Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.061659

ARTICLE

Edge-Based Data Hiding and Extraction Algorithm to Increase Payload Capacity and Data Security

Hanan Hardan[1,*], Osama A. Khashan[2,*] and Mohammad Alshinwan[1]

[1]Faculty of Information Technology, Applied Science Private University, Amman, 11931, Jordan
[2]Research and Innovation Centers, Rabdan Academy, Abu Dhabi, 114646, United Arab Emirates

*Corresponding Authors: Hanan Hardan. Email: h_hardan@asu.edu.jo; Osama A. Khashan. Email: okhashan@ra.ac.ae

Received: 29 November 2024; Accepted: 18 March 2025; Published: 09 June 2025

ABSTRACT: This study introduces an Edge-Based Data Hiding and Extraction Algorithm (EBDHEA) to address the problem of data embedding in images while preserving robust security and high image quality. The algorithm produces three classes of pixels from the pixels in the cover image: edges found by the Canny edge detection method, pixels arising from the expansion of neighboring edge pixels, and pixels that are neither edges nor components of the neighboring edge pixels. The number of Least Significant Bits (LSBs) that are used to hide data depends on these classifications. Furthermore, the lossless compression method, Huffman coding, improves image data capacity. To increase the security of the steganographic process, secret messages are encrypted using the XOR encryption technique before being embedded. Metrics such as the Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM) are used to assess the efficacy of this algorithm and are compared to previous methods. The findings demonstrate that the suggested approach achieves high similarity between the original and modified images with a maximum PSNR of 60.7 dB for a payload of 18,750 bytes, a maximum SSIM of 0.999 for a payload of 314,572.8 bytes, and a maximum Video Information Fidelity (VIF) of 0.95 for a payload of 23,592 bytes. Normalized Cross-Correlation (NCC) values are very close to 1. In addition, the performance of EBDHEA is implemented on Secure Medical Image Transmission as a real-world example, and the performance is tested against three types of attacks: RS Steganalysis, Chi-square attack, and visual attack, and compared with two deep learning models, such as SRNet and XuNet.

KEYWORDS: Steganography; least significant bit (LSB); edge detection; stego-image; data hiding

1 Introduction

With the fast-paced advancements in electronic telecommunications and extensive Internet use, protecting information from unauthorized access—whether by hackers or accidental recipients—has become a crucial issue today. To tackle this challenge, researchers have created a range of security techniques for data transmission, such as steganography and cryptography. In the quest for better protection, some methods combine both approaches, utilizing their strengths to reach higher security levels [1].

Cryptography involves using mathematical principles to encode and decode information, ensuring messages remain secure by converting understandable data (plaintext) into an unreadable format (cipher text) [2]. A cryptosystem comprises plaintext, encryption and decryption algorithms, cipher text, and a key. Plaintext refers to data in its normal, readable form. Encryption involves converting plaintext into cipher text using a specific key, while decryption reverses this process, extracting plaintext from the cipher text. The key

is crucial information to manage the cryptosystem and is shared only between the sender and receiver [2]. Despite its effectiveness in data security, cryptography can be vulnerable to cryptanalysts who may break ciphers by analyzing the cipher text contents to retrieve the plaintext.

Steganography is the process of concealing secret messages within various types of media files, such as text, audio, image, and video, in a manner that completely hides the presence of the secret message. This involves replacing redundant bits in the chosen medium with bits from the secret message [3]. Image Steganography specifically focuses on using digital images as the carrier for secret data, making it a significant area of research in recent years.

Steganography can be applied in two domains: spatial domain and frequency domain [4]. However, applying it in the spatial domain is preferable due to its lower computational requirements, simplicity, and higher storage capacity compared to the frequency domain. In this domain, a variety of techniques are employed, including but not limited to Least Significant Bits (LSB), Exploiting Modification Direction (EMD), Histogram-based techniques, Mapping-based approaches, Pixel Value Difference (PVD), Multi-Base Notation System (MBNS), Pixel/Block Indicator methods, Pixel Value Prediction (PVP), Edge-based strategies, Pixel Pair Matching (PPM), and applications based on Machine Learning [5,6].

In image steganography, data is hidden in a color or gray scale image called a cover image (CI), while the image including the secret data is called the stego-image (SI).

In contrast, Steganalysis involves uncovering concealed content. Steganalysis employs statistical and/or visual analysis techniques to unveil secret data within a stego-image [7]. Various analytical techniques have been developed to retrieve significant concealed data from stego-images. As a result, the concealment method proposed in this research aims to withstand new forms of visual and statistical attacks.

TAML framework that incorporates time-correlation in the sieving process of malicious application detection techniques was proposed by AlSobeh et al. [8]. Their study demonstrated the significance of temporal data in creating and applying Machine Learning algorithms for mobile malware detection. Zhang et al. suggest an information-loud image transmission technique which uses a mapping dictionary to hide covert data within images. Through semantic segmentation and style transfer, blend images are transformed into powerful public images called stego-images, which conceal secret images. Subsequently, a particular trained reconstruction network efficiently retrieves and decodes the concealed information [9]. Akram et al. attempt to develop a ML based steganography image classification technique that uses Curvelet transformation to capture features from both low and high payload images. As a means of classification, the image is determined as either a stego image or a cover image by using Support Vector Machine (SVM), which is one of the widely used classification methods [10].

Another example is the FloodDetector system, which integrates machine learning techniques to detect DoS flooding attacks in SDN that have not been defined. This shows how intelligent systems can adapt to such a dynamic environment. All in all, these plusses show how machine learning and context understanding can be beneficial in boosting security across different spheres of technology.

This paper introduces an improved steganography technique leveraging the Human Visual System (HVS), which is less sensitive to alterations in sharp-edge regions. The robustness of this technique is ensured by utilizing the Canny edge detection method, enhanced with dilation morphological operators, for the detection of edge regions in modified images. The effectiveness of edge detection is crucial in determining how well an algorithm performs across different criteria. When edge detection is accurate, it allows for the precise identification of embedding regions, which enhances embedding accuracy and stability by reducing data loss during extraction. It also boosts capacity by expanding the number of edge pixels available for

embedding, especially in high-contrast areas. Moreover, high-quality detection preserves visual integrity by limiting changes to less noticeable parts of the image.

In contrast, ineffective edge detection decreases embedding capacity, introduces visible distortions, and makes the image more susceptible to processing operations, negatively affecting the hidden data's robustness [11]. The method determines the number of bits used per pixel for embedding secret messages based on the pixel's location within the image. Pixels are divided into three categories: those within edge regions, those adjacent to edge regions, and those outside both the edge and adjacent regions. As a result, the number of bits used for embedding is highest for pixels in the edge regions, slightly lower for pixels adjacent to edges, and lowest for pixels outside these areas. Since the number of bits used for embedding varies from pixel to pixel, it is more difficult for attackers to detect and retrieve the secret message. The proposed algorithm hides information with a high embedding rate by developing the least significant bits (LSB) and enhancing data hiding based on edge detection, where the secret message is embedded in the image in different proportions depending on the location of pixels. Edge areas and areas near them are included in higher proportions than the rest of the image to reduce the possibility of revealing hidden information while maintaining image quality. On the other hand, the hidden information was encrypted to build a highly secure steganography algorithm. Information compression was also used to increase the upload rate within the cover image.

The remaining sections of the paper are structured as follows: Section 2 outlines the related work, Section 3 introduces the proposed algorithm, Section 4 discusses the experimental results, and finally, Section 5 concludes this work.

2 Related Work

Numerous steganography techniques have been devised to preserve concealed information against diverse threats. Some approaches take advantage of the LSB approach, incorporating enhancements to support its security and increase the payload capacity of Secret Messages (SM).

Steganography involves embedding secret data, such as text, sounds, or videos, within another file or into different files. This practice is becoming more necessary due to the increasing need to share confidential information online securely. Steganalysis is the field dedicated to detecting and decoding such hidden information.

In the work by Ref. [12], it's noted that selecting an encryption method involves balancing speed and security. Quick encryption methods might compromise security, whereas more straightforward methods are fast but less secure.

Research by Ref. [13] introduced techniques that embed multiple bits into each pixel using a refined version of the least significant bit (LSB) technique to evade detection by steganalysis. This development was spurred by findings that statistical methods like the chi-square or K-S tests could uncover hidden information, leading to the advancement of modified LSB (MLSB) techniques.

According to Ref. [14], the main categories of media used in steganography are text, images, audio, and video. The first three categories hide data within the same type of file. More complex techniques aim to disguise information within protocols or platforms. Research by Ref. [15] explored steganography using redundant residue number system (RRNS) codes and introduced methods for RRNS-based steganography without distorting the original data.

The challenge in hiding data within pixels revolves around the number of bits altered per pixel. The goal is to maintain high security and balance the amount of hidden data and alteration visibility.

The simplest method in steganography, the least significant bit (LSB) technique, alters the least noticeable bits of data, making it challenging to detect with the naked eye but vulnerable to statistical detection methods. Research by Ref. [16] suggested combining modified LSB with multi-directional pixel-value differencing (MDPVD). Research by Ref. [17] and others have developed methods for embedding data in images using variations of the LSB technique, enhancing security through different strategies, including data shuffling and compression.

Converting data within images faces several challenges, such as ensuring data security, maintaining image quality, optimizing the amount of hidden data, and minimizing computational demands. A method leveraging the less detectable aspects of the LSB technique and focusing on significant features detectable by human vision has been proposed for color images. This approach, suggested by Swain (2019), divides the image into blocks of pixels, using two bits for LSB in each pixel and the other six for quotient value differencing (QVD), balancing concealment, image integrity, and detection avoidance.

The image size of the multi-level system expands progressively from one level to the subsequent level. Research by Ref. [18] combines LSB substitution and pixel-value differencing (PVD) to enhance the capacity for hiding information in digital images. This method increases the amount of data that can be embedded and has also proven effective in resisting RS detection attacks. As a result, it significantly enhances the security of transmitting confidential messages.

Additionally, approaches within this domain focus on utilizing edges for improved effectiveness. Research by Ref. [19] proposed fuzzy logic and canny edge detection for a color image, which exhibits effective embedding capacity. Nonetheless, this approach lacks a detection mechanism for identical edge pixels within the stego-image. Research by Ref. [20] proposed a novel approach to detect edges, tailored explicitly for steganography, where the original image is segmented into 3×3 non-overlapping blocks for edge detection. Among these blocks, four corner pixels are a benchmark for accurately identifying edge blocks within the stego-image. The distinction between these reference pixels' horizontal, vertical, and diagonal pairs determines whether a block qualifies as an edge. Only five pixels are utilized for embedding, employing a highly efficient and rapid embedding technique. This methodology underwent testing in both spatial and transform domains, revealing superior data embedding capacity within the spatial domain. Research by Ref. [21] proposed a steganography technique involving hybrid edge detection applied to images was developed. Multiple edge detectors were employed on images cleared of m-bits, and the resulting edge images were combined using the AND operator. The cover pixels were categorized as either edge or non-edge pixels. Simultaneously, the secret message was encrypted using chaotic methods. X-bits were then embedded into edge pixels and y-bits into non-edge pixels, with x being greater than y, creating a stego-image. Research by Ref. [5] proposed a novel steganography method that relies on fuzzy edge detection for efficient image data concealment. The cover image undergoes masking, and fuzzy edge detection is applied to preserve edge details. The embedding of bits in a pixel is contingent upon its status as an edge pixel, with more bits for such pixels. For non-edge and non-background pixels, the amount of data embedded is determined by the Euclidean distance from the nearest edge pixel, guided by a Gaussian function.

To improve data-driven security for Internet of Things systems, an application layer that combines the Behavior-Interaction-Priority (BIP) feature, model checking, and self-adaptation model is created. Research by Ref. [22] provides an integrated method for IoT security. This strategy might greatly enhance mitigating and detecting security issues like virus assaults, phony data, and safe system transactions. The framework model comprises four main parts: Secure Threat Detection and Response, Secure Data Collection and Storage, Data Analytics, and Continuous Monitoring.

It is essential to consider information security when exchanging sensitive data. Secret data is effectively kept hidden via steganography and cryptography. Transforming private messages into an unreadable

format is the primary goal of cryptography. Cryptography must be used with other more secure methods since it raises doubts about clandestine communication. On the other hand, steganography hides private information in a cover medium to allay suspicions of secret communication. Most practitioners employ various steganographic algorithms chosen for their ability to convey heavy payloads together with a high-quality cover image. Research by Ref. [23] presents a new technique with improved payload and strong imperceptibility to hide sensitive data in digital photos using fuzzy logic. The degree to which a pixel is related to any edge in an image is determined by fuzzy logic, which we use to identify the edges of the image. Then, a factor is added to the data to disguise it before embedding it in the low-value pixels. Our technique generates a high-quality stego-image that guarantees confidentiality in communication across the unreliable public network while successfully hiding substantial amounts of secret information.

The growing reliance on cloud-assisted IoT systems for storing and retrieving images has underscored the importance of secure and traceable methods. As highlighted by Zhang et al. [24], secure multikey image retrieval frameworks utilize privacy-preserving techniques to guarantee that only authorized users can access certain images. These systems combine multikey access control with traceability features to keep track of any unauthorized activities. Such frameworks are crucial when multiple users have access to the same database, ensuring data confidentiality and accountability. This idea enhances the proposed steganographic approach by adding extra layers of security in cloud environments.

Advanced image analysis techniques have been created to capture more detailed image features for various applications. One significant method involves using fractional-order weighted spherical Bessel-Fourier moments, which improve feature representation by merging spherical harmonics with fractional-order derivatives [25]. This approach is especially effective for tasks, such as pattern recognition and image classification, providing high accuracy and resilience against noise. While these techniques are generally used for feature extraction, combining them with steganography could enhance the identification of optimal embedding regions, thus improving the imperceptibility and security of concealed data.

Efficient and secure content-based image retrieval (CBIR) systems in cloud-assisted IoT environments have attracted considerable interest due to their capability to handle large volumes of visual data securely. Chen et al. [26] introduced a CBIR framework that merges encryption techniques with feature extraction, ensuring the secure indexing and retrieval of images while safeguarding user privacy. This framework closely aligns with the objectives of steganography, as it focuses on achieving a balance between security and efficiency in cloud-based systems. By integrating CBIR principles into steganography, the effectiveness of hidden data could be further improved, allowing for secure and efficient retrieval of embedded information in distributed systems. Table 1 compares the proposed approach and exciting methods.

Table 1: Comparison of proposed method and existing methods

Aspect	Proposed method	Existing methods
Embedding capacity	Higher capacity is due to expanded edge areas and three-pixel classes.	Limited capacity, as edge areas, are not optimized effectively.
Imperceptibility	Preserves high visual quality with minimal distortion (PSNR and SSIM values).	Distortion can be noticeable, especially with larger payloads.
Robustness	More resilient against steganalysis techniques.	Vulnerable to attacks, especially those analyzing edge or histogram patterns.

(Continued)

Table 1 (continued)

Aspect	Proposed method	Existing methods
Computational efficiency	Optimized with Canny edge detection and dilation, ensuring efficient processing.	Computationally intensive, requiring more resources (e.g., DCT, DWT).
Flexibility	Works consistently for both grayscale and color images.	Often limited to grayscale or requires modifications for color.
Security	Improved security through XOR encryption and complex pixel selection.	Weaker security due to predictable patterns or simpler detection methods.
Complexity	Slightly more complex due to the edge detection and dilation process.	Simpler implementation but compromises key features like robustness and security.

The steganography algorithm presented in this study focuses on Edge-Based Data Hiding and Extraction Algorithm (EBDHEA). This method uses the least significant bits (LSBs) in storage. The number of bits used in each pixel depends on how close it is to the edge areas in the cover image. This method combines the Huffman compression algorithm and XOR encryption to improve the steganographic process. Huffman coding is essential, offering a straightforward and efficient compression technique that reduces data size, boosts hiding effectiveness, and maintains low computational requirements, making it suitable for resource-limited, real-time applications. This compression maximizes the space available for hiding data without noticeable changes between the stego-image (SI) and the cover image, preserving the invisibility of the concealed information. XOR encryption, in contrast, adds a layer of protection by obscuring the data, making it harder for unauthorized individuals to detect the hidden content. While XOR does not reduce data size like Huffman coding, it ensures the data remains concealed, adding extra security to the steganographic process. By integrating both Huffman coding and XOR encryption, the EBDHEA algorithm achieves a balancebetween efficient data compression, maximized payload capacity, and secure data hiding, effectively meeting the goals of improving hiding effectiveness, increasing payload capacity, and ensuring no perceptible differences in the stego-image.

3 The Proposed Hiding Algorithm

The human visual system shows less sensitivity to density changes in sharp-edge regions than uniform regions within an image. Leveraging this characteristic, this research proposes an Edge-Based Data Hiding and Extraction Algorithm (EBDHEA). The proposed (EBDHEA) algorithms are implemented in two stages, with the first stage applying the edge-based data hiding algorithm (EBDHA) and the second stage using the edge-based data extraction algorithm (EBDEA).

3.1 Edge-Based Data Hiding Algorithm (EBDHA)

In the proposed technique, we categorize pixels within the cover image into three categories: Edge Pixel (EP) identified through the canny edge detection method, Pixels resulting from the expansion of adjacent edge (AEP) using the Dilation morphological operation depending on square structure of width equal 3, and pixel that are neither edge nor part of the adjacent edge pixels (NEP), see Figs. 1 and 2. The quantity of Least Significant bits (LSBs) utilized for concealing information depends on these categories. Three least significant

bits (3-LSBs) are used to embed data in the first category, two least significant bits (2-LSBs) are used in the second category, and one least significant bit (1-LSB) is used in the third category. The concealed message (SM) is embedded within a cover image (CI) following encryption through the XOR key and compression using the Huffman coding algorithm, see Fig. 3.

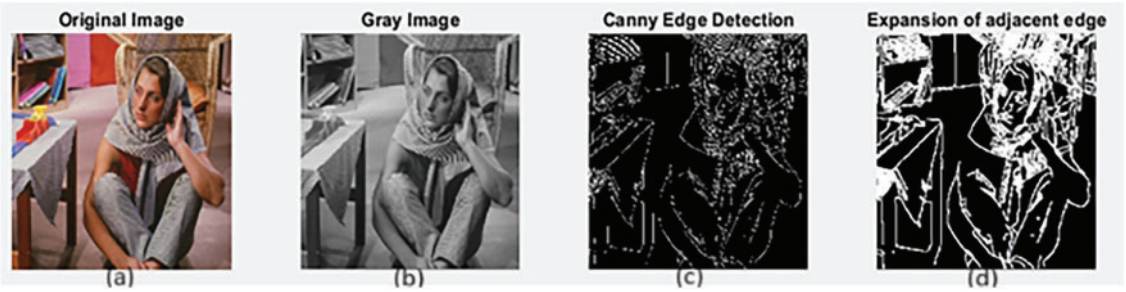

Figure 1: **(a)** The original image, **(b)** The result of converting the original image from a color image to a grayscale image, **(c)** The result of applying canny edge detection, **(d)** The result of applying dilation morphological operation in the image

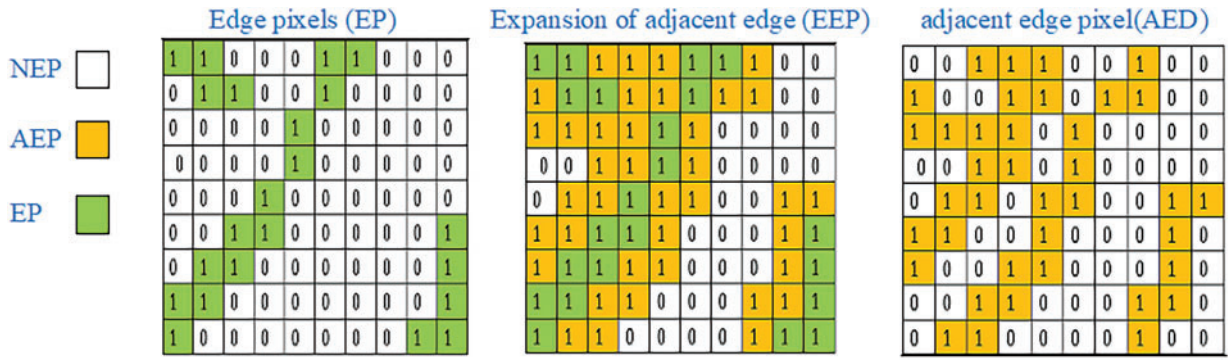

Figure 2: Categorize pixels within the cover image into three categories: EP, AEP, and NEP

Furthermore, a color cover image has been employed to conceal a secret message (SM) by utilizing the RGB image components. The process involves concealing information row by row within the pixels (24 bits) of the cover image, creating the stego-image as shown in Fig. 3. The number of bits to be concealed within a byte (i) (NBPBi) is computed according to Eq. (1) and stored in the corresponding location within a new matrix called reference matrix (RM) of the same size as the (CI). It is observed that the maximum limit for the replacement of bits per byte is three.

$$NBPB_i = \begin{cases} 3\,if\ (p\,is\,EP) \\ 2\,if\,(p\,is\,AEP) \\ 1\,if\,(p\,is\,NEP) \end{cases} \tag{1}$$

where (p) is a pixel, (EP) is an edge pixel, (AEP) is an adjacent edge pixel, and (NEP) neither an edge nor part of the adjacent edge pixels.

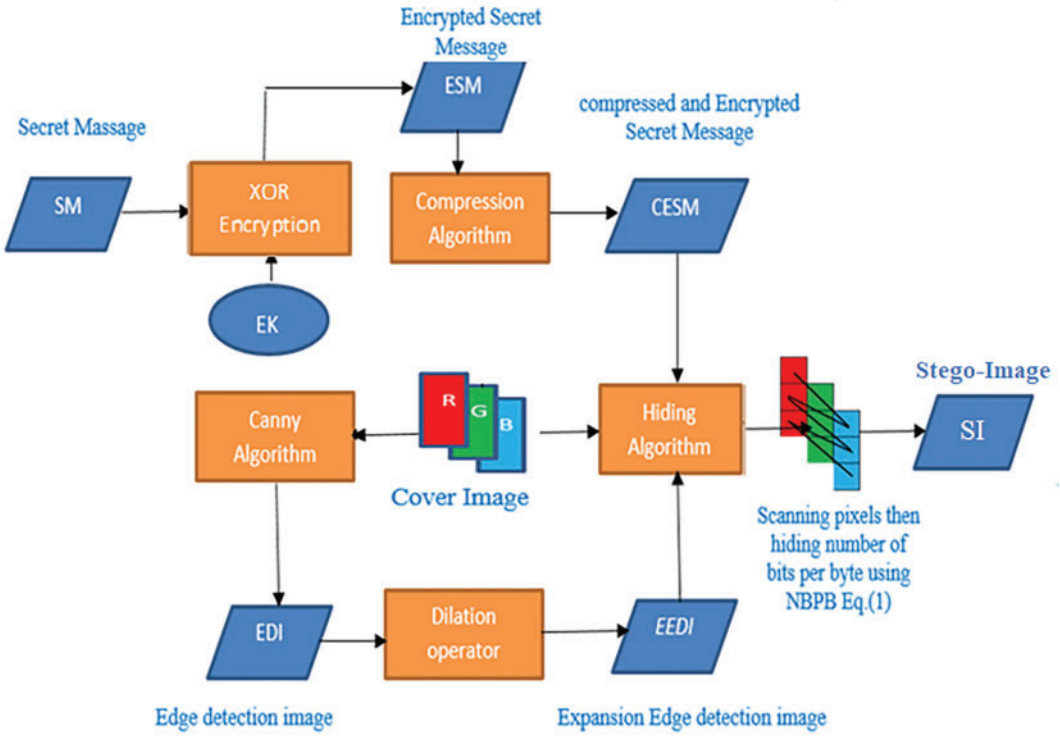

Figure 3: Proposed hiding stage

Edge-Based Data Hiding and Extraction Algorithm (EBDHEA) relies on key parameters such as thresholds, filter sizes, dilation filter configurations, Huffman coding, XOR encryption keys, and image properties. These elements directly impact edge detection, embedding accuracy, security, and overall image quality. Incorrect configurations can result in inefficiencies, image distortions, or degraded performance. To address these challenges, thresholds and kernel sizes are dynamically adjusted, secure encryption and efficient coding are employed, and the algorithm is tested on diverse images to optimize parameters for different scenarios. In addition, the algorithm ensures balanced embedding to maintain image quality.

3.1.1 The Hiding Algorithm

The main algorithm EBDHA (Algorithm 1) takes the secret message (SM), encryption key (EK), and the cover image (CI) as inputs and then hides the secret message using Sub-Algorithms (1, 2, 3, and 4). The output of the main algorithm is the stego-image

- Sub-Algorithm 1: PreProcessSecretMessage (.) This sub-algorithm encrypts the secret message (SM) using the encryption key (EK) implemented through the XOR approach and then compresses the result using the Huffman coding algorithm to get the compressed encryption secret message (CESM), Algorithm 2.
- Sub-Algorithm 2: EdgeMatrix (.) This sub-algorithm applies the Canny Algorithm after converting the cover image (CI) to the grayscale image to get the edge detection matrix (EDM), Algorithm 3.
- Sub-Algorithm 3: Adjacent EdgeMatrix (.) This sub-algorithm applies the Dilation morphological operation depending on the square structure of width equal to 3 on EDM to get the expansion edge detection matrix (EEDM), then subtracts EDM from EEDM to get the adjacent Edge detection matrix (AEDM), Algorithm 4.

- Sub-Algorithm 4: HidingMessage (.) This sub-algorithm hides the CESM in the cover image (CI) using an RGB order along the rows moving from left to right. The number of bits used in each pixel depends on the calculated number of bits per byte (NBPB) according to Eq. (1) based on EDM and AEDM. Finally, the algorithm returns to the stego-image, Algorithm 5.

Algorithm 1: EBDHA algorithm

1: **Start** the EBDHA
2: **Initialize** the following parameters:

- SM as the secret message
- EK as the encryption key
- ESM as the encryption secret message
- SI as the stego-image
- CESM as the compressed of encryption secret message
- CI as the cover image
- NBPB as the number of bits per byte
- EDM as the Edge detection matrix
- EEDM as the Expansion Edge detection matrix
- AEDM as the adjacent Edge detection matrix
- RM as the Reference matrix

3: **Input** SM, EK, CI
4: Convert SM into integer value representation
5: Find the total hiding pixel and total hiding data to ensure sufficient hiding
6: Call PreProcessSecretMessage(SM, EK) which return CESM
7: Call EdgeMatrix(CI) which return EDM
8: Call AdjacentEdgeMatrix(EDM) which returns AEDM
9: Call HidingMessage(CI, CESM, EDM, AEDM) which return the SI
10: Output SI
11: **End**

Algorithm 2: Sub-Algorithm 1: PreProcessSecreteMessage (.)

1: **Input** SM, EK
2: Encrypting SM using EK implemented through the XOR approach to get
3: Compressing ESM using the Huffman coding algorithm to get CESM;
4: **Return** CESM
5: **End** PreProcessSecreteMessage

Algorithm 3: Sub-Algorithm 2: EdgeMatrix (.)

1: **Input** CI;
2: Convert the (CI) to the grayscale image if it
3: Apply the Canny Algorithm to get EDM
4: **Return** EDM
5: **End** EdgeMatrix

Algorithm 4: Sub-Algorithm 3: Adjacent EdgeMatrix (.)

1: **Input** EDM;
2: Apply Dilation morphological operation depending on getting EEDM
3: Subtract EDM from EEDM to get AEDM
4: **Return** AEDM
5: **End** Adjacent
6: Edge Matrix

Algorithm 5: Sub-Algorithm 4: HidingMessage (.)

1: **Input** CI, CESM, EDM, AEDM;
2: **For** each pixel in CI using an RGB order along the rows moving from left to right
DO:
3: Calculate Nbpb according to Eq. (1) based on EDM and AEDM and store the result in (RM)
4: Hides several CESM bits in the pixel equal to the value calculated in the previous step
5: Shift to the next bit in the CESM.
6: **End for**
7: **Return** SI;
8: **End** HidingMessage

3.2 Edge-Based Data Extraction Algorithm (EBDEA)

The secret message extraction process from the stego-image has been implemented according to the reference matrix (RM) after decrypted by the XOR decryption key and decompressed by Huffman coding, as shown in Fig. 4.

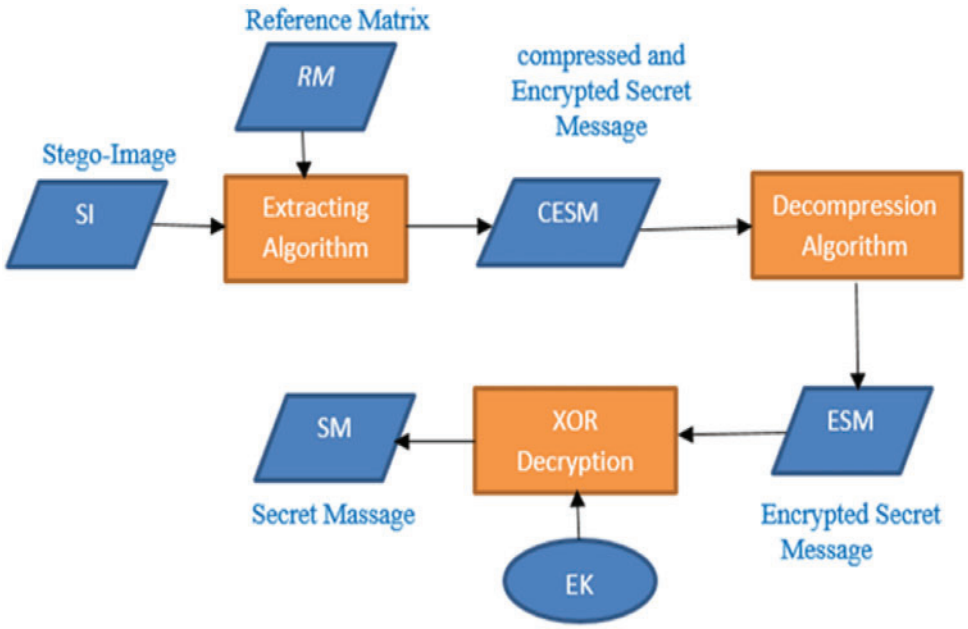

Figure 4: Proposed extraction stage

3.2.1 Extraction Algorithm

This algorithm inputs the stego-image (SI) and decryption key using the XOR approach (DK). The algorithm constructs the Reference Matrix (RM), which contains the number of bits used to hide the secret message for each pixel. The Compressed Encrypted Secret Message (CESM) is then extracted based on the reference matrix, decompressed using the Huffman compression algorithm, and decrypted using the DK.

The steps of the proposed extraction algorithm (EBDEA) have been delineated in Algorithms 6 and 7.

Algorithm 6: EBDEA (.)

1: **Start** the EBDHA
2: **Initialize** the following parameters:
- SM as a secret message.
- DK as the decryption key using the XOR approach; // DK=EK
- ESM as the encryption secret message
- CESM as the compressed of encryption secret message.
- SI as a stego-image.
- CI be a cover image.
- RM as the Reference Matrix

3: **Input** SI, DK;
4: Call BuildReferenceMatrex (CI), which returns RM;
5: **For** each pixel in the SI, the RGB arrangement is used along the rows, moving from left to right, to extract the CESM bits stored in the pixels according to the RM to finally obtain the CESM **DO**
6: Decompress of CESM to get ESM;
7: Message decrypted using same symmetric XOR DK to get SM;
8: Send SM to the output file;
9: **End for**
10: **End EBDEA.**

Algorithm 7: Sub-Algorithm 5: BuildReferenceMatrex (.)

1: **Input CI;**
2: Call EdgeMatrix(CI) which return EDM;
3: Call AdjacentEdgeMatrix(EDM) which returns AEDM
4: **For** each pixel in CI using an RGB order along the rows moving from left to right **DO**:
5: Calculate Nbpb according to Eq. (1) based on EDM and AEDM and store the result in (RM).
6: **End for**
7: **Return RM.**
8: **End** BuildReferenceMatrex.

3.3 Implementation of Hiding Algorithm (EBDHA)

Assume that we have one byte from the secret message, and we need to hide this byte in a cover image. The proposed technique categorizes pixels within the cover image into three categories: Edge Pixel (EP),

Pixels resulting from the expansion of adjacent edge (AEP), and pixels that are neither edge nor part of the adjacent edge pixels (NEP), the quantity of Least Significant bits (LSBs) utilized for concealing information depends on these categories. Three least significant bits (3-LSBs) are used to embed data in the first category, two least significant bits (2-LSBs) are used in the second category, and one least significant bit (1-LSB) is used in the third category. See Fig. 5.

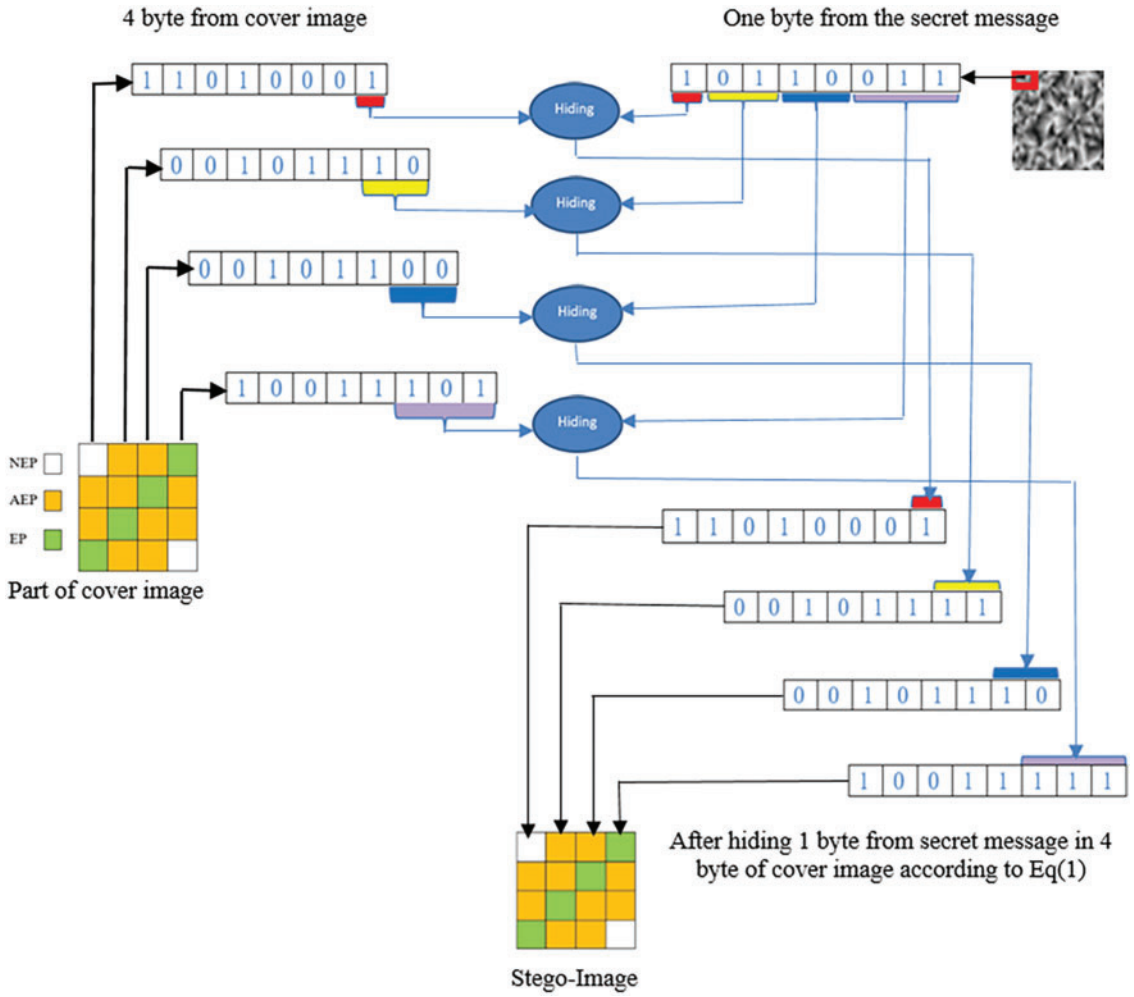

Figure 5: Implementation of the hiding process in the cover image

4 Discussion the Results

In this section, we examine the experimental outcomes conducted to assess the effectiveness of the proposed algorithm. The experimentation involved using color images sourced from the UCID v2 Database (with dimensions of 512 × 384 and 384 × 512) and standard test images such as Barbara, Baboon, and Peppers. Various metrics were employed to substantiate the achieved level of security.

4.1 Image Quality vs. Payload Capacity

The performance of the suggested method is assessed by utilizing variously sized images, as illustrated in Fig. 6. The evaluation metrics utilized consist of Peak Signal-to-Noise Ratio (PSNR), Mean Square Error (MSE), and Normalized Cross Correlation (NCC), as described by Eqs. (2)–(5).

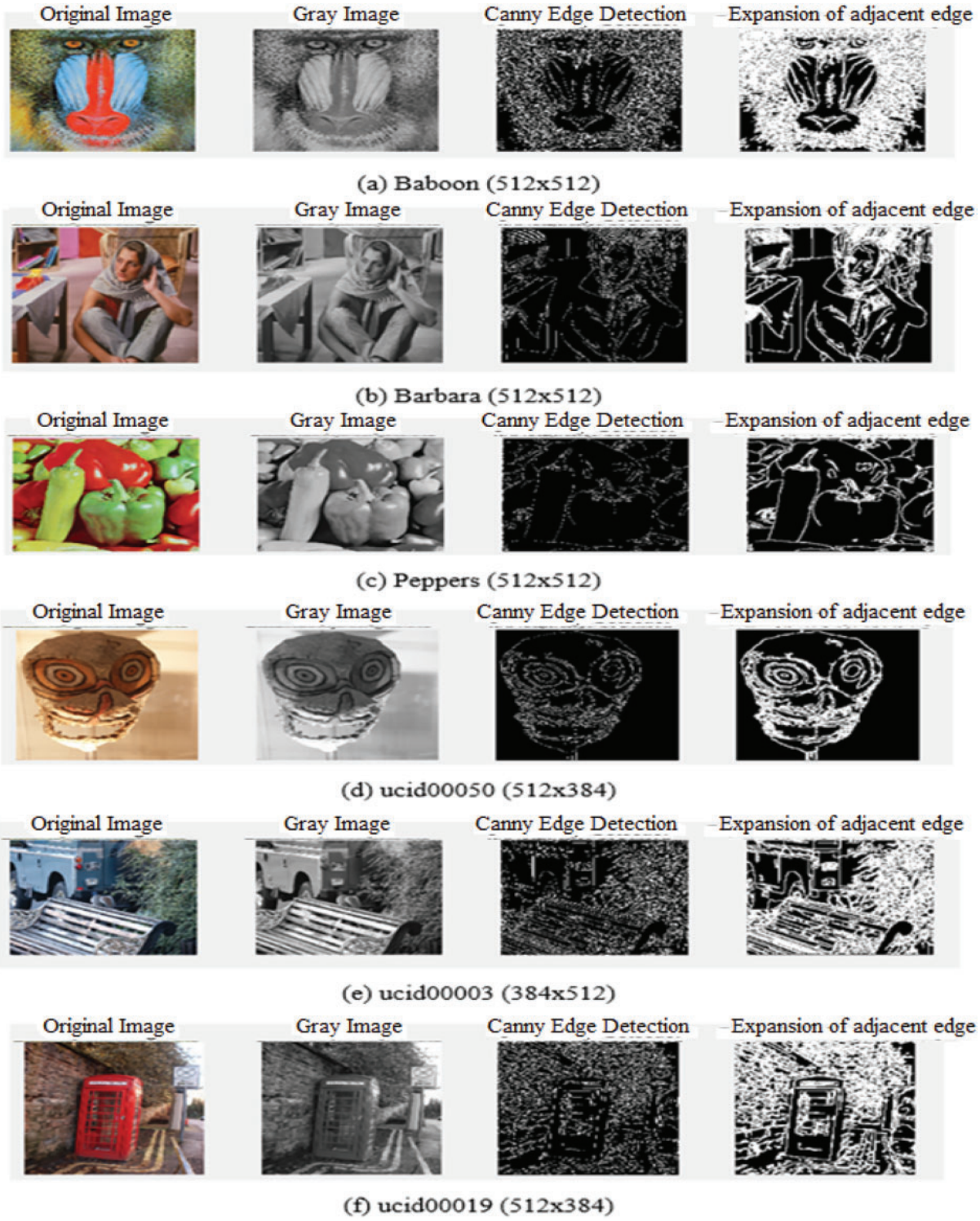

Figure 6: List of images used for testing the proposed method

$$PSNR = 10 * log_{10} \frac{255 * 255}{MSE} \tag{2}$$

$$MSE = \frac{MSE_R + MSE_G + MSE_B}{3} \tag{3}$$

where R, G, B are the Red, Green, and Blue image colors

$$MSE_k = \frac{1}{m * n} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(C_{ij} - S_{ij}\right)^2 ; k \in \{R, G, B\} \tag{4}$$

where (i, j) is the pixel in the cover image and stego-image at the same location

$$(NCC)_{i,j} = \frac{\sum_i^m \sum_j^n \left(\left(CI_{ij} - \mu_{CI}\right) \times \left(SI_{ij} - \mu_{SI}\right)\right)}{\sqrt{\sum_i^m \sum_j^n \left(CI_{ij} - \mu_{CI}\right)^2} \sqrt{\sum_i^m \sum_j^n \left(SI_{ij} - \mu_{SI}\right)^2}} \tag{5}$$

where μ_{SI} and μ_{CI} are the average pixels of the stego-image and cover image.

The benefits of employing EBDHEA methods include achieving a Normalized Cross Correlation (NCC) value closer to one, high Peak Signal-to-Noise Ratio (PSNR) values, and low Mean Square Error (MSE) values, as depicted in Table 2. These characteristics make it challenging for the human eye to discern any alterations in the cover image.

Table 2: Performance comparison of the proposed method and state-of-the-art methods

Image size (512 × 512)	Payload capacity (bits) × 10^4	Using MDLSB [27] PSNR (dB)	Using MCDHEA [28]			Using the proposed EBDHEA		
			PSNR (dB)	MSE (dB)	NCC (dB)	PSNR (dB)	MSE (dB)	NCC (dB)
Baboon	2.0	59.2	0.0181	65.6	1	0.0131	67.0	0.999995
	2.8	56.1	0.0250	64.2	1	0.0183	65.5	0.999992
	3.6	54.9	0.0324	63.0	1	0.0240	64.3	0.999990
	4.4	53.2	0.0401	62.1	0.999967	0.0298	63.4	0.999988
	5.6	53.4	0.0512	61.0	0.999933	0.0380	62.3	0.999984
Peppers	2.0	61.2	0.0122	67.3	1	0.0095	68.3	0.999997
	4.0	59.1	0.0212	64.9	0.999967	0.0172	65.8	0.999995
	6.0	55.3	0.0296	63.4	0.999967	0.0246	64.2	0.999993
	8.0	53.7	0.0390	62.2	0.999933	0.0327	63.0	0.999991
	10.5	53.7	0.0501	61.2	0.999933	0.0420	61.9	0.999989
Barbara	2.0	63.2	0.0232	65.2	1	0.0113	67.6	0.999996
	5.0	59.6	0.0341	64.6	1	0.0275	64.7	0.999991
	7.0	58.5	0.0361	63.8	0.999986	0.0386	62.3	0.999987
	10	55.9	0.0404	62.6	0.999966	0.0551	60.7	0.999982
	12.5	54.8	0.0514	61.8	0.999934	0.0686	59.8	0.999978

The proposed EBDHEA hiding algorithm is compared with the previous works: the MCDHEA algorithm by Ref. [28] and the proposed algorithm by Ref. [27], as shown in Table 2, and it was found that the proposed method achieved the best performance for the same payload for the same images.

Furthermore, the performance of the proposed method is evaluated using a different set of images with three metrics: PSNR, MSE, and SSIM, with payload capacities of 10%, 30%, and 40%. These metrics were applied to various images, including "Gold Hill" (a color image with a resolution of 720 × 576), "Carpet" (a grayscale image with a resolution of 1024 × 1024), and "Malamute" (a color image with a resolution of 1616 × 1080), as illustrated in Fig. 7. The results in Table 3 demonstrate that the proposed method performs exceptionally well across different types of images and resolutions.

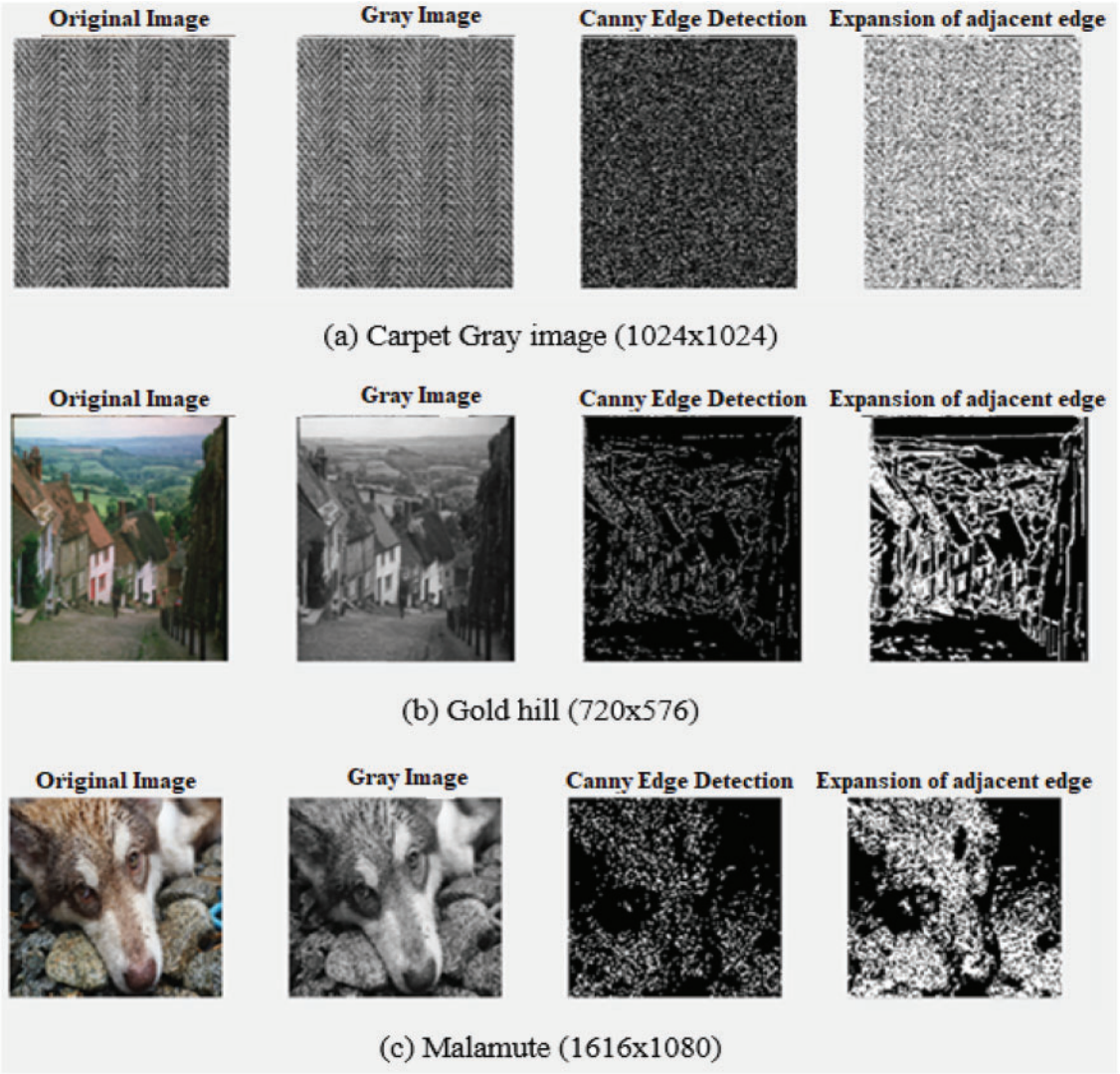

Figure 7: List of images used for testing the proposed method

Table 3: Performance comparative of different sets of images and payload

Image type	Measure	Payload capacity		
		10%	30%	40%
Gold hill (Color image) 720 × 576	PSNR	61.90285	57.06328	55.61734
	MSE	0.04196	0.127867	0.178382
	SSIM	0.9996	0.9991	0.9989
Carpet (Gray image) 1024 × 1024	PSNR	54.409851	49.666738	48.419860
	MSE	0.235557	0.702112	0.935609
	SSIM	0.9998	0.9995	0.9993
Malamute (Color image) 1616 × 1080	PSNR	68.03097	63.09059	61.78995
	MSE	0.010233	0.031919	0.043062
	SSIM	0.9999	0.9998	0.9997

4.2 Structural Similarity Index Measure (SSIM)

SSIM is a tool that measures how similar the structures of two images are when compared. It's a way of assessing how closely the patterns and details in one image match those in another Eq. (6).

$$SIMM\left(CIM, SIM\right) = \frac{\left(2\mu_{CIm}\mu_{SIm} + \left(\left(2^{24}-1\right)*0.01\right)^2\right)\left(2\sigma_{CIm,SIm} + \left(\left(2^{24}-1\right)*0.03\right)^2\right)}{\left(\mu_{CIm}^2 + \mu_{SIm}^2 + \left(\left(2^{24}-1\right)*0.01\right)^2\right)\left(\sigma_{CIm}^2 + \sigma_{SIm}^2 + \left(\left(2^{24}-1\right)*0.03\right)^2\right)} \tag{6}$$

where σ_{cIm},^{2} σ_{sIM}^2 are the variance of a cover and stego images, μ_{CIM}, μ_{SIM} are the mean of a cover and stego images, and σ_{CIM} σ_{SIM} is the covariance of a cover and stego images. For this examination, we utilized 50 color pictures chosen at random from the UCID v2 Database. These images come in sizes of 512 × 384 and 384 × 512 pixels, and we applied payload percentages of 10%, 30%, and 40%. The proposed method was compared with the previous works: Hardan et al. (2022) and the proposed algorithm by Elshare et al. (2018), as shown in Table 4, and it was found that the proposed method achieved good performance for the same payload for the images.

Table 4: The average values of SSIM by different steganography algorithms using 50 color pictures of size 512 × 384 and 384 × 512 were chosen at random from the UCID v2 database

Payload capacity	SSIM using MDLSB [27]	SSIM using MCDHEA [28]	SSIM using proposed EBDHEA
10%	0.9997	0.9999	0.9999
30%	0.9998	0.9997	0.9998
40%	0.9997	0.9996	0.9997

4.3 Euclidean Norm Test

The Euclidean norm test, as described in Eq. (7), was employed to illustrate the suggested algorithm's effectiveness in combating visual attacks. This test involves computing the distance (D) between the stego

image and the original cover image.

$$D = \sqrt{(R_{CI} - R_{SI})^2 + (G_{CI} - G_{SI})^2 + (B_{CI} - B_{SI})^2} \tag{7}$$

This experiment employed three color images sized at 512 × 512 pixels, each with payload percentages of 10%, 30%, and 40%. The aim was to evaluate the effectiveness of the proposed algorithm and compare its results with those of previous studies. MDLSB algorithm by Elshare, S., EL-Emam, N. 2018, and the proposed algorithm by Hardan. 2022. The smallest Euclidean norm (D) was reached using the proposed algorithm, as shown in the Figs. 8–10. Fig. 8 illustrates the Euclidean norm testing for the Lena image, Fig. 9 shows the Euclidean norm testing of the Baboon image, and Fig. 10 shows the Euclidean norm testing of the Peppers image.

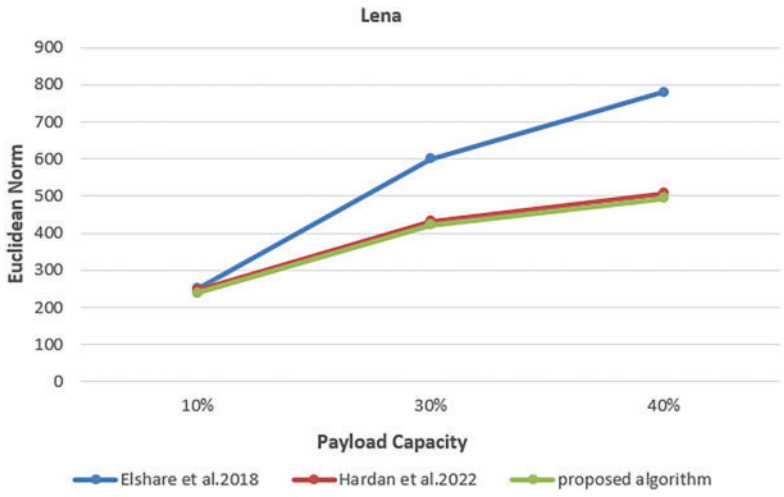

Figure 8: Euclidean norm testing of lena image

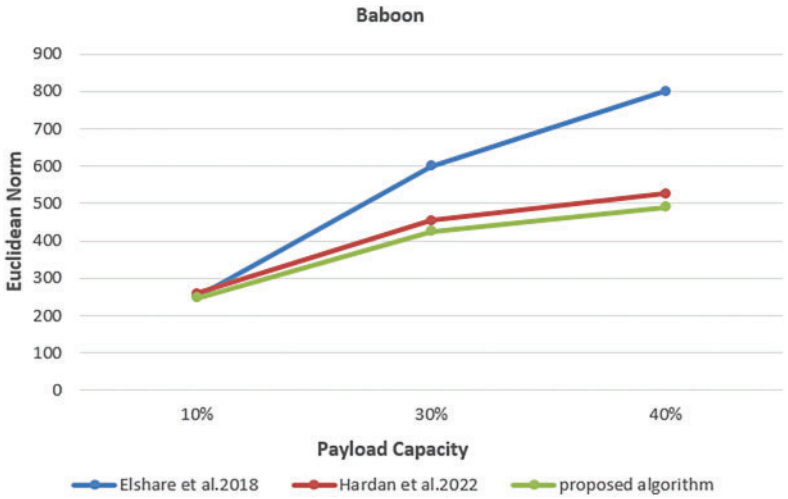

Figure 9: Euclidean norm testing of baboon image

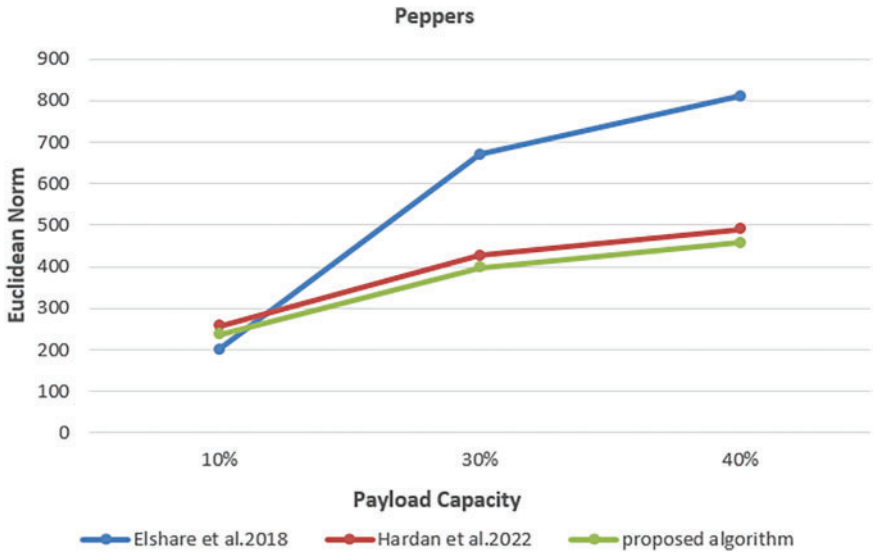

Figure 10: Euclidean norm testing of peppers image

4.4 Dissimilarity between Adjacent Pixels

Eqs. (8) and (9) calculate the dissimilarity between neighboring pixels in the stego-image and cover images. Here, $D(i, j)_{CIM}$ and $D(i, j)_{SIM}$ denote the disparity between horizontal adjacent pixel pairs for the cover image (CIM) and stego image (SIM), respectively, where $P(i, j)_{CIM}$ and $P(i, j)_{SIM}$ represent two pixels at position (i, j).

$$D(i,j)_{CI} = \left|P(i,j)_{CI} - P(i,j+1)_{CI}\right| \tag{8}$$

$$D(i,j)_{SI} = \left|P(i,j)_{SI} - P(i,j+1)_{SI}\right| \tag{9}$$

For this examination, three color images sized at 512 × 512 pixels were utilized, each embedded with a 40% payload. The disparity values range from −255 to +255, and the occurrence of each disparity value is tallied. Subsequently, a graph is generated depicting the pixel disparity values on the X-axis and their respective frequencies on the Y-axis, as depicted in Figs. 11–13. Notably, it was observed that the disparity values between the stego image and the cover image are highly similar.

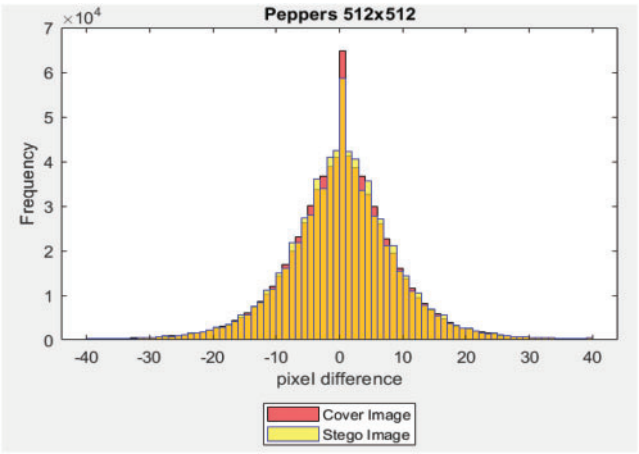

Figure 11: Dissimilarity between adjacent pixels with payload 40%

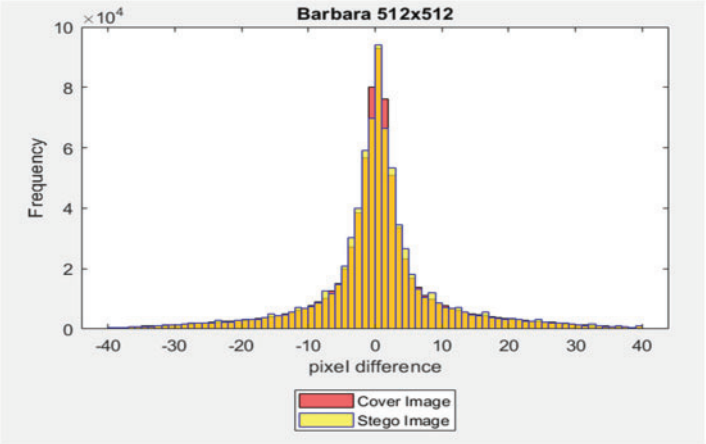

Figure 12: Dissimilarity between adjacent pixels with payload 40%

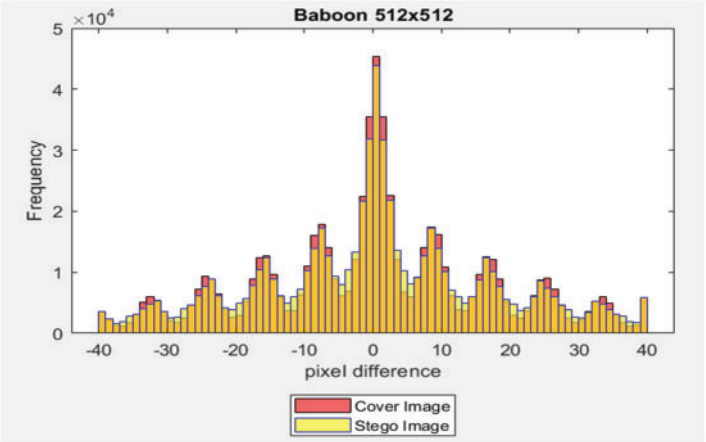

Figure 13: Dissimilarity between adjacent pixels with payload 40

4.5 Visual Information Fidelity

The Visual Information Fidelity (VIF) test evaluates how closely a stego-image resembles its original cover image. It utilizes the natural scene statistics (NSS), the Gaussian scale mixture (GSM) model, and the reference denotation (RD) along with image distortion (ID) and human visual system (HVS) metrics (REF). The VIF test is computed according to Eq. (10), which incorporates two mutual information measures. The initial measure relates to the information transferred between the inputs and outputs of undistorted HVS channels. In contrast, when distorted, the second measure deals with the exchange between inputs and outputs of the HVS channels. The ultimate production analyzed is the stego-image.

$$VIF = \frac{\sum_{jb\ and}\sum_{i}\log_2\left(\frac{\left(\lambda_{ji}^{CIM,SIM}\right)^2}{\left(\left(\lambda_{ji}^{SIM}\right)^2\times\left(\lambda_{ji}^{CIM}\right)^2-\left(\lambda_{ji}^{CIM,SIM}\right)^2+\lambda_{\mu}^2\times\left(\lambda_{ji}^{CIM}\right)^2\right)}+1\right)}{\sum_{jb\ and}\sum_{i}\log_2\left(\frac{\left(\lambda_{ji}^{CIM}\right)^2}{\lambda_{\mu}^2}+1\right)} \tag{10}$$

In each block at the jth sub-band, (λ) represents the standard deviation of the cover image (CIM) and the stego-image (SIM) in that block. Table 5 presents the VIF measure, indicating the fidelity of visual information. This study applies the testing to three images sourced from the standard color image database [29], each sized at 256 × 256 pixels. The proposed hiding process is evaluated based on payload capacities and the VIF metric. Results confirm the superiority of the proposed algorithm over other work such as [13]. The suggested algorithm demonstrates proficient performance.

Table 5: The visual information fidelity evaluation results

256 × 256 colored stego-image	Payload	Proposed work on [13]	Proposed work
Barbara	12%	92%	95%
Peppers	22%	N.A	93%
Baboon	32%	86%	91%

4.6 Secure Medical Image Transmission

One of the main issues facing e-healthcare is the medical image authentication procedure in medical image transfer. Medical picture transmission has extensively used digital watermarking systems as a data authentication tool. However, the act of watermarking digital images will cause some persistent distortions to the watermarked image, which could result in an incorrect diagnosis. One of the main problems with using digital watermarking systems for medical picture transmission is that the watermarked image will always be permanently distorted [30]. In this work, the EBDHEA secures the authentication process of the medical images during the transition. The following steps show the threat model in terms of attacks and countermeasures.

1. Attacks:
   - Statistical Attack: Attackers employ statistical analysis to find anomalies in the image that can point to hidden data.
   - Visual Attack: Attackers examine the pictures visually to look for any obvious variations that suggest data concealment.
   - Cipher text-only Attack: Attackers examine the encrypted data and try to decrypt the secret message embedded in it.
2. Countermeasures:
   - Edge-Based Embedding: This technique uses the inherent noise in the edge and surrounding regions—identified by the Canny edge detection method—to embed and conceal data.
   - By encrypting the secret message before embedding it, XOR encryption adds extra protection to guard against unwanted access.
   - Huffman Coding: Message compression boosts payload capacity and further obscures the buried data.

Table 6 shows the results of the Medical Image transmission, which achieve the following:

1. PSNR: An 18,750-byte payload can attain a maximum PSNR of 60.7 dB. This high PSNR value guarantees little visual deterioration because it shows that the image quality is nearly identical to the original image, even after data embedding.
2. SSIM: The highest reported SSIM for a payload of 314,572.8 bytes is 0.999. This number shows relatively little difference between the original and stego-images, suggesting that the images' structural integrity is maintained after embedding.

3. VIF: The technique attains a maximum VIF of 0.95 with a payload of 23,592 bytes. This high VIF value indicates that the embedded data preserves the overall visual quality of the image by not appreciably changing its visual information.
4. NCC: The NCC values are nearly 1 in every case. This high correlation proves that the embedding process introduces no appreciable differences, indicating a great similarity between the original and transformed.

Table 6 shows that the EBDHEA method performs safely by embedding large amounts of data while preserving excellent image quality and structural integrity.

Table 6: Performance measures for secure medical image transmission using EBDHEA

Metric	Value	Payload (Byte)
PSNR (dB)	60.7	18.750
SSIM	0.999	314.57280
VIF	0.95	23.592
NCC	0.9999	N/A

4.7 Evaluation of Security Using Deep Learning Based Steganalysis

To conduct a detailed analysis of the efficacy of the proposed Edge-based Data Hiding and Extraction Algorithm (EBDHEA), we set out to measure the algorithm's performance using two advanced deep learning-based models of steganalysis, SRNet and XuNet. These models are well-known in steganalysis for their power to reveal the presence of data embedded in stego-images by the delicate changes caused by the embedding process.

The UCID dataset and benchmark images like Baboon, Peppers, and Barbara are used to create the stego-images analyzed in the experiments and the EBDHEA algorithm is utilized. The payload capacities are 10%, 30%, and 40% of the cover image capacities. These results are attained by measuring detection accuracy, false positive rate (FPR) and false negative rate (FNR) measurements, which provide insight into how well the algorithm can secure. The performance of SRNet, XuNet, and EBDHEA in detecting payload capacities is assessed, with the findings presented in Table 7. A lower detection accuracy suggests a stronger resistance to steganalysis attacks.

Table 7: Evaluation results using deep learning algorithms and EBDHEA algorithm

Model	Payload (%)	Accuracy (%)	False Positive Rate (FPR) (%)	False Negative Rate (FNR) (%)
SRNet	10	51.8	48.2	49.6
	30	58.3	41.7	42.1
	40	64.7	35.3	39.2
XuNet	10	54.1	45.9	47.8
	30	61.5	38.5	41.6
	40	68.9	31.1	37.4
EBDHEA	10	48.7	51.3	52.0
	30	55.2	44.8	45.5
	40	62.4	37.6	38.9

The findings emphasize the robust security of the EBDHEA algorithm, especially when dealing with low to moderate payload capacities, where advanced steganalysis models struggle to detect hidden information. By utilizing edge-based embedding and adjusting the number of bits concealed per pixel, the algorithm creates variability that complicates the consistent detection of hidden data by steganalysis models.

4.8 Attack Analysis

This section tests the proposed method's performance against three types of attacks: RS Steganalysis, Chi-square attack, and Visual attack. The results achieved are compared in terms of different types of encryption modes to show the effectiveness of the proposed work, such as RC4 and EPR.

4.8.1 RS Steganalysis

RS Steganalysis is an analytical technique designed to detect hidden information in digital images, a common practice in steganography where data is concealed within seemingly normal media files. This method classifies pixel groups within an image as regular or singular based on predictability from adjacent pixels. By inverting the pixel values within these groups and analyzing shifts between these classifications, the RS method can infer the presence of embedded data. This approach is particularly valuable because it does not depend on knowledge of the specific steganography techniques used to hide the data, making it an effective universal tool for uncovering concealed information in images [31]. This work employs RS steganalysis to evaluate the proposed strategy's effectiveness. This technique scrutinizes 500 images with embedded data to uncover any concealed information. Through this approach, we can ascertain the efficacy of the embedding phase model described in Eq. (11).

$$\lambda = S\frac{\left[(a-b)^4\right]}{\left[(a-b)^2\right]^2} \tag{11}$$

In this context, the file size is labelled as S, while a represents random variables, and the mean variance is denoted as b. Additionally, the analysis pair is symbolized as λ. Typically, $\lambda = 3$ is used for standard analysis, whereas values of $\lambda > 3$ are considered for more advanced analysis. Table 8 shows that the EBDHEA method surpasses the other encryption modes in terms of the detection rate of secret images.

Table 8: Performance of the proposed method against the RS steganalysis

Payload	Comparative methods		
	RC4	**EPR**	**EBDHEA**
250	0.17	0.22	0.51
450	0.15	0.17	0.49
650	0.10	0.14	0.48
850	0.09	0.11	0.43
1050	0.06	0.10	0.41
1250	0.06	0.08	0.31
1450	0.046	0.06	0.21

4.8.2 Chi-Square Attack

A Chi-square attack is a statistical method used in cryptanalysis to help determine if there's a relationship between two categorical variables. It is commonly utilized in frequency analysis to study patterns within data sets. In this attack, the attacker aims to identify weaknesses or correlations in the encrypted data that can be exploited to decrypt the information. The Chi-square attack is based on the Chi-square test, a statistical hypothesis test that evaluates how likely an observed distribution is due to chance [32]. In a Chi-square attack, the attacker compares the expected frequency of specific patterns in the encrypted data with the actual frequency of those patterns. By calculating the Chi-square statistic, the attacker can assess whether the deviation between the expected and observed frequencies is significant enough to indicate a potential vulnerability in the encryption algorithm. This method is particularly effective when the encryption algorithm is not properly randomizing the data, leading to predictable patterns that attackers can leverage [33].

Additionally, the Chi-square attack can be used to analyze the randomness of a random number generator used in encryption schemes. If the random number generator is not producing truly random numbers, it may introduce biases or correlations that weaken the overall security of the encryption system. By subjecting the output of the random number generator to a Chi-square test, cryptanalysts can detect deviations from expected randomness and potentially exploit these weaknesses [34]. Table 9 illustrates the secret image identification of the EBDHEA method and other encryption modes; the result shows that the EBDHEA outperforms the RC4 and EPR.

Table 9: Performance of the proposed method against the Chi-square attack

Payload	Comparative methods		
	RC4	**EPR**	**EBDHEA**
250	0.10	0.13	0.39
450	0.06	0.12	0.35
650	0.049	0.11	0.33
850	0.041	0.045	0.33
1050	0.04	0.039	0.32
1250	0.032	0.045	0.32
1450	0.028	0.045	0.18

4.8.3 Visual Attack

The visual attack stands as the most straightforward steganalysis method. It involves visually inspecting the stego image with the naked eye to discern any concealed data. If the steganalysis model proves ineffective, a visual attack may occur. The rule governing visual attacks dictates that the size of the hidden message must be smaller than the higher bit-level dimension. Furthermore, visual attacks typically succeed more often with unencrypted data [35]. Table 10 demonstrates the reading ratio of the secret image of the EBDHEA method and other encryption modes; the result shows that the EBDHEA exceeds the RC4 and EPR.

Table 10: Performance of the proposed method against the visual attack

Payload	Comparative methods		
	RC4	EPR	EBDHEA
250	0.06	0.12	0.35
450	0.047	0.09	0.33
650	0.037	0.047	0.31
850	0.036	0.049	0.32
1050	0.035	0.042	0.25
1250	0.028	0.043	0.18
1450	0.026	0.043	0.19

Table 11 shows the results of combinations of different algorithms for compression and encryption in using the proposed EBDHEA. The comparison is important because it shows how different choices affect the algorithm's performance, image quality, and security. We assessed Huffman and LZW compression methods and XOR and AES encryption techniques on Gold Hill, Malamute, and Baboon, three popular images. According to the findings, Huffman compression is faster than LZW compression, leading to better performance in real-time applications. On the other hand, LZW compression provides better data compression at the cost of a slower processing speed. XOR encryption is also a very lightweight and quick protocol. It works well in cases where speed is critical but does not give the most substantial security. Moreover, AES encryption is slower but provides stronger security against attack. Depending on the application, the speed/security trade-off becomes critical. For example, a situation that requires high-speed processing with moderate security may prefer Huffman with XOR. In contrast, more sensitive data would require the robust security of AES, even though it is slower. Using metrics such as MSE, PSNR, SSIM, and NCC, it is shown throughout that the image quality is not affected while the data is embedded securely using our algorithm. Table 11 shows that EBDHEA can be modified according to requirements. Various parameters can be changed for practical applications, and this may be done optimally.

Table 11: Performance of the proposed method against the visual attack

Image	Compression method	Encryption method	Payload capacity	Time (MS)	MSE	PSNR	SSIM	NCC
Gold hill (Colorful) 720 × 576	Huffman	XOR	10%	2.82E + 04	0.040359	62.07195	0.9996	1
	Huffman	AES		1.14E + 04	0.051450	61.01694	0.9996	1
	LZW	XOR		2.29E + 04	0.033994	62.81717	0.9997	1
	LZW	AES		2.61E + 04	0.043557	61.74032	0.9997	1
	Huffman	XOR	30%	1.19E + 05	0.140433	56.65611	0.9992	1
	Huffman	AES		4.27E + 04	0.187536	55.39994	0.9991	0.9999
	LZW	XOR		1.17E + 05	0.089188	58.62778	0.9994	1
	LZW	AES		4.98E + 04	0.116097	57.48266	0.9994	1
	Huffman	XOR	40%	2.12E + 05	0.197435	55.17658	0.999	1
	Huffman	AES		6.30E + 04	0.268081	53.84815	0.9989	0.9999
	LZW	XOR		8.57E + 04	0.113678	57.57405	0.9993	1
	LZW	AES		1.18E + 05	0.149955	56.37122	0.9993	0.9999

(Continued)

Table 11 (continued)

Image	Compression method	Encryption method	Payload capacity	Time (MS)	MSE	PSNR	SSIM	NCC
Malamute (Colorful) 1616 × 1080	Huffman	XOR	10%	4.72E + 04	0.010016	68.124	0.9999	1
	Huffman	AES		1.47E + 04	0.013215	66.92032	0.9999	1
	LZW	XOR		1.60E + 04	0.008188	68.99958	0.9999	1
	LZW	AES		2.41E + 04	0.010641	67.86137	0.9999	1
	Huffman	XOR	30%	1.12E + 05	0.032049	63.07261	0.9998	1
	Huffman	AES		4.72E + 04	0.042685	61.8281	0.9998	1
	LZW	XOR		7.37E + 04	0.02167	64.77222	0.9998	1
	LZW	AES		8.49E + 04	0.028495	63.58337	0.9998	1
	Huffman	XOR	40%	2.32E + 05	0.043844	61.71174	0.9997	1
	Huffman	AES		6.82E + 04	0.058449	60.4631	0.9997	1
	LZW	XOR		1.18E + 05	0.026465	63.90417	0.9998	1
	LZW	AES		8.56E + 04	0.034934	62.69844	0.9998	1
Baboon (Colorful) (512 × 512)	Huffman	XOR	10%	1.55E + 04	0.10848	57.78781	0.9998	1
	Huffman	AES		8.11E + 03	0.126836	57.1065	0.9998	0.9999
	LZW	XOR		1.06E + 04	0.089872	58.60497	0.9999	1
	LZW	AES		1.53E + 04	0.104129	57.96414	0.9998	0.9999
	Huffman	XOR	30%	6.58E + 04	0.329434	52.96093	0.9995	0.9999
	Huffman	AES		2.88E + 04	0.374826	52.40267	0.9994	0.9998
	LZW	XOR		6.16E + 04	0.226258	54.59395	0.9996	0.9999
	LZW	AES		2.87E + 04	0.256321	54.05255	0.9996	0.9999
	Huffman	XOR	40%	1.11E + 05	0.435018	51.75336	0.9993	0.9998
	Huffman	AES		4.45E + 04	0.49438	51.20017	0.9992	0.9998
	LZW	XOR		4.19E + 04	0.273707	53.76674	0.9995	0.9999
	LZW	AES		4.91E + 04	0.309438	53.23499	0.9995	0.9999

Finally, we examine the effect of different edge detection methods on the performance of our data-hiding procedure. Since edge regions in photographs are more resistant to perceptual distortion, these regions are optimal for concealing information. To explore the most promising methodology for achieving this goal, we compared three well-known edge detection techniques: Canny, Sobel, and Prewitt. We aimed to evaluate the impact of each method on key performance characteristics: embedding capacity, image quality, robustness to steganalysis, and computational efficiency, thereby providing an informed choice of the best alternative depending on specific application needs. Table 12 summarizes these results and the trade-offs associated with each technique. The Canny edge detection method offers a respectable balance, achieving an embedding capacity of 25,000 bytes and maintaining high image quality with a PSNR of 62.1 dB and an SSIM of 0.9996. Canny also demonstrated excellent resistance against steganalysis attacks, with a resilience score 0.95. However, Canny's peak computational time of 120 MS is a significant limitation, which may restrict its usability in real-time applications. In contrast, the Sobel edge detection method achieved a higher embedding capacity of 27,000 bytes, making it ideal for scenarios requiring substantial data concealment. This benefit, however, comes at the cost of reduced image quality, reflected by a PSNR of 58.5 dB and an SSIM of 0.9985. Sobel exhibited moderate robustness to steganalysis, scoring 0.85 in resilience. With a computational time of 90 MS, Sobel strikes a fair balance between performance and efficiency.

Table 12: Comparison of edge detection methods

Edge detection method	Embedding capacity (Bytes)	PSNR (dB)	SSIM	Resilience to steganalysis (Score)	Computational time (MS)
Canny	25,000	62.1	0.9996	0.95	120
Sobel	27,000	58.5	0.9985	0.85	90
Prewitt	22,000	57.2	0.9978	0.80	70

On the other hand, the Prewitt method was the most efficient, with a computational time of only 70 MS. However, this speed is offset by a lower embedding capacity of 22,000 bytes and further declines in image quality, indicated by a PSNR of 57.2 dB and an SSIM of 0.9978. Prewitt also had the lowest resilience score of 0.80, making it the least secure option.

To conclude, as illustrated in Table 12, an edge detection method should align with the application's specific needs. When prioritizing high image quality and strong security, Canny stands out as the best choice despite its computational demands. Sobel provides a more favorable balance for applications that require embedding large data. Conversely, Prewitt is suitable when maximizing processing speed.

Furthermore, the study was extended by comparing the outcomes of image processing using different combinations of compression and encryption techniques (Huffman, LZW, XOR, and AES) on three images: "Gold Hill," "Malamute," and "Baboon." These techniques were applied with payload capacities of 10%, 30%, and 40%, as detailed in Table 11. Various metrics, such as processing time, PSNR, MSE, NCC, and SSIM, were used to evaluate the results, as shown in Table 11. For example, as the payload on "Baboon" increases from 10% to 40%, the PSNR decreases from 57.787 at 10% payload to 51.753 at 40% payload when using the Huffman-XOR combination. Figs. 14 and 15 will further clarify the impact of the payload on image quality by illustrating how different payload capacities affect PSNR and SSIM when applying various compression and encryption methods in the proposed method. Overall, LZW compression and XOR encryption offer a solid balance between maintaining image quality and processing time. Although increasing the payload capacity reduces image quality, the techniques still preserve high fidelity across the tested metrics. As shown in Figs. 14 and 15.

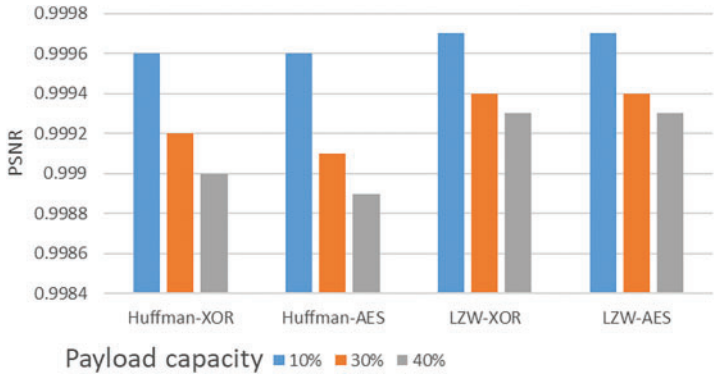

Figure 14: PSNR testing of Gold Hill image (720 × 576) using various combinations of compression and encryption methods (Huffman, LZW, XOR, and AES) with different payload capacities

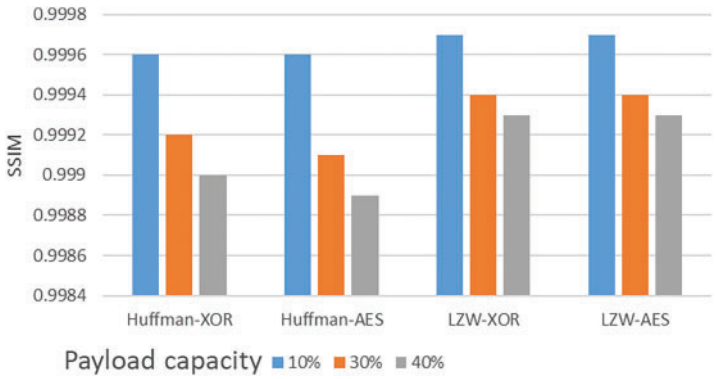

Figure 15: SSIM testing of Gold Hill image (720 × 576) using various combinations of compression and encryption methods (Huffman, LZW, XOR, and AES) with different payload capacity

4.8.4 Computational Complexity of EBDHEA

The EBDHEA's computational complexity mainly stems from its edge-based embedding, encryption, and compression methods. A key component of EBDHEA is the Canny edge detection algorithm, which operates with a complexity of O(N), where N represents the total number of pixels in the image. This step is crucial as it ensures that embedding occurs in less visually noticeable areas, thereby enhancing security and imperceptibility.

After edge detection, the dilation process used to identify adjacent edge pixels adds only a minimal overhead, keeping the overall pixel classification complexity linear. The XOR-based encryption applied to the secret message is a lightweight computational step with a complexity of O(M), where M is the size of the secret message. Additionally, using Huffman coding for compression results in a complexity of O(M log M), effectively reducing the message size and increasing embedding capacity while still being computationally efficient. Compared to more straightforward methods like Modified Least Significant Bit (MLSB), EBDHEA is more computationally demanding due to its advanced preprocessing steps. Yet, it is less intensive than hybrid techniques that involve frequency-domain transformations such as Discrete Cosine Transform (DCT). This balance between complexity and performance makes EBDHEA a suitable choice for applications that require high security, improved embedding capacity, and strong resistance to steganalysis.

4.8.5 Challenges and Limitations

The algorithm designed to improve the payload capacity in LSB image steganography through dilated hybrid edge detection may encounter certain challenges when applied to images with diverse characteristics, such as low-contrast or edge-poor images. Key limitations include:

- Edge area detection challenges: In low-contrast images, the edge detection process using the Canny detector might struggle to identify significant edge areas, resulting in fewer regions suitable for embedding data, ultimately reducing the payload capacity.
- Edge-poor image sensitivity: Images with fewer noticeable edges inherently limit the areas where pixel value changes can occur without compromising the image's visual quality. This makes embedding messages more difficult while maintaining imperceptibility.
- Noise amplification: In low-contrast images, even small changes in pixel values can become more noticeable, introducing unwanted noise and compromising the imperceptibility of the stego-image. These limitations highlight the challenges of applying the proposed method across a wide range of image types and underscore the need for further refinement to improve its robustness and versatility.

5 Conclusion

The Edge-Based Data Hiding and Extraction Algorithm (EBDHEA) discussed in this paper presents an advanced solution to hide data within images securely. It combines techniques such as the least significant bit method and Canny edge detection. This allows the algorithm to adjust the number of hidden bits in each pixel depending on how close it is to the edge areas in the original image. The goal here is to make the concealed data less noticeable in the resulting stego-image, taking advantage of the fact that changes in edge regions are harder to detect compared to smoother areas.

Furthermore, by incorporating XOR encryption keys, the algorithm enhances the security of the hidden data, providing an additional layer of defense against unauthorized access. Additionally, the lossless compression using the Huffman coding algorithm increases the capacity for embedding data within the image. The performance of the EBDHEA algorithm is meticulously assessed using established metrics like Peak Signal-to-Noise Ratio (PSNR), Mean Squared Error (MSE), and Structural Similarity Index (SSIM) and compared to other existing methods. The evaluation results underscore the proposed approach's superior embedding performance. Furthermore, the algorithm ensures no visible difference between the original cover image and the stego-image, preserving the image's quality. Moreover, the EBDHEA algorithm guarantees the complete recovery of concealed data. However, the proposed work is limited to dealing with dependency on edge regions; since the method embeds messages in the edge regions of images, its effectiveness is highly dependent on the quality and characteristics of these edges. Images with fewer or less distinct edges might achieve a different level of performance in terms of embedding capacity and robustness. More hybrid steganography techniques that integrate edge-based embedding with alternative approaches, such as texture-based and frequency domain techniques, will be used for future work. This combination aims to enhance various image types' robustness and embedding capacity.

Acknowledgement: The authors gratefully acknowledge the Rabdan Academy for their invaluable support and resources provided throughout this research, which significantly contributed to its successful completion.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Hanan Hardan; data collection: Hanan Hardan and Mohammad Alshinwan; analysis and interpretation of results: Hanan Hardan and Osama A. Khashan; draft manuscript preparation: Hanan Hardan, Osama A. Khashan, and Mohammad Alshinwan. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Lin Y, Xie Z, Chen T, Cheng X, Wen H. Image privacy protection scheme based on high-quality reconstruction DCT compression and nonlinear dynamics. Expert Syst Appl. 2024;257:124891. doi:10.1016/j.eswa.2024.124891.
2. Babu KR, Kumar SU, Babu AV. A survey on cryptography and Steganography methods for information security. Int J Comput Appl. 2010;12(3):13–7.
3. Mohamad FS, Yasin NSM. Information hiding based on audio steganography using least significant bit. Int J Eng Technol. 2018;7(4.15):536–8. doi:10.14419/ijet.v7i4.15.28363.
4. Liao X, Yin J, Guo S, Li X, Sangaiah AK. Medical jpeg image steganography based on preserving inter-block dependencies. Comput Elect Eng. 2018;67:320–9.

5. Dhargupta S, Chakraborty A, Ghosal SK, Saha S, Sarkar R. Fuzzy edge detection based steganography using modified gaussian distribution. Multimed Tools Appl. 2019;78:17589–606. doi:10.1007/s11042-018-7123-x.
6. Kim P-H, Ryu K-W, Jung K-H. Reversible data hiding scheme based on pixel-value dif-ferencing in dual images. Int J Distrib Sens Netw. 2020;16(7):1550147720911006. doi:10.1177/1550147720911006.
7. Yang C, Kang Y, Liu F, Song X, Wang J, Luo X. Color image steganalysis based on embedding change probabilities in differential channels. Int J Distrib Sens Netw. 2020;16(5):1550147720917826. doi:10.1177/1550147720917826.
8. AlSobeh AM, Gaber K, Hammad MM, Nuser M, Shatnawi A. Android malware detection using time-aware machine learning approach. Cluster Comput. 2024;27:12627–48. doi:10.1007/s10586-024-04484-6.
9. Zhang X, Zhang M, Wang X, Huang S, Di F. Constructive robust steganography algorithm based on style transfer. Comput, Mater Continua. 2024;81(1):1433–48. doi:10.32604/cmc.2024.056742.
10. Akram A, Khan I, Rashid J, Saddique M, Idrees M, Ghadi YY, et al. Enhanced steganalysis for color images using curvelet features and support vector machine. Comput Mater Contin. 2024;78(1):1311–28. doi:10.32604/cmc.2023.040512.
11. Nagasankar T, Ankaryarkanni B. Performance analysis of edge detection algorithms on various image types. Indian J Sci Technol. 2016;9(21):1–7. doi:10.17485/ijst/2016/v9i21/95207.
12. Swain G. Very high capacity image steganography technique using quotient value differencing and LSB substitution. Arab J Sci Eng. 2019;44(4):2995–3004. doi:10.1007/s13369-018-3372-2.
13. El-Emam NN. New data-hiding algorithm based on adaptive neural networks with modified particle swarm optimization. Comput Secur. 2015;55:21–45. doi:10.1016/j.cose.2015.06.012.
14. Latika, Gulati Y. A comparative study and literature review of image steganography techniques. Int J Sci Technol Eng. 2015;1(10):238–41.
15. Belhamra MA, Souidi EM. Steganography over redundant residue number system codes. J Inf Secur Appl. 2020;51:102434. doi:10.1016/j.jisa.2019.102434.
16. Darabkh KA, Al-Dhamari AK, Jafar IF. A new steganographic algorithm based on multidirectional PVD and modified LSB. Inf Technol Control. 2017;46(1):16–36. doi:10.5755/j01.itc.46.1.15253.
17. Akbay K, Konyar MZ, Ilkın S, Sonda AS. Data hiding using shuffle algorithm and lsb method. In: 2018 26th Signal Processing and Communications Applications Conference (SIU). Izmir, Turkey; 2018. p. 1–4.
18. Lee C-F, Shen J-J, Ou-Yang T-Y. A high payload edge detection-based image steganography robust to rs-attack by using lsb substitution and pixel value differencing. In: Recent Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceeding of the Fourteenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing; 2018 Nov 26–28; Sendai, Japan; 2019. p. 26–8.
19. Chen W-J, Chang C-C, Le THN. High payload steganography mechanism using hybrid edge detector. Expert Syst Appl. 2010;37(4):3292–301. doi:10.1016/j.eswa.2009.09.050.
20. Sultana H, Kamal A, Hossain G, Kabir MA. A novel hybrid edge detection and LBP code-based robust image steganography method. Future Internet. 2023;15(3):108. doi:10.3390/fi15030108.
21. Sultana H, Kamal A. Image steganography system based on hybrid edge detector. In: 2021 24th International Conference on Computer and Information Technology (ICCIT). Dhaka, Bangladesh; 2021. p. 1–6.
22. Alsobeh A, Shatnawi A. Integrating data-driven security, model checking, and self-adaptation for IoT systems using bip components: a conceptual proposal model. In: International Conference on Advances in Computing Research; 2023 May 8–10; Orlando, FL, USA: Springer; 2023. p. 533–49.
23. Théophile I, De La Croix NJ, Ahmad T. Fuzzy logic-based steganographic scheme for high payload capacity with high imperceptibility. In: 2023 11th International Symposium on Digital Forensics and Security (ISDFS). Chattanooga, TN, USA; 2021. p. 1–6.
24. Zhang J, Jiang ZL, Li P, Yiu SM. Privacy-preserving multikey computing framework for encrypted data in the cloud. Inf Sci. 2021;575:217–30. doi:10.1016/j.ins.2021.06.017.
25. Yang T, Li Y, He J, Liu Z, Ren F, Wang T, et al. Secure and traceable multikey image retrieval in cloud-assisted internet of things. IEEE Internet Things J. 2024;11(24):40875–87. doi:10.1109/jiot.2024.3457017.
26. Chen L, Yang Y, Yang L, Xu C, Miao Y, Liu Z, et al. Efficient and secure content-based image retrieval in cloud-assisted internet of things. IEEE Internet Things J. 2025;12(5):6001–13. doi:10.1109/jiot.2024.3489957.

27. Elshare S, El-Emam NN. Modified multi-level steganography to enhance data security. Int J Commun Netw Inform Secur. 2018;10(3):509. doi:10.17762/ijcnis.v10i3.3614.
28. Hardan H, Alawneh A, El-Emam NN. New deep data hiding and extraction algorithm using multi-channel with multi-level to improve data security and payload capacity. PeerJ Comput Sci. 2022;8:e1115. doi:10.7717/peerj-cs.1115.
29. Mohammad Imtiaz. Standard test images for image processing [Internet]; 2019. [cited 2025 Mar 17]. Available from: https://github.com/mohammadimtiazz/standard-test-images-for-Image-Processing
30. Malayil MV, Vedhanayagam M. A novel image scaling based reversible watermarking scheme for secure medical image transmission. ISA Trans. 2021;108:269–81. doi:10.1016/j.isatra.2020.08.019.
31. Yang Z, Yang H, Chang C-C, Huang Y, Chang C-C. Real-time steganalysis for streaming media based on multi-channel convolutional sliding windows. Knowl Based Syst. 2022;237:107561. doi:10.1016/j.knosys.2021.107561.
32. Lin W-B, Lai T-H, Chou C-L. Chi-square-based steganalysis method against modified pixel-value differencing steganography. Arab J Sci Eng. 2021;46(9):8525–33. doi:10.1007/s13369-021-05554-2.
33. Mo L, Zhu L, Ma J, Wang D, Wang H. MDRSteg: large-capacity image steganography based on multi-scale dilated ResNet and combined chi-square distance loss. J Electron Imaging. 2021;30(1):013018. doi:10.1117/1.jei.30.1.013018.
34. Alghamdi Y, Munir A, Ahmad J. A lightweight image encryption algorithm based on chaotic map and random substitution. Entropy. 2022;24(10):1344. doi:10.3390/e24101344.
35. Chatterjee A, Pati SK. Data hiding with digital authentication in spatial domain image steganography. In: Computational Intelligence in Pattern Recognition: Proceedings of CIPR 2019. Cham, Switzerland: Springer; 2020. p. 897–907.