A Tweak-Cube Color Image Encryption Scheme Jointly Manipulated by Chaos and Hyper-Chaos

Li-Lian Huang*, Shi-Ming Wang and Jian-Hong Xiang

College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China; wangshiming@hrbeu.edu.cn (S.-M.W.); xiangjianhong@hrbeu.edu.cn (J.-H.X.)
* Correspondence: huangllilan@hrbeu.edu.cn

Received: 14 October 2019; Accepted: 8 November 2019; Published: 13 November 2019

Featured Application: Since the proposed scheme can efficiently perform image encryption, it can be applied to real-time communication, such as telemedicine.

Abstract: This paper proposes a novel tweak-cube color image encryption scheme jointly manipulated by chaos and hyper-chaos. One-dimensional (1D) chaotic maps are effortless to operate, but the key space is relatively small. The hyperchaotic system has complex dynamics properties, which are capable of compensating for the defects of 1D chaotic maps. Thus, we first raise an improved 1D chaotic map with an increased key space. Then, we associate it with a four-dimensional (4D) hyperchaotic system to generate the key streams and further rotate and shift the rows and columns of each component of Red (R), Green (G), and Blue (B) for the color image. The permuting mode is to disturb the original position of the pixels by mimicking the way of twisting the Rubik’s cube. Moreover, the key stream updated by the plain images is also utilized for diffusion and scramble at the bit level. As a consequence, our cryptosystem enhances the security without at the expense of increasing time cost.

Keywords: tweak-cube; image encryption; chaos; hyper-chaos; security

1. Introduction

Digital image is one of the most popular multimedia forms at present, and it is widely used in politics, economy, national defense, and education. Since images contain a lot of information, we must worry about data being leaked, falsified, and counterfeited. Image encryption is a very effective means of protecting images by processing them into an unrecognizable form. Moreover, image data has its own unique properties: large amount of data, high redundancy, and strong correlation between adjacent pixels [1]. These features make the traditional encryption algorithms such as data encryption standard (DES), advanced encryption standard (AES), and Rivest-Shamir-Adleman (RSA) unsatisfactory. Thus, plenty of new encryption methods have been developed based on various technologies, such as chaos [2–5], compressed sensing [6,7], wave transmission [8,9], hash function [10,11], deoxyribonucleic acid coding [12,13], genetics [14], gray code [15], elliptical curve [16], and so on. Among them, the chaos-based image encryption schemes have received extensive attention due to the advantages of chaotic systems. They are ergodic, dispersive, and highly sensitive to initial conditions.

In comparison with the methods of encrypting with multidimensional (MD) chaotic maps, the design of cryptosystems with one-dimensional (1D) chaotic maps has the superiorities of simple structures and commodious operations. The most frequently utilized is the Logistic map. At the outset, Pareek et al. employed two Logistic maps to control the tracks of image pixels for encryption [17]. Singh et al. proposed an encryption method that combines the Logistic map and Hartley transform to improve robustness [18]. Then, Wang et al. applied the Logistic map to encrypt the R, G, B components...
of a color image at the same time and make these components affect each other [19]. Nonetheless, the Logistic map still has several defects: the chaos region is small, the output sequences are uneven, and the dynamic characteristics are not complicated enough. As a consequence, some researchers have ameliorated the Logistic map to design algorithms. Wu et al. presented a cryptosystem with a two-dimensional (2D) Logistic map of complicated basin structures and attractors [20]. Zhou et al. combined the Logistic map with another two existing 1D maps to raise a novel chaotic system for image encryption [21], and Hua et al. also introduced a new 2D Sine Logistic modulation map (2D-SLMM) to change the image pixel positions [22]. Even recently, some scholars yet have been hammering at encryptions with Logistic map transformation. Pak et al. adopted a linear–nonlinear–linear theory to shuffle plain images [23]. Hanis et al. put forward a novel butterfly-like structure algorithm by using the improved Logistic map [24].

Considering that the above techniques have insufficient finite parameter space to resist brute force attacks and are not conducive to real-time communication, the contributions we have made are summarized as follows. (1) We propose a novel 1D chaotic map with larger key space and better chaotic properties. Simultaneously, it is combined with a 4D hyperchaotic map to control the generation of the key streams, which makes the randomness more superior. (2) We excogitate a color image encryption scheme that mimics the Rubik’s cube motion, which can be effectively used for real-time communication and authentication without compromising time cost and safety quality. Moreover, due to our scheme can scramble images by rotating and shifting them at the pixel-level and bit-level promptly, it is suitable as a secure transmission mode in telemedicine systems. The rest of this paper is organized as follows. Section 2 introduces the modified 1D chaotic map and hyperchaotic system. Section 3 presents our cryptosystem. Section 4 gives simulation results. Section 5 evaluates the security performance. And the final section draws a conclusion.

2. Preliminaries

2.1. The Logistic-Fraction Hybrid Chaotic Map

In this section, a new chaotic map called the Logistic-Fraction hybrid chaotic map (LFHCM) is recommended, which is defined as Equation (1). The LFHCM is derived from the classical Logistic map and Fraction map [25], and they can be mathematically represented as Formulas (2) and (3).

\[ x_{k+1} = L(x_k) = ax_k(1 - x_k)^2 \times \frac{1}{x_k^2 + 1} \]  
\[ y_{k+1} = L(y_k) = by_k(1 - y_k) \]  
\[ z_{k+1} = F(z_k) = \frac{1}{z_k^2 + 0.1} - cz_k \]

where control numbers \( a \in [0, 11.5] \), \( b \in [0, 4] \), and \( c \in [0, 1] \). In addition, the output ranges of the three maps are also different, \( x_k \in [0, 1.56] \), \( y_k \in (0, 1) \), and \( z_k \in [-10.0025, 10.0025] \). Then, the analytical tests can verify that the LFHCM has more complex dynamic characteristics.

2.1.1. Bifurcation Graph

The bifurcation graph can objectively reflect the region and state of the chaotic behavior of a map. Figure 1 exhibits the bifurcation graphs of the LFHCM, Logistic map, and Fraction map. It can be found that the Logistic map shows chaotic characteristics only when \( b \in (3.6, 4) \). Although the Fraction map has more complex dynamic properties than the Logistic map, its chaotic interval length is also less than 1. The interval in which the LFHCM presents chaotic state is \( a \in [5.9, 11.5] \). It is obvious that the chaotic region of the new chaotic map is increased, which means that the key space is also improved. Besides, in contrast to the intervals of some 1D chaotic maps in Table 1, the LFHCM has the largest region.
2.1.2. Lyapunov Exponent

The Lyapunov exponent is used to characterize the motion of a system. When it is calculated as a positive number, it indicates that a map has chaotic behavior. Not only that, the larger the Lyapunov exponent, the better the chaotic properties of the map. Besides, the definition of the Lyapunov exponent for a 1D chaotic map \( x_{k+1} = F(x_k) \) is as follows:

\[
\lambda(x_0) = \lim_{n \to \infty} \frac{1}{n} \sum_{k=0}^{n-1} \ln |F'(x_k)|
\]

Since the Lyapunov exponent \( \lambda \) is independent of the initial value \( x_0 \), we arbitrarily chose \( x_0 = 0.5 \). Moreover, in order to observe the change between the adjacent iteration points as a whole, the number of iterations \( n \) should be chosen as large as possible, so we set \( n = 20,000 \). Then, calculated the derivatives of the LFHCM, Logistic map, and Fraction map according to Equations (1)–(3), and substituted the results into Equation (4) to get the Lyapunov exponent plots exhibited in Figure 2. It can be seen that the LFHCM has larger Lyapunov exponents than the other two maps, which means that its orbit is rapidly separated in the direction of the positive Lyapunov exponent and it is sensitive to initial values.

![Figure 1](image1.png)

**Figure 1.** The bifurcation graphs of the (a) Logistic-Fraction hybrid chaotic map (LFHCM); (b) Logistic map; (c) Fraction map.

**Table 1.** Comparison of chaotic intervals between the LFHCM and other one-dimensional (1D) chaotic maps.

| Chaotic Maps                        | Chaotic Regions                  |
|------------------------------------|----------------------------------|
| The LFHCM                          | [5.9, 11.5]                      |
| The Logistic map                   | (3.6, 4)                         |
| The Fraction map                   | [0.07, 0.15] ∪ [0.25, 0.52] ∪ [0.57, 0.68] |
| Modified logistic map in [24]      | [3.9, 6.27]                      |
| Sine-Tent system in [26]           | [0, 4]                           |

![Figure 2](image2.png)

**Figure 2.** The Lyapunov exponents of the (a) LFHCM; (b) Logistic map; (c) Fraction map.
2.1.3. Approximate Entropy

Approximate entropy (ApEn) is a nonlinear dynamic parameter used to quantify the regularity and unpredictability of time series fluctuations [27]. It can also measure the complexity of a chaotic map. Moreover, the larger the calculated ApEn value, the more complicated the sequence generated by the chaotic map is. We set the control parameters 11.44, 3.99, and 0.31 for the LFHCM, Logistic map, and Fraction map separately. Then, we calculated the ApEn of the three maps with different thresholds, as shown in Table 2. It can be seen that the proposed LFHCM has larger ApEn values, which proves that it can produce more complex sequences.

Table 2. Approximate entropy (ApEn) values of 1D chaotic maps.

| Chaotic Maps    | Thresholds |
|----------------|------------|
|                | 0.02 0.04 0.06 0.08 0.10 0.12 0.14 0.16 0.18 0.20 0.22 0.24 0.26 |
| The LFHCM      | 0.36 0.55 0.66 0.71 0.75 0.76 0.81 0.82 0.84 0.85 0.84 0.86 0.85 |
| The Logistic map | 0.35 0.47 0.53 0.53 0.55 0.56 0.57 0.59 0.27 0.56 0.54 0.53 0.54 |
| The Fraction map | 0.15 0.23 0.28 0.33 0.37 0.38 0.41 0.42 0.45 0.47 0.46 0.47 0.46 |

2.1.4. NIST Test

The NIST (National Institute of Standards and Technology) SP800-22 Revla test consists of 15 sub-items that are used to detect the randomness of the sequences [28]. The method of detection is by comparing the calculated \( p \)-value of each sub-item to a prescribed significant level 0.01, and the generated values are all expected to greater than 0.01 to pass the test. We iterated Equation (1) with different initial values to generate 100 binary streams with 1,000,000 bits, then calculated their average values. The results are shown in Table 3. It is obvious that the calculated \( p \)-value of each sub-item falls into \([0.01, 1]\), that is, the outputs generated by LFHCM can pass all the tests of NIST.

Table 3. NIST SP800-22 Revla test results of binary streams generated by LFHCM.

| Statistical Tests       | \( p \)-Value | Result |
|-------------------------|--------------|--------|
| Frequency               | 0.3699       | Pass   |
| Block frequency         | 0.7872       | Pass   |
| Cumulative sums         | 0.6584       | Pass   |
| Runs                    | 0.2645       | Pass   |
| Longest run             | 0.6385       | Pass   |
| Rank                    | 0.2022       | Pass   |
| Fast fourier transform   | 0.3492       | Pass   |
| Non-overlapping template| 0.6286       | Pass   |
| Overlapping template    | 0.5575       | Pass   |
| Universal               | 0.6917       | Pass   |
| Approximate entropy     | 0.5434       | Pass   |
| Random excursions       | 0.5753       | Pass   |
| Random excursions variant| 0.5349     | Pass   |
| Linear complexity       | 0.4639       | Pass   |
| Serial                  | 0.3972       | Pass   |

2.2. Hyperchaotic System

Since the hyperchaotic system contains more than two positive Lyapunov exponents, its chaotic properties are better, and its trajectory is more unpredictable. When a hyperchaotic system is applied
to image encryption, the cryptosystem has a larger key space. The 4D hyperchaotic system utilized in this paper [29] is defined as follows:

\[
\begin{align*}
\dot{x} &= \lambda(y - x) + \beta_1 w \\
\dot{y} &= \alpha x - xz + \beta_2 w \\
\dot{z} &= -\sigma z + xy + \beta_3 w \\
\dot{w} &= -\delta x 
\end{align*}
\]  
(5)

where \( \lambda, \alpha, \sigma, \delta, \beta_1, \beta_2, \) and \( \beta_3 \) are control parameters of the system. Only when \( \lambda = 35, \alpha = 35, \sigma = 3, \delta = 5, \beta_1 = 1, \beta_2 = 0.2, \) and \( \beta_3 = 0.3, \) the system exhibits hyperchaotic behavior. Moreover, the phase trajectory of the system is displayed in Figure 3.

![Phase Trajectories](image)

**Figure 3.** The phase trajectories of the (a) \( x, y \) plane; (b) \( x, z \) plane; (c) \( x, w \) plane; (d) \( y, z \) plane; (e) \( y, w \) plane; (f) \( x, y, z \) space.

### 3. The Proposed Image Cryptosystem

#### 3.1. Key Stream Generation

In this subsection, we introduce the key stream generation process of the encryption scheme in this paper. The specific steps are as follows:

**Step 1:** Decompose the plain image \( P \) of size \( m \times n \) into three matrices, \( P_R, P_G, \) and \( P_B. \) Set initial values \( x_{i0} \) and \( \{x_0, y_0, z_0, w_0\} \) for the LFHCM and 4D hyperchaotic system, respectively. Then apply the plain image \( P \) to update the initial values according to Equation (6)

\[
\begin{align*}
x_{20} &= x_{10} + \sum_{i=1}^{m} \sum_{j=1}^{n} P_R(i,j) \\
x_{20}' &= x_{10} + \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} P_B(i,j)}{256^3} \times \frac{x_0 + y_0 + w_0 + z_0 + 1}{x_0 + y_0 + w_0 + z_0 + 2}
\end{align*}
\]  
(6)
where the key streams $K_i$ of the three matrixes is subjected to Equation (15).

Step 2: Select two control parameters $c_1, c_2$ for the LFHCM, and iterate it $(m + 500)$ and $(n + 500)$ times with the initial values $[x_{20}, x_{200'}]$. Discard the first 500 elements to form two chaotic sequences, labeled $(x_{21}(k))_{k=1}^m$ and $(x_{22}(k))_{k=1}^n$.

Step 3: Pre-iterate the 4D hyperchaotic system described in the former section 1000 times. Then, iterate another $u$ and $v$ times to generate two pseudo-random sequences, $s_1(k)$ and $s_2(k)$, with initial values $[x_0', y_0', z_0', w_0']$, respectively. They can be defined as follows:

$$u = \left\lceil \frac{(m + 1)}{4} \right\rceil$$
$$v = \left\lceil \frac{(n + 1)}{4} \right\rceil$$

$$\begin{align*}
(s_1(k))_{k=1}^u &= ((x_1(k))_{k=1}^u, (y_1(k))_{k=1}^u, (z_1(k))_{k=1}^u, (w_1(k))_{k=1}^u) \\
(s_2(k))_{k=1}^v &= ((x_2(k))_{k=1}^v, (y_2(k))_{k=1}^v, (z_2(k))_{k=1}^v, (w_2(k))_{k=1}^v)
\end{align*}$$

where $\lceil \cdot \rceil$ denotes that the element returns the smallest integer in the infinite direction. Then, arbitrarily intercept $m$ and $n$ elements corresponding to the sequences $(s_1(k))_{k=1}^u$ and $(s_2(k))_{k=1}^v$ to produce sequences $(s_1'(k))_{k=1}^m$ and $(s_2'(k))_{k=1}^n$.

Step 4: Perform the following operations for the four chaotic sequences, $(x_{21}(k))_{k=1}^m$, $(x_{22}(k))_{k=1}^n$, $(s_1'(k))_{k=1}^m$, and $(s_2'(k))_{k=1}^n$ constructed in Step 2 and Step 3:

$$\begin{align*}
\hat{x}_{21}(k) &= \left\lfloor 10^5 \times x_{21}(k) \right\rfloor - 10^5 \times x_{21}(k) \\
\hat{x}_{22}(k) &= \left\lfloor 10^5 \times x_{22}(k) \right\rfloor - 10^5 \times x_{22}(k)
\end{align*}$$

$$\begin{align*}
\hat{s}_1'(k) &= 10^6 \times s_1'(k) - \left\lfloor 10^6 \times s_1'(k) \right\rfloor \\
\hat{s}_2'(k) &= 10^6 \times s_2'(k) - \left\lfloor 10^6 \times s_2'(k) \right\rfloor
\end{align*}$$

where $\lfloor \cdot \rfloor$ represents the function that make the element return the nearest integer towards minus infinity.

Step 5: Further convert these sequences according to Formulas (13) and (14).

$$\begin{align*}
\hat{x}_{21}(k) &= \hat{x}_{21}(k) \times 1000 \mod m + 1 \\
\hat{x}_{22}(k) &= \hat{x}_{22}(k) \times 1000 \mod n + 1
\end{align*}$$

$$\begin{align*}
\hat{s}_1'(k) &= \text{fix}((\hat{s}_1'(k) + 100) \mod 1) \times 10^{12} \mod m + 1 \\
\hat{s}_2'(k) &= \text{fix}((\hat{s}_2'(k) + 500) \mod 1) \times 10^{12} \mod n + 1
\end{align*}$$

The resulting four sequences are treated as key streams that can be expressed as $K_1(\hat{x}_{21}(k))_{k=1}^m$, $K_2(\hat{x}_{22}(k))_{k=1}^n$, $K_3(s_1'(k))_{k=1}^m$ and $K_4(s_2'(k))_{k=1}^n$.

### 3.2. Encryption Scheme

**Input:** Color plain image $P$ of size $m \times n$. **Output:** Color cipher image of the same size.

**Step 1:** Calculate the initial conditions based on the plain image $P$, and set the control parameters $c_1, c_2$ for the LFHCM. After selecting the initial values for the 4D hyperchaotic system, the key streams $K_1(\hat{x}_{21}(k))_{k=1}^m$, $K_2(\hat{x}_{22}(k))_{k=1}^n$, $K_3(s_1'(k))_{k=1}^m$ and $K_4(s_2'(k))_{k=1}^n$ are generated by the preceding subsection.

**Step 2:** The color plain image is decomposed into three matrixes of $P_R$, $P_G$, and $P_B$, then each row $i$ of the three matrices is subjected to Equation (15).
When the computed values $\lambda_{R,G,B}(i) = 0$, the $P_R$, $P_G$, and $P_B$ matrices are respectively cyclically shifted to the left by $K_1(\hat{s}_{21}(k))_{k=1}^n$ positions, where $k = i$. Then, when $\lambda_{R,G,B}(i) = 1$, the cyclic shifts are performed correspondingly to the right $K_1(\hat{s}_{21}(k))_{k=1}^n$ positions. As shown in Figure 4, similar to the principle of Rubik’s Cube, the pixels of the image can also be rotated and shifted to the left and right.

![Figure 4. Schematic diagram of cyclic shifting in all directions.](image)

**Figure 4.** Schematic diagram of cyclic shifting in all directions.

**Step 3:** For each column of $P_R$, $P_G$, and $P_B$, the operations are as follows:

$$
\begin{align*}
\lambda_R(i) &= \sum_{i=1}^{m} P_R(i,j) \mod m \\
\lambda_G(i) &= \sum_{i=1}^{m} P_G(i,j) \mod m \\
\lambda_B(i) &= \sum_{i=1}^{m} P_B(i,j) \mod m
\end{align*}
$$

(16)

When the calculated values $\lambda_{R,G,B}(j) > 128$, the $P_R$, $P_G$, and $P_B$ matrices are respectively cyclically shifted upward by $K_2(\hat{x}_{22}(k))_{k=1}^n$ positions, where $k = j$. Then, when $\lambda_{R,G,B}(j) \leq 128$, the cyclic shifts are performed correspondingly downward to $K_2(\hat{x}_{22}(k))_{k=1}^n$ positions. The upward and downward cyclic shift diagram is also displayed in Figure 4, and three scrambled matrixes, $C_{1R}$, $C_{1G}$, and $C_{1B}$, are obtained.

**Step 4:** Execute bit-wise diffusion operation on each row of the matrixes $C_{1R}$, $C_{1G}$, and $C_{1B}$ according to Equation (17).

$$
\begin{align*}
C_{2R} &= (C_{1R}(i,j) + K_2(\hat{x}_{22}(k))_{k=1}^n) \mod 256 \oplus K_2(\hat{x}_{22}(k))_{k=1}^n \\
C_{2G} &= (C_{1G}(i,j) + K_2(\hat{x}_{22}(k))_{k=1}^n) \mod 256 \oplus K_2(\hat{x}_{22}(k))_{k=1}^n \\
C_{2B} &= (C_{1B}(i,j) + K_2(\hat{x}_{22}(k))_{k=1}^n) \mod 256 \oplus K_2(\hat{x}_{22}(k))_{k=1}^n
\end{align*}
$$

(17)

**Step 5:** For the odd rows of the matrices $C_{2R}$, $C_{2G}$, and $C_{2B}$, they are cyclically shifted to the left by $K_3(s'_{1}(k))_{k=1}^m$ positions, and the even rows are rotated to the right by the same positions. Meanwhile, for the odd columns of the matrices $C_{2R}$, $C_{2G}$, and $C_{2B}$, they are respectively cycle shifted upward by $K_4(s'_{2}(k))_{k=1}^m$ positions, and the even columns are correspondingly shifted downward by the same positions. Then, three disordered matrixes, $C_{3R}$, $C_{3G}$, and $C_{3B}$, are produced.
Step 6: Finally, a bit-level scrambling is implemented by the following formula:

\[
\begin{align*}
C_{ER} &= C_{3R}(i, j) << (3 & 8 \text{bit}) \\
C_{EG} &= C_{3G}(i, j) >>> (4 & 8 \text{bit}) \\
C_{EB} &= C_{3B}(i, j) << (5 & 8 \text{bit})
\end{align*}
\]  

(18)

Merge the three components, \(C_{ER}, C_{EG},\) and \(C_{EB}\), to form the final encrypted image \(C_{E}\). The flowchart of our encryption scheme is displayed in Figure 5.

![Flowchart of the encryption scheme.](image)

3.3. Decryption Scheme

The decryption procedure is the inverse process of the encryption scheme, and it can be described briefly as follows:

Step 1: Apply the same initial conditions as encryption to iterate Equations (1) and (9) to generate the key streams \(K_1(\hat{x}_1(k))_{i=1}^m, K_2(\hat{x}'_2(k))_{i=1}^n, K_3(s'_1(\hat{\tilde{k}}))_{i=1}^{m_1}\), and \(K_4(s'_2(\hat{\tilde{k}}))_{i=1}^{n_1}\).

Step 2: Execute bit shift to the left or right of the components \(C_{ER}, C_{EG},\) and \(C_{EB}\) to get the matrices \(C_{3R}, C_{3G},\) and \(C_{3B}\).

Step 3: Cyclic shift the rows and columns of the matrices \(C_{3R}, C_{3G},\) and \(C_{3B}\) according to the opposite direction of step 5 in the previous subsection, then the matrices \(C_{2R}, C_{2G},\) and \(C_{2B}\) are acquired.

Step 4: Perform bit-wise operations utilizing Formula (19).

\[
\begin{align*}
C_{1R} &= (C_{2R}(i, j) \oplus K_2(\hat{x}'_2(k))_{i=1}^n + 256 - K_2(\hat{x}'_2(k))_{i=1}^n) \mod 256 \\
C_{1G} &= (C_{2G}(i, j) \oplus K_2(\hat{x}'_2(k))_{i=1}^n + 256 - K_2(\hat{x}'_2(k))_{i=1}^n) \mod 256 \\
C_{1B} &= (C_{2B}(i, j) \oplus K_2(\hat{x}'_2(k))_{i=1}^n + 256 - K_2(\hat{x}'_2(k))_{i=1}^n) \mod 256
\end{align*}
\]  

(19)
Step 5: Implement second reverse circular shifts on the rows and columns of the three matrices, $C_{1G}$, $C_{1C}$ and $C_{1B}$, according to Equations (15), (16), (19) and (20).

Step 6: Merge the three obtained matrixes to form the original image $P$.

4. Experimental Results

As exhibited in Figure 6, three 256 $\times$ 256 color plain images, “Couple”, “House” and “Tree”, were encrypted by python version 3.6 on a PC with an Intel Core i5, 3.4 GHz CPU, 8 GB memory.

The noise-like phenomena appear in cipher images, which proves the feasibility of the scheme from the visual perspective.

![Figure 6](image)

**Figure 6.** Encryption results: (a) the original images; (b) the encrypted images; (c) the decrypted images.

The encryption results are based on the following conditions:

$x_{10} = 0.26534695122143$, $c_1 = 11.422$, $c_2 = 11.439$, $x_0 = 3.68526527215001$, $y_0 = 1.15265332455324$, $z_0 = 4.65289567518252$, $w_0 = 2.68526527217589$. 
5. Performance Analysis

This section describes the performance analysis of our encryption scheme in terms of key space, key sensitivity, histogram analysis, correlation distribution, and information entropy evaluation, and so on.

5.1. Key Space

The key space is a collection of all legitimate keys. Only when it is large enough can it resist a variety of powerful attacks. Here the data range of the LFHCM is [0, 1.56], and the approximate intervals about the initial values of the 4D hyperchaotic system are $x_0 \in [-30, 27]$, $y_0 \in [-34, 30]$, $z_0 \in [6, 63]$, and $w_0 \in [-52, 40]$. In general, the precision of initial values and control numbers can reach $10^{-14}$. Thus, the size of the key space is able to achieve $(10^{14})^7 \times 10^3 = 10^{101} \approx 2^{335} \gg 2^{212}$ [24].

5.2. Key Sensitivity Analysis

The key sensitivity analysis is designed to observe the impact of small changes in the keys on cipher images. As for a chaotic map, the nuances of the initial conditions can produce two completely different sequences. Thus, we just changed initial value $x_{10}$ with $10^{-14}$, then we re-encrypted the original images in Figure 6a. The cipher images in Figure 7a were obtained. Figure 7b exhibits the pixel-by-pixel difference between Figures 6b and 7a, and it can be discovered that a subtle change to the keys can result in a completely different encryption. Moreover, Table 4 lists the difference rates between two encrypted images with only one of the keys changed, and these data are greater than 99.5%. Therefore, we can conclude that our encryption scheme has strong key sensitivity.

![Figure 7](image-url)
Table 4. Difference between cipher images with modified keys.

| Modified Key | Couple  | House  | Tree   |
|--------------|---------|--------|--------|
| $x_{10} + 10^{-14}$ | 99.6155 | 99.5880 | 99.5931 |
| $c_{1} + 10^{-14}$ | 99.6094 | 99.6190 | 99.6089 |
| $c_{2} + 10^{-14}$ | 99.6043 | 99.6002 | 99.6007 |
| $x_{0} + 10^{-14}$ | 99.6190 | 99.6237 | 99.6038 |
| $y_{0} + 10^{-14}$ | 99.6038 | 99.6165 | 99.6185 |
| $z_{0} + 10^{-14}$ | 99.5956 | 99.6185 | 99.6048 |
| $w_{0} + 10^{-14}$ | 99.6140 | 99.5946 | 99.6063 |

5.3. Histogram Analysis

The histogram is an important statistical feature of the image. An outstanding cryptosystem has the capacity to evenly distribute the histogram of the cipher image. In this case, the attacker cannot extract statistical features of the plain image from its cipher. Figure 8 presents the histograms of the plain and cipher images in Figure 7, which demonstrates that encryption makes the histogram information safely obscured.

![Figure 8](image_url)

Figure 8. Histograms of the (a) plain images in Figure 6a; (b) cipher images in Figure 6b.

In order to further verify the encryption effect, we used the statistical magnitude $\chi^2$ to measure the pixel values distribution of the plain and cipher images, which can be defined as follows:

$$\chi^2 = \sum_{i=0}^{255} \frac{(f_i - g_i)^2}{g_i}$$  \hspace{1cm} (20)

where $f_i$ denotes occurrence frequency of pixel value $i$, and $g_i$ represents the theoretical frequency of pixel value $i$. As displayed in Table 5, the chi-square test was performed on six images at a significant level of 0.05. Since the values are not only much smaller than the chi-square values of the plain images,
but they are also lower than the critical value $\chi^2_{0.05}(255) = 293.24783$, they all pass the test. That is, our cryptosystem can effectively resist statistical attack.

| Table 5. Chi-square test of plain and cipher images. |
|------------------------------------------------------|
| $\chi^2$ | Plain Image | Cipher Image |
| R | G | B | R | G | B |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Couple | 2.1036 x 10^5 | 3.3786 x 10^5 | 2.8964 x 10^5 | 247.6016 | 234.6563 | 249.8984 |
| House | 2.5858 x 10^5 | 2.9916 x 10^5 | 3.9404 x 10^5 | 274.3672 | 277.3203 | 286.9922 |
| Tree | 8.1371 x 10^4 | 5.7009 x 10^4 | 1.2982 x 10^6 | 254.0859 | 244.8125 | 264.3594 |
| Lena | 2.4628 x 10^5 | 1.1766 x 10^5 | 3.4033 x 10^5 | 257.1094 | 261.9603 | 202.1699 |
| Peppers | 5.4984 x 10^4 | 7.2077 x 10^4 | 1.1435 x 10^5 | 279.4395 | 268.0254 | 273.2910 |
| Airplane | 1.6115 x 10^5 | 1.5940 x 10^5 | 2.6505 x 10^5 | 234.2305 | 283.9785 | 243.3086 |
| Lena in [30] | - | - | - | 231.6144 | 248.3650 | 254.4806 |
| Pepper in [30] | - | - | - | 266.1728 | 273.2120 | 252.6552 |
| Airplane in [30] | - | - | - | 275.2378 | 271.7001 | 228.2813 |

5.4. Correlation Analysis

The correlation of two adjacent pixels can reflect the degree of diffusion about image pixels. Figures 9 and 10 exhibit the correlation distribution of the three channels, R, G and B, in the horizontal, vertical, main-diagonal and back-diagonal directions of the plain Tree image and cipher Tree image, respectively. Furthermore, we randomly chose 3000 pairs of adjacent pixels from a few original images and corresponding encrypted images. Then, we analyzed the correlation coefficients in horizontal, vertical, and main-diagonal directions according to Equation (21).

$$\rho_{xy} = \frac{E[(x - \mu_x)(y - \mu_y)]}{\sigma_x \sigma_y}$$

(21)

where $\mu$ and $\sigma$ indicate the mean value and standard deviation. The results are shown in Tables 6 and 7. It can be discovered that the correlation coefficients of the plain images are close to 1, while the results of the cipher images are near to 0. Hence, the encryption effect is satisfactory.

| Table 6. Correlation coefficients of plain images. |
|-----------------------------------------------------|
| Image | Plain Image |
| | Horizontal | Vertical | Main-Diagonal | Back-Diagonal |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| Couple | 0.9566 | 0.9566 | 0.9566 | 0.9566 |
| House | 0.9348 | 0.9693 | 0.9163 | 0.9588 |
| Tree | 0.9298 | 0.9542 | 0.9071 | 0.9047 |
| Lena | 0.9872 | 0.9756 | 0.9633 | 0.9709 |
| Peppers | 0.9466 | 0.9452 | 0.9100 | 0.9186 |
| Baboon | 0.8847 | 0.9099 | 0.8251 | 0.8120 |
5.4. Correlation Analysis

The correlation of two adjacent pixels can reflect the degree of diffusion about image pixels. Figures 9 and 10 exhibit the correlation distribution of the three channels, R, G and B, in the horizontal, vertical, main-diagonal and back-diagonal directions of the plain Tree image and cipher Tree image, respectively. Furthermore, we randomly chose 3000 pairs of adjacent pixels from a few original images and corresponding encrypted images. Then, we analyzed the correlation coefficients in horizontal, vertical, and main-diagonal directions according to Equation (21).

\[
\rho = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}
\]

where \(\mu\) and \(\sigma\) indicate the mean value and standard deviation. The results are shown in Tables 6 and 7. It can be discovered that the correlation coefficients of the plain images are close to 1, while the results of the cipher images are near to 0. Hence, the encryption effect is satisfactory.

Figure 9. Correlation coefficients of plain Tree image in each direction: (a) R channel; (b) G channel; (c) B channel.

Table 7. Correlation coefficients of cipher images.

| Image               | Plain Image | Horizontal | Vertical | Main-Diagonal | Back-Diagonal |
|---------------------|-------------|------------|----------|---------------|---------------|
| Couple              |             | 0.0024     | 0.0018   | -0.0017       | 0.0027        |
| House               |             | 0.0002     | -0.0004 | -0.0016       | 0.0008        |
| Tree                |             | 0.0001     | -0.0007 | 0.0015        | 0.0013        |
| Lena                |             | -0.0009    | 0.0008   | 0.0021        | 0.0004        |
| Peppers             |             | 0.0019     | -0.0010 | -0.0008       | 0.0021        |
| Baboon              |             | -0.0003    | -0.0028 | -0.0001       | -0.0013       |
| Lena in [14]        |             | 0.0056     | 0.0065   | -0.0073       | -             |
| Lena in [31]        |             | -0.0068    | -0.0054 | 0.0010        | -             |
| Lena in [32]        |             | 0.0059     | -0.0042 | 0.0180        | -             |
| Lena in [33]        |             | 0.0033     | 0.0027   | 0.0014        | -             |
| Peppers in [34]     |             | -0.0047    | -0.0024 | -0.0028       | -             |
| Peppers in [35]     |             | 0.0002     | 0.0022   | 0.0010        | -             |
| Baboon in [35]      |             | 0.0004     | -0.0047 | 0.0001        | -             |
Figure 10. Correlation coefficients of cipher Tree image in each direction: (a) R channel; (b) G channel; (c) B channel.

5.5. Information Entropy Analysis

Information entropy is expressed as the probability of discrete random events, which is used to measure the randomness of a system. It can be defined as follows:

$$H(s) = -\sum_{i=0}^{2^n-1} P(s_i) \log_2[P(s_i)]$$  \hspace{1cm} (22)$$

where $P(s_i)$ denotes the probability that the gray value $s_i$ may appear. For a grayscale image with data field of $[0, 255]$, the maximum value of information entropy is 8. In other words, if the result is close to 8, the algorithm is proved to be quite safe. It can be seen from Table 8 that the entropy values of ciphers are very close to the ideal values, so the possibility of information leakage is quite small.
Table 8. The results of information entropy.

| Image     | Plain Image | Cipher Image |
|-----------|-------------|--------------|
|           | R  | G  | B | R  | G  | B  |
| Couple    | 6.2499 | 5.9642 | 5.9309 | 7.9973 | 7.9974 | 7.9973 |
| House     | 6.4311 | 6.5389 | 6.2320 | 7.9970 | 7.9973 | 7.9967 |
| Tree      | 7.2104 | 7.4136 | 6.9207 | 7.9972 | 7.9973 | 7.9971 |
| Lena      | 7.2682 | 7.5901 | 6.9951 | 7.9993 | 7.9993 | 7.9994 |
| Peppers   | 7.3319 | 7.5242 | 7.0793 | 7.9992 | 7.9993 | 7.9992 |
| Baboon    | 7.7067 | 7.4744 | 7.7522 | 7.9992 | 7.9993 | 7.9993 |
| Lena in [35] | - | - | - | 7.9872 | 7.9875 | 7.9869 |
| Lena in [36] | - | - | - | 7.9970 | 7.9972 | 7.9970 |
| Lena in [37] | - | - | - | 7.9972 | 7.9973 | 7.9972 |
| Peppers in [35] | - | - | - | 7.9869 | 7.9880 | 7.9884 |
| Peppers in [37] | - | - | - | 7.9971 | 7.9975 | 7.9974 |
| Baboon in [35] | - | - | - | 7.9876 | 7.9879 | 7.9876 |
| Baboon in [37] | - | - | - | 7.9972 | 7.9972 | 7.9972 |

5.6. Noise Attacks Analysis

When the encrypted image is transmitted through the physical channel, it may be damaged by attackers [38]. Therefore, the cryptosystem should have the ability to resist noise attacks. That is, the cipher image after being attacked can still recover the information conveyed by the original image. In the experiment, we added Gaussian noise, Salt and pepper noise, and Speckle noise to the three cipher images in Figure 6b. As displayed in Figure 11, the decrypted images can still be successfully identified after being assaulted.

![Decrypted images with (a) Gaussian noise; (b) salt & pepper noise; (c) speckle noise.](image_url)
5.7. Data Loss Attack

A preeminent cryptosystem should withstand the cost of data loss during transmission and storage [39]. Similar to the test of noise attack, we also verified by decrypting the cipher image after data loss. Figure 12a,c reveals the data loss experiment on the cipher Tree image of different positions and sizes, respectively. It can be found from the corresponding decrypted images in Figure 12b,d that the Tree image after data leakage can still be recognized.

![Cipher images with lost data at different locations](image1.png)

(a) Cipher images with lost data at different locations; (b) decrypted images of (a); (c) cipher images with different sizes of data loss; (d) decrypted images of (c).

**Figure 12.** (a) Cipher images with lost data at different locations; (b) decrypted images of (a); (c) cipher images with different sizes of data loss; (d) decrypted images of (c).
5.8. Differential Attack Analysis

The sensitivity to the plain image determines the ability of an algorithm to resist differential attacks. The measurement methods are the number of pixels change rate (NPCR) and unified average changing intensity (UACI) [40]. The formulas are as follows:

\[
\text{NPCR}_{R,G,B} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left| E_{R,G,B}(i,j) \right| \times 100\%
\]

\[
\text{UACI}_{R,G,B} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left| \frac{M_{R,G,B}(i,j) - M'_{R,G,B}(i,j)}{255} \right| \times 100\%
\]

where \( M \) and \( M' \) are two cipher images whose plain images have only one different pixel. \( E_{R,G,B}(i,j) \) is utilized to determine the distinction between two cipher images, which can be defined as:

\[
E_{R,G,B}(i,j) = \begin{cases} 
0, & M_{R,G,B}(i,j) = M'_{R,G,B}(i,j) \\
1, & M_{R,G,B}(i,j) \neq M'_{R,G,B}(i,j)
\end{cases}
\]

In 2011, Wu et al. proposed more stringent standards of NPCR and UACI [41], which clearly demonstrates how high NPCR/UACI is, such that the encryption algorithm indeed has a high security level. The critical value \( N^*_\alpha \) of significance level \( \alpha \) is calculated by

\[
N^*_\alpha = \frac{F - \Phi^{-1}(\alpha)}{\sqrt{\frac{F}{MN}}}/(F + 1)
\]

where \( \Phi^{-1}(\cdot) \) is the inverse cumulative density function (CDF) of the standard Normal distribution, and \( F \) denotes the largest supported pixel value compatible with the ciphertext image format. Assuming that the calculated value is greater than \( N^*_\alpha \), then it can be proved to pass the test. The maximum \( N^{*+}_\alpha \) and minimum \( N^{-+}_\alpha \) of UACI can be computed according to Equation (27).

\[
\begin{align*}
N^{-+}_\alpha &= \mu_u - \Phi^{-1}(\alpha/2)\sigma_u \\
N^{*+}_\alpha &= \mu_u + \Phi^{-1}(\alpha/2)\sigma_u
\end{align*}
\]

where \( \mu_u = (F + 2)/(3F + 3) \), \( \sigma^2_u = (F + 2)(F^2 + 2F + 3)/(18(F + 1)^2)MN. \)

If the results of a scheme are between \( N^{-+}_\alpha \) and \( N^{*+}_\alpha \), it can be considered to pass the test. Table 9 reveals the theoretical values of NPCR and UACI at different significance levels for images of size 256 \times 256.

**Table 9.** The theoretical values of number of pixels change rate (NPCR) and unified average changing intensity (UACI).

| Size         | NPCR (%)                              | UACI (%)                             |
|--------------|---------------------------------------|--------------------------------------|
| 256 \times 256 | \( N^*_{0.05} \) | \( N^*_{0.01} \) | \( N^{++}_{0.001} \) | \( (N^{++}_{0.001}) \) | \( (N^*_{0.05}, N^{++}_{0.001}) \) | \( (N^*_{0.01}, N^{++}_{0.001}) \) |
| 256 \times 256 | 99.5693                          | 99.5527                        | 99.5341                        | (33.2824, 33.6447) | (33.2255, 33.7061) | (33.1594, 33.7677) |

In our experiment, we encrypted plain images with the initial keys. Then, we randomly chose one pixel in each original image and changed them slightly. The modified images were encrypted with the same keys. After reckoning the two types of ciphers, it can be found that all the NPCR values in Table 10 are greater than \( N^*_\alpha \), and draw near to the ideal value of 99.6094\%. Simultaneously, the UACI results for each image in Table 11 are within the theoretical interval. Thus, the algorithm can effectively resist differential attacks.
Table 10. The actual values of NPCR for different color images.

| Image       | Calculated NPCR (%) | Theoretical NPCR (%) |
|-------------|---------------------|----------------------|
|             | R       | G       | B       | N₀₁₀₀₁  | N₀₀₁₀₀₁ | N₀₀₀₁₀₀₀₁ |
| Couple      | 99.5956 | 99.6124 | 99.6185 | Passed  | Passed  | Passed    |
| House       | 99.6017 | 99.6268 | 99.5956 | Passed  | Passed  | Passed    |
| Tree        | 99.6094 | 99.6096 | 99.6129 | Passed  | Passed  | Passed    |
| Lena        | 99.6037 | 99.6128 | 99.5995 | Passed  | Passed  | Passed    |
| Peppers     | 99.5904 | 99.6246 | 99.6033 | Passed  | Passed  | Passed    |
| Baboon      | 99.5921 | 99.6109 | 99.6170 | Passed  | Passed  | Passed    |
| Lena in [35]| 99.6357 | 99.6416 | 99.6224 | -       | -       | -         |
| Lena in [37]| 99.6505 | 99.6444 | 99.6627 | -       | -       | -         |
| Lena in [42]| 99.6300 | 99.5900 | 99.6700 | -       | -       | -         |
| Lena in [43]| 99.6445 | 99.6353 | 99.6429 | -       | -       | -         |
| Peppers in [35]| 99.5693 | 99.6435 | 99.6171 | -       | -       | -         |
| Peppers in [37]| 99.6357 | 99.6158 | 99.6247 | -       | -       | -         |
| Baboon in [35]| 99.6563 | 99.6428 | 99.6493 | -       | -       | -         |
| Baboon in [37]| 99.6536 | 99.6078 | 99.6520 | -       | -       | -         |
| Baboon in [43]| 99.6461 | 99.6628 | 99.6277 | -       | -       | -         |

Table 11. The actual values of UACI for different color images.

| Image       | Calculated UACI (%) | Theoretical UACI (%) |
|-------------|---------------------|----------------------|
|             | R       | G       | B       | (N₀₀₁₀₀₁, N₀₁₀₀₁) | (N₀₀₀₁₀₀₀₁, N₀₀₁₀₀₁) | (N₀₀₀₁₀₀₀₁, N₀₁₀₀₁) |
| Couple      | 33.5911 | 33.4298 | 33.4019 | Passed  | Passed  | Passed    |
| House       | 33.5739 | 33.4600 | 33.4721 | Passed  | Passed  | Passed    |
| Tree        | 33.4029 | 33.4018 | 33.4793 | Passed  | Passed  | Passed    |
| Lena        | 33.5056 | 33.4533 | 33.4719 | Passed  | Passed  | Passed    |
| Peppers     | 33.4634 | 33.5485 | 33.4818 | Passed  | Passed  | Passed    |
| Baboon      | 33.5139 | 33.4682 | 33.5363 | Passed  | Passed  | Passed    |
| Lena in [35]| 33.5839 | 33.3954 | 33.4687 | -       | -       | -         |
| Lena in [37]| 33.4462 | 33.4131 | 33.4399 | -       | -       | -         |
| Lena in [42]| 33.4300 | 33.3900 | 33.5100 | -       | -       | -         |
| Lena in [43]| 33.1147 | 30.5298 | 27.7471 | -       | -       | -         |
| Peppers in [35]| 33.4317 | 33.5392 | 33.6326 | -       | -       | -         |
| Peppers in [37]| 33.4570 | 33.4705 | 33.4423 | -       | -       | -         |
| Baboon in [35]| 33.3687 | 33.5956 | 33.4528 | -       | -       | -         |
| Baboon in [37]| 33.4753 | 33.5090 | 33.4176 | -       | -       | -         |
| Baboon in [43]| 29.8421 | 28.1446 | 30.7024 | -       | -       | -         |

5.9. Time and Computational Complexity Analysis

Time and speed analysis is also a critical measurement feature for an encryption algorithm. Thus, the security of a scheme cannot be at the expense of increasing time consumption. The time costs of images with different sizes were measured separately, and the results are listed in Table 12. Obviously, the encryption time of our scheme is less than that of some recent encryption algorithms.

Ulteriorly, we will discuss the computational complexity of our scheme. At the outset, the generation process of the key streams made the computational complexity become $\Theta(4 \times m \times n)$. Afterwards, the rows and columns of the three components R, G, and B were simultaneously cyclically shifted twice so that the computational complexity changed to $\Theta(16 \times m \times n)$. Besides, the bit-level operations do not affect the complexity, hence the computational complexity of this paper is $\Theta(16 \times m \times n)$. 
Table 12. The encryption time and speed values for images with different sizes.

| Image Size    | Time (s) | Speed (bit/s) |
|---------------|----------|---------------|
| 256 × 256     | 0.4557   | 1091702.01    |
| 512 × 512     | 1.7383   | 1048478.39    |
| 1024 × 1024   | 6.4518   | 1127884.25    |
| 256 × 256 image in [31] | 0.4862 | -             |
| 256 × 256 image in [40] | 1.85   | -             |
| 256 × 256 image in [44] | 1.32   | -             |
| 512 × 512 image in [40] | 3.76   | -             |
| 1024 × 1024 image in [40] | 26.34  | -             |

6. Conclusions

In this paper, we devise a novel 1D chaotic map LFHCM with increased parameter space. The simulation results exhibit that the LFHCM has better chaotic properties than congeneric chaotic maps. In addition, combining LFHCM with a 4D hyperchaotic system, we also excogitate a color image encryption scheme based on tweak-cube theory. On the one hand, the secondary key streams updated by the plain image guarantee that the image withstands known-plaintext attack and chosen-plaintext attack. On the other hand, fast disorganizing at the bit-level and pixel-level allows our solution to be applied in crucial scenarios for real-time communication, such as telemedicine, authentication, and so forth. Compared with some state-of-arts, our work has achieved excellent encryption effects.

Author Contributions: Conceptualization, L.-L.H. and S.-M.W.; methodology, L.-L.H.; software, S.-M.W.; validation, L.-L.H., S.-M.W. and J.-H.X.; formal analysis, J.-H.X.; investigation, S.-M.W.; resources, L.-L.H.; data curation, L.-L.H.; writing—original draft preparation, S.-M.W.; writing—review and editing, L.-L.H.; visualization, J.-H.X.; supervision, L.-L.H.; project administration, S.-M.W.; funding acquisition, L.-L.H.

Funding: This research was funded by National Natural Science Foundation of China (No. 61203004, 61306142) and Natural Science Foundation of Heilongjiang Province (Grant No. F201220).

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

References
1. Wu, X.; Kan, H.; Kurths, J. A new color image encryption scheme based on DNA sequences and multiple improved 1D chaotic maps. Appl. Soft Comput. 2015, 37, 24–39. [CrossRef]
2. Li, C.; Luo, G.; Ke, Q.; Li, C. An image encryption scheme based on chaotic tent map. Nonlinear Dyn. 2017, 87, 127–133. [CrossRef]
3. Ye, G.; Chen, P.; Huang, X.; Mei, Q. An efficient pixel-level chaotic image encryption algorithm. Nonlinear Dyn. 2018, 94, 1–12. [CrossRef]
4. Kaur, M.; Kumar, V. Efficient image encryption method based on improved Lorenz chaotic system. Electron. Lett. 2018, 54, 562–564. [CrossRef]
5. Wang, X.; Çavuşoğlu, Ü.; Kacar, S.; Akgul, A.; Pham, V.-T.; Jafari, S.; Alsaadi, E.F.; Nguyen, Q.X. S-Box Based Image Encryption Application Using a Chaotic System without Equilibrium. Appl. Sci. 2019, 9, 781. [CrossRef]
6. Zhou, N.; Li, H.; Di, W.; Pan, S.; Zhou, Z. Image compression and encryption scheme based on 2D compressive sensing and fractional Mellin transform. Opt. Commun. 2015, 343, 10–21. [CrossRef]
7. Liu, X.; Mei, W.; Du, H. Optical image encryption based on compressive sensing and chaos in the fractional Fourier domain. J. Mod. Opt. 2014, 61, 1570–1577. [CrossRef]
8. Ye, G. A block image encryption algorithm based on wave transmission and chaotic systems. Nonlinear Dyn. 2014, 75, 417–427. [CrossRef]
9. Chai, X.L.; Zhu, C.J.; Yang, K.; Gao, Y.L. Block Image Encryption Scheme Based on Wave Transmission and Hyperchaotic Financial System. J. Chin. Comput. Syst. 2016, 37, 1329–1333.
10. Guesmi, R.; Farah, M.A.B.; Kachouri, A.; Samet, M. A novel chaos-based image encryption using DNA sequence operation and Secure Hash Algorithm SHA-2. Nonlinear Dyn. 2016, 83, 1–14. [CrossRef]
11. Guesmi, R.; Farah, M.A.B.; Kachouri, A.; Samet, M. Hash key-based image encryption using crossover operator and chaos. *Multimed. Tools Appl.* 2016, 75, 4753–4769. [CrossRef]
12. Yong, Z. The image encryption algorithm based on chaos and DNA computing. *Multimed. Tools Appl.* 2018, 77, 21589–21615.
13. Chai, X.; Chen, Y.; Brodye, L. A novel chaos-based image encryption algorithm using DNA sequence operations. *Opt. Lasers Eng.* 2017, 88, 197–213. [CrossRef]
14. Wang, X.; Zhang, H.L. A novel image encryption algorithm based on genetic recombination and hyper-chaotic systems. *Nonlinear Dyn.* 2016, 83, 333–346. [CrossRef]
15. Chen, J.X.; Zhu, Z.L.; Chong, F.; Zhang, L.B.; Zhang, Y. An efficient image encryption scheme using lookup table-based confusion and diffusion. *Nonlinear Dyn.* 2015, 81, 1151–1166. [CrossRef]
16. 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. [CrossRef]
17. Pareek, N.K.; Patidar, V.; Sud, K.K. Image encryption using chaotic logistic map. *Image Vis. Comput.* 2006, 24, 926–934. [CrossRef]
18. Singh, N.; Sinha, A. Optical image encryption using Hartley transform and logistic map. *Opt. Commun.* 2009, 282, 1104–1109. [CrossRef]
19. Wang, X.; Lin, T.; Xue, Q. A novel colour image encryption algorithm based on chaos. *Signal Process.* 2012, 92, 1101–1108. [CrossRef]
20. Yue, W.; Yang, G.; Jin, H.; Noonan, J.P. Image encryption using the two-dimensional logistic chaotic map. *J. Electron. Imaging* 2012, 21, 304.
21. Zhou, Y.; Long, B.; Chen, C.L.P. A new 1D chaotic system for image encryption. *Signal Process.* 2014, 97, 172–182. [CrossRef]
22. Hua, Z.; Zhou, Y.; Pun, C.M.; Chen, C.L.P. 2D Sine Logistic modulation map for image encryption. *Inf. Sci.* 2015, 297, 80–94. [CrossRef]
23. Pak, C.; Huang, L. A new color image encryption using combination of the 1D chaotic map. *Signal Process.* 2017, 138, 129–137. [CrossRef]
24. Hanis, S.; Amutha, R. A fast double-keyed authenticated image encryption scheme using an improved chaotic map and a butterfly-like structure. *Nonlinear Dyn.* 2019, 95, 421–432. [CrossRef]
25. Lu, J.A.; Wu, X.; Lü, J.; Kang, L. A new discrete chaotic system with rational fraction and its dynamical behaviors. *Chaos Solitons Fractals* 2004, 22, 311–319. [CrossRef]
26. Zhu, S.; Wang, G.; Zhu, C. A Secure and Fast Image Encryption Scheme Based on Double Chaotic S-Boxes. *Entropy* 2019, 21, 790. [CrossRef]
27. Pincus, S. Approximate entropy (ApEn) as a complexity measure. *Chaos* 1995, 5, 110–117. [CrossRef]
28. He, J.; Cai, J. Design of a New Chaotic System Based on Van Der Pol Oscillator and Its Encryption Application. *Mathematics* 2019, 7, 743. [CrossRef]
29. Guangzhou and China. A new hyperchaotic system and its adaptive tracking control. *Acta Phys. Sin.* 2012, 61, 273–335.
30. Wang, X.Y.; Li, P.; Zhang, Y.Q.; Liu, L.Y.; Zhang, H.; Wang, X. A novel color image encryption scheme using DNA permutation based on the Lorenz system. *Multimed. Tools Appl.* 2018, 77, 6243–6265. [CrossRef]
31. Sun, S. A novel hyperchaotic image encryption scheme based on DNA encoding, pixel-level scrambling and bit-level scrambling. *IEEE Photonics J.* 2018, 10. [CrossRef]
32. Enayatifar, R.; Sadaei, H.J.; Abdullah, A.H.; Lee, M.; Isnin, I.F. A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata. *Opt. Lasers Eng.* 2015, 71, 33–41. [CrossRef]
33. Xie, Y.; Yu, J.; Guo, S.; Ding, Q.; Wang, E. Image Encryption Scheme with Compressed Sensing Based on New Three-Dimensional Chaotic System. *Entropy* 2019, 21, 819. [CrossRef]
34. Cicenk, I.; Pusane, A.E.; Dundar, G. A novel design method for discrete time chaos based true random number generators. *Integr. VLSI J.* 2014, 47, 38–47. [CrossRef]
35. Liu, H.; Kadir, A.; Sun, X. Chaos-based fast color image encryption scheme with true random number keys from environmental noise. *IET Image Process.* 2017, 11, 324–332. [CrossRef]
36. Parvaz, R.; Zarebnia, M. A combination chaotic system and application in color image encryption. *Opt. Laser Technol.* 2018, 101, 30–41. [CrossRef]
38. Liu, H.; Kadir, A. Asymmetric color image encryption scheme using 2D discrete-time map. *Signal Process.* 2015, 113, 104–112. [CrossRef]

39. Wang, L.; Song, H.; Ping, L. A novel hybrid color image encryption algorithm using two complex chaotic systems. *Opt. Lasers Eng.* 2016, 77, 118–125. [CrossRef]

40. Wu, X.; Zhu, B.; Hu, Y.; Ran, Y. A Novel Color Image Encryption Scheme Using Rectangular Transform-Enhanced Chaotic Tent Maps. *IEEE Access* 2017, 5, 6429–6436.

41. Wu, Y. NPCR and UACI Randomness Tests for Image Encryption. *Cyber J. J. Sel. Areas Telecommun.* 2011, 1, 31–38.

42. Wang, X.; Zhao, Y.; Zhang, H.; Guo, K. A novel color image encryption scheme using alternate chaotic mapping structure. *Opt. Lasers Eng.* 2016, 82, 79–86. [CrossRef]

43. Suri, S.; Vijay, R. A synchronous intertwining logistic map-DNA approach for color image encryption. *J. Ambient Intell. Humaniz. Comput.* 2019, 10, 2277–2290. [CrossRef]

44. Xu, L.; Li, Z.; Li, J.; Hua, W. A novel bit-level image encryption algorithm based on chaotic maps. *Opt. Lasers Eng.* 2016, 78, 17–25. [CrossRef]