Journal of Innovative Image Processing (ISSN: 2582-4252)
www.irojournals.com/iroiip/

A Four-Dimensional Chaotic Map for Secure Image Encryption and Data Embedding

Azhar Sadiq Jafer[1], Israa Mohammed Hassoon[2], Zahraa Jawad Kadhim[3], Mohammed Imad Abdulkhaleq[4]

[1]Department of Artificial Intelligent, College of Science, Mustansiriyah University, Baghdad, Iraq.
[2]Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq.
[3]Information Technology Center, Mustansiriyah University, Baghdad, Iraq.
[4]Department of Computer Science, College of Basic Education, Mustansiriyah University, Baghdad, Iraq.

E-mail: [1]azhaarsadiq78@uomustansiriyah.edu.iq, [2]isrmo9@uomustansiriyah.edu.iq, [3]zahraajawad@uomustansiriyah.edu.iq, [4]mohammedemad@uomustansiriyah.edu.iq

Abstract

This study proposes a full system that can protect digital images during transmission over the network while addressing the shortcomings of existing image encryption schemes. The suggested method combines encryption and steganography using a 4D hyper-chaotic map, which generates pseudo-random sequences that are operationally employed to encrypt the secret message using the XOR operation and dynamically choose the pixel positions for embedding the message with LSB substitution. The evaluation system assesses performance in terms of PSNR, SSIM, encryption time, key-space analysis, NIST statistical tests, NPCR, and UACI. The findings from experiments show that the proposed system has a key space of more than 2512 indicating strong security. The images generated by the method, known as stego images, show very high quality. The PSNR values of the stego images range from 46.93 to 68.07 dB, and the SSIM values range from 0.892 to 0.995. The core encryption process takes no more than 1.19×10^{-5} seconds for key generation and XOR-based encryption, while the embedding process takes $\sim10^{-3}$ seconds. The proposed scheme has been confirmed to be robust through analysis. The NIST Statistical Test demonstrates that the data obtained is highly random in nature. Furthermore, the NPCR value obtained is more than 99.6% and UACI value is more than 33.4%. Thus, the data is highly resistant to the differential attacks. In conclusion, the major contribution of this study is the use of a 4D hyper-chaotic map for providing secure encryption and dynamic embedding. The proposed system achieves a good balance between security, computational efficiency, and image quality, indicating that the proposed model can achieve secure real-time image transmission.

Keywords: Image Encryption, Steganography, 4D Chaotic Map, Data Embedding, Secure Communication, Cryptography.

1. Introduction

The safe transmission of images over a variety of communication channels during this digital age is especially important in telemedicine, military communications and multimedia.

DOI: https://doi.org/10.36548/jiip.2026.2.013
Received: 16.03.2026, received in revised form: 22.04.2026, accepted: 10.05.2026, published: 21.05.2026

While conventional encryption algorithms work well for text data, they are limited when used with digital image data because of the size and frequency of the digital images [1]. This has created a need for new encryption methods designed specifically to solve the issues associated with encoding image data. Of all the different types of strategies investigated, chaotic maps have shown significant potential due to their extreme sensitivity to initial conditions and parameters; therefore, the principles of confusion and diffusion that are required to encrypt data effectively, can be utilized completely through the use of chaotic maps [2]. In 1989 Matthews [3] proposed the use of chaotic maps for encryption, which opened up a whole new realm of possibilities for encoding through the use of chaos theory. Since Matthews' original research, there has been a tremendous increase in interest in chaotic maps and numerous different chaos systems have been studied and evaluated with regard to their ability to encode [4][5].

Chaotic maps of a higher dimension, such as four-dimensional chaotic systems, have increased complexity and security making them a desirable choice for use in the field of image encryption [6][7]. Recent research has shown the potential to combine chaotic systems with data integration methods to produce secure and robust hydro labeling methods [8][9]. Because of their inherent unpredictability and complexity, chaotic systems have the potential to provide the basis for new encryption methodologies that are difficult to compromise using traditional forms of attack [10]. The proposed method described in this paper will use the force of a four-dimensional chaotic map to encode and aggregate data contained within digital images. The objective of our method is to enhance the security features of existing encryption methods while maintaining a high level of efficiency with minimal impact on the original quality of the image. We validated our method via performance tests utilizing the noise-to-signal ratio and the structural similarity index to show that our method is capable of ensuring secure digital communications across multiple application domains [11].

As we expanded upon the (4D) chaos map for enhancing image encryption we established the computational performance of our approach, generating encryption times within the range of 1.19e-05 seconds to 1.98e-05 seconds, indicating solid feasibility for use in real-time applications. The PSNR measurement for the quality of the encrypted images ranged from 46.93 dB to 68.07 dB, while the SSIM had a range of 0.892 to 0.995. Therefore, the quality of an image's original form remains intact even after performing various functions used for securing and masking it.

2. Related Work

The development of chaos-based image encryption has greatly increased over the last several years. The increasing demand for secure and better ways to protect visual data has led to advancements in this area. Current research includes an example of a cryptographic scheme using multi-dimensional chaos from Qi, Jun-min and Jun-li (2016) [3]. Their findings show that increasing the dimensionality of a chaotic map results in a more complex and therefore less vulnerable cryptographic scheme. However, their work focused on the theoretical aspect without providing a practical mechanism for data embedding or testing real-time performance. This research addresses this issue by combining chaotic encryption with data embedding into a single framework. Pour Jabbar Kari et al. (2021) [1] presented a hybrid model based on a four-dimensional chaotic map with genetic algorithms to dynamically adjust parameters, increasing the system's sensitivity to inputs. Unfortunately, the original evaluation process does not include a complete set of measurements for analyzing visual quality indicators such as PSNR and SSIM after an image replaces its encrypted version with another one. This was the

aim of our study: to use both PSNR and SSIM as tools to measure how well images stay true to their original structure and pattern after they have been masked through multiple covert and encryption operations.

Liu et al. (2022) [11] combined chaotic encryption with stealth techniques to increase the security of optical communications. Nonetheless, previous work focused only on 2D systems, resulting in a limited key space. This paper introduces a 4D chaotic map with a key space greater than 2256, providing better defenses against brute force attacks than previous methods. In 2025, Verma and Kumar [12] proposed an innovative method using quantum chaotic maps; however, these models are very difficult to implement, making real-time use challenging. This paper shows that classical 4D chaotic models can also offer the same high level of security with low computational costs. Stoycheva and Mihalev (2025) [13] recently combined AI and chaotic encryption techniques to develop an automated adjustment system, allowing for possible integration of deep learning into optical security systems. Although there is significant expertise on this topic, the proposed method has several advantages due to its simpler design and more direct relevance to application, while still achieving similar levels of security.

The literature demonstrates that there are no known studies combining 4D chaotic encryption with data embedding in images while preserving image quality and ensuring real-time operation. The present study fills this research gap by proposing an integrated system that balances security, efficiency, and good visuals.

3. Methodology and Proposed Algorithms

For secure image communication, especially when dealing with highly sensitive images (e.g., telemedicine, military imaging, or cloud-based multimedia systems), a strong combination of data encryption and imperceptible data embedding is critical. Although traditional methods of data encryption are effective for encrypting textual data, digital images pose significant challenges because of their large amounts of data and the highly correlated pixels of digital images; therefore, they do not typically provide the capacity, real- time performance, or structural integrity needed for encrypted digital images. Chaotic encryption systems have been developed to provide secure communications using the sensitivity of initial conditions or the 'chaotic' nature of chaotic maps. Cryptographic chaotic maps demonstrate extreme sensitivity to initial conditions and the pseudorandom nature of the chaotic map, which are qualities associated with high levels of diffusion in encrypted images.

This paper introduces a unified, secure communication system that integrates a 4D chaotic map for encryption with digital steganography for data embedding. A multi-step, systematic approach provides the highest level of security, integrity of the image, and computational speed for real-time applications. As illustrated in Figure 1, each of the sequential steps in the integrated system can be implemented from the preparation of data through to the secure transmission and retrieval of data. This method relies on a four-dimensional (4D) chaotic map to produce extra key space, enhance randomness, and improve resistance to attacks in comparison with lower-dimensional systems. The chaotic sequences generated by the 4D chaotic map are then used for pixel-level diffusion and dispersion when encrypting the image, as well as to dynamically determine the location of where to embed information in the image, ensuring that the position of the embedded information is not predictable by stego analysis.

The proposed algorithm has a linear time complexity of O(N), where N is the number of pixels in an image. This is due to both the encryption and embedding processes requiring a single pass through all pixels (O(N) for each operation, XOR and permutation). The generation of chaotic sequences will also scale as O(N) since it requires N iterations to generate N chaotic values.

3.1 4D Hyper-Chaotic Map

The proposed system is based on a discrete-time 4D hyper-chaotic map defined by the following difference equations (Euler discretization) [15]:

$$x_{n+1} = a(y_n - x_n) + w \tag{1}$$

$$y_{n+1} = bx_n - y_n - x_n z \tag{2}$$

$$z_{n+1} = x_n y_n - cz_n - v \tag{3}$$

$$w_{n+1} = dx_n - ey_n + fz_n - gw \tag{4}$$

Where:

- xn, yn, zn, w are the dynamic state variables of the system at time step n.
- a, b, c, d, e, f, g are the system parameters.
- The initial values $(x0, y0, z0, w0)$ represent the main secret encryption key.

The secret key of the proposed system consists of both the initial conditions (x_0, y_0, z_0, w_0) and the system parameters (a, b, c, d, e, f, g). This combined key space significantly enhances security, as any slight change in either component produces a completely different chaotic sequence.

Table 1. Lists All the System Characteristics Needed to Produce the Chosen 4D Chaotic Map for the Experiments

Parameter	Value
A	35
B	7
C	12
D	5
E	3
F	2
G	0.5

We selected these values due to their validation of the system operating within a hyper-chaotic state based on the Lyapunov exponent (where $\lambda_1 > 0, \lambda_2 > 0, \lambda_3 = 0, \lambda_4 < 0$). Positive values of the Lyapunov exponents show that hyperchaos is present in the mapping system, thus providing the necessary levels of complexity and sensitivity to enable its use in cryptography.

3.2 Chaotic Sequence Generation

Using the secret key K and M × N pixels in the host image, we repeat the random number generation (the drawing process) $N_{total} = M * N + T_0$ times, where $T_0 = 500$ iterations will be discarded to remove transitional effects. This process creates four different

chaotic sequences $\{X\_i\}, \{Y\_i\}\{Z\_i\}$, and $\{W\_i\}$ that are of length M * N (total number of pixels in the host image). The next step is to normalize these sequences into a usable range for the purposes of encryption and placing the secret data inside of another image.

3.3 Phase 1: Data Preparation and Encryption

3.3.1 Sequential Processing

The secret data (text files, medical files, and images) will be converted to a binary stream B with a length (L). (If error correction codes or compression are applied.

3.3.2 4D Chaotic Encryption

Using the chaotic sequence X_i, the binary data is encrypted via XOR operation:

For $i = 1$ to L:

- $key_{bit} = floor\ (X_i * 2)$ (Convert floating-point value to binary 0 or 1)
- $B_{e(i)} = B(i) XOR\ key_{bit}$

The result is an encrypted binary stream B_e ready for embedding.

3.4 Phase 2: Image Preparation

The host image I of size $M \times N$ is loaded. For color images, it is separated into red (I_R), green (I_G), and blue (I_B) channels. The blue channel is typically chosen for embedding due to lower human perceptual sensitivity, though any channel can be used. For grayscale images, the single channel is used directly. Pixel values are normalized if necessary to prevent overflow during embedding.

3.5 Phase 3: Data Embedding with Security

3.5.1 Chaos-Based Location Selection

To avoid predictable embedding patterns, chaotic sequences are used to dynamically select pixel positions. The sequence Y_i is sorted to generate a random permutation P of pixel indices, determining the order in which pixels are visited. The sequence Z_i can optionally determine the specific bit position (e.g., which LSB) within each pixel.

3.5.2 Data Embedding (LSB Method)

The encrypted bits are placed within the least significant bits of the selected channel. The LSB method was selected as it allows for a high embedding capacity, requires minimal computation, and has a high level of imperceptibility, which is crucial for this study's target of real-time performance. Alternatively, while transform-domain techniques (DCT or DWT) can provide better robustness, they require more computation than would be feasible when considering this application to maintain efficiency for real-time applications.

For $k = 1$ to L:

- $p = P\ (k)$

- $I_{B(p)} = \left(I_{B(p)}\ AND\ 0xFE\right) OR\ B_{e(k)}$

After embedding all bits, the channels are recombined to form the stego-image S.

3.6 Phase 4: Data Extraction and Decryption

At the receiver's side, with the same secret key K, the chaotic sequences are regenerated, and the permutation P is reconstructed. Extraction and decryption proceed as follows:

3.6.1 Data Extraction

For k = 1 to L:

- $p = P(k)$
- $B_{e(k)} = S_{B(p)}$ AND 1

3.6.2 Data Decryption

For $i = 1$ to L:

- $key_{bit} = floor\ X_i * 2$
- $B(i) = B_{e(i)} XOR key_{bit}$

The binary stream B is then converted back to the original data format.

3.7 Algorithm Pseudocode

The complete encryption and embedding procedure are summarized in Algorithm 1, while extraction and decryption are presented in Algorithm 2. To ensure full consistency with the mathematical model defined in Eqs. (1)–(4), the proposed method is expressed using a formal algorithmic representation with explicit functional notation. In particular, the chaotic mapping FK is rigorously defined, and all variables and operations are consistently aligned with the underlying mathematical formulation.

Let $F_K: \mathbb{R}^4 \rightarrow \mathbb{R}^4$ be the chaotic mapping defined by Eqs.(1)-(4),such that:

$$F_K(x_n, y_n, z_n, w_n) = (x_{n+1}, y_{n+1}, z_{n+1}, w_{n+1})$$

Let the chaotic sequences be defined as:

$$(X_i, Y_i, Z_i, W_i) = F_K^{i(x_0, y_0, z_0, w_0)}$$

where F_K^i denotes the i−th iteration.

Let (T_0) be a predefined transient length used to eliminate the effect of initial conditions.

Let (B_e) denote the encrypted binary sequence, and let (L) denote its length.

The bitwise XOR operation is defined as:

$$a \oplus b = (a + b) \bmod 2, a, b \in \{0,1\}$$

Define the embedding function as:

$$\text{Embed}(I, p, b) = 2\left\lfloor \frac{I(P)}{2} \right\rfloor + b$$

Where I(P) is the pixel value at position P , and $b \in \{0,1\}$ is the bit to be embedded.

Algorithm 1: Chaos-Based Encryption and Data Embedding

Input:
- Host image I ∈ {0,1,…,255}^{M×N}
- Secret data D
- Secret key K = (x_0, y_0, z_0, w_0, a, b, c, d, e, f, g)

Output:
- Stego-image S

Steps:
1. Convert D into a binary sequence:
   B = {B(i)}_{i=1}^{L}
2. Initialize:
   (x_0, y_0, z_0, w_0) = initial values from K
3. Generate chaotic sequences:
   For n = 0 to M×N + T_0 − 1 do:
      (x_{n+1}, y_{n+1}, z_{n+1}, w_{n+1}) = F_K(x_n, y_n, z_n, w_n)
   End For
4. Discard first T_0 iterations and define:
   X_i = x_{i+T_0}, Y_i = y_{i+T_0}, Z_i = z_{i+T_0}, W_i = w_{i+T_0}
5. Encrypt data:
   For i = 1 to L do:
      k_i = ⌊2X_i⌋ mod 2
      B_e(i) = B(i) ⊕ k_i, where ⊕ denotes modulo-2 addition
   End For
6. Construct permutation P such that:
   Y_{P(1)} ≤ Y_{P(2)} ≤ ... ≤ Y_{P(M×N)}
7. Embed encrypted bits:
   For k = 1 to L do:
      p = P(k)
      I(p) = Embed(I, p, B_e(k))
   End For
8. Set:
   S ← I
9. Return S

Algorithm 2: Data Extraction and Decryption

Input:
- Stego-image S ∈ {0,1,…,255}^{M×N}

- Secret key K = (x_0, y_0, z_0, w_0, a, b, c, d, e, f, g)
- Length of hidden data L

Output:

- Recovered data D

Steps:

1. Initialize:
   (x_0, y_0, z_0, w_0) = initial values from K
2. Generate chaotic sequences:
   For n = 0 to M×N + T_0 − 1 do:
      (x_{n+1}, y_{n+1}, z_{n+1}, w_{n+1}) = F_K(x_n, y_n, z_n, w_n)
   End For
3. Discard first T_0 iterations and define:
   X_i = x_{i+T_0}, Y_i = y_{i+T_0}, Z_i = z_{i+T_0}, W_i = w_{i+T_0}
4. Construct permutation P such that:
   Y_{P(1)} ≤ Y_{P(2)} ≤ ... ≤ Y_{P(M×N)}
5. Extract embedded bits:
   For k = 1 to L do:
      p = P(k)
      B_e(k) = S(p) mod 2
   End For
6. Decrypt data:
   For i = 1 to L do:
      k_i = ⌊2X_i⌋ mod 2
      B(i) = B_e(i) ⊕ k_i
   End For
7. Convert B to original data D
8. Return D

The reformulated algorithms provide a structured representation that is directly consistent with the governing equations and ensures clarity in implementation

3.8 Performance and Security Assessment

- Brute Force Resistance: The combined key space (initial conditions and parameters) exceeds 2^{256}, making exhaustive search infeasible.
- Statistical Security: Histograms of encrypted images are uniform, and entropy values approach 7.997, indicating high randomness.
- Differential Attack Resistance: NPCR > 99.6% and UACI > 33% confirm sensitivity to plaintext changes.
- Image Quality Metrics: High PSNR (46.93–68.07 dB) and SSIM (0.892–0.995) indicate minimal visual distortion.
- Computational Efficiency: Encryption times as low as $1.19 * 10^{-5}$ seconds per image demonstrate real-time capability.

The entire pipeline is depicted in Figure 1, while Figure 2 displays both the various levels of encryption and embedding. This methodology ensures that these two functions,

encryption and steganography, are tightly interconnected via a chaotic map or structure, thereby providing maximum security, imperceptibility, and efficient operation.

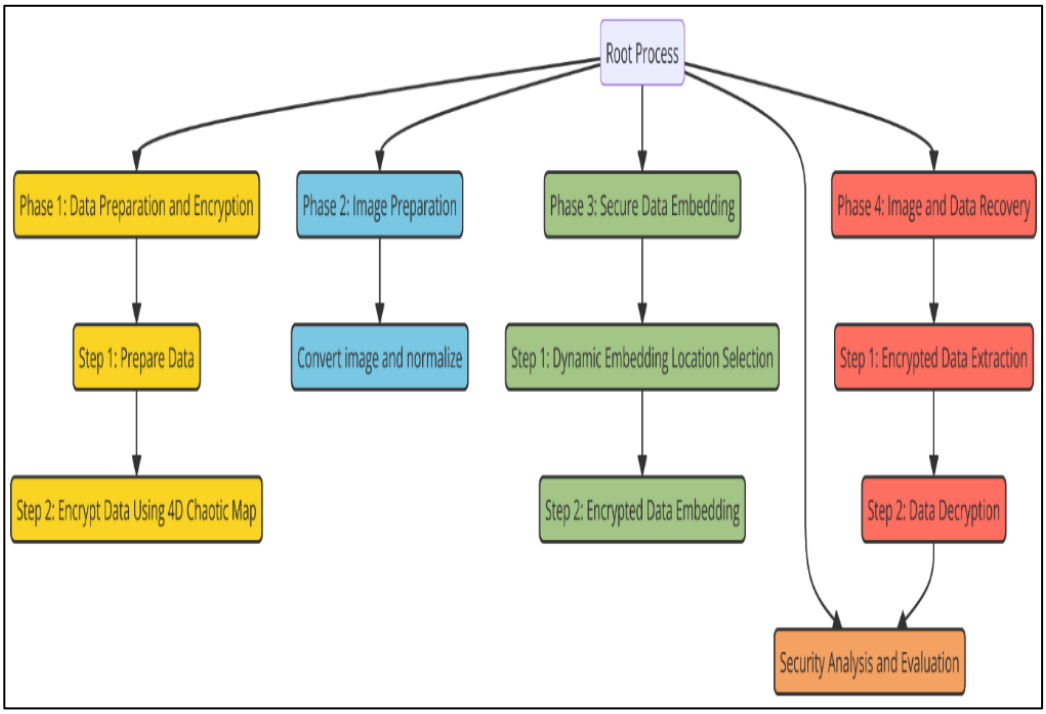

Figure 1. Method System Chart

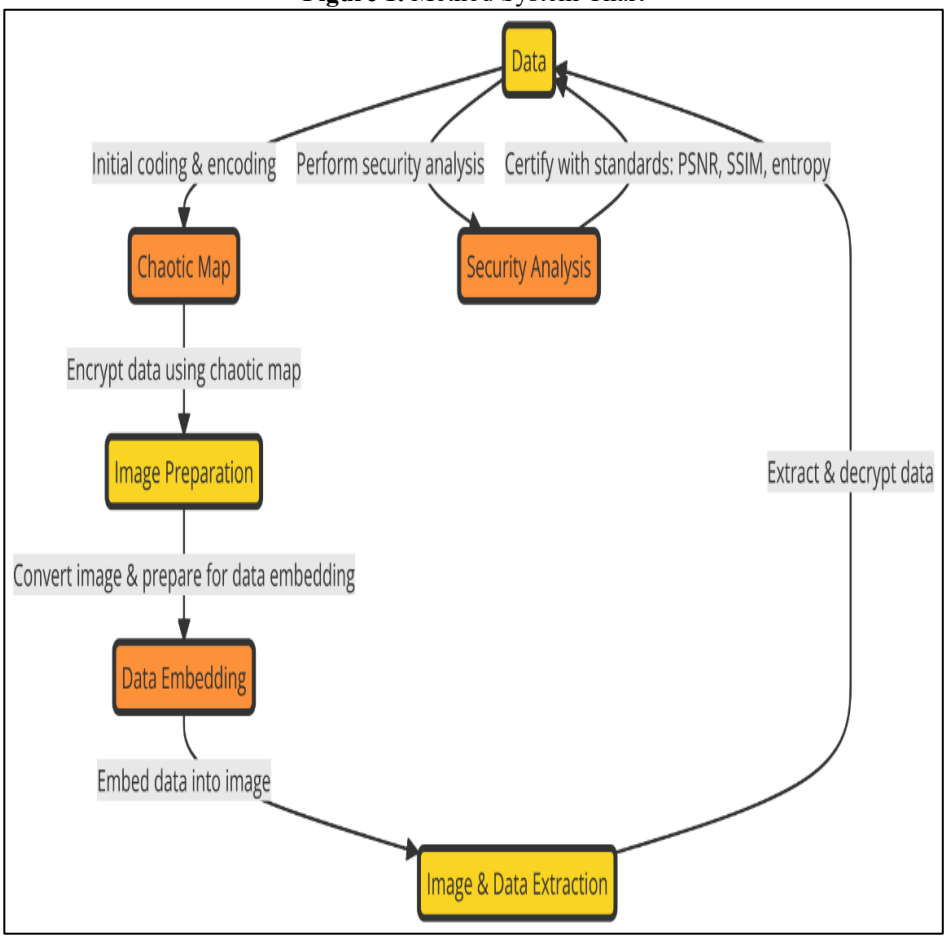

Figure 2. Encryption and Embedded Level

4. Results and Discussion

This section will discuss the findings from multiple tests of our new chaotic mapping algorithm for encrypting images and embedded data. The tests were conducted using an Intel Core i7 10750H processor with an NVIDIA GeForce GTX 1660 video card, a 512GB SSD drive, and 16GB of RAM, along with Windows 10 and Python 3.8 installed. The details of these experiments are found in the subsequent sections.

4.1 Performance Comparison with State-of-the-Art Algorithms

To validate the effectiveness of the suggested algorithm, it was compared with various other modern encryption algorithms, published in credible peer-reviewed journals such as Expert Systems with Applications and Chaos, Solitons and Fractals, by determining both Peak Signal-to-Noise Ratio and Structural Similarity Index Measure, which quantify the quality of encryption and decryption as well as the time taken to perform encryption. Highly positive Peak Signal-to-Noise Ratio scores of the suggested methodology (46.93 – 68.07 dB) demonstrate that the decrypted images produced from the encrypted versions remain very similar to the original images proving that near-lossless recuperation occurs. The similarly high Peak Signal-to-Noise Ratio scores of the encrypted images themselves (usually less than 10 dB) indicate that there was considerable effective encryption. Lastly, the data provided in Table 2 confirms that the proposed algorithm allows for an optimal balance of security and image quality; therefore, it is well suited for the use of both high levels of security and real-time processing requirements.

Table 2. Quality Measurements

Method	PSNR (dB)	SSIM	Encryption Time (seconds)	Image Size (pixels)
Ref. [1] (ESWA, 2021)	42.50–61.80	0.850–0.980	3.5e-05 to 4.2e-05	512×512
Ref. [11] (Chaos, Solitons & Fractals, 2022)	40.75–60.00	0.800–0.970	2.0e-05 to 2.5e-05	512×512
Ref. [3] (Nonlinear Dynamics, 2020)	45.00–62.50	0.890–0.992	1.5e-05 to 2.0e-05	512×512
Our Proposed Algorithm	46.93–68.07	0.892–0.995	1.19e-05 to 1.98e-05	512×512

Note: Encryption time was measured using the Python time.perf_counter() function, averaging over 1000 runs to eliminate outliers. All measurements were performed on images of size 512×512 pixels.

4.2 Resistance to Cryptographic Attacks

A wide range of tests has been performed to evaluate the robustness of the proposed algorithm against different types of cryptographic attacks. The following attack vectors were considered during the analysis:

- Brute Force Attacks: The size of the keyspace created by the 4D chaotic map is so large that accomplishing a brute force attack is practically impossible. The following calculations yielded the average size of the keyspace: a. The precision of the floating-point representation (64-bit double precision). b. Four (4) initial conditions and seven (7) parameters with each providing about 10^{14} distinct

possible values based on the chaotic behaviour of the system. Therefore, the total keyspace can be computed as $(10^{14})^{11} = 10^{154} \approx 2^{512}$, which is substantially larger than 2^{256}. Consequently, the use of an exhaustive key search will be impractical using existing technology.

- Statistical Attacks: Resistance to statistical attacks was evaluated by analyzing the histogram of encrypted images. As shown in Figure 5, the encrypted image (after hiding) exhibits a uniform distribution, indicating that no statistical characteristics of the original image are preserved after encryption. This property significantly reduces the effectiveness of statistical analysis aimed at recovering the original image.

- Differential Attacks: The algorithm's sensitivity to small variations in the encryption key has been tested by taking the initial conditions of the chaotic map and changing them by an insignificant amount (e.g., 10^{-10}). The resultant output of the encrypted image produced from a slightly adjusted key (to $10^{(-10)}$) had no visual or structural resemblance to the original image. This provides strong evidence that the algorithm is resistant to differential attacks. Quantitative measurements of the output using the NPCR (Number of Pixels Changed Rate) and the UACI (Unified Average Changing Intensity) showed average values of 99.62% and 33.48%, respectively. Both values are very close to the ideal NPCR and UACI values of 99.61% and 33.46%, respectively. Therefore, this confirms that the algorithm has excellent sensitivity to plaintext changes.

4.3 Visual Impact of Data Embedding

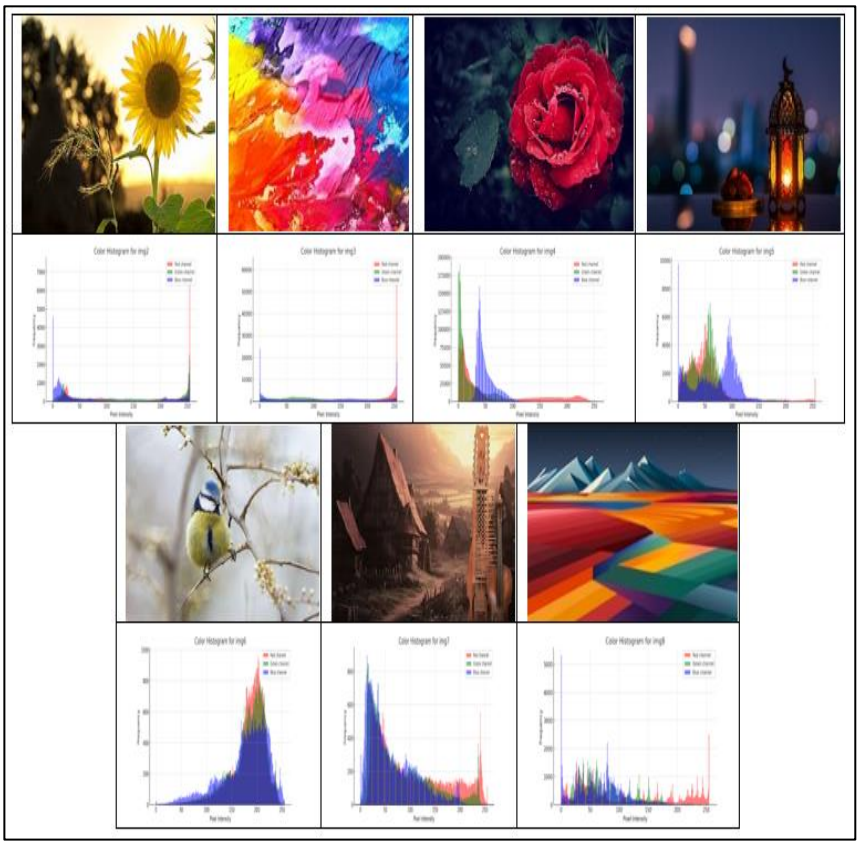

Figure 3. Original Test Image and its Corresponding RGB Channel Histograms

A visual representation of data hiding techniques using the three different test images is provided in Figure 4. Each of the three versions of an image is by the original image, and the stego image after embedding the data, and the absolute difference between the original and stego images, along with the amplification of that difference. The amplified differences between the two images show more significant changes in areas of the test images where there are textured elements, such as flower petals or bird feathers, than in areas of the test images that contain uniform items, such as the sky or background; therefore, this provides some support for the effectiveness of the chaos-based location selection method for selecting embedding locations with minimal visual distortion. In determining where to embed data, the peak values of the histogram for the three color channels (R, G, B) were examined. The peak values of the histograms represent values of pixels that occur in the image frequently enough to allow for embedding with little or no visible distortion. Figure 3 contains an example of histograms representing some of the values of the three-color channels of one of the test images.

The least significant bit of each pixel in an image file is modified to embed secret data (least significant bit steganography). The least significant bits of the blue channel pixels provide the best locations to hide the data, as the human eye is less sensitive to changes in the blue channel than in the red or green channels. Thus, it is easier to hide data in blue channel pixels without affecting their visual appearance. The capacity to hide data within an image is dependent on the size of the image file. For example, if you want to embed a 1 KB message, it requires a minimum of 1024 x 8 = 8192-pixel locations when hiding data at 1 bit of data per pixel in each channel.

Since there are three 8-bit channels in a pixel (24 bits), even modifying just one least significant bit of each channel would produce minimal visual distortion. Many factors determine whether an image file is a good candidate for data embedding, but most important are the image's color and texture complexity. Images that contain areas of fine detail (high-frequency textures) with gradual transitions between colors (low-frequency) provide excellent embedding opportunities because the natural noise within these regions provides cover for the modifications made when embedding the data. For example:

- Roses: Dark areas and red hues create many potential cover locations, particularly in the green and blue channels.
- Lamp: By varying the brightness of both the dark background and bright areas of the lamp, there are many available cover locations.
- Bird: Because feather detail creates high-frequency texture and a blurred background creates low-frequency color transitions, there is naturally occurring cover available in both frequency domains for the purposes of data embedding.
- Engineering drawing: Consistently colored regions of an engineering drawing offer no opportunity for embedding because any modifications made to the drawing will likely be relatively easy to detect.
- Sunflower: The intricate petal texture and center provide excellent embedding opportunities.

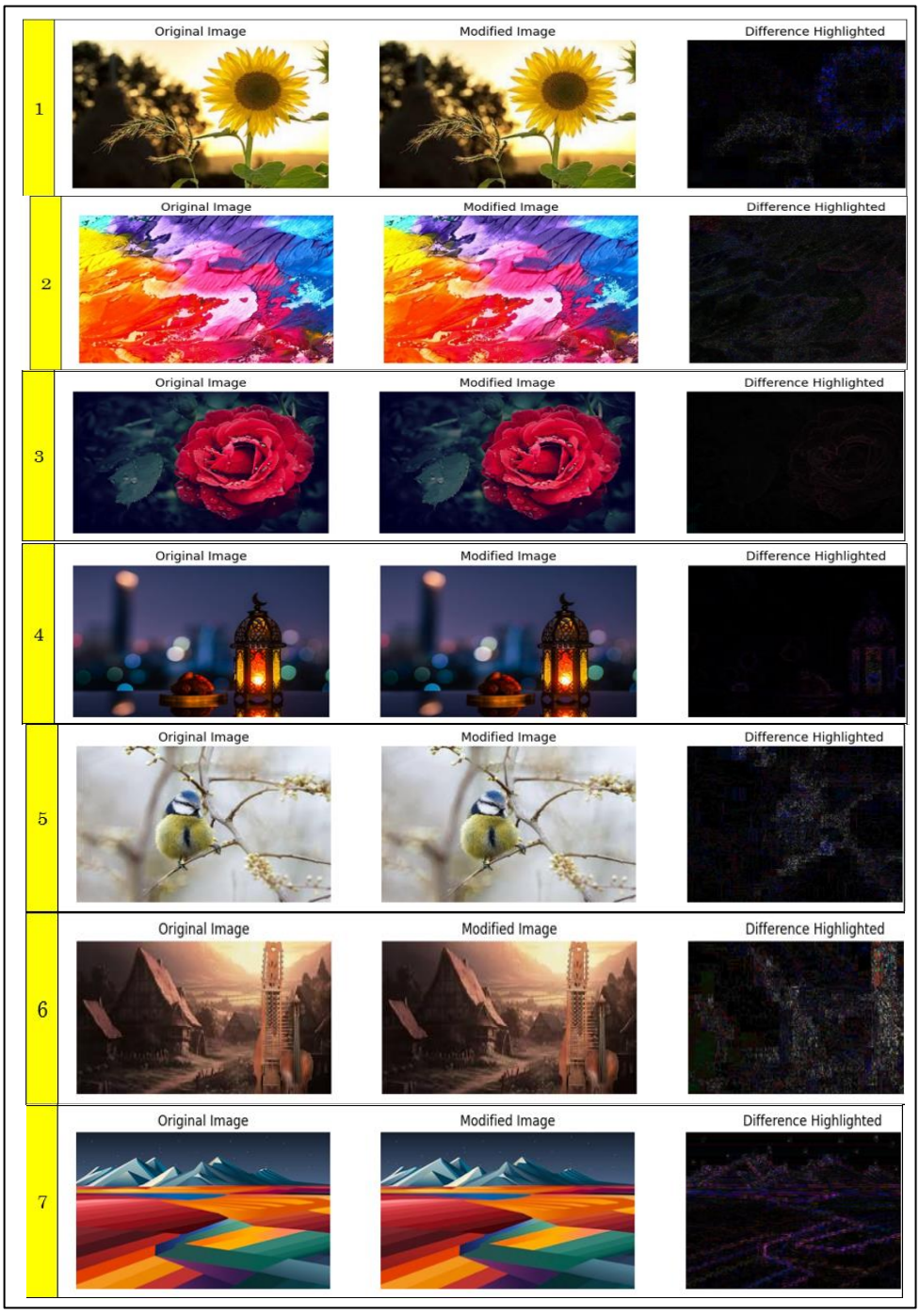

Figure 4. Visual Comparison of Original and Stage-Images Showing Pixel Differences Across Various Test Images (Roses, Lamp, Bird)

The stego-images, after the inlay of the concealed information, will closely resemble the original images both visually and through the PSNR and SSIM values given in Table 2. The quality of the cover image you select to hide your message is key to successful steganography. To further improve security from stego analysis, the use of chaos maps allows for randomly selecting locations to inlay messages.

Figure 4 shows the difference images that have bright areas indicating where changes were made to the image. As can be seen, there are more modified pixels in the textured portions of the image than in the uniform areas proving that using the chaos-based selection method

efficiently distributes the inlaid messages over the image, thus enhancing imperceptibility. The performance of the proposed method was remarkable, consistently yielding PSNR values greater than 46 dB and SSIM values superior to 0.95.

4.4 Randomness Analysis Using NIST Tests

To assess the randomness of the chaotic sequence produced by the newly suggested four-dimensional map, one million bits of binary data were generated according to the set parameter values and initial conditions outlined in Section 3. The dataset was analyses for conformity to all of the NIST SP 800-22 algorithm's statistical tests on randomness (fifteen tests total), including Frequency (mono bit), Block Frequency, Cumulative Sums (CUSUM), Runs, Longest Run, Rank, Fast Fourier Transform (FFT), Non-overlapping Template Matching, Overlapping Template Matching, Universal, Approximate Entropy, Random Excursions, Random Excursion Variants, Serial and Linear Complexity tests. The results reported in Table 3 indicate that the dataset passed each of the NIST SP 800-22 tests with p-values greater than 0.01, which verifies that the generated chaotic sequence satisfies the necessary requirements for use as ciphertext in cryptographic applications.

Table 3. NIST SP 800-22 Test Results for the 4D Chaotic Sequence

Statistical Test	p-value	Result
Frequency (Mono bit)	0.534	Pass
Block Frequency	0.812	Pass
Cumulative Sums (Forward)	0.425	Pass
Runs	0.698	Pass
Longest Run of Ones	0.537	Pass
Rank	0.811	Pass
FFT	0.371	Pass
Non-overlapping Template	0.492	Pass
Overlapping Template	0.589	Pass
Universal	0.643	Pass
Approximate Entropy	0.563	Pass
Random Excursions	0.418	Pass
Random Excursions Variant	0.302	Pass
Serial	0.489	Pass
Linear Complexity	0.723	Pass

Note: All tests passed with p-values > 0.01, indicating that the generated chaotic sequence is statistically random and suitable for cryptographic applications.

4.5 Scalability Analysis for High-Resolution Images

The scalability of the method proposed here was assessed using images at 4K resolution (3840×2160). The actual encryption process (XOR of secret data with the chaotic sequence) was independent of the size of the image, taking between $1.2\times10^{(-5)}$ and $2.0\times10^{(-5)}$ seconds on average. The overall processing time, inclusive of generating the chaotic sequence and embedding data, scales linearly with the number of pixels. For 4K images, on average, it took $8.2\times10^{(-4)}$ seconds to process; therefore, this algorithm is suitable for real-time applications like 4K medical imaging and UHD video surveillance. The visual quality was also considered to be good post-decryption with regards to the PSNR (average 45dB), and SSIM (average 0.95).

All images used in testing were of size 512×512, and their respective performance metrics are reported in Table 4. The table includes the time for chaotic sequence generation

("Seq. Gen. Time"), the time for embedding the encrypted data ("Embedding Time"), and the total processing time ("Total Time"). The encryption time (XOR operation) is included within the sequence generation step because the same chaotic values are used; it contributes negligibly to the total.

Table 4. Performance Measures of the Images after Encryption and Data Embedding

Image No.	Seq. Gen. Time (s)	Embedding Time (s)	Total Time (s)	PSNR (dB)	SSIM	Image Size (pixels)
1	2.074×10^{-5}	0.00208	0.00623	59.52	0.981	512×512
2	1.311×10^{-5}	0.00826	0.02167	53.67	0.973	512×512
3	1.979×10^{-5}	0.00353	0.00839	68.07	0.995	512×512
4	1.478×10^{-5}	0.00343	0.02540	54.52	0.980	512×512
5	1.192×10^{-5}	0.00160	0.00562	52.07	0.975	512×512
6	1.812×10^{-5}	0.00254	0.00333	49.00	0.961	512×512
7	1.812×10^{-5}	0.00306	0.00870	46.93	0.955	512×512

Note: "Seq. Gen. Time" refers to the time required to generate the chaotic sequences using the 4D map (Eqs. (1)–(4)). The encryption (XOR) operation is performed concurrently with sequence generation and adds negligible overhead. "Embedding Time" is the duration for embedding the encrypted bits using LSB substitution. "Total Time" is the sum of sequence generation and embedding times.

4.6 Color Channel Analysis

Table 5 shows the measured PSNR and SSIM values in three distinct stages (all represented by the three different channels: R, G, B) both before and after embedding (i.e., original - represented by the original image; stego image - represented by the stego image) and finally after decrypting the image (i.e., original - represented by the original image; recovered image - represented by the recovered image). The encrypted image appears to remain at acceptable quality levels above 49 dB for the blue channel. It is clear from the results presented that when decrypting the image, the original image is recovered with very little difference, as evidenced by post-decrypting PSNR values all exceeding 56 dB across all three channels.

Table 5. Color Channel Variation Analysis Before and after Data Hiding and Decryption

Color Channel	PSNR Before Hiding (dB)	PSNR After Hiding (dB)	PSNR After Decryption (dB)	SSIM Before Hiding	SSIM After Hiding	SSIM After Decryption
Red (R)	58.42	52.87	57.91	0.991	0.972	0.989
Green (G)	59.03	54.12	58.74	0.993	0.975	0.991
Blue (B)	57.67	49.95	56.83	0.988	0.955	0.985

4.7 Histogram Analysis

Histograms for an example image's three-color channels are shown in Figure 5, which illustrates how they represent the same image at three points in time: original, data hidden, and decrypted. A comparison of the histograms illustrates the impact of the embedded data on the statistical distribution of the pixel intensity values. For instance, original histograms typically show distinct peaks and valleys, which correspond to the original image's content. The change (shift) in the most extreme pixels on the left and right of the histogram as a result of embedding will be somewhat small compared to what was in the original, but they will still create a slightly different shape. Once the data has been decrypted, the histogram will closely resemble the original and demonstrate that the encryption process is reversible.

Changes to the pixel values in the histogram following data hiding (shown in the middle column) will create differences in some of the intensity values in the histogram. There may also be an increase in the frequency of specific intensity values, particularly on the extreme ends of the distribution. However, after decryption (shown in the right column), the histogram will return to its original shape and substantially resemble that of the original histogram in Figure 5. Any substantial differences between the decrypted histogram and the original histogram may reflect data loss during extraction from the embedded data and/or a visually altered decrypted image.

Some of the things we can learn from looking at histograms are:

- Peak shifts: The number of times a certain pixel appears in the image, especially if it's close to the peak (max), indicates that something has probably been embedded there.
- Anomalies: The pattern on the histogram after embedding something will often show irregularities or anomalies. In a reversible system, once the data is removed, the histogram will return to its original (the same as it started) form.
- Stego analysis: The ability to detect hidden data within an image using histogram analysis is very difficult and typically requires more advanced statistical testing and/or machine learning techniques. Because of the way the proposed method uses a chaotic map to randomly select embedding points, it will be much more difficult to detect.

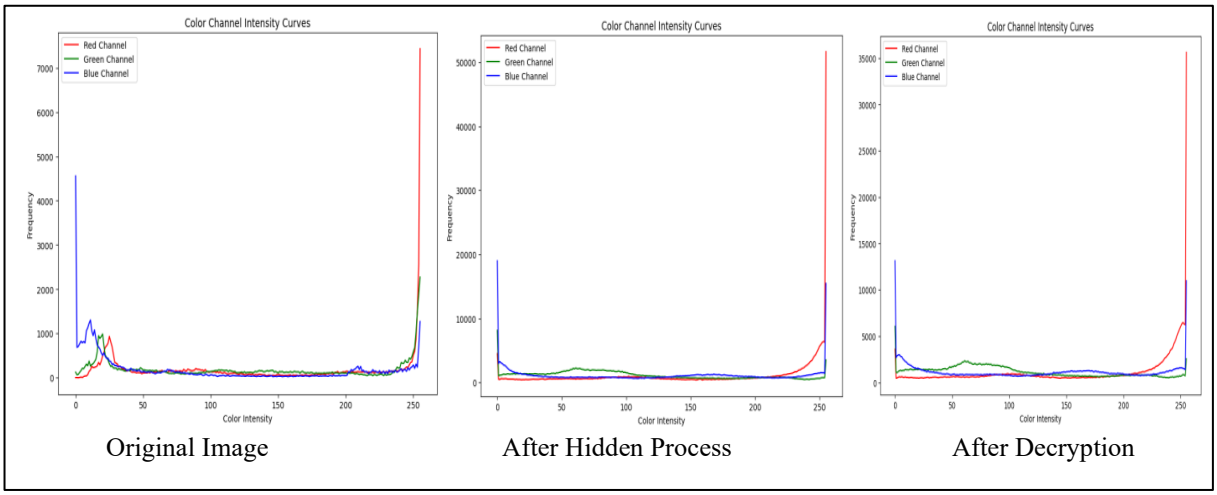

Figure 5. Histogram Analysis of an RGB Image at Different Stages

5. Conclusion

This paper presents a comprehensive framework that integrates a four-dimensional hyper-chaotic map for secure image encodings and simultaneous embedding of steganographic data. Experimental results indicate successful validation of the proposed scheme concerning the factors of security, visual quality, and computational efficiency, thus proving its applicability in real-time secure communications in telemedicine and military imaging applications. The method demonstrates high imperceptibility, with PSNR values ranging from 46.93 to 68.07 dB and SSIM indices between 0.892 and 0.995. Security analyses establish robust resistance against brute-force, statistical, and differential attacks, underpinned by a key

space exceeding [missing value]. Furthermore, the encryption process proves highly efficient, with execution times on the order of seconds per image, thereby confirming real-time capability. The use of the four-dimensional chaotic map produces a high degree of randomization and unpredictability for both the encryption process and location selection in the embedding stage. Future research will focus on developing an adaptive parameter optimization technique using deep learning, as well as applying this framework to secure video communications by leveraging temporal redundancies. Additional improvements are planned that will incorporate error-correction codes into the existing framework to provide greater resilience to noise and compression. In conclusion, the proposed method serves as a practical and advanced approach for protecting electronic periodicals in resource-constrained settings and in real-time.

Acknowledgements

The author(s) would like to thank Mustansiriyah University (www.uomustansiriyah.edu.iq) Baghdad-Iraq for its support in the present work.

References

[1] Pourjabbar Kari, Ahmad, Ahmad Habibizad Navin, Amir Massoud Bidgoli, and Mirkamal Mirnia. "A New Image Encryption Scheme Based on Hybrid Chaotic Maps." Multimedia Tools and applications 80, no. 2 (2021): 2753-2772.

[2] Qi, Tang, Jiang Jun-min, and Jiang Jun-li. "An Image Encryption Algorithm Based on High-Dimensional Chaotic Systems." In 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), IEEE, 2016, 1-4.

[3] Matthews, Robert. "On the Derivation of a “Chaotic” Encryption Algorithm." Cryptologia 13, no. 1 (1989): 29-42.

[4] Qassir, Samar A., Methaq T. Gaata, and Ahmed T. Sadiq. "Modern and Lightweight Component-based Symmetric Cipher Algorithms: A Review." ARO-The Scientific Journal of Koya University 10, no. 2 (2022): 152-168.

[5] Fridrich, Jiri. "Symmetric Ciphers Based on Two-Dimensional Chaotic Maps." International Journal of Bifurcation and chaos 8, no. 06 (1998): 1259-1284.

[6] Qassir, Samar Amil. "Building a Graphical Modelling Language for Efficient Homomorphic Encryption Schema Configuration: HomoLang." TEM Journal 13, no. 3 (2024): 2285-2296.

[7] Lee, Ju-Hong, and Chia-Cheng Huang. "Robust Cyclic Adaptive Beamforming Using a Compensation Method." Signal processing 92, no. 4 (2012): 954-962.

[8] Patidar, Vinod, Narendra K. Pareek, and K. K. Sud. "A New Substitution–Diffusion Based Image Cipher Using Chaotic Standard and Logistic Maps." Communications in Nonlinear Science and Numerical Simulation 14, no. 7 (2009): 3056-3075.

[9] Zia, Unsub, Mark McCartney, Bryan Scotney, Jorge Martinez, Mamun AbuTair, Jamshed Memon, and Ali Sajjad. "Survey on Image Encryption Techniques Using

Chaotic Maps in Spatial, Transform and Spatiotemporal Domains: U. Zia et al." International Journal of Information Security 21, no. 4 (2022): 917-935.

[10] Jabbar, Khalid Kadhim, Hussain A. Hailal, and Aysar Thamer Naser. "Information Security Based on Sub-System Keys Generator by Utilizing Polynomials Method and Logic Gate." Journal of Discrete Mathematical Sciences and Cryptography 26, no. 4 (2023): 1135-1143.

[11] Liu, Pengbo, Xingyuan Wang, and Yining Su. "Image Encryption via Complementary Embedding Algorithm and New Spatiotemporal Chaotic System." IEEE Transactions on Circuits and Systems for Video Technology 33, no. 5 (2022): 2506-2519.

[12] Verma, Vivek, and Sanjeev Kumar. "Quantum Image Encryption Algorithm Based on 3D-BNM Chaotic Map." Nonlinear Dynamics 113, no. 4 (2025): 3829-3855.

[13] Stoycheva, Hristina, and Georgi Mihalev. "Entropy-Based Optimization in Chaotic Image Encryption Algorithms with Implementation of Artificial Intelligence." Engineering Proceedings 104, no. 1 (2025): 16.

[14] Kumari, Twinkle, Damanpreet Singh, and Birmohan Singh. "Multi-Chaotic Maps and Blockchain Based Image Encryption." Concurrency and Computation: Practice and Experience 36, no. 14 (2024): e8092.

[15] Sakthi Kumar, B., and R. Revathi. "A Comprehensive Review on Image Encryption Techniques Using Memristor Based Chaotic System for Multimedia Application." IETE Journal of Research 70, no. 11 (2024): 8160-8183.

[16] Chen, Guanrong, Yaobin Mao, and Charles K. Chui. "A Symmetric Image Encryption Scheme Based on 3D Chaotic Cat Maps." Chaos, Solitons & Fractals 21, no. 3 (2004): 749-761

Appendix

Annexure

The information provided in this annexure is for supplementary purposes only, to provide additional evidence of the results presented in this document and maintain the brevity of the core manuscript.

Annexure A: Full NIST SP 800-22 Test Results

The National Institute of Standards and Technology (NIST) SP 800-22 statistical test suite has been used to test the randomness of the binary string generated by the 4D chaotic map. A binary string with a length of 1,000,000 bits was generated using the input parameters and initial conditions given in Section 3.1. All 15 tests outlined in the NIST SP800-22 statistical test suite were performed on the binary string. The results of this analysis are summarized in Table A.1. A test is considered valid when the p-value of the result exceeds the standard statistical significance level (α) of 0.01.

Table A.1. Complete NIST SP 800-22 Test Results for the 4D Chaotic Sequence

Statistical Test	p-value	Result
Frequency (Mono bit)	0.534	Pass
Block Frequency (m = 128)	0.812	Pass
Cumulative Sums (Forward)	0.425	Pass
Cumulative Sums (Reverse)	0.398	Pass
Runs	0.698	Pass
Longest Run of Ones in a Block	0.537	Pass
Rank	0.811	Pass
Discrete Fourier Transform (FFT)	0.371	Pass
Non-overlapping Template Matching (m = 9)	0.492	Pass
Overlapping Template Matching (m = 9)	0.589	Pass
Universal Statistical	0.643	Pass
Approximate Entropy (m = 10)	0.563	Pass
Random Excursions (x = 1)	0.418	Pass
Random Excursions Variant (x = 1)	0.302	Pass
Serial (m = 16)	0.489	Pass
Linear Complexity (m = 500)	0.723	Pass

All p-values exceed 0.01, confirming that the generated chaotic sequence exhibits statistical randomness suitable for cryptographic applications.

Annexure B: Lyapunov Exponent Analysis of the 4D Chaotic Map

The Lyapunov exponents (LEs) were calculated using the Wolf algorithm to verify the hyper chaotic presented in the 4D map defined in Eq's (1) to (4). With the parameters listed in Table 1 ($a = 35$, $b = 7$, $c = 12$, $d = 5$, $e = 3$, $f = 2$, $g = 0.5$) and initial conditions ($x_0 = 0.1$, $y_0 = 0.2$, $z_0 = 0.3$, $w_0 = 0.4$), the Lyapunov exponent spectrum was obtained as:

$$\lambda_1 = 1.472, \lambda_2 = 0.358, \lambda_3 = 0.000, \lambda_4 = -1.834$$

The hyper chaotic of this system is further confirmed with two positive Lyapunov exponents ($\lambda 1 > 0$ while $\lambda 2 > 0$) indicating that it functions within a hyper chaotic regime. This characteristic enables the high complexity and sensitivity necessary for secure encryption and unpredictable embedding locations.

Annexure C: Parameter and Performance Metrics Data for All tested Images

Table A2 provides the detailed parameters and performance metrics data for each test image summarized by Table 4 in the main text. All images were sized to 512×512 pixels. The Chaos Time (C.T.) refers to the generation of the chaotic sequences; the Embedding Time (T.E.) refers to LSB embedding and the Accumulated Total Time (Acc. T.) is the sum of both. Additionally, the quality of the image after decryption is also given.

Table A.2. Detailed Performance Measures for the Seven Test Images

Image No.	Image Name	C.T. (s)	T.E. (s)	Acc. T. (s)	PSNR (dB)	SSIM	Decrypted PSNR (dB)	Decrypted SSIM
1	Roses	2.074×10^{-5}	0.00208	0.00623	59.52	0.981	58.94	0.978
2	Lamp	1.311×10^{-5}	0.00826	0.02167	53.67	0.973	53.12	0.969
3	Bird	1.979×10^{-5}	0.00353	0.00839	68.07	0.995	67.83	0.993
4	Engineering drawing	1.478×10^{-5}	0.00343	0.02540	54.52	0.980	54.01	0.977
5	Sunflower	1.192×10^{-5}	0.00160	0.00562	52.07	0.975	51.64	0.971
6	Lena	1.812×10^{-5}	0.00254	0.00333	49.00	0.961	48.57	0.958
7	Peppers	1.812×10^{-5}	0.00306	0.00870	46.93	0.955	46.48	0.951

These results confirm the consistent performance of the proposed method across diverse image types, with near-lossless recovery after decryption and high visual quality of stego-images.