Image encryption algorithm based on 2D hyper-chaotic system and central dogma of molecular biology

Article *in* Physica Scripta · July 2023

DOI: 10.1088/1402-4896/ace5ee

3 authors, including:

Xiaoqiang Zhang
China University of Mining and Technology
46 PUBLICATIONS **789** CITATIONS

SEE PROFILE

PAPER

Multiple-image encryption algorithm based on genetic central dogma

Physica Scripta

PAPER

Multiple-image encryption algorithm based on genetic central dogma

RECEIVED
20 January 2022

REVISED
7 April 2022

ACCEPTED FOR PUBLICATION
12 April 2022

PUBLISHED
25 April 2022

Xiaoqiang Zhang and Jingxi Tian

School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221116, People's Republic of China

E-mail: grayqiang@163.com

Keywords: multiple-image encryption, chaotic system, DNA, bit plane, genetic central dogma

Abstract

A large number of images are generated and distributed in various fields every day. To protect the image information from being stolen during the network transmission, this paper proposes a multiple-image encryption algorithm based on genetic central dogma and three-dimensional (3D) bit planes. This paper simulates the genetic central dogma and defines the 3D bit planes. Firstly, *k* original images are transformed into 8-bit binary and converted into a 3D matrix; secondly, the 3D matrix is permutated by rotating the bit plane and performing permutation among the bit planes; thirdly, we encode the scrambled 3D matrix into DNA codes. The diffusion is realized by imitating the genetic central dogma and introducing RNA mutations; finally, the encrypted images are obtained by the RNA decoding operation. Experimental results and algorithm analyses indicate that the proposed algorithm has strong security and desirable performance.

1. Introduction

In recent years, more and more attention has been paid to the security of networks and information systems. Image has become an important information carrier in people's daily life, which can transmit a large amount of information intuitively and vividly. It is widely used in communication, military, telemedicine, etc. However, due to the openness of the Internet, the information is easily intercepted or leaked during network transmission. Data hiding and image encryption are two common technologies to keep the image secure, but the former suffers some limitations without enough embedding ability [1]. In comparison, image encryption can make the plain image beyond recognition and effectively protect images. Thus, how to encrypt images efficiently and securely becomes significant.

The image cannot be encrypted by traditional data encryption algorithms, such as Data Encryption Standard (DES), Advanced Encryption Standard (AES) and International Data Encryption Algorithm (IDEA), because of its large data capacity, high inter-pixel correlation and data redundancy [2, 3]. Recently, various types of image encryption algorithms have been proposed, such as image encryption based on chaos [4, 5], image encryption based on compressed sensing [6, 7], image encryption based on genetic algorithm [8] and image encryption based on the neural network [9]. However, the above algorithm encrypts a single image, whose encryption capacity is limited. With the advent of the big-data era, the encryption efficiency attracts experts' attention in the multiple-image security field and becomes an important index of encryption algorithms.

Some inherent characteristics of chaotic systems, such as pseudo-randomness, sensitivity to initial conditions and system parameters, ergodicity and highly complex behavior, are highly consistent with the requirements of the cryptosystem. The theory of chaotic systems is widely used in the field of image encryption [10, 11]. In 1989, Matthews firstly combined the chaotic system with an encryption system [12]. Matthews proposed a generalized Logistic map and used the sequence of pseudo-random numbers generated by this map to encrypt text data. On the basis of a two-dimensional (2D) continuous chaotic map, Fridrich proposed a discrete chaotic map to scramble the positions of image pixels and implemented an image encryption algorithm with the scrambling diffusion structure [13].

Chaotic image encryption algorithms mainly include four categories: the low-dimensional (LD) chaotic system, high-dimensional (HD) chaotic system, hyperchaotic system and compound chaotic system. The LD chaotic system is simple, outstanding and easy to implement. But its key distribution is not uniform, and its generated sequence is also unstable. After all, the control parameters of the system are too few, and the key space is relatively small [14]. To solve these problems, some HD chaotic systems with complex dynamic characteristics have appeared, such as the Lorentz system [15], Chen system [16] and Bao system [17]. These HD chaotic systems can effectively suppress common attacks. However, their computational complexity increases. Therefore, employing multiple chaotic systems on a single image encryption system may be an ideal strategy [18, 19]. In this paper, two chaotic systems, i.e., the 2D Logistic-adjusted-Sine map (2D-LASM) and the three-dimensional (3D) sine map, are combined to participate in the scrambling and diffusion processes of the algorithm.

In 1994, the Turing prize winner Adleman solved a 7-node directed Hamiltonian path problem by using DNA [20], a tool of molecular biology.He firstly proved the feasibility of computing at the molecular level through experiments and the powerful parallel computing power of DNA molecules. Meanwhile, DNA computing has some significant advantages, such as huge storage space and ultra-low power consumption [21]. The biological computer is the development trend in the future [22]. With the continuous exploration in the DNA field, people have a more comprehensive grasp. The trend of combining DNA, a tool of biological molecular science, with traditional cryptography comes into being. The excellent performance of DNA computing in terms of parallel computing power, energy efficiency and storage capacity has made DNA attract scientists' much attentionin the new field of DNA encryption. Xue, *et al* encrypted color images by combining a memeristive hyperchaotic system and DNA. However, the encryption time is unsatisfactory, and the encryption time of the DNA encryption operation occupies 36% of the overall encryption time [23]. Cun *et al* further localized the image after pre-processing to reduce the encryption time of DNA operations, but the experimental results apparently showed that the results were not excellent [24]. Zhu, *et al* proposed an image encryption algorithm based on 3D DNA horizontal permutation. This algorithm combines the dynamic DNA encoding and decoding, but it is only applied to encrypt a single image, which cannot meet the requirements of multiple image encryption (MIE) [25]. To strengthen the connection between DNA encryption algorithm and biology, this paper simulates the process of gene central dogma and shows the complete process of genetic information transmission, including transcription, translation, RNA mutation and other biological processes.

The main contributions of this paper are listed as follows: (1) We define the 3D bit planes and design a scrambling method in the bit plane, which can synchronously disturb several original images; (2) The SHA-256 hash value of the plain images and external parameters jointly generate the initial values and control parameters of the chaotic systems; (3) This paper simulates the genetic central dogma, including DNA encoding, transcription, translation and amino acid synthesis, so that the image encryption algorithm is closely integrated with the biological genetic process; (4) RNA mutation is introduced in the encryption process to simulate the mutation process in nature, and the randomness of the mutation mode and position further improves the security of the encryption algorithm. Finally, the RNA operation is combined with the chaos-generated RNA matrix to achieve further diffusion; (5) We propose an MIE algorithm based on genetic central dogma and 3D bit planes. The experimental results and algorithm analyses prove that the algorithm is feasible and secure.

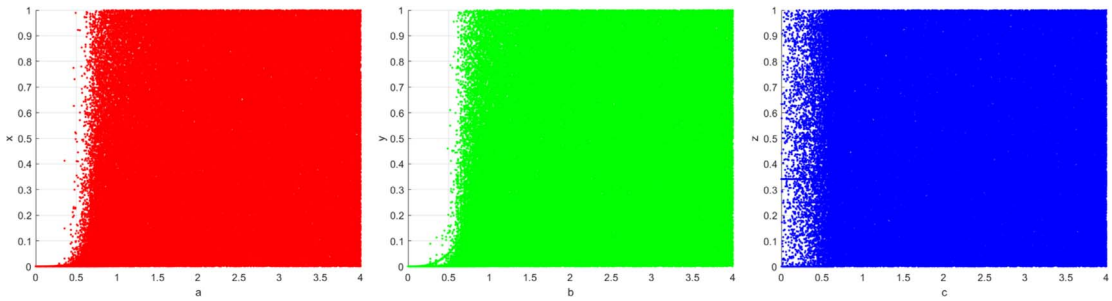
The rest of our paper is organized as follows: section 2 introduces the theoretical principles, including the chaotic systems, the 3D bit planes, the genetic central dogma, the nucleic acid encoding and operating rules, and the principle of RNA mutation. Section 3 proposes an MIE algorithm based on genetic central dogma and 3D bit planes. Experiments are performed in section 4. The algorithm analyses are made in section 5. The conclusions are drawn in section 6.

2. Theoretical principles

2.1. Chaotic systems

The 3D sine map [26] is defined by

$$\begin{cases} x_i = \dfrac{a^3 \sin \pi x_i(1-y_i)}{\sin \pi y_i(1-z_i)} \bmod 1 \\ y_i = \dfrac{b^3 \sin \pi y_i(1-z_i)}{\sin \pi z_i(1-x_i)} \bmod 1, \\ z_i = \dfrac{c^3 \sin \pi z_i(1-x_i)}{\sin \pi x_i(1-y_i)} \bmod 1 \end{cases}$$



(a) x and a (b) y and b (c) z and c

Figure 1. Bifurcation diagrams.

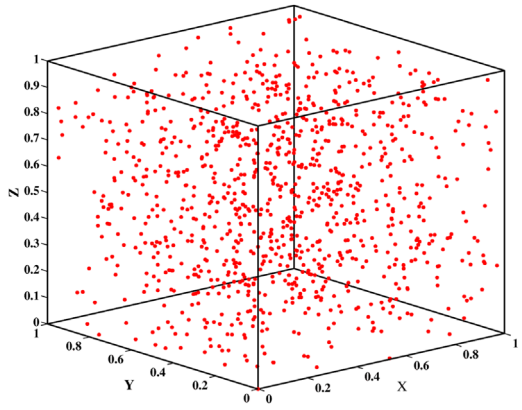


Figure 2. Attractor of the 3D sine map.

where mod $(\cdot)$ denotes the modulus after division, the initial values $x_0, y_0, z_0 \in [0, 1]$, and the control parameters $a, b, c \in [1, 4]$. The bifurcation diagram is shown in figure 1. The visual representation of the attractor for the 3D sine map is shown in figure 2. These figures show the excellent chaotic behavior of the 3D sine map.

The 2D-LASM [27] is defined by

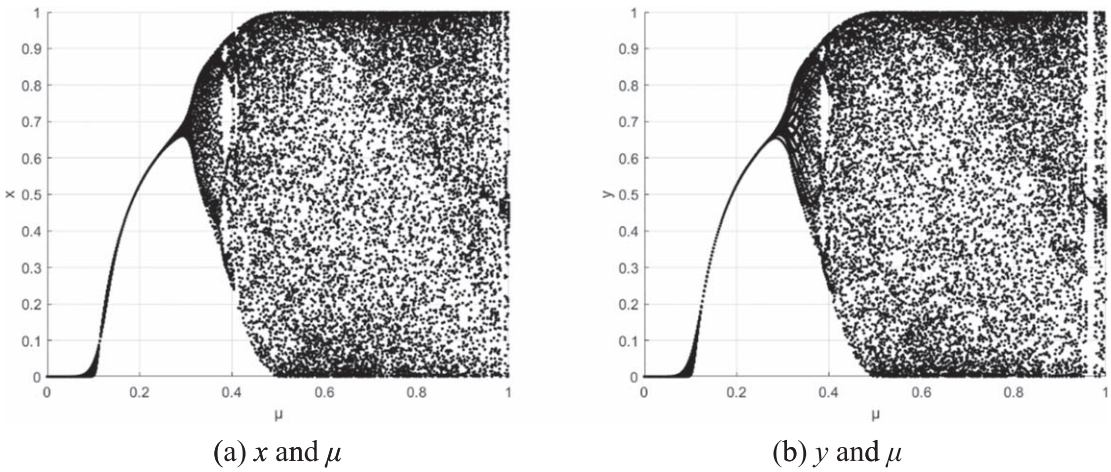
$$\begin{cases} p_{i+1} = \sin(\pi\mu(q_i + 3)p_i(1 - p_i)) \\ q_{i+1} = \sin(\pi\mu(p_i + 3)q_i(1 - q_i)) \end{cases},$$

where the systems parameter $\mu \in [0, 1]$, and the state variables $p, q \in (0, 1)$. Figure 3 plots the bifurcation diagrams of the chaotic system with the initial values $x_0 = 0.2$ and $y_0 = 0.8$. When $\mu = 0.7$, the Lyapunov exponents are 0.7884 and 0.7495. From figure 3 and the Lyapunov exponents, it can be seen that the 2D-LASM has desirable chaotic characteristics.

2.2. 3D bit planes

The pixel values of grayscale images lie between 0 and 255, and we can convert the decimal pixel values to 8-bit binary values. The zeros or ones of different pixels located in the same bit can form bit planes, i.e. 2D bit planes, so a grayscale image has 8-bit planes. 8-bit planes of Lotus are shown in figure 4, and different bit planes contain different image information. Let the plain image be $A_{m\times n}$ in the 2D space. Each element of A can be converted into an 8-bit binary value. According to the construction principle of the 3D spatial coordinate system. We put the first plane in the XOY plane. Then, the 8 bits of each pixel are successively raised from the low position to the high position in the 3D space. For example, the pixel value of (x_1, y_1) is 156 in figure 5(a), which can be converted to '10011100' in the binary system, as shown in figure 5(b).

Furthermore, the bit image can be extended to a 3D image. Taking the Lotus image of 512×512 as an example, it is converted into the 3D bit plane with a size of $512 \times 512 \times 8$, as shown in figure 6(a), and it is reconstructed into a standard cube with the bit plane size of $128 \times 128 \times 128$, as shown in figure 6(b). Table 1 shows the images number required for the given image size to merge and form a $128 \times 128 \times 128$ cube. If there are other size images or numbers, the standard cube can be constructed by the zero-filling operation.



(a) x and μ (b) y and μ

Figure 3. Bifurcation diagrams.

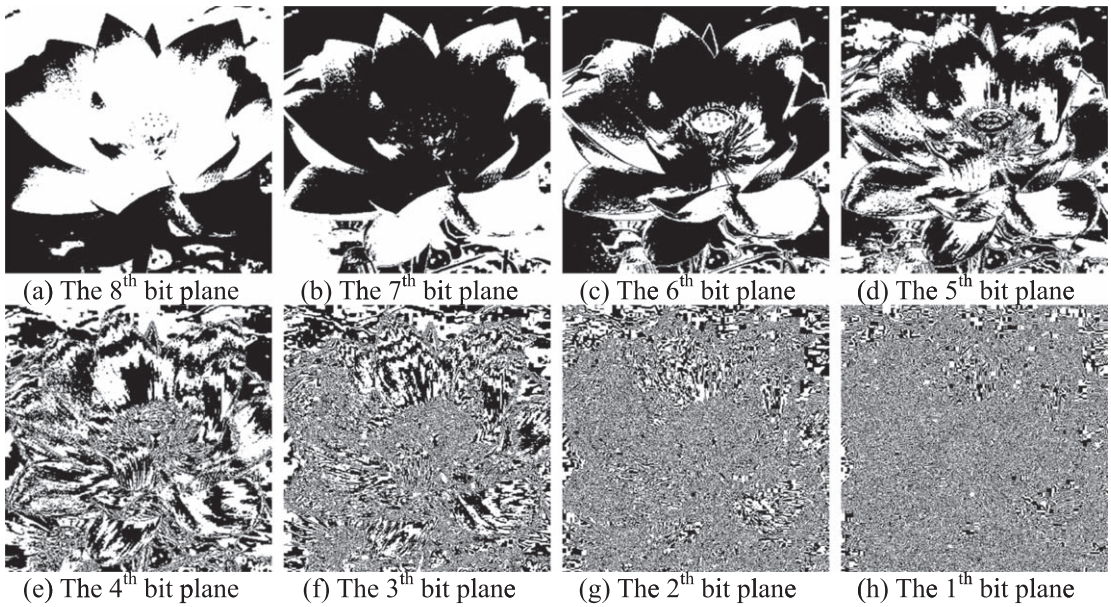
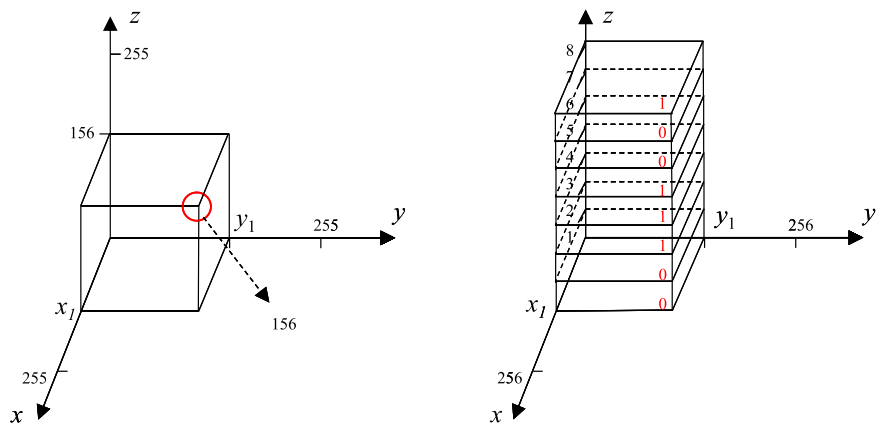


Figure 4. 8-bit planes of Lotus image.



(a) Decimal pixel in the 3D plane (b) Corresponding binary pixels in the 3D plane

Figure 5. 3D bit-level matrix structure.

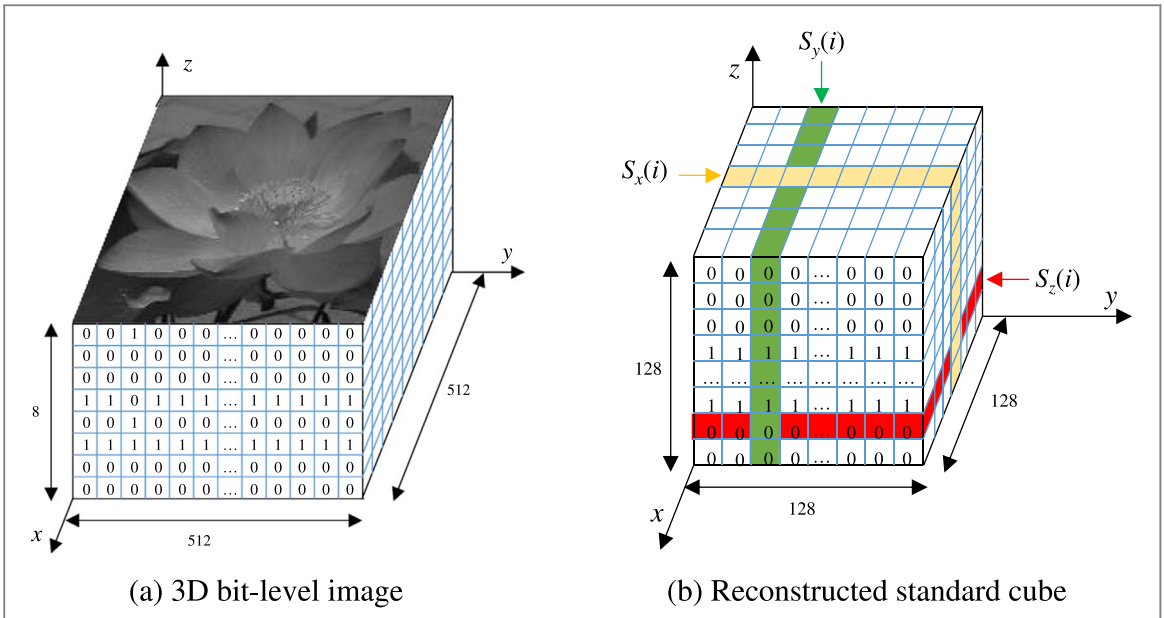


Figure 6. 3D bit planes.

Table 1. Image number.

Size of cube	Size of image	No. of images required to generate the cube
128 × 128 × 128	128 × 128	16
	256 × 256	4
	512 × 512	1

Table 2. Binary encoding rules for DNA sequences.

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

The 3D bit planes are the 2D bit planes of the 3D cube in nature, which are defined in the X, Y and Z coordinates. Therefore, the 3D bit planes include the X-bit plane, the Y-bit plane and the Z-bit plane.

(1) Elements with the same Z coordinate value are considered to be in the same bit plane, named the Z-bit plane. As shown in figure 6(b) marked in red, there are 128 Z bit planes, denoted as $S_z(i)$, $i = 1, 2, \ldots, 128$.

(2) Elements with the same Y coordinate value are considered to be in the same bit plane, named the Y-bit plane. As shown in figure 6(b) marked in green, there are 128 Y bit planes, denoted as $S_y(i)$, $i = 1, 2, \ldots, 128$.

(3) Elements with the same X coordinate value are considered to be in the same bit plane, named the X-bit plane. As shown in figure 6(b) marked in yellow, there are 128 X bit planes, denoted as $S_x(i)$, $i = 1, 2, \ldots, 128$.

2.3. Nucleic acid coding and operating rules

2.3.1. DNA coding and operating rules

DNA is a basic molecule of life that carries genetic factors to guide the development of human beings. Its basic unit is a nucleotide, which is composed of four kinds of deoxyribonucleotides arranged and combined according to certain rules to form a long chain. The arrangement rules are Adenine (A) to Thymine (T) or Cytosine (C) to Guanine (G) [28]. When DNA is encoded, each base can be represented by two binary numbers. However, there are only eight encoding rules that meet both DNA complementary rules (A-T complementary, G-C complementary) and binary complementary rules (0-1 complementary), as shown in table 2. Eight DNA decoding rules correspond to the eight encoding rules.

Table 3. DNA addition and subtraction rules.

+/−	A	T	C	G
A	A/A	T/C	C/T	G/G
T	T/T	G/A	A/G	C/C
C	C/C	A/G	G/A	T/T
G	G/G	C/T	T/C	A/A

Table 4. DNA XOR rule.

XOR	A	T	C	G
A	A	T	C	G
T	T	A	G	C
C	C	G	A	T
G	G	C	T	A

Table 5. Binary encoding rules for RNA sequences.

Rule number	1	2	3	4	5	6	7	8
A	00	00	01	01	10	10	11	11
U	11	11	10	10	01	01	00	00
C	01	10	00	11	00	11	01	10
G	10	01	11	00	11	00	10	01

Table 6. RNA XOR rule.

XOR	A	U	C	G
A	A	U	C	G
U	U	A	G	C
C	C	G	A	U
G	G	C	U	A

To increase the complexity of the encryption algorithm, we can adopt different or dynamic rules. For example, a pixel point with a value of 156 corresponds to a binary value of [10011100]. Using the first encoding rule, the DNA sequence [GCTA] can be obtained. When the fourth decoding rule is used, the corresponding binary result is [00111001]. However, when the eighth decoding rule is used, the corresponding binary result is [01100011].

DNA operations mainly refer to the DNA addition, subtraction and exclusive OR (XOR). The DNA operations are dependent on the DNA encoding rules. Through the calculation of the binary number corresponding to the base, we further get the base operation results. Different encoding rules correspond to the different DNA operation rules. Corresponding to eight DNA encoding rules, there are also eight DNA operation rules. Taking the first encoding rule as an example, table 3 shows the DNA addition and subtraction operations and table 4 shows the DNA XOR operations.

2.3.2. RNA coding and operating rules

DNA is usually a double helix structure, while RNA is usually a single strand, local of that can form a double helix structure. The bases in DNA are A, G, C and T, while in RNA there is not T, but Uracil (U). The encoding and decoding rules and operations of RNA are the same as those of DNA. The encoding rules of RNA are shown in table 5, and the RNA XOR rule corresponding to the first encoding rule 1 is shown in table 6.

2.4. Genetic central dogma

As one of the most basic and significant rules of molecular biology, the genetic central dogma promoted the study of modern biology. The famous geneticist Tan Jiazhen once said that the central principle is the second milestones in biology since Darwin came up with the theory of evolution. The genetic central dogma firstly demonstrated the biological system from the perspective of information and clarified the law of genetic information transmission in organisms. The theoretical framework of modern biology research was established, which further promoted the fundamental change of modern biology research strategy.

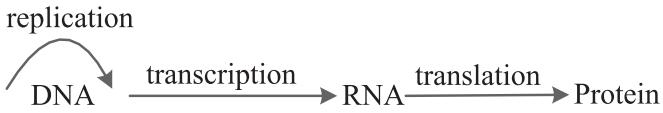


Figure 7. Schematic diagram of central dogma.

Table 7. Transcription rules.

Nucleic acid	Base			
DNA	A	T	C	G
RNA	U	A	G	C

Table 8. Translation rules.

Type	Base			
RNA	A	U	C	G
Amino acid	U	A	G	C

Table 9. RNA mutation modes.

Initial value	A	U	G	C
Mutation mode 0	A	U	G	C
Mutation mode 1	U	A	C	G
Mutation mode 2	G	C	A	U
Mutation mode 3	C	G	U	A

In 1958, Crick firstly proposed the genetic central dogma, which contains two basic contents in this version: DNA self-replication and protein synthesis. Most organisms store genetic information in DNA. In most cases, the genetic information is encoded by DNA. It flows from DNA to RNA via transcription, and from RNA to protein via translation [29].

The genetic central dogma originally described by Crick is shown in figure 7.

Transcription is the process of transforming genetic information from DNA to RNA. Transcription is a synthetic step of messenger RNA (mRNA) and non-coding RNA (tRNA, rRNA, etc.). In transcription, a DNA gene is read and copied into mRNA, which is completed by RNA polymerase and transcription factors. The transcription rules are shown in table 7.

When the mRNA matures, it moves closer to the ribosome for translation. The ribosome reads the message on the mRNA with three codons. The complex of initiation and elongation factor brings the aminoacyl tRNA (tRNAs) into the ribosome-mRNA complex. Amino acids can be added according to the codon sequence on the mRNA when the codon on the mRNA matches the anti-codon on the tRNA. Once the amino acids are joined together into a peptide chain, they fold until the original polypeptide chain is released from the ribosome to form a mature protein. The translation rules are shown in table 8.

2.5. RNA mutation

The RNA mutation here refers to the possibility of replication errors during the process of RNA mismatching, which can cause changes in the RNA sequence and display new traits. RNA is a single-stranded structure, which is prone to mutation than the double helix structure of DNA and cannot be repaired after mutation. Therefore, RNA mutation exists widely in biology.

The RNA mutation model can be realized by the base substitution operation. In figure 8, a base in the RNA sequences at a random site performs the A $\rightarrow$ U mutation operation, and a new RNA sequence S' is generated. The four modes of variants are designed in table 9.

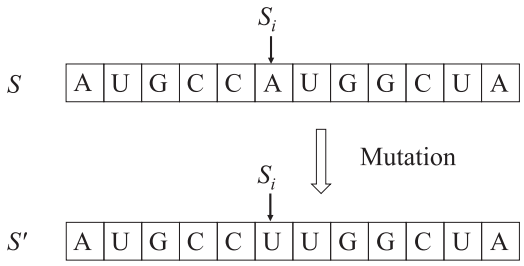


Figure 8. RNA mutation.

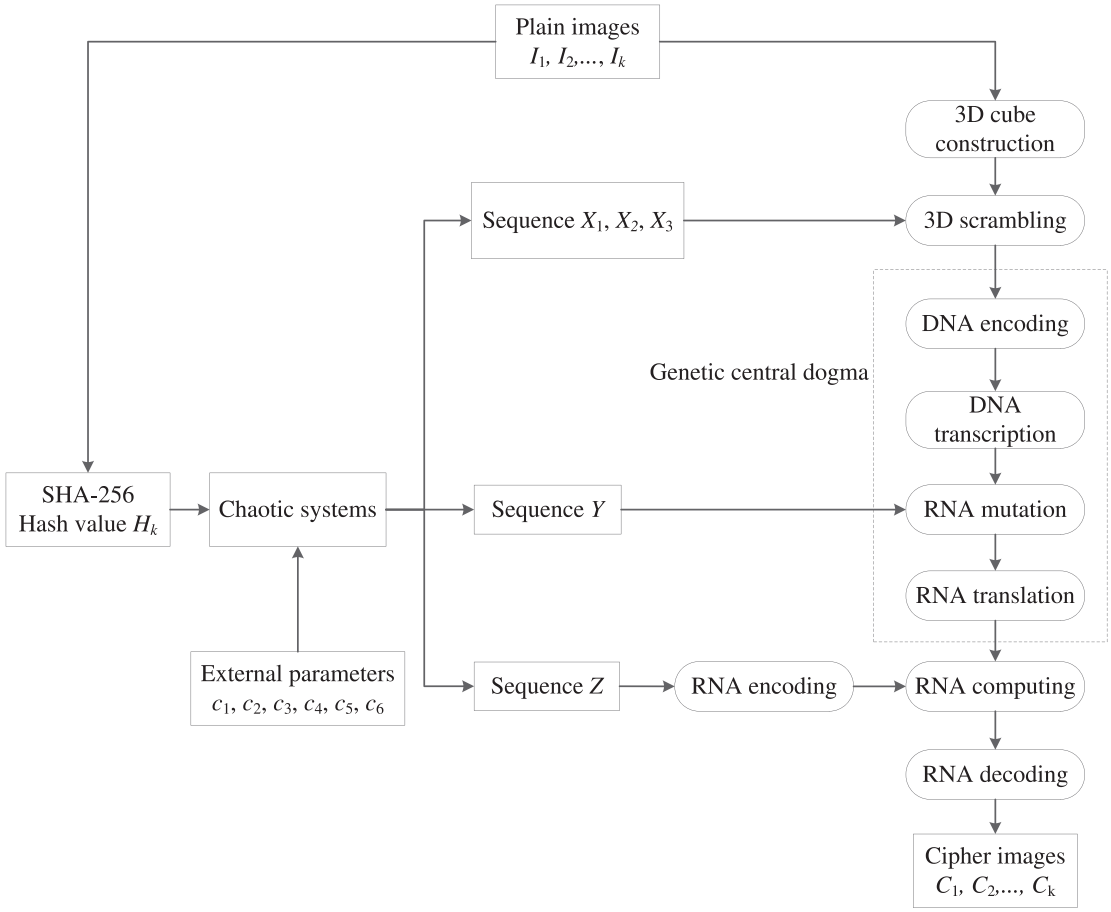


Figure 9. Flowchart of the image encryption process.

3. Proposed algorithm

3.1. Key parameter generation

To make the connection between the algorithm and the plain image strong and able to resist plain-text attacks, we introduce SHA-256. The hash value of the plain image is involved in the generation of chaotic initial values and parameters together with the external keys input by the user. Since the hash value is very sensitive to plain images, even the smallest difference in two images can produce a completely different hash value. SHA-256 hash value H_k of plain images is divided into 8-bit blocks,

$$H_k = k_1, k_2, ..., k_{32}, \tag{1}$$

where $k_i, i = 1, 2, \ldots, 32$ are 8-bit blocks.

Six intermediate parameters $h_1, h_2, h_3, h_4, h_5, h_6$ can be derived as

$$\begin{cases} h_1 = c_1 + k_1 \oplus k_2 \oplus k_3 \oplus k_4 \oplus k_5/256 \\ h_2 = c_2 + k_6 \oplus k_7 \oplus k_8 \oplus k_9 \oplus k_{10}/256 \\ h_3 = c_3 + k_{11} \oplus k_{12} \oplus k_{13} \oplus k_{14} \oplus k_{15}/256 \\ h_4 = c_4 + k_{16} \oplus k_{17} \oplus k_{18} \oplus k_{19} \oplus k_{20}/256 \\ h_5 = c_5 + k_{21} \oplus k_{22} \oplus k_{23} \oplus k_{24} \oplus k_{25} \oplus k_{26}/256 \\ h_6 = c_6 + k_{27} \oplus k_{28} \oplus k_{29} \oplus k_{30} \oplus k_{31} \oplus k_{32}/256 \end{cases}, \tag{2}$$

where c_1, c_2, c_3, c_4, c_5 and c_6 are the external keys input by the user and $\oplus$ denotes the XOR operation.

Further, using the obtained $h_1, h_2, h_3, h_4, h_5, h_6$, the five initial values x_0, y_0, z_0, p_0 and q_0 of the chaotic systems can be computed via

$$\begin{cases} x_0 = \mod((h_1 + h_2 + h_5) \times 10^8, 256)/255 \\ y_0 = \mod((h_3 + h_4 + h_6) \times 10^8, 256)/255 \\ z_0 = \mod((h_1 + h_2 + h_3 + h_4) \times 10^8, 256)/255. \\ p_0 = \mod((h_1 + h_2 + h_3) \times 10^8, 256)/255 \\ q_0 = \mod((h_4 + h_5 + h_6) \times 10^8, 256)/255 \end{cases} \tag{3}$$

Besides, the four parameters a, b, c and μ of the chaotic systems can be obtained as

$$\begin{cases} a = \mod\left(\dfrac{h_1 + h_2}{h_1 + h_2 + h_3 + h_4 + h_5 + h_6} \times 100, 3\right) + 1 \\ b = \mod\left(\dfrac{h_3 + h_4}{h_1 + h_2 + h_3 + h_4 + h_5 + h_6} \times 100, 3\right) + 1 \\ c = \mod\left(\dfrac{h_5 + h_6}{h_1 + h_2 + h_3 + h_4 + h_5 + h_6} \times 100, 3\right) + 1 \\ \mu = \mod\left(\dfrac{h_1 + h_2 + h_3}{h_1 + h_2 + h_3 + h_4 + h_5 + h_6}, 1\right) \end{cases}. \tag{4}$$

3.2. Encryption process

The proposed algorithm consists of three stages: 3D scrambling operations, the application of the genetic central dogma and the RNA operations. Create a scenario where Alice is the sender and Bob is the recipient. Keys are transmitted through an asymmetric system, such as ECC (Elliptic Curve Cryptography). The flowchart of the image encryption process is shown in figure 9, and the detailed steps are described as follows.

3.2.1. 3D scrambling operations

Step 1. 3D cube construction.

Alice inputs k original images $I_1, I_2, \ldots, I_k$ with the size of $m \times n$. Through the construction of the standard cube as described in Subsection 2.2, she can obtain the 3D standard matrix C with the size of $u \times u \times u$ in the bit plane.

Step 2. key and chaotic sequence generation.

Alice uses the 256-bit hash value H_k of the plain images and the external parameters to obtain the initial values x_0, y_0, z_0, p_0, q_0 and control parameters a, b, c, μ of the chaotic systems by equations (1)–(4). According to x_0, y_0, z_0 and a, b, c, the 3D Sine chaos is iterated $1000 + u$ times and the first 1000 iterations are discarded to produce three chaotic sequences X_1, X_2 and X_3. According to p_0, q_0 and μ, the 2D LASM chaos is iterated $1000 + u^3/2$ times and the first 1000 iterations are discarded to produce two chaotic sequences Y and Z'. Further, the chaotic sequence Z takes the first $u^2/8$ values of Z', i.e., the Y sequence of length $u^3/2$ and the Z sequence of length $u^2/8$ are generated.

Step 3. 3D scrambling operations.

Alice uses the chaotic sequences X_1, X_2 and X_3 to scramble the 3D standard matrix C. Alice firstly operates the Z-bit plane. The chaotic sequence X_1 is sorted in ascending order. She calculates

$$S_1 = sort(x_1(i)),\ i = 1,\ 2,\ ...,\ u,$$
$$T_1 = \mathrm{mod}(S_1,\ 4) \times 90,$$

where $sort(\cdot)$ is the sorting function, and $x_1(i) \in X_1$. S_1 is the index of the sort sequence, which is used to reorder the bit planes. The value of T_1 is 0, 90, 180 and 270, which represents the rotation value of the selected bit plane. C is scrambled by

$$P_z^1(i) = imrotate\ (C_z(i),\ T_1(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $C_z(i)$ denotes the ith Z-bit plane of C, P^1 is the generated 3D matrix, and $P_z^1(i)$ denotes the ith Z-bit plane of P^1.

$$P_z^2(i) = P_z^1(S_1(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_z^1(S_1(i))$ denotes the $S_1(i)$th Z-bit plane of P^1, P^2 is the generated 3D matrix, and $P_z^2(i)$ denotes the ith Z bit plane of P^2.

Secondly, Alice operates the Y-bit plane with X_2. She calculates

$$S_2 = sort(x_2(i)),\ i = 1,\ 2,\ ...,\ u,$$
$$T_2 = \mathrm{mod}(S_2,\ 4) \times 90,$$

where $x_2(i) \in X_2$. P^2 is scrambled by

$$P_y^3(i) = imrotate(P_y^2(i),\ T_2(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_y^2(i)$ denotes the ith Y-bit plane of P^2, P^3 is the generated 3D matrix, and $P_y^3(i)$ denotes the ith Y-bit plane of P^3.

$$P_y^4(i) = P_y^3(S_2(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_y^3(S_2(i))$ denotes the $S_2(i)$th Y-bit plane of P^3, P^4 is the generated 3D matrix, and $P_y^4(i)$ denotes the ith Y-bit plane of P^4.

Finally, Alice operates the X-bit plane and calculates

$$S_3 = sort(x_3(i)),\ i = 1,\ 2,\ ...,\ u,$$
$$T_3 = \mathrm{mod}(S_3,\ 4) \times 90,$$

where $x_3(i) \in X_3$. P^4 is scrambled by

$$P_x^5(i) = imrotate(P_x^4(i),\ T_3(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_x^4(i)$ denotes the ith X-bit plane of P^4, P^5 is the generated 3D matrix, and $P_x^5(i)$ denotes the ith X-bit plane of P^5.

$$P_x^6(i) = P_x^5(S_3(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_x^5(S_3(i))$ denotes the $S_3(i)$th X-bit plane of P^5, P^6 is the generated 3D matrix, and $P_x^6(i)$ denotes the ith X-bit plane of P^6. P^6 is the final 3D scrambled matrix.

3.2.2. Application of the genetic central dogma

Step 4. DNA encoding.

The DNA encoding operation is performed on P^6. At the encoding stage, Alice focuses on the X-bit plane and calculates

$$L(i) = \mathrm{mod}(sum(P_x^6(i),\ 8)),$$

where $P_x^6(i)$ denotes the ith X bit plane of P^6. $P_x^6(i)$ is encoded by the $L(i)$th encoding rule in table 4, and then the DNA encoding matrix P^7 with the size of $u/2 \times u \times u$ is obtained.

Step 5. DNA transcription.

P^7 is transcribed to obtain the RNA matrix P^8 based on the transcription rules shown in table 2.

Step 6. RNA mutation.

Y is transformed into the integer by

$$Y^1 = floor\ (\mathrm{mod}(Y \times 10^5,\ 4)),$$

where Y^1 is the integer sequence, and the value of Y_1 is 0, 1, 2 and 3. According to some rules, Y^1 is converted into the 3D matrix Y^2 with the size of $u/2 \times u \times u$. Using Y^2 and equation (5), RNA mutation is performed on P^8.

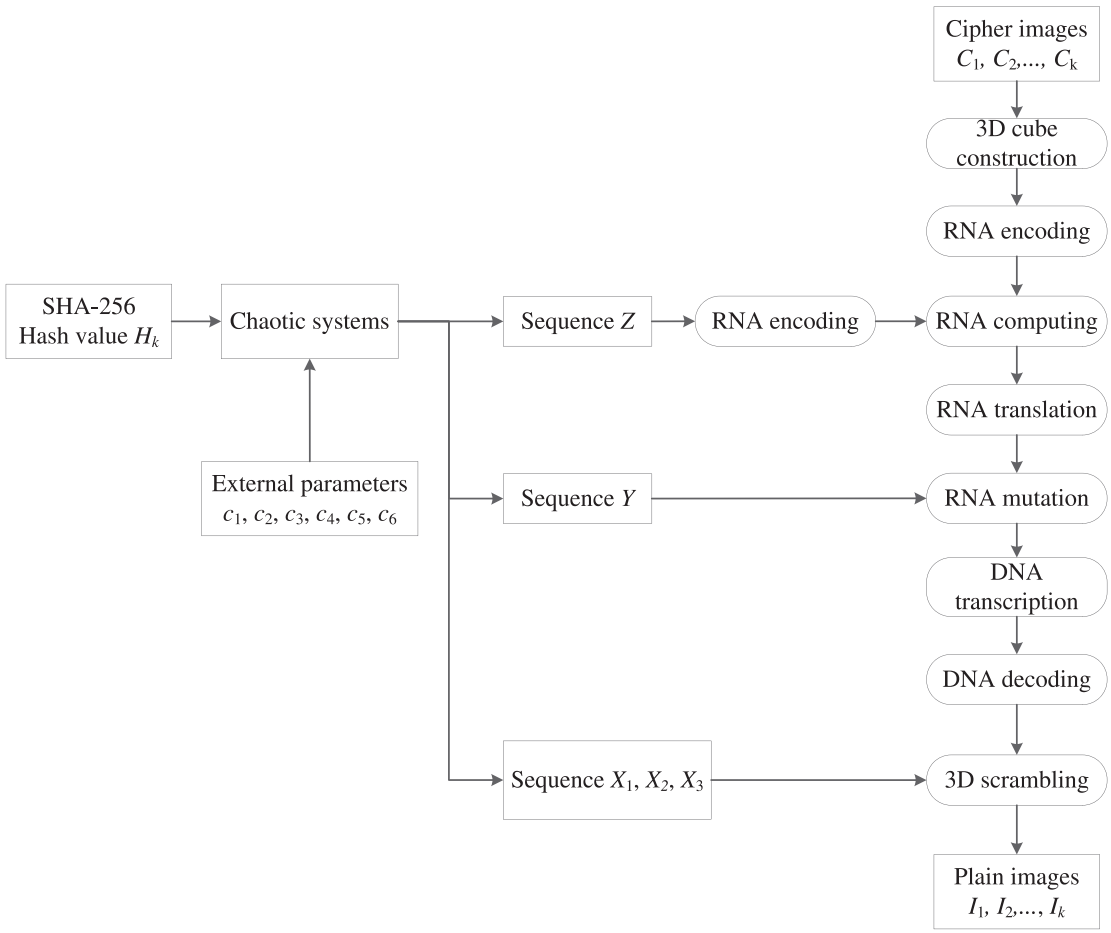


Figure 10. Flowchart of the image decryption process.

$$P^9(w_1, w_2, w_3) = \text{Mutation}(P^8(w_1, w_2, w_3), Y^2(w_1, w_2, w_3)), \tag{5}$$

where $w_1 = 1, 2, \ldots, d/2, w_2 = 1, 2, \ldots, d, w_3 = 1, 2, \ldots, d, Mutation(x), (y)$ denotes that the base x is mutated according to the mutation rule y as shown in table 9. She obtains the RNA mutation matrix P^9.

Step 7. RNA translation.

According to the translation rules shown in table 3, P^9 is translated to obtain the RNA translation matrix P^{10}.

3.2.3. RNA operations

Step 8. RNA computing.

Z is transformed into the range of 0-255 by

$$Z^1 = floor(\text{mod}(Z \times 10^5, 256)).$$

Then, the RNA matrix Z^2 is obtained by RNA encoding and adjusting the size to $(u \times u/8) \times 4$. The diffusion matrix P^{11} is obtained by the RNA XOR operation between Z^2 and P^{10}. The specific operation is designed by

$$\begin{cases} P_x^{11}(1) = Z^2 \oplus P_x^{10}(1) \\ P_x^{11}(j) = P_x^{11}(j-1) \oplus P_x^{10}(j), j = 2, 3, ..., u \end{cases},$$

where $P_x^{10}(j)$ denotes the jth X-bit plane of P^{10}, P^{11} is the generated 3D matrix, and $P_x^{11}(j)$ denotes the jth X-bit plane of P^{11}.

Step 9. RNA decoding.

RNA decoding is performed on P^{11}. At the decoding stage, Alice focuses on the Z-bit plane and calculates

$$J(i) = \mathrm{mod}(sum(P_z^{11}(i)),\ 8)),$$

where $P_z^{11}(i)$ denotes the ith Z-bit plane of P^{11}. $P_z^{11}(i)$ is decoded by the $J(i)$th decoding rule in table 7, and then the DNA decoding matrix P^{12} with the size $u \times u \times u$ is obtained. The zero elements added in Step 1 of the matrix are removed, and the 8-bit binary values in the matrix are converted to decimal pixel values to obtain the encrypted matrix C_p with the size of $u \times u \times u/8$. By reconstructing C_p, the encrypted images $C_1, C_2, \ldots, C_k$ can be obtained.

3.3. Decryption process

The decryption process is the reverse process of encryption. After Bob obtains the SHA-256 value H_k of plain images and external parameters $c_1, c_2, \ldots, c_6$ from Alice, he can decrypt the encrypted images $C_1, C_2, \ldots, C_k$ through the inverse process corresponding to the encryption process. The flowchart of the image decryption process is shown in figure 10, and the detailed steps are described as follows.

Step 1. 3D cube construction.

Bob inputs the cipher images $C_1, C_2, \ldots, C_k$. Through the construction of the standard cube as described in Subsection 2.2, he can obtain the 3D standard matrix D with the size of $u \times u \times u$ in the bit plane.

Step 2. RNA encoding.

RNA encoding is performed on D. At the encoding stage, he focuses on the Z-bit plane, and calculates

$$W(i) = \mathrm{mod}(sum(D_z(i)),\ 8),$$

where $D_z(i)$ denotes the ith Z-bit plane of D. $D_z(i)$ is encoded by the $W(i)$th encoding rule in table 7, and then Bob can obtain the DNA encoding matrix R with the size $u/2 \times u \times u$.

Step 3. RNA computing.

The RNA operation is designed by

$$\begin{cases} R_x^1(1) = Z^2 \oplus R_x(1) \\ R_x^1(j) = R_x^1(j-1) \oplus R_x(j),\ j = 2,\ 3,\ldots,u \end{cases},$$

where Z^2 is obtained in the encryption process, the diffusion matrix R^1 is obtained by the RNA XOR operation between Z^2 and R.

Step 4. RNA translation.

According to the translation rules shown in table 3, R^1 is translated to obtain the matrix R^2.

Step 5. RNA mutation.

Bob realizes the RNA mutation by

$$R^3(w_1,\ w_2,\ w_3) = Mutation(R^2(w_1,\ w_2,\ w_3),\ Y^2(w_1,\ w_2,\ w_3)).$$

He obtains the RNA mutation matrix R^3.

Step 6. DNA transcription.

R^3 is transcribed to obtain the RNA matrix R^4 based on the transcription rules shown in table 2.

Step 7. DNA decoding.

DNA decoding is performed on R^4. At the decoding stage, Bob focuses on the X-bit plane and calculates

$$H(i) = \mathrm{mod}(sum(R_x^4(i)),\ 8),$$

where $R_x^4(i)$ denotes the ith X-bit plane of R^4. $R_x^4(i)$ is decoded by the $H(i)$th decoding rule in table 4, and then he can obtain the DNA decoding matrix R^5 with the size of $u \times u \times u$.

Step 8. 3D scrambling operations.

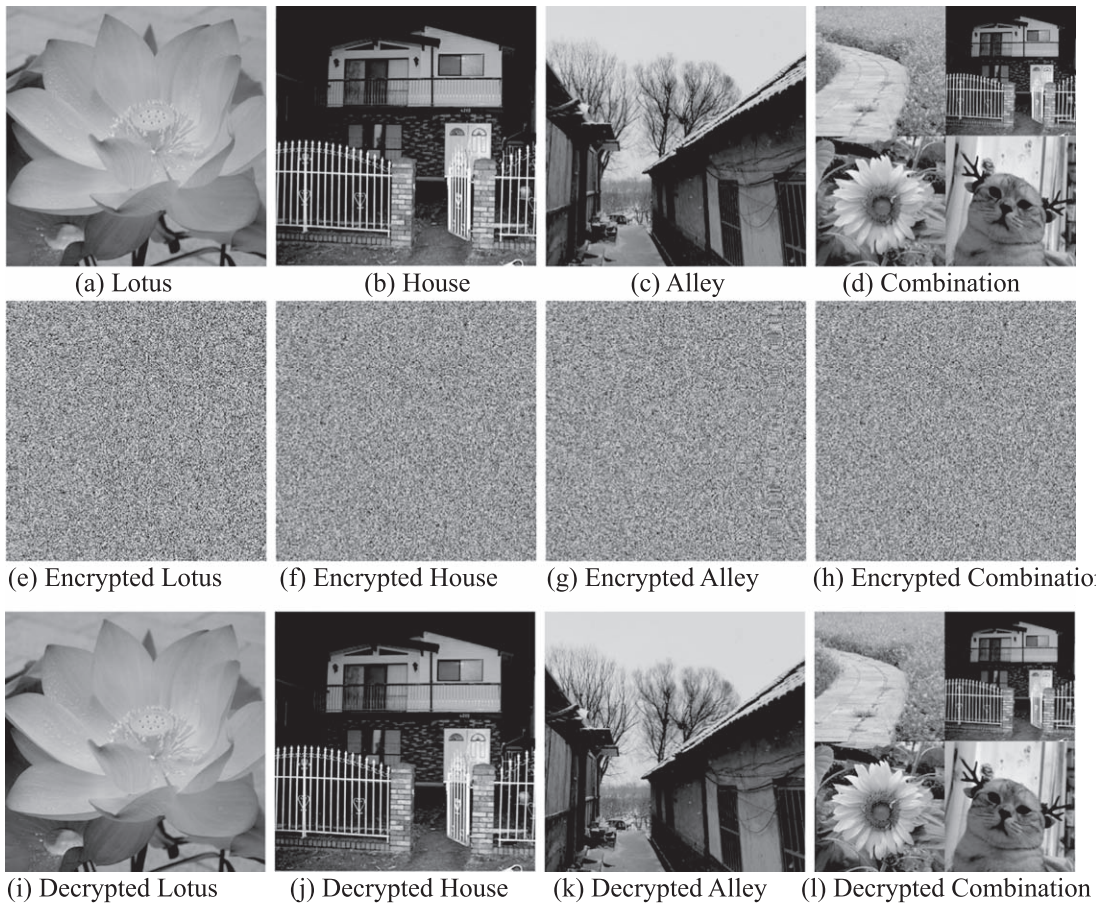


Figure 11. Experimental results.

Bob uses S_1, S_2, S_3 and T_1, T_2, T_3 obtained in the encryption process to scramble R^5. Bob firstly operates the Z-bit plane. R^5 is scrambled by

$$R_z^6(i) = imrotate(R_z^5(i),\ T_1(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_z^5(i)$ denotes the ith Z-bit plane of P^5, P^6 is the generated 3D matrix, and $P_z^6(i)$ denotes the ith Z-bit plane of P^6.

$$R_z^7(i) = R_z^6(S_1(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_z^6(S_1(i))$ denotes the $S_1(i)$th Z-bit plane of P^6, P^7 is the generated 3D matrix, and $P_z^7(i)$ denotes the ith Z-bit plane of P^7.

Secondly, Bob operates the Y-bit plane. R^7 is scrambled by

$$R_y^8(i) = imrotate(R_y^7(i),\ T_2(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_y^7(i)$ denotes the ith Y-bit plane of P^7, P^8 is the generated 3D matrix, and $P_y^8(i)$ denotes the ith Y-bit plane of P^8.

$$R_y^9(i) = R_y^8(S_2(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_y^8(S_2(i))$ denotes the $S_2(i)$th Y-bit plane of P^8, P^9 is the generated 3D matrix, and $P_y^9(i)$ denotes the ith Y-bit plane of P^9.

Finally, Bob operates the X-bit plane. R^9 is scrambled by

$$R_x^{10}(i) = imrotate(R_x^9(i),\ T_3(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_x^9(i)$ denotes the ith X-bit plane of P^9, P^{10} is the generated 3D matrix, and $P_x^{10}(i)$ denotes the ith X-bit plane of P^{10}.

$$R_x^{11}(i) = R_x^{10}(S_3(i)),\ i = 1,\ 2,\ ...,\ u,$$

where $P_x^{10}(S_3(i))$ denotes the $S_3(i)$th X-bit plane of P^{10}, P^{11} is the generated 3D matrix, and $P_x^{11}(i)$ denotes the ith X-bit plane of P^{11}. The 8-bit binary values are converted to decimal pixel values to generate the encrypted matrix R^{12} with the size of $u \times u \times u/8$. By reconstructing R^{12}, the decrypted images $I_1, I_2, \ldots, I_k$ can be obtained.

Table 10. External parameters.

Items	Values	
External key parameters	$c_1 = 0.6723$	$c_2 = 0.1839$
	$c_3 = 0.2343$	$c_4 = 0.0982$

4. Experiments

The proposed algorithm was implemented on Matlab R2016b software. The computer is configured with a 64-bit Windows 10 operating system, M-5Y71 CPU with 1.20 GHz processor and 8 GB of RAM. . We choose four grayscale images as test images as shown in figures 11(a)–(d), where Lotus, House, Alley are all with the size 512 × 512, and Combination is stitched by four images with the size of 256 × 256. Using the Lotus as an example, the 256-bit hash value is $H = aeb5ac108b0151490d\ 5de042d4a6cd76b3825b3ccf1a0ad936e4d24b3\ 613c75b$.

The external parameters are shown in table 10. The encrypted and decrypted images are shown in figures 11(e)–(h) and (i)–(l), respectively. It is obvious that the encrypted images look confusing and one cannot get any useful information from the visual. The decryption algorithm can recover the plain images without distortion when the key is correct and the image is not subjected to tampering attacks. Therefore, it can be concluded that the proposed algorithm has a superior encryption effect.

5. Algorithm analyses

A qualified encryption system must be able to resist a certain level of attacks and guarantee the security during image transmission. We analyzed the key space, information entropy, key sensitivity, etc, and compared our algorithm with several similar algorithms in this section.

5.1. Key space analysis

The key space is used to evaluate the capability of an algorithm to resist the brute-force attack [30]. Generally, a sufficiently large key space will make the attack ineffective. In our algorithm, the key consists of the 256-bit hash H_k of the plain images and the external parameters $c_1, c_2, \ldots, c_6$. If the computer's calculation accuracy is 10^{-14}, the key space of our algorithm is about $10^{14\times6} \times 2^{256} \approx 1.579 \times 10^{161} \approx 2^{536}$, which is much larger than the required value 2^{100} in the cryptosystem. It is obvious that the key space of the proposed algorithm is large enough to resist the brute-force attack.

5.2. Histogram analysis

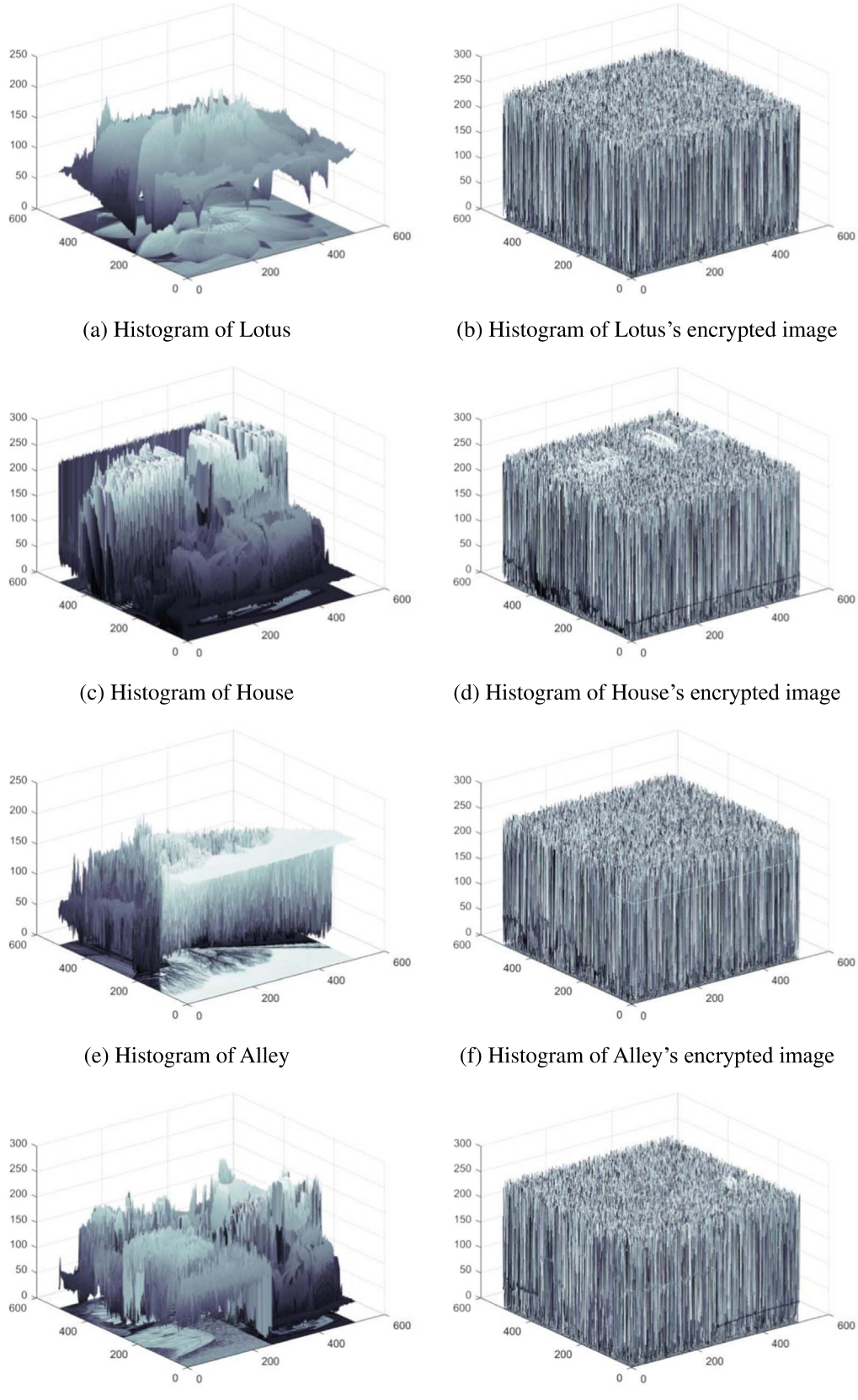
The image histogram visually represents the statistical characteristics of the image [31]. The encrypted image is supposed to be submerged in the noise and hide the original image information, to resist the statistical attack during transmission, so the histogram of the encrypted image should be evenly distributed. Figure 12 below shows the 3D histograms of the plain and encrypted images of the test images. It is obvious that the 3D histograms of encrypted images are more uniform and scattered than those plain images. It can be concluded that the proposed algorithm has a strong ability to resist statistical attacks.

5.3. Chi-squared test

The histogram allows us to perceive the effect of the algorithm encryption from a visual point of view, showing the homogeneity of the pixel values. To further analyze the values more accurately and quantitatively, we introduce the Chi-square test as a metric. It is able to give a statistical representation of the pixel uniformity across the grayscale values. The formula of Chi-square is defined by

$$\chi^2_{test} = \sum_{L=0}^{255} \frac{(F_i - G)^2}{G},$$

where F_i and G denote the observed and theoretical frequency distributions, respectively, as well as $G = MN/256$. The Chi-square values of the encrypted images are listed in table 11. The critical value of the 5% probability for a degree of freedom of 255 is 293.2478 [32], and our algorithm is well below the critical value. Therefore, our algorithm can resist statistical attacks.



(a) Histogram of Lotus

(b) Histogram of Lotus's encrypted image

(c) Histogram of House

(d) Histogram of House's encrypted image

(e) Histogram of Alley

(f) Histogram of Alley's encrypted image

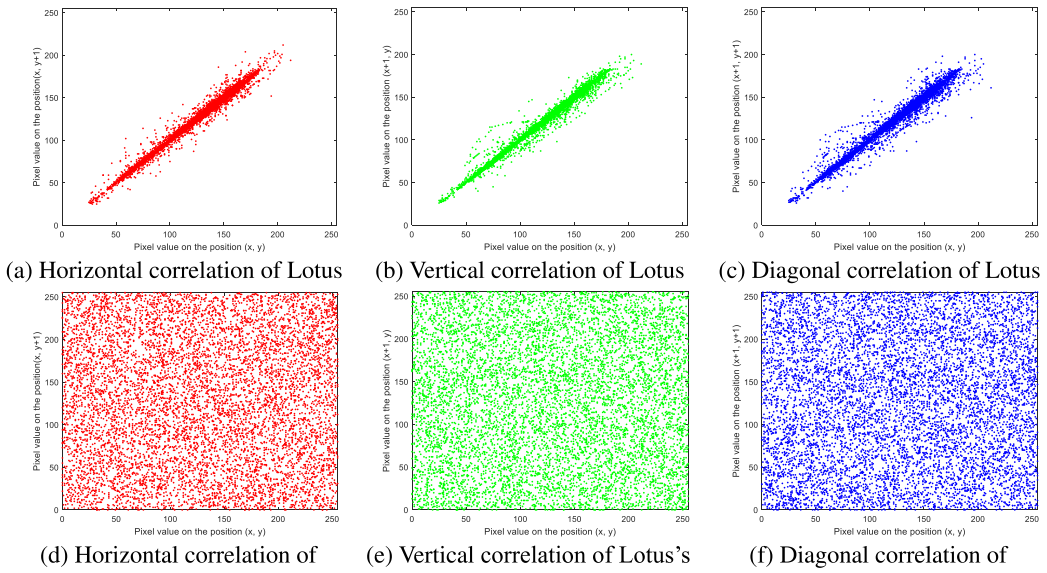
(g) Histogram of Combination

(h) Histogram of Combination's encrypted image

Figure 12. 3D histograms of the plain images and corresponding encrypted images.

5.4. Correlation of adjacent pixels

The fact that adjacent pixels are highly correlated is a prominent feature of plain images, especially the pixels of eight adjacent domains [33]. If a pixel and its adjacent pixels remain adjacent before and after encryption, it is



(a) Horizontal correlation of Lotus (b) Vertical correlation of Lotus (c) Diagonal correlation of Lotus

(d) Horizontal correlation of (e) Vertical correlation of Lotus's (f) Diagonal correlation of

Figure 13. Correlation analysis of Lotus's plain and encrypted images.

Table 11. χ^2 test about the pixel histogram.

Images	χ^2_{cipher}	χ^2_{255}	Results
Lotus	254.6152		Pass
House	274.2852	293.2478	Pass
Alley	217.9766		Pass
Combination	244.4785		Pass

highly vulnerable to area analysis attacks. The pixel correlation index can reflect the degree of image scrambling. A larger correlation between adjacent pixels, a worse scrambling effect. To calculate the correlation of two pixels adjacent vertically, horizontally and diagonally, 1000 pairs of pixels adjacent horizontally, vertically and diagonally are randomly selected from the plain image and the encrypted image. The calculation is defined by

$$r_{x,y} = \frac{E((x - E(x))(y - E(y)))}{\sqrt{D(x)D(y)}},$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i,$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2,$$

where x and y represent the pixel values of the image before and after encryption, N is the selected number of image pixels, $N = 1000$, E and D are the expectation function and variance function, respectively. Figure 13 and table 12 show the calculation results of the correlation coefficients in the horizontal, vertical and diagonal directions of the plain image and the corresponding encrypted image in the form of the figures and values, respectively.

The experimental data show that the correlation coefficients of the plain images are all greater than 0.9 and close to 1. Therefore, the plain images have a strong correlation. While the correlation coefficients of encrypted images are close to 0, the correlation between adjacent pixels is weak. The correlation intensity of adjacent pixels can be observed intuitively from figure 13. The correlation of the plain image is linearly and proportionally distributed, while the encrypted image is randomly distributed and uniformly covers the whole coordinates. Therefore, our algorithm can achieve sufficient scrambling and protect image information.

5.5. Information entropy analysis

Information entropy is an index used to evaluate the randomness of the pixel distribution in an image, and the value of information entropy represents the randomness of the image [33]. The information entropy of an 8-bit

Table 12. Correlation coefficients of plain images and encrypted images.

Algorithm	Image	Plain images			Cipher images		
		Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Proposed	Lotus	0.9933	0.9916	0.9843	−0.0051	−0.0026	0.0003
	House	0.8726	0.9466	0.8272	−0.0039	−0.0036	−0.0012
	Alley	0.9586	0.9747	0.9437	−0.0032	−0.0003	−0.0013
	Combination	0.9155	0.9546	0.8787	0.0005	−0.0001	0.0050
Reference [33]	Fruits (Red)	0.9831	0.9676	0.8404	0.0015	−0.0008	0.0004
	Fruits (Green)	0.9867	0.9705	0.8669	−0.0050	0.0060	−0.0061
	Fruits (Blue)	0.9721	0.9425	0.7486	0.0017	0.0048	−0.0016
Reference [34]	Panda	0.9601	0.9825	0.9659	0.0006	−0.0016	−0.0028
Reference [35]	Pepper	0.9629	0.9652	0.9296	0.0006	0.0038	0.0010
Reference [36]	Boat	0.9431	0.9480	0.8779	0.0045	0.0034	0.0045
Reference [37]	Pepper	0.9575	0.9639	0.9338	−0.0083	0.0081	−0.0142
Reference [38]	Couple	0.8977	0.9261	0.8643	−0.0028	0.0183	0.0265
Reference [39]	Baboon (Red)	0.9145	0.9437	0.8928	-0.0023	-0.0041	-0.0019
	Baboon (Green)	0.8242	0.8593	0.7793	0.0014	-0.0044	-0.0055
	Baboon (Blue)	0.9046	0.9125	0.8650	0.0029	-0.0033	-0.0004

Table 13. Information entropy.

Algorithm	Image	Information entropy
Proposed	Lotus	7.9993
	House	7.9992
	Alley	7.9994
	Combination	7.9993
Reference [33]	Fruits (Red)	7.9968
	Fruits (Green)	7.9970
	Fruits (Blue)	7.9970
Reference [36]	Boat	7.9972
Reference [39]	Baboon (Red)	7.9970
	Baboon (Green)	7.9973
	Baboon (Blue)	7.9969
Reference [40]	Barbara	7.9976
Reference [41]	Pepper	7.9964
Reference [42]	Pepper	7.9971

image is defined by

$$H(m) = -\sum_{i=0}^{255} P(m_i)\log_2 P(m_i),$$

where m is the ith gray level and $P(m_i)$ is the probability of m appearing in the image. The information entropy of the encrypted image obtained by a qualified encryption algorithm should be close to 8. The information entropy of the encrypted and plain image is shown in table 13. We can get that the information entropy of the encrypted image is close to the theoretical value 8. It shows that the encrypted image has good randomness, which can well hide the plain image information and improve the security during the transmission process.

5.6. Occlusion attack analysis

Some data of the encrypted images may be lost during transmission or deliberately blocked by the attacker. In the following experiment, we analyze the ability to resist occlusion attacks by cropping the encrypted image at different levels to simulate the loss of data during transmission. Figure 14 shows the encrypted images with different amounts of data loss and figure 15 shows the corresponding decrypted images. Although the decrypted images have become blurred, the main features of the plain image are still preserved. Therefore, the algorithm is robust to data loss.

5.7. Noise attack analysis

Considering the instability of the actual transmission channel, encrypted images are extremely prone to be mixed with various noises during transmission. A qualified cryptosystem must have some resistance to noise attacks. In the following experiment, we added the Gaussian noise to the encrypted image. The mean value of

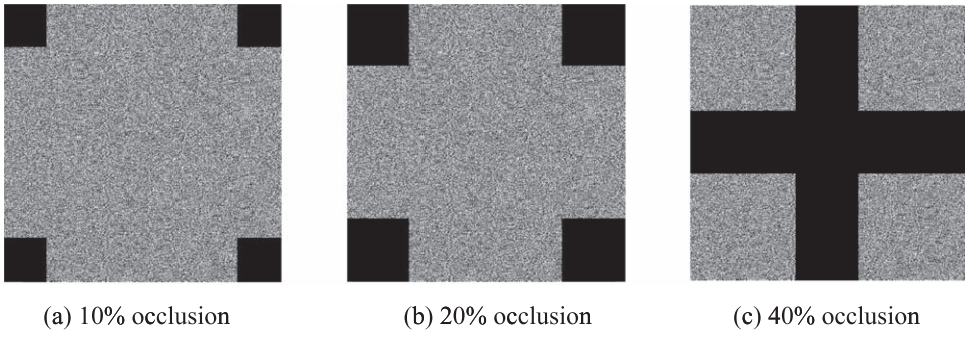


Figure 14. Occlusion attack.

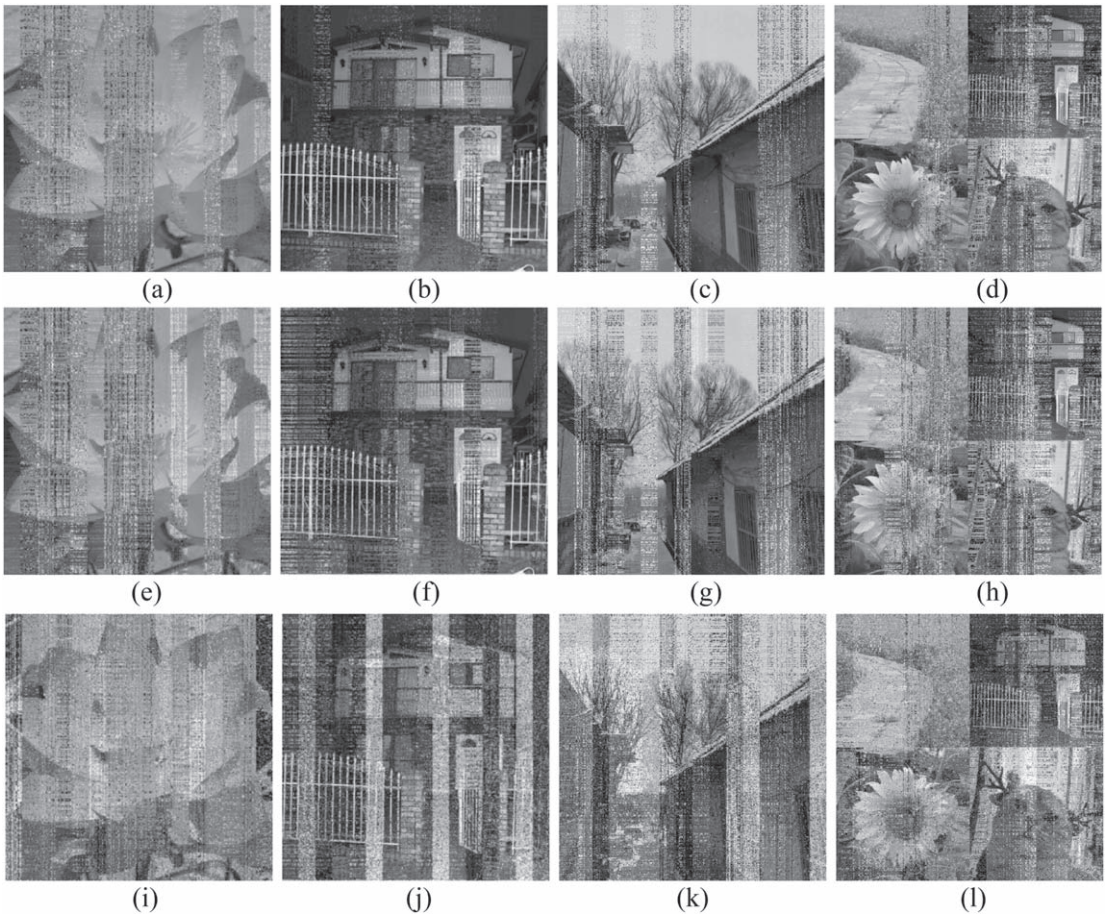


Figure 15. Decryption results: (a)–(d) decrypted images of figures 14(a); (e)–(h) decrypted images of figure 14(b); (i)–(l) decrypted images of figure 14(c).

Gaussian noise is set to 0, and the variance is set to 0.00001, 0.00003 and 0.00005. The corresponding decrypted image is shown in figure 16. It can be seen that the larger the variance, the more noise points in the decrypted image. When the variance is 0.00005, the recovered image is still distinguishable, which indicates that the proposed algorithm can withstand the noise attack to some extent.

5.8. Key sensitivity analysis

Key sensitivity characterizes the impact of a slight change in the key on the encryption algorithm. In an algorithm with high key sensitivity, when the encryption key changes slightly, different encrypted images will be generated. On the other hand, a slight change in the decryption key will result in incorrect decryption. The key of our algorithm consists of 5 initial values and 4 system parameters. Taking the Lotus image as an example, we added slight variations in the 6 decryption keys x_0, y_0, z_0, a, b and c for sensitivity analysis. Figure 17 shows the

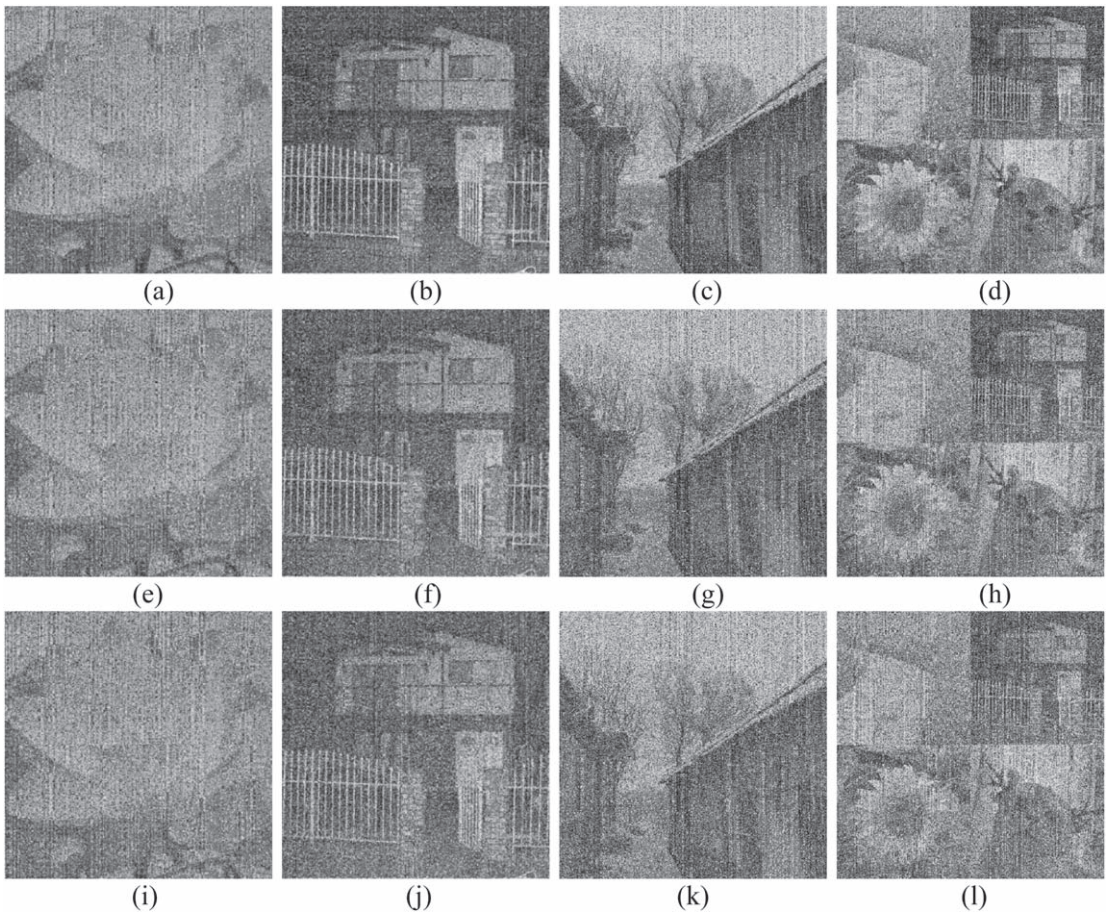


Figure 16. Decrypted images under Gaussian noise with different level of noise: (a)–(d) encrypted image with mean = 0, variance = 0.00001; (e)–(h) encrypted image with mean = 0, variance = 0.00003; (i)–(l) encrypted image with mean = 0, variance = 0.00005.

incorrectly decrypted images, and table 14 shows the difference between the incorrectly decrypted image and the plain image. It can be seen that the difference rate is greater than 99.50%. Experimental results show that the algorithm is highly sensitive to the key. Even if the key is slightly different, the encryption or decryption results will be significantly different.

5.9. Differential attack analysis

The sensitivity to the plain images is one of the important criteria to measure the competence of an encryption algorithm because it is related to the resistance of the encryption algorithm to differential attacks. That is, a qualified encryption system should generate an encrypted image that is significantly different from the original encrypted image when the plain image is slightly changed. The indicators of the algorithm's ability to resist differential attacks are the Number of Pixel Change Rate (NPCR) and Unified Average Changing Intensity (UACI), which are defined by

$$NPCR = \frac{\sum_{i,j} D(i,j)}{m \times n} \times 100\%,$$

$$UACI = \frac{\sum_{i,j} |C_1(i,j) - C_2(i,j)|}{m \times n \times 255} \times 100\%,$$

where $C_1(i,j)$ and $C_2(i,j)$ denote the encrypted image of two plain images with only one pixel difference, and $D(i,j)$ is defined by

$$D(i,j) = \begin{cases} 0, & C_1(i,j) = C_2(i,j) \\ 1, & otherwhise \end{cases}.$$

The ideal values of NPCR and UACI are 99.61% and 33.46%, respectively [38]. We can see that the test values of the four sample images are very close to the ideal values. In addition, we compared our algorithm with other different algorithms. The results are shown in table 15. It can be seen that the performance of our algorithm is

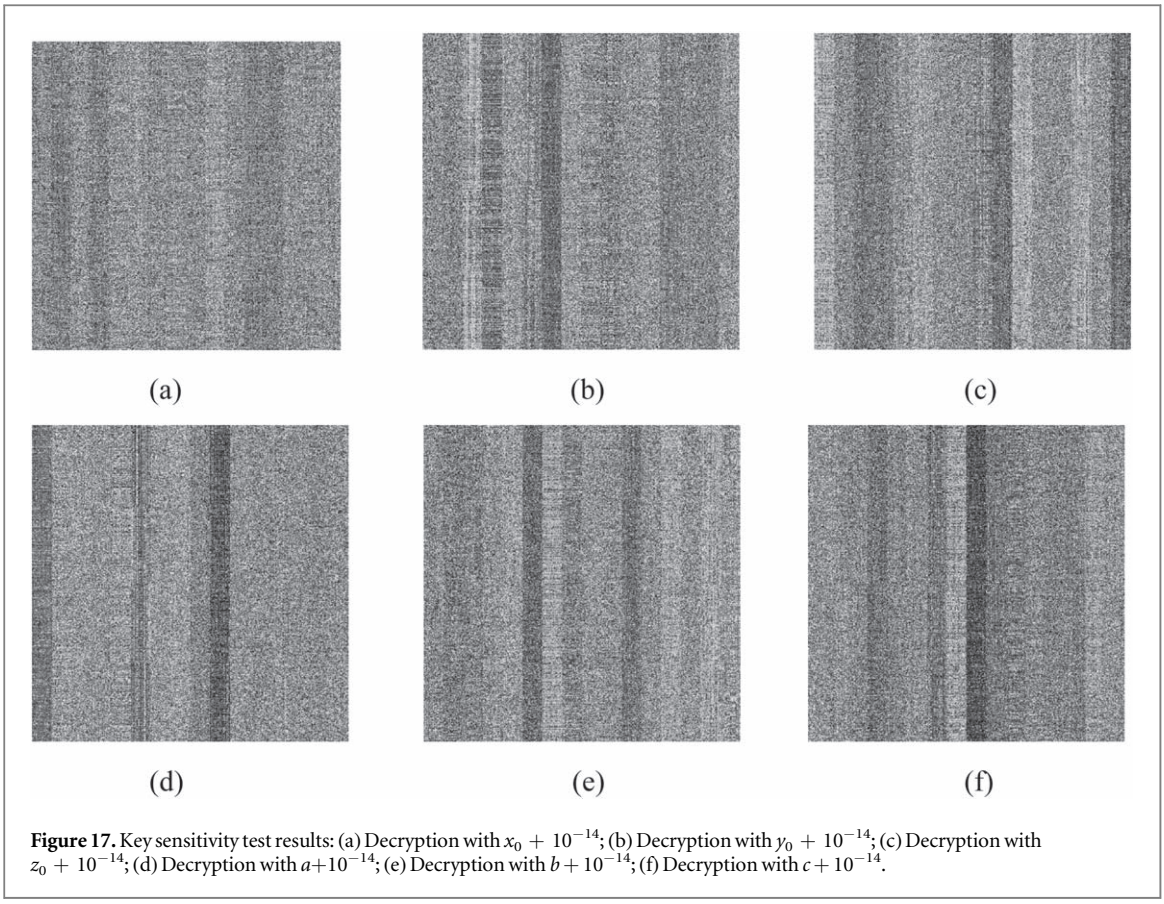


Figure 17. Key sensitivity test results: (a) Decryption with $x_0 + 10^{-14}$; (b) Decryption with $y_0 + 10^{-14}$; (c) Decryption with $z_0 + 10^{-14}$; (d) Decryption with $a+10^{-14}$; (e) Decryption with $b + 10^{-14}$; (f) Decryption with $c + 10^{-14}$.

Table 14. Difference between decryption results generated from slightly modified keys.

Figure	Modified key	Pixel difference ratios
Figure 17(a)	$x_0 + 10^{-14}$	99.58%
Figure 17(b)	$y_0 + 10^{-14}$	99.62%
Figure 17(c)	$z_0 + 10^{-14}$	99.60%
Figure 17(d)	$a+10^{-14}$	99.59%
Figure 17(e)	$b+10^{-14}$	99.59%
Figure 17(f)	$c+10^{-14}$	99.62%

closer to the ideal value. Therefore, a slight change in the plain image will produce a completely different encrypted image, and the algorithm has excellent resistance to differential attacks.

5.10. Chosen-plaintext attack

To measure the algorithm security, it is assumed that the attacker knows the design of the encryption system and further attacks the algorithm by analyzing the relationship between before and after image encryption. The common attacks are the known-plain text attack, chosen plain-text attack, ciphertext only attack and chosen cipher-text attack. If the algorithm resists the chosen plain-text attack, then the algorithm will be able to resist the rest of the attacks [46]. In the proposed algorithm, we have used the SHA-256 to generate the keys, which are different for different images. For example, we changed the pixel value 132 Lotus at the position (1, 1) for 43 to obtain a changed Lotus. We encrypt Lotus and the changed Lotus separately with the proposed algorithm. The encryption results are shown in figures 18(a) and (b). Figure 18(c) represents the difference between figures 18(a) and (b). Figures 18(d)–(f) shows the histograms of figures 18(a)–(c), and it can be seen that the pixel distributions of figures 18(d) and (e) tend to be homogenized, but there is a large difference between them. Therefore, the proposed algorithm has a high sensitivity to plain images.

5.11. Computational complexity analysis

The processing objects are k original images with the size of $m \times n$ in our algorithm. The algorithm processing time is mainly allocated to two major parts. The first part is the series of operations of chaotic sequences. This

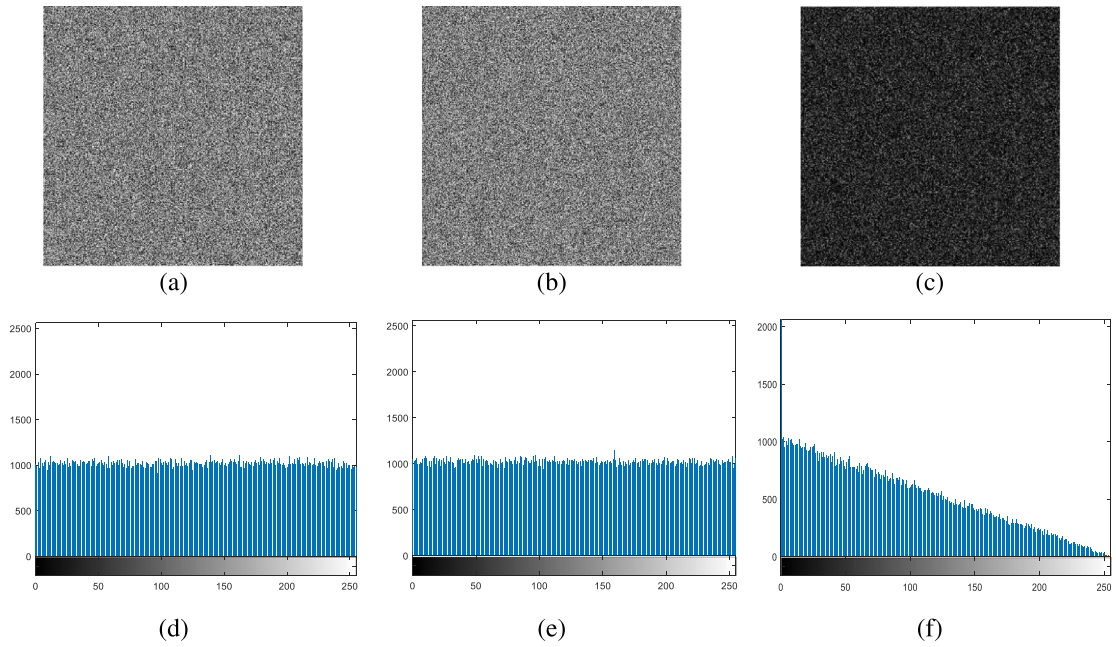


Figure 18. Chosen-plaintext attack results (a) Lotus's encrypted image; (b) Changed Lotus's encrypted image; (c) the difference between (a) and (b); (d)–(f) histograms of (a)–(c), respectively.

Table 15. NPCR and UACI values.

Algorithm	Image	NPCR (%)	UACI (%)
Proposed	Lotus	99.61	33.37
	House	99.58	33.39
	Alley	99.58	33.50
	Combination	99.60	33.40
Reference [33]	Fruits (Red)	99.58	33.63
	Fruits (Green)	99.61	33.33
	Fruits (Blue)	99.59	33.47
Reference [35]	Pepper	99.61	33.53
Reference [39]	Baboon (Red)	99.63	33.30
	Baboon (Green)	99.66	33.28
	Baboon (Blue)	99.66	33.31
Reference [24]	Pepper	99.61	33.44
Reference [43]	Pepper	99.61	33.43
Reference [44]	Baboon	99.62	31.22
Reference [45]	Baboon	99.61	33.49

Table 16. Time consumption of encryption algorithms.

Algorithm	Test image size	Encryption time (second)
Proposed	512 × 512 (Lotus)	0.848
	512 × 512 (House)	0.826
	512 × 512 (Alley)	0.831
	512 × 512 (Combination)	0.829
Reference [42]	512 × 512	6.186
Reference [47]	256 × 256	2.443
Reference [48]	512 × 512	1.431
Reference [49]	512 × 512	1.155

part generates five sequences by iterating two chaotic systems, i.e., the 3D sine map and the 2D-LASM map, whose maximum length is less than kmn, so the complexity of the iteration algorithm is $O(k \times m \times n)$. The second part is the base operations, including DNA encoding, DNA transcription, RNA translation, RNA mutation, RNA decoding, and RNA computing, and its complexity is also $O(k \times m \times n)$. Therefore, the computational complexity of our algorithm is $O(k \times m \times n)$. This is a linear computational complexity, so the time consumption depends on the size of the plain image and the number of images encrypted at once.

We conducted experiments on the time-consuming of the proposed algorithm. Since the proposed algorithm adopts a scrambling-diffusion structure, the main time-consuming is also reflected in the two stages, i.e., the scrambling stage and the diffusion stage. At the scrambling stage, the time consumption is mainly caused by the initialization and iteration of the chaotic system. At the diffusion stage, dynamic DNA encoding and calculations take major time. The specific encryption time is shown in table 16. It can be seen that the encryption time of the proposed algorithm is controlled within 0.9 second. Compared with the rest of similar algorithms, our encryption speed is significantly superior.

6. Conclusion

To improve security and encryption efficiency, this paper proposes an MIE algorithm based on genetic central dogma and 3D bit planes, which is related to the plain images. Meanwhile, the proposed algorithm is based on the complexity of the genetic central dogma and 3D bit planes, so our algorithm is very secure in theory. Experimental results and algorithm analyses demonstrated that the proposed algorithm is efficient and sufficiently secure against most common attacks, such as the brute-force attack and statistical analysis attack. Therefore, our algorithm is meant to be an excellent candidate to ensure the network security of multiple images in the fields of military, medical, educational, etc. The precision of data can affect the quality of chaotic sequences, even the algorithm performance. We will pay more attention to the chaotic degradation problem in the future. Meanwhile, we will further optimize our algorithm and improve the speed through hardware.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

ORCID iDs

Xiaoqiang Zhang https://orcid.org/0000-0002-7686-2841
Jingxi Tian https://orcid.org/0000-0002-2344-4742

References

[1] Weng S *et al* 2019 Dynamic improved pixel value ordering reversible data hiding *Inf. Sci.* **489** 136–54
[2] Zhu S, Zhu C and Wang W 2018 A new image encryption algorithm based on chaos and secure hash SHA-256 *Entropy* **20** 1–18
[3] Wang X, Le F and Zhao Hongyu 2019 Fast image encryption algorithm based on parallel computing system *Inf. Sci.* **486** 340–58
[4] Midoun M A, Wang X and Talhaoui Mohamed Zakariya 2021 A sensitive dynamic mutual encryption system based on a new 1D chaotic map *Opt. Lasers Eng.* **139** 1–13
[5] Xian Y *et al* 2020 Image encryption based on chaotic sub-block scrambling and chaotic digit selection diffusion *Opt. Lasers Eng.* **134** 1–14
[6] Chen F *et al* 2021 Multi-MSB compression based reversible data hiding scheme in encrypted images *IEEE Trans. Circuits Syst. Video Technol.* **31** 905–16
[7] Fan H *et al* 2020 Subdata image encryption scheme based on compressive sensing and vector quantization *Neural Computing & Applications* **32** 12771–87
[8] Niu Y, Zhou Z and An X Z 2020 image encryption approach based on chaotic maps and genetic operations *Multimedia Tools Appl.* **79** 25613–33
[9] Wang X and Gao S 2020 Image encryption algorithm for synchronously updating Boolean networks based on matrix semi-tensor product theory *Inf. Sci.* **507** 16–36
[10] Liu L *et al* 2019 Noise robust method for analytically solvable chaotic signal reconstruction *Circuits Syst. Signal Process.* **38** 4096–114
[11] Li C *et al* 2022 The graph structure of the generalized discrete arnold's Cat Map *IEEE Trans. Comput.* **71** 364–77
[12] Matthews R 1989 On the derivation of a 'chaotic' encryption algorithm *Cryptologia* **13** 29–42
[13] Fridrich J. 1998 Symmetric ciphers based on two-dimensional chaotic maps *International Journal of Bifurcation & Chaos* **8** 1259–84
[14] Cai Q 2019 A secure image encryption algorithm based on composite chaos theory *Trait. Signal* **36** 31–6
[15] Al-Maadeed T A *et al* 2021 A image encryption algorithm based on chaotic Lorenz system and novel primitive polynomial S-boxes *Multimedia Tools Appl.* **80** 24801–22
[16] Zheng J and Hu H 2021 A symmetric image encryption scheme based on hybrid analog-digital chaotic system and parameter selection mechanism *Multimedia Tools Appl.* **80** 20883–905
[17] Singh L-D and Singh K-M 2018 Visually meaningful multi-image encryption scheme *Arab. J. Sci. Eng.* **43** 7397–407

[18] Abanda Y and Tiedeu A 2016 Image encryption by chaos mixing *IET Image Proc.* **10** 742–50
[19] Wang X *et al* 2020 A new image encryption scheme based on coupling map lattices with mixed multi-chaos *Sci. Rep.* **10** 1–15
[20] Adleman L M 1994 Molecular computation of solutions to combinatorial problems *Science* **266** 1021–4
[21] Chai X and Chen Y 2017 Lucie Broyde. A novel chaos-based image encryption algorithm using DNA sequence operations *Opt. Lasers Eng.* **88** 197–213
[22] Chen L, Li C and Li C 2022 Security measurement of a medical communication scheme based on chaos and DNA coding *J. Visual Commun. Image Represent.* **83** 1–13
[23] Xue O *et al* 2020 A color image encryption method based on memristive hyperchaotic system and DNA encryption *Int. J. Mod. Phys.* **34** 1–29
[24] Cun Q *et al* 2021 Selective image encryption method based on dynamic DNA coding and new chaotic map *Optik* **243** 1–16
[25] Zhu C *et al* 2020 An image encryption algorithm based on 3-D DNA level permutation and substitution scheme *Multimedia Tools Appl.* **79** 7227–58
[26] Valandar M Y *et al* 2019 An integer wavelet transform image steganography method based on 3D sine chaotic map *Multimedia Tools Appl.* **78** 9971–89
[27] Hua Z and Zhou Y 2016 Image encryption using 2D Logistic-adjusted-Sine map *Inf. Sci.* **339** 237–53
[28] Watson J D and Crick F H 1953 Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid *Nature* **269** 737–8
[29] Sustar P 2007 Crick's notion of genetic information and the 'central dogma' of molecular biology *Br. J. Phil. Sci.* **58** 13–24
[30] Chai X *et al* 2021 An efficient approach for encrypting double color images into a visually meaningful cipher image using 2D compressive sensing *Information Sciences* **556** 305–40
[31] Chai X 2017 An image encryption algorithm based on bit level Brownian motion and new chaotic systems *Multimedia Tools Appl.* **76** 1159–75
[32] Sakshi Patel V *et al* 2021 Colour image encryption based on customized neural network and DNA encoding *Neural Computing and Applications* **33** 14533–50
[33] Chai X *et al* 2019 A color image cryptosystem based on dynamic DNA encryption and chaos *Signal Process.* **155** 44–62
[34] Rehman A U *et al* 2019 Block mode image encryption technique using two-fold operations based on chaos, MD5 and DNA rules *Multimedia Tools Appl.* **78** 9355–82
[35] Wu J, Liao X and Yang B 2018 Image encryption using 2D Hénon-Sine map and DNA approach *Signal Process.* **153** 11–23
[36] Wang T and Wang Minghui 2020 Hyperchaotic image encryption algorithm based on bit-level permutation and DNA encoding *Opt. Laser Technol.* **132** 1–13
[37] Wang X and Zhao M 2021 An image encryption algorithm based on hyperchaotic system and DNA coding *Opt. Laser Technol.* **143** 1–14
[38] Xu M and Tian Z 2019 A novel image cipher based on 3D bit matrix and latin cubes *Inf. Sci.* **478** 1–14
[39] Chai X *et al* 2021 Combining improved genetic algorithm and matrix semi-tensor product (STP) in color image encryption *Signal Process.* **183** 1–24
[40] Rim Z, Ridha E and Mourad Z 2021 An improved partial image encryption scheme based on lifting wavelet transform, wide range Beta chaotic map and Latin square *Multimedia Tools Appl.* **80** 15173–91
[41] Wang X and Image Y S 2021 encryption based on compressed sensing and DNA encoding *Signal Process. Image Commun.* **95** 1–12
[42] Wang X, Chen S and Zhang Y 2021 A chaotic image encryption algorithm based on random dynamic mixing *Opt. Laser Technol.* **138** 1–17
[43] Wang X and Guan N 2020 A novel chaotic image encryption algorithm based on extended Zigzag confusion and RNA operation *Opt. Laser Technol.* **131** 1–17
[44] Saljoughi A S and Mirvaziri H 2019 A new method for image encryption by 3D chaotic map *Pattern Anal. Appl.* **22** 243–57
[45] Zhao H *et al* 2020 Efficient image encryption using two-dimensional enhanced hyperchaotic Henon map *J. Electron. Imaging* **29** 1–27
[46] Malik D S and Shah T 2020 Color multiple image encryption scheme based on 3D-chaotic maps *Math. Comput. Simul* **178** 646–66
[47] Ben Farah M.A., Farah A and Farah T 2020 An image encryption scheme based on a new hybrid chaotic map and optimized substitution box *Nonlinear Dyn.* **99** 3041–64
[48] Wang X and Zhao M 2021 An image encryption algorithm based on hyperchaotic system and DNA coding *Opt. Laser Technol.* **143** 1–14
[49] Huang L *et al* 2019 On symmetric color image encryption system with permutation-diffusion simultaneous operation *Opt. Lasers Eng.* **115** 7–20