting in updated values: R′ = 80, G′ = 79, B′ = 80. This process is repeated for all four bit-repetition pairs, producing the stego pixel values presented in Table 3. The overall embedding procedure is outlined in Algorithm 1.

Table 2. Four RGB cover pixels

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	01010000_2 $(80)_{10}$	01101100_2 $(108)_{10}$	01101101_2 $(109)_{10}$	10001110_2 $(142)_{10}$
G	01001111_2 $(79)_{10}$	10000001_2 $(129)_{10}$	10011000_2 $(152)_{10}$	10101010_2 $(170)_{10}$
B	01010001_2 $(81)_{10}$	01011010_2 $(90)_{10}$	10001001_2 $(137)_{10}$	10110010_2 $(178)_{10}$

Table 3. Four RGB stego pixels

	Pixel 1	Pixel 2	Pixel 3	Pixel 4
R	01010000_2 $(80)_{10}$	01101100_2 $(108)_{10}$	01101100_2 $(108)_{10}$	10001111_2 $(143)_{10}$
G	01001111_2 $(79)_{10}$	10000000_2 $(128)_{10}$	10011001_2 $(151)_{10}$	10101010_2 $(170)_{10}$
B	01010001_2 $(81)_{10}$	01011010_2 $(90)_{10}$	10001001_2 $(137)_{10}$	10110010_2 $(178)_{10}$

Algorithm 1

EPIT-LSBMR embedding Algorithm for RGB images

Input:
  RGB cover image C
  Pre-processed secret message M= [($bv1$, $f1$), ($bv2$, $f2$), ..., (bvn, fn)]
Output:
  Stego image S
Variables:
  G: Indicator channel (Green) of the cover image
  R, B: Red and Blue channels of the cover image
  R′, B′: Modified Red and Blue channels for stego image
  k: Message index
  $m1$, $m2$: Bits from the repetition factor f
  msg_limit: Length of the message (number of tuples)
  $\mathfrak{B}$: Set of selected blocks with $C^B > T'$
  Initialize $k \leftarrow 0$
  Set $S \leftarrow C$
For each block $B_p \in \mathfrak{B}$ do
  For each pixel $(i, j) \in B_p$ do
    If $k \geq msg_limit$ then
      Exit loops
  End If

  Extract $G\,[i, j]$, $R\,[i, j]$, $B\,[i, j]$
  Extract bv $\leftarrow M[k].bv$
  Extract $f \leftarrow M[k].f$
  If $LSB\,(G\,[i, j]) \neq bv$ then
    Set $LSB\,(G\,[i, j]) \leftarrow bv$
  End If

  $m1 \leftarrow MSB(f)$
  $m2 \leftarrow LSB(f)$
  Apply LSBMR to $R\,[i, j]$ and $B\,[i, j]$ using $m1$

29. and $m2$
30. Get modified values $R'[i, j]$, $B'[i, j]$

31. Set $S[i, j]$. $G \leftarrow G[i, j]$
32. Set $S[i, j]$. $R \leftarrow R'[i, j]$
33. Set $S[i, j]$. $B \leftarrow B'[i, j]$
34. $k \leftarrow k + 1$
35. End For
36. End For

37. Return S

ii. Embedding process in grayscale images

The embedding process for grayscale images follows the same EPIT-LSBMR principle as the RGB version, but it operates on three consecutive pixels rather than on color channels. Each triplet of pixels (x_i, x_{i+1}, x_{i+2}) serves a role analogous to the (G, R, B) components in a color image: the first pixel x_i functions as the indicator pixel, while x_{i+1} and x_{i+2} constitute the data-embedding pair carrying the repetition bits. To further reduce distortion, the technique operates on two adjacent triplets, (x_i, x_{i+1}, x_{i+2}) and (x_{i+3}, x_{i+4}, x_{i+5}). In this configuration, the indicator pixels x_i and x_{i+3} are modified using LSBMR to embed two consecutive bit types of the secret message, which correspond to the bits m_1 and m_2. The data-carrying pairs (x_{i+1}, x_{i+2}) and (x_{i+4}, x_{i+5}) are then used to embed the associated repetition counts, also through LSBMR. This procedure defines the EPIT-LSBMR-X-Gray technique.

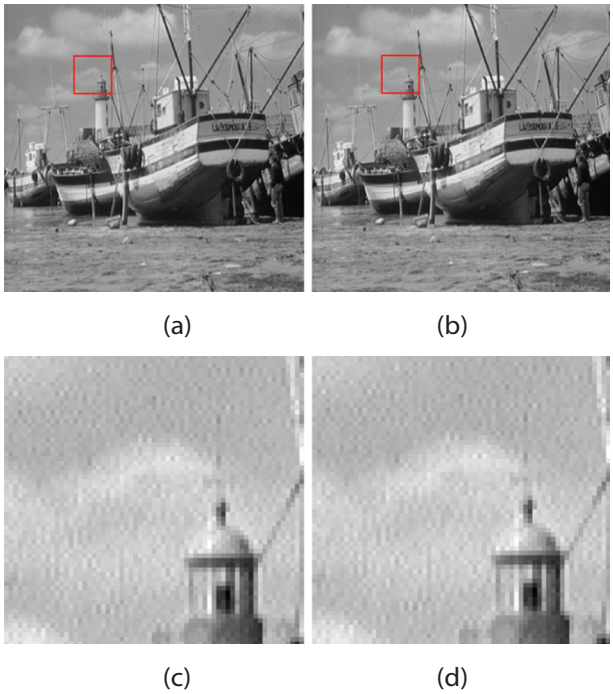
For example, given the secret bitstream 1000_2, it is first transformed into bit type and repetition counts: (1, 1) and (0, 3). Using the cover pixel values in Table 4, the two triplets (x_1 =72, x_2 =66, $x3$ = 46) and (x_4 =76, x_5 =56, x_6 = 66) are used to embed these pairs. The bit types 1_2 and 0_2 are embedded into the indicator pixels x_1 and x_4 using LSBMR, producing the modified values x'_1 =73 and x'_4 =76. The repetition value 1 corresponds to the binary representation $(01)_2$, which is embedded into (x_2, x_3) using LSBMR, resulting in x'_2=66 and x'_3=46. The repetition value 3 corresponds to $(11)_2$, and embedding it into (x_5, x_6) via LSBMR yields x'_5=56 and x'_6=66. The embedding results are provided in Table 5. Fig. 7 presents the cover image alongside the corresponding stego image, and their zoomed regions show no visible differences after embedding 16 KB using the 8-neighbor concept and a 32×32 block size.

Table 4. Six cover pixels

Pixel No.	Pixel ID	Pixel value
1	x_1	$(72)_{10}$
2	x_2	$(66)_{10}$
3	x_3	$(46)_{10}$
4	x_4	$(76)_{10}$
5	x_5	$(56)_{10}$
6	x_6	$(66)_{10}$

Table 5. Six stego pixels

Pixel No.	Pixel ID	Pixel value
1	x_1	$(73)_{10}$
2	x_2	$(66)_{10}$
3	x_3	$(46)_{10}$
4	x_4	$(76)_{10}$
5	x_5	$(55)_{10}$
6	x_6	$(66)_{10}$



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

Fig. 7. Visual comparison of cover and stego images with 16 KB embedded. Subplots (**a**)–(**b**) show the original and stego images with zoom markers, and (**c**)–(**d**) present the enlarged regions

4.3. THE EXTRACTION PROCESS

The extraction process begins by converting the RGB stego image to grayscale and resetting the LSB of each pixel to zero, followed by identifying complex image blocks. The subsequent steps vary slightly depending on whether the image is grayscale or RGB. For grayscale images, the algorithm first checks the LSB of the initial pixel in each block to determine whether LSB inversion was applied; if so, the corresponding LSBs are toggled. Three consecutive pixels are then read: the first acts as an indicator, and the next two store the encoded bits. The indicator, represented by the LSB of the first pixel, specifies the secret bit value, while its repetition count is embedded in the remaining two pixels. The LSBMR method is applied to extract these two bits, which are then decoded using EPIT and Table 1 to reconstruct the original data. For RGB images, the LSB of the blue channel in the first pixel indicates whether LSB inversion was applied within the block. If inversion is detected, the relevant LSBs are toggled. The red and blue channels are subsequently processed using LSBMR to extract the two bits representing the repetition count, whereas the

LSB of the green channel stores the actual secret bit type. This procedure iterates across all blocks until the complete secret message is recovered. An overview of the extraction workflow is provided in Fig. 8.

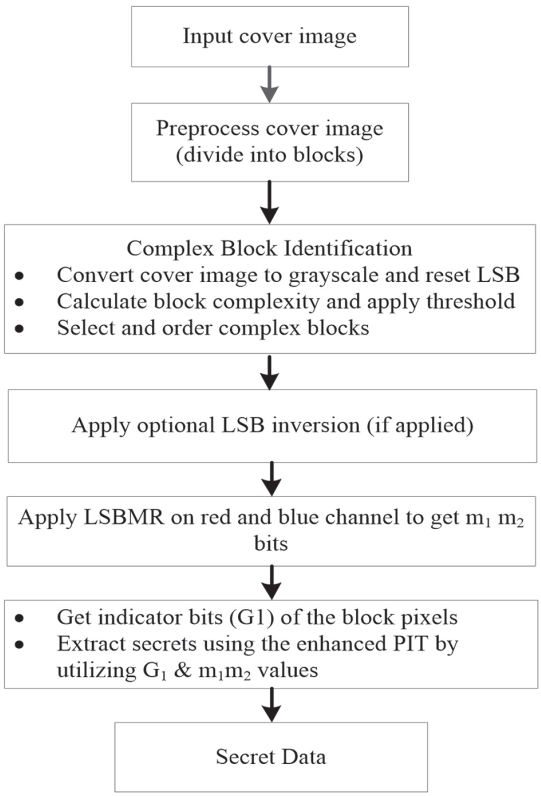


Fig. 8. Proposed steganographic technique: extraction phase

5. EXPERIMENTAL SETTING, RESULTS, AND ANALYSIS

The performance of the proposed method is assessed and evaluated in this section through comparative experiments. The experimental parameters are presented in Table 6.

5.1. EXPERIMENTAL SETTING AND DATASET

The technique was implemented on a PC equipped with an Intel Core i7 x64 CPU running at 2.8 GHz and 8 GB of RAM. MATLAB 2022a was used due to its readily available toolboxes, which facilitate implementation. Three image datasets were employed in the experiments. The first dataset is the USC-SIPI dataset [32], which consists of standard grayscale and 24-bit RGB benchmark images, such as Peppers, Boat, Airplane, and Baboon, as shown in Fig. 9. Each image has a resolution of 512×512 pixels. The second dataset, BOSSbase 1.01, is a benchmark collection of 10,000 raw grayscale images in PGM format that is widely adopted in the literature [33]. Additionally, the RGB-BMP Steganalysis Dataset [34] was utilized for the RGB experiments; it contains 3,000 RGB-BMP images with dimensions of 512×512 pixels, commonly used in steganography, steganalysis, and related image-processing research.

The secret messages were stored in text files ranging from 1 KB to 32 KB. These files were randomly generated online using ONLINEFILETOOLS [35] and saved as text files to serve as confidential data. The proposed technique is evaluated in terms of imperceptibility (image quality) and security.

Table 6. Experimental design

Technique	EPIT-LSBMR-X-Gray & EPIT-LSBMR-X
Neighbouring concept	4 8
Block size	8x8 16x16 32x32
Metrics	• PSNR • wPSNR • SSIM • Difference maps and pixel-difference histograms • Chi-Square • RS • SRM+Ensemble classifier • XuNet

Fig. 9. Sample of standard images

5.2. IMAGE QUALITY METRICS

Image quality (IQ) metrics, including PSNR, weighted PSNR (wPSNR), and the SSIM, were used to assess changes in the stego image relative to the original. The techniques were evaluated using block sizes of 8×8, 16×16, and 32×32, along with two different neighboring concepts (4- and 8-neighbor). Furthermore, we compared the obtained results with those reported in state-of-the-art studies.

All reported average values represent the simple arithmetic mean across the evaluated images, and the corresponding standard deviations are included in the tables to quantify the variability of each metric.

5.2.1. PSNR

PSNR is a standard metric for evaluating image quality in steganography. It quantifies the distortion introduced during data embedding as the ratio between the maximum possible pixel value (e.g., 255) and the MSE between the cover and stego images [36]:

$$PSNR = 10\ \log_{10} \frac{MAX^2}{MSE}. \quad (10)$$

$$MSE = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} (X_{ij} - Y_{ij}) \quad (11)$$

Here, X and Y denote the cover and stego images, and W and H represent the image width and height, respectively. Higher PSNR values and lower MSE indicate reduced distortion and improved imperceptibility.

Table 7 presents the PSNR values for EPIT-LSBMR-X-Gray across varying embedding capacities, block sizes, and neighboring concepts. The table reports the results from four standard test images, along with the average value and standard deviation of images from the BOSSbase_1.01 dataset, which were converted to BMP format using MATLAB commands. Across all configurations, the PSNR values remain high—ranging from approximately 65 dB at a 1 KB payload to around 56 dB at 8 KB—indicating minimal perceptual distortion due to data embedding. Notably, the 32×32 block size supports more efficient embedding and yields slightly higher average PSNR values compared to the smaller block sizes, with no significant differences observed between the two neighboring concepts. Overall, the consistently high PSNR values confirm that the visual quality of the images is well preserved after data hiding.

Table 7. PSNR, wPSNR, and SSIM for stego images using EPIT-LSBMR-X-Gray techniquea

Neighboring concept	EC	Image	PSNR			wPSNR			SSIM		
			8x8	16x16	32x32	8x8	16x16	32x32	8x8	16x16	32x32
8N	1KB	Airplane	65.29	65.32	65.35	82.62	98.26	99.04	0.99994	0.99996	0.99995
		Baboon	65.27	65.41	65.43	97.1	97.26	97.38	0.99987	0.99984	0.99988
		Peppers	65.34	65.4	65.39	89.78	88.61	91.29	0.99989	0.99992	0.99991
		Average	65.31	65.37	65.39	91.47	92.11	92.58	0.99990	0.99991	0.99991
		Std dev	0.046	0.042	0.042	5.77	5.99	6.34	0.00005	0.00005	0.00005
	2KB	Airplane	62.2	62.28	62.29	77.57	81.47	95.93	0.99986	0.99987	0.99988
		Baboon	62.26	62.32	62.29	93.98	94.13	94.21	0.99996	0.99995	0.99994
		Peppers	62.22	62.3	62.33	86.45	87.45	85.51	0.99984	0.99982	0.99983
		Average	62.23	62.29	62.31	87.16	87.76	88.43	0.99989	0.99988	0.99988
		Std dev	0.026	0.029	0.032	5.55	5.82	6.04	0.00009	0.00009	0.00009
	4KB	Airplane	59.3	59.33	59.38	73.03	75.3	80.59	0.99962	0.99967	0.99969
		Baboon	59.28	59.33	59.33	90.81	91.02	90.98	0.99988	0.99986	0.99985
		Peppers	59.32	59.37	59.37	83.67	85.01	82.88	0.99952	0.99955	0.99954
		Average	59.28	59.38	59.36	83.1	83.65	84.51	0.99967	0.99969	0.99969
		Std dev	0.020	0.023	0.024	4.86	5.27	5.36	0.00020	0.00020	0.00020
	8KB	Airplane	56.25	56.34	56.35	69.42	71.78	73.58	0.9993	0.99932	0.99933
		Baboon	56.26	56.34	56.35	87.23	87.49	87.63	0.99973	0.9997	0.99969
		Peppers	56.29	56.34	56.32	81.01	81.02	81.17	0.99926	0.99929	0.99926
		Average	56.26	56.33	56.34	80.69	80.96	81.46	0.99943	0.99944	0.99943
		Std dev	0.018	0.019	0.021	3.48	4.04	4.38	0.00020	0.00020	0.00030
4N	1KB	Airplane	65.27	65.33	65.39	80.18	98.13	99.01	0.99994	0.99996	0.99995
		Baboon	65.25	65.37	65.44	97.05	97.41	97.44	0.99987	0.99983	0.99989
		Peppers	65.37	65.37	65.34	88.52	88.31	85.71	0.99989	0.99991	0.9999
		Average	65.31	65.37	65.39	90.69	91.47	92.16	0.99990	0.99990	0.99991
		Std dev	0.037	0.037	0.04	6.16	6.33	6.63	0.00005	0.00005	0.00006
	2KB	Airplane	62.25	62.26	62.34	75.98	77.97	80.74	0.99986	0.99988	0.99986
		Baboon	62.19	62.32	62.31	93.98	94.32	94.37	0.99994	0.99993	0.99995
		Peppers	62.24	62.3	62.35	85.67	86.63	84.54	0.99978	0.99979	0.99977
		Average	62.23	62.3	62.31	86.63	87.36	88.11	0.99986	0.99987	0.99986
		Std dev	0.028	0.029	0.03	5.71	6.02	6.11	0.00009	0.0001	0.0001
	4KB	Airplane	59.31	59.37	59.35	72.53	74.67	80.66	0.99968	0.9997	0.99972
		Baboon	59.29	59.34	59.38	90.89	91.16	91.34	0.99990	0.99988	0.99986
		Peppers	59.3	59.33	59.36	83.39	83.79	82.75	0.99958	0.99961	0.99962
		Average	59.28	59.35	59.36	82.93	83.58	84.44	0.99972	0.99973	0.99973
		Std dev	0.022	0.0217	0.02	4.86	5.38	5.58	0.00020	0.00020	0.00020
	8KB	Airplane	56.26	56.31	56.34	69.21	71.17	73.49	0.99932	0.99935	0.99933
		Baboon	56.28	56.34	56.35	87.08	87.52	87.68	0.99976	0.99972	0.99974
		Peppers	56.26	56.31	56.34	80.83	80.74	80.97	0.99926	0.99928	0.99927
		Average	56.26	56.33	56.34	80.93	81.35	81.94	0.99945	0.99945	0.99945
		Std dev	0.016	0.019	0.02	3.36	3.67	4.22	0.00020	0.00020	0.00020

Table 8 presents the PSNR results of the proposed EPIT-LSBMR-X technique for RGB images across different block sizes and embedding capacities (1–16 KB). The PSNR values range from approximately 70 dB at 1 KB to 58 dB at 16 KB.

These values are notably high, particularly at lower embedding rates, indicating strong imperceptibility of the embedded data. Across the tested block sizes, the differences in PSNR were minimal. Specifically, using 32×32 blocks yielded a marginal average improvement over smaller block sizes, and no block size resulted in a significant performance gain.

In general, block size and image nature do not substantially affect PSNR, as the metric quantifies pixel-wise error between the cover and stego images and is insensitive to spatial structure or regional complexity.

Table 8. PSNR, wPSNR, and SSIM for stego images using the proposed EPIT-LSBMR-X

Neighboring	EC	Image	PSNR			wPSNR			SSIM		
			8x8	16x16	32x32	8x8	16x16	32x32	8x8	16x16	32x32
8N	1KB	Airplane	70.18	70.15	70.17	100.8	101.8	102.6	0.999999	0.999995	0.999997
		Baboon	70.13	70.15	70.17	88.87	89.01	89.26	0.999998	0.999992	0.999995
		Boat	70.20	70.10	70.16	91.45	92.54	92.19	0.999997	0.999994	0.999993
		Peppers	70.16	70.13	70.14	82.08	83.67	80.20	0.999996	0.999993	0.999992
		Average	70.12	70.14	70.16	90.79	91.75	91.05	0.999997	0.999994	0.999994
		Std Dev	0.045	0.041	0.039	6.716	6.969	7.722	0.000001	0.000008	0.000003
	4KB	Airplane	64.12	64.12	64.16	81.65	81.57	95.07	0.999983	0.999996	0.999997
		Baboon	64.10	64.12	64.12	82.54	82.75	82.93	0.999985	0.999995	0.999996
		Boat	64.11	64.13	64.16	83.84	84.56	85.38	0.999986	0.999994	0.999995
		Peppers	64.08	64.10	64.14	76.92	77.16	77.2	0.999984	0.999993	0.999994
		Average	64.10	64.12	64.12	81.24	81.51	85.14	0.999984	0.999995	0.999996
		Std Dev	0.028	0.027	0.025	7.477	7.641	7.821	0.000017	0.000004	0.000004
	16KB	Airplane	58.07	58.09	58.10	72.03	72.64	71.95	0.999920	0.999995	0.999995
		Baboon	58.08	58.09	58.11	76.07	76.32	76.4	0.999911	0.999993	0.999996
		Boat	58.08	58.11	58.10	75.06	75.56	76.16	0.999912	0.999991	0.999995
		Peppers	58.08	58.11	58.11	71.82	72.00	72.95	0.999914	0.999990	0.999994
		Average	58.07	58.09	58.10	73.75	74.13	74.37	0.999914	0.999992	0.999995
		Std Dev	0.019	0.017	0.020	6.628	6.816	7.336	0.000089	0.000008	0.000005
	32KB	Airplane	55.07	55.10	55.09	68.94	69.16	69.67	0.999800	0.999990	0.999994
		Baboon	55.06	55.09	55.10	71.67	72.24	72.3	0.999807	0.999992	0.999995
		Boat	55.07	55.10	55.09	71.46	71.8	72.19	0.999810	0.999993	0.999995
		Peppers	55.06	55.10	55.10	68.79	68.96	69.13	0.999802	0.999991	0.999994
		Average	55.06	55.08	55.09	70.21	70.54	70.82	0.999805	0.999992	0.999994
		Std Dev	0.018	0.020	0.022	5.073	5.756	6.408	0.000193	0.000008	0.000006
4N	1KB	Airplane	70.15	70.15	70.16	100.3	101.1	102	0.999998	0.999991	0.999993
		Baboon	70.15	70.15	70.16	89.10	88.92	89.27	0.999999	0.999994	0.999999
		Boat	70.10	70.17	70.17	88.35	92.66	93.39	0.999997	0.999992	0.999996
		Peppers	70.09	70.17	70.10	77.70	77.47	79.65	0.999999	0.999992	0.999989
		Average	70.13	70.15	70.15	88.86	90.05	91.08	0.999998	0.999991	0.999994
		Std Dev	0.044	0.034	0.041	5.398	6.042	6.697	0.000002	0.000009	0.000008
	4KB	Airplane	64.13	64.13	64.16	91.73	94.16	94.05	0.999980	0.999992	0.999985
		Baboon	64.09	64.09	64.12	82.82	82.96	82.95	0.999990	0.999995	0.999993
		Boat	64.10	64.17	64.14	83.15	84.12	85.61	0.999995	0.999990	0.999998
		Peppers	64.08	64.16	64.13	73.54	73.10	71.2	0.999995	0.999999	0.999998
		Average	64.10	64.12	64.12	82.81	83.59	83.45	0.999990	0.999994	0.999993
		Std Dev	0.023	0.024	0.025	6.395	6.735	7.160	0.000010	0.000005	0.000009
	16KB	Airplane	58.08	58.1	58.09	79.48	80.88	84.14	0.999850	0.999990	0.999992
		Baboon	58.07	58.09	58.08	76.11	76.33	76.45	0.999950	0.999995	0.999999
		Boat	58.06	58.1	58.11	75.29	76.18	77.02	0.999920	0.999997	0.999995
		Peppers	58.09	58.11	58.10	69.18	68.52	68.41	0.999990	0.999993	0.999999
		Average	58.07	58.09	58.10	75.02	75.48	76.5	0.999928	0.999994	0.999996
		Std Dev	0.016	0.017	0.017	6.424	6.652	7.130	0.000077	0.000009	0.000010
	32KB	Airplane	55.07	55.09	55.10	74.38	75.01	75.37	0.999600	0.999992	0.999996
		Baboon	55.07	55.07	55.08	71.98	72.16	72.27	0.999800	0.999996	0.999999
		Boat	55.08	55.09	55.09	71.07	71.53	72.67	0.999950	0.999999	0.999998
		Peppers	N/A	55.1	55.09	67.14	67.13	67.01	0.999870	0.999995	0.999994
		Average	55.06	55.08	55.09	71.14	71.46	71.83	0.999805	0.999999	0.999997
		Std Dev	0.018	0.019	0.022	5.385	5.805	6.706	0.000203	0.000008	0.000003

5.2.2. wPSNR

The wPSNR enhances the conventional PSNR by incorporating the sensitivity characteristics of the human visual system (HVS) [37]. Unlike PSNR, which treats all pixel errors equally, wPSNR assigns higher penalties to distortions occurring in perceptually important regions, thereby providing a more accurate and perceptually relevant evaluation of imperceptibility:

$$wPSNR = 10\ log_{10} \frac{255^2}{wMSE} \tag{12}$$

$$wMSE = \frac{\sum_{h=1}^{H} \sum_{w=1}^{W} (X(w,h) - Y(w,h))^2 \times n(X(w,h))^2}{W \times H} \tag{13}$$

The weighting factor n ($C\,(w, h)$), defined as the noise visibility function (NVF) [37], is given by:

$$n\big(C(w,h)\big) = \frac{1}{1+\sigma_{w,h}^2} \tag{14}$$

Table 7 presents the wPSNR values of the EPIT-LSBMR-X-Gray technique across different embedding capacities, block sizes, and neighboring concepts. Among the tested configurations, the 32×32 block size combined with the 8-neighbor concept yielded slightly better performance than the 8×8 and 16×16 block sizes. This marginal improvement can be attributed to the 8-neighbor concept's enhanced ability to capture block complexity by incorporating information from all surrounding directions, thereby providing a more comprehensive assessment of pixel similarity. The wPSNR values vary noticeably across individual test images due to differences in structural and textural characteristics. For instance, the Baboon image—known for its high textural richness—achieved a wPSNR of 94.21 dB for a 2 KB payload with a 32×32 block size, indicating minimal perceptual distortion introduced by embedding. In contrast, the Peppers image, which contains smoother and more uniform regions, produced a lower wPSNR of 85.51 dB under the same configuration, reflecting its greater sensitivity to modifications in low-variance regions.

Table 8 reports the wPSNR values obtained using the EPIT-LSBMR-X technique. Across all configurations, the wPSNR values remain consistently high, ranging from approximately 68 dB to over 102.6 dB at lower embedding capacities, indicating strong preservation of visual quality. The 8-neighbor concept with 32×32 block sizes yields slightly higher wPSNR values, benefiting from more comprehensive pixel similarity analysis in all directions and improved identification of regions suitable for data embedding. When applied to larger block sizes, this approach further enables the algorithm to more accurately detect textured regions where embedding can be performed with minimal perceptual distortion. Performance also varies with image complexity. Textured images, such as Baboon, consistently exhibit higher wPSNR values due to their greater ability to conceal embedding noise, whereas simpler images yield comparatively lower values. Overall, these results demonstrate the robustness of the proposed method in maintaining high image quality under varying conditions.

5.2.3. SSIM

SSIM is a widely used performance metric for evaluating the similarity between two images. It measures the perceptual differences between the cover and stego images. SSIM is computed as a combination of three components—luminance, contrast, and structure—as shown in Eq. (13) [38]. A higher SSIM value indicates better embedding quality.

$$SSIM\ (x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x{}^2 + \mu_y{}^2 + C_1)(\sigma_x{}^2 + \sigma_y{}^2 + C_2)} \tag{15}$$

Here, μ_x, μ_y denote the mean intensity values of the cover and stego images (x and y), respectively. The terms $\sigma_x{}^2$ and $\sigma_y{}^2$ represent the variances of x and y, while σ_{xy} denotes their covariance. Constants C_1 and C_2 are included to prevent numerical instability caused by division by zero.

The SSIM values for the EPIT-LSBMR-X-Gray technique are presented in Table 7. Regardless of the number of embedding bits, the SSIM values remained high, indicating superior stego-image quality. All parameters yielded practically identical results, with the exception of the 8×8 block size combined with the 8-neighbor concept, which demonstrated a slight advantage. Additionally, the SSIM values for the EPIT-LSBMR-X technique, presented in Table 8, remain high, particularly when 16×16 and 32×32 block sizes are used. These high SSIM values indicate that the structural integrity, luminance, and contrast of the image are minimally affected by the concealment of a secret message.

5.2.4 Visual Analysis of Embedding Effects

In this section, we provide visual analyses of the steganographic effects using cover–stego difference maps and pixel-difference histograms for payloads of 1 KB and 8 KB. As shown in Fig. 10, the cover and stego images appear visually indistinguishable, confirming their high perceptual quality. The corresponding binary difference maps reveal scattered and localized alterations: the 1 KB payload produces only minimal changes, whereas the 8 KB payload introduces slightly more concentrated modifications around textured regions. The pixel-difference histograms further corroborate these observations, exhibiting a distinct peak at zero difference, with the majority of nonzero differences confined to the ±1 intensity level. Together, these visual results complement the quantitative metrics (PSNR and SSIM), demonstrating that the proposed embedding strategy maintains high visual quality and distributes distortions uniformly across the image.

5.2.5. Comparative Evaluation with State-of-the-Art Methods

Table 9 presents a comparative analysis of the imperceptibility performance of the proposed EPIT method against two related techniques [25, 26], evaluated across different embedding capacities and standard

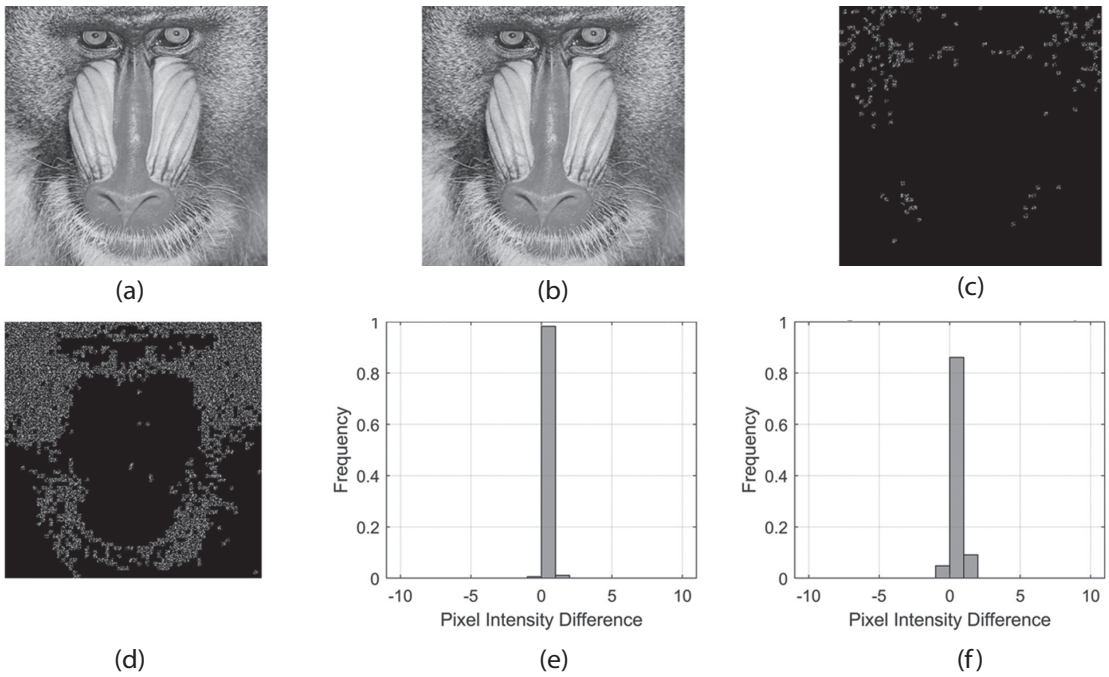


Fig. 10. Visual analysis of steganographic perturbations for payloads of 1 KB and 8 KB. Subplots (**a**)–(**b**) show the cover and stego images. Subplots (**c**) and (**d**) present the binary difference maps. Subplots (**e**) and (**f**) display the pixel-difference histograms.
(**a**) Cover Image, (**b**) Stego Image (1 KB), (**c**) Binary Difference Map (1 KB), (**d**) Bynary Difference Map (8 KB), (**e**) Histogram of Pixel Differences (1 KB), (**f**) Histogram of Pixel Differences (8 KB)

test images. EPIT serves as the baseline approach, embedding data sequentially without the use of LSBMR or complex block selection mechanisms, similar to the techniques to which it is compared. Overall, EPIT consistently outperforms the other methods in terms of PSNR, SSIM, and wPSNR metrics. It achieves higher PSNR values across all test images and embedding capacities, indicating reduced pixel-level distortion and improved imperceptibility. The SSIM values remain consistently close to 1, reflecting strong structural preservation and high visual similarity to the original cover images. Although method [25] attains slightly higher SSIM values in certain cases, potentially due to its strategy of preserving more of the indicator channel's original values, EPIT achieves a stronger balance across all evaluation metrics (PSNR, wPSNR, and SSIM). Furthermore, EPIT demonstrates a clear advantage in terms of wPSNR, despite its sequential embedding strategy. The consistently higher wPSNR values across all embedding capacities indicate that EPIT introduces less perceptible distortion, thereby maintaining superior perceptual quality, even in regions where the human visual system is more sensitive to changes.

Table 10 demonstrates the imperceptibility results of the proposed EPIT-LSBMR-X-Gray method, compared with two related techniques reported in [8][9], at an embedding rate of 0.3 bpp, which corresponds to (9.6×1024×8) / (512×512). Our technique achieved PSNR values of 55.65 for the Baboon image and 55.61 for the Barbara image. In contrast, the methods in [8, 9] yielded lower PSNR values of 55.39 and 53.56, and 50.34 and 50.65, respectively. For the remaining test images, the method in [8] achieves higher PSNR values than both the proposed technique and [9], primarily due to its multi-bit embedding algorithm, which introduces fewer pixel modifications compared to the LSB-based approaches used in our method. However, the proposed technique exhibits superior performance in terms of wPSNR for nearly all images when compared with [8][9]. This suggests that the proposed technique more accurately identifies the appropriate complex regions in the images that achieve high imperceptibility, surpassing the existing techniques.

Fig. 11 presents a comparative analysis of the imperceptibility achieved by the proposed EPIT-LSBMR-X-Gray technique and the edge-based methods reported in [14] and [15] . In this experiment, [14] employs graded fuzzy edge detection to improve upon a prior approach that used dilation combined with hybrid edge detection applied to the three MSBs of the cover images [16]. Across all payload sizes (1 KB to 8 KB), the proposed method consistently achieves superior imperceptibility. For example, at a payload of 1 KB, the Baboon image attains a PSNR of 65.43, a wPSNR of 97.38, and an SSIM of 1.0, outperforming the results of [14] (58.13, 90.16, 1.0, respectively) and [15] (63.84, 95.87, 1.0, respectively). Even at higher payloads, such as 8 KB, the proposed method maintains higher visual quality; for instance, the Boat image yields a PSNR of 56.33 and a wPSNR of 81.69, compared to [14] (50.3, 77.46, respectively) and [15] (55.95, 74.98, respectively). These results demonstrate the proposed method's ability to preserve image quality without relying on edge-based information.

Table 9. Comparison between the proposed EPIT technique with the techniques [25] and [26].

EC	Image	PSNR			wPSNR			SSIM		
		[26]	[25]	EPIT	[26]	[25]	EPIT	[26]	[25]	EPIT
1KB	Baboon	63.6	67.37	68.56	88.05	88.57	91.62	0.99997	0.9999	0.99999
	Barbara	61.53	67.29	68.49	69.89	79.92	80.44	0.9999	0.9999	0.99997
	Boat	60.25	67.34	68.46	64.81	75.76	77.64	0.9995	0.9999	0.99998
	Peppers	59.03	67.23	68.51	74.66	95.96	84.21	0.9998	1	0.99998
4KB	Baboon	57.8	61.3	62.54	81.51	79.89	81.69	0.99998	0.9999	0.9999
	Barbara	55.32	61.36	62.51	63.54	71.84	72.81	0.9995	0.9999	0.99997
	Boat	55.66	61.26	62.53	61.28	67.5	69.52	0.9987	0.9998	0.9998
	Peppers	53.01	61.28	62.55	65.7	78.08	75.03	0.9989	0.9999	0.99997
16KB	Baboon	51.36	55.24	56.50	70.91	73.66	75.33	0.9997	0.9999	0.9999
	Barbara	48.93	55.25	56.51	57.37	64.63	66.04	0.9976	0.9998	0.9998
	Boat	51.52	55.26	56.50	58.56	61.69	63.98	0.9973	0.9989	0.9991
	Peppers	47.25	55.21	56.49	59.28	69.38	68.11	0.9951	0.9999	0.9999
32KB	Baboon	48.1	52.25	53.50	64.06	68.15	70.66	0.9991	0.9997	0.9998
	Barbara	45.87	52.27	53.49	54.64	62.17	63.64	0.9951	0.9996	0.9996
	Boats	-	52.27	53.49	-	60.06	62.65	-	0.9981	0.9984
	Peppers	44.1	52.19	53.47	56.09	66.8	65.47	0.9895	0.9999	0.9999

Table 10. PSNR, wPSNR, and SSIM comparison of the proposed EPIT-LSBMR-X-Gray technique with [8] and [9] at embedding capacity of 0.3 bpp

Image	PSNR			wPSNR			SSIM		
	[9]	[8]	Proposed	[9]	[8]	Proposed	[9]	[8]	Proposed
Baboon	50.34	55.39	55.65	80.86	81.93	86.60	0.9997	0.9999	0.9999
Plane	50.60	57.56	55.66	78.34	67.55	73.05	0.9996	0.9999	0.9993
Boat	50.64	56.45	55.59	78.21	79.21	80.66	0.9995	0.9999	0.9998
Barbara	50.65	53.56	55.61	79.80	76.38	85.23	0.9997	0.9999	0.9996
Peppers	50.95	57.11	56.57	75.44	71.38	81.40	0.9993	0.9999	0.9996
Couple	51.07	56.28	55.58	77.88	80.86	82.86	0.9996	0.9999	0.9995
Average	50.71	56.05	55.78	78.42	76.22	81.63	0.9995	0.9999	0.9996

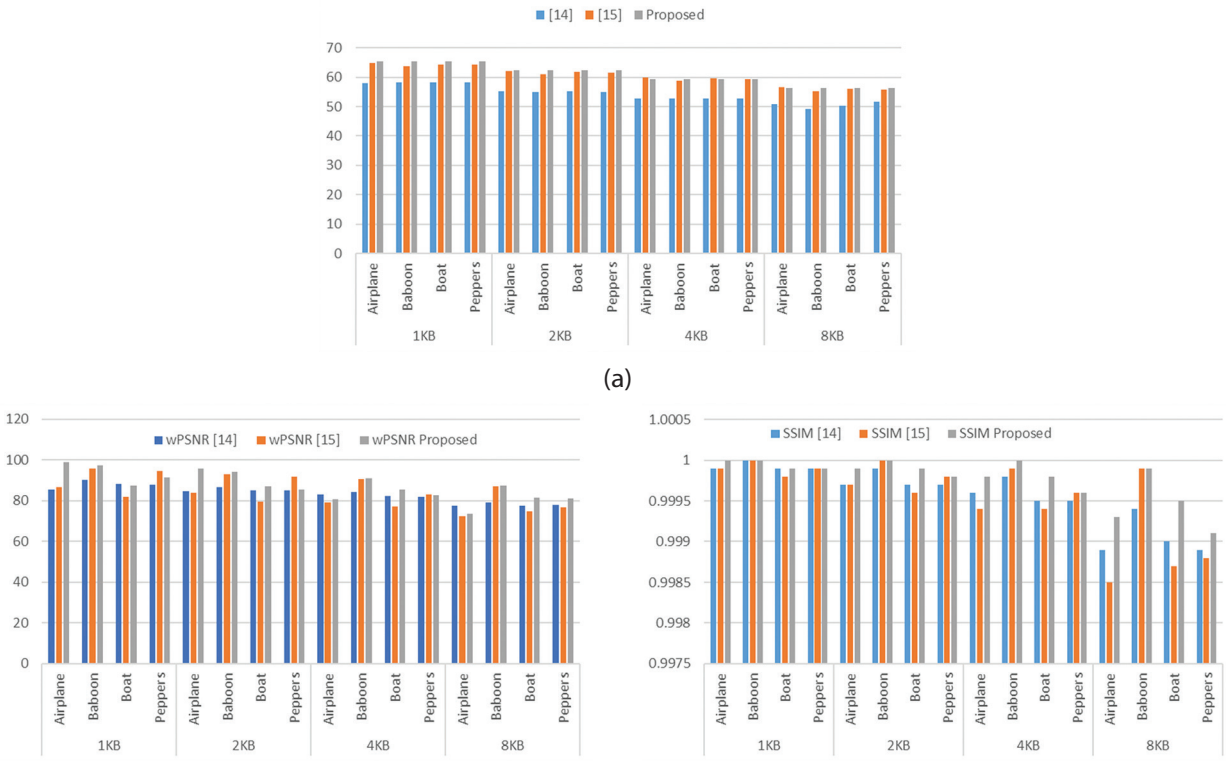


Fig. 11. PSNR, wPSNR, and SSIM comparisons between the proposed EPIT-LSBMR-X-Gray method and the edge-based approaches in [14] and [15] across payloads from 1 KB to 8 KB. : (**a**) PSNR, (**b**) wPSNR, and (**c**) SSIM

Fig. 12 illustrates the PSNR performance of the proposed EPIT-LSBMR-X method compared to three existing techniques from [16, 17], and [13], using the Cat, Lion, and Tiger images (each 256×256 pixels). Across all tested embedding capacities (1024, 4096, 8192, and 16,384 bits), EPIT-LSBMR-X consistently achieved higher PSNR values than the other methods.

This consistent performance demonstrates the method's strength in preserving image quality and reducing pixel-level distortion during data embedding.

These results confirm the improved imperceptibility of EPIT-LSBMR-X, highlighting its suitability for steganographic applications where visual quality is paramount.

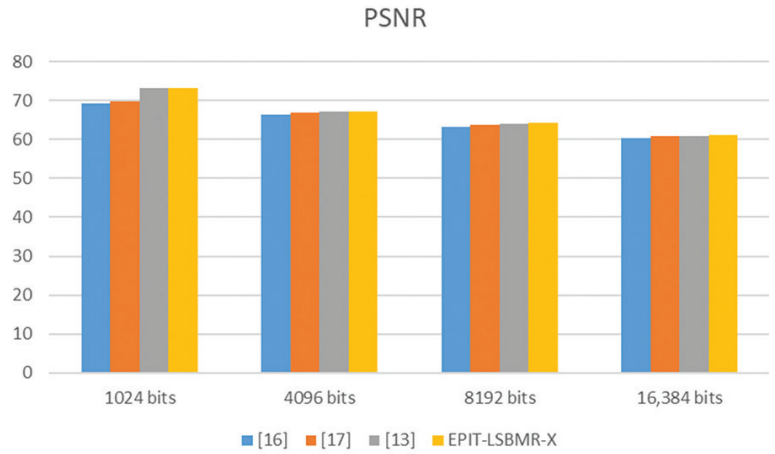


Fig. 12. PSNR comparison: proposed EPIT-LSBMR-X vs. [16], [17], and [13]

Table 11 presents the imperceptibility results of the proposed EPIT-LSBMR-X technique compared with the method in [18]. Across all payloads and images, both approaches yield comparable SSIM values and very similar PSNR, with [18] showing a marginal advantage in PSNR. This confirms that both methods maintain high imperceptibility and strong structural preservation. However, the proposed method consistently achieves higher wPSNR, particularly on textured images such as Baboon and Barbara, reflecting improved perceptual quality in visually sensitive regions. Overall, the results demonstrate that the proposed technique offers superior perceptual imperceptibility while matching [18] in conventional PSNR and SSIM performance.

Table 11. Comparison of imperceptibility metrics (PSNR, wPSNR, SSIM) between the proposed EPIT-LSBMR-X technique and the method in [18] across multiple payloads and images

Capacity	Image	PSNR		wPSNR		SSIM	
		[18]	proposed	[18]	proposed	[18]	proposed
1Kbits	Airplane	79.79	79.57	101.60	108.34	0.999999	0.999999
	Baboon	80.01	79.52	102.74	107.39	0.999999	0.999999
	Barbara	79.91	79.54	101.16	105.59	0.999999	0.999999
	Boats	79.80	79.66	106.03	108.72	0.999999	0.999999
	Peppers	79.79	79.66	95.24	92.14	0.999999	0.999999
2Kbits	Airplane	72.86	72.48	94.07	102.95	0.999997	0.999997
	Baboon	72.87	72.51	96.15	100.57	0.999998	0.999998
	Barbara	72.92	72.48	94.14	98.22	0.999998	0.999998
	Boats	72.99	72.58	99.58	94.59	0.999998	0.999998
	Peppers	72.93	72.47	88.75	86.44	0.999999	0.999999
1KB	Airplane	70.98	70.59	91.93	101.52	0.999996	0.999996
	Baboon	70.88	70.64	94.29	98.77	0.999998	0.999998
	Barbara	70.92	70.59	91.89	95.84	0.999997	0.999997
	Boats	71.03	70.63	97.12	94.76	0.999998	0.999997
	Peppers	70.93	70.57	86.79	84.48	0.999999	0.999998
2KB	Airplane	-	67.56	-	94.23	-	0.999991
	Baboon	67.90	67.54	91.32	95.29	0.999996	0.999995
	Barbara	68.04	67.53	89.16	92.91	0.999995	0.999994
	Boats	67.96	67.55	95.07	88.92	0.999995	0.999987
	Peppers	-	67.55	-	82.32	-	0.999997

5.3. ABLATION STUDY

To assess the contribution of each component of the proposed framework, we conducted an ablation experiment on benchmark images with payloads of 1 KB, 4 KB, and 8 KB, comparing LSBMR only, EPIT only, and the proposed EPIT-LSBMR-X. As summarized in Table 12, LSBMR only maintains high PSNR and SSIM values, indicating strong imperceptibility, but its wPSNR values decrease with an increase in the payload. The EPIT only shows lower PSNR compared to LSBMR but maintains comparable SSIM and wPSNR performance.

The proposed EPIT-LSBMR-X achieves a balanced trade-off; it enhances EPIT performance by inheriting high imperceptibility from LSBMR while improving robustness through a block-selection strategy. Notably, the combined model consistently yields higher wPSNR values than either module alone, confirming that the integration enhances perceptual quality.

Table 12. Ablation study comparing LSBMR only, EPIT only, and the combined EPIT-LSBMR at payloads of 1 KB, 4 KB, and 8 KB.

Image	Method	PSNR			wPSNR			SSIM		
		1KB	4KB	8KB	1KB	4KB	8KB	1KB	4KB	8KB
Airplane	LSBMR	66.14	60.18	57.15	94.55	74.64	73.21	0.9999	0.99958	0.99918
	EPIT-LSBMR	65.32	59.33	56.34	98.87	77.43	73.73	0.999981	0.99977	0.99931
	EPIT	63.73	57.75	54.72	92.09	72.20	70.43	0.999785	0.99908	0.99817
Baboon	LSBMR	66.22	60.13	57.15	89.24	86.97	83.91	0.999962	0.99991	0.99976
	EPIT-LSBMR	65.41	59.33	56.34	97.26	91.02	87.49	0.999993	0.99997	0.99992
	EPIT	63.77	57.75	54.73	86.85	84.26	81.77	0.999916	0.99980	0.99951
Barbara	LSBMR	66.25	60.19	57.18	88.08	82.04	78.36	0.99993	0.99967	0.99933
	EPIT-LSBMR	65.35	59.31	56.33	97.16	90.05	85.63	0.999992	0.99994	0.99975
	EPIT	63.75	57.76	54.74	85.63	80.26	75.41	0.999846	0.99935	0.99842
Boat	LSBMR	66.20	60.19	57.15	85.46	80.26	78.49	0.999916	0.99969	0.99943
	EPIT-LSBMR	65.41	59.35	56.34	91.18	84.58	81.56	0.999967	0.99979	0.99951
	EPIT	63.59	57.73	54.67	82.88	77.64	75.73	0.999811	0.99932	0.99873
Peppers	LSBMR	66.15	60.17	57.15	91.28	80.82	77.82	0.999914	0.99963	0.99923
	EPIT-LSBMR	65.40	59.37	56.34	88.61	85.01	81.02	0.999903	0.99964	0.99919
	EPIT	63.75	57.76	54.75	89.02	79.51	75.57	0.999821	0.99926	0.99833

5.4. SECURITY ANALYSIS

Standard attacks were conducted on the test images for steganalysis assessment to evaluate the proposed technique's security performance. The Chi-square [39], RS steganalysis [40], Spatial Rich Model (SRM) + Ensemble Classifier [41], and XuNet CNN-based steganalyzer [42] were applied to stego images containing varying payload sizes. The Java-based steganalysis tool StegExpose [43] was used to perform the Chi-square and RS steganalysis attacks, with default settings applied to generate the resulting test reports. StegExpose is specifically designed to identify steganography in lossless images such as BMPs.

5.4.1. Chi-Square Steganalysis

The technique proposed by [39] detects the presence of hidden data in digital images based on the statistical analysis of pixel value distributions. It compares the distribution of pixel values in a cover image with that of a stego image. A significant difference between the two distributions suggests the presence of hidden data. The Chi-squared statistic, denoted as χ^2, is calculated using the following formula:

$$\chi^2 = \sum_{i=1}^{v+1} \frac{(y_i - y_i^*)^2}{y_i^*} \quad (16)$$

Here, v is the number of degrees of freedom, y_i is the observed frequency of pixel values in the stego image, and y_i^* is the expected frequency of pixel values in the cover image. A higher χ^2 value indicates a greater difference between the two distributions and a greater probability of an image containing hidden data. Table 13 displays the results of the tests conducted using the StegExpose tool for the EPIT-LSBMR-X-Gray technique, tested with secret messages of 1 KB and 4 KB for selected images with various block sizes. The steganalysis results for RGB stego images using the EPIT-LSBMR-X technique, tested with 1 KB, 4 KB, and 8 KB embedding capacities, are shown in Table 14. The Chi-square values suggest that the embedding probability is nearly zero for most images, further highlighting the largely unaltered nature of the statistical characteristics of the cover images after embedding. Evidently, Chi-square-based attacks consistently fail to detect the presence of hidden data embedded using the proposed method. Consequently, the EPIT-LSBMR-X and EPIT-LSBMR-X-Gray techniques demonstrate strong resistance to Chi-square-based steganalysis. Additionally, texture-rich images offer greater robustness against detection, whereas using different block sizes or neighboring concepts does not provide any significant security advantage.

5.4.2. RS Steganalysis

The technique proposed in [40] is an effective method for detecting hidden messages in images and estimating their size. It exploits spatial domain correlations by analyzing bit-plane relationships. A discrimination function divides the image into regular (R) and singular (S) groups, wherein the relative frequency of each group changes according to the length of the embedded data.

The flipping functions use a mask M, which represents the function applied to each element of a group. The number of regular groups should exceed that of singular groups. If the relation $R_M \approx R_{-M} > S_M \approx S_{-M}$ holds, it indicates no hidden data in the image. Conversely, an image contains hidden data if R_{-M} and S_M increase while R_M and S_{-M} decrease [44].

Tables 13 and 14 show the average differences between R_M and R_{-M}, as well as S_M and S_{-M}, for different embedding capacities, block sizes, and neighboring strategies used in the proposed techniques. The consistently small differences indicate that the proposed technique is highly resistant to RS steganalysis. This strength arises from utilizing encoded bit patterns through the EPIT structure and the LSBMR algorithm rather than traditional bit-flipping methods [44]. Additionally, embedding is limited to the most complex image blocks, which helps avoid detection by excluding predictable or visually sensitive regions. Among the tested configurations, the 8-neighbor approach generally offers better protection than the 4-neighbor one. For grayscale images, a 32×32 block size performs slightly better, while for the RGB technique, smaller blocks such as 16×16 or 8×8 yield enhanced results.

Table 13. StegExpose results for the EPIT-LSBMR-X-Gray technique across selected stego images.

Neighboring Concept	EC	Image	Chi-Square			RS analysis		
			8x8	16x16	32x32	8x8	16x16	32x32
8N	1KB	Couple	0.0014	0.0014	0.0014	0.0134	0.0165	0.0174
		House	0.0001	0.0001	0.0001	0.0046	0.0038	0.0095
		Peppers	0.0028	0.0037	0.0028	0.0596	0.0583	0.0571
	4KB	Couple	0.0014	0.0018	0.0014	0.0124	0.0207	0.0134
		House	0.0001	0.0001	0.0001	0.0066	0.0142	0.0021
		Peppers	0.0025	0.0033	0.0029	0.0629	0.0566	0.0617
4N	1KB	Couple	0.0014	0.0014	0.0014	0.0142	0.0136	0.0174
		House	0.0001	0.0001	0.0001	0.0056	0.0033	0.0095
		Peppers	0.0026	0.0029	0.0028	0.0626	0.0635	0.0571
	4KB	Couple	0.0015	0.0017	0.0014	0.0103	0.0139	0.0089
		House	0.0001	0.0001	0.0001	0.0138	0.0117	0.0054
		Peppers	0.0029	0.0033	0.0031	0.0604	0.0632	0.0634

Table 14. StegExpose results for the EPIT-LSBMR-X technique across selected stego images

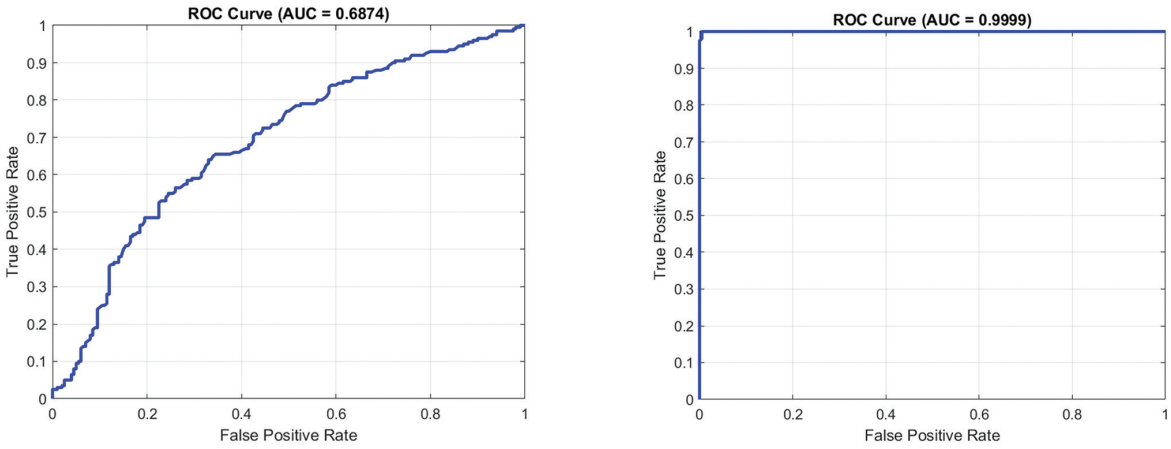
Neighboring Concept	EC	Image	Chi-Square			RS analysis		
			8x8	16x16	32x32	8x8	16x16	32x32
8N	1KB	Baboon	0.0002	0.0002	0.0002	0.0194	0.0179	0.0219
		Boat	0.0011	0.0011	0.0012	0.0809	0.0859	0.0851
		Peppers	0.0013	0.0013	0.0013	0.0758	0.0757	0.0784
	8KB	Baboon	0.0004	0.0004	0.0002	0.0326	0.0436	0.0456
		Boat	0.0016	0.0019	0.0030	0.0849	0.0811	0.0745
		Peppers	0.0021	0.0015	0.0026	0.0821	0.0808	0.0849
4N	1KB	Baboon	0.0002	0.0003	0.0002	0.0202	0.0182	0.0181
		Boat	0.0012	0.0013	0.0014	0.0783	0.0806	0.0817
		Peppers	0.0013	0.0013	0.0013	0.0763	0.0776	0.0772
	8KB	Baboon	0.0011	0.0005	0.0003	0.0249	0.0331	0.0401
		Boat	0.0018	0.0024	0.0041	0.0659	0.0779	0.0705
		Peppers	0.0016	0.0016	0.0019	0.0791	0.0796	0.0843

5.4.3. Security Against Ensemble Classifier-Based Steganalysis

To evaluate the proposed scheme's security for different images, an ensemble classifier is utilized [41]. This method combines the predictions of multiple base learners, improving overall accuracy by reducing the impact of individual base learner errors. In this experiment, a feature-based steganalysis algorithm, the ensemble classifier [11], is employed to assess the security performance of the proposed approach. The SRM extracts a high-dimensional set of noise-residual features (34,671 features) designed to capture weak embedding artifacts in spatial-domain images [41]. The ensemble classifier offers significantly lower training complexity, enabling effective steganalysis with high-dimensional cover models and larger training sets.

For this experiment, a balanced dataset of cover and stego grayscale images was constructed using pay-

loads of 1 KB and 8 KB with an 8×8 block size and the 8-neighbor configuration. SRM features were extracted and evaluated using a standard 70/30 train-test split, with a total of 1000 images. As illustrated in the ROC curves in Fig. 13, the proposed method shows moderate resistance at low payloads but becomes more detectable at higher capacities. At 1 KB, the classifier achieves 65.5% accuracy and an AUC of 0.6874, indicating that the stego images remain relatively difficult to differentiate from natural covers. However, at 8 KB, detectability increases sharply, reaching 99.5% accuracy and an AUC of 0.9999, demonstrating low resistance at high payloads due to the stronger artifacts introduced by the embedding process.



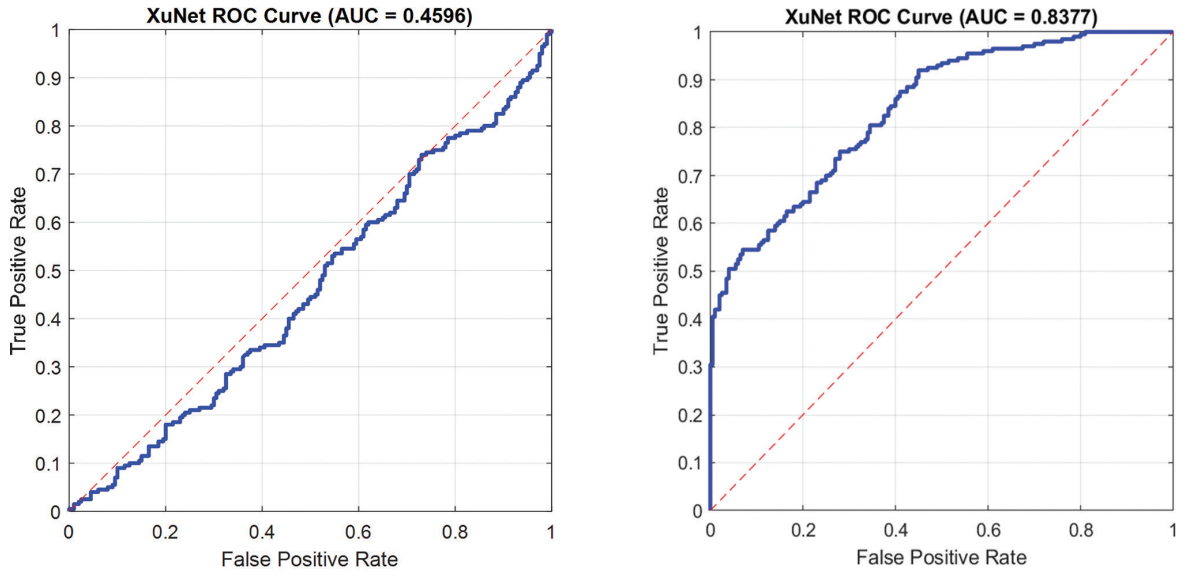
(a) (b)

Fig. 13. ROC curves obtained using SRM + Ensemble Classifier steganalysis of 1 KB (**a**) and 8 KB (**b**)

5.4.4. Security Against Convolutional Neural Network-Based Steganalysis (XuNet)

To further evaluate the proposed embedding framework's security, XuNet, a convolutional neural network (CNN)-based steganalysis approach [42], was employed. In this experiment, an 8×8 block size and the 8-neighbor configuration were employed to embed 1,000 images from the BOSS 1.01 dataset with payloads of 1 KB and 8 KB. The dataset was balanced between cover and stego images and divided using a 70/30 train-test split to ensure fair performance evaluation.

Fig. 14 shows that the proposed method exhibits a moderate level of robustness with low embedding payload. At a payload of 1 KB, XuNet achieves a detection accuracy of 48.5%, reflecting a limited ability to distinguish stego images from cover images, making accurate classification relatively difficult. However, with an increase in the embedding payload, steganalysis performance improves gradually. With a payload of 8 KB, XuNet reaches a detection accuracy of 73%, demonstrating a moderate ability to distinguish between cover and stego images, corresponding to more statistical distortions introduced by higher embedding rates.



(a) (b)

Fig. 14. ROC curves of XuNet steganalysis for payloads of 1 KB (**a**) and 8 KB (**b**)

5.5. ROBUSTNESS AND DATA RECOVERY ANALYSIS

To assess the reliability of the proposed EPIT-LSBMR technique under high payload and distortion conditions, a robustness evaluation was conducted using common image manipulations, including JPEG compression, noise addition, cropping, and rotation, at payloads of 1 KB and 4 KB. For each attack, visual quality was evaluated using PSNR and SSIM computed between the original stego image and the attacked stego image, while data recovery performance was measured using the bit-error rate (BER) between the original embedded message and the extracted message.

As shown in Table 15, the BER results indicate that while the method maintains good visual quality after moderate compression and noise (PSNR ≈ 39–35 dB, SSIM > 0.9), data recovery is highly sensitive to distortions, with BER values close to 50% under JPEG 90%, JPEG 70%, Gaussian noise with variance 0.001, and 5% salt-and-pepper noise.

Geometric attacks such as cropping, applied by removing approximately 10% of the image area from the borders while preserving the central region, and rotation, limited to right-angle rotations (90°), result in a noticeable degradation of visual quality, as reflected by lower PSNR and SSIM values. In these cases, the BER also remains close to 50% for both payload sizes, indicating a failure to reliably recover the embedded data. These results highlight the limitations of pixel-domain embedding against such attacks.

Table 15. Average BER, PSNR, and SSIM values under JPEG compression, Gaussian noise, salt and pepper noise, cropping, and rotation for 1 KB and 4 KB payloads

Attack Type	1KB			4KB		
	BER	PSNR	SSIM	BER	PSNR	SSIM
JPEG (90%)	49.98%	39.7	0.967	49.88%	39.70	0.967
JPEG (70%)	50.00%	35.3	0.927	49.90%	35.31	0.930
Gaussian (0.01)	49.99%	30.01	0.755	50.27%	30.01	0.754
Salt & Pepper (0.05)	49.89%	18.4	0.398	49.84%	18.34	0.397
Crop	49.98%	12.86	0.428	49.55%	13.09	0.242
Rotate	49.06%	11.83	0.393	49.89%	11.73	0.208

5.6. IMPACT OF IMAGE COMPLEXITY ON EMBEDDING CAPACITY AND IMPERCEPTIBILITY

The proposed technique begins the embedding process by selecting the most complex image blocks, which are capable of concealing secret data while preserving visual quality and resisting structural steganalysis. Since image complexity varies significantly, the embedding capacity and imperceptibility depend on the image's characteristics. High wPSNR values and increased complexity thresholds indicate that highly complex images, such as Baboon, support larger payloads with minimal perceptual distortion.

This is reflected in their higher embedding capacities as shown in Table 16, reaching up to ~11.8 KB for 8×8 blocks using 4N and 8N. Such images also exhibit a low number of zero-complexity blocks, suggesting a rich presence of textured regions suitable for secure embedding.

In contrast, images such as Peppers and Airplane demonstrate lower average complexity thresholds and a higher count of zero-complexity blocks. These results indicate that smooth, uniform regions are more common than textured areas in these images, making them less suitable for data embedding without compromising imperceptibility. Consequently, these images limit the availability of safe embedding regions, reducing capacities to approximately 8.3–8.8 KB. The modest variations in payload are primarily driven by texture richness. Table 16 highlights that images with higher complexity thresholds, a greater number of blocks exceeding the threshold, and fewer zero-complexity blocks are more suitable for steganographic applications. Such characteristics ensure that the embedding process remains both imperceptible and resistant to detection.

Table 16. Image complexity information using 8x8 Block size

Image	Threshold = Complexity Avg	Number of blocks >= Threshold	Max block complexity	Number of blocks with zero complexity	8x8-4N	8x8-8N
Baboon	0.3060	2127	0.7048	368	11.617KB	11.824KB
Boat	0.1802	1914	0.6571	1220	9.835KB	9.619KB
Peppers	0.1430	1823	0.5571	1228	8.625KB	8.763KB
Airplane	0.1614	1835	0.6905	1319	8.340KB	8.348KB
Barbara	0.2617	1797	0.7952	0806	8.726KB	8.855KB

4.7. COMPLEXITY AND RUNTIME ANALYSIS

The proposed EPIT-LSBMR algorithm operates on an $N \times N$ image divided into blocks of size $B \times B$. Since the image contains $(N/B)^2$ blocks and each block has B^2 pixels, the total work is $(N/B)^2 \times B^2 = N^2$ yielding an overall complexity of $O(N^2)$, i.e., linear in the number of pixels. Table 17 presents the average runtime in seconds across five payloads (1–10 KB). EPIT-LSBMR requires 0.047–0.089 per image, compared with 0.008–0.057 for EPIT and 0.003–0.007 for LSBMR. The slightly higher cost of EPIT-LSBMR arises from block-complexity evaluation and adaptive embedding.

Table 17. Average per image runtime (in seconds) of EPIT LSBMR compared with EPIT and LSBMR across different payloads

Payloads	1KB	2KB	4KB	8KB	10KB
EPIT	0.0081	0.0151	0.027	0.047	0.0573
EPIT-LSBMR	0.047	0.059	0.0639	0.079	0.089
LSBMR	0.0032	0.0028	0.0038	0.007	0.0073

6. CONCLUSION

This study presents a secure, region-based image steganography technique integrating the EPIT with LSBMR, selectively applied to high-complexity blocks. By incorporating human visual system sensitivity, the method avoids embedding in low-variation (smooth) regions, thereby preserving imperceptibility. To balance embedding capacity and visual quality, only textured blocks that ensure the complexity threshold are used, While the threshold can be lowered for larger payloads, blocks with near-zero complexity are excluded to maintain imperceptibility.

While EPIT utilizes an indicator channel to control embedding the bit type and repetition count within the data channels, LSBMR minimizes pixel modifications, further enhancing security. Ultimately, this combination improves both imperceptibility and resistance to detection.

At 0.3 bpp, the proposed EPIT-LSBMR-X-Gray technique achieves PSNR values between 55.58 dB and 56.57 dB and SSIM scores between 0.9993 and 0.9999, suggesting excellent visual quality. The wPSNR results range from 73.05 dB to 86.60 dB, outperforming related methods in preserving perceptual quality. The combination of a larger block size (32×32) and the 8-neighbor concept enhances visual quality and efficiency, thus minimizing embedding artifacts and reducing computational overhead. Additionally, robustness is validated through resistance to Chi-square, RS, ensemble classifier, XuNet, and steganalysis attacks. Future research should focus on optimizing the indicator mechanism and extending the approach to video and audio steganography.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support provided by Universiti Kebangsaan Malaysia (UKM) for this work.

REFERENCES

[1] A. M. Fadhil, H. N. Jalo, O. F. Mohammad, "Improved Security of a Deep Learning-Based Steganography System with Imperceptibility Preservation", International Journal of Electrical and Computer Engineering Systems, Vol. 14, No. 1, 2023, pp. 73-81. doi: https://doi.org/10.32985/ijeces.14.1.8.

[2] S. Kamil, M. Ayob, S. N. H. S. Abdullah, M. Zulkifli Ahmad, "Lightweight and Optimized Multi-Layer Data Hiding using Video Steganography", International Journal of Advanced Computer Science and Applications, Vol. 9, No. 12, 2018. doi: 10.14569/IJACSA.2018.091237.

[3] A. H. Ali, L. E. George, A. A. Zaidan, M. R. Mokhtar, "High capacity, transparent and secure audio steganography model based on fractal coding and chaotic map in temporal domain", Multimedia Tools and Applications, Vol. 77, No. 23, 2018, pp. 31487-31516. doi: 10.1007/s11042-018-6213-0.

[4] O. F. A. Adeeb, S. J. Kabudian, "Arabic Text Steganography Based on Deep Learning Methods", IEEE Access, Vol. 10, 2022, pp. 94403-94416. doi: 10.1109/ACCESS.2022.3201019.

[5] D. R. I. M. Setiadi, S. Rustad, P. N. Andono, G. F. Shidik, "Digital Image Steganography Survey and Investigation (Goal, Assessment, Method, Development, and Dataset)", Signal Processing, Vol. 206, 2023. doi: 10.1016/j.sigpro.2022.108908.

[6] S. Ghoul, R. Sulaiman, Z. Shukur, K. Malaysia, U. K. M. Bangi, "A Review on Security Techniques in Image Steganography", International Journal of Advanced Computer Science and Applications, Vol. 14, No. 6, 2023. doi: 10.14569/IJACSA.2023.0140640.

[7] M. A. Hameed, M. Hassaballah, S. Aly, A. I. Awad, "An adaptive image steganography method based on histogram of oriented gradient and PVD-LSB techniques", IEEE Access, Vol. 7, 2019, pp. 185189-185204. doi: 10.1109/ACCESS.2019.2960254.

[8] A. Saeed et al. "An accurate texture complexity estimation for quality-enhanced and secure image steganography", IEEE Access, Vol. 8, pp. 21613-21630, 2020. doi: 10.1109/ACCESS.2020.2968217.

[9] D. Laishram, T. Tuithung, "A novel minimal distortion-based edge adaptive image steganography scheme using local complexity: (BEASS)", Multime-

dia Tools and Applications, Vol. 80, No. 1, 2021, pp. 831-854. doi: 10.1007/s11042-020-09519-9.

[10] G. Paul, I. Davidson, I. Mukherjee, S. S. Ravi, "Keyless dynamic optimal multi-bit image steganography using energetic pixels", Multimedia Tools and Applications, Vol. 76, No. 5, 2017, pp. 7445-7471. doi: 10.1007/s11042-016-3319-0.

[11] N. D. Tuan, "A secure image steganography based on Hamming codes and image block complexity estimation using a zig-zag order", Journal of Science and Technology on Information security, Vol. 2, No. 25, 2025, pp. 19-40. doi: 10.54654/isj.v2i25.1123.

[12] R. Jamatia, B. Bhuyan, "A Contrast-Channel Embedding Approach for Image Steganography: Balancing Capacity and Imperceptibility", IEEE Access, Vol. 13, 2025, pp. 1180539-180565. doi: 10.1109/access.2025.3618875.

[13] S. Bilgaiyan, R. Ahmad, S. Sagnika, "Adaptive Image Steganography Using Rotating Color Channels and Inverted LSB Substitution", SN Computer Science, Vol. 4, No. 5, 2023. doi: 10.1007/s42979-023-01949-0.

[14] D. R. I. M. Setiadi, S. Rustad, P. N. Andono, G. F. Shidik, "Graded fuzzy edge detection for imperceptibility optimization of image steganography", Imaging Science Journal, Vol. 72, No. 6, 2024, pp. 693-705. doi: 10.1080/13682199.2023.2219880.

[15] B. Patwari, U. Nandi, S. K. Ghosal, "Image steganography based on difference of Gaussians edge detection", Multimedia Tools and Applications, Vol. 82, No. 28, 2023, pp. 43759-43779. doi: 10.1007/s11042-023-15360-7.

[16] D. R. I. M. Setiadi, "Payload enhancement on least significant bit image steganography using edge area dilation", International Journal of Electronics and Telecommunications, Vol. 65, No. 2, 2019, pp. 287-292. doi: 10.24425/ijet.2019.126312.

[17] O. Y. Abdulhammed, "A novel approach of steganography by using strong edge detection and chaos theory", Multimedia Tools and Applications, Vol. 81, No. 13, 2022, pp. 17875-17888. doi: 10.1007/s11042-022-12643-3.

[18] J. Singh, A. K. Singh, S. S. Chauhan, "Enhanced edge-based steganography using image segmentation and random LSB substitution for secure data hiding", International Journal on Smart Sensing and Intelligent Systems, Vol. 18, No. 1, 2025. doi: 10.2478/ijssis-2025-0049.

[19] R. Shmueli, D. Mishra, T. Shmueli, O. Hadar, "A novel technique for image steganography based on maximum energy seam", Multimedia Tools and Applications, Vol. 83, No. 28, 2024, pp. 70907-70920. doi: 10.1007/s11042-024-18476-6.

[20] T. D. Nguyen, H. Q. Le, "A novel secure image steganography scheme based on Hamming encoding and LSB matching revisited using Lah Transform", Multimedia Tools and Applications, Vol. 83, No. 32, 2024, pp. 77347-77375. doi: 10.1007/s11042-024-18633-x.

[21] A. Yerimbetova, E. Daiyrbayeva, E. Merzlyakova, M. Sambetbayeva, A. Bayekeyeva, U. Berzhanova, "Adaptive Image Steganography Using Linear Hash Functions and Chessboard-Partitioned Blocks", Proceedings of the 10th International Conference on Computer Science and Engineering, Istanbul, Turkiye, September 2025, pp. 1362-1367. doi: 10.1109/UBMK67458.2025.11206801.

[22] F. Albkosh, M. Elsharif, "An Improved Image Steganography Method for Secure and High-Capacity Data Transmission Using Adaptive LSB Embedding", Journal of Alasmarya University: Applied Sciences, Vol. 10, No. 2, 2025, pp. 163-181. doi: 10.59743/jau10.v2.2182.

[23] A. A. A. Gutub, "Pixel Indicator Technique for RGB Image Steganography", Journal of Emerging Technologies in Web Intelligence, Vol. 2, No. 1, 2010, pp. 56-64. doi: 10.4304/jetwi.2.1.56-64.

[24] J. Pandey, K. Joshi, M. Jangra, M. Sain, "Pixel Indicator Steganography Technique with Enhanced Capacity for RGB Images", Proceedings of the International Conference on Intelligent Computing and Control Systems, Madurai, India, 15-17 May 2019, pp. 738-743.

[25] F. M. A. Albkosh, A. S. S. Mohamed, A. A. Albakoush, A. A. Albkush, "An Enhanced Method to Secure the High Embedding Capacity Steganography Based on LSB Indicator", Proceedings of the IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering, Tripoli, Libya, 19-21 May 2024, pp. 494-499. doi: 10.1109/MI-STA61267.2024.10599765.

[26] R. S. Hameed, S. S. Mokri, M. Sabah Taha, M. M. Taher, "High Capacity Image Steganography System based on Multi-layer Security and LSB Exchanging Method", International Journal of Advanced Computer Science and Applications, Vol. 13, No. 8, 2022. doi: 10.14569/IJACSA.2022.0130814.

[27] S. Ghoul, R. Sulaiman, "Imperceptible Image Steganography Technique Using a Novel PIT-Based Technique", Proceedings of the International Conference on Cyber Resilience, Dubai, United Arab Emirates, 6-7 October 2022, pp. 1-7. doi: 10.1109/ICCR56254.2022.9995838.

[28] A. Stojkovic, I. Shopovska, H. Luong, J. Aelterman, L. Jovanov, and W. Philips, "The effect of the color filter array layout choice on state-of-the-art demosaicing", Sensors, Vol. 19, No. 14, 2019. doi: 10.3390/s19143215.

[29] H. K. Tayyeh, A. S. A. Al-Jumaili, "A combination of least significant bit and deflate compression for image steganography", International Journal of Electrical and Computer Engineering, Vol. 12, No. 1, 2022, pp. 358-364. doi: 10.11591/ijece.v12i1.pp358-364.

[30] J. Khandelwal, V. K. Sharma, D. Singh, A. Zaguia, "DWT-SVD based image steganography using threshold value encryption method", Computers, Materials and Continua, Vol. 72, No. 2, 2022, pp. 3299-3312. doi: 10.32604/cmc.2022.023116.

[31] J. Mielikainen, "LSB matching revisited", IEEE Signal Processing Letters, Vol. 13, No. 5, 2006, pp. 285-287. doi: 10.1109/LSP.2006.870357.

[32] A. G. Weber, "SIPI (Signal and Image Processing Institute)", Signal and Image Processing Institute, University of Southern California, Department of Electrical Engineering, Los Angeles, CA, USA, http://sipi.usc.edu/database/database.php?volume=misc (accessed: 2025)

[33] P. Bas, T. Filler, T. Pevný, "BOSS (Break Our Steganographic System)", https://dde.binghamton.edu/download/ImageDB/BOSSbase_1.01.zip (accessed: 2025)

[34] Al-Jarrah, Mudhafar, "RGB-BMP Steganalysis Dataset", https://data.mendeley.com/datasets/sp4g8h-7v8k/1 (accessed: 2025)

[35] Browserling Inc., "On line file tools", https://online-filetools.com/generate-random-text-file (accessed: 2025)

[36] A. A. Almayyahi, R. Sulaiman, F. Qamar, A. E. Hamzah, K. Malaysia, U. Bangi, "High-Security Image Steganography Technique using XNOR Operation and Fibonacci Algorithm", International Journal of Advanced Computer Science and Applications, Vol. 11, No. 10, 2020, pp. 511-522. doi: 10.14569/IJACSA.2020.0111064.

[37] K. Prabha, I. Shatheesh Sam, "An effective robust and imperceptible blind color image watermarking using WHT", Journal of King Saud University - Computer and Information Sciences, Vol. 34, No. 6, 2022, pp. 2982-2992. doi: 10.1016/j.jksuci.2020.04.003.

[38] R. Apau, M. Asante, F. Twum, J. Ben Hayfron-Acquah, K. O. Peasah, "Image steganography techniques for resisting statistical steganalysis attacks: A systematic literature review", PLoS One, Vol. 19, No. 9, 2024, p. e0308807. doi: 10.1371/journal.pone.0308807.

[39] A. Westfeld, A. Pfitzmann, "Attacks on steganographic systems breaking the steganographic utilities ezstego, jsteg, steganos, and s-tools-and some lessons learned", Lecture Notes in Computer Science, Springer, 2000, pp. 61-76. doi: 10.1007/10719724_5.

[40] J. Fridrich, M. Goljan, "Practical steganalysis of digital images: state of the art", Proceesdings of SPIE 4675, Security and Watermarking of Multimedia Contents IV, San Jose, CA, United States, April 2002, pp. 1-13. doi: 10.1117/12.465263.

[41] J. Fridrich, J. Kodovsky, "Rich models for steganalysis of digital images", IEEE Transactions on Information Forensics and Security, Vol. 7, No. 3, 2012, pp. 868-882. doi: 10.1109/TIFS.2012.2190402.

[42] G. Xu, H. Z. Wu, Y. Q. Shi, "Structural design of convolutional neural networks for steganalysis", IEEE Signal Processing Letters, Vol. 23, No. 5, 2016, pp. 708-712. doi: 10.1109/LSP.2016.2548421.

[43] B. Boehm, "StegExpose - A Tool for Detecting LSB Steganography", https://github.com/b3dk7/StegExpose (accesse: 2025)

[44] D. B. Khadse, G. Swain, "Data Hiding and Integrity Verification based on Quotient Value Differencing and Merkle Tree", Arabian Journal for Science and Engineering, Vol. 48, No. 2, 2023, pp. 1793-1805. doi: 10.1007/s13369-022-06961-9.