Electronic Research Archive

ERA, 33(11): 7172–7197.
DOI: 10.3934/era.2025317
Received: 02 October 2025
Revised: 02 November 2025
Accepted: 05 November 2025
Published: 26 November 2025

http://www.aimspress.com/journal/era

Research article

Bit-level image encryption algorithm based on chaotic systems and DNA encoding

Tingting Liu[1], Dongsheng Cheng[2] and An Song[3,*]

[1] School of Computer and Software, Shenzhen University of Information Technology, Shenzhen 518000, China
[2] School of Artificial Intelligence, Shenzhen University of Information Technology, Shenzhen 518000, China
[3] School of Information Technology, Bailie Vocational College, Zhangye 734000, China

* **Correspondence:** Email: saplmath@163.com.

Abstract: The image encryption technology is an important means to ensure network digital data transmission security. To improve the security and encryption computational efficiency, we proposed a novel bit-level image encryption algorithm that integrates hyper-chaotic systems with DNA encoding operations. First, we generated high-quality pseudo-random sequences through K-means clustering, which were applied to hyper-chaotic sequences; second, we implemented multi-level scrambling at both bit-plane and bit-level; finally, we employed dynamic DNA encoding rules combined with bidirectional diffusion. Experimental results demonstrated that the proposed algorithm achieves a large key space (approximately 2^{256}), high information entropy (close to 7.9899), and strong resistance to differential attacks (NPCR = 99.61%, UACI ≈ 33.46%). Our results also showed that the histograms of the encrypted images are even, and the correlation coefficients approach 0. Moreover, the proposed algorithm can effectively resist the chosen plaintext attack, brute force attack, clipping attack, salt and pepper noise attack, etc.

Keywords: image encryption; bit-level encryption; security analysis; hyper-chaotic system; DNA encoding; K-means clustering

1. Introduction

Digital images play a crucial role in numerous aspects of our work and daily lives, serving as carriers of data and a means of sharing information. However, it is crucial to note that these digital images often contain sensitive and private information such as facial features and fingerprints. Moreover, they might also encapsulate valuable trade secrets or classified military information [1]. Unfortunately, due to the rapid growth of the Internet, big data, and cloud computing, digital images are constantly exposed to the threats of unauthorized access, interception, or decryption [2,3]. Consequently, the encryption of digital images has emerged as a significant area of research and development.

Due to the large data size and strong pixel correlation of images, most traditional encryption methods such as DES and IDEA are not suitable for real-time image encryption [4]. As chaotic systems are sensitive to initial values, unpredictable, and irreducible, the generated chaotic sequences are widely used in image encryption. Since Matthew [5] first proposed an efficient chaotic encryption algorithm in 1989, there are many studies on image encryption based on chaos that are proposed.

Chaotic systems are widely employed in image encryption due to their sensitivity to initial conditions, ergodicity, and pseudo-randomness. A key research direction involves designing chaotic systems with higher complexity and better performance. For example, systems with high-dimensional hyperchaos have been developed, such as 3D [6,7], 4D [8,9], 5D [10], and 6D [11–13] models, which provide larger key spaces and more complex dynamic behaviors, thereby effectively resisting brute-force and dynamic analysis attacks. Beyond complex chaotic designs, hybrid models that integrate chaos with other cryptographic techniques (e.g., ECC) and optimization algorithms (e.g., GA) have shown promise in enhancing security and efficiency [14]. Further improvements include the introduction of memristors [3,12] and fractional-order calculus [15], which bring features such as multi-stability, grid multi-scroll attractors, and memory effects, enriching the randomness and unpredictability of chaotic sequences. In terms of efficiency, optimization strategies such as vector-level operations [16,17], bit-level circular shifts [18], and FPGA-based parallel acceleration have been adopted, achieving encryption speeds of over 0.8 Gbps [19], making them suitable for high-speed real-time scenarios.

Beyond still images, chaos-based encryption techniques have also been extended to video data. As highlighted in a comprehensive review by Gao et al. [20], chaos theory provides a powerful foundation for enhancing the security and efficiency of video encryption, particularly in resource-constrained environments. Their work classifies common chaotic systems and evaluates their performance metrics, offering theoretical guidance for selecting suitable chaotic systems. Concurrently, the exploration of novel chaotic systems continues, with memristor-based models offering enhanced dynamics for encryption, such as 3D maps with dual memristors for robust image security [21] and hyperchaotic maps with infinite attractors for multi-image encryption [22]. In terms of implementation, parallel encryption algorithms based on chaotic neuron maps, such as the 2D Logistic-Rulkov neuron map (2D-LRNM), significantly reduce computation time while maintaining high security [23]. For video content, segment-based encryption methods like “Encrypt a story” (EAS) further improve efficiency by selectively encrypting key frames or segments, saving up to 90% of the time without compromising security [24].

In addition to chaotic encryption, the combination of chaos with DNA encoding has become an important research direction. However, cryptanalytic studies reveal that such hybrid schemes require careful security validation. For instance, the medical image encryption scheme MPPS based on coupled

chaotic systems and DNA coding was broken via a low-complexity chosen-plaintext attack [25]. Similarly, an algorithm combining variable step-length Josephus traversing with DNA dynamic encoding was shown to have impractical key design and process flaws [26], while a Feistel network-based scheme incorporating dynamic DNA encoding demonstrated vulnerabilities in key design and encryption process [27]. These findings underscore that merely combining chaos and DNA operations does not inherently ensure security. Despite these challenges, researchers continue to develop more sophisticated DNA-based encryption schemes. Advances include eight-base DNA algebraic systems [28,29] that extend traditional four-base encoding, DNA cubes [29], and DNA trees [30] for spatial scrambling and key generation. A prominent trend is the deep integration of chaotic systems with DNA operations, where chaotic sequences dynamically control DNA encoding rules [31], govern DNA computations [32]. Based on the cloud environment, an efficient multi-image encryption strategy integrating DNA coding and orthogonal arrays is proposed [33]. This synergy, when properly designed, enhances resistance to statistical and differential attacks [34–38], though careful consideration of structural and key management vulnerabilities remains crucial to withstand modern cryptanalytic threats.

Building upon the advancements and lessons learned from chaos-based and DNA-encoded encryption schemes, we propose a novel bit-level image encryption algorithm that synergistically combines a hyper-chaotic system with dynamic DNA encoding. The major contributions are summarized as follows:

- We convert hyper-chaotic sequences into four normalized two-dimensional arrays. Each array undergoes K-means clustering to establish multiple cluster centers. Subsequently, we compute directional differentials along x and y axes for all points, which are then binarized to generate cryptographic pseudo-random sequences.
- During the encryption stage, the scrambled image undergoes an additional shuffling process both at bit-plane level and bit-level with the pseudo-random binary sequence. Consequently, this process not only alters the arrangement of the bits in the image pixels but also modifies the pixel values. By employing this approach, the correlation of the image is further disrupted.
- Furthermore, DNA encoding is applied independently to the encrypted image and the pseudo-random binary sequence, with the encoding rules for each being generated from chaotic sequences. Subsequently, DNA operations are carried out on the encoded ciphertext DNA and random DNA, followed by decoding using the DNA rules obtained from the pseudo-random binary sequence. This process significantly improves the randomness of the DNA encoding outcomes, thereby bolstering their effectiveness against brute-force attacks.

The rest of this paper is organized as follows: In Section 2, we introduce the preliminary materials. In Section 3, we provide details of the proposed encryption and decryption algorithms. In Section 4, we present experimental results and security analysis. In the last section, we conclude our study.

2. Preliminaries

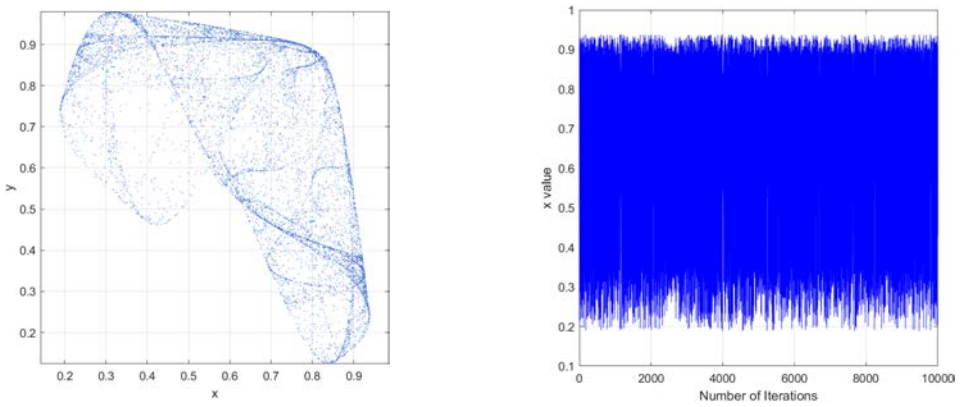
In this section, we outline the core components of the proposed encryption algorithm. First, the 2D Logistic map and Chen's hyper-chaotic system are introduced as entropy sources, and their chaotic robustness is confirmed by phase portraits and Lyapunov exponents. Then, DNA encoding principles are detailed, including its eight coding rules and algebraic operations, which enable confusion and diffusion at the molecular level. Together, these elements establish the foundation for the algorithm.

2.1. 2D Logistic map

The 2D Logistic map [39] is as follows:

$$\begin{cases} u_{n+1} = a_1 u_n \left(1 - u_n\right) + \lambda_1 v_n{}^2 \\ v_{n+1} = a_2 v_n \left(1 - v_n\right) + \lambda_2 \left(u_n{}^2 + u v_n\right) \end{cases} \tag{1}$$

where $a_1, a_2, \lambda_1, \lambda_2$ are the control parameter of the chaotic system. When $2.75 < a_1 \le 3.4, 2.7 < a_2 \le 3.45$, $0.15 < \lambda_1 \le 0.21, 0.13 < \lambda_2 \le 0.15$, the system is in a chaotic state. Figure 1 illustrates the chaotic behavior of the 2D Logistic map with parameters $a_1 = 3.1, a_2 = 3.2, \lambda_1 = 0.18, \lambda_2 = 0.14$ and initial values $u_0 = 0.2, v_0 = 0.3$. The corresponding Lyapunov exponents are $\lambda_1 = 0.242897, \lambda_2 = -0.000334$, yielding a Kaplan-Yorke dimension of $D_KY = 2.000000$.



(a) (b)

Figure 1. Chaotic behavior of 2D logistic iteration: (a) Phase portrait of the 2D logistic map; and (b) temporal evolution of the x-sequence.

2.2. Chen's hyper-chaotic system

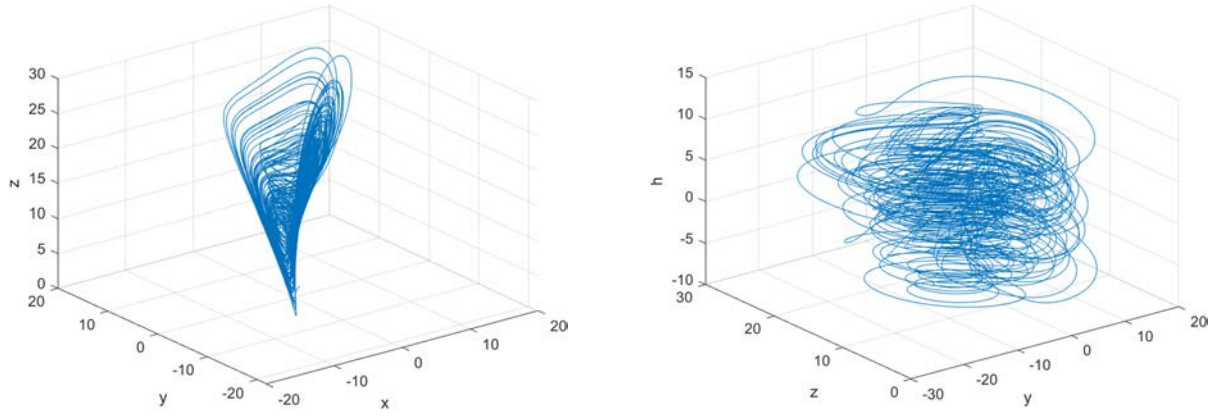
High-dimensional hyper-chaotic systems have more complex dynamical behavior than low-dimensional discrete chaotic systems, and the chaotic sequences generated are more random and unpredictable. Using the chaotic sequence generated by the hyper-chaotic system to encrypt the image will make the cipher image more secure. The mathematical model of the Chen hyper-chaotic system [40] is as follows:

$$\begin{cases} \dot{x} = a(y - x), \\ \dot{y} = -(d + z)x + cy - h, \\ \dot{z} = xy - bz, \\ \dot{h} = x + r \end{cases} \tag{2}$$

where a, b, c, d, r is the control parameter of the hyper-chaotic system, and x, y, z, h is the state variable of hyper-chaotic system. The system is in a hyper-chaotic state [41] when $a = 36, b = 3, c = 28, d = 16, \quad -0.7 \le r \le 0.7$.

When $r = 0.2$, the system has two positive Lyapunov exponents. Figure 2 shows the attractors of

the hyper-chaotic system in (2).



(a) (b)

Figure 2. The attractors of hyper-chaotic system: (a) x-y-z plane; and (b) y-z-h plane.

2.3. DNA encoding and operation

A DNA molecule is composed of four distinct nucleic acid bases [42]: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). These bases form complementary pairs, where A pairs with T, and C pairs with G. To represent these bases using binary coding rules, A, C, G, and T are encoded as 00, 01, 10, and 11, respectively. There are eight coding rules that correspond to biological models, as shown in Table 1.

Table 1. DNA encoding rules.

Rules	1	2	3	4	5	6	7	8
00	A	A	G	G	T	T	C	C
01	C	G	A	T	C	G	A	T
10	G	C	T	A	G	C	T	A
11	T	T	C	C	A	A	G	G

The binary code 00011011 corresponds to pixel 27. According to rule 1, this code can be converted into a DNA sequence, which would be represented as ACGT.

Using principles of binary arithmetic, Tables 2 and 3 illustrate the operations of DNA addition and subtraction in rule 1, correspondingly.

Table 2. DNA addition operation.

+	C	T	A	G
C	C	T	A	G
T	T	A	G	C
A	A	G	C	T
G	G	C	T	A

Table 3. DNA subtraction operation.

–	C	T	A	G
C	C	G	A	T
T	T	C	G	A
A	A	T	C	G
G	G	A	T	C

3. Proposed image encryption algorithm

In this section, we detail a novel bit-level image encryption scheme. First, a high-entropy chaotic sequence, optimized via K-means and validated by NIST tests, serves as the keystream. Then, multi-level scrambling disrupts correlations across bit-planes and positions. Finally, dynamic DNA encoding and diffusion, controlled by chaotic sequences, introduce non-linearity and security. The symmetric decryption process is also outlined.

3.1. Generating a pseudo-random binary sequence

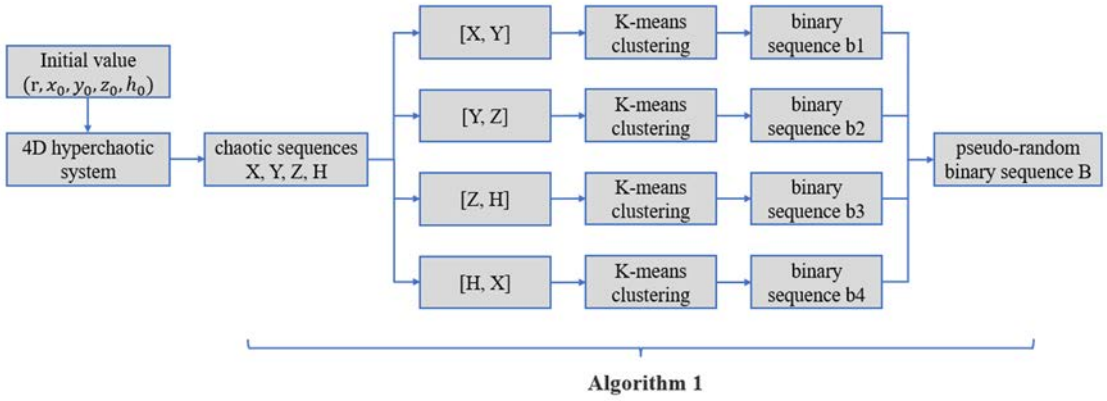


Figure 3. Flow chart of generating pseudo-random binary sequence.

First, we generate four chaotic sequences $X' = \{x_1', ..., x_L'\}, Y' = \{y_1', ..., y_L'\}, Z' = \{z_1', ..., z_L'\}$, $H' = \{h_1', ..., h_L'\}$, based on the chaos formula in (2), where $L \geq M \times N \times 8$. The chaotic sequences are transformed into four new sequences $X = \{x_1, ..., x_L\}, Y = \{y_1, ..., y_L\}, Z = \{z_1, ..., z_L\}, H = \{h_1, ..., h_L\}$, by:

$$x_i = \mathrm{mod}(round(x_1' \times 10000), 256) \tag{3}$$

Then, as shown in Figure 3, we employ Algorithm 1 to create pseudo-random binary sequence B.

Algorithm 1

Step 1: For the two-dimensional sequence $[X, Y]$, select the maximum value $X_{\max}$ and the minimum value $x_{\min}$ of sequence X. Let T be a positive integer, and divide the sequence X into equal parts by step $t_x = (X_{\max} - X_{\min})/T$. Each equidistant point is represented by:

$$p_i = X_{\min} + i * t_x, i = 0,1,2,...,T \tag{4}$$

Similarly, divide Y into T equal parts, where $t_y = (Y_{\max} - Y_{\min}) / T$, and each equidistant point is represented by:

$$q_j = Y_{\min} + j * t_y, j = 0,1,2,...,T \tag{5}$$

Let

$$u_i = \left[\frac{p_{i-1} + p_i}{2}, \frac{q_{i-1} + q_i}{2} \right], i = 1,2,...,T \tag{6}$$

Step 2: Take X and Y as two dimensions to form a two-dimensional data set R, and the *i*-th data set is represented as follows:

$$R_i = [x_i, y_i] \tag{7}$$

Step 3: Initialize *i* and S_j with:

$$\begin{cases} i = 1, \\ S_j = \varnothing, j = 1,2,...,T \end{cases} \tag{8}$$

Step 4: Iterate i, let u_i be the initial center of the *i*-th cluster of k-means, use the formula to calculate the cluster k of R_i, and renew $S_k = S_k U \{R_i\}$.

$$k := \arg \min_{j \in \varphi(T-1)} \sqrt{(x_i - u_j)^2 + (y_i - u_j)^2}, k \in \varphi(T-1) \tag{9}$$

Step 5: Calculate the average of the observed values of each dimension in each cluster to obtain *k* new centroid positions.

$$u'_j = [\sum_{l \in S_j} x_l / n_j, \sum_{l \in S_j} y_l / n_j] \tag{10}$$

Use u' as the new clusters center, and repeat Steps 3 through 5 until $\left| u'_j - u_j \right| \le 10^{-4}$.

Step 6: After K-means clustering, calculate the distance between the elements in each cluster and the center of the cluster. For $R_k = [x_k, y_k]$ in cluster S_j, the distance difference to the center of the cluster is $\left[x_k - u'_{j1}, y_k - u'_{j2} \right]$. The distance difference calculated after all the elements in the cluster $S_1, \cdots, S_T$ are arranged in the order of the cluster generates a new sequence H, which can be described as

$$\begin{cases} H_i = [h_{i1}, h_{i2}] \\ h_{i1} = x_k - u'_{j1} \\ h_{i2} = y_k - u'_{j2} \end{cases} \tag{11}$$

where $i = 1, 2, ..., L$, and j is the cluster serial number it belongs to.

Step 7: Binarize h_{i1}, h_{i2} respectively by:

$$h'_{ij} = \begin{cases} 1, if\ h_{ij} \geq 0 \\ 0, else \end{cases} \tag{12}$$

where $i = 1, 2, ..., L; j = 1, 2$.

Step 8: Let $b1 = \{b1_1, ..., b1_i, ..., b1_L\}$, where $b1_i = [h'_{i1}, h'_{i2}], i = 1, 2, ..., L$.

Step 9: Similarly, for $[Y, Z], [Z, H], [H, X]$, repeat Steps 1 through 7 to get $b2, b3$ and $b4$.

Step 10: Finally, combine $b1, b2, b3, b4$ into a pseudo-random binary sequence $B(B_1, ..., B_{8L})$ by:

$$\begin{cases} B_{8i-7} = b1_{i1} \\ B_{8i-6} = b2_{i1} \\ B_{8i-5} = b3_{i1} \\ B_{8i-4} = b4_{i1} \\ B_{8i-3} = b1_{i2} \\ B_{8i-2} = b2_{i2} \\ B_{8i-1} = b3_{i2} \\ B_{8i} = b4_{i2} \end{cases} \tag{13}$$

The generated pseudo-random binary sequence B passed the NIST test with the following results:

The generated pseudo-random binary sequence B passed the NIST test with a sequence length of $8L$, where $L = M \times N \times 8$, and M and N denote the number of rows and columns of the image, respectively. The test results are shown in Table 4 and Figure 4.

Table 4. Results of the NIST statistical test.

Sequence number	Statistical test	P-value	Result
1	Frequency	0.7505	Passed
2	Block frequency	0.2481	Passed
3	Runs	0.5879	Passed
4	Longest run	0.5000	Passed
5	Binary matrix rank	0.8000	Passed
6	Spectral	0.0500	Passed
7	Nonoverlapping template	0.3170	Passed
8	Overlapping template	0.9273	Passed
9	Universal	0.8000	Passed
10	Linear complexity	0.7000	Passed
11	Serial	0.0100	Passed
12	Cumulative sums	0.5451	Passed
13	Approximate entropy	0.5664	Passed
14	Random excursions	0.0789	Passed
15	Random excursions variant	0.1360	Passed

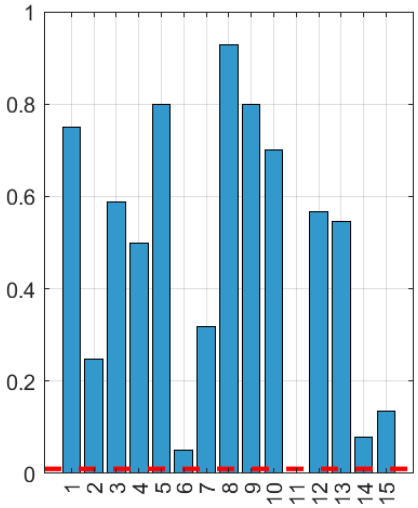


Figure 4. P-value distribution plots.

3.2. Encryption algorithm

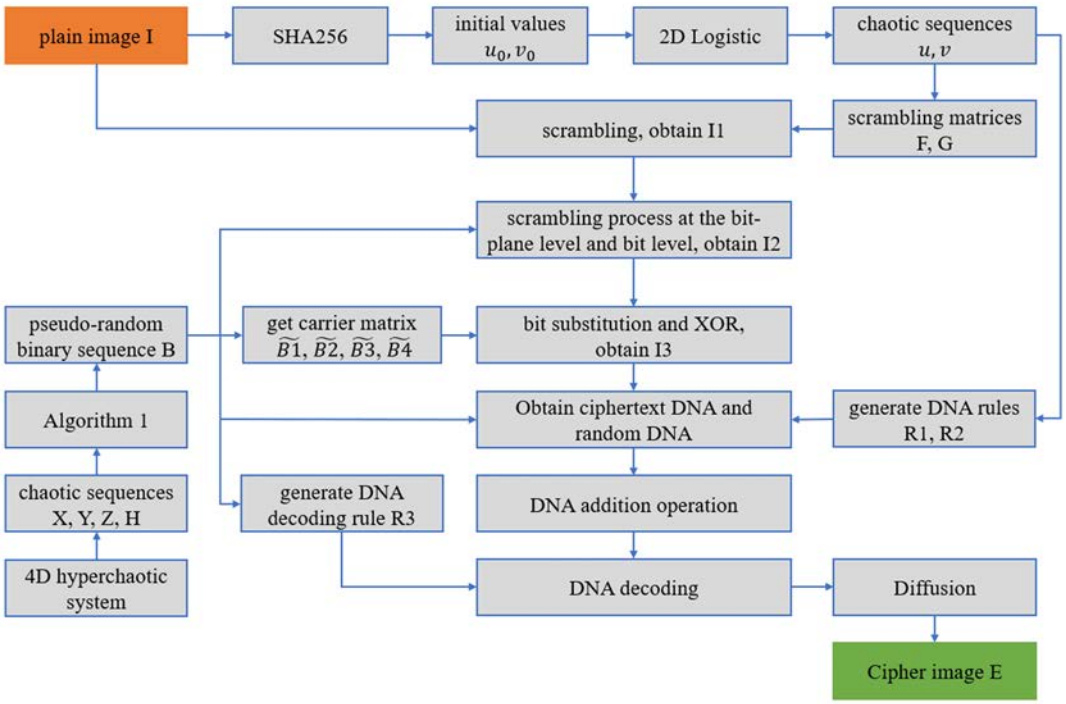


Figure 5. Flow chart of the proposed scheme.

Figure 5 describes the proposed scheme's flow chart. The detailed description of the encryption algorithm process is as follows:

Step 1: Key generation from plain image hash.

Denote the plain image I, where the size of I is $M \times N$. To enhance the sensitivity of the encryption algorithm to plain image, we calculate the SHA-256 hash value of the plain image I,

resulting in a 256-bit hash. This hash is then converted into 32 decimal values, named as follows: $h=\{h_1,h_2,...,h_{32}\}$. Subsequently, we utilize Eq (14) to generate two initial values, u_0 and v_0, for the 2D Logistic sequences.

$$\begin{cases} u_0=\sum_{i=1}^{16} h_i/16/256 \\ v_0=\sum_{i=16}^{32} h_i/16/256 \end{cases} \tag{14}$$

Step 2: Initial pixel-level scrambling.

Two chaotic sequences u,v are generated using 2D Logistic in Eq (1), and reshape u,v into two matrices U,V with size $M\times N$. Sort U and V to obtain two scrambling matrices F and G. Shuffle I with P_x and P_x to obtain shuffled image $I1$. This procedure is described by $I1=F\bullet I\bullet G$.

Step 3: Pseudo-random sequence segmentation

Take eight sequences $B1(B_1,...,B_L), B2(B_{L+1},...,B_{2L}), B3(B_{2L+1},...,B_{3L}), B4(B_{3L+1},...,B_{4L}), B5(B_{4L+1},...,B_{5L}), B6(B_{5L+1},...,B_{6L}), B7(B_{6L+1},...,B_{7L}), B8(B_{7L+1},...,B_{8L})$, of length $L=M\times N$ in order from the pseudo-random binary sequence B generated by Algorithm 1.

Step 4: Generation of Bit-plane scrambling matrices.

Convert $B1,...,B8$ to eight decimal matrices of size $M\times N$, then define them as $\tilde{B}1,...,\tilde{B}8$. Sort $\tilde{B}1,...,\tilde{B}8$ in each column to obtain eight scrambling matrices $Px_1,...,Px_8$, and $\tilde{B}1,...,\tilde{B}8$ in each row to obtain eight scrambling matrices $Px_1,...,Px_8$.

Step 5: Bit-level scrambling

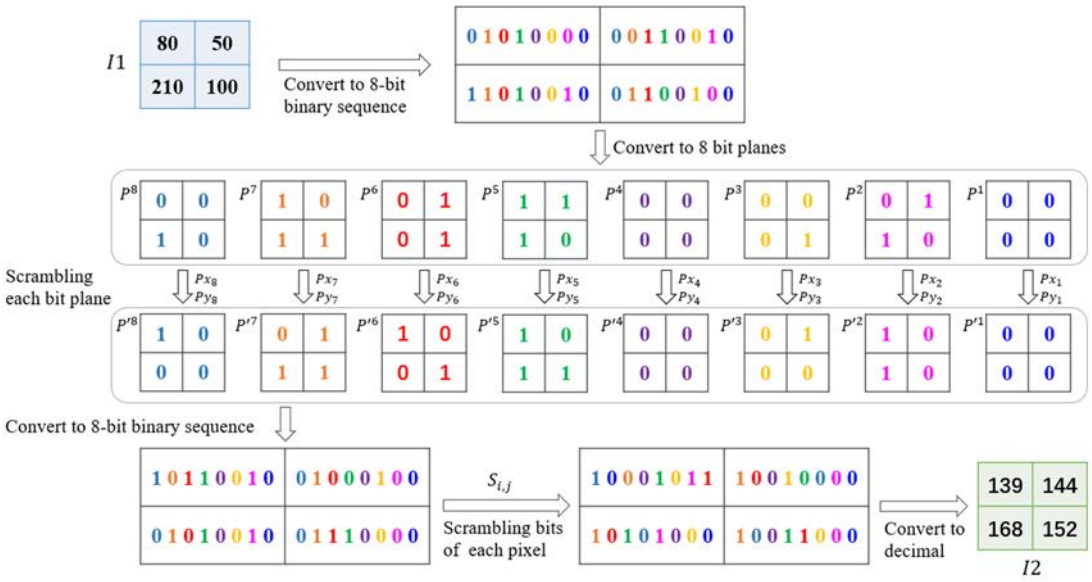


Figure 6. Flow chart of a scrambling process at the bit-plane level and bit-level.

The plain image considered in this paper is an 8-bit gray image with 256 levels of gray. Therefore, each pixel can be represented by an 8-bit binary sequence, and thus the image can be decomposed into 8 bit planes. We use the pseudo-random binary sequence B to scramble the permuted matrix $I1$ at the bit-plane level and bit-level. The scrambling process is shown in Figure 6. First, each pixel of the first permuted matrix $I1$ is converted into an 8-bit binary sequence. The elements of bit8, bit7, bit6, bit5, bit4, bit3, bit2, and bit1 of each pixel are grouped into bit-planes, resulting in 8 bit-planes, labeled

$P^8, P^7, P^6, P^5, P^4, P^3, P^2, P^1$. We use Px_i and Py_i to scramble the p^i bit-plane. The scrambling algorithm is defined as: $P'^i = Px_i \bullet P^i \bullet Py_i$. Then, the 8 bit-planes $P'^8, P'^7, P'^6 P'^5, P'^4, P'^3, P'^2, P'^1$ are converted back into binary pixel sequences in the order of bit8, bit7, bit6, bit5, bit4, bit3, bit2, and bit1, denoted by $P2$. The pseudo-random binary sequence B is converted into a decimal sequence, labeled $\tilde{B}$. $\tilde{B}$ is then divided into segments of 8 decimal numbers, which are rearranged to form an $M \times N$ matrix O. O is described by:

$$O_{i,j} = \tilde{B}(1, 8*((i-1)*256+j)-7 : 8*((i-1)*256+j)) \tag{15}$$

After sorting $O_{i,j}$, scrambled matrix $S_{i,j}$ is generated. $S_{i,j}$ is then used to scramble the 8 bit positions of $P2_{i,j}$, and we obtain the binary matrix $P3$. $P3$ is then converted into decimal to obtain the second permuted image $I2$.

Step 6: Bit-level embedding and XOR operation.

Next, the bit8 and bit7 of the encryption matrix $I2$ are taken and used to replace the corresponding bit8 and bit7 of the matrix $\tilde{B}1$, generating matrix $A1$. Similarly, the bit6 and bit5 of $I2$ are used to replace the corresponding bit6 and bit5 of the matrix $\tilde{B}2$, generating matrix $A2$. The bit4 and bit3 of $I2$ are used to replace the corresponding bit4 and bit3 of $\tilde{B}3$, generating matrix $A3$. Last, the bit2 and bit1 of $I2$ are used to replace the corresponding bit2 and bit1 of $\tilde{B}4$, generating matrix $A4$. The matrices $A1, A2, A3$ and $A4$ are then performed XOR operation together to generate matrix $I3$, as shown in the following formula:

$$I3 = A1 \oplus A2 \oplus A3 \oplus A4 \tag{16}$$

For this step, $A1', A2', A3', A4'$ can be calculated by Eq (17) during decryption. The bit8 and bit7 of $A1'$ correspond to bit8 and bit7 of $I2$ that before the XOR operation. Analogously, the encrypted matrix $I2$ can be recovered through bit8 and bit7 of $A1'$, bit6 and bit5 of $A2'$, bit4 and bit3 of $A3'$, and bit2 and bit1 of $A4'$.

$$\begin{cases} A1' = I3 \oplus \tilde{B}2 \oplus \tilde{B}3 \oplus \tilde{B}4 \\ A2' = I3 \oplus \tilde{B}1 \oplus \tilde{B}3 \oplus \tilde{B}4 \\ A3' = I3 \oplus \tilde{B}1 \oplus \tilde{B}2 \oplus \tilde{B}4 \\ A4' = I3 \oplus \tilde{B}1 \oplus \tilde{B}2 \oplus \tilde{B}3 \end{cases} \tag{17}$$

Step 7: DNA encoding and operation.

Then, using the 2D logistic sequences u and v generated in Step 2, as well as the sequence $\tilde{B}5$, three DNA encoding rule matrices are calculated for DNA encoding and decoding. The generation formulas are as follows:

$$\begin{cases} R1_{i,j} = ceil(u_{i*j} / 0.125) \\ R2_{i,j} = ceil(v_{i*j} / 0.125) \\ R3_{i,j} = \mathrm{mod}(\tilde{B}5_{i,j}, 8) + 1 \end{cases} \tag{18}$$

DNA encoding is performed on the ciphertext $I3$ and sequence $\tilde{B}6$, generating matrices S_{I3} and S_{B6}, respectively. To ensure the randomness of the encoding process, the DNA encoding rule matrix $R1$ is used to determine the DNA encoding rule for each element of $I3$, and DNA encoding

rule matrix $R2$ is used to determine the DNA encoding rule for each element of $\tilde{B}6$. The matrices S_{I3} and S_{B6} are then subjected to DNA addition operation, and the resulting sequence is decoded using DNA decoding rule $R3$. Finally, the encryption matrix C is generated.

Step 8: Bidirectional diffusion

To further enhance security, the pixel values of the permuted image C are subjected to bidirectional diffusion, both forward and backward. The scrambling matrices F and G generated through 2D logistic in step 2 are used as diffusion matrices. The resulting matrices after bidirectional diffusion are denoted as D and E, respectively.

Forward diffusion is performed by:

$$\begin{cases} D_{i,j} = \mathrm{mod}(k_1 D_{i,j-1} + k_2 F_{i,j} + C_{i,j}, 256) \\ D_{i,0} = D_{i-1,N} \end{cases} \tag{19}$$

where k_1, k_2 are control parameters.

Backward diffusion is performed by:

$$\begin{cases} E_{i,j} = \mathrm{mod}(k_3 E_{i,j+1} + k_4 G_{i,j} + D_{i,j}, 256) \\ E_{i,N+1} = E_{i+1,0} \end{cases} \tag{20}$$

where k_3, k_4 are control parameters.

After diffusion in both directions, E is the final cipher image.

3.3. Decryption algorithm

As we can see from the encryption process, each step of encryption is reversible, so the decryption process is the inverse of the encryption process. The original plain image can be obtained by the sequence of keys in the encryption process, following the inverse calculation.

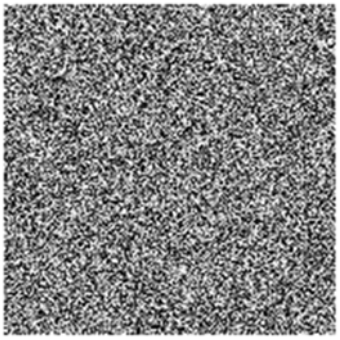
4. Experiments

In this section, the proposed algorithm was used to conduct a series of encryption and decryption experiments on the test image dataset. The proposed encryption scheme was evaluated using several grayscale test images. All simulations were conducted on a workstation equipped with an 12th Gen Intel(R) Core(TM) i5-12600KF CPU running at 3.70 GHz and 32.0 GB of available RAM. The implementation was carried out in the MATLAB R2024a environment. Figures 7 and 8 present the encrypted and decrypted images, respectively. The ciphertext images exhibit a noise-like appearance without any discernible patterns, while the decrypted images are reconstructed perfectly, confirming the algorithm's validity and visual security. Table 5 shows the initial values and control parameters of the encryption algorithm.

The gray images of Lena, Cameraman, all zeros, and all ones, as well as their corresponding ciphertext and decrypted images, are shown in Figure 7. The test results of the color image peppers are shown in Figure 8. We can see that the encryption algorithm is applicable to both gray images and color images, and it can encrypt them into images that resemble noise.



(a) Plain image of Lena



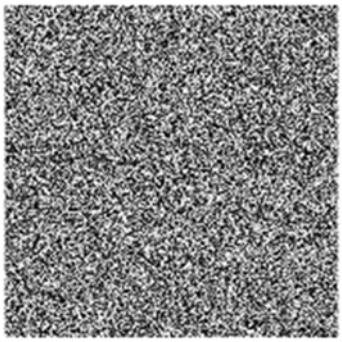
(b) Cipher image of Lena



(c) Decrypted image of Lena



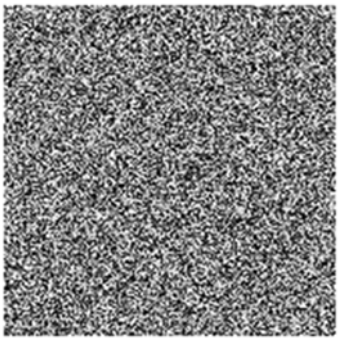
(d) Plain image of cameraman



(e) Cipher image of cameraman



(f) Decrypted image of cameraman



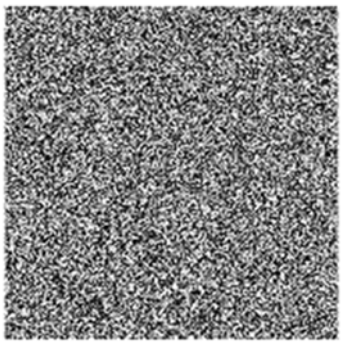
(g) Plain image of all ones

(h) Cipher image of all ones

(i) Decrypted image of all ones



(j) Plain image of all zeros

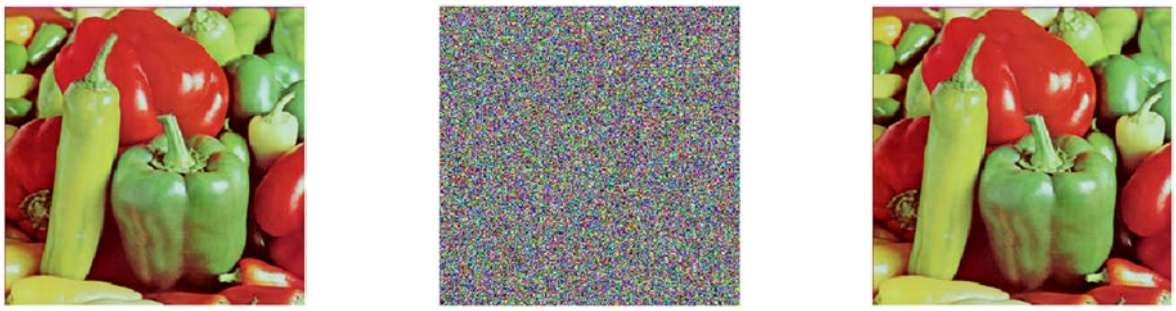


(k) Cipher image of all zeros



(l) Decrypted image of all zeros

Figure 7. Test results for gray images.



(a) Plain image of peppers (b) Cipher image of peppers (c) Decrypted image of peppers

Figure 8. Test results for color image.

Table 5. The initial values and control parameters of the encryption algorithm.

Item	Value
Control parameters and initial values of 2D Logistic	$(a_1, a_2, \lambda_1, \lambda_2) = (3.1, 3.2, 0.18, 0.14)$ (u_0, v_0) Calculated from the SHA value of the image.
Control parameters and initial values of Chen's hyper-chaotic system	$(r, x_0, y_0, z_0, h_0) = (0.2, 1, 0.1, 1.3, 4)$
the control parameters of diffusion sequence	$(k_1, k_2, k_3, k_4) = (2, 6, 3, 7)$

4.1. Key space

The key space, defined as the total number of valid secret keys, must be sufficiently large to resist brute-force attacks. For chaos-based image encryption, a key space greater than 2^{100} is considered secure [43]. In the proposed algorithm, the secret key consists of either a 256-bit hash value H or nine parameters: $a_1, a_2, \lambda_1, \lambda_2, r, x_0, y_0, z_0$, and h_0.

1) If the 256-bit hash value H is used as the key, the key space is 2^{256}.

2) If the parameters $a_1, a_2, \lambda_1, \lambda_2, r, x_0, y_0, z_0$, and h_0 are used as keys, with the computer precision assumed to be 10^{-14}, the key space for each parameter is 10^{14}. Therefore, the total key space for all nine parameters is $(10^{14})^9 = 10^{126}$, which is approximately equal to 2^{418}.

Since an attacker may exploit the smaller key space between the two, the overall key space is determined by the hash value, i.e., 2^{256}. This exceeds the 2^{100} security requirement and ensures strong resistance to brute-force attacks.

Table 6. Comparison of key space.

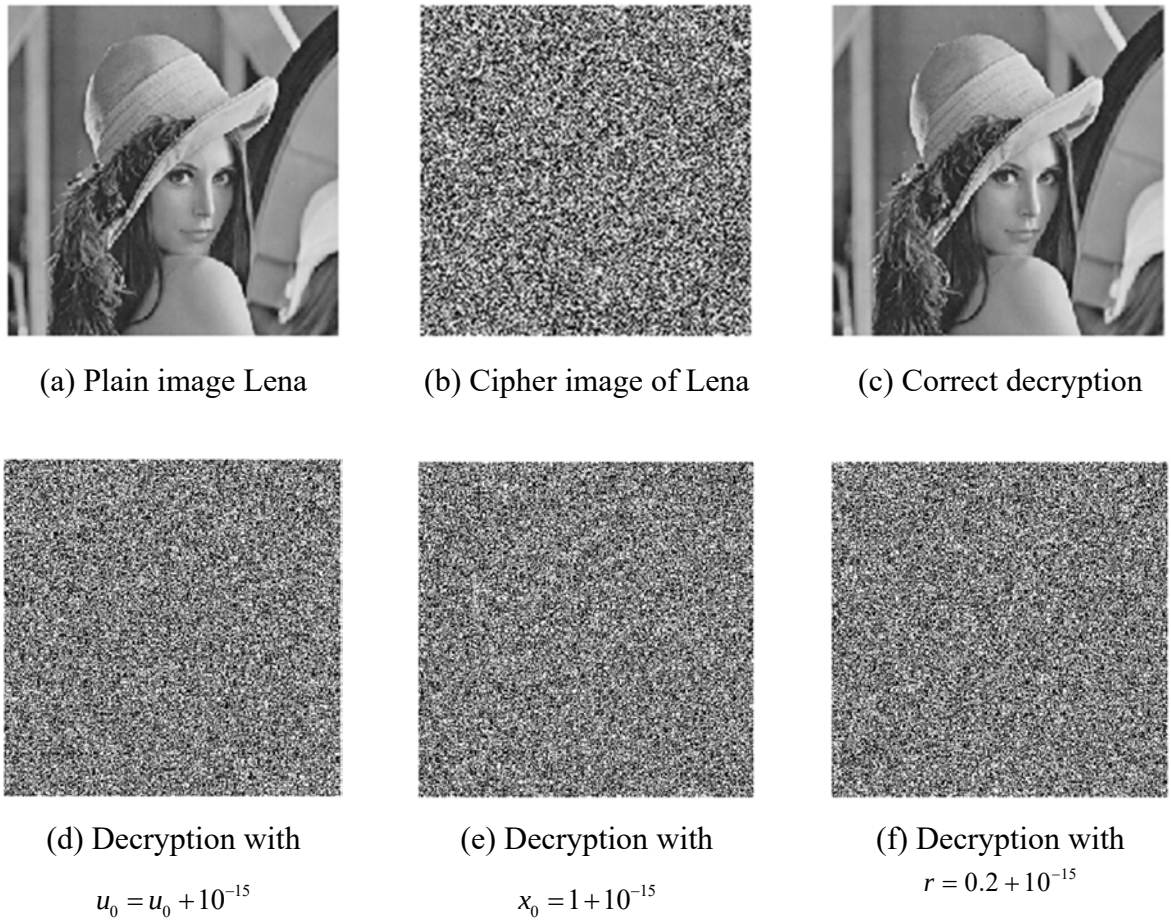
	Proposed	Ref. [16]	Ref. [17]	Ref. [44]
key space	2^{256}	2^{200}	2^{205}	2^{199}

4.2. Sensitivity analysis

Key space is closely related to the number of keys and key sensitivity. We analyze key sensitivity

by decrypting the cipher image with a key that is different from the actual key.

First, set the initial keys as Table 5. Use this key to encrypt the image, and decrypt the cipher image with key changed 10^{-15}. For example, set $u_0 = u_0 + 10^{-15}, x_0 = 1 + 10^{-15}, r = 0.2 + 10^{-15}$, . The test results are shown below.



(a) Plain image Lena (b) Cipher image of Lena (c) Correct decryption

(d) Decryption with $u_0 = u_0 + 10^{-15}$ (e) Decryption with $x_0 = 1 + 10^{-15}$ (f) Decryption with $r = 0.2 + 10^{-15}$

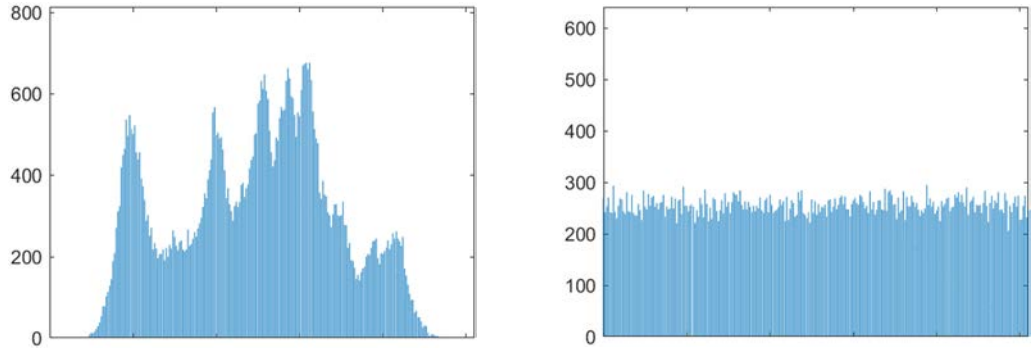
Figure 9. Sensitivity test.

4.3. Statistical analysis

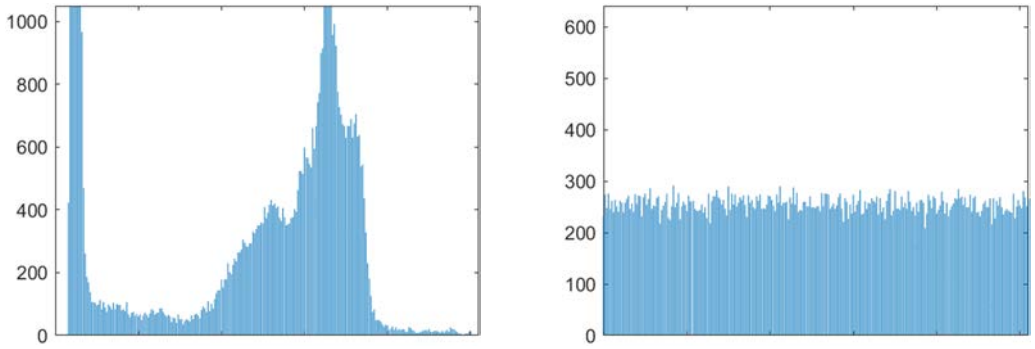
4.3.1. Histogram analysis

The histogram of an image can reflect the statistical characteristics of the image pixel values. In general, the histogram pixels of a meaningful image have a steep distribution. Therefore, in order to hide this information, the histogram of the cipher image should tend to be homogeneous to resist attacks effectively.

The histograms of gray images and cipher images Lena and Cameraman are shown in Figure 10. The Histograms of color image peppers are shown in Figure 11. From these histograms, it is clear that the histograms of the cipher images are uniformly distributed, which are different from the plain image.

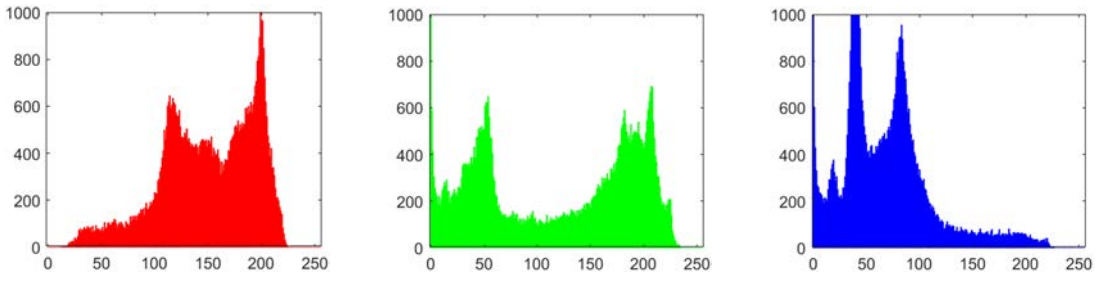


(a) Lena (b) Cipher image Lena

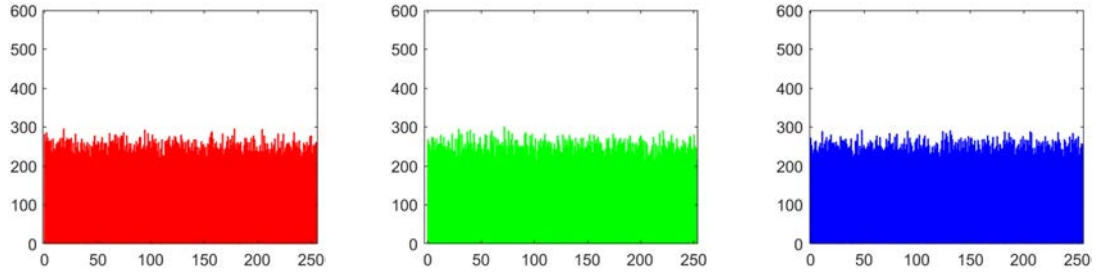


(c) Cameraman (d) Cipher image cameraman

Figure 10. Histograms of gray images and ciphertext.



(a) Peppers_R (b) Peppers_G (c) Peppers_B



(d) Cipher image Peppers_R (e) Cipher image Peppers_G (f) Cipher image Peppers_B

Figure 11. Histograms of color image peppers.

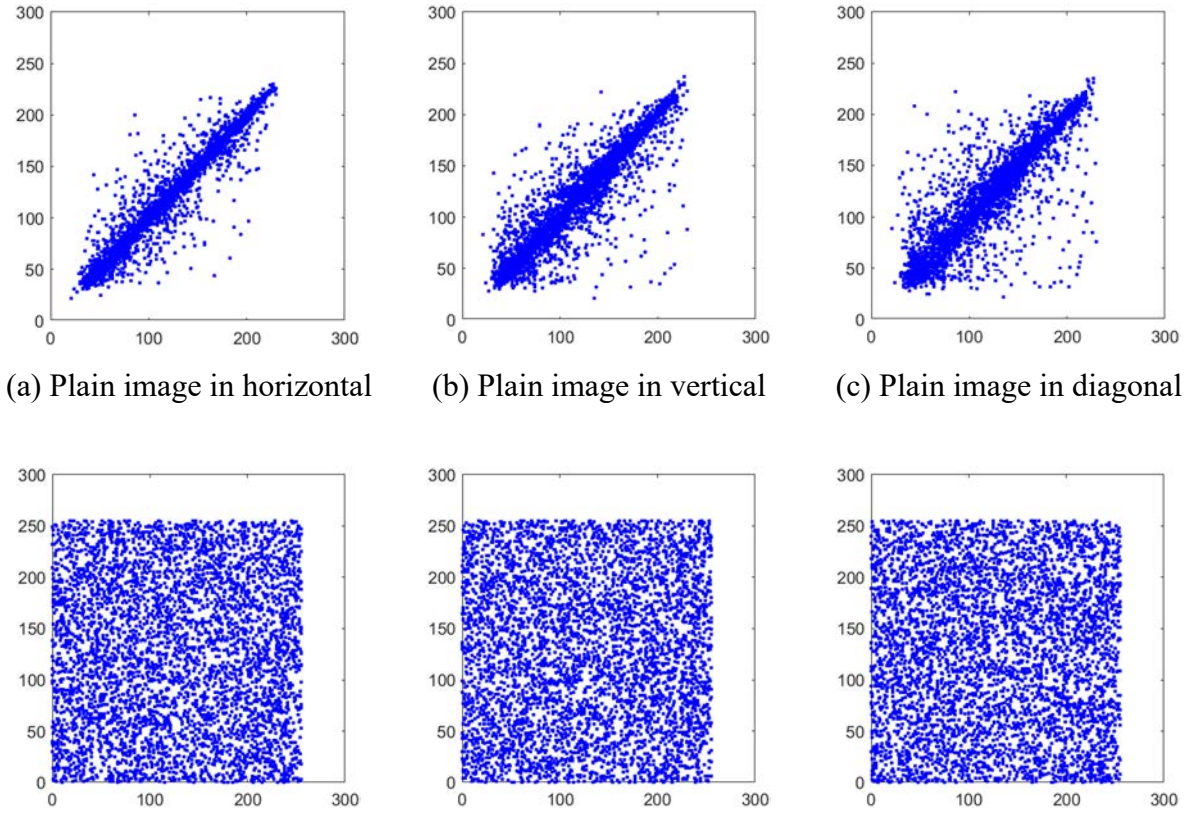
4.3.2. Correlation analysis

In meaningful plain images, there is a high correlation between neighboring pixels, which makes statistical analysis attacks possible. Therefore, an ideal encryption algorithm should effectively reduce the correlation between pixels.

The following tests the correlation of two adjacent pixels by randomly selecting 6000 pairs of adjacent pixels from the plain image and cipher image, respectively, and then calculating the correlation coefficient between adjacent pixels of the sequence. For the plain image and cipher image, Table 7 shows the correlation coefficients in three directions, and Figure 12 shows the correlation distribution of two adjacent pixels in three directions. It can be seen that the pixel correlation of the plain image is effectively reduced.

Table 7. Correlation coefficients between two adjacent pixels.

	horizontal	vertical	diagonal
Plain image of Lena	0.9682	0.9368	0.9030
Cipher image of Lena	0.0007	0.0019	0.0018



(a) Plain image in horizontal (b) Plain image in vertical (c) Plain image in diagonal

(d) Cipher image in horizontal (e) Cipher image in vertical (f) Cipher image in diagonal

Figure 12. Correlation of two adjacent pixels of Lena.

4.4. Information entropy

Information entropy is one of the most important indicators for evaluating the security of encryption. In the image, information entropy is used to measure the distribution of gray values in the image. The higher the information entropy, the more uniform (random) the gray value distribution and the stronger the image's ability to resist statistical attacks. An ideal information entropy value of a 256-level greyscale image is 8. If the information entropy of the cipher image is close to 8, it indicates that the encrypted system has sufficient security and can be effective against attacks.

The information entropy formula is as follows, where $p(x_i)$ is the probability of occurrence of x_i:

$$H(x) = -\sum_{i=1}^{n} p(x_i)\log(p(x_i)) \tag{21}$$

Table 8 shows the Information Entropy of cipher images Lena and peppers. The information entropy calculated is very close to the random ciphertext and effectively improves the security of the cipher image.

Table 8. The Information Entropy of images.

	Lena	Peppers		
		R	G	B
Information entropy	7.9899	7.9895	7.9881	7.9898

4.5. Analysis of a differential attack

Differential attack is a common method of choosing plain image attack, which obtains the key by analyzing the influence of specific plain difference on the corresponding cipher image difference. Fighting differential attacks requires encryption algorithms to be highly sensitive to plain image. This explicit sensitivity is measured NPCR and UACI. NPCR measures the rate of change of cipher image pixels, the closer to the ideal expected value of 99.61%, the more the encryption algorithm is sensitive to changes in plain image, and the stronger it is to resist plain image attacks. The UACI measures the average intensity of change of the cipher image pixels, and the closer it is to the desired value of 33.46%, the more effective the encryption system in resisting attacks. For a gray image with 256 gray levels, the NPCR and UACI are defined as follows:

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \tag{22}$$

$$D(i,j) = \begin{cases} 0, if\ C(i,j) = C'(i,j) \\ 1, if\ C(i,j) \neq C'(i,j) \end{cases} \tag{23}$$

$$UACI = \frac{\sum_{i,j} |C(i,j) - C'(i,j)|}{M \times N \times 256} \times 100\% \tag{24}$$

where M, N are the height and width of the image, C is the cipher images, and C' is the cipher image corresponding to image that differ from the plain image by only one bit pixel.

Table 9 shows the NPCR and UCAI of the proposed encryption algorithm, and Table 10 compares the average NPCR and UACI of cipher images with previous studies. It can be seen that the NPCR and UACI of the cipher images are very close to the ideal values and slightly better than several other algorithms.

Table 9. NPCR and UCAI of the proposed encryption algorithm.

Position	Images		NPCR (%)	UACI (%)
(30, 68)	Lena		99.62	33.49
	Peppers	R	99.62	33.74
		G	99.63	33.41
		B	99.62	33.47
(95, 104)	Lena		99.62	33.47
	Peppers	R	99.61	33.49
		G	99.61	33.52
		B	99.60	33.52
(127, 190)	Lena		99.60	33.41
	Peppers	R	99.61	33.38
		G	99.63	33.53
		B	99.67	33.40
(201, 175)	Lena		99.61	33.37
	Peppers	R	99.58	33.37
		G	99.54	33.30
		B	99.60	33.40

Table 10. Comparison of NPCR and UACI of different methods.

Scheme	Proposed	Ref. [32]	Ref. [45]	Ref. [46]	Ref. [47]
NPCR (%)	99.6106	99.6139	99.4830	99.6109	99.6105
UACI (%)	33.4588	33.4535	33.4166	33.5034	33.4656

4.6. Cropping and noise attacks analysis

A robust image encryption algorithm should effectively conceal visual information while resisting data loss or interference during transmission over noisy channels. To evaluate the robustness of the proposed bit-level encryption method, cropping attacks and salt-and-pepper noise attacks were applied. The quality of the decrypted images was assessed quantitatively using the Mean square error (MSE) and peak signal-to-noise ratio (PSNR), which measure the average squared difference and peak error in decibels between the original and decrypted images, respectively. These metrics are defined as follows:

$$MSE = \frac{1}{M \times N} \sum_{i}^{M} \sum_{j}^{N} [I(i, j) - D(i, j)]^2 \tag{25}$$

$$PSNR = 10 \times \log_{10}(\frac{255^2}{MSE}) \tag{26}$$

where M and N are the image dimensions, I is the original plain image, and D is the decrypted image. A lower MSE and a higher PSNR indicate a decrypted image that is closer to the original.

Resistance to cropping attacks. A cropping attack simulates data loss during transmission. Different proportions (1/16 to 1/2) of the encrypted "Lena" image were removed. As shown in Figure 13(a)–(d), the corresponding decrypted images in Figure 13(e)–(h) remain recognizable even with 1/4 data loss. This demonstrates strong error tolerance, attributable to the bit-level scrambling and bidirectional diffusion stages, which distribute plain-image pixel information across the cipher image. Consequently, data loss causes only localized degradation without complete reconstruction failure.

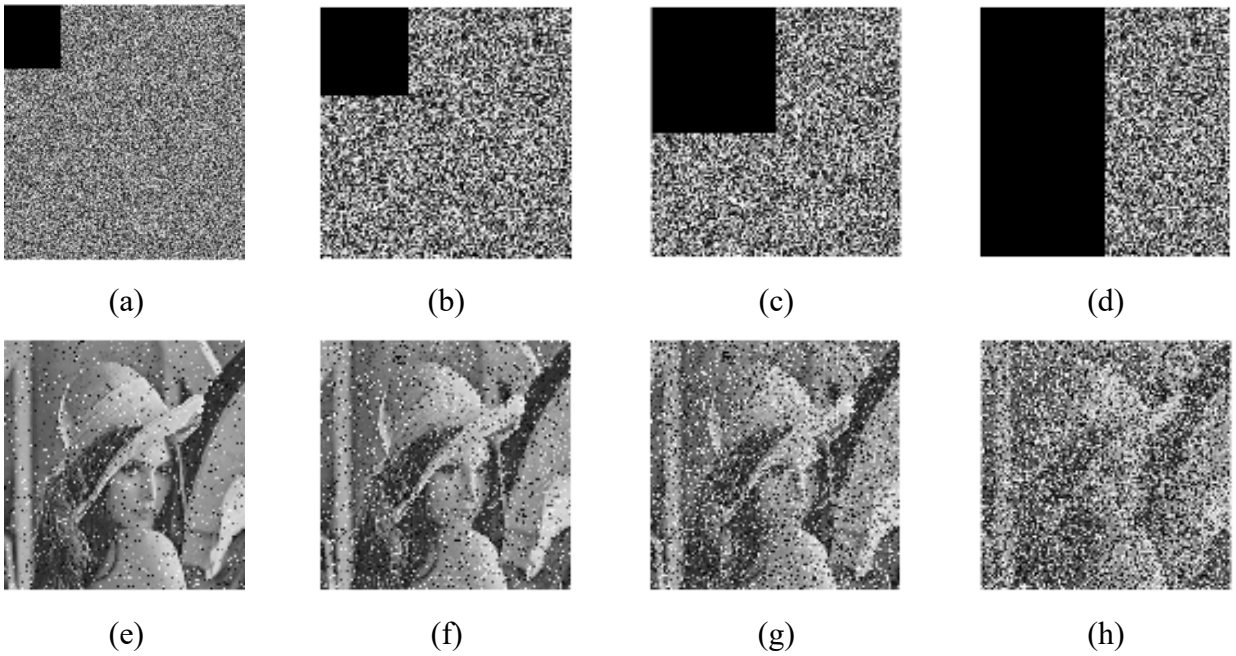


Figure 13. Analysis of clipping attacks: (a) 1/16 data loss in the encrypted images, (e) the decrypted image obtained from (a); (b) 1/8 data loss in the encrypted images, (f) the decrypted image obtained from (b); (c) 1/4 data loss in the encrypted images, (g) the decrypted image obtained from (c); and (d) 1/16 data loss in the encrypted images, (h)the decrypted image obtained from (d).

The quantitative analysis of the cropping attack on the "Lena" image is summarized in Table 11. As expected, the MSE increases and the PSNR decreases as the cropping area expands. Nonetheless, the PSNR values remain at identifiable levels, confirming the visual observations from Figure 13.

Table 11. PSNR and MSE of clipping attacks.

	Metric	1/16	1/8	1/4	1/2
This paper	MSE	311.0467	1488.2029	2662.7579	4502.9012
	PSNR	23.2025	16.4042	13.8775	11.5959
Ref. [6]	MSE	515.6768	986.4857	1994.9501	3960.0381
	PSNR	21.0070	18.1899	15.1315	12.1538
Ref. [48]	-	-	-	-	-
	PSNR	21.4118	-	14.8134	11.6602

The algorithm's robustness against salt-and-pepper noise was evaluated by applying it to the encrypted "Lena" image at densities from 0.01 to 0.2. As shown in Figure 14(a)–(h), the decrypted images preserve essential features even at 20% noise density. This resilience is attributed to the encryption structure, where the decryption process can largely reverse the scrambling and diffusion, mitigating the impact of ciphertext corruption.

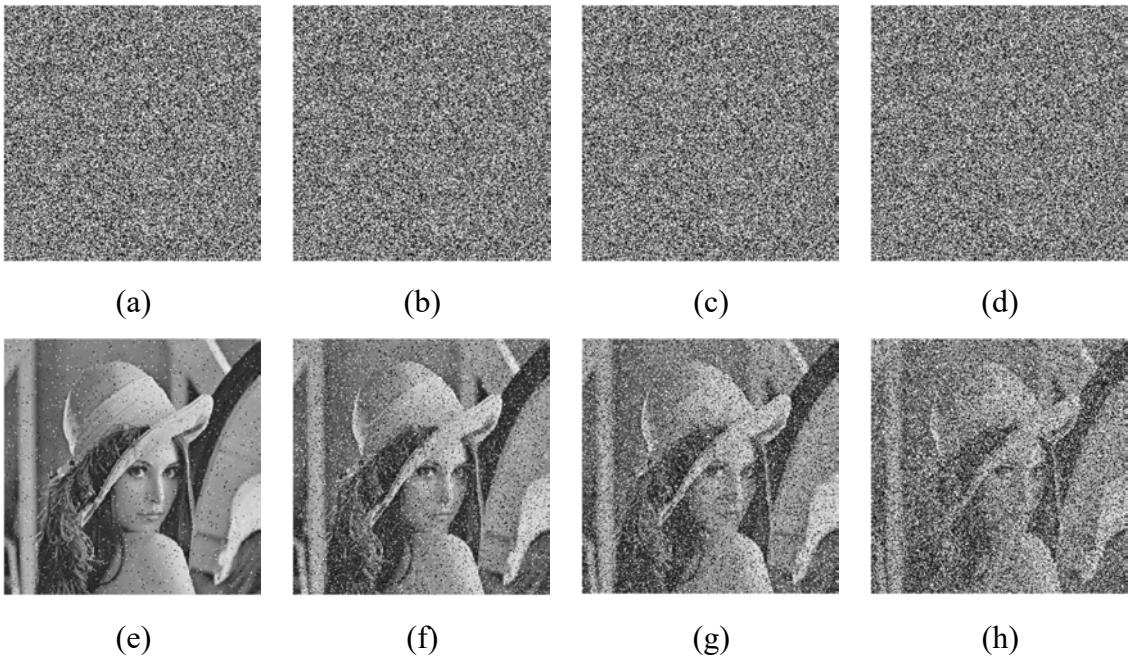


Figure 14. Analysis of salt and pepper noise: (a) Encrypted images by accumulating pepper and salt noise with noise density 0.01, (e) the decrypted images obtained from (a); (b) encrypted images by accumulating pepper and salt noise with noise density 0.05, (f) the decrypted images obtained from (b); (c) encrypted images by accumulating pepper and salt noise with noise density 0.1, (g) the decrypted image obtained from (c); and (d) encrypted images by accumulating pepper and salt noise with noise density 0.2, (h) the decrypted image obtained from (d).

The quantitative analysis of the noise attack, as shown in Table 12, aligns with the visual results. The PSNR decreases as the noise density increases, but the values indicate that the decrypted image quality, while degraded, remains sufficient for information retrieval under considerable noise stress.

Table 12. PSNR and MSE of the salt and pepper noise attack.

	Metric	0.01	0.05	0.1	0.2
This paper	MSE	705.8546	1357.1686	2593.6688	4770.7734
	PSNR	19.6437	16.8045	13.9917	11.3449
Ref. [6]	MSE	-	2124.0528	3712.3536	5735.5641
	PSNR	-	14.8592	12.4343	10.5450
Ref. [48]	MSE	-	-	-	-
	PSNR	-	21.4332	18.4747	-

4.7. Computational complexity analysis

Computational complexity can measure the computational resources required by an algorithm. The proposed encryption algorithm consists of six major components: Pseudo-random binary sequence generation, initial pixel-level scrambling, bit-plane and bit-level scrambling, bit-level embedding and XOR operation, DNA encoding and operation, and bidirectional diffusion. Assuming the size of the encrypted image is $m \times n$, the complexity of each component is analyzed as follows: The pseudo-random binary sequence generation process is $O(4 \times m \times n)$; the initial pixel-level scrambling is $O(m \times n + L \log L + m \times n)$, and since $L \log L \ll m \times n$, the complexity of this stage remains $O(2 \times m \times n)$; the bit-plane and bit-level scrambling is $O(8 \times m \times n)$; the bit-level embedding and XOR operation is $O(m \times n)$; the DNA encoding and operation is $O(m \times n)$; and the bidirectional diffusion is $O(m \times n)$. Therefore, the overall computational complexity of the proposed encryption algorithm is $O(17 \times m \times n)$.

5. Conclusions

We proposed a novel bit-level image encryption algorithm that integrates hyper-chaotic systems with DNA encoding. Its primary contribution is a novel pre-processing technique that employs K-means clustering on normalized hyper-chaotic sequences to generate an NIST SP 800-22-validated pseudo-random binary sequence, overcoming the limitations of direct chaotic sequence usage. The multi-stage encryption framework, comprising 2D Logistic map-based scrambling, bit-level permutation, chaotic DNA encoding, and bidirectional diffusion, effectively disrupts statistical correlations and provides strong non-linearity. Security analyses confirm the algorithm's robustness, demonstrating a large key space, high key/sensitivity, near-ideal NPCR/UACI values, and superior statistical security.

Our research is primarily validated on grayscale images, and its computational efficiency requires further optimization for real-time or resource-constrained environments. The security and complexity trade-offs associated with the multi-layer structure, while beneficial for security, warrant more in-depth investigation for specific application scenarios.

Future efforts will focus on optimizing computational performance to enhance practicality for real-time applications. Extending the algorithm to secure color images, video streams, and other data formats represents a key direction. Furthermore, exploring its application in emerging fields such as cloud-based secure storage and privacy-preserving deep learning will be pursued.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported by the Natural Science Foundation of Gansu Province (Grant No 24JRRG017).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. T. Umar, M. Nadeem, F. Anwer, Chaos based image encryption scheme to secure sensitive multimedia content in cloud storage, *Expert Syst. Appl.*, **257** (2024), 125050. https://doi.org/10.1016/j.eswa.2024.125050
2. T. W. Kang, A multidimensional image encryption and decryption technology, *J. Franklin Inst.*, **361** (2024), 107315. https://doi.org/10.1016/j.jfranklin.2024.107315
3. F. Yu, S. He, W. Yao, S. Cai, Q. Xu, Bursting firings in memristive hopfield neural network with image encryption and hardware implementation, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 2025. https://doi.org/10.1109/TCAD.2025.3567878
4. B. S. Liu, J. Sun, Z. Q. Xu, An improved image encryption algorithm based on chaotic system, *J. Comput.*, **4** (2009), 1091–1100. https://doi.org/10.4304/jcp.4.11.1091-1100
5. R. Matthews, On the derivation of a "chaotic" encryption algorithm, *Cryptologia*, **13** (1989), 29–42. https://doi.org/10.1080/0161-118991863745
6. Q. S. Gao, X. Q. Zhang, Multiple-image encryption algorithm based on a new composite chaotic system and 3D coordinate matrix, *Chaos Solitons Fractals*, **189** (2024), 115587. https://doi.org/10.1016/j.chaos.2024.115587
7. J. Lu, X. Xue, X. Q. Zhang, R. Y. Zhao, Y. S. Zhang, Multiple-image encryption algorithm based on a new 3D hyperchaotic map and Whac-A-Mole scrambling model, *Expert Syst Appl.*, **290** (2025), 128393. https://doi.org/10.1016/j.eswa.2025.128393.
8. I. Al-Dayel, M. F. Nadeem, M. A. Khan, B. S. Abraha, An image encryption scheme using 4-D chaotic system and cellular automaton, *Sci. Rep.,* **15** (2025), 19499. https://doi.org/10.1038/s41598-025-95511-y
9. W. Alexan, K. Hosny, M. Gabr, A new fast multiple color image encryption algorithm, *Cluster Computing*, **325** (2025). https://doi.org/10.1007/s10586-024-04919-0
10. W. Alexan, N. H. E. Shabasy, N. Ehab, E. A. Maher, A secure and efficient image encryption scheme based on chaotic systems and nonlinear transformations, *Sci. Rep.*, **15** (2025), 31246. https://doi.org/10.1038/s41598-025-15794-z
11. M. Youssef, M. Gabr, W. Alexan, M. B. M. Mansour, K. Kamal, H. Hosny, Enhancing satellite image security through multiple image encryption via hyperchaos, SVD, RC5, and dynamic S-Box generation, *IEEE Access*, **12** (2024), 123921–123945. https://doi.org/10.1109/ACCESS.2024.3454512
12. F. Yu, Y. M. Gracia, R. Guo, Z. Ying, J. Xu, W. Yao, et al., Dynamic analysis and application of 6D multistable memristive chaotic system with wide range of hyperchaotic states, *Axioms*, **14** (2025), 638. https://doi.org/10.3390/axioms14080638
13. W. Alexan, M. Youssef, H. H. Hussein, K. K. Ahmed, K. M. Hosny, A. Fathy, et al., A new multiple image encryption algorithm using hyperchaotic systems, SVD, and modified RC5, *Sci. Rep.*, **15** (2025), 9775. https://doi.org/10.1038/s41598-025-92065-x
14. K. Pandey, D. Sharma, Novel image encryption algorithm utilizing hybrid chaotic maps and Elliptic Curve Cryptography with genetic algorithm, *J. Inf. Secur. Appl.*, **89** (2025), 103995. https://doi.org/10.1016/j.jisa.2025.103995

15. W. Feng, K. Y. Zhang, J. Zhang, X. Y. Zhao, Y. Chen, B. Cai, et al., Integrating fractional-order hopfield neural network with differentiated encryption: Achieving high-performance privacy protection for medical images, *Fractal Fract.*, **9** (2025), 426. https://doi.org/10.3390/fractalfract9070426
16. W. Feng, J. Zhang, Y. Chen, Z. T. Qin, Y. S. Zhang, M. Ahmad, et al., Exploiting robust quadratic polynomial hyperchaotic map and pixel fusion strategy for efficient image encryption, *Expert Syst. Appl.*, **246** (2024), 123190. https://doi.org/10.1016/j.eswa.2024.123190
17. H. Li, S. Yu, W. Feng, Y. Chen, J. Zhang, Z. T. Qin, et al., Exploiting dynamic vector-level operations and a 2D-Enhanced logistic modular map for efficient chaotic image encryption, *Entropy*, **25** (2023), 1147. https://doi.org/10.3390/e25081147
18. L. Moysis, M. Lawnik, W. Alexan, S. K. Goudos, M. S. Baptista, G. Fragulis, Exploiting circular shifts for efficient chaotic image encryption, *IEEE Access*, **13** (2025), 92997–93016. https://doi.org/10.1109/ACCESS.2025.3572589
19. W. Alexan, Y. Megalli, A new fast high dimensional and memristive hyperchaotic multiple image encryption method and its FPGA implementation. *Discover Electron.*, **2** (2025), 75. https://doi.org/10.1007/s44291-025-00116-4
20. S. Gao, R. Wu, H. H. Iu, U. Erkan, Y. H. Cao, Q. Li, et al., Chaos-based video encryption techniques: A review, *Comput. Sci. Rev.*, **58** (2025), 100816. https://doi.org/10.1016/j.cosrev.2025.100816
21. S. Gao, H. H. Iu, U. Erkan, C. Simsek, A. Toktas, Y. H. Cao, A 3D memristive cubic map with dual discrete memristors: Design, implementation, and application in image encryption, *IEEE Trans. Circuits Syst. Video Technol.*, **35** (2025), 7706–7718. https://doi.org/10.1109/TCSVT.2025.3545868
22. S. Gao, S. Q. Ding, H. H. Iu, U. Erkan, A. Toktas, C. Simsek, et al., A three-dimensional memristor-based hyperchaotic map for pseudorandom number generation and multi-image encryption, *Chaos*, **35** (2025), 073105. https://doi.org/10.1063/5.0270220
23. S. Gao, Z. Y. Zhang, H. H. Iu, S. Q. Ding, J. Mou, U. Erkan, et al., A parallel color image encryption algorithm based on a 2-D logistic-rulkov neuron map, *IEEE Internet Things J.*, **12** (2025), 18115–18124. https://doi.org/10.1109/JIOT.2025.3540097
24. S. Gao, Z. Y. Zhang, Q. Li, S. Q. Ding, H. H. Iu, Y. H. Cao, et al., Encrypt a story: A video segment encryption method based on the discrete sinusoidal memristive rulkov neuron, *IEEE Trans. Dependable Secure Comput.*, **22** (2025), 8011–8024. https://doi.org/10.1109/TDSC.2025.3603570
25. L. Chen, C. Q. Li, C. Li, Security measurement of a medical communication scheme based on chaos and DNA coding, *J. Visual Commun. Image Represent.*, **83** (2022), 103424. https://doi.org/10.1016/j.jvcir.2021.103424
26. W. Feng, J. Zhang, Z. T. Qin, Y. G. He, Cryptanalysis of image encryption algorithm based on variable step length Josephus traversing and DNA dynamic encoding, *J. Electron. Inf. Technol.*, **44** (2022), 3635–3642. https://doi.org/10.11999/JEIT210791
27. W. Feng, Z. Qin, J. Zhang, M. Ahmad, Cryptanalysis and improvement of the image encryption scheme based on Feistel network and dynamic DNA encoding, *IEEE Access*, **9** (2021), 145459–145470. https://doi.org/10.1109/ACCESS.2021.3123571

28. M. J. Zhao, L. X. Li, Z. Yuan, A multi-image encryption scheme based on a new n-dimensional chaotic model and eight-base DNA, *Chaos, Solitons Fractals*, **186** (2024), 115332. https://doi.org/10.1016/j.chaos.2024.115332
29. L. M. Wu, Z. J. Tian, W. Chen, Color image encryption scheme based on 5D fractional-order complex chaotic system and eight-base DNA cubes, *Expert Syst. Appl.*, **299** (2026), 129950. https://doi.org/10.1016/j.eswa.2025.129950.
30. M. Alawida, A novel DNA tree-based chaotic image encryption algorithm, *J. Inf. Secur. Appl.*, **83** (2024), 103791. https://doi.org/10.1016/j.jisa.2024.103791
31. X. D. Wang, X. B. Liu, J. W. Peng, A novel image encryption scheme based on a 3D enhanced chaotic map and DNA computing model, *J. Franklin Inst.*, **362** (2025), 107790. https://doi.org/10.1016/j.jfranklin.2025.107790
32. J. X. Tian, X. Q. Zhang, M. Liu, S. C. Jin, D. X. Shi, S. W. Yang, Remote sensing image encryption algorithm based on DNA convolution, *J. Supercomput.*, **566** (2025). https://doi.org/10.1007/s11227-025-06982-9
33. L. L. Zhou, Q. L. Chen, F. Tan, C. X. Wu, A chaotic-system-based parallel image encryption algorithm with orthogonal arrays supporting thumbnail decryption, *Expert Syst. Appl.*, **297** (2026), 129272. https://doi.org/10.1016/j.eswa.2025.129272
34. J. F. Jie, Y. Yang, P. Zhang, The construction method of chaotic system model based on state variables and uncertain variables and its application in image encryption, *Appl. Math. Modell.*, **144** (2025), 116097. https://doi.org/10.1016/j.apm.2025.116097
35. J. H. Wu, X. F. Liao, B. Yang, Image encryption using 2D Hénon-Sine map and DNA approach, *Signal Process.*, **153** (2018), 11–23. https://doi.org/10.1016/j.sigpro.2018.06.008
36. J. X. Chen, L. Chen, Y. C. Zhou, Cryptanalysis of a DNA-based image encryption scheme, *Inf. Sci.*, **520** (2020), 130–141. https://doi.org/10.1016/j.ins.2020.02.024
37. X. L. Chai, Y. R. Chen, L. Broyde, A novel chaos-based image encryption algorithm using DNA sequence operations, *Opt. Lasers Eng.*, **88** (2017), 197–213. https://doi.org/10.1016/j.optlaseng.2016.08.009
38. P. F. Fang, M. M. Lou, M. Liu, H. Liu, Image encryption algorithm based on hyperchaotic system and DNA coding, in *International Conference on Computer Communication and Artificial Intelligence (CCAI)*, IEEE, (2021), 41–46. https://doi.org/10.1109/CCAI50917.2021.9447470
39. H. J. Liu, Z. L. Zhu, H. Y. Jiang, B. L. Wang, A novel image encryption algorithm based on improved 3D chaotic cat map, in *2008 The 9th International Conference for Young Computer Scientists*, (2008), 3016–3021. https://doi.org/10.1109/ICYCS.2008.449
40. T. G. Gao, Z. Q. Chen, Z. Z. Yuan, G. R. Chen, A hyperchaos generated from Chen's system, *Int. J. Mod. Phys. C*, **17** (2006), 471–478. https://doi.org/10.1142/S0129183106008625
41. Y. Zhu, C. Wang, J. Sun, F. Yu, A chaotic image encryption method based on the artificial fish swarms algorithm and the DNA coding, *Mathematics*, **11** (2023), 767. https://doi.org/10.3390/math11030767
42. L. M. Adleman, Molecular computation of solutions to combinatorial problems, *Science*, **266** (1994), 1021–1024. https://doi.org/10.1126/science.7973651
43. G. Alvarez, S. J. Li, Some basic cryptographic requirements for chaos-based cryptosystems, *Int. J. Bifurcation Chaos*, **16** (2006), 2129–2151. https://doi.org/10.1142/S0218127406015970
44. J. H. Sun, X. Q. Zhang, C. X. Chen, Image encryption algorithm based on V-shaped scanning and matrix multiplication, *Phys. Scr.*, **100** (2025), 035218. https://doi.org/10.1088/1402-4896/adb103

45. B. Rezaei, H. Ghanbari, R. Enayatifar, An image encryption approach using tuned Henon chaotic map and evolutionary algorithm, *Nonlinear Dyn.*, **111** (2023), 9629–9647. https://doi.org/10.1007/s11071-023-08331-y
46. X. Y. Wang, M. C. Zhao, An image encryption algorithm based on hyperchaotic system and DNA coding, *Opt. Laser Technol.*, **143** (2021), 107316. https://doi.org/10.1016/j.optlastec.2021.107316
47. Y. J. Xian, X. Y. Wang, X. P. Yan, Q. Li, X. Y. Wang, Image encryption based on chaotic sub-block scrambling and chaotic digit selection diffusion, *Opt. Lasers Eng.*, **134** (2020), 106202. https://doi.org/10.1016/j.optlaseng.2020.106202
48. X. Wang, C. Liu, A novel and effective image encryption algorithm based on chaos and DNA encoding, *Multimedia Tools Appl.*, **76** (2017), 6229–6245. https://doi.org/10.1007/s11042-016-3311-8



AIMS Press

©2025 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0)