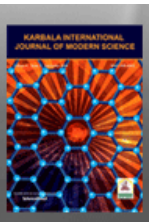

Karbala International Journal of Modern Science

Volume 11 | Issue 3 Article 7

A Novel Chaotic DNA-Based Image Cryptosystem Leveraging Euclidean Division, Dynamic Josephus Traversal, and Reservoir Computing

Ahmed Kareem Shibeeb
Department of Computer Science, Faculty of Computer Science and Maths, University of Kufa, Najaf, Iraq., ahmed.kareem@mtu.edu.iq

Salah Albermany
Department of Computer Science, Faculty of Computer Science and Maths, University of Kufa, Najaf, Iraq.

Sadiq A. Mehdi
Department of Computer Science, College of Education, Mustansiriyah University, Baghdad, Iraq

Recommended Citation

Shibeeb, Ahmed Kareem; Albermany, Salah; and Mehdi, Sadiq A. (2025) "A Novel Chaotic DNA-Based Image Cryptosystem Leveraging Euclidean Division, Dynamic Josephus Traversal, and Reservoir Computing," *Karbala International Journal of Modern Science*: Vol. 11 : Iss. 3 , Article 7.
Available at: https://doi.org/10.33640/2405-609X.3415

A Novel Chaotic DNA-Based Image Cryptosystem Leveraging Euclidean Division, Dynamic Josephus Traversal, and Reservoir Computing

Abstract

This study presents a new chaotic DNA-based image cryptosystem that combines Euclidean division, dynamic Josephus traversal (DJT), and reservoir computing to address the weaknesses of current methods. Old chaotic DNA cryptosystems usually have problems such as using the same keys for different messages, simple DNA processes, and being vulnerable to attacks where the attacker can choose the input or try many options. The cryptosystem in this study uses a 7D hyperchaotic system that starts with keys created from SHA-512 hashes to produce changing keystreams based on the plaintext, making it very strong against such attacks. The proposed cryptosystem uses a base-4 numeral system instead of traditional DNA encoding, which significantly reduces the amount of computing power required to convert pixels to DNA. Dynamic Josephus traversal introduces an adaptive permutation by dynamically altering the traversal parameters to enhance unpredictability. Simultaneously, reservoir computing selects from 14 new DNA operations, such as upshift, downshift, and composite NOT transformations, to improve data mixing. Thorough security tests show outstanding results: a keyspace of 2^{860}, almost perfect randomness (7.9999), an encryption time of 1.2925 s for a 512×512 image, NPCR (99.6331%), UACI (33.5379%), and high encryption quality. The cryptosystem shows strong resistance to noise and cropping attacks, with PSNR values of 19.2801 dB at 0.1 noise density and 12.3207 dB with 50% data missing, which is better than the best current methods. The NBCR metric (49.98%) confirms that the key sensitivity is almost perfect, meaning that there is almost no link between ciphertexts made with slightly different keys. Comparative studies indicate that it is more resistant to differential, statistical, and adversarial attacks and can encrypt data faster than other DNA-based cryptosystems.

Keywords

Chaotic system, DNA cryptography, Dynamic Josephus Traversal, Image encryption, Reservoir Computing

RESEARCH PAPER

A Novel Chaotic DNA-based Image Cryptosystem Leveraging Euclidean Division, Dynamic Josephus Traversal, and Reservoir Computing

Ahmed Kareem Shibeeb [a,b,*], Salah Albermany [a], Sadiq A. Mehdi [c]

[a] Department of Computer Science, Faculty of Computer Science and Maths, University of Kufa, Najaf, Iraq
[b] Department of Computer Systems, Technical Institute - Suwaira, Middle Technical University, Baghdad, Iraq
[c] Department of Computer Science, College of Education, Mustansiriyah University, Baghdad, Iraq

Abstract

This study presents a new chaotic DNA-based image cryptosystem that combines Euclidean division, dynamic Josephus traversal (DJT), and reservoir computing to address the weaknesses of current methods. Old chaotic DNA cryptosystems usually have problems such as using the same keys for different messages, simple DNA processes, and being vulnerable to attacks where the attacker can choose the input or try many options. The cryptosystem in this study uses a 7D hyperchaotic system that starts with keys created from SHA-512 hashes to produce changing keystreams based on the plaintext, making it very strong against such attacks. The proposed cryptosystem uses a base-4 numeral system instead of traditional DNA encoding, which significantly reduces the amount of computing power required to convert pixels to DNA. Dynamic Josephus traversal introduces an adaptive permutation by dynamically altering the traversal parameters to enhance unpredictability. Simultaneously, reservoir computing selects from 14 new DNA operations, such as upshift, downshift, and composite NOT transformations, to improve data mixing. Thorough security tests show outstanding results: a keyspace of 2^{860}, almost perfect randomness (7.9999), an encryption time of 1.2925 s for a 512 × 512 image, NPCR (99.6331 %), UACI (33.5379 %), and high encryption quality. The cryptosystem shows strong resistance to noise and cropping attacks, with PSNR values of 19.2801 dB at 0.1 noise density and 12.3207 dB with 50 % data missing, which is better than the best current methods. The NBCR metric (49.98 %) confirms that the key sensitivity is almost perfect, meaning that there is almost no link between ciphertexts made with slightly different keys. Comparative studies indicate that it is more resistant to differential, statistical, and adversarial attacks and can encrypt data faster than other DNA-based cryptosystems.

Keywords: Chaotic system, DNA cryptography, Dynamic Josephus traversal, Image encryption, Reservoir computing

1. Introduction

Image encryption is vital for protecting sensitive data from unauthorized access and alterations. Chaotic systems have become increasingly significant in image encryption because of their random behavior, sensitivity to initial conditions, and aperiodicity [1]. DNA-based image cryptosystems, which use simple mathematics and logic with low computing requirements, have recently attracted interest for improving the security of sensitive data. Recognizing the strengths of both approaches, researchers are increasingly using chaotic systems and DNA cryptography to design more secure cryptosystems [2]. This synergy exploits the unpredictability of chaotic systems and the efficient encoding mechanisms of DNA. For instance, Kalpana and Murali [3] introduced a color image cryptosystem using chaotic systems and DNA operations. The system separates images into R, G, and B channels, encodes them using rules set by chaos, and spreads them using a DNA matrix. However, the algorithm can be vulnerable to certain types of attacks, which illustrates the importance of better key

Received 15 March 2025; revised 17 June 2025; accepted 19 June 2025.
Available online 15 July 2025

* Corresponding author at: Department of Computer Systems, Technical Institute - Suwaira, Middle Technical University, Baghdad, Iraq
E-mail address: ahmed.kareem@mtu.edu.iq (A.K. Shibeeb).

https://doi.org/10.33640/2405-609X.3415

security and stronger DNA processes [4]. Hu et al. [5] developed an image encryption method using a spatiotemporal chaotic system and DNA cryptography. The key stream is derived from the input keys and plain images, ensuring resilience against plaintext attacks. The encoded image undergoes a DNA-level permutation. Wang and Liu [6] suggested a new way to encrypt images using a Piecewise Linear Chaotic Map (PWLCM) and a Logistic Map to create parameters and DNA encoding as an extra security measure. The cryptosystem generates a chaotic key image that undergoes DNA encoding based on logistic map rules. The encoded key performs standard DNA operations on the plaintext image, which makes the final cipher image fully encrypted. However, the cryptosystem was cracked because of its limited key space and simplistic DNA operations [7]. Quantum chaotic systems and DNA cryptography image cryptosystems, which include mixing pixels and replacing DNA, were combined for image encryption in Ref. [8]. However, detailed analysis showed major weaknesses, such as the absence of a similar key and the system's vulnerability to attacks because it does not have a robust structure for mixing and spreading information in DNA-level encryption [9]. Additionally, Mir and Lone [10] employed basic DNA methods to mix the image and utilized the Baker map to rearrange the pixels; however, there was no connection between the original image and the encryption key. In Ref. [11], an image cryptosystem that combined a 6D chaotic system with DNA cryptography was introduced. Initially, a plain image undergoes permutation and diffusion using chaotic keystreams at the decimal level. Subsequently, DNA encoding is applied, followed by additional permutation and diffusion at the DNA level using various flipped-chaotic sequences. Finally, the encoded DNA sequences are decoded and combined to produce cipher images. In addition, the cipher key is uncorrelated with the plaintext image. Feng et al. [12] introduced a novel one-dimensional chaotic map and an accompanying image encryption algorithm. The chaotic sequence rearranges the image pixels and generates sequences for DNA coding. The encryption process uses DNA to encode and decode data and mixes it using chaotic map settings derived from a hash algorithm based on the original image, making it difficult for attackers to target specific parts of the image. A dual-layer security framework for cloud-based medical imaging systems was proposed in Ref. [13]. The framework separates the important area (RoI) to check its accuracy and uses regular DNA along with a Lorenz chaotic system to encrypt the medical images. In Ref. [14], the researchers used various chaotic maps to create keys and dynamically control DNA operations, scrambling, and diffusion. Initially, the image undergoes DNA encoding and scrambling using random sequences, followed by iterative diffusion to change the pixel values.

Most previous chaotic DNA-based image cryptosystems have the following vulnerabilities:

- DNA-based cryptosystems suffer from time complexity owing to multiple encoding and decoding steps, including three steps: converting pixel values to binary, grouping bits, and encoding them into DNA bases.
- DNA cryptosystems that utilize low-dimensional chaotic maps exhibit inadequate security, with small keyspaces that render them susceptible to brute-force attacks.
- DNA-based cryptosystems require a long processing time because they must undergo several steps to encode and decode information, such as changing pixel values into binary, grouping bits, encoding them into DNA bases, and then reversing these steps when decoding.
- Studies indicating that fixed keystreams for different images weaken security show that these algorithms are not excellent at protecting against chosen plaintext attacks.
- Most DNA encryption algorithms lack dynamicity because they use a single DNA encoding rule and a single operation.
- Current DNA operations are restricted to limited types, including XOR, addition, subtraction, multiplication, and complementarity.

To address these challenges, this study proposes a novel image encryption algorithm that uses base-4 DNA cryptography, dynamic Josephus traversal, and reservoir computing. The encryption process uses the SHA-512 hash function to create plaintext-linked, secret keys. It also sets up a 7D hyperchaotic system and a base-4 number system to facilitate DNA coding and decoding. Reservoir computing chooses 14 new DNA operations to diffuse pixel values, whereas dynamic Josephus traversal adds unpredictability. The integration of the DJT, Euclidean division, and RC offers a novel solution to the vulnerabilities in existing systems. This system differs from conventional DNA cryptosystems. The key contributions of this paper are as follows:

1. Introducing 14 distinct DNA operations enhances randomness and resistance to cryptanalysis.
2. Replacing the slow DNA encoding process with base-4 numbers simplifies the encoding and decoding processes using the Euclidean division.

3. SHA512 of the plain image is used in our cryptosystem to determine the initial values of the 7D hyperchaotic system. This mechanism produces a wide key space, which improves the resilience of the method to known and chosen plaintext cryptanalysis and successfully reduces brute-force attacks.
4. Implementing dynamic Josephus traversal provides a dynamic permutation mechanism.
5. This work is the first to combine RC with DNA-based cryptosystems for image encryption. Unlike traditional chaotic map-based methods that control a single, fixed DNA operation, RC dynamically selects from 14 operations, thereby offering enhanced efficiency and faster processing.

The remainder of this paper is structured as follows: Section 2 elucidates all definitions and concepts pertinent to the proposed cryptosystem. Section 3 describes the fundamental components of the proposed cryptosystem. Section 4 presents the experimental results, performance evaluations, and comparative studies to substantiate the efficacy of the proposed cryptosystem. Finally, Section 5 concludes the study and delineates the prospects for subsequent research endeavors.

2. Basic definitions and related concepts

2.1. Euclidean division

Euclidean division is a key concept in number theory with broad applications in various disciplines. It is described as follows:

Quotient-Remainder Theorem: Given any two integers c and d, where $d \neq 0$, there exist two unique integers q and r such that $c = qd + r$ with $0 \leq r < d$. Here, q is the quotient obtained by dividing c by d, and r is the remainder of the division [15]. This theorem illustrates that every integer can be uniquely expressed as an ordered pair (q, r), demonstrating the decomposition capability of the Euclidean division.

2.2. Deoxyribonucleic acid (DNA)

Deoxyribonucleic acid (DNA) is a biological molecule. It carries genetic codes that survive and continue for generations. Adenine (A), guanine (G), cytosine (C), and thymine (T) are the bases of DNA. Computing fields, such as cryptography, extensively apply this concept because it simplifies the conversion of binary numbers to nucleotide sequences. In a binary system, we can convert all four bases into a pair of bits. There are eight rules to represent these bit pairs as bases, as illustrated in Ref. [16]. Furthermore, we can execute five DNA operations using these four bases. Table 1 shows the operational results for all possible combinations of DNA bases.

DNA-based cryptography has multiple significant advantages that render it a formidable element in the development of current cryptosystems. A major benefit is its ability to store more data, as each nucleotide base can represent two binary bits, making it easier to encode large amounts of digital information. Moreover, DNA computing enhances processing speed and high-resolution image encryption by enabling parallel processing. Its eight different encoding rules make it more adaptable and less predictable, and the biologically inspired methods, as detailed in Table 1, improve the diffusion process. These operations, particularly when integrated with chaotic maps, induce significant nonlinearity in the system. Additionally, the encoding process changes the statistical features of the original image, leading to an even distribution in histograms and lower correlation in the ciphertext, which provides a strong defense against statistical attacks.

2.3. Base-4 numeral system

A numerical system that utilizes a base of four is commonly called a quaternary numeral system. This system employs four distinct symbols, namely 0, 1, 2, and 3, for numerical representation. Moreover, it functions as a positional system that assigns a precise weight to each position [17]. The positions from right to left represent 4^0, 4^1, and so on. In contrast, the conversion from a base-4 number system to decimal includes multiplying each digit by the power of 4 corresponding to its position and summing the results.

2.4. 7D hyperchaotic system

Yu et al. improved a six-dimensional chaotic system by adding a nonlinear controller x_6. They

Table 1. The five basic DNA operations are (⊕) XOR, (+) addition, (−) subtraction, (×) multiplication, and (~) NOT.

⊕	A	C	G	T	+	A	C	G	T	−	A	C	G	T	×	A	C	G	T	~
A	A	C	G	T	**A**	C	A	T	G	**A**	C	G	T	A	**A**	T	G	C	A	T
C	C	A	T	G	**C**	A	C	T	G	**C**	A	C	G	T	**C**	G	T	A	C	G
G	G	T	A	C	**G**	T	G	C	A	**G**	T	A	C	G	**G**	C	A	T	G	C
T	T	G	C	A	**T**	G	T	A	C	**T**	G	T	A	C	**T**	A	C	G	T	A

transformed the system into a seven-dimensional hyperchaotic system, as detailed in Ref. [18]. The mathematical model of their system is given by the following set of equations:

$$\begin{cases} \dot{x}_1 = \mu_1(x_2 - x_1) + x_4 - x_5 - x_7 \\ \dot{x}_2 = \mu_2 x_1 - x_2 - x_1 x_3 - x_6 \\ \dot{x}_3 = \mu_3 x_3 + x_1 x_2 \\ \dot{x}_4 = \mu_4 x_4 + x_2 x_3 \\ \dot{x}_5 = \mu_5 x_7 + x_2 x_3 \\ \dot{x}_6 = \mu_6 x_1 + x_2 x_3 \\ \dot{x}_7 = \mu_7 x_5 \end{cases} \quad (1)$$

Here, μ_1, μ_2, μ_3, μ_4, μ_5, μ_6, and μ_7 are system parameters that influence the system dynamics. The system demonstrates hyperchaotic behavior with the initial condition X(0) = (1,1,1,1,1,1,1) and parameter values (μ_1, μ_2, μ_3, μ_4, μ_5, μ_6, μ_7) = (10,28,2.7,-1,8,1,5). The Lyapunov exponents (LEs) of the system are as follows: $LE_1 = 0.5885$, $LE_2 = 0.1782$, $LE_3 = 0$, $LE_4 = -0.1395$, $LE_5 = -0.4046$, $LE_6 = -0.8016$, $LE_7 = -14.0881$.

2.5. Josephus traversal

Josephus traversal is a classic mathematical problem based on the historical account of Josephus Flavius. In this problem, N individuals arranged in a circle eliminate every kth person in succession until only one remains [19]. The process used in cryptography to determine a survivor's position is illustrated using the example of eight individuals in Fig. 1. The problem is mathematically defined by parameters such as the total number of elements (n), the starting position (s), and the step size (t). The output sequence of the Josephus traversal (O) is calculated using Equation (2).

$$O = F(n, s, t) \quad (2)$$

2.6. Reservoir computing

Reservoir computing (RC) is a type of recurrent neural network architecture that eliminates the need for backpropagation over time, making computing much faster, reducing training costs, and optimizing the objective function by utilizing a linear ridge regression method. An input weight

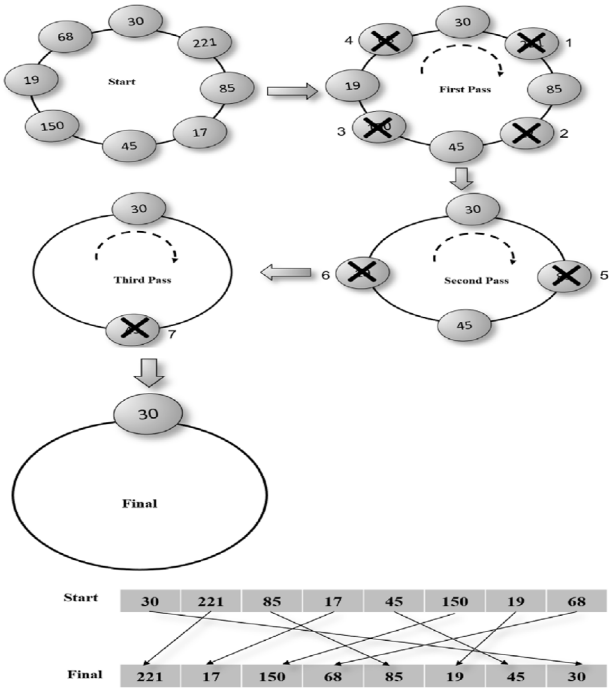

Fig. 1. An example of Josephus traversal with eight individuals.

matrix, W_{in}, maps the input data to the reservoir state x(t). The right matrix, W_{in}, maps the input data to reservoir state x(t). The reservoir layer contains a network with fixed internal connections between nodes [20]. Fig. 2 shows the RC architecture. We describe the RC state as follows:

$$x(t+1) = \tanh[W \times x(t) + W_{in} \times u(t)] \quad (3)$$

Where u(t) is the input at time t, W_{in} is the input-to-reservoir coupling matrix, W is the internal reservoir connectivity matrix, and tanh is the element-wise activation function applied to ensure nonlinearity.

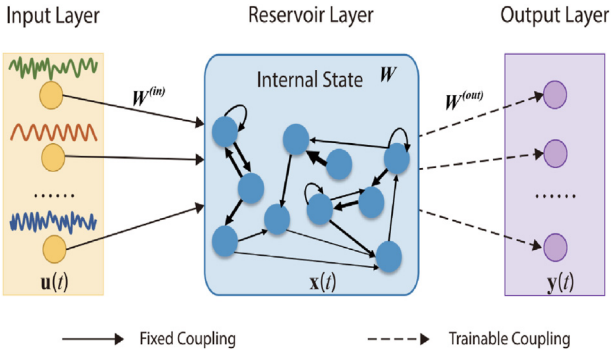

Fig. 2. Illustration of reservoir computing architecture, where the input weights (W_{in}), reservoir connections (W), and output weights (W_{out}) are represented. Only Wout is trained, whereas W_{in} and W remain fixed.

3. An image cryptosystem

This framework emphasizes the encryption of grayscale and color images characterized by dimensions M × N. It includes creating keys, shrinking the image size, changing it to base-4 format, a shuffling method based on dynamic Josephus traversal (DJT), and a spreading technique using reservoir computing DNA (RCDNA).

3.1. Key generation

This process uses a plain image (Img) to create a unique value using the SHA-512 hash function (M). This method ensures that different plain images produce unique keys, thereby enhancing security and rendering the system resistant to specific plaintext and ciphertext attacks. The 7D hyperchaotic system was selected for its ability to produce high-dimensional chaotic sequences, thereby increasing the key space and randomness of the cryptosystem. Its complicated and unpredictable behavior makes it very difficult for attackers to break the encryption using methods such as brute-force or statistical analysis. The initial values of the 7D-hyper chaotic system are generated by converting M into a 128-bit hexadecimal number, which is then divided into subkeys M_i using the following equation:

$$M_i = hex2dec\left(\frac{M(i-1)\times 2+1 : i\times 2}{2^8}\right) \quad For\, i\in[1,58] \quad (4)$$

The following equation computes Y_i based on the derived values of M_i:

$$Y_i = \mathrm{mod}\left(\sum_{j=0}^{3} M_{4(i-1)+j+1}, 1\right) \quad for\, i\in[1,19] \quad (5)$$

The 7D hyperchaotic system generates its initial values using the M_i Sequence. However, the remaining 12 hexadecimal numbers were converted to decimal numbers and XORed with each other to obtain one decimal number used as an idle iteration (Id) to avoid transient effects in a chaotic system, as follows:

$$D_i = hex2dec\left(M_{(i-1)}\times 2+1 : i\times 2\right) \quad For\, i\in[1,16] \quad (6)$$

$$Id = D_1 + D_2 + D_3 + D_4 + D_5 + D_6 \quad (7)$$

Thereafter, the researchers iterated the 7D hyperchaotic system for 3 × 4 × ceil (W × H/7) times to generate an R array of size 1 × 12 WH. The generated chaotic array R will be rearranged in ascending order to obtain index subscripts that form a permutation matrix *Rp*. To obtain the encoding and diffusion keys, Euclidean division is applied on the *Rp* matrix (where c = *Rp* matrix and d = 8) to produce two matrices: quotient matrix Rq_1 and reminder matrix Rr_1 which is used for the encoding rule selection. Then, the diffusion matrix Rr_2 is generated by applying modular four to the matrix Rq_1. Algorithm 1 explains how to create keys for an image encryption system using the size of the image, its SHA-512 hash, chaotic system steps, and modular math to produce diffusion and encoding keys for safe encryption.

Algorithm 1: Pseudocode of Key Generation

Input: Img; // Plain image
Output: Rp, Rq1, Rr1, Rr2; // Permutation matrix, encoding keys and diffusion key

```
W, H ← GetDimensions(Img); // Extract image width (W) and height (H)
M ← SHA512(Img); // Compute SHA-512 hash of the plain image
for i = 1 to 58 do
    M_i ← hex2dec((M((i-1) × 2 + 1: i × 2)) / 2^8); // e.g. M_1 = hex2dec(a9)/2^8 = 169/2^8
end for
S ← 0; // Initialize sum S to 0
for i = 1 to 19 do
    for j = 0 to 3 do
        S ← S + M(4(i-1) + j + 1); // Add subkey M(4(i-1) + j + 1) to S
    end for
    Y_i ← mod(S, 1); // e.g. Y_1 = mod((M_1+ M_2+ M_3+ M_4),1)
end for
Generating idle iterations (Id).
for i = 1 to 6 do
    D_i ← hex2dec(H((i-1) × 2 + 1: i × 2)); // e.g. D_1 = hex2dec(a9) = 169
end for
Id ← D_1D_2D_3D_4D_5D_6
N ← 3 × 4 × ceil(W × H / 7); // Compute number of iterations for chaotic system
R ← IterateHyperchaoticSystem(N); // Generate chaotic array R of size 1 × 12 WH
Rp ← sort(R); % Rearrange chaotic array R in ascending order to form matrix Rp
for each element in Rp do
    Rq1, Rr1 ← EuclideanDivision(Rp, 8)+1; // Compute array of encoding rule
    Rr2 ← Rq1 mod 4; // Generate diffusion key Rr2 using modular 4
end for
```

3.2. Reducing dimensions and base-4 representation

The proposed system analyzes color images by splitting the data into distinct channels that represent specific colors. The primary channels are red, green, and blue, which form an RGB color space.

The isolation of these channels reduces the efficiency of the cryptosystem. If multiple channels are present, each is converted into a 1D array of size W × H, converting the input image for independent processing. They were then mixed in a single 1D array (Img) of size 3 × W × H. This method allows for shuffling and spreading operations to be performed in a 1D array instead of using three 2D matrices, which significantly reduces the amount of computing power required. However, DNA-based cryptosystems suffer from the conversion required to first convert pixels from decimal to binary. Second, these bits are divided into pairs and finally encoded into their corresponding DNA bases (A, C, G, and T), and vice versa, during the decoding process. This approach requires more time for both encryption and decryption processes owing to the multiple encoding and decoding steps involved. To reduce the time taken, the researchers proposed replacing the DNA bases with base-4 numbers using Euclidean division. We employed the Euclidean division process to directly convert decimal pixel values into base-4. For instance, to convert pixel c = 57 from decimal to base-4:

- Divide 57 by 4: Quotient = 14, Remainder = 1
- Divide 14 by 4: Quotient = 3, Remainder = 2
- Divide 3 by 4: Quotient = 0, Remainder = 3

Reading the remainders from bottom to top yields $(0321)_4$.

This change simplifies the process by requiring only the conversion of pixels in the Img array from the decimal system to four digits in the base-4 number system during encoding, and reversing this conversion during decoding. As a result, DNA computing significantly reduces the time required for the aforementioned conversions. Then, the researchers use the Rr_2 array, which contains values from 1 to 8, to choose the encoding rule for each digit. The size of the output image in this phase (Img2) was 4 × 3 × W × H.

3.3. Permutation using dynamic Josephus traversal (DJT)

The main problems in classic Josephus traversals are their dependence on one direction (clockwise), a fixed step size (t), and fixed starting location(s), which is the first location. Additionally, when represented using an array, it relies on the principle of deleting the target element, which is time consuming. As illustrated in Fig. 1, removing an element from an array requires shifting all subsequent elements to the left to fill in the gap. Arrays also have a fixed size, meaning their size cannot be dynamically increased or decreased without creating a new array. This limitation makes it inefficient to remove elements from an array. Therefore, Dynamic Josephus Traversal (DJT) is proposed in this study, which makes the step size, starting location, and direction dynamic depending on the outputs of the chaotic system. Furthermore, we replace deletion operations with replacement operations to shorten the encryption process time. This approach enhances efficiency and introduces an element of unpredictability, thereby increasing the overall security and performance of the encryption. Fig. 3 and Algorithm 2 illustrate the DJT-based permutation.

Algorithm 2: Pseudocode of image permutation based on DJT

Input: Img1, Rp; // Image after encoding and permutation matrix
Output: Img2; // Permuted image

```
S ← 0, old ← 0, new ← 0; // Initialize variable S, old position, and new position to 0
for i = 1 to 4 × 3 × W × H do
  T ← Img2(i); // Store the current element in T
  S ← S + Rp(i); // Add the corresponding element of Random matrix to S
    if 4 × W × H - old < S then
      new ← S - (4 × 3 × W × H - old); // Calculate the new position
      new ← mod(new, 4 × 3 × W × H); // Adjust new using modulo operation
      if new = 0 then
        new ← 4 × 3 × W × H;
      end if
    end if
    else
      new ← S + old; // Set new position to the sum of S and old position
    end if
    Img2(i) ← Img2(new); // Swap the elements in image
    Img2(new) ← T; // Place T in the new position
    old ← new; // Update old to the new position
end for
```

3.4. Diffusion using reservoir computing DNA (RCDNA)

This study proposes a set of 14 DNA operations, as shown in Table 2. The first operation did not cause any changes in DNA bases. The second operation is NOT. The third and fourth sections introduce new operations, upshift and downshift, which complement each other. Table 3 illustrates the use of one operation for encryption and the other for decryption. These operations differ from previous operations that used the same operation for encryption and decryption in the proposed scheme. The fifth,

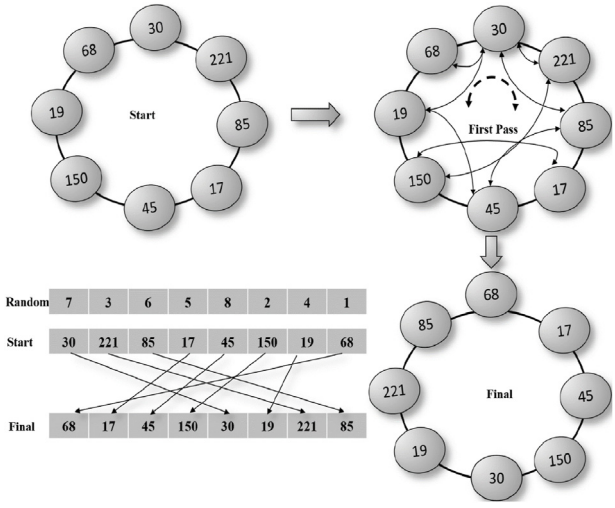

Fig. 3. An example of dynamic Josephus traversal (DJT) with eight individuals.

sixth, seventh, and eighth operations involve DNA XOR, addition, subtraction, and multiplication, respectively. However, the ninth to fourteenth operations execute a NOT operation on DNA bases first and then execute DNA upshift, downshift, XOR, addition, subtraction, and multiplication, respectively. Selecting the type of operation between the permutated image and diffusion key is the most challenging aspect of this algorithm. To address this, reservoir computing (RC) was employed to determine the type of operation. Algorithm 3 outlines the process of applying reservoir computing to the diffusion process. The reservoir computing (RC) framework was configured with a reservoir size of n reservoir = 500 neurons, sliding window size of window = 10, input weight scaling of k = 0.5, and ridge regression regularization parameter of $\lambda reg = 1 \times 10^{-8}$. These parameters were selected to balance the system complexity and computational efficiency, ensuring the effective transformation of chaotic states into dynamic DNA operation selection. The reservoir states were then flattened, and a specific number of values was extracted for further processing. We transformed these values into probabilities using the softmax function, sorted them in descending order, and mapped them to a smaller range using a modulo 14 for compatibility. The final result is a list of numbers that show how the system changes, which can be used to choose the type of DNA operation to perform between the rearranged image (Img3) and the diffusion key (Rr_2). The integration of reservoir computing with DNA-based approaches enhances the dynamic nature of DNA operations.

Next, we decoded the DNA by reversing the conversion from the base-4 numeral system to the decimal system. The resulting one-dimensional matrix is then divided into three W × H matrices, where each matrix represents a color layer (R_{new}, G_{new}, and B_{new}), thereby reconstructing the cipher color image. If the original image is grayscale, the matrix is converted directly into a two-dimensional matrix to produce a cipher image, thereby eliminating the need for matrix division.

Algorithm 3: Pseudocode of diffusion process based on RCDNA

Input: Img2, Rr2, WindowSize, $N_{Reservoir}$; // Permutated image, diffusion key and reservoir parameters
Output: Img3; // Diffused cipher image

States ← Itreate (7D hyperchaotic system); // Get states from 7D hyperchaotic system
X ← []; // Initialize the input matrix
for i = 1 to size(states, 1) − WindowSize **do**
  WindowData ← SlidingWindow(states);
  X ← Flatten(WindowData); // Prepare input matrix
end for
Y ← states; // Define the target output matrix
W_{in}, W ← Itreate (7D hyperchaotic system); // Initialize input and reservoir weights
R_{State} ← Zeros($N_{Reservoir}$, 1); // Initialize the reservoir state
for each Element in X **do**
  R_{State} ← Tanh(W_{in} × Element + W × R_{State}); // Update reservoir state
  StatesReservoir ← R_{State}; // Initialize the reservoir state
end for
NumValues ← 4 × 3 × W × H; // Compute the number of readout values
ReadoutValues ← Reshape (StatesReservoir); // Convert StatesReservoir into 1D array
ExpValues ← Exp(ReadoutValues - Max(ReadoutValues)); // Compute exponentials
SoftmaxProbs ← ExpValues / Sum(ExpValues); // Apply softmax.
Index ← Sort(SoftmaxProbs); // Get indices of SoftmaxProbs sorted in descending order
for i = 1 to NumValues **do**
  Operation [i] ← Mod(Index, 14) + 1; // Generate operations array
  Img3 ← Diffusion (Operation [i], Img2, Rr2); // Choose the operation from Table 2 to apply it between Img2 and Rr2.
end for
Img4 ← Base4ToDecimal(Img3) % Decode DNA
if Channels(Img) > 1 **then**
  R_{new}, G_{new}, B_{new} ← Split)Img4(; // Divide Img4 into three matrices of size W × H
  CipherImage ← Concatenate(R_{new}, G_{new}, B_{new}); // Reconstruct color image of size 3 × W × H
  else
  CipherImage ← Reshape (Img4); // Reconstruct grayscale image of size W × H
end if

Table 2. Proposed DNA operations.

ID	Operation Name	Symbol	Description
1	No Change	≡	No changes were made.
2	NOT	~	Converts an element into its complement.
3	Up-Shift	↑	An upshift operation is applied.
4	Down-Shift	↓	A downshift operation is applied.
5	XOR	⊕	Applies XOR operation.
6	Addition	+	Adds the elements.
7	Subtraction	−	Subtracts the elements.
8	Multiplication	×	Multiplies the elements.
9	NOT + Up-Shift	~↑	First, it applies the NOT operation, and then upshifts.
10	NOT + Down-Shift	~↓	First, the NOT operation is applied, followed by the downshift operation.
11	NOT + XOR	⊙	First applies NOT operation, then XOR operation.
12	NOT + Addition	~+	First applies NOT operation, then addition operation.
13	NOT + Subtraction	~−	First applies NOT operation, then subtraction operation.
14	NOT + Multiplication	~×	First applies NOT operation, then multiplication operation.

Table 3. Upshift (↑) and downshift (↓) operations.

↑	A	C	G	T	↓	A	C	G	T
A	A	C	G	T	**A**	A	T	G	C
C	C	G	T	A	**C**	C	A	T	G
G	G	T	A	C	**G**	G	C	A	T
T	T	A	C	G	**T**	T	G	C	A

3.5. Encryption process

Fig. 4 illustrates the encryption process of the proposed cryptosystem. It comprises five principal stages within a single round, as detailed below:

1. The 7D hyperchaotic system was initialized using user-specified control parameters (μ_1, μ_2, …, μ_7). Additionally, we computed the SHA-512 hash of the plaintext image to determine the initial conditions and the number of idle iterations. This two-part method allows users to customize settings while also making the system strongly dependent on the original data, which greatly improves the system's ability to resist attacks that target known or chosen data, as explained in the key generation process in Section 3.1.

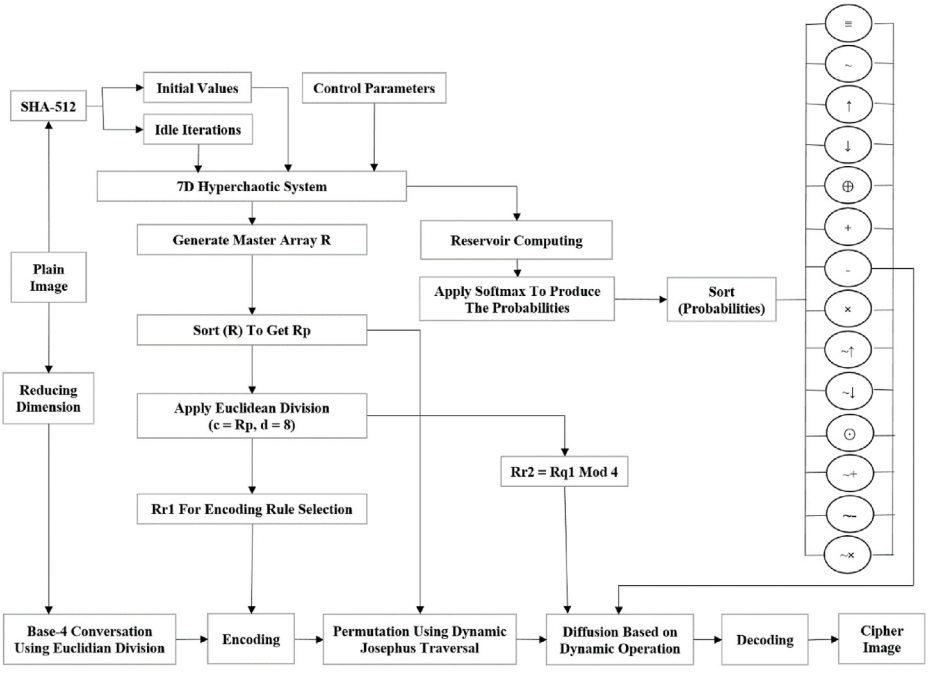

Fig. 4. Block diagram of encryption process.

2. The input image was analyzed for the channel quantity, and if it contained multiple channels (e.g., RGB), the channels were merged into a single 1D array. The pixel values are changed from decimal to a base-4 numeral system using a method called Euclidean division, as explained in Section 3.2, making DNA encoding easier and less complicated to compute than other methods.
3. The base-4 converted array is rearranged using the DJT method, which changes the step size, starting point, and direction according to chaotic sequences.
4. In this step, reservoir computing dynamically selects 14 distinct DNA operations to diffuse the permuted image. The details of the DNA operations and their integration with reservoir computing are provided in Section 3.4.
5. We decoded the diffused image into decimal format. The resulting 1D matrix is then divided into separate channels (if the input image is multichannel) and reshaped into its original dimensions to produce the final cipher image.

3.6. Decryption process

The decryption process reverses these encryption steps. The SHA-512 value and control parameters were input into the 7D hyperchaotic system to regenerate the chaotic sequences. The cipher image is changed into a DNA format using a base-4 number system, and then it goes through steps to reverse the mixing and rearranging of the pixel values and positions. Finally, the DNA sequence was decoded back to its decimal form, reshaped, and reconstructed as a plain image.

4. Simulation and security evaluation

This section presents analytical examinations to validate the protective mechanisms of the proposed image cryptosystem. Numerical simulations were conducted using a Windows 10 operating system, Intel Core i5-7300U CPU, 8 GB RAM, and MATLAB R2023a software. Six gray and color images from the USC-SIPI image database [21] were tested (Airplane, Boat, All white, All black, Tree, and Peppers), as shown in Fig. 5, using hyperchaotic system parameters such as $\mu 1 = 10$, $\mu 2 = 28$, $\mu 3 = 2.7$, $\mu 4 = -1$, $\mu 5 = 8$, $\mu 6 = 1$, and $\mu 7 = 5$.

4.1. The histogram metric

We used the histogram metric to confirm the security and resilience of the image cryptosystems by evaluating the uniformity of the ciphered image's histogram. The histogram of an image serves as a crucial metric that depicts the distribution of pixel values within the image. An encrypted image with a nearly uniformly distributed histogram thwarts attackers' attempts to extract meaningful information [22]. Fig. 5 shows the 3D histograms of several images before and after encryption. The histograms of the ciphered images exhibit a more uniform distribution than that of the plain image.

4.2. Key size

In cryptography, the key size is a crucial factor that influences the strength of encryption algorithms. The key size of an optimal cryptographic system must be sufficiently large to render brute-force attacks impractical, typically exceeding 2^{100} [23]. The key size of our encryption algorithm was 512 bits of hash value and seven control parameters. The key size of the hash value is 2^{512}. The control parameters μ_1, μ_2, μ_3, μ_4, μ_5, μ_6, and μ_7 were computed with a precision of 10^{-15}. Consequently, the total key space is represented by Equation (8):

$$KeySize = 2^{512} \times \prod_{k=1}^{7} 10^{15} \tag{8}$$

The calculated key size was $2^{512} \times 10^{105}$, which was significantly larger than 2^{860}. Table 4 presents a comparison of the key sizes among the cryptosystems. The analysis indicates that the key size discussed in this paper is significantly larger than that of other cryptosystems, thereby enhancing the resistance to brute-force attacks.

4.3. Key sensitivity

An effective image cryptosystem should be sensitive to minimal alterations in the cipher key. This implies that utilizing the correct cipher key during decryption yields the plain image, whereas modifying even a single bit of the key renders the image indecipherable [25]. The number of bits change rate (NBCR) metric evaluates key sensitivity by sequentially altering individual bits of the cipher key [26]. The entire key was modified, yielding a series of bit variations. We used these modified keys to encrypt the image. The NBCR assesses the differences between the cipher images based on the original and modified keys as follows:

$$NBCR = \frac{HamDis\,(E_1, E_2)}{L} \tag{9}$$

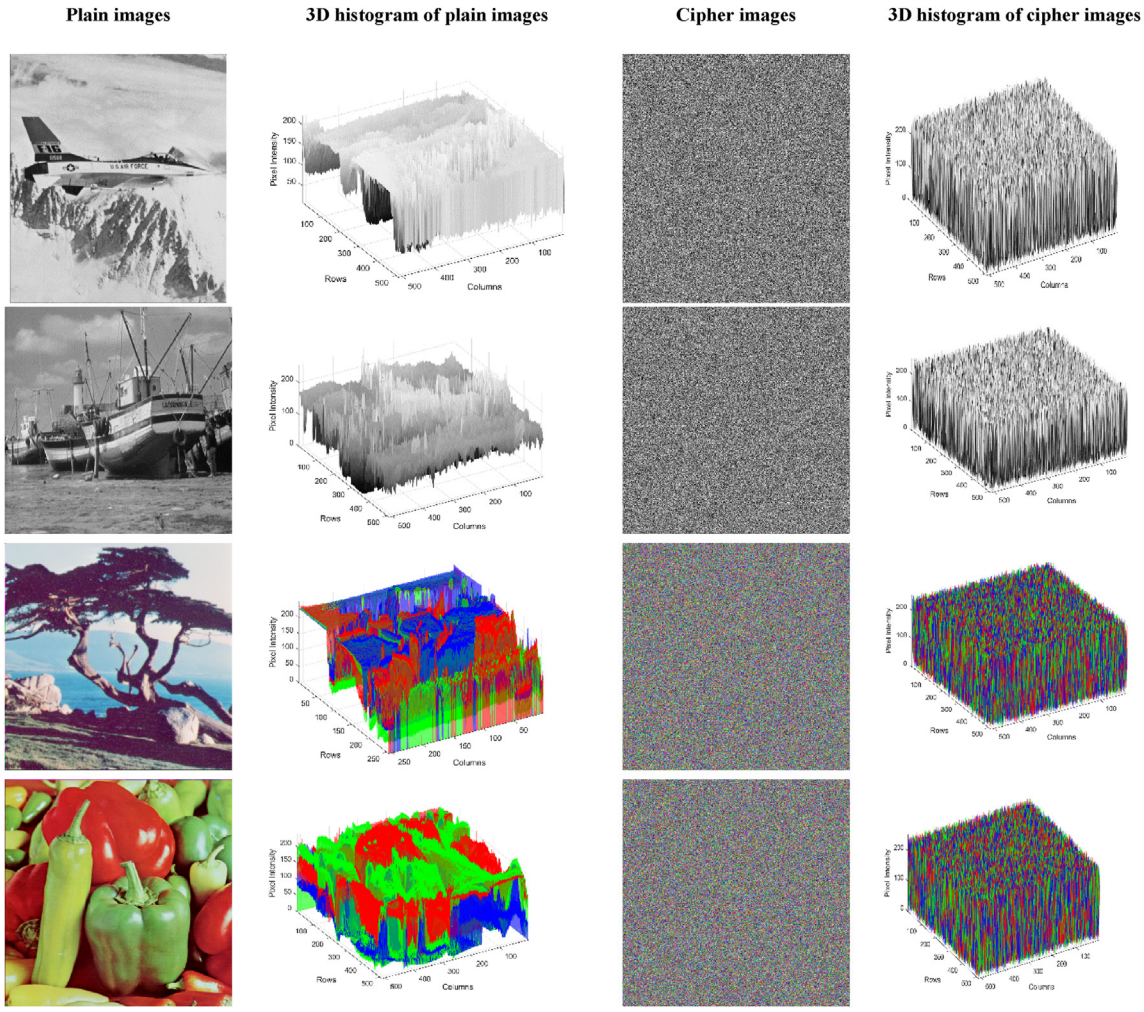

Fig. 5. 3D histogram of plain images and their corresponding cipher images.

Table 4. Key sizes of the proposed and state-of-the-art systems.

Our system	[3]	[5]	[24]	[10]	[11]	[12]	[13]	[14]
2^{860}	2^{230}	2^{249}	2^{186}	2^{408}	2^{300}	2^{256}	2^{150}	2^{392}

Where HamDis(E_1, E_2) is the Hamming distance between two encrypted images E_1 and E_2. L is the key length. For a sensitive cryptosystem, an NBCR value of approximately 50 % indicates a significant difference between E_1 and E_2. Fig. 6 illustrates the NBCR plot for the airplane image based on the altered keys. The proposed cryptosystem exhibited an NBCR close to the ideal value.

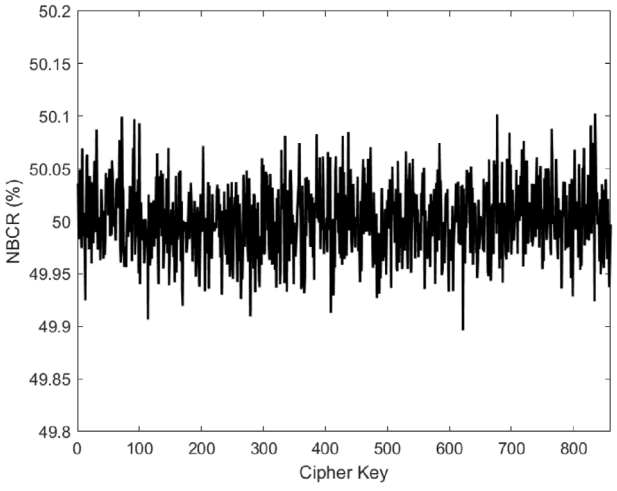

Fig. 6. Results of NBCR metric.

4.4. Cipher image quality

The quality of the reconstructed image was evaluated through quantitative analysis utilizing metrics such as the Mean Square Error (MSE), Peak Signal-to-Noise Ratio (PSNR), root mean square error (RMSE), and Structural Similarity Index (SSIM). These metrics were computed by contrasting the original images with their decrypted counterparts

[27]. The Mean Square Error (MSE) is a dispersion test that evaluates the difference between an original and cipher image. It was calculated by squaring the difference in pixels between the original and cipher images and then averaging them. A larger MSE value indicates that the cryptosystem is advantageous. This variable is defined as follows:

$$MSE = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(O(i,j)_r - D(i,j)_p \right)^2 \quad (10)$$

Where O (i, j) and D (i, j) are the pixel values of the original and deciphered images, respectively. The peak signal-to-noise ratio (PSNR), which is a significant metric for measuring the quality of a deciphered image, is calculated using the following equation:

$$PSNR = 10 \times \log_{10} \left(\frac{255 \times 255}{MSE} \right) \quad (11)$$

A low PSNR value (8–12 dB) is ideal post-encryption, as it reflects significant distortion compared to a plain image.

The root mean square error (RMSE) is defined in Equation (12). It contrasts the original and deciphered images by delineating their relative differences and similarities. These proportions serve to gauge the efficacy of the cryptosystem in restoring image fidelity post-decryption. A lower outcome indicates a superior cryptosystem performance.

$$RMSE = \sqrt{MSE} \quad (12)$$

However, the SSIM serves as a metric quantifies the similarity between the original and deciphered images, encompassing brightness, contrast, and structural elements. The calculation involved the following equation:

$$SSIM = \frac{(2\mu_O\mu_D + F_1)(2\sigma_{OD} + F_2)}{(\mu_O{}^2 + \mu_D{}^2)(\sigma_O{}^2 + \sigma_D{}^2 + F_2)} \quad (13)$$

Here, $F_1 = (0.01 \times 255)^2$, $F_2 = (0.01 \times 255)^2$, and σ_O, σ_D, μ_O, μ_D, σ_{OD} represent the variances, mean and covariance of the original and deciphered images. The SSIM values ranged from 0 to 1. A score of 1 indicated that the two images were identical, whereas values closer to 1 indicated greater similarity. Conversely, as the score approached zero, the images were increasingly dissimilar. Table 5 presents the values of the above metrics, indicating that the deciphered image nearly achieved optimum quality.

Table 5. Results of the encryption quality metrics.

Image Name	SSIM	PSNR	MSE	RMSE	Entropy
Airplane	0.0104	8.6157	8943.6265	94.5707	7.9992
Boat	0.0101	9.3028	7634.8123	87.3774	7.9996
All white	0.0106	8.6345	8905.0159	94.3664	7.9999
All black	0.0099	9.2208	7780.3711	88.2064	7.9999
Tree	0.0103	8.6135	8948.0094	94.5939	7.9997
Peppers	0.0098	8.6098	8955.6773	94.6344	7.9996
Average	0.0102	8.8329	8527.9186	92.2915	7.9997
[3]	NA	11.1621	4287.1814	65.4766	7.9974
[5]	NA	14.1248	2511.6	50.1159	7.9975
[10]	NA	9.2834	7641	87.4128	7.9974

4.5. Entropy test

Entropy measures the degree of randomness or unpredictability of a given information source. A high information entropy value suggests an increased level of uncertainty within the source, making it more difficult for the attacker to anticipate or interpret information accurately. The entropy metric, denoted as E(m), of an information source m was calculated using the following equation [28]:

$$E(m) = -\sum \Pr(m) \times \log_2 \Pr(m) \quad (14)$$

The probability of symbol m is represented by Pr(m), and the optimal entropy value of the encrypted image is 8. Table 5 also shows the entropy results for the cipher images. All cipher images successfully passed this metric, indicating that the proposed method outperformed the existing methods.

4.6. Correlation between adjacent pixels

To prevent statistical attacks, a secure image encryption algorithm must minimize the correlation between adjacent pixels. The proposed method calculates the correlation coefficients using the following mathematical formula [29]:

$$\begin{cases} Corr = \dfrac{\mathrm{Cov}(I_1, I_2)}{\sqrt{F(I_1)F(I_2)}} \\ \mathrm{Cov}(I_1, I_2) = E([I_1 - E(I_1)][I_2 - E(I_2)]) \\ E(I) = \dfrac{1}{M} \sum_{x=1}^{M} I_x \\ F(I) = \dfrac{1}{M} \sum_{x=1}^{M} [I_x - E(I)]^2 \end{cases} \quad (15)$$

In this instance, I_1 and I_2 denote close pixel pairs, where E (I_1) and E (I_2) depict the predicted pixel intensity values, F (I_1) and F (I_2) indicate the

variances, and M represents the sample count of pixel pairs. Table 6 presents the correlation coefficients for the encrypted images, which are approximately zero. Fig. 7 demonstrates a significant decrease in the correlation observed in scatter plots. These findings confirm that the proposed cryptosystem substantially mitigates pixel correlation, thereby bolstering image security.

4.7. Differential attack analysis

Differential cryptanalysis is a crucial method for evaluating the degree of similarity between distinct images and is frequently utilized as an essential parameter in the metric analyses of many image cryptosystems. The Unified Average Changing Intensity (UACI) and Number of Pixels Change Rate (NPCR) are two methods for measuring the resistance of image cryptosystems to differential cryptanalysis. Elevated NPCR values and diminished UACI values suggest enhanced resistance to such attacks. The UACI and NPCR measures were computed as follows [12]:

Table 6. Comparison of image correlation coefficients across all three directions.

Image	Plain image			Cipher image		
	V	H	D	V	H	D
Airplane	0.9648	0.9686	0.9397	0.0035	0.0028	−0.0003
Boat	0.9404	0.9677	0.9241	−0.0001	0.0005	−0.0001
All white	1	1	1	0.0036	−0.0008	−0.0006
All black	1	1	1	−0.0003	0.0002	0.0018
Tree	0.9282	0.9704	0.9131	−0.0011	−0.0007	0.0009
Peppers	0.9668	0.9631	0.9381	−0.0021	0.0003	−0.0001
Average	0.9667	0.9783	0.9525	0.0006	0.0004	0.0003
[3]	0.9801	0.9505	0.8414	−0.0094	−0.0177	−0.0042
[5]	0.9376	0.9688	0.9177	0.0002	−0.0077	0.0055
[10]	0.9285	0.9427	0.9056	0.0037	−0.0018	0.0023
[11]	0.9607	0.9535	0.9478	0.0028	−0.0057	0.0037
[12]	0.9666	0.9433	0.9186	−0.0052	−0.0036	−0.0096

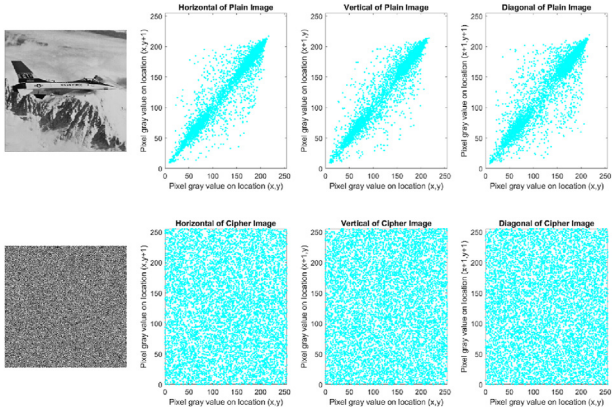

Fig. 7. The result of the correlation coefficient metric in all directions.

$$UACI = \frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} \frac{CI_1(x,y) - CI_2(x,y)}{255} \times 100\% \quad (16)$$

$$NPCR = \frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} D(x,y) \times 100\% \quad (17)$$

Here, $CI_1(x,y)$ stands for the ciphered image of the plain image, $CI_2(x,y)$ stands for the ciphered image corresponding to the plain image after a slight change, and $D(x, y) = 0$ if $CI_1(x, y) = CI_2(x, y)$. Otherwise, $D(x, y) = 1$. The NPCR metric focuses on the aggregate quantity of pixels, which is subject to variation when subjected to differential cryptanalysis.

Wu et al. posited that a 512 × 512 image satisfies all theoretical NPCR critical thresholds (N_a^*) at any predetermined significance level when the NPCR outcome is equal to or exceeds 99.5893 %. Additionally, it successfully reaches the UACI threshold (Ua) if the obtained result falls within the critical interval ($U_a^- = 33.373$ %, $U_a^+ = 33.5541$ %) at a significance level of $\alpha = 0.05$ [30]. Table 7 demonstrates the proposed cryptosystem's resilience against attacks, particularly its notable resistance to differential cryptanalysis.

4.8. Efficiency analysis

A high level of security can be attained through complex cryptosystems; however, excessive complexity may render a system impractical owing to slow performance. Therefore, an effective encryption algorithm must balance robust security and computational efficiency [31]. We performed an efficiency analysis from the beginning of the encryption process to its completion. During key

Table 7. Results of the NPCR and UACI metrics.

Image	NPCR $N_a^* \geq 99.5893$ %	UACI $U_a^- = 33.373$ % $U_a^+ = 33.5541$ %
Airplane	99.6353	33.5431
Boat	99.6078	33.4978
All white	99.6122	33.5503
All black	99.6827	33.5488
Tree	99.6101	33.5494
Peppers	99.6505	33.5381
Average	99.6331	33.5379
[3]	99.6231	33.6698 **(Failed)**
[5]	99.5911	33.3851
[8]	99.59	28.54 **(Failed)**
[10]	99.6117	33.4322
[12]	99.6127	33.4768
[14]	99.9	32.7 **(Failed)**

generation, the SHA-512 hash calculation and the setup of the 7D hyperchaotic system take the same amount of time every time, meaning that it is constant time or O(1), because the number of steps does not change. The Dynamic Josephus Traversal (DJT) scrambles all M × N pixels with a time complexity of O(M × N). Similarly, the sliding window technique used in reservoir computing for dynamic DNA operation selection and the subsequent diffusion stage, where DNA operations are applied to every pixel, both run at O(M × N). Consequently, the overall time complexity of our algorithm is O(M × N), which guarantees scalability for images of different dimensions.

Table 8 shows how the efficiency of encoding differs between the old DNA encoding method (which includes converting to binary, grouping into two bits, and mapping to bases) and our new base-4 encoding method. Our approach reduces the encoding time from 0.7226 s to 0.0042 s and decreases memory usage from 2.1 MB to 1.4 MB, while also simplifying the operation and decoding complexity.

Table 9 presents a performance comparison of various permutation techniques. The proposed DJT method exhibits a speed of 0.0913 s and delivers a superior permutation effect, in contrast to the standard Josephus, circular shift, sorting, Fisher-Yates, and 2D cat map methods, which either demonstrate poor permutation effects or low efficiency. Table 10 provides a comparative analysis of the encryption speed. Our cryptosystem accomplishes an encryption time of 1.2925 s, surpassing similar cryptosystems explained in prior works.

4.9. Impact of RC on security and efficiency

We compared the computational efficiency and security metrics (entropy, NPCR, UACI, and encryption time) with and without Reservoir Computing (RC) in order to demonstrate the importance of RC in the proposed cryptosystem. The results in Table 11 clearly show that using RC makes the cryptosystem more unpredictable, increases the entropy to nearly the best possible levels, and ensures better protection against statistical and differential attacks. Additionally, the addition of RC maintained effective encryption times without appreciably increasing the computational overhead. In contrast to static DNA operation selection, this analysis highlights the role of RC in dynamically adapting DNA operations, thereby providing a strong defense mechanism.

Table 9. Performance comparison of permutation techniques.

Permutation Method	Speed (s)	Performance Observations
Standard Josephus	6.2136	Poor permutation effect Low efficiency
Circular shift	0.0732	Low permutation effect High efficiency
Sorting	2.2105	Better permutation effect Low efficiency
Fisher-Yates	12.4508	Better permutation effect Low efficiency
2D cat map	18.4162	Poor permutation effect Low efficiency
Proposed DJT	0.0913	Better permutation effect High time efficiency

Table 10. Results of the speed analysis.

Cryptosystem	Proposed	[5]	[8]	[12]	[14]
Time (sec.)	1.2925	2.3861	2.213	1.372	3.7119

Table 11. Comparative analysis with and without reservoir computing (RC) using airplane images.

Metric	With RC	Without RC
Entropy	7.9992	7.9941
NPCR	99.6353	99.5918
UACI	33.5431	33.3547 **(Failed)**
Encryption Time (sec)	1.2925	2.8036

4.10. NIST 800–22 test suits

To assess the pseudo-random number generator for use in cryptographic contexts, we analyzed the encrypted image generated by our proposed cryptosystem. Initially, we transformed the created sequence into a binary format, enabling us to apply the National Institute of Standards and Technology (NIST) test to assess its statistical properties [32]. When conducting tests, the p-value always falls

Table 8. Comparison of the efficiency between traditional DNA mapping and the proposed base-4 representation.

Feature	Traditional DNA Encoding	Proposed Base-4 Encoding
Encoding Steps	Binary → 2-bit grouping → Base map	Decimal → Base-4 conversion
Speed	0.7226 Seconds	0.0042 Seconds
Memory Usage	2.1 MB	1.4 MB
Operation Complexity	Multi-step with condition checking	Single-step Euclidean division
Decoding Complexity	Reverse mapping with binary parsing	Simple base-4 to decimal conversion

between 0 and 1. A p-value exceeding the threshold of m = 0.01 indicates that the sequence is random and passes the test. Table 12 presents the 15 statistical assessments that constitute the NIST statistical test on the airplane image. Remarkably, our method successfully met all the statistical criteria for randomness.

4.11. Chosen-plaintext cryptanalysis

Chosen-plaintext cryptanalysis is a strong strategy in cryptographic security that poses a significant threat to cryptosystems. In such attacks, adversaries exploit their access to the encryption algorithm by deliberately selecting plaintext inputs and observing the corresponding ciphertext outputs, thereby inferring sensitive information about the encryption key. The capacity of an encryption algorithm to resist chosen plaintext cryptanalysis typically indicates its robustness against other cryptanalysis methodologies, including ciphertext-only, known-plaintext, and chosen-ciphertext cryptanalysis [33]. Given that the encryption key is contingent on the plaintext image (i.e., it depends on the hash value of the plaintext image) and considering the sensitivity of the proposed cryptosystem to variations in the plaintext image, as described in Subsection 3.1, it becomes infeasible to establish any discernible correlation between the resulting encrypted image and the ciphertext derived from the designated plaintext image. Consequently, our cryptosystem exhibits resilience to chosen-plaintext cryptanalysis and other attack scenarios.

Table 12. NIST SP 800-22 metrics for the cipher airplane image.

Test item	P-value	Passed
Frequency	0.843176	✓
Block frequency	0.377521	✓
Cumulative sums (Forward mode)	0.732936	✓
Cumulative sums (Reverse mode)	0.352317	✓
Runs	0.963374	✓
Longest run of one	0.638439	✓
Serial (Test 1)	0.783926	✓
Serial (Test 2)	0.667384	✓
Non-overlapping templates	0.811485	✓
FFT	0.939541	✓
Rank	0.427343	✓
Approximate Entropy	0.796146	✓
Overlapping templates	0.735351	✓
Random excursions	0.844935	✓
Random excursions variant	0.537664	✓
Linear complexity	0.941153	✓
Universal	0.700826	✓

4.12. Noise and cropping attack

The cryptosystem should possess the capability to resist malevolent noise assaults directed towards the cipher image. To illustrate, we consider the incorporation of salt and pepper noise [5]. Fig. 8 shows images featuring salt-and-pepper noise with concentrations of 0.005, 0.05, and 0.1. The experimental results show that the reconstructed image can reveal details about the original image, suggesting that this cryptosystem is robust against noise attacks. However, a cropping attack involves manipulating certain pixel values within the encrypted image to make it black, thereby mimicking the effect of image cropping [34]. Fig. 9 shows the ciphered images with data occlusion ratios of 1/16, 1/4, and 1/2.

In related studies, these two important attacks were either ignored or measured only visually, as in Refs. [10,12], except for [5], where the image quality was tested both visually and numerically using only the PSNR metric. To be fair, we visually tested the image quality after these two attacks, as shown in Figs. 8 and 9, using the PSNR and SSIM metrics, and compared them with the results of [5] as shown in Table 13. After the decryption process, high PSNR

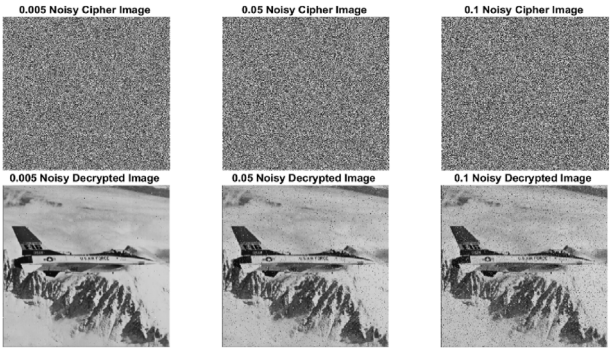

Fig. 8. Simulation results of noise attacks for airplane image.

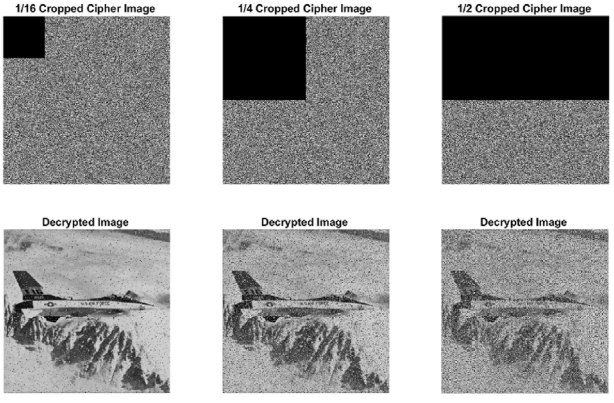

Fig. 9. Simulation results of cropping attacks for airplane image.

Table 13. The quality of the decrypted images is affected by cropping and by noise attacks.

Image	Crop	PSNR	SSIM	Noise	PSNR	SSIM
Airplane	1/16	20.5785	0.4456	0.005	31.5446	0.9217
	1/4	14.6359	0.1679	0.05	21.6298	0.5124
	1/2	11.6228	0.0814	0.1	18.5999	0.3344
Boat	1/16	21.3756	0.5139	0.005	32.5887	0.9932
	1/4	15.3298	0.1854	0.05	22.3357	0.5734
	1/2	12.3207	0.0834	0.1	19.2801	0.3854
All white	1/16	16.7942	0.1431	0.005	27.5746	0.8239
	1/4	10.7875	0.0167	0.05	17.7648	0.1947
	1/2	7.7794	0.0091	0.1	14.7935	0.0649
All black	1/16	16.8102	0.0929	0.005	27.7953	0.8208
	1/4	10.7822	0.0004	0.05	17.7038	0.1437
	1/2	7.7651	0.0001	0.1	14.775	0.0256
Tree	1/16	20.6306	0.4511	0.005	31.8233	0.9224
	1/4	14.6444	0.1671	0.05	21.6446	0.5112
	1/2	11.6377	0.0827	0.1	18.6798	0.3361
Peppers	1/16	20.6541	0.4517	0.005	31.3802	0.9187
	1/4	14.6775	0.1685	0.05	21.7057	0.5162
	1/2	11.6527	0.0832	0.1	18.6243	0.3342
[5]	1/16	20.7788	NA	0.005	30.8394	NA
	1/4	14.734	NA	0.05	21.4269	NA
	1/2	11.4238	NA	0.1	18.2828	NA

and SSIM values are essential to confirm the near-perfect reconstruction of the plain image.

5. Conclusions

This study addresses the vulnerabilities of traditional chaotic DNA-based cryptosystems. These systems typically suffer from a small key space, simple operations, and are vulnerable to noise, cropping, and chosen-plaintext attacks. We propose an innovative cryptosystem built on a seven-dimensional (7D) hyperchaotic system, dynamic Josephus traversal (DJT), and reservoir computing (RC). Our method leverages the SHA-512 hash function to generate plaintext-dependent keys and employs base-4 encoding for efficient DNA representation. This approach achieves a key space of 2^{860}, exhibits optimal entropy, meets cipher image quality metrics, and reduces computational costs. The DJT mechanism enhances security by dynamically reordering image pixels based on unpredictable chaotic-sequence outputs. Reservoir computing improves diffusion by dynamically selecting 14 new DNA operations. These enhancements yielded NPCR and UACI values of 99.6331 % and 33.5379 %, respectively. The cryptosystem maintains its structural integrity under challenging conditions, with PSNR values of 12.3207 dB when 50 % of the data are obscured and 19.2801 dB under 0.1-density salt-and-pepper noise. Performance evaluations show that our system outperforms existing cryptosystems. It achieves near-ideal key sensitivity (NBCR ≈ 50 %) and encrypts a 512 × 512 image in just 1.2925 s in a single round. A notable limitation of this study is the time required to compute the SHA-512 hash value. Future work may address this issue by adopting a more efficient hashing algorithm and exploring the proposed cryptosystem for IoT and embedded system applications.

Ethical information

This research did not involve human participants, human data, or animals. Therefore, ethical review and approval were not required for this study in accordance with institutional guidelines and national regulations.

Funding

This research was supported by the University of Kufa, Iraq (Grant Numbers: 1313).

Conflict of interest

The authors declare that they have no conflicts of interest.

Acknowledgements

The Authors sincerely acknowledge the support of the University of Kufa, Iraq, which provided resources and facilities that significantly contributed to the successful completion of this research.

References

[1] F. Toktas, U. Erkan, Z. Yetgin, Cross-channel color image encryption through 2D hyperchaotic hybrid map of optimization test functions, Expert Syst. Appl. 249 (2024) 123583, https://doi.org/10.1016/j.eswa.2024.123583.
[2] I. Qiqieh, J. Alzubi, O. Alzubi, DNA cryptography based security framework for health-cloud data, Computing 107 (2024) 35, https://doi.org/10.1007/s00607-024-01393-9.
[3] J. Kalpana, P. Murali, An improved color image encryption based on multiple DNA sequence operations with DNA synthetic image and chaos, Optik (Stuttg). 126 (2015) 5703–5709, https://doi.org/10.1016/j.ijleo.2015.09.091.
[4] H. Wen, Y. Lin, Cryptanalyzing an image cipher using multiple chaos and DNA operations, J. King Saud Univ. - Comput. Inf. Sci. 35 (2023) 101612, https://doi.org/10.1016/j.jksuci.2023.101612.
[5] T. Hu, Y. Liu, L.H. Gong, S.F. Guo, H.M. Yuan, Chaotic image cryptosystem using DNA deletion and DNA insertion, Signal Process. 134 (2017) 234–243, https://doi.org/10.1016/j.sigpro.2016.12.008.
[6] X. Wang, C. Liu, A novel and effective image encryption algorithm based on chaos and DNA encoding, Multimed. Tools Appl. 76 (2017) 6229–6245, https://doi.org/10.1007/s11042-016-3311-8.
[7] Y. Zhao, Q. Shi, Q. Ding, Cryptanalysis of an image encryption algorithm using DNA coding and chaos, Entropy 27 (2025) 40, https://doi.org/10.3390/e27010040.
[8] J. Zhang, D. Huo, Image encryption algorithm based on quantum chaotic map and DNA coding, Multimed. Tools Appl. 78 (2019) 15605–15621, https://doi.org/10.1007/s11042-018-6973-6.

[9] H. Wen, Y. Lin, Cryptanalysis of an image encryption algorithm using quantum chaotic map and DNA coding, Expert Syst. Appl. 27 (2024) 1–21, https://doi.org/10.1016/j.eswa.2023.121514.
[10] P.N. Lone, U.H. Mir, A. Gaffar, Hyperchaotic image encryption using DNA coding and discrete cosine transform, J. Franklin Inst. 360 (2023) 13318–13338, https://doi.org/10.1016/j.jfranklin.2023.10.010.
[11] Q. Li, L. Chen, An image encryption algorithm based on 6-dimensional hyper chaotic system and DNA encoding, Multimed. Tools Appl. 83 (2024) 5351–5368, https://doi.org/10.1007/s11042-023-15550-3.
[12] S. Feng, M. Zhao, Z. Liu, Y. Li, A novel image encryption algorithm based on new one-dimensional chaos and DNA coding, Multimed. Tools Appl. 83 (2024) 84275–84297, https://doi.org/10.1007/s11042-024-19090-2.
[13] N. Chidambaram, K. Thenmozhi, P. Raj, R. Amirtharajan, DNA-chaos governed cryptosystem for cloud-based medical image repository, Clust. Comput. 27 (2024) 4127–4144, https://doi.org/10.1007/s10586-024-04391-w.
[14] S.T. Allawi, Y.M. Mohialden, N.M. Hussien, Protection of the image data by using chaotic maps and DNA sequence, Int. J. Intell. Eng. Syst. 17 (2024) 451–462, https://doi.org/10.22266/IJIES2024.0831.35.
[15] K. Ireland, M. Rosen, A Classical Introduction to Modern Number Theory, 2nd ed., Springer, New York, NY, New York. 2013 https://doi.org/10.1007/978-1-4757-2103-4.
[16] X. Wang, Y. Su, An audio encryption algorithm based on DNA coding and chaotic system, IEEE Access 8 (2020) 9260–9270, https://doi.org/10.1109/ACCESS.2019.2963329.
[17] R. Rani, L.K. Singh, N. Sharma, FPGA implementation of fast adders using quaternary signed digit number system, 2009 Int. Conf. Emerg. Trends Electron. Photonic. Devices Syst. (2009) 132–135, https://doi.org/10.1109/electro.2009.5441154.
[18] W. Yu, J. Wang, J. Wang, H. Zhu, M. Li, Y. Li, D. Jiang, Design of a new seven-dimensional hyperchaotic circuit and its application in secure communication, IEEE Access 7 (2019) 125586–125608, https://doi.org/10.1109/ACCESS.2019.2935751.
[19] F. Artuğer, Strong s-box construction approach based on Josephus problem, Soft. Comput. 7 (2024) 10201–10213, https://doi.org/10.1007/s00500-024-09751-7.
[20] L. Shi, H. Wang, S. Wang, R. Du, S.-X. Qu, Predicting non-smooth chaotic dynamics by reservoir computing, Phys. Rev. E. 109 (2024) 14214, https://doi.org/10.1103/PhysRevE.109.014214.
[21] USC-SIPI Image Database, (n.d.). http://sipi.usc.edu/database/database.php (accessed March 8, 2025).
[22] M. Wang, X. Song, S. Liu, X. Zhao, N. Zhou, A novel 2D Log-Logistic–Sine chaotic map for image encryption, Nonlinear Dyn. 113 (2025) 2867–2896, https://doi.org/10.1007/s11071-024-10331-5.
[23] S.K. Tripathi, B. Gupta, S.S. Lamba, A companion matrix-based efficient image encryption method, Signal Process. 228 (2025) 109753, https://doi.org/10.1016/j.sigpro.2024.109753.
[24] X. Zhang, X. Wang, Multiple-image encryption algorithm based on DNA encoding and chaotic system, Multimed. Tools Appl. 78 (2019) 7841–7869, https://doi.org/10.1007/s11042-018-6496-1.
[25] A.K. Shibeeb, M.H. Ahmed, A.H. Mohammed, A new chaotic image cryptosystem based on plaintext-associated mechanism and integrated confusion-diffusion operation, Karbala Int. J. Mod. Sci. 7 (2021) 176–188, https://doi.org/10.33640/2405-609X.3117.
[26] L. Li, A self-reversible image encryption algorithm utilizing a novel chaotic map, Nonlinear Dyn. 113 (2025) 1–33, https://doi.org/10.1007/s11071-024-10726-4.
[27] N. Mahendiran, C. Deepa, A comprehensive analysis on image encryption and compression techniques with the assessment of performance evaluation metrics, SN Comput. Sci. 2 (2021) 1–12, https://doi.org/10.1007/s42979-020-00397-4.
[28] S. Khan, H. Peng, A secure and adaptive block-based image encryption: a novel high-speed approach, Nonlinear Dyn. 112 (2024) 16445–16473, https://doi.org/10.1007/s11071-024-09870-8.
[29] Y. Wu, L. Zhang, S. Berretti, S. Wan, Medical image encryption by content-aware DNA computing for secure healthcare, IEEE Trans. Ind. Informatics 19 (2023) 2089–2098, https://doi.org/10.1109/TII.2022.3194590.
[30] G. Hu, B. Li, Coupling chaotic system based on unit transform and its applications in image encryption, Signal Process. 178 (2021) 1–17, https://doi.org/10.1016/j.sigpro.2020.107790.
[31] B. Sakthi kumar, R. Revathi, A comprehensive review on image encryption techniques using memristor based chaotic system for multimedia application, IETE J. Res. 70 (2024) 8160–8183, https://doi.org/10.1080/03772063.2024.2373899.
[32] M.U. Safdar, T. Shah, A. Ali, Design of nonlinear component of block cipher over non-chain semi-local ring with its application to color image encryption, Arab. J. Sci. Eng. 50 (2025) 785–806, https://doi.org/10.1007/s13369-024-09010-9.
[33] M. Yildirim, A color image encryption scheme reducing the correlations between R, G, B components, Optik (Stuttg) 237 (2021) 166728, https://doi.org/10.1016/j.ijleo.2021.166728.
[34] Y. Xian, X. Wang, Fractal sorting matrix and its application on chaotic image encryption, Inf. Sci. (Ny) 547 (2021) 1154–1169, https://doi.org/10.1016/J.INS.2020.09.055.