A Novel Chaos-Based Image Encryption Scheme by Using Randomly DNA Encode and Plaintext Related Permutation

Zhen Li 1,2*, Changgen Peng 1*, Weijie Tan 1 and Liangrong Li 2

1 College of Computer Science and Technology, State Key Laboratory of Public Big Data, Guizhou University, Guiyang 550025, China; zli6@gzu.edu.cn (Z.L.); wjtan@gzu.edu.cn (W.T.)
2 College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China; lrl@gzu.edu.cn
* Correspondence: cgpeng@gzu.edu.cn

Received: 30 September 2020; Accepted: 21 October 2020; Published: 23 October 2020

Abstract: To ensure the security and privacy of digital image when its transmitting online or storing in the cloud, we proposed a novel chaos based image encryption scheme by using randomly DNA encode and plaintext related permutation. In our scheme, we first randomly encode plain image into a nucleotide sequence under the control by the piecewise linear chaotic map (PWLCM). After that, the plaintext related permutation would be done under the control sequence which generated by hyper chaotic Lorenz system (HCLS). Next, we make diffusion processing with key DNA sequence which is generated by another PWLCM system and also encoded randomly. Finally, we decode DNA sequence into cipher image matrix. In addition, we used many common security analysis methods to test our scheme, and the result compared with other works. The tests and comparison results are shown that our proposed image cryptosystem has excellent security performance to ensure the digital image security on communication.

Keywords: image encryption; DNA computing; plaintext related; DNA randomly encode

1. Introduction

With the popularity of smart phones, digital images are becoming one of the most important data formats in our daily life, and it is easier to get than in the old days. However, security issues arise when these images need to be transferred over the internet or stored on the cloud, especially some of them contain privacy. Moreover, digital images are also used in many sensitive areas such as medical, national defense and legal affairs. Therefore, the security of digital images is becoming an eye-catching issue for many researchers [1–3].

As we all know, encryption is the most easy way to protect information security, and there are many excellent algorithms for encrypting text structure data which play important roles for protecting data security in network communication, e.g., DES, AES, RSA, etc. Unfortunately, those algorithms weakly adapt to the image data characteristics, such as high redundancy, strong correlation between pixels, etc. As a result, text structure encryption algorithms are not good at image encryption, and may cause some problems such as low efficiency, contour information leakage, etc. [4,5].

Chaos systems which were first proposed by Lorenz [6] have many good characteristics, e.g., sensitivity of initial values and parameters, etc., are very suitable for designing an image cryptosystem. The first chaotic cryptosystem is proposed by Matthews [7] who is a British scientist. After that, the chaotic system was beginning to be used to design image encryption systems. Usually, chaotic image encryption schemes use chaotic systems to generate key stream or some scrambling control sequences or nonlinear components, and then do permutation and diffusion to plain image to
generate cipher image. In recent years, there are many image encryption schemes based on chaotic systems proposed to the public [8–13].

DNA computing was first proposed by Adleman [14] in 1994. After then, DNA computing had found some good characteristics such as ultra-high information density, great parallelism and ultra-low energy consumption. Gehani et al. [15] was first proposed the DNA computing in designing image cryptosystem. After Gehani, many image encryption schemes based on DNA computing appeared in public [16–21]. Zhang et al. [22] found that if the algorithm adopts binary coding, the coding efficiency is low, and it would be easy cracked when cryptosystem is not related to plain image. Xie et al. [23] found that an image encryption system based on DNA computing must contain diffusion part, if only scramble processing would not be security. Liu et al. [24] cracked an DNA computing based encryption system, and found that image cryptosystem must be able to resist differential attack, otherwise can reconstruct the equivalent key by part of known plaintext attack.

Recently, some interesting works on image encryption by using DNA computing. Wang et al. [16] proposed an image encryption scheme based on coupled map lattices and DNA encoding. In this scheme, rows and columns cyclic shift operation is used to scramble the image DNA matrix. The diffusion performance is good, however, the sensitivity of plaintext seems a little weak. Zefreh [20] proposed an image encryption scheme using DNA computing and hash function. This scheme related plaintext by hash function to make image cryptosystem has enough sensitivity to resist differential attack. However, the hash value of plain image also needs to transfer to the decryption part, it is harmful for key management. Babaei et al. [21] proposed a DNA computing based image encryption scheme using cellular automata. This scheme is cleverly designed but real security key only the initial value of logistic map, that it is hard to resist brute force attack.

To overcome the shortage mentioned above, we proposed a novel chaos-based image encryption scheme by using randomly DNA encode and plaintext related permutation. The main contribution of this paper is that we designed a reversible plaintext related permutation algorithm that the information of plain image also can directly calculated in the decryption part without transmitting from the encryption part. In addition, the reversible plaintext related permutation makes our image cryptosystem have enough sensitivity of plain image to against differential attack. Furthermore, chaos controlled randomly DNA encode and decode could make better diffusion performance. The detailed contributions of this paper are as follows: (1) A novel reversible plaintext related permutation method was given, (2) A novel image encryption scheme using randomly DNA encode and plaintext related permutation was given, (3) The detailed security analysis of our scheme was given.

The organization of this paper is as follows: In Section 2, the basic knowledge of chaotic system and DNA calculation used in this paper is given. In Section 3, the details of our proposed image encryption scheme are given. In Section 4, the simulation and some common security analysis of our scheme are given. In Section 5, the conclusion of this paper is given.

2. Preliminary

2.1. Chaotic Systems

In our scheme, chaotic systems are used to generate the control sequences for DNA encoding, permutation, diffusion, etc. There are two types of chaotic systems used in our image cryptosystem: the piecewise linear chaotic map (PWLCM) and the hyper chaotic Lorenz system (HCLS). PWLCM is used in controlling the DNA random encode process and generating key sequence. HCLS is used in controlling the permutation process.

The PWLCM [25] is given by

\[
F(x) = \begin{cases} 
\frac{x}{p} & x \in [0, p); \\
\frac{x-p}{0.5-p} & x \in (p, 0.5]; \\
F(1-x, p), & x \in (0.5, 1].
\end{cases}
\] (1)
where $p \in (0, 0.5)$ is a parameter of PWLCM.

The HCLS [26] is given by

\[
\begin{align*}
\dot{x} &= a(y - x) + w \\
\dot{y} &= cx - y - xz \\
\dot{z} &= xy - bz \\
\dot{w} &= -yz + \gamma w
\end{align*}
\]

where $a$, $b$, $c$ and $\gamma$ are parameters of this hyper chaotic system. When $a = 10$, $b = 8/3$, $c = 28$ and $\gamma \in [-1.52, -0.06]$, the system is chaotic and its attractors are shown in Figure 1.

![Figure 1. Attractors of hyper chaotic Lorenz system. (a) x–y plane, (b) x–z plane, (c) x–w plane, (d) z–w plane, (e) y–w plane and (f) y–z plane.](image)

2.2. DNA Operations

There are 4 kinds of nucleic acid bases in a DNA sequence: adenine (A), guanine (G), cytosine (C) and thymine (T). In the rules of Watson–Crick base pairing, adenine and thymine are a complementary pair, cytosine and guanine are a complementary pair. Thus, there are only 8 kinds of encoding modes satisfied with this complementary rule, the details are shown in Table 1. Every two bits can be encoded into one of the nucleic acid bases. For example, if we select rule 1 which is shown in Table 1, then ‘00’, ‘11’, ‘10’ and ‘01’ can be encoded into ‘A’, ‘T’, ‘C’ and ‘G’, respectively. Therefore, the $M \times N$ 8-bits gray image can be encoded as a 4MN length nucleotide string. There are three types of operations of DNA computing, addition operation rules are given in Table 2, the subtraction operation rules can be seen in Table 3 and the XOR operation rules are shown in Table 4.
Table 1. DNA encode/decode rules.

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

Table 2. Addition operation of DNA code.

|   | A  | C  | G  | T  |
|---|----|----|----|----|
| + A | T  | A  | C  | G  |
| C  | A  | C  | G  | T  |
| G  | C  | G  | T  | A  |
| T  | G  | T  | A  | C  |

Table 3. Subtraction operation of DNA code.

|   | A  | C  | G  | T  |
|---|----|----|----|----|
| − A | C  | A  | T  | G  |
| C  | G  | C  | A  | T  |
| G  | T  | G  | C  | A  |
| T  | A  | T  | G  | C  |

Table 4. XOR operation of DNA code.

| XOR | A  | C  | G  | T  |
|-----|----|----|----|----|
| A   | A  | C  | G  | T  |
| C   | C  | A  | T  | G  |
| G   | G  | T  | A  | C  |
| T   | T  | G  | C  | A  |

3. Our Proposed Scheme

3.1. Encryption Scheme

The block diagram of the encryption scheme is shown in Figure 2, the detailed processes are as follows:

**Step 1:** Input an 8-bit gray level plain image \( P \) with size of \( M \times N \), and an initial key \( K \) into the encryption scheme.

**Step 2:** Input initial key \( K \) into Algorithm 1 to generate a rules selection control sequence \( RSCS \).

**Step 3:** Change the plain image matrix \( P \) into an array \( PA \) by rows.

**Step 4:** Input image array \( PA \) and rules selection control sequence \( RSCS \) into Algorithm 2 to randomly encode the image array \( PA \) under the DNA encode rules which are shown in Table 1, and generate an image DNA sequence \( IDS \).

**Step 5:** Input image DNA sequence \( IDS \) and initial key \( K \) into Algorithm 3 to generate permutation control sequence \( PCS \).

**Step 6:** Input image DNA sequence \( IDS \) and permutation control sequence \( PCS \) into Algorithm 4 to do plaintext related permutation, and get a permutated image sequence \( PIS \).

**Step 7:** Put initial key \( K \) and rules selection control sequence \( RSCS \) into Algorithm 5 to generate a key DNA sequence \( KDS \).
Step 8: Input permuted image sequence PIS and key DNA sequence KDS into Algorithm 6 to do DNA diffusion operation, and generate a diffused sequence DS.

Step 9: Put diffused sequence DS and rules selection control sequence RSCS into Algorithm 7 to decode DNA sequence to a recovered image array RIA.

Step 10: Reshape recovered image array RIA as M × N to a cipher image matrix C.

Step 11: Output cipher image C and the encryption process is finished.

Algorithm 1 Generation algorithm of rules selection control sequence (RSCS).

**INPUT** initial key K

1. \( \text{KEYHASH} \leftarrow \text{SHA256}(K) \)
2. \( A1 \leftarrow \text{KEYHASH}(1:64) \)
3. \( A2 \leftarrow \text{KEYHASH}(65:128) \)
4. \( x \leftarrow \text{mod}(A1/10^{15}, 1) \quad p \leftarrow \text{mod}(A2/10^{15}, 0.5) \)
5. Put \( x, p \) into Equation (1) to generate a sequence \( X \) by iterating.

\[\text{RSCS} = \text{mod}(\text{floor}(X \times 10^{15}), 8)\]

**OUTPUT** rules selection control sequence RSCS.

Algorithm 2 DNA random encode algorithm.

**INPUT** rules selection control sequence RSCS and data array A

1. Build a DNA encode rules table RT (see Table 1).
2. Change data array A to a bit array AB
3. Set len ← Length(AB); n ← 1
4. Set DS ← NULL
5. For \( i = 1 \) to \( len - 1 \) step 2 do
   1. \( DS(n) \leftarrow RT(AB(i, i + 1), \text{RSCS}(n)) \)
   2. \( n \leftarrow n + 1 \)
6. **OUTPUT** DNA sequence DS.

Algorithm 3 Generation algorithm of permutation control sequence (PCS).

**INPUT** image DNA sequence IDS and initial key K

1. Build a DNA addition operation table AT (see Table 2), a DNA XOR operation table XT (see Table 4).
2. Set len ← length(IDS); \( T1 \leftarrow IDS(1); T2 \leftarrow IDS(1) \)
3. For \( i = 2 \) to \( len \) do
   1. \( T1 \leftarrow AT(T1, IDS(i)) \)
   2. \( T2 \leftarrow XT(T2, IDS(i)) \)
4. Set \( \text{DSHASH} \leftarrow \text{SHA256}([T1, T2]); \text{KEYHASH} \leftarrow \text{SHA256}(K) \)
5. Set \( T5 \leftarrow \text{DSHASH} \oplus \text{KEYHASH} \)
6. Set \( A1 \leftarrow T5(1:64); A2 \leftarrow T5(65:128); A3 \leftarrow T5(129:194); A4 \leftarrow T5(195:256); \)
7. Set \( x0 \leftarrow \text{mod}(\text{fix}(A1/10^8), 80) - 40 + (A1/10^{14} - \text{fix}(A1/10^{14})) \)
8. Set \( y0 \leftarrow \text{mod}(\text{fix}(A2/10^8), 80) - 40 + (A2/10^{14} - \text{fix}(A2/10^{14})) \)
9. Set \( z0 \leftarrow \text{mod}(\text{fix}(A3/10^8), 80) + 1 + (A3/10^{14} - \text{fix}(A3/10^{14})) \)
10. Set \( w0 \leftarrow \text{mod}(\text{fix}(A4/10^8), 500) - 250 + (A4/10^{14} - \text{fix}(A4/10^{14})) \)
11. Put \( x0, y0, z0, w0 \) into Equation (2) to generate a sequence \( X \) by iterating.
12. Set \( PCS = \text{mod}(\text{floor}(X \times 10^{15}), len) \)
13. Remove the repeated elements from PCS, and then put the absent numbers at the end.

**OUTPUT** permutation control sequence PCS.
Algorithm 4 Permutation/Reverse permutation algorithm.

**INPUT** image DNA sequence \( IDS \) and permutation control sequence \( PCS \).

\[
\text{len} \leftarrow \text{length}(IDS)
\]

for \( i = 1 \) to \( \text{fix}(\text{len}/2) \) do

\[
IDS(PCS(i)) \leftrightarrow IDS(PCS(\text{len} - i + 1))
\]

end for

\[
PIS \leftarrow IDS
\]

**OUTPUT** permutated image sequence \( PIS \).

Algorithm 5 Generation algorithm of key DNA sequence (KDS).

**INPUT** initial key \( K \) and rules selection control sequence \( RSCS \).

\[
\text{KEYHASH} \leftarrow \text{SHA256}(K)
\]

\[
A1 \leftarrow \text{KEYHASH}(129:194)
\]

\[
A2 \leftarrow \text{KEYHASH}(195:256)
\]

\[
x = \text{mod}(A1/10^{15}, 1) \quad p = \text{mod}(A2/10^{15}, 0.5)
\]

Put \( x, p \) into Equation (1) to generate a sequence \( X \) by iterating.

\[
\text{DSA} = \text{mod}(\text{floor}(X \times 10^{15}), 256)
\]

Input \( RSCS \) and \( DSA \) into Algorithm 2, then output DNA sequence \( KDS \).

\[
\text{OUTPUT} \text{ key DNA sequence } KDS
\]

Algorithm 6 DNA diffusion algorithm.

**INPUT** permutated image sequence \( PIS \) and key DNA sequence \( KDS \).

Build a DNA addition operation table \( AT \) (see Table 2) and a DNA XOR operation table \( XT \) (see Table 4).

\[
\text{len} \leftarrow \text{length}(PIS)
\]

\[
T(1) \leftarrow AT(PIS(1), KDS(1));
\]

\[
T(1) \leftarrow XT(T(1), KDS(1));
\]

for \( i = 2 \) to \( \text{len} \) do

if \( \text{mod}(i, 2) = 0 \) then

\[
T(i) \leftarrow XT(PIS(i), KDS(i));
\]

\[
T(i) \leftarrow XT(T(i), T(i - 1));
\]

else

\[
T(i) \leftarrow AT(PIS(i), KDS(i));
\]

\[
T(i) \leftarrow XT(T(i), T(i - 1));
\]

end if

end for

\[
\text{DS} \leftarrow T
\]

**OUTPUT** diffused sequence \( DS \).

Algorithm 7 DNA decode algorithm.

**INPUT** DNA sequence \( DS \) and rules selection control sequence \( RSCS \).

Build a DNA encode rules table \( RT \) (see Table 1).

\[
\text{len} \leftarrow \text{Length}(DS); n \leftarrow 1
\]

\[
\text{NS} \leftarrow \text{NULL}
\]

for \( l = 1 \) to \( \text{len} \) do

\[
\text{NS}(n, n + 1) \leftarrow RT(DS(i), RSCS(i))
\]

\[
n \leftarrow n + 2
\]

end for

Change bit sequence \( NS \) to a numerical array \( NA \) by each 8 bits.

**OUTPUT** numerical array \( NA \).
Figure 2. The block diagram of the encryption scheme.

3.2. Decryption Scheme

The block diagram of decryption scheme is shown in Figure 3, the detailed processes are as follows:

**Step 1:** Input a cipher image $C$, and an initial key $K$ into the decryption scheme.

**Step 2:** Put initial key $K$ into Algorithm 1 to generate a rules selection control sequence $RSCS$.

**Step 3:** Change the plain image matrix $C$ into an array $CA$ by rows.

**Step 4:** Input image array $CA$ and rules selection control sequence $RSCS$ into Algorithm 2 to randomly encode the image array $CA$ under the DNA encode rules which are shown in Table 1, and generate a cipher image DNA sequence $CIDS$.

**Step 5:** Put initial key $K$ into Algorithm 5 to generate a key DNA sequence $KDS$.

**Step 6:** Input cipher image DNA sequence $CIDS$ and key DNA sequence $KDS$ into Algorithm 8 to do reverse DNA diffusion operation, and generate a reverse diffused sequence $RDS$.

**Algorithm 8** Reverse DNA diffusion algorithm.

**INPUT** cipher image DNA sequence $CIDS$ and key DNA sequence $KDS$.

Build a DNA substraction operation table $ST$ (see Table 3) and a DNA XOR operation table $XT$ (see Table 4).

$len \leftarrow \text{length}(CIDS)$

**for** $i = len$ **to** $2$ **do**

if $mod(i, 2) = 0$ then

$T(i) \leftarrow XT(CIDS(i), KDS(i))$;

$T(i) \leftarrow XT(T(i), CIDS(i - 1))$;

else

$T(i) \leftarrow XT(CIDS(i), CIDS(i - 1))$;

$T(i) \leftarrow ST(T(i), KDS(i - 1))$;

**end if**

**end for**

$T(1) \leftarrow XT(CIDS(1), KDS(1))$;

$T(1) \leftarrow ST(T(1), KDS(1))$;

$RDS \leftarrow T$

**OUTPUT** reverse diffused sequence $RDS$.

**Step 7:** Input reverse diffused sequence $RDS$ and initial key $K$ into Algorithm 3 to generate permutation control sequence $PCS$.

**Step 8:** Input reverse diffused sequence $RDS$ and permutation control sequence $PCS$ into Algorithm 4 to do reverse permutation, and get a reverse permutated image sequence $RPIS$.

**Step 9:** Put reverse permutated image sequence $RPIS$ and rules selection control sequence $RSCS$ into Algorithm 7 to decode DNA sequence to a recovered image array $RIA$.  

---

**Figure 2.** The block diagram of the encryption scheme.
**Step 10:** Reshape recovered image array RIA as $M \times N$ to a cipher image matrix $P$.

**Step 11:** Output decrypted image $P$ and the decryption process is finished.

**Figure 3.** The block diagram of the decryption scheme.

### 4. Simulation and Security Analysis

In this section, we evaluate our proposed scheme by software simulation, and all of the simulation and security tests are implemented in MATLAB R2016a. Our tests were worked on a personal computer with Inter(R) Core i5-8500 CPU 3.00 GHz, 8 GB memory and the software was run on the operation system of Window 10 home edition. For simulation, we select initial key as ‘FE02DE31AB456ACFB223013CADAB58902342FEFD’ in hexadecimal, and parameters of hyper chaotic system which, given in Equation (2), are $a = 10$, $b = 8/3$, $c = 28$ and $\gamma = -1$. The images for testing are 512 $\times$ 512 pixels with 8-bit gray level. The simulation results are shown in Figure 4. The plain images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’ are shown in Figure 4a–e, the corresponding cipher images are shown in Figure 4f–j and the recovery images from decryption are shown in Figure 4k–o. In the following subsections, we provide several commonly used security test result to prove our proposed image encryption scheme has an excellent security performance.

#### 4.1. Key Space Analysis

Key space is an important security index to evaluate the cryptosystem on whether it has enough capability to resist a brute-force attack. In our scheme, we use two PWLCM systems for generating DNA random encode rules selection sequence and key stream, and use the HCLS system for generating the permutation control sequence. The true keys of our cryptosystem are two parameters $p \in (0,0.5)$ and two initial values $x \in (0,1)$ of two PWLCM systems, and four initial values $x_0 \in (-40,40)$, $y_0 \in (1,81)$, $w_0 \in (-250,250)$ of the HCLS system. We suppose the change step of each initial value and parameter is $10^{-15}$, the key space of our proposed cryptosystem can be calculated as $S = (0.5 \times 10^{15})^2 \times (1 \times 10^{15})^2 \times (80 \times 10^{15})^3 \times (500 \times 10^{15}) = 6.4 \times 10^{127} \approx 2^{424}$. The key space is required to be at least $2^{100}$ to protect from a brute-force attack [16], and our key space is large enough for security.

#### 4.2. Differential Attack

The sensitivity of plain image is a very important index to an image cryptosystem, and it can help cryptosystem to resist the differential attack which is a kind of chosen-plaintext attack. The differential attack usually changes the value of one pixel in the plain image, and compares the difference between two corresponding cipher images. There are two features to evaluate the sensitivity of the plain image: number of pixels change rate (NPCR) and unified average changing intensity (UACI) [8,27]. The NPCR and UACI are calculated by Equations (3) and (4), respectively.
Figure 4. Encryption simulation. (a–e) The plain images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’; (f–j) the corresponding cipher images; (k–o) the recovery images from the decryption.
\[ \text{NPCR} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} D(i,j)}{M \times N} \times 100\% , \]

\[ D(i,j) = \begin{cases} 
0, & C_1(i,j) = C_2(i,j) \\
\frac{1}{2}, & C_1(i,j) \neq C_2(i,j) 
\end{cases} \]

\[ \text{UACI} = \frac{1}{M \times N} \left( \sum_{i=1}^{M} \sum_{j=1}^{N} \left| C_1(i,j) - C_2(i,j) \right| \right) \times 100\% , \]

where \( C_1(i,j) \) and \( C_2(i,j) \) are the cipher images which correspond two one-pixel different images. \( M \) and \( N \) are the height and width of those images, respectively. The NPCR calculates the percentage of pixels which are different values at the same position between \( C_1(i,j) \) and \( C_2(i,j) \). The UACI is a quantitative assessment of the average distance between two pixels at the same position in \( C_1(i,j) \) and \( C_2(i,j) \).

To evaluate the NPCR and UACI results, the critical value is given by Wu et al. \cite{28}. For a significance level \( \alpha \), a critical value of NPCR is given by:

\[ N_{\alpha}^* = \frac{Q - \Phi^{-1}(\alpha)\sqrt{Q/H}}{Q + 1} , \]

where \( H \) represents the total number of pixels in an image, \( Q \) represents the largest allowed pixel value in the image. An image encryption scheme can be considered to pass the NPCR if the obtained NPCR is larger than \( N_{\alpha}^* \).

The critical interval \( (U_{\alpha}^-, U_{\alpha}^+) \) of UACI is given by:

\[ \begin{align*}
U_{\alpha}^- &= \mu_u - \Phi^{-1} \left( \frac{\alpha}{2} \right) \sigma_u , \\
U_{\alpha}^+ &= \mu_u + \Phi^{-1} \left( \frac{\alpha}{2} \right) \sigma_u ,
\end{align*} \]

where

\[ \mu_u = \frac{Q + 2}{3Q + 3} , \]

and

\[ \sigma_u = \sqrt{\frac{(Q + 2)(Q^2 + 2Q + 3)}{18(Q + 1)^2QH}} . \]

If the UACI value falls into interval \( (U_{\alpha}^-, U_{\alpha}^+) \), it means that the two test images have much difference. When the size of test image is 512 \( \times \) 512 and the significance level \( \alpha = 0.05 \), the critical value of NPCR is \( N_{0.05}^* = 99.5893\% \) and the critical interval of UACI is \( (U_{0.05}^-, U_{0.05}^+) = (33.3730\%, 33.5541\%) \).

For this test, we randomly changed one pixels’ value from plain images, and calculated the NPCR and UACI. We test NPCR and UACI 100 times, and the test results are shown in Table 5. The expectation of NPCR and UACI between two random images are given in Equations (9) and (10), respectively.

\[ \text{NPCR}_E = \left( 1 - \frac{1}{2^{8g_2L}} \right) \times 100\% , \]

\[ \text{UACI}_E = \frac{1}{L^2} \left( \frac{\sum_{i=1}^{L} \sum_{j=1}^{L} x(i+1) - x(i-1)}{L(L-1)} \right) \times 100\% , \]

where \( L \) is the gray level of test images. If the test images are 8-bit gray levels, the expectation of NPCR is 99.6094\%, and the expectation of UACI is 33.4635\%. In Table 5, the average of NPCR is higher than 99.6094\% and the average of UACI is close to 33.4635\%. The test result means that our proposed image cryptosystem is secure enough to resist the differential attack.
Table 5. The results of number of pixels change rate (NPCR) and unified average changing intensity (UACI).

| Image   | NPCR(%) Max | NPCR(%) Min | NPCR(%) Average | UACI(%) 99.5893% Max | UACI(%) 99.5893% Min | UACI(%) 99.5893% Average | UACI(%) 0.05% = 33.3730% | UACI(%) 0.05% = 33.5541% |
|---------|-------------|-------------|-----------------|-----------------------|-----------------------|---------------------------|---------------------------|---------------------------|
| Lena    | 99.6437     | 99.5731     | 99.6159         | Pass                  | 33.5589               | 33.4246                   | 33.4846                   | Pass                      |
| Baboon  | 99.6407     | 99.5816     | 99.6168         | Pass                  | 33.6188               | 33.4176                   | 33.4808                   | Pass                      |
| Pepper  | 99.6453     | 99.5777     | 99.6217         | Pass                  | 33.5315               | 33.4212                   | 33.4963                   | Pass                      |
| Airplane| 99.6425     | 99.5926     | 99.6138         | Pass                  | 33.5369               | 33.4102                   | 33.4725                   | Pass                      |
| Lake    | 99.6488     | 99.5853     | 99.6170         | Pass                  | 33.5589               | 33.4246                   | 33.4846                   | Pass                      |

4.3. Statistical Analysis

4.3.1. Histogram Analysis

Statistical indiscernibility is one of the important indexes to measure the quality of a cipher image. A good cipher image, which can resist statistical analysis, should have a uniform histogram at each gray level. The histogram analysis results are shown in Figure 5. The test plain images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’ are shown in Figure 5a–e, and the corresponding histograms in Figure 5f–j. The corresponding ciphers are in Figure 5k–o, and the corresponding histograms of these cipher images are in Figure 5p–t. We use chi-squared test to evaluate the uniformity of encrypted image’s histogram. When we set the significance level $\alpha = 0.05$, chi-squared test results of cipher images are given in Table 6. According to our histogram analysis results, our proposed cryptosystem has good diffused property to resist the statistical attack.

4.3.2. Correlation Coefficient

Correlation coefficient analysis is a measurement of the correlation among the adjacent pixels in the image. The encryption process should break the correlation of adjacent pixels; therefore, the less correlation among adjacent pixels in the cipher image, the better the security.

The correlation coefficient is given in Equation (11).

$$ r_{ab} = \frac{\text{cov}(a,b)}{\sqrt{D(a)D(b)}}, \quad (11) $$

where $a$ and $b$ are two gray values of adjacent pixels, and

$$ E(a) = \frac{1}{N} \sum_{i=1}^{N} a_i, \quad (12) $$

$$ D(a) = \frac{1}{N} \sum_{i=1}^{N} (a_i - E(a))^2, \quad (13) $$

$$ \text{cov}(a,b) = \frac{1}{N} \sum_{i=1}^{N} (a_i - E(a))(b_i - E(b)). \quad (14) $$
Table 6. Histogram uniformity evaluation by chi-squared test.

| Image     | p-Value | Decision (H = 0 or H = 1) |
|-----------|---------|---------------------------|
| Lena      | 0.25502 | Pass                      |
| Baboon    | 0.7198  | Pass                      |
| Pepper    | 0.97584 | Pass                      |
| Airplane  | 0.67795 | Pass                      |
| Lake      | 0.15236 | Pass                      |

Figure 5. Histograms. (a–e) The plain images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’; (f–j) the corresponding histograms of these plain images; (k–o) the corresponding cipher images by encryption; (p–t) the corresponding histograms of these cipher images.

For this test, 10,000 pairs of adjacent pixels have been randomly selected from plain images and corresponding cipher images. The test results are shown in Table 7. The correlation distributions are shown in Figure 6 and the rows (a)–(e) corresponds to images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’, respectively; the column (1) shows corresponding plain images; columns (2)–(4)
correspond to the plain images’ distributions of ‘horizontal direction’, ‘vertical direction’ and ‘diagonal direction’, respectively; column (5) is the corresponding cipher images by encryption, and columns (6)–(8) correspond to the cipher images’ distributions of ‘horizontal direction’, ‘vertical direction’ and ‘diagonal direction’, respectively.

Table 7. Correlation coefficients.

| Image  | Horizontal Plain | Horizontal Cipher | Vertical Plain | Vertical Cipher | Diagonal Plain | Diagonal Cipher |
|--------|------------------|-------------------|----------------|----------------|----------------|----------------|
| Lena   | 0.9839           | 0.00025           | 0.9717         | 0.0019         | 0.9589         | 0.00033         |
| Baboon | 0.7670           | 0.0029            | 0.8728         | 0.0021         | 0.7369         | 0.0036          |
| Pepper | 0.9782           | 0.0035            | 0.9765         | -0.00028       | 0.9615         | -0.0059         |
| Airplane| 0.9642           | -0.0082           | 0.9666         | 0.0086         | 0.9380         | 0.0056          |
| Lake   | 0.9723           | 0.0043            | 0.9754         | -0.0052        | 0.9573         | 0.0056          |

Figure 6. Correlation distributions. Rows (a)–(e) correspond to images of ‘Lena’, ‘Baboon’, ‘Pepper’, ‘Airplane’ and ‘Lake’, respectively; Column (1) shows the corresponding plain images; Columns (2)–(4) correspond to the plain images’ distributions of ‘horizontal direction’, ‘vertical direction’ and ‘diagonal direction’, respectively; Column (5) is the corresponding cipher images by encryption; Columns (6)–(8) correspond to the cipher images’ distributions of ‘horizontal direction’, ‘vertical direction’ and ‘diagonal direction’, respectively.

4.4. Key Sensitivity Analysis

Key sensitivity is the basic requirement of image cryptosystem security, and it can reflect the performance of diffusion. For this test, we select ‘Lena’ to be the test image. Firstly, we encrypt ‘Lena’ using initial key, which was declared at the beginning of Section 4, and denote the cipher image as C1. C1 was used as the benchmark for comparison experiments, then we modify a parameter or initial value of the chaotic system which are the real keys of our proposed scheme. In the first test, we modify a parameter of PWLCM which is used in generating the rules selection control sequence as \( p = p + 10^{-15} \), and denote the corresponding cipher image as C2. In the second test, we modify the initial value of PWLCM which is used in generating key sequence as \( x = x + 10^{-15} \), and denote the corresponding cipher image as C3. In the third test, we modify the initial value of hyper chaotic Lorenz system which is used in permutation as \( x_0 = x_0 + 10^{-15} \), and denote the corresponding cipher image as C4. In the fourth test, we modify the initial value of hyper chaotic Lorenz system as \( y_0 = y_0 + 10^{-15} \), and denote the corresponding cipher image as C5. The test result is shown in Figure 7. The difference
between C1 and C2 is shown in Figure 7g, the difference between C1 and C3 is shown in Figure 7h, the difference between C1 and C4 is shown in Figure 7i and the difference between C1 and C5 is shown in Figure 7j.

We also use NPCR and UACI to quantitatively measure the difference between cipher images. The result is shown in Table 8. According to this test result, our proposed image cryptosystem has enough key sensitivity.

Table 8. Quantitative analysis of key sensitivity.

| Cipher Images | NPCR(%) | $\Delta_{99.5893}$ | UACI(%) | $U_{99.5893}^*$ = 33.3730% | $U_{99.5893}^*$ = 33.5541% |
|---------------|---------|---------------------|---------|---------------------------|---------------------------|
| C1 and C2     | 99.6185 | Pass                | 33.405  | Pass                      | Pass                      |
| C1 and C3     | 99.6197 | Pass                | 33.3950 | Pass                      | Pass                      |
| C1 and C4     | 99.6174 | Pass                | 33.4428 | Pass                      | Pass                      |
| C1 and C5     | 99.5956 | Pass                | 33.4201 | Pass                      | Pass                      |

Figure 7. Key sensitivity. (a) Plain image of ‘Lena’; (b–f) C1, C2, C3, C4 and C5, respectively; (g) $|C1 - C2|$; (h) $|C1 - C3|$; (i) $|C1 - C4|$; (j) $|C1 - C5|$.

4.5. Information Entropy

4.5.1. Global Shannon Entropy

The degree of uncertainty of a cipher image can reflect the diffusion performance of an image cryptosystem, and the global information entropy is a measurement to indicate the uncertainty degree of the whole image information. The global information entropy is given in Equation (15).

$$H(s) = \sum_{i=0}^{2^k-1} P(s_i) \log_2 \left( \frac{1}{P(s_i)} \right),$$

where $K$ is the bit depth of the test image, e.g., $K = 8$ for an 8-bit gray image, and $P(s_i)$ means the probability of $s_i$. In the ideal case, the information entropy of 8-bit gray image is $H(s) = 8$ bits. The entropy test results are shown in Table 9. In our test results, entropies of cipher images are very close to the ideal value 8. Thus, our cryptosystem has good diffusion performance.
4.5.2. Local Shannon Entropy

Global Shannon entropy reflects the total randomness of an image, and it has certain limitations in some occasions. Therefore, local Shannon entropy (LSE) was proposed by Wu et al. [29]. To measure local entropy, we first randomly select $k$ non-overlapping image blocks $B_1, B_2, \ldots, B_k$ with $T_B$ pixels from image $I$, then the LSE is defined by:

$$H_{k,T_B}(I) = \frac{1}{k} \sum_{i=1}^{k} H(B_i),$$

where $H(B_i)$ is the Shannon entropy of image block $B_i$, and can be calculated by Equation (15). In this test, we select parameters $(k, T_B) = (30, 1936)$. Under these parameters, the ideal value of LSE is 7.902469317. When we set significance $\alpha = 0.05$, we consider the tests passed when the test LSE values fall into the interval $(7.901901305, 7.903037329)$. The LSE test results are shown in Table 10.

| Image  | Lena | Baboon | Pepper | Airplane | Lake  |
|--------|------|--------|--------|----------|-------|
| Entropy| 7.999258 | 7.999334 | 7.999414 | 7.999328 | 7.999282 |

4.6. Computation and Complexity Analysis

Due to DNA computing having advantages when processing in parallel, our scheme is applicable to encrypt a large amount of image data in the actual DNA computing environment. In this section, we only discuss the time-consuming part of cryptography for computing floating point numbers. We use PWLCM to control randomly DNA encoding and to generate key sequence, it needs $\Theta(5 \times M \times N)$ iterations of computing floating point number. And we use HCLS to control permutation, it needs $\Theta(M \times N)$ iterations of computing floating point number. The Matlab programming to simulate DNA calculation is low efficiency, under the hardware and software environment mentioned at beginning of Section 4, the encryption of $512 \times 512$ image is total cost 3.233 s. Although MATLAB simulation results have long running time, the proposed algorithm is still applicable to the actual DNA computing scene. After all, our purpose of this paper is to design a secure image encryption scheme in a real DNA computing scene, and simulation is only a means to verify the correctness and security of the proposed image cryptosystem.

4.7. Security Comparison

In this part, we make a comparison with other works on some common security analysis indicators, the comparison results are shown in Table 11, and shown that our proposed scheme has very excellent security performance.
Table 11. The security comparison

| Algorithms | Cipher Correlation Coefficients | Global Entropy | Local Entropy | Key Space | Plaintext Sensitivity |
|------------|---------------------------------|----------------|--------------|-----------|-----------------------|
|            | Horizontal | Vertical | Diagonal |            | NWC (%) | UACI (%) |
| Our work   | 0.0025     | 0.0019   | 0.00033  | 7.999258  | 7.902361  | 2423 | 99.6159 | 33.4846 |
| Ref. [16]  | -0.0021    | 0.0009   | 0.0003   | 7.9971656 | -         | 2.96 × 10^{149} | 99.6135 | 30.9255 |
| Ref. [20]  | 0.0037     | -0.0004  | -0.0378  | 7.9993    | -         | 2512 | 99.61 | 33.46 |
| Ref. [21]  | -0.0014    | -0.0020  | -0.0012  | 7.9993    | -         | -    | 99.6258 | 33.4153 |

5. Conclusion

The security and privacy of digital images, such as medical images, satellite remote sensing images, social network images, etc., is very important in communication. To ensure the security and privacy of digital images, we proposed a novel chaos based image encryption scheme with randomly DNA encode and plaintext related permutation in this paper. In the proposed scheme, for utilizing the ultra-high information density, great parallelism and ultra-low energy consumption of DNA, we encode the plain image randomly into a nucleotide sequence controlled by the piecewise linear chaotic map. Under the control sequence generated by hyper-chaotic Lorenz system, the plaintext related permutation is done, which can increase the plaintext sensitivity and improve resistance to differential attack. Moreover, we use key DNA sequence to make diffusion processing, which can strongly resist statistical attack. In addition, many common security analysis methods are used to test our proposed image cryptosystem. The test results shown that our proposed image cryptosystem has excellent security performance in digital image communication. The image encryption based on chaos and DNA computing are still under constant research, and there are still many problems that need to be further studied and solved. In the next stage, we will focus on the multi-image aggregation encryption and parallel DNA image encoding.

Author Contributions: Formal analysis, Z.L.; funding acquisition, C.P.; investigation, Z.L.; methodology, Z.L.; project administration, C.P.; software, W.T. and L.L.; writing—original draft preparation, Z.L.; writing—review and editing, C.P., W.T. and L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: Our research is supported by the National Natural Science Foundation of China (U1836205, 61662009, and 61772008), the open Foundation of Guizhou Provincial Key Laboratory of Public Big Data (2017BDKFJ023, 2017BDKFJ026), the Science and Technology Foundation of Guizhou (Guizhou Science-Contract-Major-Program [2018]3001, Guizhou-Science-Contract-Major-Program [2018]3007, Guizhou-Science-Contract-Major-Program [2017]3002, Guizhou-Science-Contract-Support [2019]2004, Guizhou-Science-Contract-Support [2018]2162, Guizhou-Science-Contract-Support [2018]2159, Guizhou-Science-Contract [2017]1045, Guizhou-Science-Contract [2019]1049, Guizhou-Science-Contract [2019]1249), and Scientific Research Foundation of Guizhou province, China (QKHPTRC[2017]5788). The Project of Innovative Group in Guizhou Education Department ((2013)09). The Youth Science and Technology Talents Growth Project of the Guizhou Provincial Department of Education (Guizhou-Education-Contract-KY-Word [2018]260).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DNA Deoxyribonucleic acid
PWLCM Piecewise linear chaotic map
HCLS Hyper chaotic Lorenz system
DES Data Encryption Standard
AES Advanced Encryption Standard
RSA Rivest, Shamir and Adleman
References

1. Ding, L.; Ding, Q. A Novel Image Encryption Scheme Based on 2D Fractional Chaotic Map, DWT and 4D Hyper-chaos. *Electronics* **2020**, *9*, 1280. [CrossRef]
2. Dagadu, J.C.; Li, J.; Aboagye, E.O.; Ge, X. Chaotic medical image encryption based on Arnold transformation and pseudorandomly enhanced logistic map. *Structure* **2017**, *4*, 8096–8103.
3. Chai, X.; Bi, J.; Gan, Z.; Liu, X.; Zhang, Y.; Chen, Y. Color image compression and encryption scheme based on compressive sensing and double random encryption strategy. *Signal Process.* **2020**, 107684. [CrossRef]
4. Ding, M.; Jing, F. Digital image encryption algorithm based on improved Arnold transform. In Proceedings of the 2010 International Forum on Information Technology and Applications, Kunming, China, 16–18 July 2010; pp. 174–176.
5. Hou, W.B.; Wu, C.M. Image encryption and sharing based on Arnold transform. *J. Comput. Appl.* **2011**, *10*, 2682–2686.
6. Lorenz, E. Deterministic Non-period Flows. *J. Atmos. Sci.* **1972**, *20*, 130–141. [CrossRef]
7. Matthews, R. On the derivation of a “chaotic” encryption algorithm. *Cryptologia* **1989**, *13*, 29–42. [CrossRef]
8. Zhang, Y. The unified image encryption algorithm based on chaos and cubic S-Box. *Inf. Sci.* **2018**, *450*, 361–377. [CrossRef]
9. Batool, S.I.; Waseem, H.M. A novel image encryption scheme based on Arnold scrambling and Lucas series. *Multimed. Tools Appl.* **2019**, *78*, 27611–27637. [CrossRef]
10. Wang, X.; Çavuşoğlu, Ü.; Kacar, S.; Akgul, A.; Pham, V.T.; Jafari, S.; Alsaadi, F.E.; Nguyen, X.Q. S-box based image encryption application using a chaotic system without equilibrium. *Appl. Sci.* **2019**, *9*, 781. [CrossRef]
11. Zhang, Y.; Chen, A.; Tang, Y.; Dang, J.; Wang, G. Plaintext-related image encryption algorithm based on perceptron-like network. *Inf. Sci.* **2020**, *526*, 180–202. [CrossRef]
12. Li, Z.; Peng, C.; Tan, W.; Li, L. A Novel Chaos-Based Color Image Encryption Scheme Using Bit-Level Permutation. *Symmetry* **2020**, *12*, 1497. [CrossRef]
13. Ghehani, A.; LaBean, T.; Reif, J. DNA-based cryptography. In *Aspects of Molecular Computing*; Springer: New York, NY, USA, 2003; pp. 167–188.
14. Zhang, Q.; Liu, L.; Wei, X. Improved algorithm for image encryption based on DNA encoding and multi-chaotic maps. *AEU Int. J. Electron. Commun.* **2014**, *68*, 186–192. [CrossRef]
15. Liu, Y.; Tang, J.; Xie, T. Cryptanalyzing a RGB image encryption algorithm based on DNA encoding and chaos map. *Opt. Laser Technol.* **2014**, *60*, 111–115. [CrossRef]
16. Li, S.; Chen, G.; Mou, X. On the dynamical degradation of digital piecewise linear chaotic maps. *Int. J. Bifurc. Chaos* **2005**, *15*, 3119–3151. [CrossRef]
17. Wang, X.; Wang, M.H. Hyperchaotic image encryption algorithm based on bit-level permutation and DNA encoding. *Opt. Laser Technol.* **2020**, *132*, 106355. [CrossRef]
27. Liu, H.; Kadir, A.; Niu, Y. Chaos-based color image block encryption scheme using S-box. *AEU Int. J. Electron. Commun.* 2014, 68, 676–686. [CrossRef]

28. Wu, Y.; Noonan, J.P.; Agaian, S. NPCR and UACI randomness tests for image encryption. *Cyber J. Multidiscip. J. Sci. Technol. J. Sel. Areas Telecommun.* 2011, 1, 31–38.

29. Wu, Y.; Zhou, Y.; Saveriades, G.; Agaian, S.; Noonan, J.P.; Natarajan, P. Local Shannon entropy measure with statistical tests for image randomness. *Inf. Sci.* 2013, 222, 323–342. [CrossRef]

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).