

Color image encryption algorithm using a new four-dimensional hyperchaotic system

Bo Lin(林博), Min Liu(刘敏), and Guodong Ye(叶国栋)

Citation: Chin. Phys. B, 2026, 35 (**5**): 050502. DOI: 10.1088/1674-1056/ae0bff
http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Synchronization of neuromorphic memristive Josephson junction network and its application

Dejun Yan(严德军), Fuqiang Wu(吴富强), and Wenshuai Wang(汪文帅)
Chin. Phys. B, 2026, 35 (**1**): 010505. DOI: 10.1088/1674-1056/ae1456

Dynamic analysis and DNA coding-based image encryption of memristor synapse-coupled hyperchaotic IN-HNN network

Shuang Zhao(赵双), Yunzhen Zhang(张云贞), Xiangjun Chen(陈湘军), Bin Gao(高彬), and Chengjie Chen(陈成杰)
Chin. Phys. B, 2026, 35 (**1**): 010502. DOI: 10.1088/1674-1056/ae1dec

Design of a memristive chaotic map and its adaptive regulation

Ya Wang(王亚), Xin-Lin Song(宋欣林), and Fei-Fei Yang(杨飞飞)
Chin. Phys. B, 2025, 34 (**9**): 090504. DOI: 10.1088/1674-1056/adea5c

An enhanced fingerprint template protection scheme based on four-dimensional superchaotic system and dynamic DNA coding

Baiqiang Hu(胡百强), Jiahui Liu(刘嘉辉), and Zhe Liu(刘喆)
Chin. Phys. B, 2025, 34 (**7**): 070505. DOI: 10.1088/1674-1056/adca9f

Resonant tunneling diode cellular neural network with memristor coupling and its application in police forensic digital image protection

Fei Yu(余飞), Dan Su(苏丹), Shaoqi He(何邵祁), Yiya Wu(吴亦雅), Shankou Zhang(张善扣), and Huige Yin(尹挥戈)
Chin. Phys. B, 2025, 34 (**5**): 050502. DOI: 10.1088/1674-1056/adb8bb

---

Color image encryption algorithm using a new four-dimensional hyperchaotic system

Bo Lin(林博)[1], Min Liu(刘敏)[2], and Guodong Ye(叶国栋)[1,†]

[1] *Faculty of Mathematics and Computer Science, Guangdong Ocean University, Zhanjiang* 524088, *China*
[2] *School of Management, Guangdong Ocean University, Zhanjiang* 524088, *China*

(Received 23 July 2025; revised manuscript received 10 September 2025; accepted manuscript online 26 September 2025)

With the widespread applications of digital images in fields such as medical imaging, remote sensing, and financial transactions, ensuring image confidentiality has become increasingly important. However, some existing image encryption schemes still suffer from limited key space and insufficient security. To address these, this paper proposes a novel color image encryption algorithm (CIEA-4DALHS) based on a newly constructed four-dimensional augmented Lü hyperchaotic system (4DALHS). The scheme integrates bidirectional spiral cross scrambling with arbitrary starting points and four traversal modes, bit-plane substitution for fine-grained pixel modification, and hierarchical regional segmentation diffusion with cross-channel cascading effects. These strategies significantly enhance both permutation and diffusion effects. Experimental results and security analysis demonstrate the superiority of CIEA-4DALHS. For example, the key space is sufficiently large, high ciphertext information entropy, and strong robustness against statistical and differential attacks. Compared with recent methods, the proposed scheme CIEA-4DALHS offers both high efficiency and security, highlighting its strong potential for real-world applications in image protection.

Keywords: image encryption, hyperchaotic system, bit-plane substitution, dynamic analysis, security analysis

PACS: 05.45.–a, 05.45.Gg

DOI: 10.1088/1674-1056/ae0bff **CSTR:** 32038.14.CPB.ae0bff

1. Introduction

In the digital era, digital images serve as a crucial medium for information transmission and are widely transmitted in many key fields such as medical imaging, satellite, military reconnaissance and financial transactions.[1] However, with the rapid development of Internet technology and the acceleration of global information sharing, image data faces increasingly severe security threats during transmission and storage.[2] Unauthorized access, malicious tampering, and data breaches can not only lead to the exposure of personal privacy, but also cause significant economic losses and even national security crises.[3] Consequently, there is an urgent need to design encryption algorithms that are highly secure and efficient.

In the past five years, researches on image encryption algorithms have increased significantly to enhance their applicability and security. Researchers have incorporated various approaches into image encryption, including chaotic systems,[4] neural networks,[9,10] quantum computing,[11] DNA computing,[12] and compressive sensing.[13] Among these, chaotic systems are widely employed in image encryption due to their characteristics such as sensitivity to initial conditions and high unpredictability. They can generate complex and random sequences that are difficult to predict or reproduce without exact system parameters. Therefore, improving existing chaotic systems or designing higher-performance ones has become a critical research direction in image encryption. Currently, there exist numerous one-dimensional chaotic maps such as Logistic map and Sine map, but they still have drawbacks such as relatively large periodic windows and limited parameter ranges. To overcome these limitations, Yang *et al.*[14] proposed a Sine-Logistic map, which demonstrates an extended parameter range and enhanced chaotic behavior. To address the issue of limited key space vulnerability in existing two-dimensional (2D) chaotic systems, Wang *et al.*[15] suggested a new 2D Sine-Logistic-Tent coupled map. Hong *et al.*[16] transformed a two-wing chaotic system into four-wing and eight-wing chaotic systems and designed an improved AES into image encryption.

Recent cryptanalysis studies have also revealed that some newly image encryption algorithms still have shortcomings. Lei *et al.*[17] re-analyzed a medical privacy protection scheme based on chaotic maps, and pointed out that it is insecure against chosen-plaintext attacks. Zhang *et al.*[18] examined a Feistel–DNA based encryption scheme, and found the weaknesses in its key design and encryption process. To solve these shortcomings, Li *et al.*[19] proposed a three-dimensional (3D) memristive cubic map with dual memristors and demonstrated its ability to generate high-quality pseudorandom key streams while resisting cryptanalytic attacks.

A classic encryption structure in image encryption is the permutation-diffusion framework.[20] Permutation alters

[†]Corresponding author. E-mail: guodongye@hotmail.com

© 2026 Chinese Physical Society and IOP Publishing Ltd. All rights, including for text and data mining, AI training, and similar technologies, are reserved.

pixel positions to disrupt the spatial correlations of the original image. Classical permutation methods include Spiral, Zigzag, and Arnold transformations, which have been widely employed in encryption algorithms. Xiao *et al.*[21] introduced a three-dimensional spiral transformation with selectable starting points, enabling direct conversion from 3D to one-dimensional (1D) but at the cost of longer encryption time. Ali Mansouri *et al.*[22] suggested a new image encryption scheme using an improved Arnold map. Although it achieves better performance in permutation, further improvements are still required to enhance its speed. Zheng *et al.*[23] improved the traditional Zigzag transformation by modifying its starting point. While this approach can disrupt the pixel distribution of the original image to some extent, it suffers from low security and a fixed, single scrambling pattern.

While permutation focuses on disrupting pixel positions, diffusion alters grayscale values to enhance randomness and strengthen resistance to statistical analysis. Common diffusion techniques include XOR operation diffusion, modular arithmetic diffusion and DNA-based operations, which can be applied at either the bit-level or pixel-level. Zhu *et al.*[24] proposed a 3D bit-level image encryption scheme using a Rubik's cube method, which effectively propagates subtle changes throughout the entire cipher image. Zhou *et al.*[25] employed chaotic sequences to perform bitwise XOR operations on the image's bitplanes, achieving randomized modification of pixel values. Wei *et al.*[26] adopted a dual-diffusion strategy that employed 2D XOR operations for both forward and backward diffusion. Li *et al.*[27] utilized a fractional-order hyperchaotic system to generate pseudo-random sequences, combining pixel value modification with row–column permutation and diffusion operations to significantly improve efficiency. Bit-level diffusion can provide fine-grained encryption and strong resistance to statistical attacks, but often incurs higher computational complexity.[28] In contrast, pixel-level diffusion achieves faster encryption with greater efficiency, but generally offers weaker randomness and lower attack resistance. Therefore, effective encryption schemes often seek to combine both strategies to achieve a better balance between security and efficiency.

Based on the above analysis, this paper proposes a new color image encryption algorithm (CIEA-4DALHS) which is suitable for arbitrary matrices, aiming to address certain limitations of existing encryption methods. Compared with conventional secure encryption algorithms such as RSA and ECC, which are computationally intensive and not optimized for large-scale image data, the proposed scheme provides higher efficiency while maintaining strong security, making it more suitable for practical image encryption applications. The main contributions are as follows:

(i) A four-dimensional (4D) augmented Lü hyperchaotic system (4DALHS) is proposed. It is constructed by introducing a feedback controller into the augmented Lü system. Dynamical experiments confirm its hyperchaotic behavior and the presence of a four-wing chaotic attractor.

(ii) A bidirectional spiral cross scrambling method with arbitrary starting points and four traversal modes is designed. Unlike traditional spiral scrambling with fixed and predictable patterns, this method employs chaotic sequences to dynamically determine both the spiral's starting point and traversal mode. Two spiral sequences generated from opposite directions are interleaved to form the final scrambling sequence, significantly enhancing scrambling performance.

(iii) A bit-plane substitution method is developed for fine-grained pixel modification. Each image pixel is decomposed into eight bit-planes, and chaos-driven substitution matrices are employed to reassign bit values. By operating at the bit level rather than the pixel level, this method breaks low-level dependencies among adjacent pixels and improves robustness against chosen-plaintext and statistical attacks.

(iv) A hierarchical regional segmentation diffusion strategy is proposed. Through a dual-layer design of "segmentation-multipath" and "cross-channel cascade", the method overcomes the weakness of fixed single-path diffusion and independent channel processing, thereby enhancing diffusion effectiveness and strengthening resistance to differential attacks.

Rest parts are arranged as follows: Section 2 describes the proposed 4DALHS in detail and analyzes its performance. Section 3 introduces the proposed CIEA-4DALHS, including bidirectional spiral cross scrambling, bit-plane substitution, hierarchical regional segmentation diffusion, and the complete encryption process. Section 4 presents experimental results for CIEA-4DALHS and evaluates its performance and security. Section 5 concludes the whole paper.

2. Proposed 4DALHS

2.1. Construction of 4DALHS

Chaotic systems exhibit complex dynamic behaviors, and the chaotic sequences generated possess randomness and unpredictability, making them suitable for providing high-quality random sequences in image encryption. The augmented Lü system is an extension of the classical Lü system, and its mathematical model is as follows:

$$\begin{cases} \dot{x} = -[ab/(a+b)]x - yz, \\ \dot{y} = ay + xz, \\ \dot{z} = xy + bz, \end{cases} \tag{1}$$

where x, y, and z are the state variables of the system, and a, b are the control parameters. When setting $a = -4$, $b = -10$, $x_0 = 0.5$, $y_0 = 0.5$, and $z_0 = 0.5$, the augmented Lü system exhibits chaotic behavior, with its phase diagram depicted by Fig. 1.

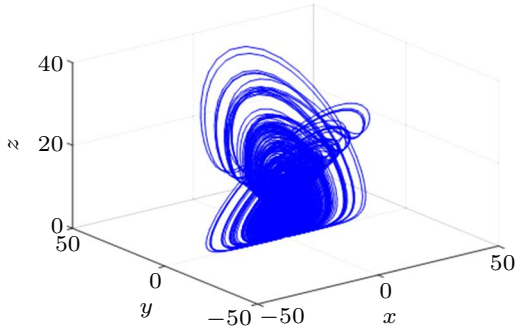


Fig. 1. Phase diagram of the augmented Lü system.

Based on the augmented Lü system, 4DALHS is designed by introducing a nonlinear state feedback controller w, in order to generate a hyperchaotic attractor with more complex dynamical behaviors and richer chaotic characteristics. The system is defined as follows:

$$\begin{cases} \dot{x} = ax + byz, \\ \dot{y} = cy + dxz + exw, \\ \dot{z} = fxy + gz, \\ \dot{w} = h(x + z), \end{cases} \tag{2}$$

where x, y, z, and w are the state variables of the system, a, b, c, d, e, f, g, and h are the parameters which control its chaotic behavior. If the system parameters are set to $a = 12$, $b = -0.1$, $c = -30$, $d = 1$, $e = 1.5$, $f = 5$, $g = -10$, and $h = -2$, 4DALHS exhibits hyperchaotic behavior. Given the initial values as $x_0 = 0.5$, $y_0 = 0.5$, $z_0 = 0.5$, and $w_0 = 0.5$, the 3D phase diagrams of 4DALHS are illustrated in Fig. 2. These dense and tangled trajectories exhibit the system's highly complex dynamic behavior.

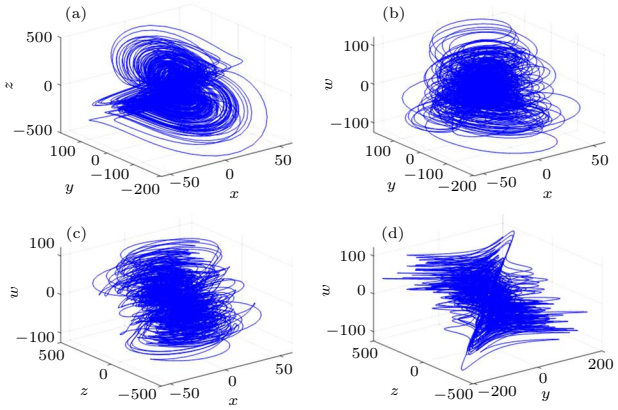


Fig. 2. Phase diagrams of the proposed 4DALHS: (a) x–y–z, (b) x–y–w, (c) x–z–w, and (d) y–z–w.

2.2. Bifurcation diagram and Poincaré section analysis

A bifurcation diagram can visually reveal the dynamic behavior of a system under different control parameters. In a dynamical system, when the control parameters change, the dynamic behavior of the system will change significantly.[29] The system parameters are set to $(a,b,c,d,e,f,g,h) = (12,-0.1,-30,1,1.5,5,-10,-2)$. Figure 3 depicts how the system variable x evolves with changes of parameters a, b, c, d, e, f, g, and h, with seven parameters kept fixed while another is varied over a specific range. These results clearly indicate that 4DALHS is capable of generating chaotic behavior across a broad range of parameter values.

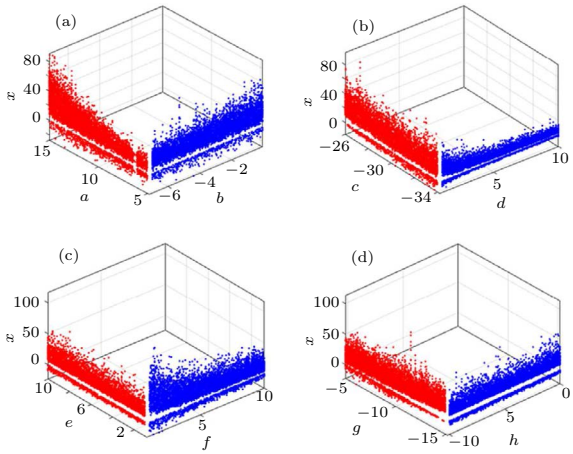


Fig. 3. Bifurcation diagram of the sequence x: (a) when $a \in [5,15]$ and $b \in [-7,0]$ respectively; (b) when $c \in [-34,-26]$ and $d \in [1,10]$ respectively; (c) when $e \in [1,10]$ and $f \in [1,10]$ respectively; (d) when $g \in [-15,-5]$ and $h \in [-10,0)$ respectively.

Poincaré section is an important tool for analyzing the behavior of a dynamical system. If a system exhibits chaotic behavior, its Poincare section typically consists of a complex set of points with a fractal structure. As shown in Fig. 4, if the system parameters are set to $(a,b,c,d,e,f,g,h) = (12,-0.1,-30,1,1.5,5,-10,-2)$, a symmetric four-wing structure is clearly observed on the $z = 0$ plane, which means that 4DALHS belongs to a class of four-wing chaotic attractors.

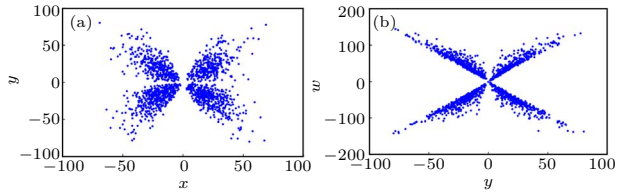


Fig. 4. Poincaré section diagram of 4DALHS on the $z = 0$ plane: (a) x–y, (b) y–w.

2.3. Dissipativity analysis

The dissipativity of 4DALHS is determined by

$$\nabla V = \frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{y}}{\partial y} + \frac{\partial \dot{z}}{\partial z} + \frac{\partial \dot{w}}{\partial w}. \tag{3}$$

For the proposed 4DALHS with parameters $a = 12$, $b = -0.1$, $c = -30$, $d = 1$, $e = 1.5$, $f = 5$, $g = -10$, and $h = -2$, the dissipativity can be calculated as $\nabla V = a + c + g = -28 < 0$, indicating that 4DALHS is dissipative and converges exponentially with rate $\mathrm{d}V/\mathrm{d}t = \mathrm{e}^{-(a+c+g)t}$. As $t \to \infty$, every volume element containing system trajectories contracts to zero at an exponential rate. Thus, all motions of 4DALHS will settle onto an attractor.

2.4. Equilibrium and stability analysis

To find the equilibrium points of 4DALHS, setting $\dot{x} = \dot{y} = \dot{z} = \dot{w} = 0$, as shown by

$$\begin{cases} ax + byz = 0, \\ cy + dxz + exw = 0, \\ fxy + gz = 0, \\ h(x+z) = 0. \end{cases} \tag{4}$$

Solving Eq. (4) shows that 4DALHS has a unique equilibrium point $S_0 = (0,0,0,0)$.Then, 4DALHS is linearized to get a Jacobian matrix as

$$J = \begin{bmatrix} a & bz & by & 0 \\ dz+ew & c & dx & ex \\ fy & fx & g & 0 \\ h & 0 & h & 0 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & g & 0 \\ h & 0 & h & 0 \end{bmatrix}.$$

After computing $\det(J - \lambda E) = 0$, eigenvalues of J are $\lambda_1 = a$, $\lambda_2 = c$, and $\lambda_3 = g$. If setting $a > 0$, 4DALHS becomes unstable at the equilibrium point, indicating a chaotic behavior.

2.5. Lyapunov exponent

Lyapunov exponent is a quantitative tool of sensitivity analysis to initial conditions, and it is particularly useful for characterizing chaotic behavior in nonlinear systems. A positive Lyapunov exponent indicates sensitive dependence on initial conditions, that is to say, any small changes in initial values will lead to rapid divergence of trajectories, signifying chaotic behavior. As shown in Fig. 5(a), with parameters $a = -10$ and $b = -4$, Lyapunov exponents of augmented Lü system are $LE_1 = 1.1474$, $LE_2 = -0.0128$, and $LE_3 = -12.2872$. As shown in Fig. 5(b), with parameters $a = 12$, $b = -0.1$, $c = -30$, $d = 1$, $e = 1.5$, $f = 5$, $g = -10$, and $h = -2$, Lyapunov exponents of 4DALHS are $LE_1 = 4.4990$, $LE_2 = 0.9925$, $LE_3 = -0.0057$, and $LE_4 = -32.9856$. Compared with the augmented Lü system, 4DALHS exhibits two positive Lyapunov exponents, indicating its hyperchaotic nature. So, a higher degree of dynamical complexity and increased sensitivity to initial conditions can be achieved, which can lead to a superior performance in terms of unpredictability, pseudo-randomness and overall complexity.

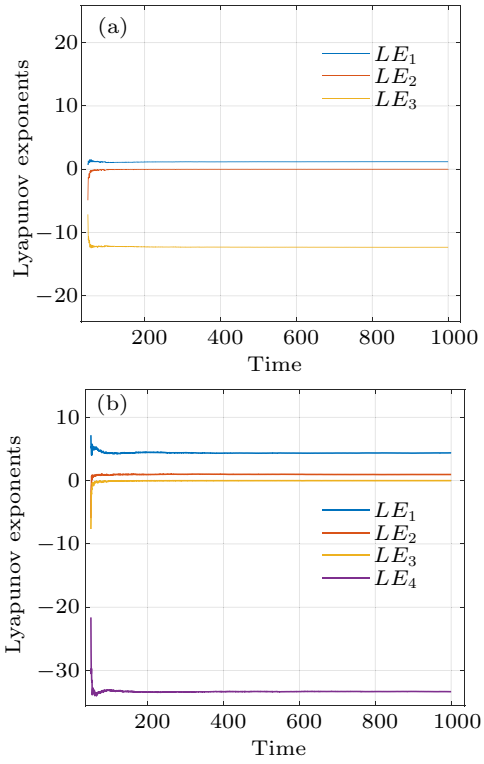


Fig. 5. Lyapunov exponents for: (a) augmented Lü system, (b) 4DALHS, where the horizontal axis represents the iteration time with a time step of 0.01, and the vertical axis represents the magnitude of the Lyapunov exponents.

2.6. NIST randomness analysis

NIST randomness test suite is a set of standardized methods used to evaluate the randomness for a sequence, aiming to determine whether the sequence exhibits sufficient random characteristics. Table 1 lists the tests for sequences generated by 4DALHS. So, these results indicate that the sequences can pass all individual tests, demonstrating that the generated sequences possess a high degree of randomness and unpredictability.

Table 1. NIST tests for 4DALHS.

Statistical test	*P*-value				Result
	X	*Y*	*Z*	*W*	
The frequency (Monobit) test	0.1443	0.4852	0.5809	0.8855	pass
Frequency test within a block	0.0774	0.2224	0.4072	0.9469	pass
The runs test	0.4741	0.4949	0.2249	0.7429	pass
Tests for the longest run of ones in a block	0.4872	0.0667	0.0958	0.2087	pass
The binary matrix rank test	0.0465	0.1514	0.2106	0.0192	pass
The discrete fourier transform (spectral) test	0.3683	0.1389	0.1105	0.9537	pass
The non-overlapping template matching test	0.6931	0.9249	0.0547	0.6611	pass
The overlapping template matching test	0.2681	0.7228	0.9228	0.1405	pass
Maurer's "universal statistical" test	0.4720	0.0294	0.5349	0.4644	pass
The linear complexity test	0.9422	0.9406	0.9044	0.2841	pass
The serial test	0.0118	0.6434	0.3145	0.9382	pass
The approximate entropy test	0.3753	0.6663	0.9260	0.2141	pass
Cusums-forward	0.5666	0.9675	0.7967	1.0000	pass
Cusums-reverse	0.9486	1.0000	0.2105	1.0000	pass
The random excursions test	0.1777	0.8540	0.1671	0.6578	pass
The random excursions variant test	0.5175	0.7794	0.5874	0.1522	pass

2.7. 0–1 test

As it is known that 0–1 test is a statistical evaluation method to assess the randomness of a binary sequence. If test sequence approximates a truly random case, 0–1 test will typically exhibit irregular Brownian motion.[7] When the initial conditions are set to $(a,b,c,d,e,f,g,h,x_0,y_0,z_0,w_0) = (12,-0.1,-30,1,1.5,5,-10,-2,0.5,0.5,0.5,0.5)$, figure 6 shows the 0–1 test results of the four sequences generated by 4DALHS. It can be observed that each plot shows highly complex and irregular trajectory distributions, indicating that 4DALHS system can generate high unpredictable and random sequences.

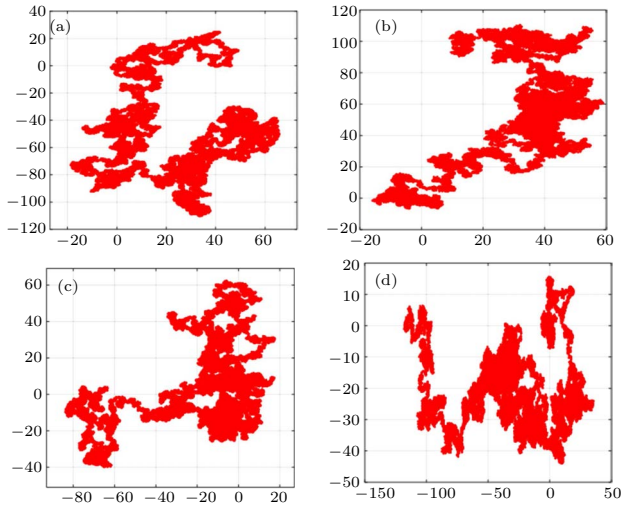


Fig. 6. 0–1 tests for 4DALHS: (a) sequence x, (b) sequence y, (c) sequence z, (d) sequence w.

3. Proposed CIEA-4DALHS

The overall encryption process of CIEA-4DALHS is displayed in Fig. 7. There are four main stages: key generation, bidirectional spiral cross scrambling, bit-plane substitution, and hierarchical regional segmentation diffusion. All the random sequences required during the encryption process are generated by 4DALHS, with encryption effect achieved through alternating scrambling and diffusion operations.

3.1. Key generation

CIEA-4DALHS employs four secret keys, comprising initial values of 4DALHS. They are produced by the plain image by hash function SHA-256. The detailed procedure is as follows:

Step 1 The plain image P is processed by the SHA-256 hash function to generate a 256-bit hash key K, which is then divided into 32 groups of 8 bits length, i.e., $k_1, k_2, k_3, \ldots, k_{32}$. Subsequently, 32 new 32-bit sequences are obtained by concatenating every four successive 8-bit segments with a one-segment shifting: $l_1 = k_1k_2k_3k_4$, $l_2 = k_2k_3k_4k_5$, $l_3 = k_3k_4k_5k_6$, $\ldots$, $l_{32} = k_{32}k_1k_2k_3$.

Step 2 Four parameters h_1, h_2, h_3, and h_4 are derived from $l_1, l_2, l_3, \ldots, l_{32}$ by

$$\begin{cases} h_1 = l_1 \oplus l_2 \oplus l_3 \oplus l_4 \oplus l_5 \oplus l_6 \oplus l_7 \oplus l_8, \\ h_2 = l_9 \oplus l_{10} \oplus l_{11} \oplus l_{12} \oplus l_{13} \oplus l_{14} \oplus l_{15} \oplus l_{16}, \\ h_3 = l_{17} \oplus l_{18} \oplus l_{19} \oplus l_{20} \oplus l_{21} \oplus l_{22} \oplus l_{23} \oplus l_{24}, \\ h_4 = l_{25} \oplus l_{26} \oplus l_{27} \oplus l_{28} \oplus l_{29} \oplus l_{30} \oplus l_{31} \oplus l_{32}, \end{cases} \tag{5}$$

where $\oplus$ denotes the bitwise XOR operation, and h_i $(i = 1,2,3,4) \in (0, 2^{32}-1)$.

Step 3 Initial values x_0, y_0, z_0 and w_0 are calculated respectively according to

$$\begin{cases} x_0 = \left(\dfrac{h_1 \oplus h_2}{2^{32}-1}\right) \bmod 1, \quad y_0 = \left(\dfrac{h_3 \oplus h_4}{2^{32}-1}\right) \bmod 1, \\ z_0 = \left(\dfrac{h_1 \oplus h_3}{2^{32}-1}\right) \bmod 1, \quad w_0 = \left(\dfrac{h_2 \oplus h_4}{2^{32}-1}\right) \bmod 1. \end{cases} \tag{6}$$

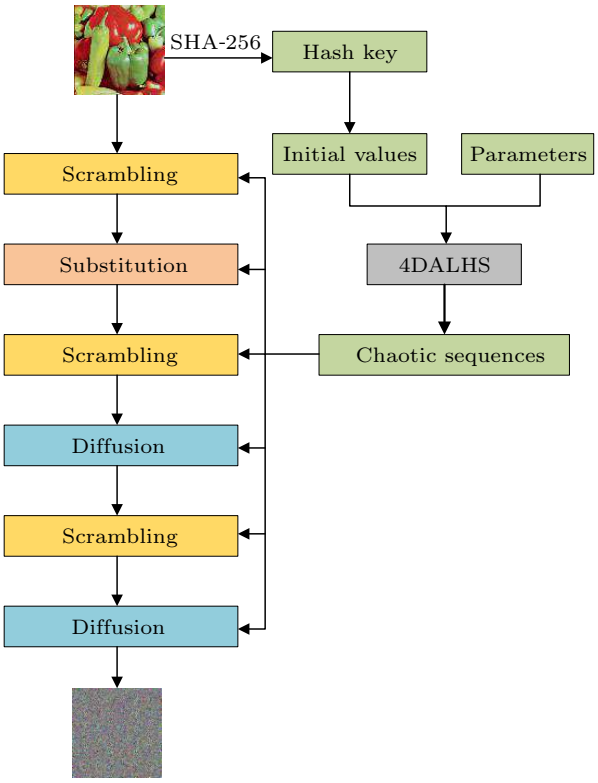


Fig. 7. Encryption flowchart.

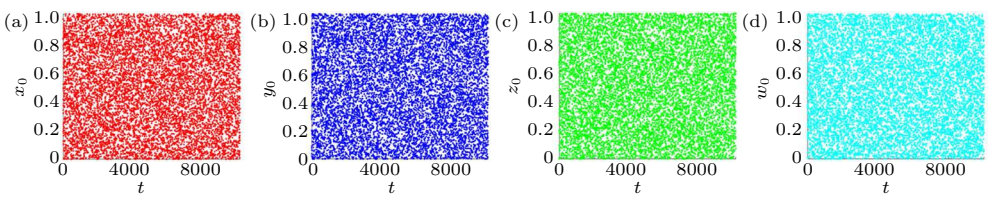


Fig. 8. Distributions: (a) x_0, (b) y_0, (c) z_0, (d) w_0.

After modulo operation, the initial values can be uniformly distributed within the interval [0,1]. To further evaluate their distributions, 10^4 matrices of size 64×64 are randomly generated for tests. As shown in Fig. 8, x_0, y_0, z_0, and w_0 are uniformly distributed within the range [0,1].

3.2. Bidirectional spiral cross scrambling

Many conventional spiral scrambling methods suffer from limited security and are vulnerable to recognition and decryption by attackers. To address this issue, this paper proposes a bidirectional spiral cross scrambling method. The method employs a bidirectional spiral traversal mechanism with four distinct traversal modes. In each mode, two traversal directions are available, producing two spiral sequences. These two sequences are then alternately merged to form the final scrambling sequence, which is reshaped into a matrix of the same size as the original image. Both the spiral's starting point and the traversal mode are dynamically determined by a chaotic sequence generated from 4DALHS.

The selection of the starting point (i, j) is determined by

$$\begin{cases} i = \text{floor}(\bmod(abs(L \times 10^8), M)), \\ j = \text{floor}(\bmod(abs(L \times 10^8), N)), \end{cases} \tag{7}$$

where L means a chaotic value generated by 4DALHS, M and N denote the size of the image. If the starting point falls on the edge or corner of the image, it must be adjusted accordingly, as described by

$$(i', j') = \begin{cases} (i, j+1), & \text{if } 1 < i < M \text{ and } j = 1, \\ (i+1, j+1), & \text{if } i = 1 \text{ and } j = 1, \\ (i, j-1), & \text{if } 1 < i < M \text{ and } j = N, \\ (i+1, j-1), & \text{if } i = 1 \text{ and } j = N, \\ (i+1, j), & \text{if } i = 1 \text{ and } 1 < j < N, \\ (i-1, j+1), & \text{if } i = M \text{ and } j = 1, \\ (i-1, j), & \text{if } i = M \text{ and } 1 < j < N, \\ (i-1, j-1), & \text{if } i = M \text{ and } j = N, \\ (i, j), & \text{otherwise}, \end{cases} \tag{8}$$

where $2 \le i' \le M-1$ and $2 \le j' \le N-1$. As a result, all selected starting points are ensured to lie within the interior of the image, thus allowing the bidirectional spiral traversal to be performed directly without encountering boundary-related issues.

There are four distinct modes of bidirectional spiral traversal. Taking a 5×5 matrix with the center element as the spiral starting point as an example, four modes are illustrated in Fig. 9. During the encryption process, the specific traversal mode is determined by

$$T = \text{floor}\left(abs(L \times 10^4)\right) \bmod 4 + 1, \tag{9}$$

where L means a chaotic value generated by 4DALHS. Then mode (a) is selected if $T = 1$ is got. Similarly, modes (b), (c), and (d) will be chosen if $T = 2$, $T = 3$, and $T = 4$ are got respectively.

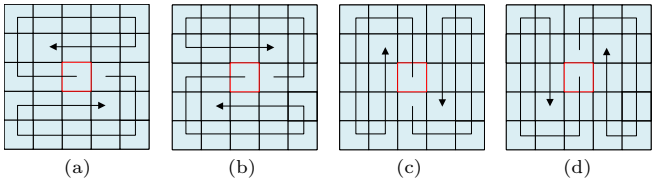


Fig. 9. Illustration of the four modes of bidirectional spiral traversal.

Taking mode (b) as an example, the bidirectional spiral traversal paths are illustrated when the selected starting point is located at the top-left corner, bottom-right corner, first column and last row, respectively, as shown in Fig. 10.

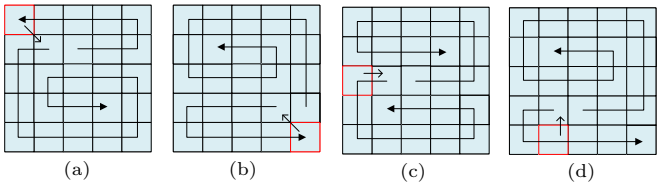


Fig. 10. Adjustment method for bidirectional spiral traversal paths under mode (b) when the starting point is on the image's edge or corner: (a) top-left corner, (b) bottom-right corner, (c) first column, (d) last row.

Figure 11 illustrates a simple example of the implementation of the bidirectional spiral cross scrambling on a 7×7 matrix using mode (b). A random element at coordinate (4, 5), with a value of 26, is selected as the spiral starting point. The row containing this element serves as the dividing line, splitting the matrix into upper and lower parts. The upper part begins a clockwise spiral traversal from the next element 27, proceeding from the outer layer inward to generate a sequence Seq 1. The lower part starts a counterclockwise inward spiral from the starting element to produce a sequence Seq 2. These two sequences are then alternately interleaved to form a new sequence Seq 3. When all elements in either sequence have been extracted, the remaining elements from the other sequence are appended to the end of Seq 3. Finally, Seq 3 is reshaped into a matrix of the same size as the original one, completing the permutation process. As shown in Fig. 11, the matrix processed by bidirectional spiral cross scrambling demonstrates significant scrambling effects, effectively disrupting the original pixel distribution and substantially enhancing the security of the encrypted image.

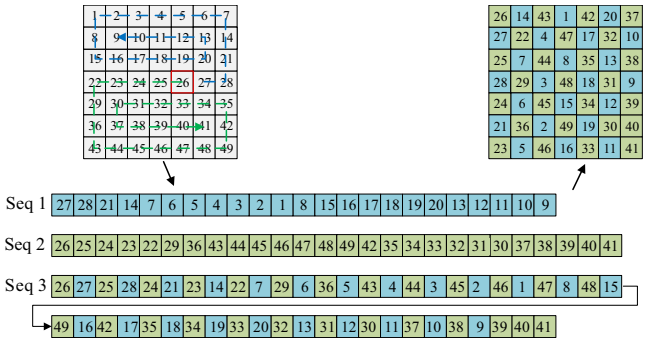


Fig. 11. Simple example of the bidirectional spiral cross scrambling.

Compared with conventional single-direction spiral scrambling, the proposed bidirectional spiral cross scrambling

introduces two main improvements. First, the dynamic selection of starting points and traversal modes, driven by chaotic sequences, prevents predictable patterns that attackers could exploit. Second, the use of two opposite spiral directions and their interleaved merging significantly increases the randomness and complexity of the scrambling sequence. These design decisions jointly improve the overall security of the encryption scheme.

3.3. Bit-plane substitution

Most existing image encryption schemes focus on manipulating plain images at the pixel level, often overlooking the complex hierarchical structure inherent within individual pixels. Therefore, this paper designs a bit-level substitution method. By operating on each bit that constitutes a pixel value, the method enables more fine-grained and deeper data scrambling.

Since pixels are typically integers ranging from 0 to 255, each image matrix can be decomposed into 8 bit-planes. Given an original image matrix of size $M \times N$, it can be decomposed into eight bit-plane matrices, each of size $M \times N$. Each bit-plane requires a substitution matric of size $M \times N$ and these substitution matrices are determined by a chaotic sequence generated by 4DALHS, namely, sequence S with length of $M \times N$. It is taken to determine a unique substitution order for the eight bit-planes of each pixel position. Based on this, eight corresponding substitution matrices are constructed. Then, all bit values at different positions will be reassigned during substitution. This substitution mechanism ensures that the bit values are thoroughly reassigned across all bit-planes, thereby altering the pixel values. As a result, it can achieve both fine-grained, bit-level permutation and encryption. Specific rule is given by

$$\begin{cases} [1,2,3,4,5,6,7,8], & \text{if } S' = 1, \\ [2,3,4,5,6,7,8,1], & \text{if } S' = 2, \\ [3,4,5,6,7,8,1,2], & \text{if } S' = 3, \\ [4,5,6,7,8,1,2,3], & \text{if } S' = 4, \\ [5,6,7,8,1,2,3,4], & \text{if } S' = 5, \\ [6,7,8,1,2,3,4,5], & \text{if } S' = 6, \\ [7,8,1,2,3,4,5,6], & \text{if } S' = 7, \\ [8,1,2,3,4,5,6,7], & \text{if } S' = 8, \end{cases} \tag{10}$$

where S' denotes a sequence after process to S, i.e., $S' = \text{floor}(\bmod(abs(S \times 10^8), 8)) + 1$.

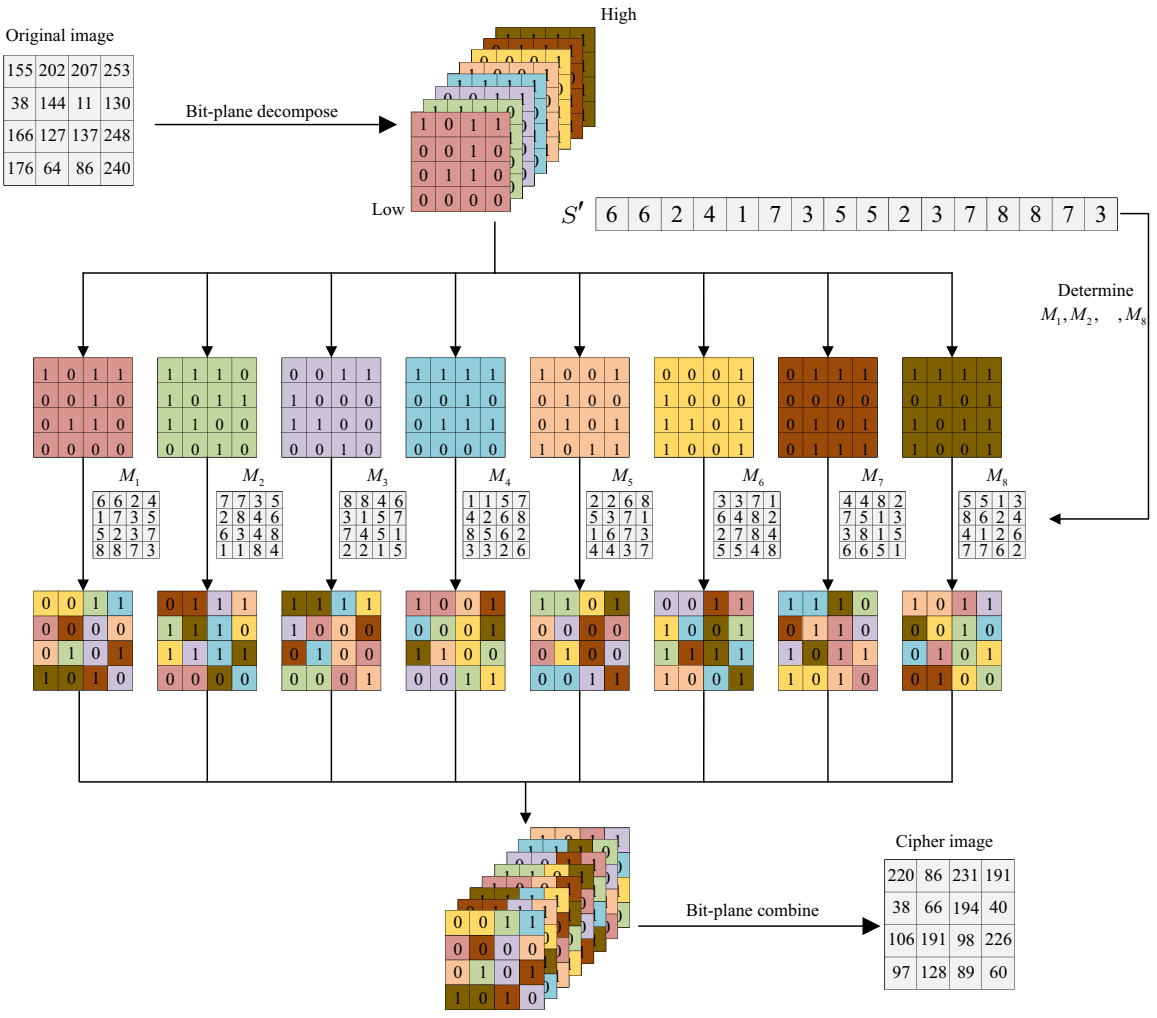


Fig. 12. Simple example of the bit-plane substitution process with a 4×4 image matrix.

As shown in Fig. 12, suppose sequence S' is [6,6,2,4,1,7,3,5,5,2,3,7,8,8,7,3] generated by 4DALHS, with a length of $4 \times 4 = 16$. Since $S'(1) = 6$, the sequence [6,7,8,1,2,3,4,5] is obtained. These eight numbers are sequentially assigned to the first position of the eight substitution matrices $M_1, M_2, \ldots, M_8$. By repeating this procedure for all subsequent elements, eight complete substitution matrices are eventually constructed. $M_1, M_2, \ldots, M_8$. During the substitution process, since the first element of the substitution matrix for the first bitplane is 6, the bit value at the first position of the sixth bitplane is substituted into the first bitplane at the same position. Similarly, the first element of the substitution matrix for the second bitplane is 7, thus the bit value from the seventh bitplane is substituted into the second bitplane at the same position, and so on, until all positions of all bit planes have been substituted. Figure 12 shows a complete example of a 4×4 image matrix using the proposed bit-plane substitution method. It can be observed that nearly all pixel values are altered after decomposition and substitution process.

3.4. Hierarchical regional segmentation diffusion

In image encryption, diffusion is a critical step to ensure the algorithm's resistance against differential attacks, aiming to propagate minor changes in a single plaintext pixel throughout the entire cipher image. However, many existing diffusion methods suffer from two major limitations: a single fixed diffusion path and mutually independent diffusion across channels. To address these issues, a hierarchical regional segmentation diffusion method is proposed in this paper. The core motivation of this approach is to enhance the complexity and thoroughness of diffusion through a dual-layer design: "segmentation-multipath" and "cross-channel cascade".

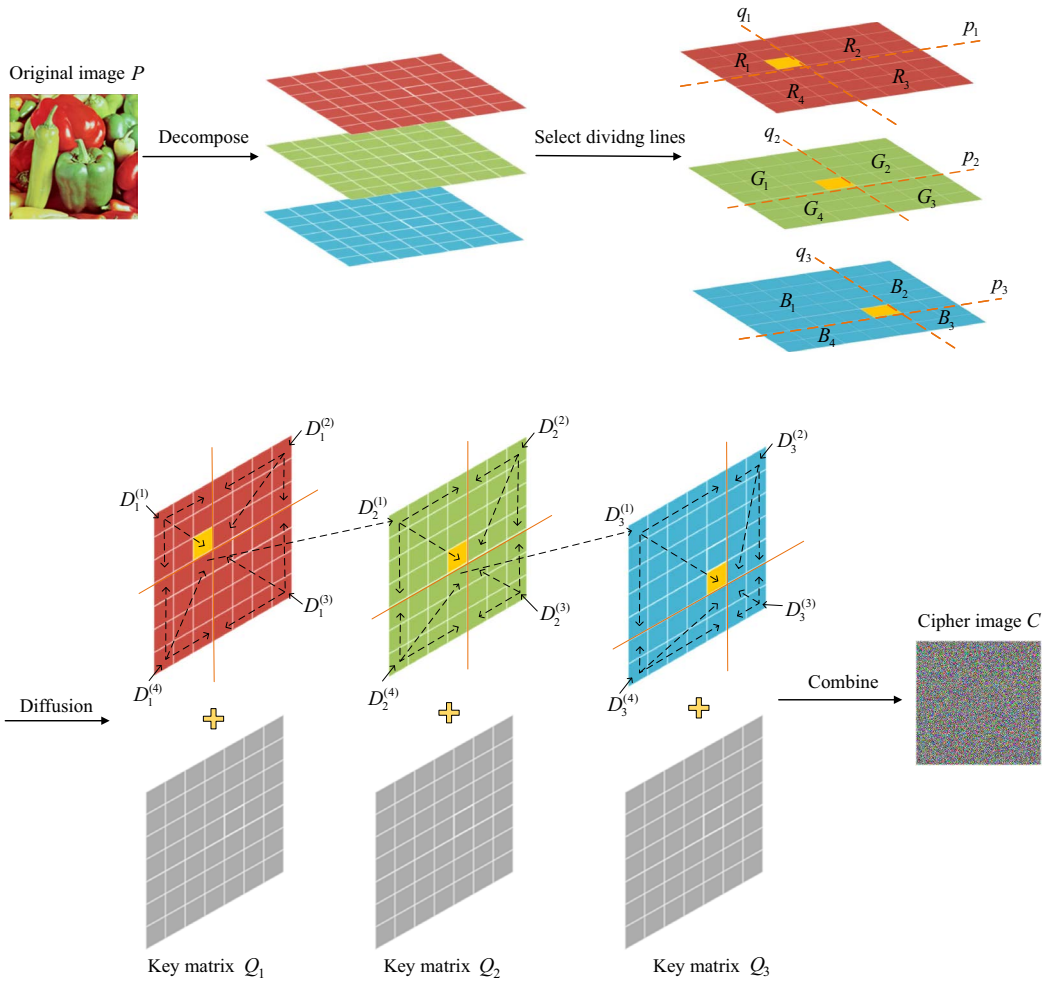


Fig. 13. Process of hierarchical regional segmentation diffusion.

Figure 13 illustrates the process of the proposed hierarchical regional segmentation diffusion method. Specifically, a color image of size $M \times N$ is first decomposed into three separate channels, i.e., R, G, and B. Each channel is then partitioned into four distinct regions based on dividing lines. Let the dividing lines for the R, G, and B channels be defined as follows: R channel is divided by lines p_1 and q_1, G channel by lines p_2 and q_2, and B channel by lines p_3 and q_3. The R, G, and B channels are randomly segmented. The four regions of the Rchannel are labeled as R_1, R_2, R_3, and R_4; The regions for Gchannel are labeled as G_1, G_2, G_3, and G_4; The regions for B channel are labeled as B_1, B_2, B_3, and B_4. They all adopt an

outward-to-inward diffusion approach. The diffusion process commences with the R channel, where its four regions are processed sequentially: $R_1 \to R_2 \to R_3 \to R_4$. After diffusion process within each region, the final pixel value is used as part of the initial condition for the diffusion of the subsequent region. Similarly, the last pixel of each channel's diffusion is incorporated into the initialization of the next channel's diffusion process. Subsequently, the same procedure is applied to the G channel, concluding with the B channel. To further enhance the security, a chaotic sequence generated by 4DALHS is used to construct a key matrix Q of size $M \times N \times 3$. This matrix is then employed in a modular addition process with a modulus of 256, to effectively perturb the original image during the diffusion process, thereby increasing the unpredictability and the attack resistance.

As shown in Fig. 13, the diffusion operations for the three color channels share a similar structure. To clearly denote the initial diffusion value for each region under each channel, define the symbol $D_k^{(v)}$, where the superscript $v \in \{1,2,3,4\}$ represents the region index, and the subscript $k \in \{1,2,3\}$ denotes the R, G, B channels, respectively. The diffusion for regions R_1, G_1, and B_1 is given by

$$\begin{cases} C(i,j,k) = \left(P(i,j,k) + Q(i,j,k) + D_k^{(1)}\right) \bmod 256, & \text{if } i=1,\ j=1, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i,j-1,k)) \bmod 256, & \text{if } i=1,\ 1<j\le q_k, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i-1,j,k)) \bmod 256, & \text{if } 1<i\le p_k,\ j=1, \\ C(i,j,k) = \begin{pmatrix} P(i,j,k) + Q(i,j,k) + C(i,j-1,k) \\ +C(i-1,j,k) + C(i-1,j-1,k) \end{pmatrix} \bmod 256, & \text{if } 1<i\le p_k,\ 1<j\le q_k, \end{cases} \tag{11}$$

where $P(i,j,k)$ represents the pixel value of the original image P, $C(i,j,k)$ represents the pixel value of the cipher image C, and $Q(i,j,k)$ represents the pixel value of the key matrix Q. The initial conditions for the first region of each color channel establish the inter-channel dependency. For region R_1, the initial condition $D_1^{(1)}$ is derived from the chaotic values generated by 4DALHS as follows:

$$D_1^{(1)} = \text{floor}(\bmod(abs(w_1 + w_2 + w_3 + w_4), 256)). \tag{12}$$

Subsequently, the initial condition $D_2^{(1)}$ for region G_1 is set to $C(p_1+1, q_1, 1)$, which is the final pixel of the R-channel's diffusion. Similarly, the initial condition $D_3^{(1)}$ for region B_1 is set to $C(p_2+1, q_2, 1)$, which is the final pixel of the G-channel's diffusion.

The diffusion for regions R_2, G_2, and B_2 is given by

$$\begin{cases} C(i,j,k) = \left(P(i,j,k) + Q(i,j,k) + D_k^{(2)}\right) \bmod 256, & \text{if } i=1,\ j=N, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i,j+1,k)) \bmod 256, & \text{if } i=1,\ q_k\le j<N, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i-1,j,k)) \bmod 256, & \text{if } 1<i\le p_k,\ j=N, \\ C(i,j,k) = \begin{pmatrix} P(i,j,k) + Q(i,j,k) + C(i,j+1,k) \\ +C(i-1,j,k) + C(i-1,j+1,k) \end{pmatrix} \bmod 256, & \text{if } 1<i\le p_k,\ q_k\le j<N. \end{cases} \tag{13}$$

The initial conditions for the second region are derived from the first region of their respective channels, creating an intra-channel cascade. Specifically, for each channel, the initial value $D_k^{(2)}$ is set to $C(p_k, q_k, k)$, which is the final pixel value from the completed diffusion of the first region of the corresponding channel.

The diffusion for regions R_3, G_3, and B_3 is given by

$$\begin{cases} C(i,j,k) = \left(P(i,j,k) + Q(i,j,k) + D_k^{(3)}\right) \bmod 256, & \text{if } i=M,\ j=N, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i,j+1,k)) \bmod 256, & \text{if } i=M,\ q_k<j\le N, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i+1,j,k)) \bmod 256, & \text{if } p_k<i\le M,\ j=N, \\ C(i,j,k) = \begin{pmatrix} P(i,j,k) + Q(i,j,k) + C(i,j+1,k) \\ +C(i+1,j,k) + C(i+1,j+1,k) \end{pmatrix} \bmod 256, & \text{if } p_k<i\le M,\ q_k<j\le N. \end{cases} \tag{14}$$

Similarly, the initial conditions for the third region depend on the results from the second region, continuing the intra-channel cascade. For each channel, the initial value $D_k^{(3)}$ is set to $C(p_k, q_k+1, k)$, which is the final pixel value from the completed diffusion of the second region of the corresponding channel.

The diffusion for regions R_4, G_4, and B_4 is given by

$$\begin{cases} C(i,j,k) = \left(P(i,j,k) + Q(i,j,k) + D_k^{(4)}\right) \bmod 256, & \text{if } i=M,\ j=1, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i,j-1,k)) \bmod 256, & \text{if } i=M,\ 1<j\le q_k, \\ C(i,j,k) = (P(i,j,k) + Q(i,j,k) + C(i+1,j,k)) \bmod 256, & \text{if } p_k<i\le M,\ j=1, \\ C(i,j,k) = \begin{pmatrix} P(i,j,k) + Q(i,j,k) + C(i,j-1,k) \\ +C(i+1,j,k) + C(i+1,j-1,k) \end{pmatrix} \bmod 256, & \text{if } p_k<i\le M,\ 1<j\le q_k. \end{cases} \tag{15}$$

Finally, the initial values for the fourth region are defined to complete the intra-channel dependency chain. For each channel, the initial value $D_k^{(4)}$ is set to $C(p_k+1, q_k+1, k)$, which is the final pixel value from the completed diffusion of the third region for the corresponding channel.

3.5. Encryption and decryption processes

CIEA-4DALHS is specifically designed for color images in this paper. Since a grayscale image can be regarded as a special case of a color image with only one channel, the algorithm can also be adapted to grayscale images with minor modifications. The encryption process described in this Section takes the case of color image as an example.

3.5.1. Encryption

For an input color image P of size $M \times N \times 3$, the encryption process proceeds as follows:

Step 1 Perform key generation operation described in Subsection 3.1 on the color image P to get all parameters and initial values required by 4DALHS.

Step 2 Generate chaotic sequences using 4DALHS. After iterating $1009 + M \times N \times 2$ times and discarding the first 1000 values, four chaotic sequences L_1, L_2, L_3, and L_4 with length $9 + M \times N \times 2$ can be obtained.

Step 3 Decompose color image P into R, G, and B channels. Then, determine a spiral starting point and a mode of bidirectional spiral traversal for each R, G, and B by

$$\begin{cases} \text{row}_i = \text{floor}(\bmod(abs(L_i(1) \times 10^8), M)), \\ \text{col}_i = \text{floor}(\bmod(abs(L_i(2) \times 10^8), N)), \\ T_i = \text{floor}(\bmod(abs(L_i(3) \times 10^4, 4)) + 1, \end{cases} \quad i = 1,2,3. \tag{16}$$

Then, perform bidirectional spiral cross scrambling operation as described in Subsection 3.2.

Step 4 Process the chaotic sequences L_1, L_2, and L_3 by

$$\begin{aligned} &LL_i = \text{floor}(\bmod(abs(L_i(j) \times 10^8), 8)) + 1, \\ &i = 1,2,3; \quad j = 10, \ldots, M \times N + 10. \end{aligned} \tag{17}$$

Then, obtain LL_1, LL_2, and LL_3, respectively. According to Equation (**??**), generate 24 bit-plane substitution matrices M_1, M_2, …, M_{24} based on LL_1, LL_2, and LL_3. So, the 8 substitution matrices derived from LL_1, LL_2, LL_3 are used for the R, G, and B channels respectively. Finally, apply the bit-plane substitution operation described in Subsection 3.3.

Step 5 Horizontally combine R, G, and B to obtain a matrix A_1 of size $M \times 3N$. Determine spiral starting point $(\text{row}_4, \text{col}_4)$ for matrix A_1, and set the bidirectional spiral traversal mode to T_4, as follows:

$$\begin{cases} \text{row}_4 = \text{floor}(\bmod(abs(L_4(1) \times 10^8), M)), \\ \text{col}_4 = \text{floor}(\bmod(abs(L_4(2) \times 10^8), 3N)), \\ T_4 = \text{floor}(\bmod(abs(L_4(3) \times 10^4), 4)) + 1. \end{cases} \tag{18}$$

Then, perform the bidirectional spiral cross scrambling operation. Finally, reshape the scrambled matrix into a matrix P_1 of size $M \times N \times 3$.

Step 6 Obtain the key matrix Z of size $M \times N \times 3$ and the initial diffusion condition C_1 as

$$\begin{cases} LS_1 = \text{floor}(\bmod(abs(L_4(j) \times 10^8), 256)), \\ \quad j = 10, \ldots, M \times N + 9, \\ LS_2 = \text{floor}(\bmod(abs(L_4(j) \times 10^8), 256)), \\ \quad j = M \times N + 10, \ldots, 2M \times N + 9, \\ LS_3 = \text{floor}(\bmod(LS_1 + LS_2), 256)), \\ Z_1 = \text{reshape}(LS_1, M, N), \\ Z_2 = \text{reshape}(LS_2, M, N), \\ Z_3 = \text{reshape}(LS_3, M, N), \\ Z = \text{combine}(Z_1, Z_2, Z_3), \\ C_1 = \text{floor}(\bmod(abs(L_1(4) + L_2(4) + L_3(4) + L_4(4)), 256)), \end{cases} \tag{19}$$

where $\text{combine}(Z_1, Z_2, Z_3)$ denotes the operation of combining Z_1, Z_2, and Z_3 into a 3D matrix. Then, compute the dividing lines p_i and q_i for each color channel as:

$$\begin{cases} p_i = \text{floor}(\bmod(abs(L_i(5) \times 10^8), M)), \\ q_i = \text{floor}(\bmod(abs(L_i(6) \times 10^8), N)), \end{cases} \quad i = 1,2,3. \tag{20}$$

After that, apply the hierarchical regional segmentation diffusion, as described in Subsection 3.4, to the matrix P_1 to obtain the matrix P_2.

Step 7 Decompose matrix P_2 into R, G, and B channels and then vertically set to form a matrix A_2 of size $3M \times N$. Let spiral starting point for matrix A_2 be $(\text{row}_5, \text{col}_5)$, and set the bidirectional spiral traversal mode to T_5 by

$$\begin{cases} \text{row}_5 = \text{floor}(\bmod(abs(((L_1(1) + L_2(1) \\ \qquad + L_3(1) + L_4(1)))/4 \times 10^8), 3M)), \\ \text{col}_5 = \text{floor}(\bmod(abs(((L_1(2) + L_2(2) \\ \qquad + L_3(2) + L_4(1)))/4 \times 10^8), N)), \\ T_5 = \text{floor}(\bmod(abs((L_1(3) + L_2(3) \\ \qquad + L_3(3) + L_4(3))/4 \times 10^4), 4)) + 1. \end{cases} \tag{21}$$

Bidirectional spiral cross scrambling operation is then performed. Finally, the scrambled matrix is reshaped into a matrix P_3 of size $M \times N \times 3$.

Step 8 Obtain the key matrix Q of size $M \times N \times 3$ and the initial diffusion condition C_2 by

$$\begin{cases} LLL_1 = \text{floor}(\bmod(abs(L_1(j) \times 10^8), 256)), \\ LLL_2 = \text{floor}(\bmod(abs(L_2(j) \times 10^8), 256)), \\ \quad j = M \times N + 10, \ldots, 2M \times N + 9, \\ LLL_3 = \text{floor}(\bmod(abs(L_3(j) \times 10^8), 256)), \\ Q_1 = \text{reshape}(LLL_1, M, N), \\ Q_2 = \text{reshape}(LLL_2, M, N), \\ Q_3 = \text{reshape}(LLL_3, M, N), \\ Q = \text{combine}(Q_1, Q_2, Q_3), \\ C_2 = \text{floor}(\bmod(abs(L_1(7) + L_2(7) + L_3(7) + L_4(7)), 256). \end{cases} \tag{22}$$

Then, calculate the dividing lines p_i and q_i for each color channel by

$$\begin{cases} p_i = \text{floor}(\bmod(abs(L_i(8)\times 10^8), M)), \\ q_i = \text{floor}(\bmod(abs(L_i(9)\times 10^8), N)), \end{cases} \quad i = 4,5,6. \quad (23)$$

After that, apply the hierarchical regional segmentation diffusion to the matrix P_3 to obtain the final encrypted image P_4.

3.5.2. Decryption

Decryption process as illustrated in Fig. 14, accurate decryption of the cipher image can be achieved using the correct secret keys. Upon receiving the cipher image and the correct secret keys, the decryption side employs the 4DALHS to generate the same pseudo-random sequences as in the encryption stage. These sequences are then used in the subsequent decryption operations. The scrambling, diffusion, and bit-plane substitution performed during encryption are executed in reverse order during decryption. By carrying out these inverse operations, the original plain image can be fully recovered. Since the decryption process of CIEA-4DALHS is the exact inverse of the encryption process, the decryption procedure will not be reiterated here.

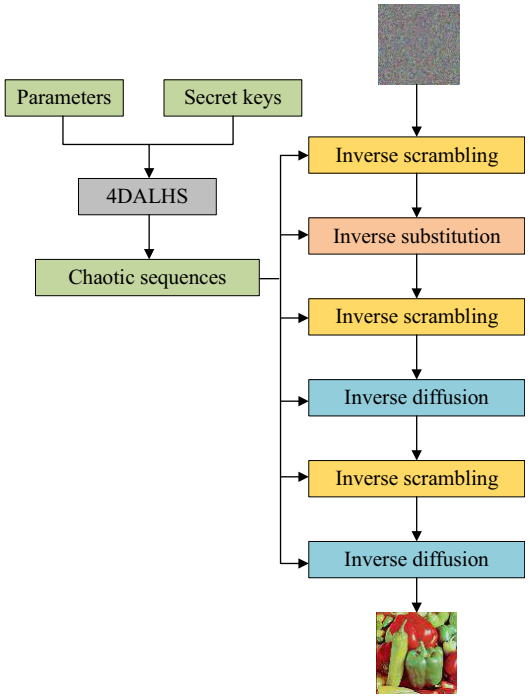


Fig. 14. Decryption flowchart.

4. Simulation results and security analysis

Experiments are conducted on a Windows 11 (64-bit) PC with an Intel i5-12600KF CPU and an Nvidia RTX 4060 (8 GB) GPU, by MATLAB 2023b. Test images include Boat (256×256×3), Pepper (512×512×3), SanFrancisco (1024×1024×3), Baboon (500×480×3), and GoldHill (720×576×3). As shown in Fig. 15, all images are successfully encrypted into noise-like outputs. Moreover, accurately decrypted can also be got using the correct key.

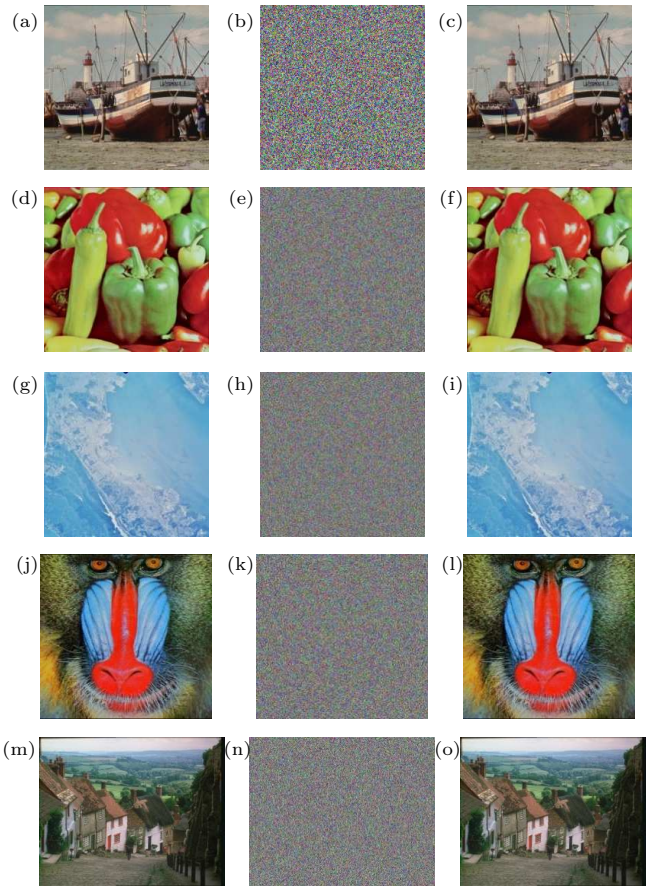


Fig. 15. Experiments: (a) Boat, (b) cipher image of panel (a), (c) decrypted image of panel (b), (d) Pepper, (e) cipher image of panel (d), (f) decrypted image of panel (e), (g) Sanfrancisco, (h) cipher image of panel (g), (i) decrypted image of panel (h), (j) Baboon, (k) cipher image of panel (j), (l) decrypted image of panel (k), (m) GoldHill, (n) cipher image of panel (m), (o) decrypted image of panel (n).

4.1. Key space analysis

Typically, an encryption algorithm should let its key space be large at least 2^{100} to be considered secure.[4] In the proposed CIEA-4DALHS, key space refers to the system initial values of 4DALHS, with a total of 4 keys. If the precision is 10^{-15}, the key space for CIEA-4DALHS is about

$$\left(10^{15}\right)^4 = 10^{60} = 2^{60\times\log_2(10)} \approx 2^{60\times 3.32} \approx 2^{199}. \quad (24)$$

So, it can highly resist to brute-force attacks for attackers. Table 2 also compares the key space of CIEA-4DALHS with other algorithms, clearly showing that CIEA-4DALHS is significantly superior to some others.

Table 2. Key space analysis..

Algorithm	Ours	Ref. [31]	Ref. [32]	Ref. [33]
Key Space	2^{199}	2^{99}	2^{121}	2^{160}

4.2. Information entropy analysis

Information entropy is an important metric for measuring the uncertainty of information within an image.[34] It is commonly used to evaluate the randomness of the pixel value distribution. The closer the entropy value is to 8, the better the

image exhibits unpredictability, uncertainty and randomness. The definition for information entropy is

$$H(X) = -\sum_{i=0}^{2^n-1} p(x_i)\log_2 p(x_i), \tag{25}$$

where variable n represents the grayscale level, typically set to 8. x_i denotes a possible pixel value, and $p(x_i)$ represents the probability of occurrence. Table 3 presents the information entropy values before and after encryption. As shown, the values for all three color channels are close to 8, indicating that CIEA-4DALHS significantly enhances the randomness. Table 4 compares the information entropy with other algorithms. So, these results show that information entropy obtained by CIEA-4DALHS is closer to 8, further demonstrating its superiority.

Furthermore, local information entropy (LIE) is used to evaluate the randomness of an image within local regions. The calculation involves randomly selecting several image blocks of equal size from the image, computing the information entropy of each block, then averaging these values to obtain the overall local information entropy. The calculation for LIE is given by

$$\overline{H_{(k,T_B)}(S)} = \sum_{i=1}^{k} \frac{H(S_i)}{k}, \tag{26}$$

where $H(S_i)$ denotes the information entropy of the i-th image block, k is the total number of blocks, and T_B represents the number of pixels in each block. When setting $k = 30$ and $T_B = 1936$ with the significance level $\alpha = 0.01$, the theoretical range for LIE is [7.901722822, 7.903215812].[34] As shown in Table 5, all cipher images fall within this ideal range, indicating that CIEA-4DALHS exhibit high local randomness for outputs.

Table 3. Information entropy tests.

Image size	Image	Plain image			Cipher image		
		R	*G*	*B*	*R*	*G*	*B*
$256\times256\times3$	Boat	7.2601	7.0553	6.8472	7.9971	7.9970	7.9974
$512\times512\times3$	Pepper	7.3388	7.4963	7.0583	7.9992	7.9993	7.9993
$512\times512\times3$	Lenna	5.0465	5.4576	4.8001	7.9993	7.9992	7.9993
$1024\times1024\times3$	SanFrancisco	7.1221	6.0810	4.5139	7.9998	7.9998	7.9998
$500\times480\times3$	Baboon	6.4998	6.4445	6.2709	7.9993	7.9992	7.9992
$720\times576\times3$	GoldHill	7.6101	7.5544	7.5540	7.9995	7.9996	7.9995

Table 4. Comparison of Pepper's information entropy.

Algorithm	Cipher image			
	R	*G*	*B*	Average
Ours	7.9992	7.9993	7.9993	7.9993
[34]	7.9969	7.9974	7.9974	7.9972
[35]	7.9982	7.9985	7.9984	7.9984
[36]	7.9970	7.9976	7.9967	7.9971

Table 5. LIE tests.

Ciher image	*R*	*G*	*B*	Average
Boat	7.9038	7.9006	7.9007	7.9017
Pepper	7.9037	7.9032	7.9008	7.9026
Lenna	7.9040	7.9006	7.9010	7.9018
SanFrancisco	7.9038	7.9027	7.9009	7.9024
Baboon	7.9023	7.9012	7.9020	7.9018
GoldHill	7.9038	7.9012	7.9029	7.9026

4.3. Histogram analysis

Histogram is usually employed to analyze pixel distribution.[6] Horizontal axis denotes pixel intensity levels, while vertical axis indicates the number of occurrences of each pixel value. Histogram of a plain image usually exhibits distinct distribution patterns. In contrast, an ideal encrypted image should have a histogram that approximates a uniform distribution, to avoid any meaningful information leakage from statistical analysis. As illustrated in Fig. 16, the histograms of five images, i.e., Boat, Pepper, SanFrancisco, Baboon, and GoldHill, are shown before and after encryption. Anyone can see that histograms of the plain images are highly uneven, with noticeable peaks and regularities. After encryption by CIEA-4DALHS, the histograms become uniformly distributed, with all gray levels appearing at roughly equal probabilities. Therefore, these results demonstrate that CIEA-4DALHS performs well in enhancing image security.

4.4. Chi-square analysis

Chi-square analysis is taken security field to evaluate the uniformity of pixel value distribution in an image.[7] By statistically examining the distribution of pixel values, it can determine whether the encryption process has effectively eliminated the original patterns. The calculation for Chi-square is given by

$$\chi^2 = \sum_{i=0}^{255} \frac{(O_i - E_i)^2}{E_i}, \tag{27}$$

where O_i represents the observed frequency for pixel i, and E_i is the expected frequency. In an ideally uniform distribution, all pixel values from 0 to 255 should occur with approximately equal frequency. So, E_i is expected to be close to $N/256$, where N is the total pixel number. A larger χ^2 value indicates a greater deviation. When the significance level is $\alpha = 0.05$, the critical value of χ^2 is 293.2478.[7] The cipher

image can be considered statistically uniform if its Chi-square is less than the critical value. Table 6 presents chi-square tests for six encrypted images, showing that three color channels of each encrypted image can successfully pass chi-square test.

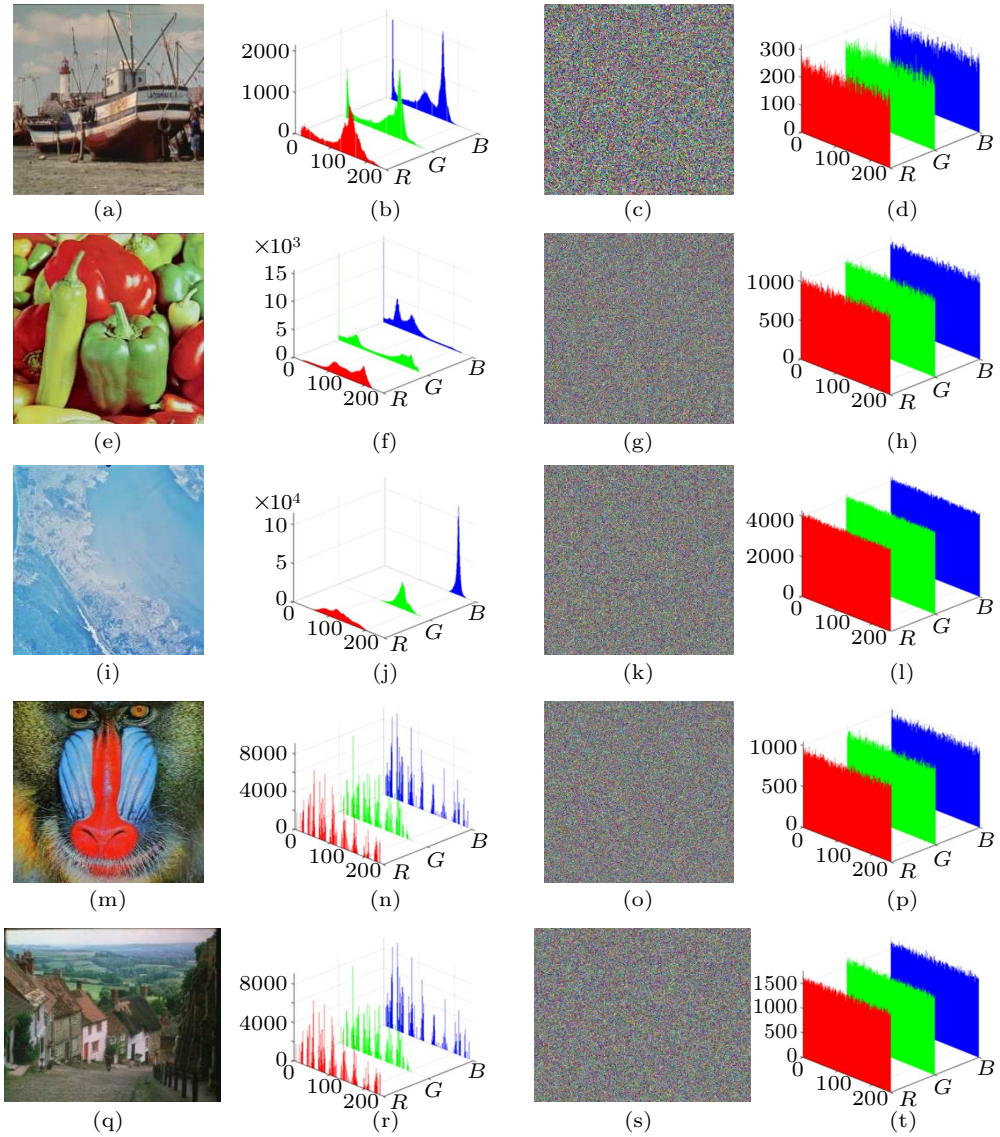


Fig. 16. Histograms before and after encryption: (a) Boat, (b) histogram of panel (a), (c) cipher image of panel (a), (d) histogram of panel (c), (e) Pepper, (f) histogram of panel (e), (g) cipher image of panel (e), (h) histogram of panel (g), (i) Sanfrancisco, (j) histogram of panel (i), (k) cipher image of panel (i), (l) histogram of panel (k), (m) Baboon, (n) histogram of panel (m), (o) cipher image of panel (m), (p) histogram of panel (o), (q) GoldHill, (r) histogram of panel (q), (s) cipher image of panel (q), (t) histogram of panel (s).

Table 6. Chi-square tests.

Cipher image	Chi-square value				Result
	R	*G*	*B*	Average	
Boat	257.7812	275.6250	238.6953	257.3672	pass
Pepper	283.4727	236.4727	248.7617	256.2357	pass
Lenna	271.7754	275.7930	269.1699	272.2461	pass
SanFrancisco	246.8955	264.6538	243.3081	251.6191	pass
Baboon	246.5429	265.5531	272.6315	261.5758	pass
GoldHill	260.2481	250.8790	281.5247	264.2173	pass

4.5. Correlation analysis

In a plain image, due to content continuity and local similarity, adjacent pixels usually have similar grayscale values and exhibit high correlation, making them vulnerable to statistical analysis attacks.[37] Therefore, a good image encryption algorithm should eliminate this strong correlation, thereby randomizing the pixel value distribution and enhancing security. Let x and y represent two adjacent pixels, the correlation

is calculated by

$$\begin{cases} r_{xy} = \frac{\mathrm{cov}(x,y)}{\sqrt{D(x)}\sqrt{D(y)}}, \\ \mathrm{cov}(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y)), \\ D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2, \\ E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i, \end{cases} \tag{28}$$

where N is the number of pixels, $E(x)$ is the expected value, $D(x)$ is the variance, and $\mathrm{Cov}(x,y)$ is the covariance. Image Pepper is chosen randomly for tests before and after encryption. 10000 pairs of adjacent pixels are randomly selected along horizontal, vertical and diagonal directions, with test results shown in Fig. 17. So, pixel distribution in plain image is near the diagonal of the coordinate axis, indicating strong correlation. In contrast, pixel distribution in cipher image is very uniform, showing no obvious correlation. Additionally, Table 7 lists the correlation coefficients for different images. Therefore, it is seen that the correlation coefficients are close to 1 before encryption, and they drop to around 0 after encryption.

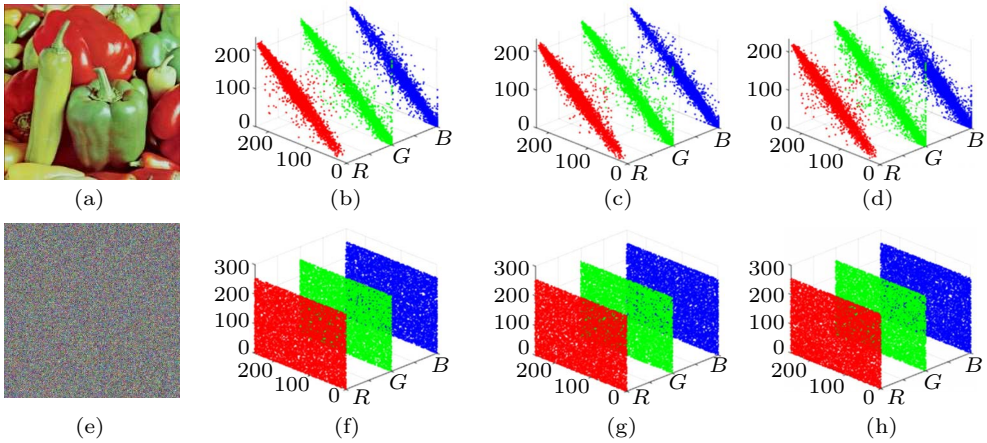


Fig. 17. Pepper correlation analysis diagram: (a) Pepper, (b)–(d) horizontal, vertical, and diagonal directions before Pepper encryption, (e) cipher image of panel (a), (f)–(h) horizontal, vertical, and diagonal directions after Pepper encryption.

Table 7. Correlation coefficient before and after encryption.

Image	Component	Plain image			Cipher image		
		Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Boat	*R*	0.9323	0.9357	0.8931	0.0284	–0.0198	0.0010
	G	0.9381	0.9453	0.8994	0.0064	0.0105	–0.0076
	B	0.9482	0.9459	0.9073	–0.0094	0.0210	0.0051
Pepper	*R*	0.9652	0.9674	0.9514	0.0015	0.0090	–0.0036
	G	0.9796	0.9674	0.9668	–0.0074	0.0084	–0.0093
	B	0.9661	0.9643	0.9464	0.0039	0.0082	–0.0017
Lenna	*R*	0.9758	0.9859	0.9636	–0.0096	0.0039	0.0057
	G	0.9652	0.9792	0.9570	–0.0019	–0.0128	0.0056
	B	0.9342	0.9568	0.9190	–0.0060	0.0102	0.0165
Sanfrancisco	*R*	0.8891	0.8919	0.8613	0.0099	0.0116	0.0051
	G	0.8209	0.8158	0.7857	–0.0040	0.0180	–0.0038
	B	0.7689	0.7726	0.7168	–0.0035	–0.0054	0.0064
Baboon	*R*	0.9244	0.8780	0.8703	0.0017	0.0014	0.0016
	G	0.8610	0.7733	0.7529	–0.0034	–0.0017	0.0119
	B	0.9062	0.8862	0.8568	0.0153	–0.0057	–0.0192
GoldHill	*R*	0.9785	0.9723	0.9584	0.0114	-0.0025	0.0187
	G	0.9806	0.9840	0.9692	0.0030	–0.0190	0.0004
	B	0.9837	0.9834	0.9736	–0.0077	–0.0081	0.0018

4.6. Differential attacks

A secure image encryption algorithm should effectively resist differential attacks,[37] meaning that any changes in the plain image should lead to significant changes in the cipher image. To assess the resistance of CIEA-4DALHS to differential attacks, two metrics are commonly used: the number

of pixels change rate (NPCR) and the unified average change intensity (UACI), which are defined by

$$\begin{cases} D(i,j) = \begin{cases} 0, \text{ if } C_1(i,j) = C_2(i,j), \\ 1, \text{ if } C_1(i,j) \neq C_2(i,j), \end{cases} \\ \text{NPCR}(C_1,C_2) = \frac{1}{M\times N}\sum\limits_{i=1}^{M}\sum\limits_{j=1}^{N} D(i,j)\times 100\%, \\ \text{UACI}(C_1,C_2) = \frac{1}{M\times N}\sum\limits_{i=1}^{M}\sum\limits_{j=1}^{N} \frac{|C_1(i,j)-C_2(i,j)|}{255}\times 100\%, \end{cases} \tag{29}$$

where C_1 represents the original cipher image. C_2 represents another cipher image after some changes. M and N are the height and width of the original image, respectively. If NPCR and UACI values are close to the theoretical values of 99.6094% and 33.4635%, it shows that the designed encryption algorithm has strong ability against differential attacks.[37] In the proposed CIEA-4DALHS, a single pixel value is randomly altered in the original image, and after 100 tests, the average NPCR and UACI values for each color channel are obtained, as shown in Table 8. Table 9 presents further the tests for image Pepper by comparisons. From these, it can be observed that the NPCR and UACI values after using CIEA-4DALHS are closer to the theoretical values.

Table 8. NPCR and UACI values using CIEA-4DALHS.

Image	NPCR (%)			UACI (%)		
	R	*G*	*B*	*R*	*G*	B
Boat	99.6307	99.6277	99.6140	33.3768	33.4925	33.4196
Pepper	99.6082	99.6166	99.6140	33.4731	33.4626	33.4582
Lenna	99.6250	99.6067	99.6075	33.4044	33.4506	33.5074
SanFrancisco	99.6100	99.6028	99.6067	33.4889	33.4452	33.4588
Baboon	99.6012	99.5883	99.5892	33.4904	33.4660	33.3979
GoldHill	99.6050	99.6226	99.6205	33.4725	33.4880	33.4951

Table 9. Comparisons of NPCR and UACI for image Pepper.

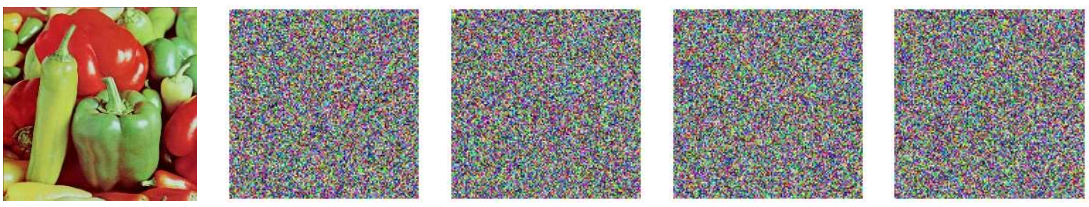
Pepper	NPCR (%)				UACI (%)			
	R	*G*	*B*	Average	*R*	*G*	*B*	Average
Ours	99.6062	99.6146	99.6120	99.6109	33.4731	33.4626	33.4582	33.4646
[34]	99.6727	99.6086	99.6506	99.6440	33.3884	33.5813	33.5137	33.4945
[38]	99.5830	99.5960	99.6430	99.6073	33.4620	33.6230	33.4040	33.4963

4.7. Key sensitivity analysis

High key sensitivity means that a small change in the key will lead to a completely different outcome.[6] Key sensitivity is typically evaluated by NPCR and UACI. To verify the key sensitivity of CIEA-4DALHS, Table 10 presents the test results, where it can be observed that with a small difference of 10^{-15} to keys, the values of NPCR and UACI are very close to the ideal cases. Figure 18 gives the test results visually, where only the correct key can decrypt the correct image. All these tests demonstrate that CIEA-4DALHS has high key sensitivity.

Table 10. Test of key sensitivity.

Image	Testkey	Difference	NPCR (%)			UACI (%)		
			R	*G*	*B*	*R*	*G*	B
Pepper	x_0	x_0+10^{-15}	99.6075	99.6269	99.6166	33.4623	33.3899	33.4726
	y_0	y_0+10^{-15}	99.6227	99.6044	99.6128	33.4231	33.4988	33.4988
	z_0	z_0+10^{-15}	99.6243	99.6231	99.5998	33.4513	33.4816	33.4264
	w_0	w_0+10^{-15}	99.6143	99.6246	99.6040	33.4957	33.4363	33.5039



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

Fig. 18. Key sensitivity decryption test: (a) correct key, (b) x_0+10^{-15}, (c) y_0+10^{-15}, (d) z_0+10^{-15}, (e) w_0+10^{-15}.

4.8. Robustness analysis

Robustness refers to the ability of an encryption algorithm when faces with various disturbances, tampering or attacks.[27] In this study, cropping attack and noise attack are performed on the encrypted images produced by CIEA-4DALHS to test whether the original image could be recovered, as shown in Fig. 19. So, it demonstrates that CIEA-4DALHS is capable of preserving image quality, indicating strong robustness.

4.9. Chosenplaintext attack

Chosen-plaintext attack (CPA) means a scenario where an attacker selects specific plaintext images and obtains their corresponding ciphertexts, aiming to deduce the internal structure of the encryption algorithm or retrieve key information by analyzing the relationship between the plaintext and ciphertext.[28] This type of attack exploits patterns or weaknesses in the algorithm when processing specific inputs. Therefore, evaluating an encryption algorithm's resistance to CPA is essential to ensuring its security. In this study, completely black and completely white images are used to test the algorithm's resistance to CPA. As shown in Fig. 20 and Table 11, the encrypted images appear as noise without any discernible information, with uniformed histograms and their information entropy close to 8. These results indicate that CIEA-4DALHS can effectively resist CPA.

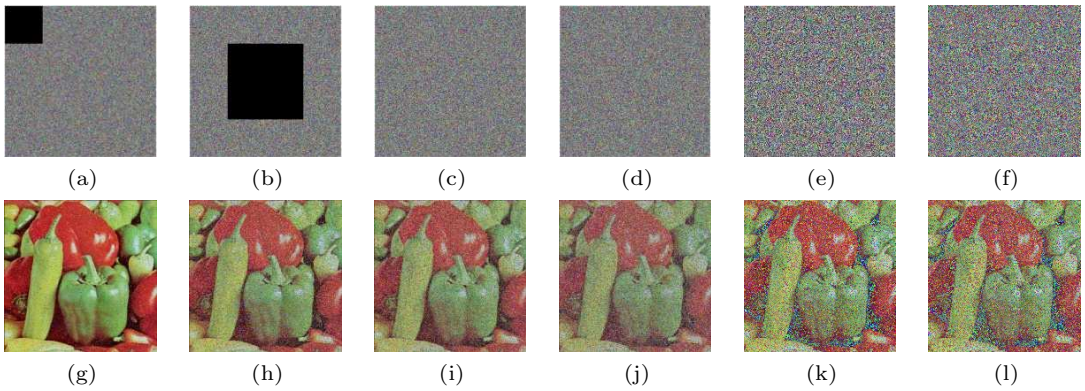


Fig. 19. Cropping attack and noise attack: (a) Cropping 128 × 128, (b) Cropping 256 × 256, (c) 5% salt and Pepper noise, (d) 7% salt and Pepper noise, (e) 0.0005% Gaussian noise, (f) 0.001% Gaussian noise, (g) decrypted image of panel (a), (h) decrypted image of panel (b), (i) decrypted image of panel (c), (j) decrypted image of panel (d), (k) decrypted image of panel (e), (l) decrypted image of panel (f).

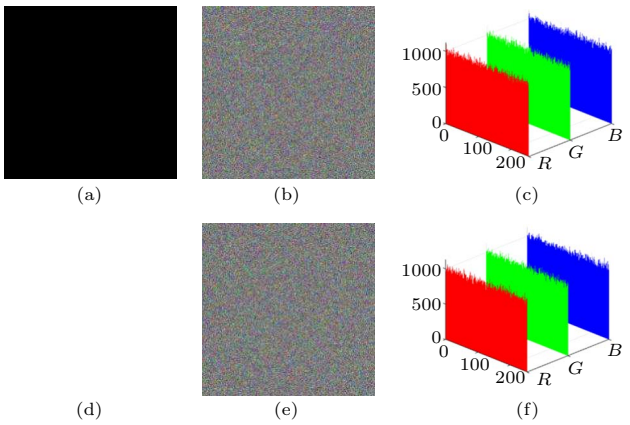


Fig. 20. All-black and all-white images tests: (a) all-black image, (b) cipher image of panel (a), (c) histogram of panel (b), (d) all-white image, (e) cipher image of panel (d), (f) histogram of panel (e).

Table 11. Information entropy of all-black and all-white images.

Image	Information entropy		
	R	*G*	*B*
All-black	7.9993	7.9993	7.9993
All-white	7.9992	7.9993	7.9992

4.10. Encryption efficiency analysis

For image encryption algorithms, the time cost is also an important performance indicator.[39] In this study, some encryption algorithms are performed on color images of sizes 256×256×3 and 512×512×3, and the average encryption times are filled in Table 12. So, the proposed CIEA-4DALHS demonstrates faster encryption speed compared with some current algorithms.

Table 12. Encryption time comparisons (in unit: s).

Image size	Ours	Ref. [39]	Ref. [40]	Ref. [41]
256×256×3	0.2051	0.2698	0.4944	0.5153
512×512×3	0.4373	1.2060	1.3408	1.7209

5. Conclusions

To sum up, a novel color image encryption algorithm, i.e., CIEA-4DALHS, was presented. The algorithm integrates bidirectional spiral cross scrambling, bit-plane substitution, and hierarchical regional segmentation diffusion. The main contributions of this work can be summarized as follows. First, 4DALHS is proposed, providing complex dynamics and high-quality pseudo-random sequences for encryption. Second, a bidirectional spiral cross scrambling method with arbitrary starting points and four traversal modes is designed to enhance scrambling performance. Third, a bit-plane substitution mechanism is introduced to achieve fine-grained encryption and strengthen resistance to statistical and chosen-plaintext at-

tacks. Finally, a hierarchical regional segmentation diffusion strategy is developed, which improves diffusion effectiveness and significantly enhances robustness against differential attacks.

Experimental evaluations confirm that CIEA-4DALHS achieves excellent security performance, including large key space, high entropy, low correlation, and strong resistance to statistical and differential attacks, while maintaining competitive efficiency compared with existing schemes. These findings demonstrate the practical potential of the proposed approach in protecting image data in various application scenarios such as medical imaging, cloud services, and secure communications.

However, this study also has some limitations. The computational efficiency, although acceptable, could be further optimized for real-time and large-scale applications. Moreover, hardware implementation and resistance to more advanced cryptanalytic models warrant further exploration. In future research, we will concentrate on enhancing the efficiency of the proposed algorithm to meet real-time application demands. In addition, we plan to investigate the integration of information hiding techniques into encrypted images, such as reversible data hiding in the encrypted domain, which can provide additional capabilities for copyright protection, covert communication, and data integrity verification.

Acknowledgements

The authors would like to thank the Editor and all Reviewers for all help on this manuscript. Project supported in part by the Guangdong Basic and Applied Basics Research Foundation (Grant No. 2026A1515011617), the Special Project in Key Area of General University in Guangdong Province of China (Grant No. 2020ZDZX3064), and the Innovation Team Project of General University in Guangdong Province of China (Grant No. 2024KCXTD042).

References

[1] Toktas F, Erkan U and Yetgin Z 2024 *Expert Syst. Appl.* **249** 123583
[2] Zhou L L and Tan F 2019 *Nonlinear Dyn.* **96** 869
[3] Ye G D, Liu S K, Xiao X C and Huang X L 2025 *Math. Comput. Simul.* **237** 316
[4] Liu M, Ye G D and Zhou J W 2025 *Sci. China Technol. Sci.* **68** 1920405
[5] Teng L, Cao P F and Liu Y 2024 *IEEE T. Circ. Syst. Vid.* **35** 1562
[6] Lai Q and Liu Y 2025 *Sci. China Technol. Sci.* **68** 1320401
[7] Lai Q and Hua H Q 2025 *Expert Syst. Appl.* **264** 125854
[8] Gao S, Zhang Z Y, Lu H H C, Ding S Q, Mou J and Erkan U 2025 *IEEE Internet Things J.* **12** 18115
[9] Yu F, He S Q, Yao W, Cai S and Xu Q 2025 *IEEE T. Comput. Aid. D* **44** 4564
[10] Feng W, Zhang K Y, Zhang J, Zhao X Y, Chen Y, Cai B, Zhu Z G, Wen H P and Ye C H 2025 *Fractal Fract.* **9** 426
[11] Hao W T, Zhang T S, Chen X Y and Zhou X Y 2023 *Signal Process.* **205** 108890
[12] Liu H, Teng L, Zhang Y, Si R and Liu P 2024 *Expert Syst. Appl.* **235** 121090
[13] Jiang D, Tsafack N, Boulila W, Ahmad J and Barba-Franco J J 2024 *Expert Syst. Appl.* **236** 121378
[14] Liu Y and Teng L 2024 *Eur. Phys. J. Plus* **139** 554
[15] Wang X Y and Guan N 2023 *J. Amb. Intel. Hum. Comp.* **14** 13399
[16] Cai H, Sun J Y, Gao Z B and Zhang H 2022 *J. Real-Time Image Pr.* **19** 775
[17] Chen L, Li C and Li C 2022 *J. Vis. Commun. Image R* **83** 103424
[18] Feng W, Qin ZT, Zhang J and Ahmad M 2021 *IEEE Access* **9** 145459
[19] Gao S, Iu H H C, Erkan U, Simsek C, Toktas A, Cao Y, Wu R, Mou J, Li Q and Wang C 2025 *IEEE T. Circ. Syst. Vid.* **35** 7706
[20] Wang S M, Peng Q Q and Du B X 2022 *Opt. Laser Technol.* **148** 107753
[21] Zhang X Q and Yang X C 2022 *Phys. Scripta* **97** 095210
[22] Mansouri A and Wang X Y 2021 *Visual Comput.* **37** 189
[23] Zheng J M and Lv T Y 2022 *IET Image Process.* **16** 3863
[24] Zhu H G, Dai L W, Liu Y T and Wu L J 2021 *Math. Comput. Simulat.* **185** 754
[25] Zhou W J, Wang X Y, Wang M X and Li D Y 2022 *Opt. Laser. Eng.* **149** 106782
[26] Wei D Y, Jiang M J and Deng Y 2023 *Expert Syst. Appl.* **213** 119074
[27] Li T Y, Fan W, Wu J and Zhang D Z 2024 *Nonlinear Dyn.* **112** 10555
[28] Zhou N R, Hu L L, Huang Z W, Wang M M and Luo G S 2024 *Expert Syst. Appl.* **238** 122052
[29] Hu B Q, Liu J H and Liu Z 2025 *Chin. Phys. B* **34** 070505
[30] Sang Y P, Sang J and Alam M S 2022 *Pattern Recogn. Lett.* **153** 59
[31] Yan S H, Li L, Gu B X, Sun X, Ren Y and Zhang Y Y 2023 *Appl. Intell.* **53** 31181
[32] Gao X H 2021 *Phys. Scripta* **96** 065203
[33] Safdar M U, Shah T and Ali A 2025 *Arab. J. Sci. Eng.* **50** 785
[34] Chai X L, Zhi X C, Gan Z H, Zhang Y S, Chen Y R and Fu J Y 2021 *Signal Process.* **183** 108041
[35] Es-sabry M, El Akkad N, Khrissi L, Satori K, El-Shafai W, Altameem T and Rathore R S 2024 *Egypt. Inform. J.* **25** 100449
[36] Su Y N and Wang X Y 2022 *Multimedia Syst.* **28** 553
[37] Gao X Y, Sun B, Cao Y H, Banerjee S and Mou J 2023 *Chin. Phys. B* **32** 030501
[38] Duan C F, Zhou J, Gong L H, Wu J Y and Zhou N R 2022 *Opt. Lasers Eng.* **150** 106881
[39] Wen J, Xu X M, Sun K H, Jiang Z H and Wang X 2023 *Nonlinear Dyn.* **111** 6813
[40] Kaur G, Agarwal R and Patidar V 2022 *Visual Comput.* **38** 1027
[41] Wang P, Liu X J, Xu J and Lu C H 2024 *Phys. Scripta* **99** 105260