Optics and Lasers in Engineering 128 (2019) 105995

Contents lists available at ScienceDirect

Optics and Lasers in Engineering

journal homepage: www.elsevier.com/locate/optlaseng

An image encryption algorithm based on a hidden attractor chaos system and the Knuth–Durstenfeld algorithm

SiCheng Wang, ChunHua Wang*, Cong Xu

College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China

ARTICLE INFO

Key word:
Image encryption
Hidden attractor chaos system
Knuth–Durstenfeld algorithm
DNA sequence

ABSTRACT

Chaotic systems have been widely applied in digital image encryption due to their complex properties such as ergodicity, pseudo randomness and extreme sensitivity to their initial values and parameters. An image encryption algorithm based on a hidden attractor chaos system and Knuth–Durstenfeld algorithm is proposed. First, a hidden attractor chaos system is used to encrypt digital image. Compared to a self-excited attractor, the hidden attractor's attracting basin does not intersect with any small neighbourhoods of the equilibria. It is difficult for attackers to reconstruct the attractor by finding equilibrium points. Therefore, the hidden attractor chaotic system is difficult to decrypt. Meanwhile, the hidden attractor chaos system is very sensitive to initial values and parameters. Second, the Knuth–Durstenfeld algorithm has good randomness. In addition, the Knuth–Durstenfeld algorithm can reduce the time complexity and the space complexity of the permutation while achieving good permutation effects. Thus, Knuth–Durstenfeld algorithm is used to permutate the digital image. Finally, DNA sequence operations are used to diffuse image pixels values. Some experimental analyses have been applied to measure the new scheme, and the experimental results illustrate the scheme possesses better encryption performances. This method can be applied in secure image communication fields.

1. Introduction

The rapid development of digital technology and the popularity of the Internet have brought great convenience to people's work and life. Digital media as the carrier of works such as books, music, images and videos, has greatly enriched people's lives due to the easy access, convenient copying, quick spread and other advantages. However, as we can see, some of the malicious behaviors aiming to intercept useful information by exploiting the characteristics of network openness and sharing, have seriously damaged the interests of communication parties. Therefore, it is urgent to develop technology for secure information communication.

Digital images with visual visibility are an important form of digital media data and have been widely spread across the Internet. Therefore, image encryption for secure transmission of digital images over the Internet has become a widely used technology. Due to some special properties of images such as large data capacity, strong correlation of pixel points and high redundancy, the developed data encryption algorithms such as DES and AES [1,2], which require a considerable amount of time to calculate, are not suitable for the secure transmission of real-time images, thus, a large number of algorithms dedicated to image encryption have been designed based on chaotic system [3–14], deoxyribonucleic acid(DNA) sequence [13,14], cellular automata [15,16], magic cube [17] and so on. In 1998, Fridrich [18] proposed the first general architecture for chaos-based image encryption, and it was composed of permutation and diffusion. Permutation is used to break the correlation between adjacent pixels of the digital images by shuffling the positions of the image pixels, and the histogram is not changed. Diffusion alters the pixel values, so the histogram is changed.

A chaotic system possesses prominent features, including extremely sensitive dependence on initial conditions and system parameters, ergodicity and random-like behaviors, which is very suitable for image encryption. Therefore, many articles [3–14] have recently applied different chaotic systems to image encryption. For example, in Ref. [11], the researcher proposed an image encryption algorithm based on a high-dimensional hyperchaotic system, pixel-level permutation and bit-level permutation. Wu et al. [13] proposed an image encryption algorithm based on two-dimensional Hénon-Sine map. The new map possesses excellent ergodicity and pseudo randomness, and the new mapping has chaos in a wide range of parameters. Chai et al. [14] used a memristive hyperchaotic system along with the cellular automata and DNA sequence operations to encrypt the image. In general, all chaotic systems in the above image encryption algorithms are generated by self-excited attractors. For a self-excited attractor, its attracting basin is associated with an equilibrium point [20]. For chaotic systems generated by self-excited attractors, the attractors can be reconstructed in phase space by

* Corresponding author.
E-mail address: wch1227164@hnu.edu.cn (C. Wang).

https://doi.org/10.1016/j.optlaseng.2019.105995
Received 5 May 2019; Received in revised form 20 December 2019; Accepted 20 December 2019

finding equilibrium points. Thus, some attackers can reconstruct chaotic signals by reconstructing the attractors of the original chaotic system, which leads to the low security of the image encryption algorithm based on self-excited attractor chaotic systems. Recently, researchers have discovered hidden attractor chaotic systems [19–26]. The hidden attractor's attracting basin does not intersect with any small neighborhoods of the equilibria. Therefore, it is difficult for attackers to reconstruct the attractors through equilibrium points. Hence, it is difficult for attackers to decrypt the chaotic system by reconstructing the attractors in the phase space. However, to date, image encryption schemes based on hidden attractor chaotic systems are rarely reported, and only one image encryption based on hidden attractor chaotic systems is mentioned by Cavusoglu U et al. [27]. However, in Ref. [27], the author focused on the analysis of the generation of the new hidden attractor chaotic system, the analysis of the image encryption algorithm based on the hidden attractor chaotic system is very weak, and a simple image encryption is only used as an application to prove correctness of the proposed new hidden attractor chaotic system. In our paper, the image encryption algorithm based on a hidden attractor chaotic system is studied in detail. An image encryption scheme based on the hidden attractor chaotic system, the Knuth–Durstenfeld algorithm and DNA is proposed, and detailed performance analyses are performed.

Image encryption algorithms usually consist of permutation and diffusion. The diffusion process extends the image portion information to the full-text range. The permutation process can break the strong correlation of adjacent pixels of a digital image by changing the pixel position, which is very important for digital image encryption. Many researchers have performed considerable work on image permutation and proposed many effective permutation algorithms [28–30,34], such as Arnold transform, baker transform, and E-curve transform, etc. These classical methods have greatly promoted research in the field of image encryption, but some problems have been exposed in later research. For example, Arnold transformation and Baker transformation have obvious periodicity. In conclusion, the above methods proposed in [28–30,34] have problems such as poor randomness. In contrast, the Knuth–Durstenfeld algorithm has good randomness. Guvenoglu E [31] uses the Knuth–Durstenfeld algorithm for image encryption. However, in Ref. [31], the Knuth–Durstenfeld algorithm is only used to generate keys, rather than permutate images. Moreover, it is a general image encryption algorithm, but not a chaotic-based image encryption algorithm. In our paper, the Knuth–Durstenfeld algorithm is used to permutate images, and a hidden attractor chaotic system is used to generate keys. Compared to other permutation algorithms in the papers of chaos-based image encryption, this Knuth–Durstenfeld permutation algorithm in our encryption scheme exhibits good randomness.

Based on the above analyses, a new algorithm based on a hidden attractor chaos system, the Knuth–Durstenfeld algorithm and DNA sequence operations is proposed. The algorithm proposed in this paper has some advantages. First, the hidden attractor chaotic system is used to encrypt images. Therefore, attacker cannot decrypt the chaotic system by reconstructing the attractors in phase space. In addition, the hidden attractor chaotic system is very sensitive to initial values and parameters, and a slight change in parameters or initial values may lead to completely different chaotic dynamics. It is also very difficult to obtain initial values and parameters by brute force attacks. Second, the Knuth–Durstenfeld algorithm is used in permutation process. This is a completely irregular random permutation algorithm. It has good randomness. The Knuth–Durstenfeld algorithm is an in-place scrambling algorithm, so it has low algorithmic space complexity, and its algorithm time complexity is also low. We know that an image is the carrier of large amounts of data, thus, in the case of limited computing resources, low time complexity and low space complexity of algorithms are important. Using the Knuth–Durstenfeld algorithm to permutate the image, the time complexity and space complexity of algorithm can be greatly reduced under the premise of ensuring the permutation effect. Third, in the diffusion phase, uniform DNA rules are made by the scheme and a statistical characteristic of the plain image is embedded into diffusion step to resist common attacks. In addition, to improve the ability to resist known-plaintext and chosen-plaintext attacks, the SHA 256 hash function of the plain image is used to generate the secret key.

The paper is organized as follows. In Section 2, we provide preliminary works. In Section 3, we describe the proposed image encryption algorithm in detail. In Section 4, simulation results are presented. In Section 5, security analysis is presented, while conclusions are reported in Section 6.

2. Preliminary works

2.1. Chaotic system

The paper adopts a new four-dimensional hidden attractor hyperchaotic system, which is developed by the extension of the generalized non-diffusion Lorenz equation. The system does not have any equilibria, but can exhibit two-scroll hyperchaotic, chaos, quasiperiodic and periodic dynamics. For certain parameter values, coexisting hidden attractors can be observed, for example hyperchaotic and periodic hidden attractors.

The new 4-dimensional hidden attractor hyperchaotic system is described as follows [22]:

$$\begin{cases} \dot{x} = a(y - x) \\ \dot{y} = -xz - cy + kw \\ \dot{z} = -b + xy \\ \dot{w} = -my \end{cases} \tag{1}$$

where a, b, c, m, k are the real parameters of the chaotic system, and $k \times m \neq 0$. When $b \neq 0$, system (1) has no equilibria.

2.1.1. Typical hidden hyperchaotic attractors in the hidden attractor hyperchaotic system

Hyperchaotic theory states that for a four-dimensional hyperchaotic autonomous system, it has at least two positive Lyapunov exponents. When the parameters are set to $a = 10, b = 25, c = -2.5, k = 1$, and $m = 1$ and the initial conditions are set to (0.2,0.1,0.75,−2), the Lyapunov exponents of the system are $L_1 = 0.9115$, $L_2 = 0.0224$, $L_3 = 0$, and $L_4 = 0.-8.4330$ and the system has a two-scroll hyperchaotic attractor.

The strange attractors and phase portraits of the hidden attractor hyperchaotic system (1) are shown in Fig. 1. System (1) has no equilibria, and no homoclinic (heteroclinic) orbits but has a two-scroll hidden hyperchaotic attractor that resembles the butterfly shape of the Chen chaotic attractor, which as a whole form a singular tornado-like shape with two inner holes (see Fig. 1).

2.1.2. Dynamical structure of the new hyperchaotic system

System (1) exhibits abundant complex chaotic dynamical behaviors over a wide range of parameters. Of particular interest is the fact that this nonlinear system can display periodic orbit, quasi-periodic orbit, chaos, and hyperchaotic features under different conditions. Remarkably, this system can display different types of coexistence of attractors with variations of only a single parameter but with no equilibria.

When we fix $a = 10$, $b = 25$, $k = 1$, $m = 1$, and $c = -4.66$, for initial values (0.2, 0.1, 0.75, −2), a hyperchaotic attractor with no equilibria can be obtained, however, for initial values (0.2, 0.8, 0.75, −2), trajectories of system (1) coverage to chaotic attractor. When we set $a = 10$, $b = 25$, $k = 1$, $m = 1$, and $c = 2$, for initial values (0.2, 0.82, 0.75, −2), trajectories of system (1) coverage to a stable period orbit. Minor change in the initial condition of the system causes wide difference of trajectories. Therefore, the system is very sensitively to initial values.

We fix $a = 10$, $b = 25$, $k = 1$, and $m = 1$, and the initial values (0.2, 0.1, 0.75, −2). When $c \in [-8.5, -7.45]$, system (1) displays the periodic orbits, quasi-periodic orbit and periodic orbit alternately with

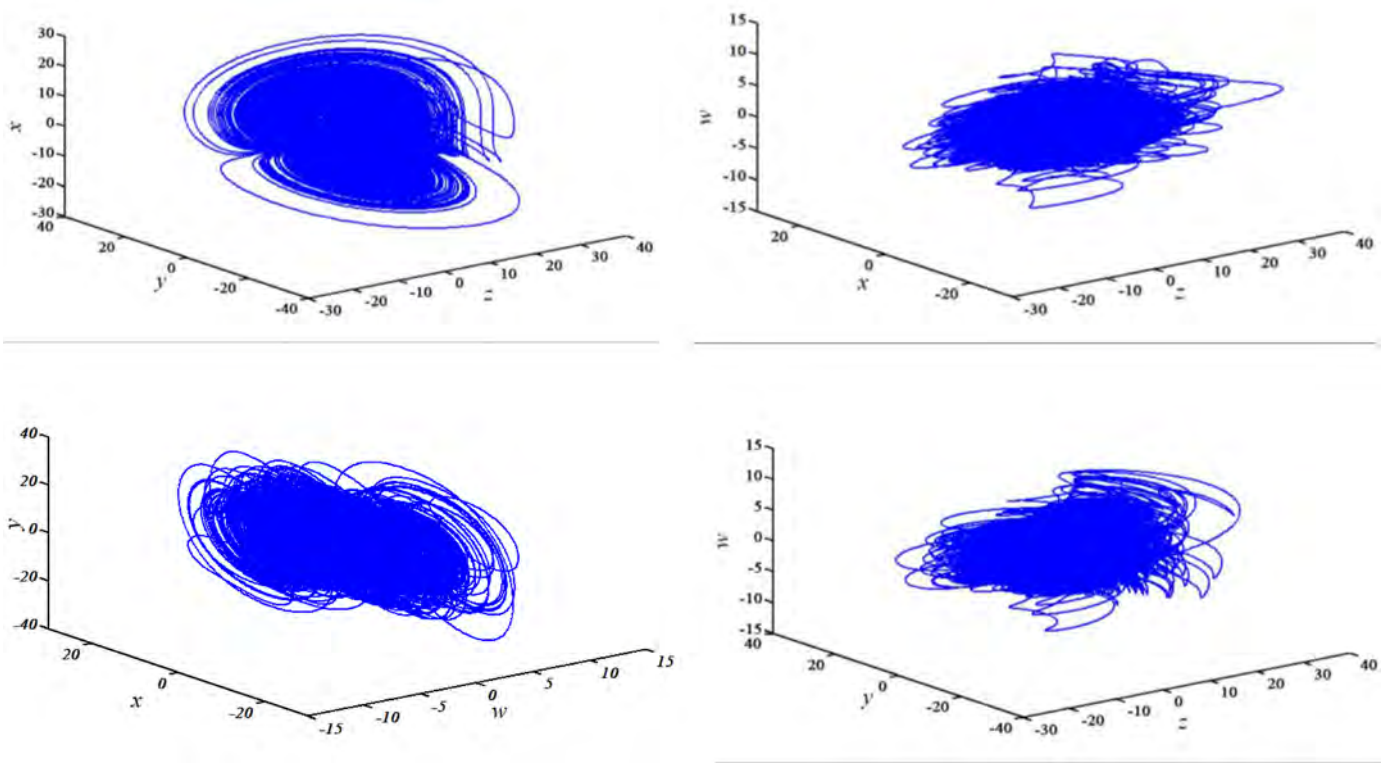

Fig. 1. Different perspectives on two-scroll hyperchaotic attractor of the 4D system (1) with no equilibria; system parameters of $a = 10$, $b = 25$, $c = -2.5$, $k = 1$, and $m = 1$; and initial values of (0.2, 0.1, 0.75, and −2).

different values of parameter c. For $c\in(-7.45, 4.96)\cup(-4.94, -4.68)\cup(-4.66, -4.12)\cup(-0.46, 0.24]$, system (1) is chaotic. When $c\in(-4.96, -4.94)\cup(-4.68, -4.66)\cup(-4.12, -0.46)\cup[1.83, 1.88]$, hyper-chaos of system (1) occurs. When $c\in(-0.24,0.154]$, system (1) displays chaos, quasi-periodic orbits and periodic orbits alternately with different values of parameter c. For $c\in[0.154,1.84)\cup(1.88,2.84)$, system (1) is periodic. When $c\in[2.84,8.54]$, the system is quasi-periodic. When $c\in(8.54,9]$, system (1) is chaotic. As the parameters change, the system exhibits different dynamic characteristics. Therefore, system (1) is very sensitive to parameters.

2.1.3. The effectiveness of the proposed chaotic system

To show that the hidden attractor chaos system is suitable for designing an image encryption algorithm, we use the National Institute of Standards and Technology (NIST) SP800-22 to test the randomness of the output sequences of the hidden attractor chaos system. The NIST SP800-22 has 15 sub-tests and each sub-test can generate a P-value. Binary streams are suggested as the input and the generated P-value is expected to fall into the range [0.01,1] to pass the corresponding sub-test. In our experiment, the double float data format is adopted for the iterative outputs of the proposed hidden attractor chaos system. For each output of the hidden attractor chaos system, we transform its fractional part to be a binary stream with 49 bits. The input binary streams are obtained by combining these binary streams from the outputs. Table 1 shows the test results, and it is clear that binary streams obtained from the outputs of the proposed hidden attractor chaos system can pass all the sub-tests. This finding indicates that the proposed hidden attractor chaos system can generate a pseudo-randomness sequence. Therefore, the hidden attractor chaos system is suitable for image encryption.

Table 1
NIST SP800-22 test results of binary sequences generated using hidden attractor chaos system.

Sub-tests		P-value ≥0.01	Result
Approximate Entropy		0.621305	Pass
Block Frequency		0.955336	Pass
Cumulative Sums	Forward	0.079280	Pass
	Reverse	0.084353	Pass
FFT		0.765214	Pass
Frequency		0.747075	Pass
Linear Complexity		0.846726	Pass
Longest Run		0.445214	Pass
Non-Overlapping Template		0.223465	Pass
Overlapping Template		0.936519	Pass
Random Excursions		0.217525	Pass
Random Excursions Variant		0.427786	Pass
Rank		0.154121	Pass
Runs		0.755034	Pass
Serial	P-value1	0.178383	Pass
	P-value2	0.422304	Pass
Universal		0.969388	Pass

2.2. Knuth–Durstenfeld shuffle algorithm

The shuffling of playing cards involves three methods, such as extract, exchange and insert. Three shuffling algorithms are derived from extract, exchange and insert. Here, extract and exchange correspond to the Fisher-Yates Shuffle and Knuth–Durstenfeld algorithms, respectively.

2.2.1. Fisher–Yates shuffle algorithms

The Fisher-Yates shuffle algorithm was proposed by Ronald A. Fisher and Frank Yates. The basic idea is to randomly take a number from the original array and place it into a new array; The details are as follows:

Step 1: Initialize the original array and the new array; The original array length is n.

Step 2: Assuming that there are still k number left in the array, then a number p between [1,k) is randomly generated.

Step 3: Take the *p*th number from the remaining *k* numbers. Then, place it in a new array.

Step 4: Repeat steps 2 and 3 until the numbers are all taken.

The new sequence taken from step3 is a scrambled sequence. The time complexity of this algorithm is $O(n \times n)$, and the space complexity is $O(n)$.

2.2.2. Knuth–Durstenfeld shuffle algorithm

Knuth and Durstenfeld improved the algorithm based on the Fisher-Yates shuffle algorithm, by interacting with numbers on the original array, and eliminating the extra $O(n)$ space. The basic idea of the algorithm is similar to the Fisher-Yates shuffle algorithm. We randomly take a number out of the unprocessed data, and then place the number in the end of the array. Thus, the number stored in the end of the array is already processed. The details are as follows:

Step 1: Create an array *arr* with an array size of *n* to store the values.

Step 2: Generate a random number *x* from 0 to *n-1*.

Step 3: Output the value of *arr* subscripted as *x*.

Step 4: Exchange the element subscripted with *x* with the element at the end.

Step 5: Similar to step2, generate a random number from 0 to *n-2*.

Step 6: Output the value of *arr* subscripted as *x*.

Step 7: Exchange the element subscripted with x with the second-to-last element.

Repeat as noted above, until *n* elements are processed.

The time complexity is $O(n)$ and the space complexity is $O(1)$. This is an in-situ disordered algorithm. The space complexity of the algorithm is improved from $O(n)$ of the Fisher-Yates shuffle algorithm to $O(1)$, and the time complexity of the algorithm is also improved from $O(n \times n)$ of the Fisher-Yates shuffle algorithm to $O(n)$.

2.3. DNA sequence operations

2.3.1. DNA encoding and decoding rules

A DNA sequence consists of four nucleic acid bases: A(adenine), C(cytosine), G(guanine) and T(thymine). A and T as well as G and C are complementary. Because 0 and 1 are complementary in the binary system, 00 and 11 are complementary. In addition, 01 and 10 are also complementary. There are 24 types of encoding rules using the four nucleic acid bases (A, C, G, and T) to encode 00,01,10 and 11. However, only 8 of them satisfy the Watson-Crick complementary rule [32] as shown in Table 2. Note that DNA decoding rule is the reverse operation of the DNA encoding rule.

Table 2
DNA encoding rules.

Rule	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

Table 3
DNA XOR operation.

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

For example, the greyscale value of a pixel is “126”, and the corresponding binary number is “01111110”. The DNA sequence “GTTC” is obtained using DNA encoding rule 2. Inversely, if the DNA sequence is “TGCA”, the binary number can be obtained by rule 8(the decoding rule is 8), that is “00011011”, the decimal number is “78”, and this is the decoding process of the DNA sequence

2.3.2. DNA XOR algebraic operation

The DNA XOR operation is manipulated according to traditional XOR in the binary format. Eight types of DNA encoding rules exist, and eight types of DNA XOR rules correspondingly appear. In this paper, XOR operation is used in the diffusion process. One type of XOR operation is used in the diffusion process. One type of XOR operation is used in the diffusion process. One type of XOR operation is shown in Table 3. An example of a DNA XOR operation is provided. Using Table 3, the XOR result of DNA sequence “AGCT” and “TGAC” is TACG.

3. Encryption scheme

The encryption process in the paper is shown in Fig. 2. First, multiple chaotic sequences are generated by the chaotic system, and two

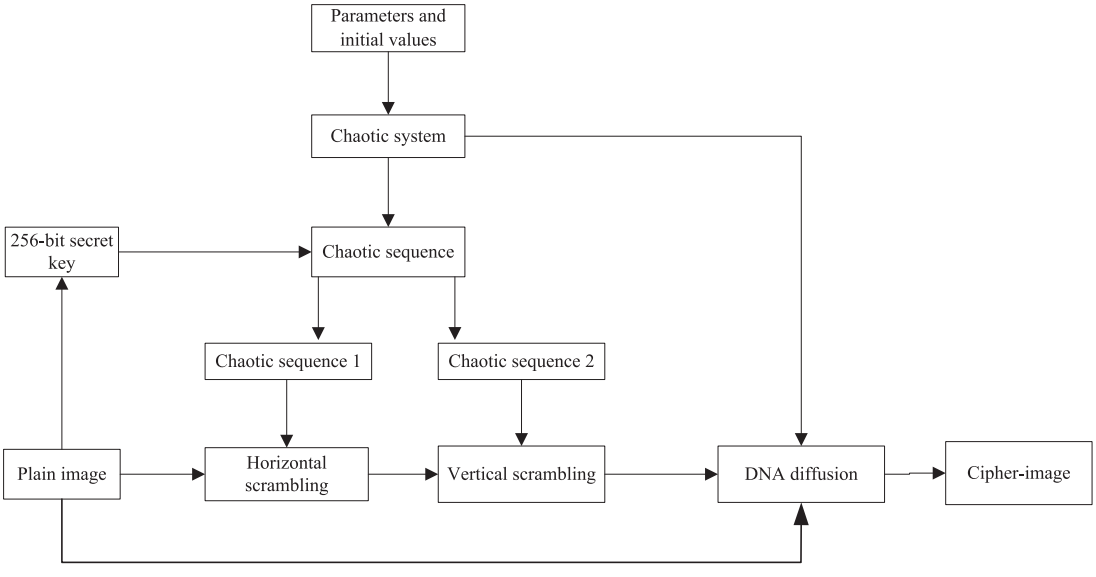

Fig. 2. The encryption processes.

chaotic sequences are selected according to the hash value of the original image. Next, the original image is permutated based on the Knuth–Durstenfeld algorithm using the selected two chaotic sequences. Finally, a cipher-image is obtained by a DNA diffusion operation. In the encryption scheme of the paper, since the parameters and initial values of the chaotic system are fixed, the method of determining the parameters and initial values of the chaotic system through plain text is not feasible. In this paper, we apply the SHA 256 function to enhance the relevance of plaintext and encryption.

3.1. Permutation process

In the permutation process, the SHA-256 hash function of the plain image is used to produce the index of the grouped chaotic sequence. For the SHA-256 function, if there is one-bit difference between two original images, their hash values will be completely different. Suppose the size of the plain greyscale image is $M \times N$, the permutation steps are as follows.

Step 1: According to the different dynamic characteristics of the hidden attractor chaotic system under different parameters and initial values, we can reasonably choose the parameters and initial values of the hidden attractor chaotic system. The hash value of the original image is calculated according to the SHA-256 algorithm. Therefore, the secret key of the encryption algorithm proposed in this paper consists of the hash value of the original image and the chaotic system's parameters and the chaotic system's initial values. The parameters selected in this paper are: $a = 10$, $b = 25$, $c = -2.5$, $k = 1$, $m = 1$; the initial value is: x0 = 0.2, y0 = 0.1, z0 = 0.75, w0 = −2. It can be seen from the dynamic characteristics of the hidden attractor chaos that the system has hidden hyperchaotic attractors when the parameters and initial values are set to the above.

Step 2: Iterate the chaotic system 1000 times with x0, y0, z0, w0, and a, b, c, k, m to avoid the transient effect. Continue to iterate the chaotic system $M \times N - 1$times to get four sequences X, Y, Z, W.

Step 3: To strengthen the correlation between the encryption scheme and the plaintext, the generated four chaotic sequences are divided into six groups, namely:A1 = (X,Y), A2 = (X,Z), A3 = (X,W), A4 = (Y,Z), A5 = (Y,W), A6 = (Z,W). Additionally, we define two vectors R1, R2. Moreover R1 = Ai (1), R2 = Ai (2), i = 1,2,3,4,5,6.

Step 4: First, we define two variables h_sum and index. According to the hash value of the original image obtained by the SHA-256 algorithm, each hexadecimal character in the hash value is converted into a decimal number, and we add all the decimal numbers converted from the hexadecimal hash value to get the h_sum value. We perform the following operations on h_sum:

Find the *index* according to the following equation: *index = mod(h_sum,6) + 1*;

When *index* = 1, then $i = 1$, $A_i = A_1$, then we get $R_1 = X$ and $R_2 = Y$;
When *index* = 2, then $i = 2$, $A_i = A_2$, then we get $R_1 = X$ and $R_2 = Z$;
When *index* = 3, then $i = 3$, $A_i = A_3$, then we get $R_1 = X$ and $R_2 = W$;
When *index* = 4, then $i = 4$, $A_i = A_4$, then we get $R_1 = Y$ and $R_2 = Z$;
When *index* = 5, then $i = 5$, $A_i = A_5$, then we get $R_1 = Y$ and $R_2 = W$;
When *index* = 6, then $i = 6$, $A_i = A_6$, then we get $R_1 = Z$ and $R_2 = W$;

Step 5: Define two vectors Row and Column. Then, according to the characteristics of the scrambling of the Knuth–Durstenfeld algorithm, R_1 and R_2 are processed as Algorithm 1:

Algorithm 1

Input: R_1, R_2
Output: Row, Column
1.for i = 1:$M \times N - 1$
2. Row(i) = mod(floor((*R_1*(i)+100) ×10^10), $M \times N$-i + 1) +1;
3. Column(i) = mod(floor((*R_2*(i)+100) ×10^10), $M \times N$-i + 1) +1;
4.end

Algorithm 2

Input: *P_Row*, and R_1
Output: Scrambled vector *P_Row*
1.for i = 1:$M \times N$-1
2. t = *P_Row*($M \times N$-i + 1);
3. P_Row($M \times N$-i + 1) = *P_Row*(*R_1*(i));
4. *P_Row*(*R_1*(i)) = t;
5.end

Step 6: Expand the original image matrix P into a one-dimensional vector *P_Row*. Then, *P_Row* is scrambled according to the Knuth–Durstenfeld algorithm by the processed chaotic sequence R_1. Detailed steps are shown in Algorithm 2.

Step 7: Convert the scrambled vector *P_Row* into a matrix of $M \times N$, and transpose the matrix. Then, the transposed matrix is expanded to get the one-dimensional vector *P_Column*.

Step 8: *P_Column* is scrambled according to Algorithm 2 by the processed chaotic sequence R_2. The sequence *P_Column* obtained after scrambling is converted into a matrix *P1* of $M \times N$.

3.2. Diffusion process

Diffusion process can enhance the resistance to statistical attack and differential attack greatly, in which the histogram of the cipher-image is fairly uniform and is significantly different from histogram of the original image. To a good diffusion process, a key stream strongly related to plain-image should be used. When encrypting different plain-images, we can get completely different result in the encryption algorithm. The diffusion process is outlined as follows.

Step 1: According to the parameters and initial values of the chaotic system during the scrambling process, we iterate the chaotic system 1000 times to avoid the transient effects of the chaotic system.

Step 2: Continue to iterate the chaotic system $M \times N$ times, and store the values in the sequence *X1, Y1, Z1, W1* to get four chaotic sequences *X1, Y1, Z1, W1*.

Step 3: By implementing the following operations to every element of *X1, Y1, Z1, W1* as described by Eq. (2)-(5), four vectors R_x, R_y, R_z, and R may be obtained.

$$R_x(i) = \mathrm{mod}\left(X1(i)'10^{10}, 8\right) + 1 \tag{2}$$

$$R_y(i) = \mathrm{mod}\left(Y1(i)'10^{10}, 8\right) + 1 \tag{3}$$

$$R_z(i) = \mathrm{mod}\left(Z1(i)'10^{10}, 8\right) + 1 \tag{4}$$

$$R(i) = \mathrm{mod}\left(W1(i)'10^{10}, 256\right) \tag{5}$$

Here, *X1*(i), *Y1*(i), *Z1*(i), and *W1*(i) denote the ith element of *X1, Y1, Z1*, and *W1*, $i \in [1, M \times N]$, and mod (a, b) returns the remainder of a divided by b.

Step 4: Expanding the scrambled matrix *P1* into a one-dimensional vector *E(i)*, $i \in [1, M \times N]$, we get a vector of $M \times N$. A variable *temp* is defined as Eq (6). In addition, we define a variable $i = 1$.

$$\mathrm{temp} = \mathrm{mod}\left(\sum_{j=1}^{M \times N} P1_i, 256\right) \tag{6}$$

Step 5: According to the coding rule corresponding to $R_z(i)$, $R(i)$ is DNA-encoded to obtain *DNA_R*(i). At the same time, according to the coding rule corresponding to *Ry*(i), *E*(i) is DNA-encoded to obtain *DNA_E*(i). Then, by XORing *DNA_R*(i) and *DNA_E*(i), we can get *New_E*(i).

Step 6: According to the rule corresponding to *Rx*(i), *New_E*(i) is decoded to obtain *de_New_E*(i). XOR the *de_New_E*(i) and *temp* to get *C_New_E*(i). Then, we modify the *temp* value to *C_New_E*(i).

Step 7: Set $i = i + 1$, do step 5–6 in a loop, until all the element of the plain image has been encrypted. Then, transform the vector to a $M \times N$ matrix, we can obtain the cipher image.

Algorithm 3
The proposed diffusion decryption algorithm.

Input: The cipher image I_c, the decryption sequence R_x, R_y, R_z, and R.
Output: Diffused decrypted sequence D_d.
1. A vector C ←Reshape I_c.
2.DNA_R ←Encode R with Rule R_z.
3.D_d 1) ←C ((1).
4. FOR i from 2 to $m \times n$
$temp$←$C(i$-1)
$D_d(i)$ ←$C(i)$ XOR $temp$
$D_d(i)$ ← Encode $D_d(i)$ with Rule $R_x(i)$
$D_d(i)$ ← $D_d(i)$ XOR $DNA_R(i)$
$D_d(i)$ ←Decode $D_d(i)$ with Rule $R_y(i)$
5.END FOR

Algorithm 4
The proposed permutate decryption algorithm.

Input: the decryption sequence X, Y. Diffused decrypted sequence D_d.
Output: The decrypted image I_d.
1.X1 ← fliplr(X), Y1 ← fliplr(Y)
2.FOR i from 1 to $m \times n$-1
$temp = D_d(i + 1)$
$D_d(i + 1) = D_d(Y1(i))$
$D_d(Y1(i)) = temp$
3.END FOR
4.FOR i from 1 to $m \times n$-1
$temp = D_d(i + 1)$
$D_d(i + 1) = D_d(X1(i))$
$D_d(X1(i)) = temp$
5.END FOR
6.I_d ← Reshape D_d to a matrix of size $m \times n$

3.3. Decryption

The decryption process is the inverse of the encryption process, and the key must be transmitted to the decryption side over a secure channel before decrypting image. The key includes the hash value of the plain image generated by the SHA-256 algorithm and the hidden attractor chaotic system's parameters and the initial values. Since in image encryption phase, we first permutate the image and then diffuse the image. Therefore, in the decryption phase, we first decrypt the diffusion, and then decrypt permutation. Prior to decryption, decryption sequences are generated using the same method as the encryption phase. The detailed diffusion decryption is presented in Algorithm 3, and the detailed permutate decryption is presented in Algorithm 4.

4. Experimental results

In this section, the standard 512×512 image of "Lena" (shown in Fig. 3(a)) is employed as the test image. All experiments are manipulated by MATLAB R2014a and we run the encryption and decryption process using a computer with a 3.3 GHz CPU, 4GB memory and Windows 10 operating system. The experiment parameters are presented in Table 4. The cipher image is shown in Fig. 3(b), and the decrypted image is illustrated in Fig. 3(c).

Table 4
Experiment parameters.

Items	Parameter values
Parameters of the hidden attractor hyperchaotic system	a = 10, b = 25, c = 4.6, k = 25, m = 1
The initial values of the hidden attractors chaotic system	X_0 = 0.2, Y_0 = 0.1, Z_0 = 0.75, W_0 = −2
256-bit secret key (in hexadecimal form)	E D D 1 B B A 8 0 B 8 E 6 0 4 7 3 7 1 5 0 8 A 2 4 5 5 3 5 7 B E 6 3 E 7 C B 7 F A B 0 0 F E F B D 6 2 0 F 4 C A 7 6 0 1 4 5 5 4

The figures show that the cipher image is a noise-like image, and there is no relationship between the original image and the cipher image. By visual observation, the decrypted image is the same as the original image. These findings show that our algorithm has good encryption and decryption effect.

5. Performance analysis

In this section, we analyse the performances of the proposed scheme, including histograms, correlation coefficients, entropy, key space analysis, key sensitivity analysis, differential analysis and known-plaintext and chosen-plaintext attacks analysis.

5.1. Key space

The key space of a good encryption algorithm should be large enough that it can resist all types of brute-force attacks from information eavesdroppers. In the proposed algorithm, the secret keys include the following:

(1) The 256-bit hash value generated by the hash function of the plain image,
(2) The parameter and the initial value of the hidden attractors chaotic system.

The key space of the SHA 256 hash function with complexity of the best attack is 2^{128} larger than 2^{100}[33], and this finding indicates that our algorithm is sufficient to prevent the exhaustive search and any brute force attack.

5.2. Statistical attack analysis

Correlation coefficient, histogram and entropy depicted in the following subsections are the three most important evaluation criteria for statistical attack analysis.

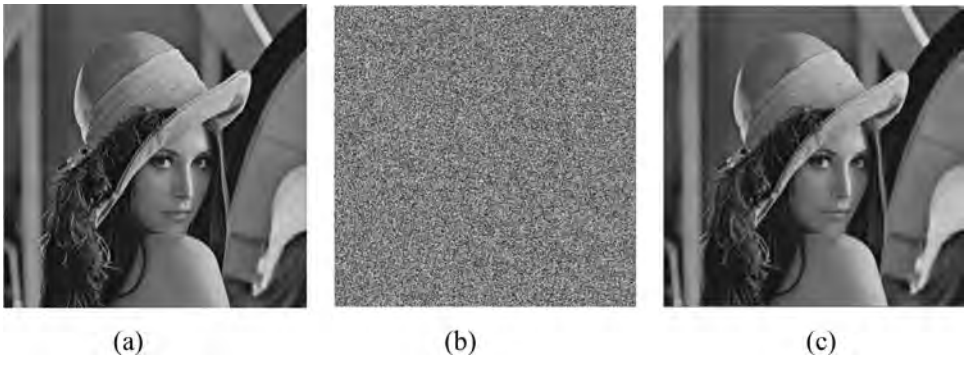

Fig. 3. Simulation results. (a) Plain image of Lena (512×512), (b) corresponding cipher image, (c) decrypted image.

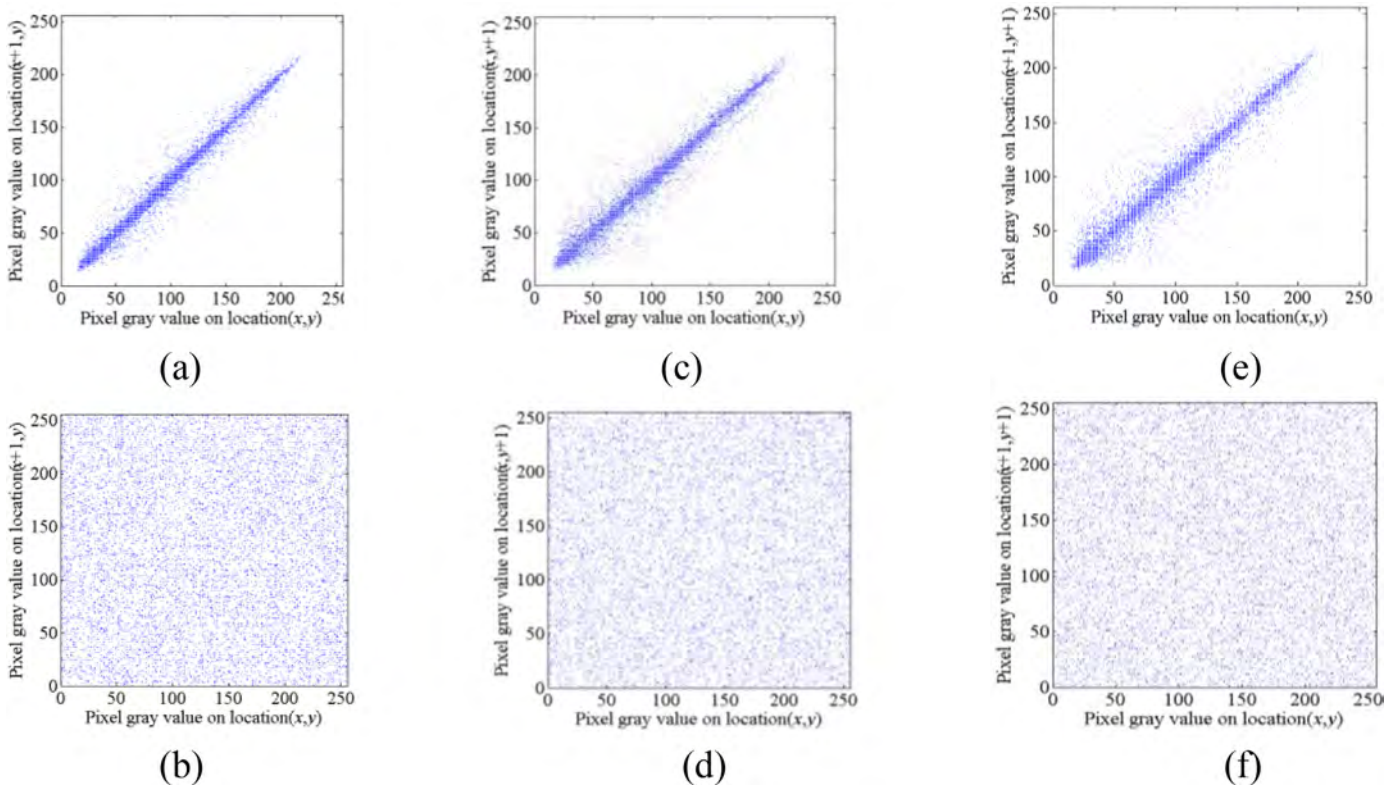

Fig. 4. Correlation of two adjacent pixels of the plain image Lena (256×256) and its cipher image. (a) Horizontal direction in plain image. (b) Horizontal direction in cipher image. (c) Vertical direction in plain image (d) Vertical direction in cipher image. (e) Diagonal direction in plain image (f) Diagonal direction in cipher image.

5.2.1. Correlation coefficient analysis

The adjacent pixels of the original image have a high correlation in the horizontal, vertical and diagonal directions. An ideal encryption algorithm can make the correlation coefficients of the pixels in the encrypted image have a sufficiently low correlation to resist statistical attacks. To analyse and compare the correlation of the adjacent pixels in the plain and cipher images, 10,000 pairs of adjacent pixels in each direction are randomly chosen from the plain image and its encrypted image. The correlation distribution of two adjacent pixels in three directions is shown in Fig. 4. As observed, the distributions of adjacent pixels in the original image are highly concentrated, which means that the original image has a strong correlation. However, the distributions of the adjacent pixels in the original image's ciphered image are random, which means that the ciphered image has a low correlation.

Moreover, we used the following formulas [34] to calculate the correlation coefficient r_{xy} of each pair:

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{7}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))^2 \tag{8}$$

$$\mathrm{cov}(x, y) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))(y_i - E(y)) \tag{9}$$

$$r_{xy} = \frac{\mathrm{cov}(x, y)}{\sqrt{D(x) \times D(y)}} \tag{10}$$

where x and y are the greyscale values of two adjacent pixels in the image, N is the total number of pixels selected from the image, and $E(x)$ and $D(x)$ denote the expectation and variance of variable x, respectively.

Fig. 4 plots the correlation of two adjacent pixels of the plain image "Lena (256×256)" and its cipher image in the horizontal, vertical and diagonal directions. Table 5 illustrates the correlation coefficients of two adjacent pixels in the plain images (shown in Fig. 5) and their corresponding cipher images. The results clearly demonstrate that the correlations between adjacent pixels in the original images are strong, and the correlation coefficients are all close to 1. However, correlation coefficients of cipher images are all less than 0.02. These findings indicate greatly reduced correlation in the cipher images. In addition, the opponents cannot obtain useful information from the cipher images by statistical attack.

5.2.2. Histogram analysis

An image histogram represents the distribution of the pixel intensity values within an image. A secure encryption system can make the encrypted image have a uniform histogram to resist any statistical attacks. The histograms of plain images and its cipher images by the proposed algorithm are shown in Fig. 6. It is clear that histogram of the cipher image is uniform and significantly different compared with the plain image. Thus, our algorithm can make the statistical attack invalid.

For quantity analyses of each key, we calculate variances of histograms to evaluate the uniformity of the distribution of the ciphered image. The lower value of variances indicates higher uniformity of the ciphered image. We also calculate the two variances of ciphered images that are encrypted by different secret keys on the same plaintext image. The closer of the two values of variances indicates the higher uniformity of ciphered images when the secret keys are varied. The variance of histograms is presented as follows:

$$\mathrm{var}(z) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}(z_i - z_j)^2 \tag{11}$$

where Z is the vector of the histogram values, and $Z = \{Z_1, Z_2, Z_3, Z_4 \ldots Z_{256}\}$. Z_i and Z_j are the numbers of pixels whose grey values are equal to i and j, respectively. In simulation experiments, we calculate

Table 5
Correlation coefficients of two adjacent pixels in the plain image and cipher image.

Images		Correlation coefficients		
		Horizontal	Vertical	Diagonal
Lena (256×256)	Plain image	0.9588	0.9260	0.9291
	cipher image	0.0004	0.0013	−0.0023
Brain (256×256)	Plain image	0.9872	0.9824	0.9768
	cipher image	0.0011	0.0029	0.0001
Baboon (256×256)	Plain image	0.8901	0.9081	0.8508
	cipher image	0.0138	0.0113	0.0053
Cman (256×256)	Plain image	0.9533	0.9199	0.8995
	cipher image	−0.0105	0.0033	−0.0037
Girl (256×256)	Plain image	0.9771	0.9715	0.9636
	cipher image	0.0030	0.0001	0.0030

Fig. 5. Test images (a) Lena (256×256) (b) Brain (256×256) (c) Baboon (256×256) (d) Cman(256×256) (e) Girl (256×256).

two variances of histograms of two ciphered images using Eq. (11) from the same plaintext image with different secret keys. Only one parameter of the secret keys is changed in such different secret keys. The variances are obtained by the initial key (x_0, y_0, w_0, z_0). The variance values are approximately 250, which indicates that the average fluctuation in number of pixels in each grey value is approximately 13 pixels. However, the variance value is 33,860.0547 for the histogram of the plaintext image Lena. It is clear that the histograms of the cipher images are fairly uniform and significantly different from that of the plain images. Therefore, it does not provide any information to the attackers.

In addition to the histogram graphic analysis, we use the chi-square test to verify the uniform histogram distribution via quantification of the cipher image. The chi-square test verifies that the statistic magnitude χ^2 obeys the chi-square distribution [35]:

$$\chi^2 = \sum_{i=1}^{k} \frac{(f_i - np)^2}{np} \tag{12}$$

where f_i is the number of pixels in i interval, n is the total number of pixels, and $p = 1/k$.

In the simulation, we set a significant level of $\alpha = 0.05$ and calculate the chi-square values and P-values of the test images. Table 6 lists the experimental results of chi-square tests for all test images and corresponding encrypted images. Table 6 shows that the P-values are greater than 0.05 for the encrypted images, and the pixel distribution is uniform. Thus, the encryption scheme does not provide useful information for attackers and can resist any statistical attacks.

5.2.3. Information entropy analysis

Information entropy is the most important measure of randomness. The source of information is defined as m, and we can obtain the following formula [34] to calculate information entropy:

$$H(m) = \sum_{i=0}^{M-1} p(m_i) \log \frac{1}{p(m_i)} \tag{13}$$

where M is the total number of symbols $m_i \in m$; $p(m_i)$ denotes the probability of symbols, and the theoretical value $H(m) = 8$ is obtained using Eq. (13). The closer the number is to 8, the less possible it is for attackers to decode cipher images. Table 6 shows the information entropy. Table 7 reveals that entropies are close to 8, thus indicating that the proposed algorithm has a good property of information entropy.

Recently, by computing the sample mean of conventional information entropy over a number of non-overlapping and randomly selected image blocks, the local Shannon entropy was proposed to measure the image randomness. Local Shannon entropy may overcome some known weaknesses of conventional information entropy. Local Shannon entropy has some advantages. First, it can capture local image block randomness that may not be correctly obtained by information entropy. Second, it is able to assess image randomness using the same set of parameters regardless of the various sizes of the test images. Finally, only a portion of the pixel information is needed to measure the image, and it has higher efficiency. Next, local Shannon entropy is used to measure the randomness of our encryption algorithm.

The (k,T_B)-local Shannon entropy with respect to local image blocks may be computed by the following steps. First, non-overlapping image blocks S_1, S_2, …, S_k with T_B pixels for a test image S are randomly selected. Then, information entropy $H(S_i)$ for all image blocks via Eq. (13) may be obtained. Finally, the local Shannon entropy over these k image blocks is computed using the following equation [36]:

$$\tilde{H}_{k,T_B}(m) = \sum_{i=1}^{k} \frac{H(S_i)}{k} \tag{14}$$

In the experiment, for three test images, we select $k = 30$ and $T_B = 1936$, and the results are presented in Table 7. Table 8 demonstrates that the local Shannon entropies of the cipher images are close to 8.0. The local image blocks are chosen randomly in the local Shannon entropy measure. Thus, the cipher images generated by the proposed encryption algorithm have good local randomness, and our algorithm is sufficiently secure to resist entropy attacks.

5.3. Key sensitivity analysis

To guarantee the security of the cryptosystem, a good cryptosystem should be sensitive to the key. The key sensitivity must be analysed. The incorrect plain image will be produced when different keys are used to decrypt the cipher image. We use the original key to encrypt the plain image and the modified key to decrypt the cipher image. Their difference lies in the last digit after the decimal point of the first parameter.

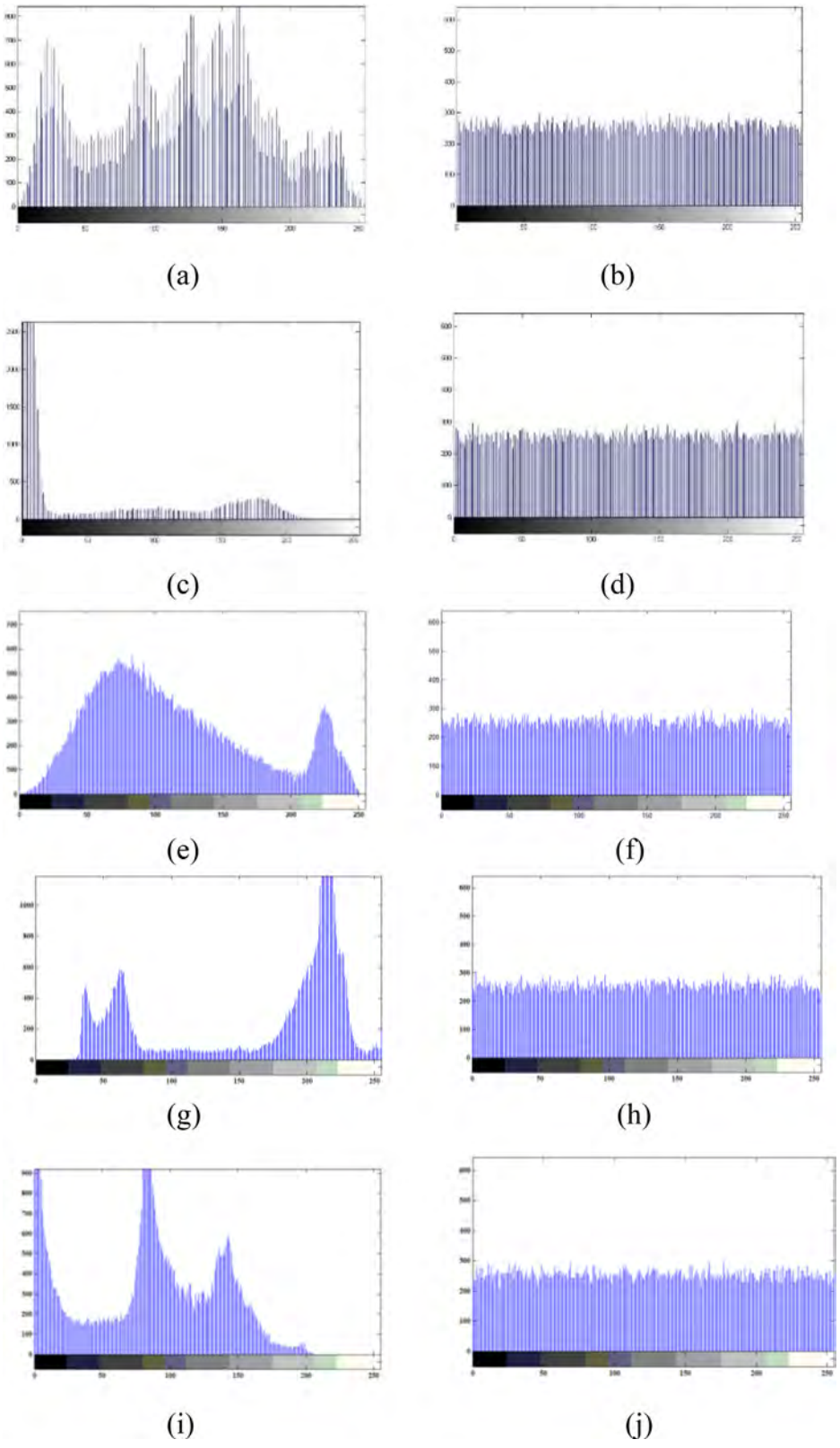

Fig. 6. Histogram analysis. (a) Histogram of the Lena image (256×256); (b) histogram of the Lena (256×256)'s encrypted image; (c) histogram of the Brain image (256×256); (d) histogram of the Brain (256×256)'s encrypted image; (e) histogram of the Baboon image (256×256); (f) histogram of the Baboon (256×256)'s encrypted image; (g) histogram of the Cman image (256×256); (h) histogram of the Cman (256×256)'s encrypted image; (i) histogram of the Girl image (256×256); (j) histogram of the Girl (256×256)'s encrypted image.

Table 6
Chi-square (χ^2) test analysis.

Image	Lena (256×256)	Brain (256×256)	Baboon (256×256)	Cman (256×256)	Girl (256×256)
χ^2	279.3040	225.9531	264.6797	269.3672	259.1016
P-values	0.7942	0.0953	0.6746	0.7434	0.5832

Table 7
The result of information entropy.

Image	Lena (256×256)	Brain (256×256)	Baboon (256×256)	Cman (256×256)	Girl (256×256)
Information entropy	7.9978	7.9973	7.9971	7.9970	7.9971

Table 8
Local entropies for the cipher images.

Images	Lena (256×256)	Brain (256×256)	Baboon (256×256)	Cman (256×256)	Girl (256×256)
Local entropies	7.9078	7.9081	7.9085	7.9089	7.9053

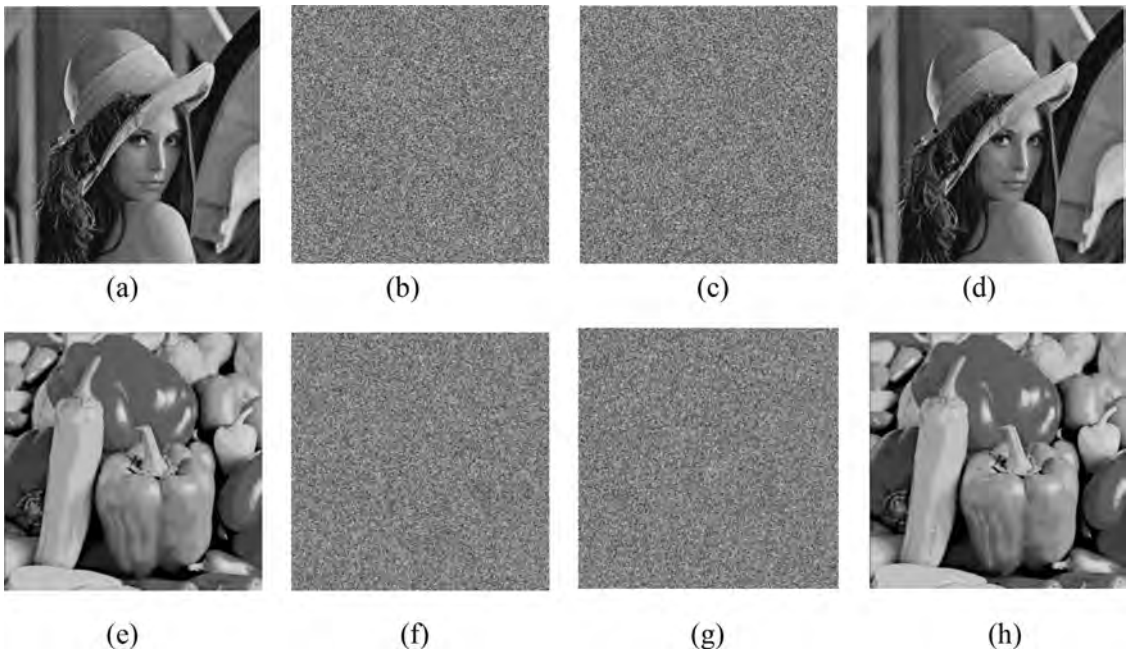

Fig. 7. Key sensitivity analysis. (a) The Lena image; (b) the encrypted image using the original key; (c) the decrypted image with an incorrect security key; and (d) the decrypted image with the correct security key. (e) The Peppers image; (f) the encrypted image using the original key; (g) the decrypted image with an incorrect security key; and (h) the decrypted image with the correct security key.

The original key is (0.2, 0.1, 0.75, −2), and the modified key is (0.2 + 10^(−13), 0.1, 0.75, −2). The original images are shown in Fig. 7(a) and Fig. 7(e), and the corresponding cipher images of the original key are shown in Fig. 7(b) and Fig. 7(f). The decrypted images for the incorrect decryption key are shown in Fig. 7(c) and Fig. 7(g), and the decrypted images for the correct decryption key is shown in Fig. 7(d) and Fig. 7(h). It is clear that the slightly different decryption key cannot decrypt the cipher-image. Therefore, the key sensitivity test shows that the proposed cryptosystem has perfect sensitivity to the key.

5.4. Differential analysis

To resist a differential attack, a good cryptosystem should ensure that any small modification in the plain-image should cause a significant difference in the cipher-image. The NPCR (number of pixels change rate) and UACI (unified average changing intensity) [37] are usually used for differential attack analysis. The ideal values of NPCR and UACI are 99.61% and 33.46%, respectively. These concepts are defined by Eqs. (15) and 16 below:

$$NPCR = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} D(i,j) \times 100\% \tag{15}$$

$$UACI = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100\% \tag{16}$$

where C_1 and C_2 are two cipher-images whose plaintext has only a different pixel, and D (i, j) is defined as:

$$D(i,j) = \begin{cases} 0 & C_1(i,j) = C_2(i,j) \\ 1 & C_1(i,j) \neq C_2(i,j) \end{cases} \tag{17}$$

NPCR and UACI for different images are presented in Table 9. Table 10 presents the results for a Lena (512×512) image when the pixel values at different positions have changed. From the two tables, we can see that UACI is greater than 0.33, and NPCR is greater than 0.99. Thus, the encryption scheme is highly sensitive to the change in the plain images, and two completely different cipher images may be gotten obtained despite a minimal change in the original images. Thus, our algorithm can effectively resist differential attacks.

5.5. Known-plaintext and chosen-plaintext attacks

In the encryption process, some methods are used to enhance the ability of the encryption scheme to resist known-plaintext and chosen-plaintext attacks. First, we use the SHA 256 hash function to compute the secret key of the encryption scheme, so our algorithm has high sensitivity to changes in the plain image. Second, in the diffusion phase, we diffuse the information of the original image to each pixel of the cipher image, which strengthens the correlation between the original image and the diffusion operation. In our algorithm, the permutation and diffusion of the image are strongly correlated with the original image. Therefore, the algorithm is highly sensitive to small changes in the

Table 9
NPCR and UACI for different images.

Images	Lena (256×256)	Brain (256×256)	Baboon (256×256)	Cman (256×256)	Girl (256×256)
UACI	0.3350	0.3358	0.3374	0.3367	0.3386
NPCR	0.9959	0.9963	0.9962	0.9963	0.9956

Table 10
The Lena (512×512) image for different positions.

Images	(1,1)	(10,33)	(203,155)
UACI	0.3344	0.3344	0.3351
NPCR	0.9962	0.9962	0.9961

original image. Therefore, the proposed algorithm could resist known-plaintext and chosen-plaintext attacks.

Some attackers always use all-white and all-black to make the permutation process of encryption methods invalid and then try to obtain some useful information. However, in our encryption scheme, the permutation and diffusion have strong correlations with the original image. It remains difficult for the attacker to crack the encryption algorithm using an all-white and all-black image method. All-white and all-black images are used as test images, and their cipher images and histograms of cipher images are illustrated in Fig. 8. Their entropies, correlation coefficients, NPCR, UACI and local entropy are provided in Table 11. The chi-square test is used to illustrate the uniform distribution of the histograms. Table 12 presents the chi-square test results. Table 12 demonstrates that all P-values are greater than 0.05 (significant level), so the pixel distribution is uniform. NPCR and UACI are used to prove that the original image and the encrypted image are two different images. The results demonstrate that the original image is different from the encrypted image. In general, the cipher images are noisy and different from the original images, and their histograms distribute uniformly. No useful information can be obtained from analysing the cipher images. Moreover, entropies and local entropies of the cipher images are greater than 7.90, and correlation coefficients in three directions are close to 0. These findings indicate that our algorithm has good encryption effect for all-white and all-black images and a high security level.

5.6. Computational complexity of the proposed scheme

The size of the plain image is denoted as $m \times n$. The time consumed by the proposed scheme is mainly divided into three parts. The first part

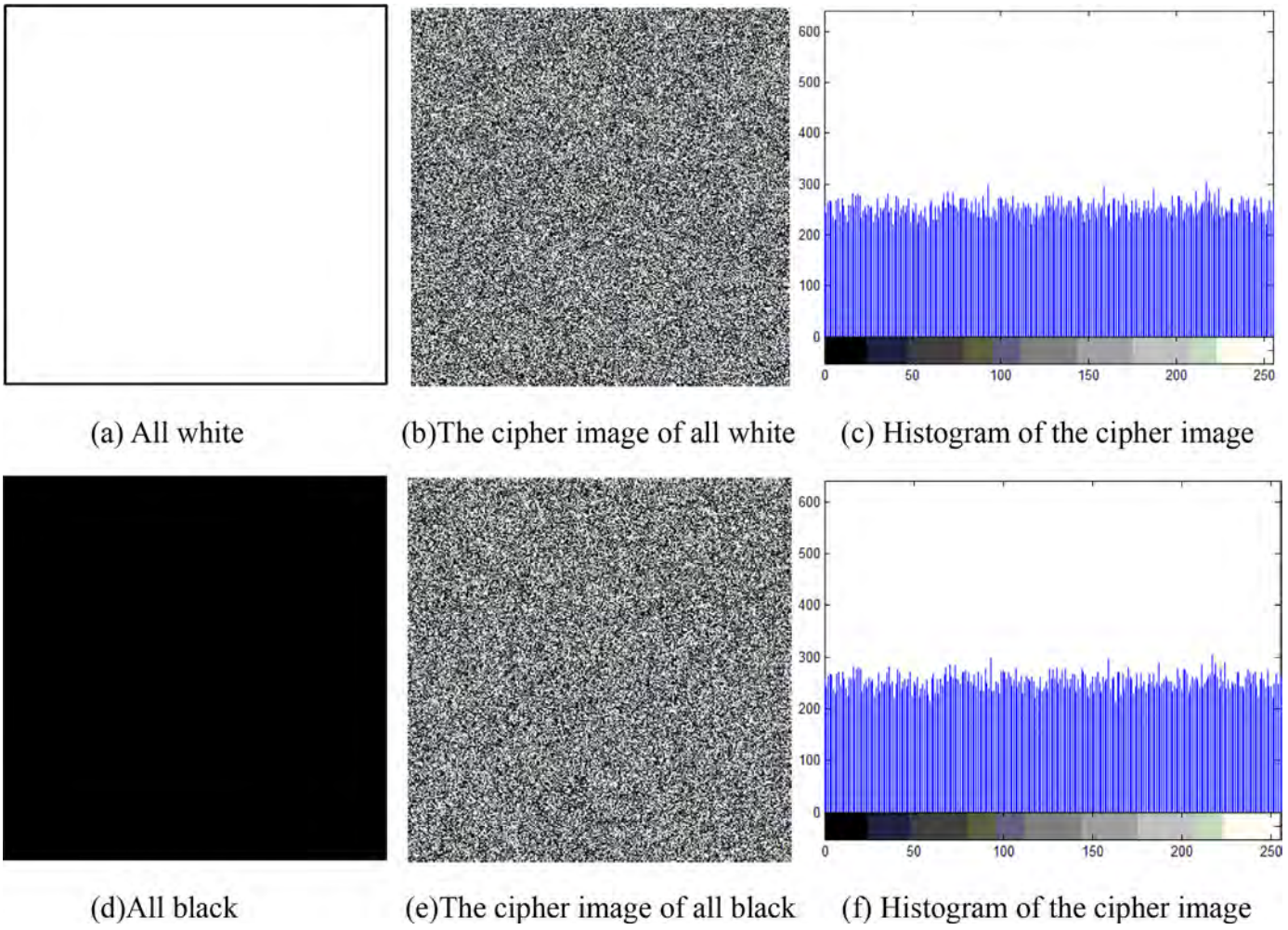

Fig. 8. Experimental results of all-white and black images. (a) All-white, (b) cipher image of the all-white image, (c) histogram of the cipher image, (d) all-black image, (e) cipher image of all-black image, and (f) histogram of the cipher image.

Table 11
The entropies, NPCR, UACI, local entropy and correlation coefficients of the plain, cipher images of all-white and all-black images.

Images	Entropies	NPCR	UACI	Local entropies	Correlation coefficients		
					Horizontal	Vertical	Diagonal
Cipher image of all white	7.9971	0.9959	0.3348	7.9053	0.0051	0.0026	0.0020
Cipher image of all black	7.9972	0.9960	0.3337	7.9062	0.0035	0.0060	0.0028

Table 12
Chi-square (χ^2) test analysis.

Images	Cipher image of all white	Cipher image of all black
χ^2	263.4922	249.8672
p-values	0.6559	0.4210

Table 13
Time complexity of different schemes.

Algorithm	Computation		
	Chaotic sequence	Confusion	Diffusion
Ref. [38]	$O(8m\times n) + O(m \times n)$	$O(4m\times n)$	$O(4m\times n)$
Ref. [39]	$O(8m\times n)$	$O(8m\times n\times\log(8m\times n))$	$O(m \times n)$
Ref. [40]	$O(m \times n)$	$O(m \times n\times\log(m \times n))$	$O(m \times n)$
Ref. [41]	$2O(4m\times n)$	$O(m \times n\times\log(m \times n))$ $+O(4m\times n\times\log(4m\times n))$	$O(4m\times n)$
Ref. [42]	$O(2m\times n)$	$O(2m\times n) + O(3m\times n)$	$O(4m\times n)$
Our scheme	$O(m \times n)$	$O(m \times n)$	$O(m \times n)$

Table 14
Performance of the proposed scheme and other methods.

	Correlation coefficient			Entropy
	Horizontal	Vertical	Diagonal	
Ref. [38]	−0.0230	0.0019	−0.0034	7.9974
Ref. [39]	0.0102	−0.0053	−0.0161	–
Ref. [40]	−0.0038	−0.0026	0.0017	–
Ref. [41]	0.0241	−0.0194	0.0243	7.9974
Ref. [42]	0.0000	−0.0011	0.0074	7.9973
Our scheme	0.0004	0.0013	−0.0023	7.9978

is the key streams generation. This part generates four key streams *X, Y, Z*, and *W*, and the lengths are all $m \times n$. The complexity of the generation algorithm is $O(m \times n)$. The second part is the permutation, which is the simplest part and has the same complexity of $O(m \times n)$. The third part is the diffusion part, which contains two DNA encoding steps and one XOR operation step, and its complexity is also $O(m \times n)$. The algorithm time complexity is arranged from low to high of $O(1)$, $O(\log(n))$, $O(n)$, $O(n\log(n))$, and $O(n^2)$. Here, $O(1)$ is the lowest time complexity, and $O(n^2)$ is the highest time complexity. Table 13 demonstrates that our encryption algorithm has low algorithm time complexity compared to the existing technology in Refs. [38,39,40,41,42]. In addition, algorithm space complexity is also an important criterion for measuring computational complexity. The Knuth–Durstenfeld algorithm is applied to image scrambling, which is an in-place scrambling algorithm, so its algorithm space complexity is $O(1)$. The algorithm space complexity $O(1)$ means that our encryption algorithm does not need to occupy extra memory resources for calculation during the scrambling stage, but many existing encryption schemes [12,14,15] need to occupy extra memory resources for calculation in the scrambling stage. The algorithm space complexity is greater than $O(1)$. In addition, Table 14 demonstrates that the encryption effect of the proposed encryption algorithm can achieve the encryption effect of the encryption algorithm in Refs. [38–42]. Therefore, the Knuth–Durstenfeld algorithm can effectively reduce the time complexity and the space complexity of the algorithm while ensuring the encryption effect, thereby improving the efficiency of the algorithm.

Table 15
Performance of the proposed scheme and other methods.

	Correlation coefficient			Entropy
	Horizontal	Vertical	Diagonal	
Ours	0.0004	0.0013	−0.0023	7.9978
Our scheme only with permutation	0.0031	0.0035	0.0025	–
Ref. [43]	0.0030	−0.0024	−0.0034	7.9976
Ref. [44]	−0.0098	−0.0050	−0.0013	7.9974
Ref. [45]	−0.0230	0.0019	−0.0034	7.9974
Ref. [46]	−0.0226	0.0041	0.0368	7.9973

5.7. Performance comparison

In Table 15, Lena (256×256) is as the test image, and the correlation coefficient and entropy of the cipher image generated from the proposed method and other methods are calculated and listed. In addition, to prove the permutation algorithm has good randomness, the correlation coefficient of the cipher image generated from the proposed method exclusively with permutation is also listed. The plain image and the image obtained after permutation process are also shown in Fig. 9(a, b). The correlation distribution of two adjacent pixels in three directions of the image obtained after permutation process is shown in Fig. 9(c-e). Table 15 clearly demonstrates that the horizontal correlation coefficient of image generated from proposed permutation method is less than the horizontal correlation coefficient of image generated from proposed method in Refs. [44,45,46] and close to that in Ref. [43], the vertical correlation coefficient of image generated from proposed permutation method is less than that in Refs. [44,46] and close to that in Refs. [43,45], and the diagonal correlation coefficient of image generated from proposed permutation method is less than that in Refs. [43,45,46] and close to that in Refs. [44]. These results demonstrate that our permutation algorithm has better ability to disrupt the correlation between adjacent pixels of an image. Table 15 demonstrates that the horizontal, vertical, and diagonal correlation coefficients of an image generated from the proposed permutation method are all greater than the horizontal, vertical, and diagonal correlation coefficients of images generated from the proposed method. These results demonstrate that the diffusion would reduce the correlation coefficient. According to the above analysis, the proposed permutation algorithm can still achieve the permutation effect of other encryption methods. In this paper, the Knuth–Durstenfeld algorithm is used for image permutation. Thus, the Knuth–Durstenfeld algorithm has good randomness. Additionally, regarding entropy, our result is greater than that reported in Refs. [43,44,45,46].3

6. Conclusion

This paper proposed a new image encryption scheme based on a hidden attractor chaos system, Knuth–Durstenfeld algorithm and DNA sequence operation. To overcome the common weaknesses of image encryption using self-excited attractor chaotic systems, the hidden attractor chaotic system is used to generate the chaotic sequences needed for image encryption. The NIST test of the chaotic sequence generated by the hidden attractor chaotic system proves that the hidden attractor chaotic system is suitable for image encryption. Because the Knuth–

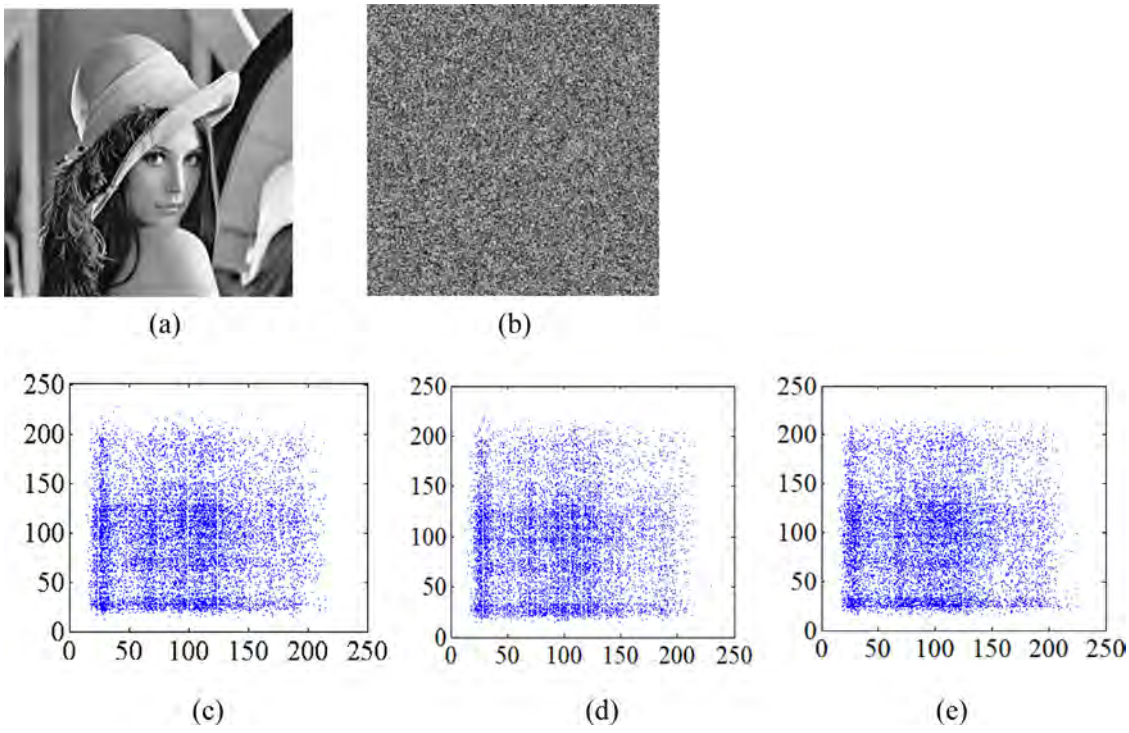

Fig. 9. (a) Plain image, (b) image obtained after the permutation process, (c) horizontal direction, (d) vertical direction, and (e) diagonal direction.

Durstenfeld algorithm has good randomness, this paper uses Knuth–Durstenfeld algorithms to better disrupt the correlation between adjacent pixels of the image. A DNA approach is used for diffusion operations. Experimental simulations and comparisons have also verified the security of the proposed scheme from four aspects: the exhaustive attack, the statistical attack, the differential attack and the known-plaintext and chosen-plaintext attacks. The scheme has a large key space and is extremely sensitive to its keys. Thus, it can resist exhaustive attack. The histogram of the scheme is uniform. The correlation coefficient is close to 0, and the entropy value is close to 8. Thus, the scheme can resist statistical attack. Both UACI and NPCR values approach their ideal values, which illustrates that the proposed scheme can resist differential attacks. The all-white and all-black image experiment also illustrates that the proposed scheme can resist the known-plaintext and chosen-plaintext attacks. All of the above findings demonstrate that the proposed scheme is efficient and practical in communications, but there are spaces to be explored and improved. For example, the proposed scheme is designed mainly for grey image, and colour image and multimedia data must first be converted to the same pattern of grey images and then encrypted with the scheme. In the future, we intend to convert the encryption scheme into the multimedia field.

Declaration of Competing Interest

The authors declared that they have no conflicts of interest to this work.

We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

CRediT authorship contribution statement

SiCheng Wang: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft, Writing - review & editing. **ChunHua Wang:** Supervision, Project administration. **Cong Xu:** Resources, Data curation, Visualization.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61971185) and the Open Fund Project of Key Laboratory in Hunan Universities (No. 18K010).

References

[1] FIPS PUB 46-3, data encryption standard(DES); 1999.
[2] FIPS PUB 197, advanced encryption standard; 2001.
[3] Fridrich J. Image encryption based on chaotic maps. IEEE International Conference on Systems. IEEE; 1997.
[4] Wu JH, Liao XF, Yang B. Color image encryption based on chaotic systems and elliptic curve elGamal scheme. Signal Process 2017;141:109–24.
[5] Pak C, Huang LL. A new color image encryption using combination of the 1D chaotic map. Signal Process 2017;138:129–37.
[6] Patidar V, Pareek NK, Sud KK. A new substitution–diffusion-based image cipher using chaotic standard and logistic maps. Commun Nonlinear Sci Numer 2009;14(7):3056–75.
[7] Gao TG, Chen ZQ. A new image encryption algorithm based on hyper-chaos. Phys Lett A 2008;372(4):394–400.
[8] Wang XY, Guo K. A new image alternate encryption algorithm based on chaotic map. Nonlinear Dyn 2014;76(4):1943–50.
[9] Zhou GM, Zhang DX, Liu YJ, et al. A novel image encryption algorithm based on chaos and line map. Neurocomputing 2015;169:150–7.
[10] Zhu HG, Zhang XD, Yu H, et al. An image encryption algorithm based on compound homogeneous hyper-chaotic system. Nonlinear Dyn 2017;89(1):61–79.
[11] Cheng GF, Wang CH, Chen H. A novel color image encryption algorithm based on hyperchaotic system and permutation-diffusion architecture. Int J Bifurcation Chaos 2019;29(9):1950115.
[12] Yin Q, Wang CH. A new chaotic image encryption scheme using Breadth-First search and dynamic diffusion. Int J Bifurcation Chaos 2018;28(4):1850047.
[13] Wu JH, Liao XF, Yang B. Image encryption using 2D hénon-sine map and dna approach. Signal Process 2018;153:11–23.
[14] Chai XL, Gan ZH, Yang K, et al. An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and DNA sequence operations. Signal Process Image Commun 2017;52:6–19.
[15] Ping P, Xu F, Wang ZJ. Image encryption based on non-affine and balanced cellular automata. Signal Process 2014;105(12):419–29.
[16] Hanis S, Amutha R. Double image compression and encryption scheme using logistic mapped convolution and cellular automata. Multimed Tools Appl 2017;77(06):6897–912.
[17] Short KM. Steps toward unmasking secure communications. Int J Bifurcation Chaos 2014;04(04):959–77.

[18] Fridrich J. Symmetric ciphers based on two-dimensional chaotic maps. Int J Bifurcation Chaos 1998;08(06):1259–84.
[19] Cang SJ, Li Y, Zhang RY, et al. Hidden and self-excited coexisting attractors in a lorenz-like system with two equilibrium points. Nonlinear Dyn 2019;95(01):381–90.
[20] Zhang S, Zeng YC, Li ZJ, et al. A novel simple no-equilibrium chaotic system with complex hidden dynamics. Int J Dyn Control 2018(23):1–12.
[21] Pham VT, Volos C, Jafari S, et al. Coexistence of hidden chaotic attractors in a novel no-equilibrium system. Nonlinear Dyn 2017;87(3):2001–10.
[22] Wei Z, Wang R, Liu A. A new finding of the existence of hidden hyperchaotic attractors with no equilibria. Math Comput Simul 2014;100:13–23.
[23] Danca M. Hidden chaotic attractors in fractional-order systems. Nonlinear Dyn 2018;89(18):1–10.
[24] Pham VT, Jafari S, Kapitaniak T. Constructing a chaotic system with an infinite nNumber of equilibrium points. Int J Bifurcation Chaos 2016;26(13):1350052–1-28.
[25] Zhou L, Wang CH, Zhou LL. A novel no-equilibrium hyperchaotic multi-wing system via introducing memristor. Int J Circuit Theory Appl 2018;46(1):84–98.
[26] Zhang X, Wang CH. Multiscroll hyperchaotic system with hidden attractors and its circuit implementation. Int J Bifurcation Chaos 2019;29(9):1950117.
[27] Cavusoglu U, Panahi S, Akgul A, et al. A new chaotic system with hidden attractor and its engineering applications: analog circuit realization and image encryption. Analog Integr Circuits Process 2019;98(1):85–99.
[28] Zhou NR, Hua TX, Gong LH, et al. Quantum image encryption based on generalized arnold transform and double random-phase encoding. Quantum Inf Process 2015;14(4):1193–213.
[29] Liu ZJ, Li S, Liu W, et al. Opto-digital image encryption by using baker mapping and 1-D fractional Fourier transform. Opt Lasers Eng 2013;3(51):224–9.
[30] Kumar M, Iqbal A, Kumar P. A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie–Hellman cryptography. Signal Process 2016;125:187–202.
[31] Guvenoglu E, Tuysuz Mehmet Ali Aksoy. An improvement for Knutt/Durstenfeld algorithm based image encryption. 23nd Signal Processing and Communications Applications Conference; 2015.
[32] Cisse II, Kim H, Ha T. A rule of seven in Watson–Crick base pairing of mismatched sequences. Nat Struct Mol Biol 2012;19(6):623–7.
[33] Alvarez G, Li SJ. Some basic cryptographic requirements for chaos-based cryptosystems. Int J Bifurcation Chaos 2006;16(08):2129–51.
[34] Chai XL, Fu XL, Gan ZH, et al. A color image cryptosystem based on dynamic DNA encryption and chaos. Signal Process 2019;155:44–62.
[35] Wang MX, Wang XY, Zhang YQ, et al. A novel chaotic encryption scheme based on image segmentation and multiple diffusion models. Opt Laser Technol 2018;108:558–73.
[36] Chai XL, Gan ZH, Yuan K, et al. A novel image encryption scheme based on dna sequence operations and chaotic systems. Neural Comput Appl 2019;31(1):219–37.
[37] Mao YB, Chen GR, Lian SG. A novel fast image encryption scheme based on 3D chaotic baker maps. Int J Bifurcation Chaos 2011;14(10):3613–24.
[38] l Xu, Z Li, Li J, et al. A novel bit-level image encryption algorithm based on chaotic maps. Opt Lasers Eng 2012;78(21):17–25.
[39] Zhou YC, Cao WJ, Chen CLP. Image encryption using binary bitplane. Signal Process 2014;100(7):197–207.
[40] Pak C, Huang L. A new color image encryption using combination of the 1d chaotic map. Signal Process 2017;138:129–37.
[41] Teng L, Wang X. A bit-level image encryption algorithm based on spatiotemporal chaotic system and self-adaptive. Opt Commun 2012;285(20):4048–54.
[42] Liu DD, Zhang W, Yu H, et al. An image encryption scheme using self-adaptive selective permutation and inter-intra-block feedback diffusion. Signal Process 2018;151:130–43.
[43] Liu WH, Sun KH, Zhu CX. A fast image encryption algorithm based on chaotic map. Opt Lasers Eng 2016;84:26–36.
[44] Wang XY, Zhang HL. A color image encryption with heterogeneous bit-permutation and correlated chaos. Opt Commun 2015;342:51–60.
[45] Xu L, Li Z, Li J, et al. A novel bit-level image encryption algorithm based on chaotic maps. Opt Lasers Eng 2016;78:17–25.
[46] Xu L, Gou X, Li Z, et al. A novel chaotic image encryption algorithm using block scrambling and dynamic index-based diffusion. Opt Lasers Eng 2017;91:41–52.