mathematics

MDPI

Article

Construction of a New 2D Hyperchaotic Map with Application in Efficient Pseudo-Random Number Generator Design and Color Image Encryption

Shenli Zhu [1,2], Xiaoheng Deng [2], Wendong Zhang [1,*] and Congxu Zhu [2,*]

[1] Software School, Xinjiang University, Urumqi 830091, China; zhuslxju@stu.xju.edu.cn
[2] School of Computer Science and Engineering, Central South University, Changsha 410083, China; dxh@csu.edu.cn
* Correspondence: wdzhang@xju.edu.cn (W.Z.); zhucx@csu.edu.cn (C.Z.); Tel.: +86-135-4968-3946 (C.Z.)

Abstract: This paper proposes a new two-dimensional discrete hyperchaotic system and utilizes it to design a pseudo-random number generator (PRNG) and an efficient color image encryption algorithm. This hyperchaotic system has very complex dynamic properties and can generate highly random chaotic sequences. The complex hyperchaotic characteristics of the system are confirmed via bifurcation diagram, chaotic attractor, Lyapunov exponents, correlation analysis, approximate entropy and permutation entropy. Compared with some traditional discrete chaotic systems, the new chaotic system has a larger range of chaotic parameters and more complex hyperchaotic characteristics, making it more suitable for application in information encryption. The proposed PRNG can generate highly random bit sequences that can fully pass all NIST testing items. The proposed color image encryption algorithm achieves cross-channel permutation and diffusion of pixels in parallel. These strategies not only greatly improve the encryption speed of color images, but also enhance the security level of cipher images. The simulation experiments and security analysis results show that the algorithm has strong robustness against differential attacks, statistical attacks and interference attacks, and has good application potential in real-time secure communication applications of color images.

Keywords: color image encryption; hyperchaotic map; PRNG; cross-channel permutation

MSC: 37E05; 68P30

Citation: Zhu, S.; Deng, X.; Zhang, W.; Zhu, C. Construction of a New 2D Hyperchaotic Map with Application in Efficient Pseudo-Random Number Generator Design and Color Image Encryption. *Mathematics* **2023**, *11*, 3171. https://doi.org/10.3390/math11143171

Academic Editor: Lingfeng Liu

Received: 3 July 2023
Revised: 15 July 2023
Accepted: 17 July 2023
Published: 19 July 2023

1. Introduction

With the development of computer vision and deep neural networks, a large amount of image data are disseminated on social networks. If these image data are not protected during transmission, this will easily lead to the disclosure of user privacy. From the perspective of confidentiality and security, it is worth studying how to protect image data [1]. Image encryption is an effective method to protect image data. Image data generally take up more space than character data, especially high-resolution color images, which consist of red, green and blue color channels. Therefore, they contain a large number of redundant pixels. So, for color image data, a traditional encryption algorithm such as RSA is unable to meet the efficient encryption of color image data [2–4]. However, in order to improve the time efficiency of the image encryption algorithm, there is usually an increase in the risk of the cryptographic algorithm being cracked. So, the question of how to improve the encryption efficiency under the premise of ensuring the security and reliability of the image encryption algorithm is a hot research issue [5–7].

Many novel image encryption algorithms have been proposed in recent years [8–10]. Among them, the image encryption algorithm using a chaotic system to generate a key sequence has attracted the attention of more and more researchers [11–13]. Due to the extreme sensitivity of chaotic systems to initial values and the high randomness of sequences

generated by chaotic systems, image encryption algorithms based on chaotic systems have been widely used [14–16]. Chaotic systems are usually referred to as mathematical models, which are classified into continuous-time differential equation systems and discrete-time iterative mapping systems [17–19], each of which has its own advantages and disadvantages. Generally speaking, a continuous-time differential equation system can only show a chaotic state in three dimension and above. However, a discrete-time iterative mapping system can show a chaotic state in one dimension. High-dimensional chaotic systems have a higher chaotic performance than low-dimensional chaotic systems. From the point of view of cryptography, this means that high-dimensional chaotic cryptosystems are more difficult to be cracked by attackers. However, the key sequence generated by a high-dimensional chaotic cryptosystem takes longer time and is difficult to be physically realized. In general, a low-dimensional chaotic system is easier to implement and saves more time than a high-dimensional chaotic system, and it can better encrypt a large amount of image stream data in real time. Therefore, the design and implementation of low-dimensional chaotic cryptosystems have attracted more and more researchers' attention [20–22]. In 2023, Lai et al. [16] proposed a high-sensitive cross-channel color image encryption algorithm using a two-dimensional chaotic map. In 2023, Zhou et al. [18] proposed a new two-dimensional discrete chaotic encryption system based on S-box. In 2022, Yang et al. [15] proposed an efficient color image encryption algorithm based on 2D sine-logistic–Gaussian coupled chaos and a multi-objective optimized S-box. In 2022, Elghandour et al. [14] proposed a new cryptographic algorithm using a two-dimensional piecewise smooth nonlinear chaotic map. In 2022, Erkan et al. [23] proposed an image encryption algorithm using a novel 2D chaotic map, which is based on Euler and Pi numbers. Although two-dimensional discrete chaotic systems are more convenient than high-dimensional continuous chaotic systems when applied to cryptographic algorithms, some existing two-dimensional discrete chaotic systems have the problems of being narrow and having a discontinuous chaotic behavior parameter interval. Such a consequence is that the key space of the cryptographic algorithm is small, which increases its risk of being cracked. Therefore, it is necessary to construct a new two-dimensional chaotic system with a better chaotic performance. Considering that the most prominent feature of a color image is the large amount of data, the time complexity of an encryption algorithm is worthy of attention for color image encryption. Therefore, a focus of this article is to reduce the time cost of encryption algorithms.

Based on the reasons above, a new 2D hyperchaotic map was constructed and applied in random number generation and color image encryption. The main contributions of this paper are as follows:

(1) A new two-dimensional hyperchaotic map with a wide range of chaotic parameters and strong chaotic performance is proposed.
(2) An efficient PRNG is designed, which can generate highly random bit sequences and can be used in various applications.
(3) A new image encryption scheme with cross-channel parallel permutation and diffusion (CCPPD) is proposed, which performs pixel scrambling and diffusion simultaneously, and the shuffling of pixel positions is across color channels. This strategy can improve the security and speed of encryption. The security of the image encryption scheme is verified using a large number of experiments and security analysis.

The rest of this paper is organized as follows. In Section 2, a new 2D hyperchaotic map is proposed, and its complex nature is analyzed and demonstrated. In Section 3, an efficient PRNG is designed and tested. A novel color image encryption algorithm based on the new 2D hyperchaotic map is proposed in Section 4. The experimental results and security analysis of the encryption scheme are provided in Section 5. Finally, the conclusion is given in Section 6.

2. The New 2D Hyperchaotic Map

This section presents the mathematic model of the 2D hyperchaotic map and numerically analyzes its dynamical complex nature in terms of attractor trajectory, bifurcation

diagrams and Lyapunov exponent (*LE*), approximate entropy (*ApEn*), permutation entropy (*PeEn*), which demonstrates that it is the preferred system in pseudo-random number generator (PRNG) design, and image encryption application.

2.1. Mathematic Model of the 2D Hyperchaotic Map

The proposed new system is a 2D discrete map consisting of the square of the sine function structure. Its mathematical model is expressed as follows:

$$\begin{cases} x_{n+1} = \sin^2(a\pi/x_n + by_n) \\ y_{n+1} = \sin^2(b\pi y_n + ax_n) \end{cases} \quad (1)$$

where a and b are the controlling parameters of the system. x_n and y_n are the state variables corresponding to the n-th discrete time point, while x_{n+1} and y_{n+1} are the state variables corresponding to the $(n + 1)$-th discrete time point. If $x_n \in \mathbb{R}$ and $y_n \in \mathbb{R}$, then $x_{n+1} \in \mathbb{R}$ and $y_{n+1} \in \mathbb{R}$. Equation (1) is a mapping of $\mathbb{R} \to \mathbb{R}$.

2.2. Bifurcation Diagram and Trajectory

A bifurcation diagram can visually show the evolution of state variables with different control parameters, which, in turn, reflects the parameter range where the system exhibits chaotic behavior. Figure 1 shows the bifurcation of variables x_n and y_n with the control parameters a in the range [0, 40], and $b = 30$ for the system with the initial state $(x_0, y_0) = (0.2, 0.3)$. From Figure 1, one can see that variables x_n and y_n can be spread over the entire value range, indicating that the 2D map is of ergodicity and has a chaotic characteristic. The trajectory of the chaotic attractor can reveal the behavior of the nonlinear system.

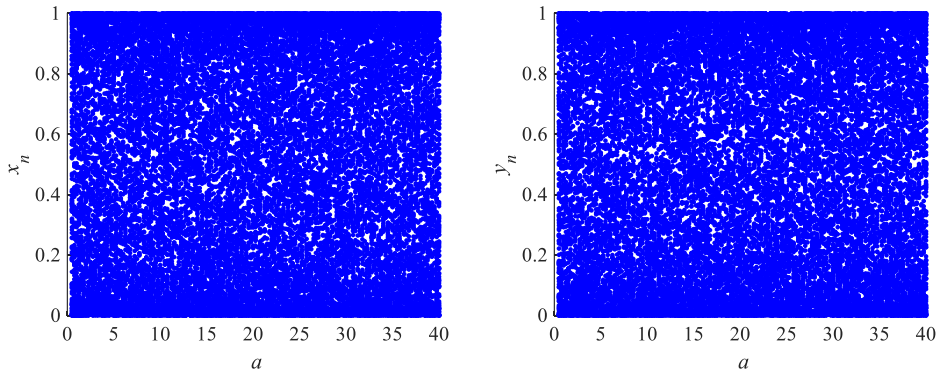


Figure 1. The bifurcation diagram of system (1): (**a**) bifurcation of variable x_n versus the control parameters a; (**b**) bifurcation of variable y_n versus the control parameters a.

Figure 2 depicts the phase diagram of the system with the initial conditions of $(x_0, y_0) = (0.2, 0.3)$ and the control parameters $a = 30$ and $b = 20$.

2.3. Lyapunov Exponent

The Lyapunov exponent (*LE*) is a characteristic quantity describing the sensitive dependence of nonlinear systems on the initial condition. If there is a positive Lyapunov exponent in a system, it means that the system is chaotic. If there are two or more positive Lyapunov exponents, then the system is said to be a hyperchaotic system, which indicates higher randomness and more complex dynamical behavior than a chaotic system. The *LE* of a one-dimensional discrete iterative mapping system $x_{n+1} = f(x_n)$ can be calculated using

$$LE = \lim_{N\to\infty} \frac{1}{N} \sum_{i=0}^{N-1} \ln|f'(x_i)| \quad (2)$$

where $f'(x_i)$ represents the value of the derivative of the iterated function $f(x_i)$ at the i-th time point, and N is the number of iterations of the system under certain fixed control parameters. The n-dimensional discrete iterative system ($n \geq 1$) has n Lyapunov exponents. The QR decomposition method is commonly used to calculate n Lyapunov exponents. For details of the QR decomposition algorithm, refer to [24]. Figure 3 intuitively displays a Lyapunov exponent graph of three 2D chaotic maps, the classical Hénon map, the recently proposed 2D-SLG map [15] and the new 2D map proposed in this paper. From Figure 3a, it can be seen that the Hénon map has only one positive Lyapunov exponent and it is very small. From Figure 3b, it can be seen that the 2D-SLG map has two positive Lyapunov exponents when $b = -0.5$, $c = 26.9$ and a varies from 0 to 20, but the exponents of this system are smaller than the exponent values of the new 2D map. Figure 3c shows the variation of the Lyapunov exponents with respect to the variation of parameter a and $b = 40$. Figure 3d shows the variation of the Lyapunov exponents with respect to the variation of parameters b and $a = 30$. From the results in Figure 3, it can be seen that the proposed system has two positive Lyapunov exponents, thus confirming that the system is hyperchaotic. Moreover, the parameter range for the hyperchaotic phenomena is very large. The above results indicate that the two-dimensional hyperchaotic system proposed in this paper has better chaotic characteristics.

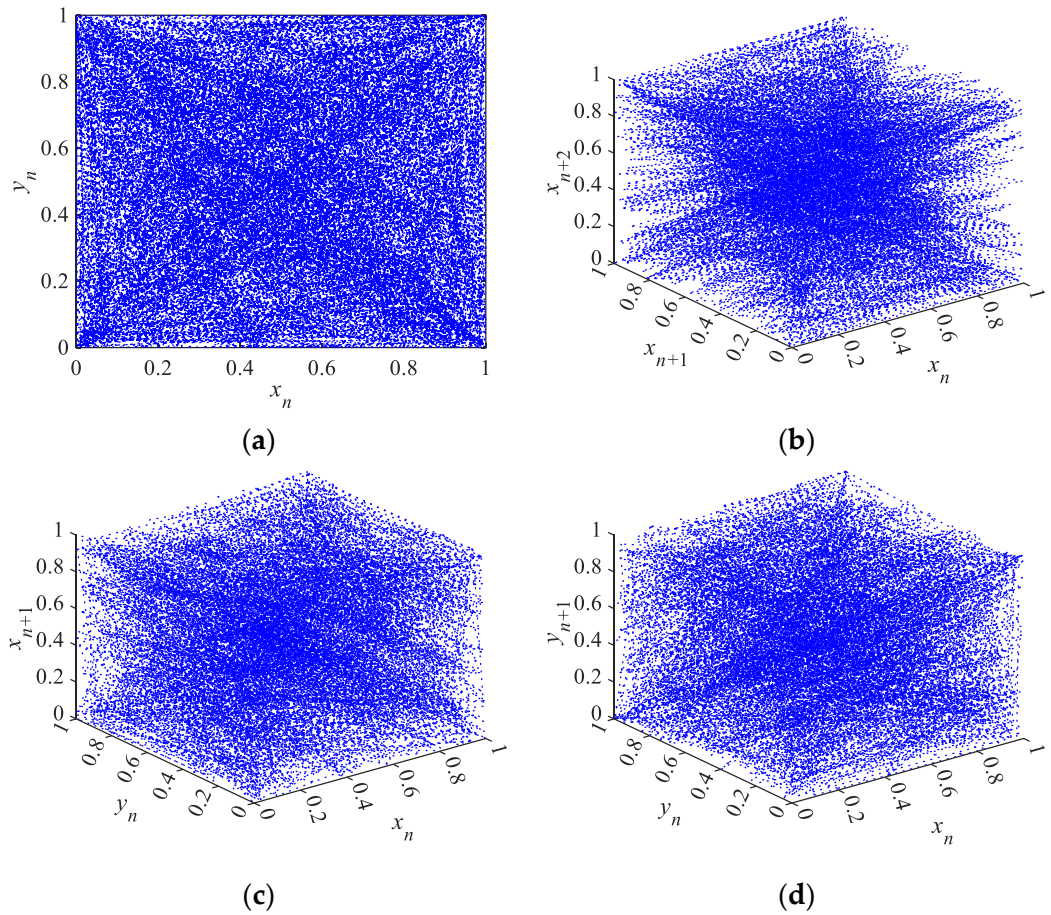


Figure 2. Phase diagrams of system (1) with control parameters $a = 30$ and $b = 20$. (**a**) The relationship between the state variables x_n and y_n; (**b**) the relationship between the three neighboring variables x_n, x_{n+1} and x_{n+2}; (**c**) The relationship between x_{n+1} and x_n, y_n; (**d**) the relationship between y_{n+1} and x_n, y_n.

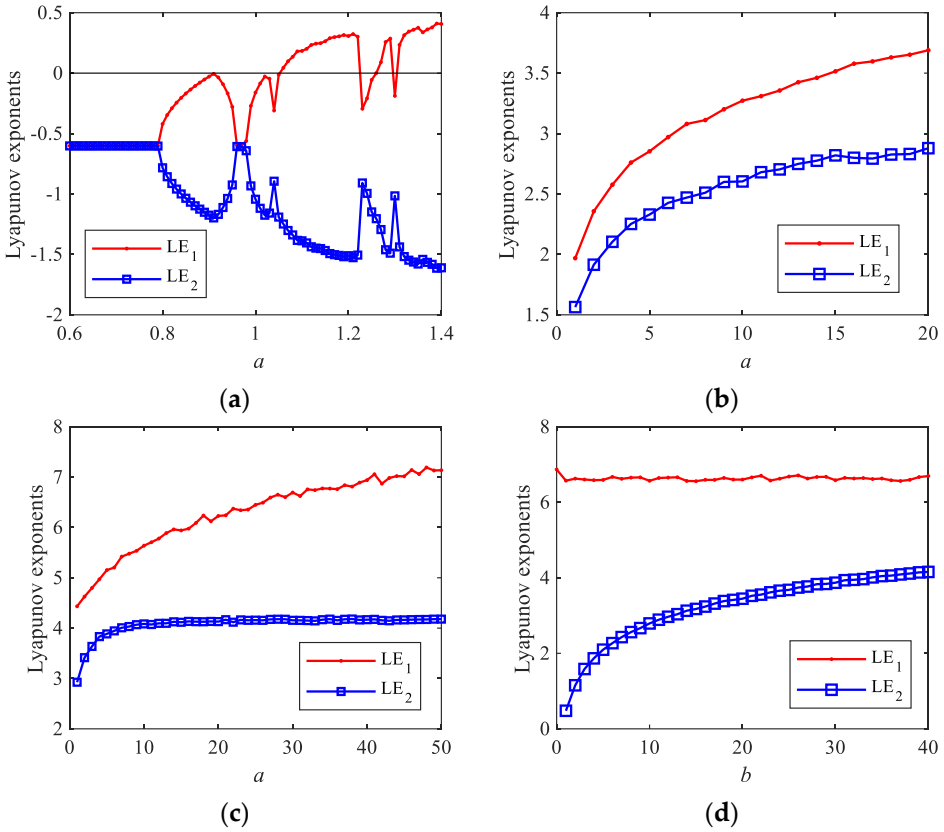


Figure 3. Lyapunov exponent graph of three 2D chaotic maps: (**a**) Lyapunov exponent of Hénon map; (**b**) Lyapunov exponent of 2D-SLG map; (**c**) Lyapunov exponent of the proposed map versus parameter a; (**d**) Lyapunov exponent of the proposed map versus parameter b.

2.4. Correlation Analysis

The correlation coefficient can describe the correlation of time series. The autocorrelation of time series with good randomness should have a form similar to the δ function, and the cross-correlation of time series with good randomness should be zero. The autocorrelation coefficient at lag k of a series $\{x(i), i = 1, 2, \ldots, N\}$ can be calculated as follows:

$$autocorr(k) = \frac{\sum_{i=1}^{N-|k|} (x(i) - \overline{x})(x(i + |k|) - \overline{x})}{\sum_{i=1}^{N-|k|} (x(i) - \overline{x})^2} \tag{3}$$

where $\overline{x}$ is the average value of the series $\{x(i)\}$.

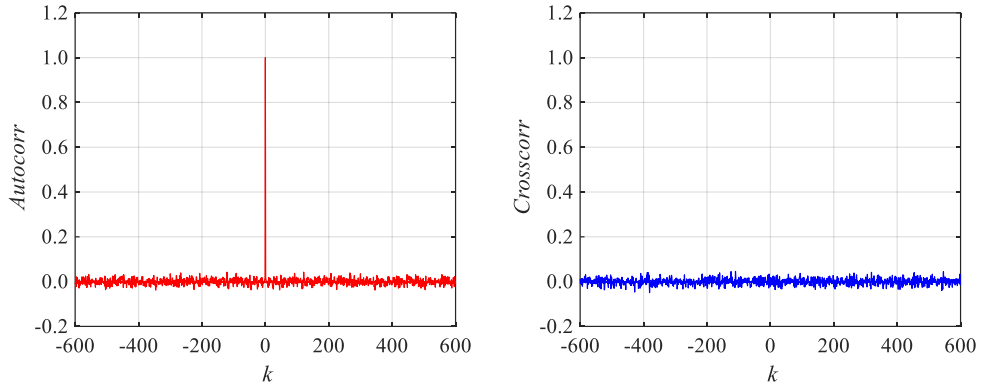
The cross-correlation of the two series $\{x(i)\}$ and $\{y(i)\}$ of length N at lag k is defined as follows:

$$crosscorr(k) = \frac{\sum_{i=1}^{N-|k|} (x(i) - \overline{x})(y(i + |k|) - \overline{y})}{\sqrt{\sum_{i=1}^{N-|k|} (x(i) - \overline{x})^2} \sqrt{\sum_{i=1}^{N-|k|} (y(i) - \overline{y})^2}} \tag{4}$$

where $\overline{x}$ and $\overline{y}$ are the average values of the series $\{x(i)\}$ and $\{y(i)\}$, respectively.

For system (1), the autocorrelation coefficient curve of the chaotic sequence $\{x(i)\}$ generated using the system parameters $a = 30$, $b = 40$ and the initial state value $(x_0, y_0) = (0.321, 0.987)$ is shown in Figure 4a, and the cross-correlation coefficient curve of the two chaotic sequences $\{x(i)\}$ and $\{y(i)\}$ generated using $a = 30$, $b = 40$ and $(x_0, y_0) = (0.321, 0.987)$ is shown in Figure 4b. Figure 4 shows that the time series generated

by the system has good randomness, and that its autocorrelation and cross-correlation coefficients meet the requirements of random sequences.



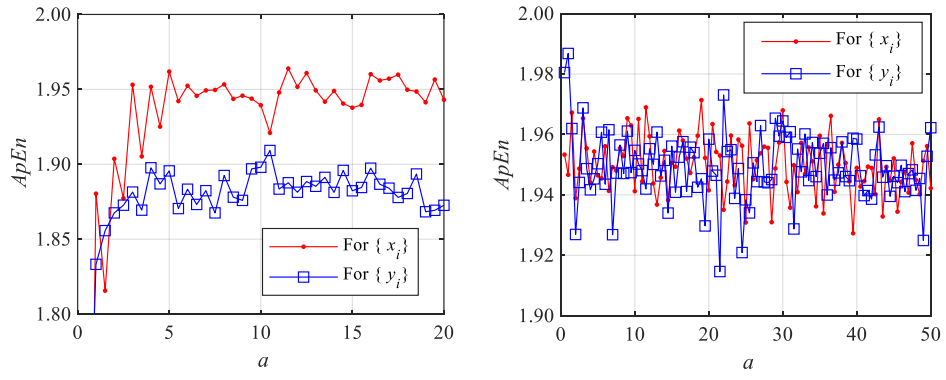
(**a**) (**b**)

Figure 4. The correlation of system (1): (**a**) autocorrelation function; (**b**) cross-correlation function.

2.5. Approximate Entropy Analysis

The approximate entropy (*ApEn*) is another important indicator used to measure the complexity of time series. The *ApEn* describes the probability of generating new patterns in a sequence with the increase in the embedding dimension. For a detailed description of the algorithm for calculating the approximate entropy of time series, please refer to [25].

Figure 5 shows the *ApEn* values of the chaotic systems. Figure 5a plots the *ApEn* of the 2D-SLG chaotic map when $b = -0.5$, $c = 26.9$ and a varies from 0 to 20. Figure 5b plots the *ApEn* of the proposed new hyperchaotic map when $b = 40$ and a varies from 0 to 50. From Figure 5, one can see that system (1) has larger approximate entropy values and a larger parameter change range $a > 0$. For the proposed 2D system, the average *ApEn* value of $\{x_i\}$ is 1.950, and the average *ApEn* value of $\{y_i\}$ is 1.9493, while for the 2D-SLG chaotic map for the proposed 2D system, the average *ApEn* value of $\{x_i\}$ is 1.9273, and the average *ApEn* value of $\{y_i\}$ is 1.8782. It can be seen that the proposed 2D hyperchaotic system has a better chaotic performance.



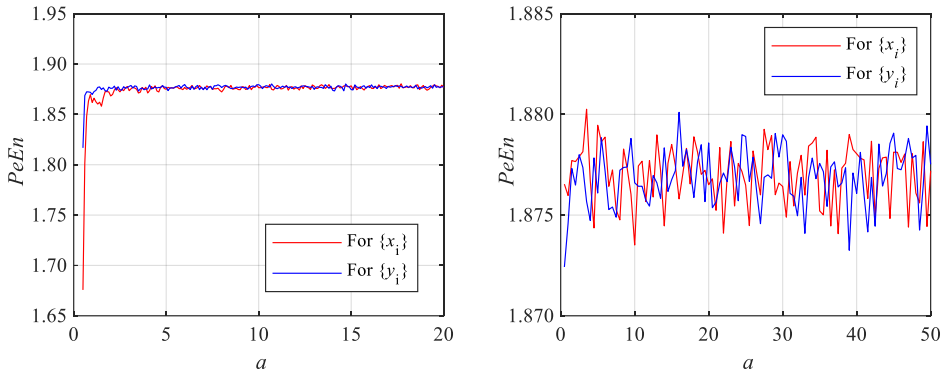
(**a**) (**b**)

Figure 5. The values of approximate entropy of the systems with $b = 40$: (**a**) approximate entropy of the 2D-SLG chaotic map; (**b**) approximate entropy of the proposed 2D hyperchaotic map.

2.6. Permutation Entropy Analysis

Permutation entropy (*PeEn*) is another important method used for analyzing the complexity of time series, which can verify the degree to which the chaotic sequence generated by the proposed mapping approaches randomness. More complex sequences have a larger *PeEn*, which is more applicable in image encryption. Figure 6 shows the *PeEn* volatility of the chaotic maps under parameter changes. Figure 6a shows the *PeEn* volatility

of the 2D-SLG system with the system parameters $b = -0.5$ and $c = 26.9$. Figure 6b shows the *PeEn* volatility of the proposed system, where the system parameter b is fixed to 40. The *PeEn* of the proposed 2D system fluctuates stably within the region [1.870, 1.880], with an average value of 1.8770 for the **x** sequence and 1.8769 for the **y** sequence, indicating that the mapping has good and stable sequence complexity. The *PeEn* of the 2D-SLG chaotic map fluctuates stably within the region [1.6757, 1.8804], with an average value of 1.8742 for the **x** sequence and 1.8764 for the **y** sequence. The results indicate that the average *PeEn* value of the 2D hyperchaotic system in this paper is slightly higher than the average *PeEn* value of the 2D-SLG system, once again proving that the proposed new system in this paper has a better chaotic performance.



(**a**) (**b**)

Figure 6. Permutation entropy of the systems with $b = 40$: (**a**) permutation entropy of the 2D-SLG chaotic map; (**b**) permutation entropy of the proposed 2D hyperchaotic map.

3. The Proposed Pseudo-Random Number Generator and Its Performance Test

Random number generators (RNG) include true random number generators (TRNG) and pseudo-random number generators (PRNG). TRNGs are based on physical processes, while PRNGs generate sequences that are computed from an initial seed value. A common requirement of PRNGs is that they possess good statistical properties, meaning their output approximates a sequence of true random numbers [26]. In this section, we describe the proposed pseudo-random number generator, which uses the proposed 2D discrete hyperchaotic map, based on the IEEE 754 double-precision floating-point number standard.

3.1. Algorithm of the Pseudo-Random Number Generator

The proposed pseudo-random number generator (PRNG) uses parameters a, b, x_0 and y_0 of the hyperchaotic map. The pseudo-random number sequences are binary bit sequences, which facilitates testing the randomness of the sequence using NIST software. The other relevant parameters are as follows: m represents the number of groups in the binary sequence (its recommended value is 1000), M represents the total length of the binary sequence composed of all group sequences and L represents the length of the chaotic sequence that should be generated. The algorithm of the proposed pseudo-random number generator is described as follows.

Step 1: Initialize parameters a, b, x_0, y_0 and $m = 1000$. $M = m \times 10^6$, $L = M/8$.

Step 2: The chaotic system (1) is iterated L times to generate chaotic sequences X and Y with a length of L.

Step 3: Store each double-precision real number in a chaotic sequence using the IEEE 754 double-precision floating-point number standard and form a 64-bit binary string using the binary numbers in memory in ascending order. Divide the 64-bit binary string into eight equal groups, converting each group into an unsigned integer. In this way, each real number in the chaotic sequence is converted into eight unsigned integers. The chaotic real number sequence with a length of L is transformed into an integer sequence with a length of $8L$, say, IntX = $\{IntX(i), i = 1, 2, \ldots, 8L\}$, $IntX(i) \in \{0, 1, 2, \ldots, 255\}$.

Step 4: Take the fifth integer from the group of integers corresponding to the real number of each chaotic sequence to form a new integer sequence. This operation is equivalent to starting from the fifth number in the sequence of IntX and sampling in steps of eight to obtain a sub-sequence of length L, say, **xb** = $\{xb(i), i = 1, 2, \ldots, L\}$ = IntX (5:8:end), $xb(i) \in \{0, 1, 2, \ldots, 255\}$.

Step 5: Save the sequence **xb** as a binary file f.bin. The binary sequence in the file f.bin is the output result of the pseudo-random number generator.

The details of the algorithm are shown in Algorithm 1.

Algorithm 1 Generating the chaotic secret key streams

Input: The security keys (x_0, y_0, a, b) and the number of groups in the binary sequence m.
1: Initialization: Set values of (x_0, y_0, a, b), m = 1000, $L = m \times 10^6/8$
2: [X, Y] = $HCS(x_0, y_0, a, b, L)$; //$HCS()$ is a function to generate chaotic sequences
3: $Intx$ = typecast(swapbytes(X),'uint8'); //Get the integer sequence $Intx$
4: xb = $Intx$(5:8:end); //Obtain the sub-sequence xb of length L
5: FID = fopen('D:\NIST\f.bin','w'); fwrite(FID,xb,'uint8'); //Save the sequence xb to a file
Output: The binary sequence in the file f.bin of the PRNG

3.2. NIST SP800-22 Test of the PRNG

The NIST (National Institute of Standards and Technology, USA) SP800-22 is a standard test software package to evaluate the randomness performance of time series. It requires multiple sequences (recommend 1000 sequences) to be tested, and the length of each sequence is 1,000,000 bits. There are two performance indicators, the p-value and pass rate, that are employed to measure the stochastic performance of time series. The default significant level $\alpha = 0.01$. The confidence interval that is used to test the pass rate is defined as $1 - \alpha \pm 3\sqrt{\alpha(1-\alpha)/m}$, where m is the number of groups of bit sequences. When $\alpha = 0.01$ and $m = 1000$, the confidence interval is $1 - 0.01 \pm 3\sqrt{0.01 \times 0.99/1000}$ = [0.9806, 0.9994], which indicates that the minimum passing rate must be 98.06%.

To test the stochastic performance of sequences generated by the proposed PRNG based on the hyperchaotic map, we generated 1000 chaotic real number sequences, each with a length of 1,000,000/8 real numbers. The parameters are set as $a = 40$, $b = 30$, $x_0 = 0.134$ and $y_0 = 0.987$. We generated a binary bit sequence using the chaotic sequence X of the algorithm in Section 3.1, and then tested the randomness of the binary sequence using the NIST software package. The NIST statistical test results are shown in Table 1. From the test results, it can be seen that each p-value is larger than 0.0,1 and the minimum p-value is 0.016488. Each pass rate is larger than 98.06%, and the minimum pass rate for each statistical test is 98.1%.

Table 1. NIST statistical test results for the PRNG.

The Statistical Test Item Name	p-Value	Pass Rate	Results
Frequency (Monobit)	0.660012	993/1000 = 99.3%	Pass
Block Frequency (m = 128)	0.417219	993/1000 = 99.3%	Pass
Cumulative Sums (Forward)	0.08151	994/1000 = 99.4%	Pass
Cumulative Sums (Reverse)	0.089301	990/1000 = 99.0%	Pass
Runs	0.263572	993/1000 = 99.3%	Pass
Longest Run of Ones	0.348869	991/1000 = 99.1%	Pass
Rank	0.769527	992/1000 = 99.2%	Pass
FFT	0.263572	993/1000 = 99.3%	Pass
Non-Overlapping Templates *	0.016488	989/1000 = 98.9%	Pass
Overlapping Templates	0.825505	989/1000 = 98.9%	Pass
Universal	0.643366	991/1000 = 99.1%	Pass
Approximate Entropy	0.496351	990/1000 = 99.0%	Pass
Random Excursions *	0.331257	623/627 = 99.4%	Pass
Random Excursions Variant *	0.080439	623/627 = 99.4%	Pass
Serial Test 1	0.213309	990/100 = 99.0%	Pass
Serial Test 2	0.181557	990/100 = 99.0%	Pass
Linear Complexity	0.19692	991/100 = 99.1%	Pass

* Note: The non-overlapping template, random excursions and random excursions variant contain 148, 8 and 18 sub-tests, respectively, and the results listed in Table 1 are the worst results among the multiple sub-tests (i.e., the result with the lowest corresponding p-value).

4. The Proposed Color Image Encryption and Decryption Scheme

The color image encryption scheme consists of the following three procedures: generating chaotic secret key streams, the first round of cross-channel parallel permutation and diffusion (CCPPD) and the second round of cross-channel parallel permutation and diffusion. Figure 7 shows the encryption structure of the proposed color image encryption scheme. The detailed descriptions of each step are given in Sections 4.1–4.3 below.

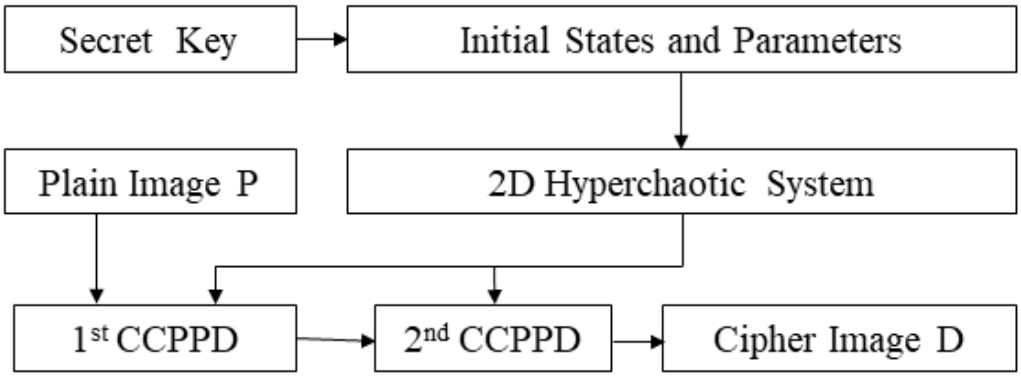


Figure 7. A block diagram of the overall image encryption scheme.

4.1. Chaotic Secret Key Streams Generation

The secret key of the encryption scheme consists of four double-precision floating-point numbers of (x_0, y_0, a, b) and an 8-bit unsigned integer c_0. Each double-precision floating-point number contains 64 bits, and the 8-bit unsigned integer c_0 has 8 bits. Therefore, the secret key has a length of 264 bits of binary data. The structure of the secret key is shown in Figure 8, where $0 < x_0 < 1$, $0 < y_0 < 1$, $a > 0$, $b > 0$ and $c_0 \in \{0, 1, 2, \ldots, 255\}$. The four double-precision real numbers $\{x_0, y_0, a, b\}$ are used as the initial state values and system parameters of the 2D hyperchaotic system. Chaotic secret key streams are generated by using the above five secret key parameters, and the steps of generating chaotic secret key streams are as follows.

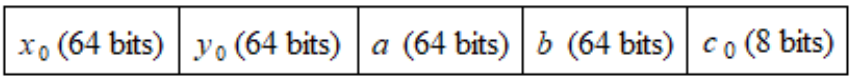


Figure 8. The structure of the secret key.

Step 1: Set the initial secret key parameters of $\{x_0, y_0, a, b, c_0\}$. Obtain the pixel row number M and column number N of one color channel in the color image to be encrypted, and calculate $L = M \times N$.

Step 2: Iterate system (1) for $(L + 500)$ times with initial state $\{x_0, y_0\}$ and system parameters $\{a, b\}$ to generate two chaotic real number sequences X and Y. Remove the first 500 state values from the sequence, and obtain the final chaotic sequences X and Y using the length of L.

Step 3: Generate two integer sequences **r** and **s** by sorting sequences **X** and **Y**. Here, **r** will be a shuffled arrangement of the integer sequence $\{1, 2, \ldots, M\}$, satisfying the condition $r(i) \neq r(j)$ if $i \neq j$, and **s** will be a shuffled arrangement of the integer sequence $\{1, 2, \ldots, N\}$, satisfying the condition $s(i) \neq s(j)$ if $i \neq j$.

Step 4: Generate a chaotic integer secret key matrix **K** by **X**. Here, **K** will be an $M \times N$ matrix, and $K(i, j) \in \{0, 1, 2, \ldots, 255\}$, $i = 1, 2, \ldots, M$, $j = 1, 2, \ldots, N$.

Step 5: Generate a 3D chaotic integer secret key matrix **t** by **Y**. Here, **t** will be an $M \times N \times 3$ matrix, and $t(i, j, k) \in \{1, 2, 3\}$, $i = 1, 2, \ldots, M$, $j = 1, 2, \ldots, N$, $k = 1, 2, 3$.

The details of the algorithm are shown in Algorithm 2.

Algorithm 2 Generating the chaotic secret key streams

Input: The row number M, column number N, and security keys (x_0, y_0, a, b).
1: Initialization: **r** = 1:M; **s** = 1:N; **K** = zeros (M, N); **t** = zeros (M, N, 3).
2: [X, Y] = $HCS(x_0, y_0, a, b, M \times N)$; //$HCS$() is a function to generate chaotic sequences.
3: [~, r] = *sort* (X(1:M)); //*sort*() is a function to sort the sequence X (1:M)
4: [~, **s**] = *sort* (Y(1:N)); //*sort*() is a function to sort the sequence Y (1:N)
5: **cs** = reshape (**X**, M, N); **K** = mod(floor(**cs** $\times$ 10^6), 256);
6: **cs** = reshape (Y, M, N); **t**(:, :, 1) = mod(floor(**cs** $\times$ 10^6), 3) + 1;
7: **t** (:, :, 2) = mod (t (:, :, 1) + 1, 3) + 1; **t** (:, :, 3) = mod (t (:, :, 2) + 1, 3) + 1;
Output: The chaotic secret key matrix: **r**, **s**, **K**, **t**.

4.2. The First Round of Cross-Channel Parallel Permutation and Diffusion

The core idea of CCPPD is to change the positions and values of the pixels simultaneously, and the change in position is across color channels. The implementation of cross-color channel permutation is based on the three integer matrices **r**, **s** and **t** generated previously. Specifically, the position is changed according to the following corresponding relationship: the pixels at the i-th row and j-th column of the k-th color channel in the encrypted image come from the pixels at the $r(i)$ row, $s(j)$ column and $t(i, j, k)$ color channel of the image to be encrypted. Suppose that the original plaintext image is represented by P, and the ciphertext image obtained from the first round of encryption is represented by **C**. Figure 9 depicts the specific details of this idea by using a small $3 \times 4 \times 3$ color image. The general permutation relationship is as follows: $C(i, j, k) \leftarrow P(r(i), s(j), t(i, j, \mathrm{k}))$.

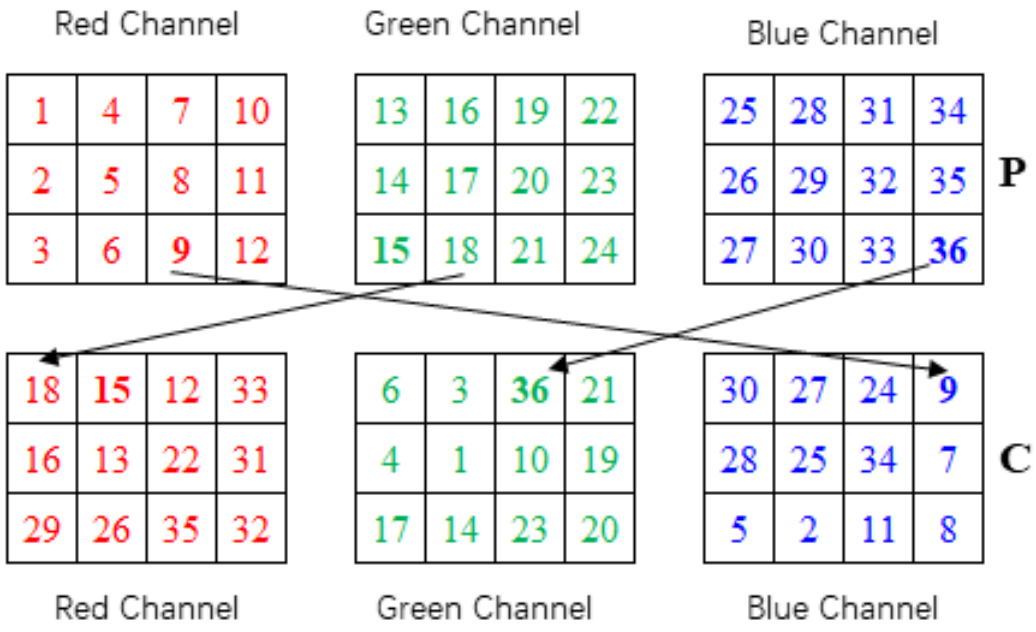


Figure 9. The diagrammatic sketch of cross-channel permutation.

The specific values of **r**, **s** and **t** in Figure 9 are **r** = [3, 1, 2], **s** = [2, 1, 4, 3] and

$$t(:,:,1) = \begin{bmatrix} 2 & 2 & 1 & 3 \\ 2 & 2 & 2 & 3 \\ 3 & 3 & 3 & 3 \end{bmatrix}, \; t(:,:,2) = \begin{bmatrix} 1 & 1 & 3 & 2 \\ 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix}, \; t(:,:,3) = \begin{bmatrix} 3 & 3 & 2 & 1 \\ 3 & 3 & 3 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

For example, $C(1, 1, 1) = 18 \leftarrow P(r(1), s(1), t(1, 1, 1)) = P(3, 2, 2) = 18$; $C(1, 3, 2) = 36 \leftarrow P(r(1), s(3), t(1, 3, 2)) = P(3, 4, 3) = 36$; $C(1, 4, 3) = 9 \leftarrow P(r(1), s(4), t(1, 4, 3)) = P(3, 3, 1) = 9$.

In our proposed color image encryption scheme, the key operational statements of the algorithm that simultaneously implement the pixel position cross-channel permutation

and pixel value transformation, and introduce the diffusion mechanism, are presented in Equation (5) as follows:

$$C(i,j,k) = mod(P(r(i),s(j),t(i,j,k)) + K(i,j) + Pre,\ 256) \tag{5}$$

where $i = 1, 2, \ldots, M$; $j = 1, 2, \ldots, N$; $k = 1, 2, 3$; Pre represents the value of a previously encrypted pixel. When encrypting the first pixel value, the value of Pre is initialized to be equal to c_0. For a detailed description of the first round of the cross-channel parallel permutation and diffusion process, please refer to Algorithm 3.

Algorithm 3 The first round of CCPPD

Input: The plain image **P**, c_0, and the chaotic secret key matrix of {**r**, **s**, **K**, **t**}.
1: Initialization: [M, N, ~] = size(**P**); **C** = zeros(M, N, 3); Pre = c_0.
2: **for** k = 1 to 3 **do**
3: **for** j = 1 to N **do**
4: **for** i = 1 to M **do**
5: $C(i, j, k)$ = mod($P(r(i), s(j), t(i, j, k))$ + $K(i, j)$ + Pre, 256);
6: Pre = $C(i, j, k)$;
7: **end for**
8: **end for**
9: **end for**
Output: The intermediate cipher image **C**.

4.3. The Second Round of Cross-Channel Parallel Permutation and Diffusion

In the second round of the cross-channel parallel permutation and diffusion process, the input image is the intermediate cipher image **C**, which is the output in the first round of encryption processing. The output image is the final encrypted image, denoted by **D**. The key operational statement of the second round of cross-channel parallel permutation and diffusion is

$$D(i,j,k) = mod(C(r(i),s(j),t(i,j,k)) + K(i,j) + Pre,\ 256) \tag{6}$$

where $i = 1, 2, \ldots, M$; $j = 1, 2, \ldots, N$; $k = 1, 2, 3$; Pre represents the value of a previously encrypted pixel. When encrypting the first pixel value in the second round of encryption, the value of Pre is equal to the last encrypted pixel value, that is, $C(M, N, 3)$. For a detailed description of the second round of cross-channel parallel permutation and diffusion process, please refer to Algorithm 4.

Algorithm 4 The second round of CCPPD

Input: The intermediate cipher image **C** and the chaotic key matrix of {**r**, **s**, **K**, **t**}.
1: Initialization: [M, N, ~] = size(**C**); **D** = zeros(M, N, 3); Pre = $C(M, N, 3)$.
2: **for** k = 1 to 3 **do**
3: **for** j = 1 to N **do**
4: **for** i = 1 to M **do**
5: $D(i, j, k)$ = mod($C(r(i), s(j), t(i, j, k))$ + $K(i, j)$ + Pre, 256);
6: Pre = $D(i, j, k)$;
7: **end for**
8: **end for**
9: **end for**
Output: The final cipher image **D**.

4.4. The Decryption Process

The operational steps of the decryption process are the reverse ones of the encryption process. The decryption operation also includes the following three stages: (1) generating the chaotic key matrices, (2) the first round of decryption operation and (3) the second

round of decryption operation. The operation of generating the chaotic key matrices is identical to the encryption process. The first round of decryption operation is to solve the intermediate ciphertext image **C** from the final ciphertext image **D**. The main difficulty is to find the first *P*re value, and the core decryption operation is the inverse operation of the corresponding encryption process. The detailed steps of the first round of decryption operation are shown in Algorithm 5.

Algorithm 5 The first round of decryption operation

Input: The final cipher image **D** and the chaotic key matrix of {**r**, **s**, **K**, **t**}.
1: Initialization: [M, N, ~] = size(**D**); **C** = zeros(M, N, 3); Pre = C(M, N, 3).
2: i = find(**r** = = **M**); //Find the index of the element has value M in matrix **r**
3: j = find(**s** = = **N**); //Find the index of the element has value N in matrix **s**
4: k = find(t(i, j, :) = = 3); //Find the 3rd index of the element t(i, j, :)
5: if (i >1) Then Pre = D(i−1, j, k);
6: if (i = = 1) & (j > 1) Then Pre = D(M, j−1, k);
7: if (i = = 1) & (j = = 1) Then Pre = D(M, N, k−1);
8: C(r(i),s(j),t(i, j, k)) = mod(D(i, j, k) − K(i, j) − Pre, 256);
9: Pre = C(r(i), s(j), t(i, j, k));
10: **for** k = 1 to 3 **do**
11: **for** j = 1 to N **do**
12: **for** i = 1 to M **do**
13: C(r(i), s(j), t(i, j, k)) = mod(D(i, j, k) − K(i, j) − Pre, 256);
14: Pre = D(i, j, k);
15: **end for**
16: **end for**
17: **end for**
Output: The intermediate cipher image **C**.

In order to better understand steps 5, 6 and 7 of Algorithm 5, further detailed explanations are provided as follows. We know that the *P*re obtained in the second round of encryption is D(i, j, k), where i, j and k in C(r(i), s(j), t(i, j, k)) satisfy r(i) = M, s(j) = N and t(i, j, k) = 3. Steps 1, 2 and 3 find {i, j, k}, which satisfy r(i) = M, s(j) = N and t(i, j, k) = 3. The order of encrypting pixels in this algorithm is from the R color channel (k = 1) to the G color channel (k = 2) and to the B color channel (k = 3). And each color channel is in the first column. The meaning of Step 5 is that if the last encrypted pixel does not belong to the first row but is after the first row, then the previous encrypted pixel value D is in the position of the previous row in the same color channel and column, so the previous ciphertext pixel of D(i, j, k) is D(i−1, j, k). The meaning of Step 6 is that if the last encrypted pixel is at the first row but is after the first column, then the previous encrypted pixel value D is at the position of the previous column and the last row in the same color channel, so the previous ciphertext pixel of D(i, j, k) is D(M, j−1, k). The meaning of Step 7 is that if the last encrypted pixel is at the first row and the first column, then the previous encrypted pixel value D is at the position of the last row and the last column in the previous color channel, so the previous ciphertext pixel of D(i, j, k) is D(M, N, k−1).

The detailed steps of the second round of decryption operation are shown in Algorithm 6.

Algorithm 6 The second round of decryption operation

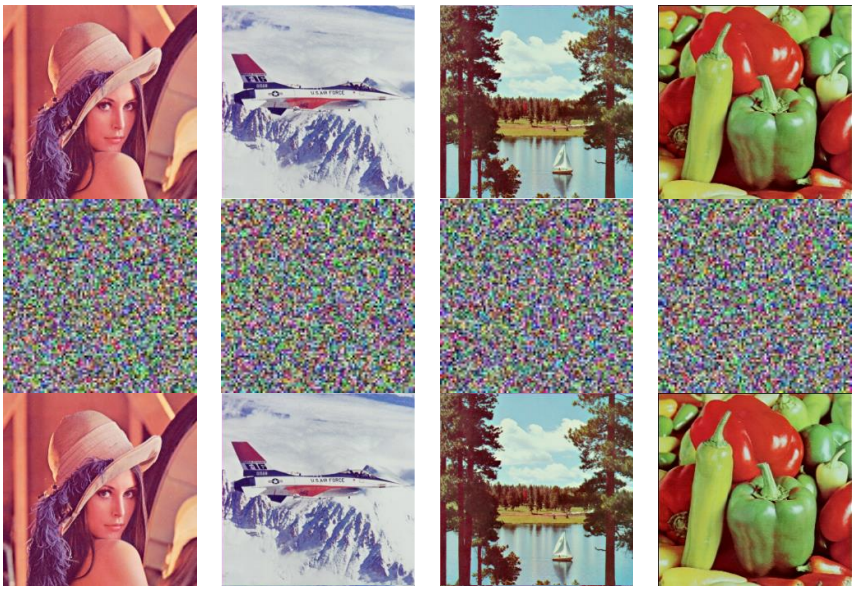
Input: The intermediate cipher image **C**, c_0, and the secret key matrix of {**r**, **s**, **K**, **t**}.
1: Initialization: [M, N, ~] = size(**C**); **P** = zeros(M, N, 3); $Pre = c_0$.
2: **for** k = 1 to 3 **do**
3: **for** j = 1 to N **do**
4: **for** i = 1 to M **do**
5: $P(r(i), s(j), t(i, j, k)) = \mathrm{mod}(C(i, j, k) - K(i, j) - Pre, 256)$;
6: $Pre = C(i, j, k)$;
7: **end for**
8: **end for**
9: **end for**
Output: The restored decrypted image **P**.

5. Experimental Results and Security Analysis

We used MATLAB 2022b to verify the proposed color image encryption algorithm on a PC with an Intel(R) Core i7-9700 @ 3.00 GHz CPU and 16.0 GB memory. The test images were obtained from USC-SIPI. The secret parameters are set as $a = 30$, $b = 40$, $x_0 = 0.134$, $y_0 = 0.987$ and $c_0 = 66$. The encryption results of several test images are shown in Figure 10.

5.1. Key Space Analysis

The performance of a cryptographic system against brute force attacks depends on the size of its key space. According to the current computing speed of the computer, when the key space of a cryptographic system is greater than 2^{100}, the cryptographic system has security against brute force attacks [27]. In the proposed cryptosystem, the length of the key is 264 bits, so its key space is approximately 2^{264}, which is much larger than 2^{100}. Therefore, the key space of the proposed scheme is large enough to effectively resist brute force attacks.



(**a**) (**b**) (**c**) (**d**)

Figure 10. The test images and the encrypted results. (**a**) The plain image, cipher image and decrypted image of 4.2.04.tif. (**b**) The plain image, cipher image and decrypted image of 4.2.05.tif. (**c**) The plain image, cipher image and decrypted image of 4.2.06.tif. (**d**) The plain image, cipher image and decrypted image of 4.2.07.tif.

5.2. Histogram Analysis

The distribution of pixel values in an image can be intuitively expressed through histograms; therefore, a histogram analysis is often one of the most intuitive indicators to

measure the distribution of pixel values in an image. Therefore, for excellent encryption algorithms, the histogram of a cipher image should have a uniform distribution pattern so that attackers find it difficult to obtain useful statistical information from the histogram distribution to analyze the cryptographic system. Figure 11 shows the two images we used to demonstrate the histogram pattern of the cipher image obtained via the proposed encryption algorithm. The results show that the histograms of the cipher images are relatively flat, so our cryptographic system has a strong resistance to histogram attacks.

5.3. Pixel Correlation Analysis

Usually, the pixel distribution of a meaningful natural image has continuity of values in its position, characterized by adjacent pixels that have very close pixel values. This property is called a strong correlation between adjacent pixels. A good image encryption algorithm should make the adjacent pixels of the ciphertext image irrelevant or weakly correlated. In order to quantitatively describe the correlation strength of the adjacent pixels in an image, the Pearson correlation coefficient (P_{cc}) is usually used as an indicator. The P_{cc} between the pixel group **x** and **y** can be calculated using

$$P_{cc} = \frac{\sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \overline{x})^2 \sum_{i=1}^{N}(y_i - \overline{y})^2}} \tag{7}$$

where $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$ represent the pixel value of a group of adjacent pixels in the image, and N is the total number of samples. $\overline{x}$ and $\overline{y}$ are the average values of $\{x_i\}$ and $\{y_i\}$, respectively. We selected the neighboring pixels of different images before and after encryption to calculate their P_{cc} values, and the results are shown in Table 2. Figure 12 shows the neighboring pixel distribution of the image Pepper before and after encryption using the proposed encryption algorithm.

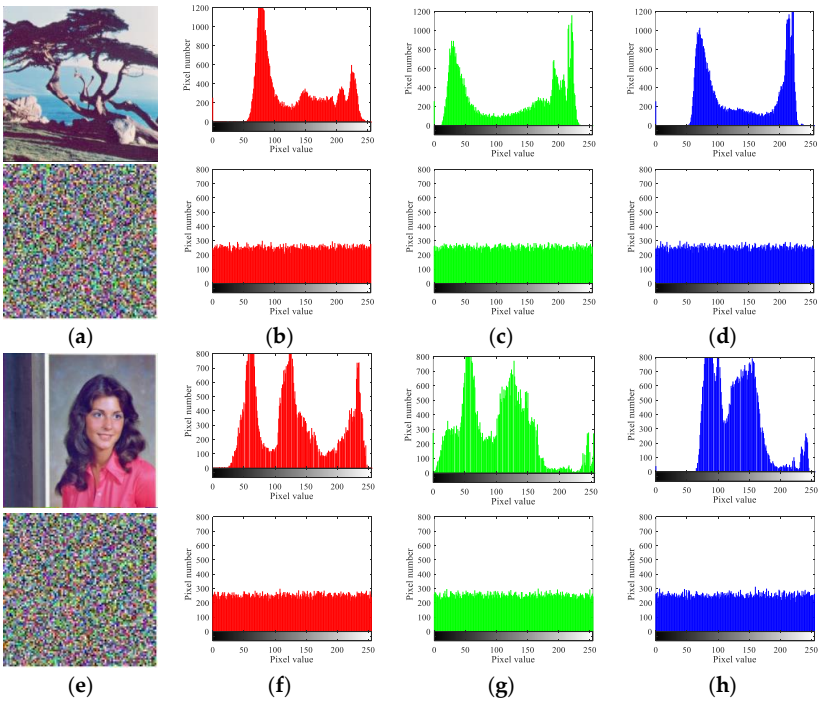


Figure 11. Plain/cipher images and their histograms. (**a**) Plain and cipher image of 4.1.06. (**b**) Histogram of R channel in plain and cipher image of 4.1.06. (**c**) Histogram of G channel in plain and cipher image of 4.1.06. (**d**) Histogram of B channel in plain and cipher image of 4.1.06. (**e**) Plain and cipher image of 4.1.04. (**f**) Histogram of R channel in plain and cipher image of 4.1.04. (**g**) Histogram of G channel in plain and cipher image of 4.1.04. (**h**) Histogram of B channel in plain and cipher image of 4.1.04.

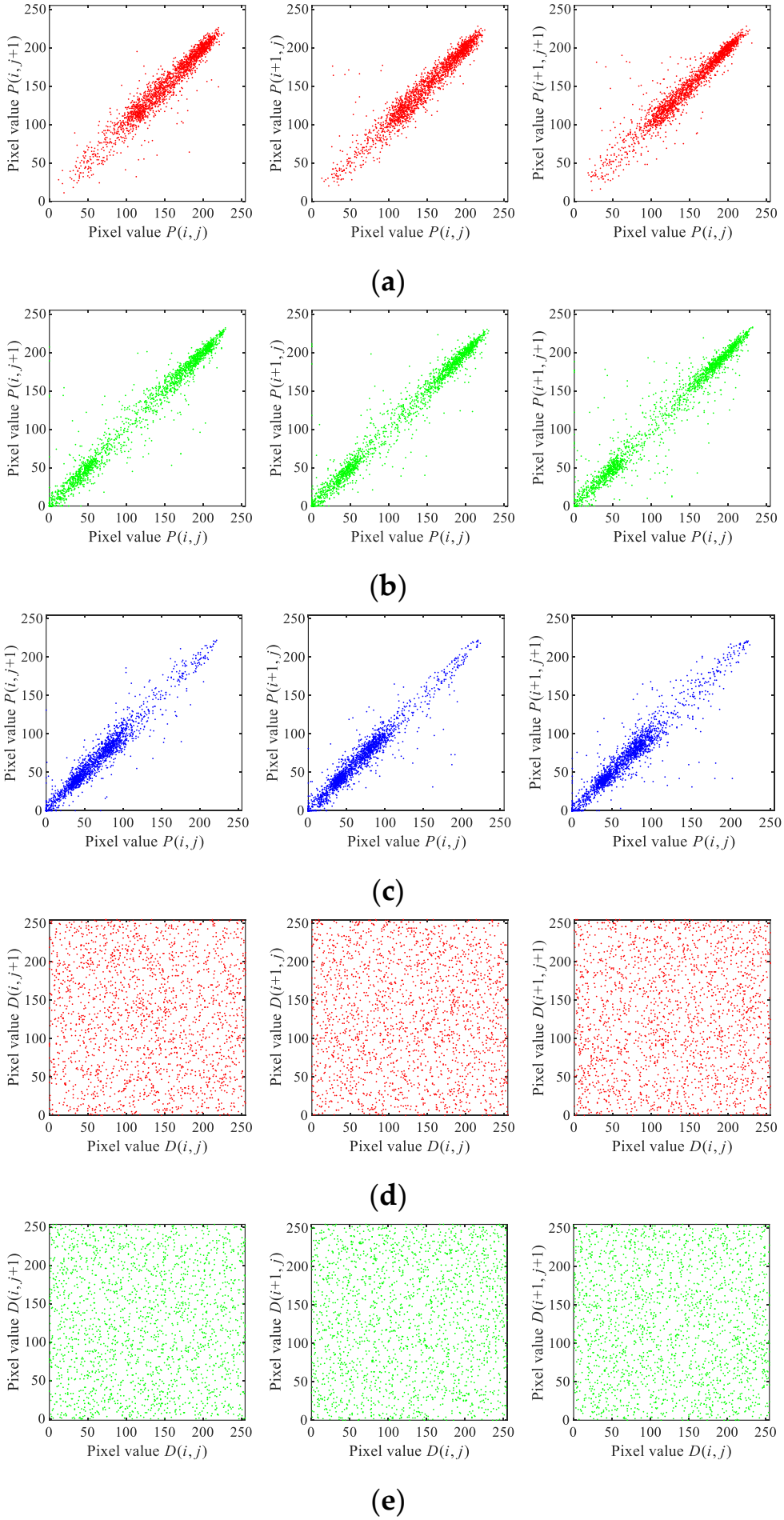
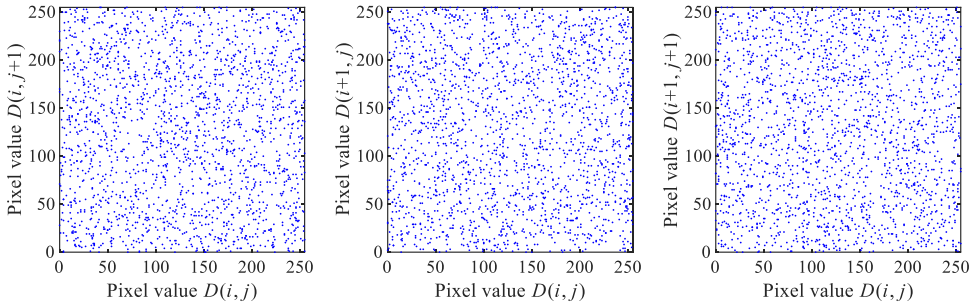


Figure 12. *Cont.*



(f)

Figure 12. Correlation test results of adjacent pixels. (**a**) R channel of Pepper in horizontal, vertical and diagonal directions; (**b**) G channel of Pepper in horizontal, vertical and diagonal directions; (**c**) B channel of Pepper in horizontal, vertical and diagonal directions; (**d**) R channel of encrypted Pepper in horizontal, vertical and diagonal directions; (**e**) G channel of encrypted Pepper in horizontal, vertical and diagonal directions; (**f**) B channel of encrypted Pepper in horizontal, vertical and diagonal directions.

Table 2. Correlation coefficients of cipher images encrypted using different algorithms.

Algorithm	Image Name	Direction	R Channel	G Channel	B Channel
This work	2.1.04/Oakland	Horizontal	0.0027	−0.0004	0.0025
This work	2.1.04/Oakland	Vertical	0.0011	**0.0014**	**0.0007**
This work	2.1.04/Oakland	Diagonal	**0.0012**	**0.0005**	−0.0009
Ref. [28]	2.1.04/Oakland	Horizontal	**−0.0012**	**0.0003**	**0.0014**
Ref. [28]	2.1.04/Oakland	Vertical	**0.0009**	0.0030	−0.0027
Ref. [28]	2.1.04/Oakland	Diagonal	−0.0014	−0.0021	−0.0009
This work	2.2.01	Horizontal	**0.0001**	−0.0007	$\mathbf{-8.7595 \times 10^{-5}}$
This work	2.2.01	Vertical	0.0003	0.0013	**0.0005**
This work	2.2.01	Diagonal	$\mathbf{7.979 \times 10^{-5}}$	0.0009	−0.0010
Ref. [29]	2.2.01	Horizontal	−0.0003	**0.0006**	0.0019
Ref. [29]	2.2.01	Vertical	0.0003	**0.0007**	0.0025
Ref. [29]	2.2.01	Diagonal	−0.004	**0.0002**	**0.0003**
This work	4.2.04	Horizontal	−0.0027	$\mathbf{-4.6744 \times 10^{-5}}$	0.0043
This work	4.2.04	Vertical	0.0029	−0.0009	$\mathbf{-5.6333 \times 10^{-5}}$
This work	4.2.04	Diagonal	0.0018	−0.0012	0.0015
Ref. [29]	4.2.04	Horizontal	**0.0006**	**−0.0004**	**0.0001**
Ref. [29]	4.2.04	Vertical	**−0.0012**	−0.0007	0.0005
Ref. [29]	4.2.04	Diagonal	**0.0008**	**0.0007**	**0.0006**
This work	4.2.05	Horizontal	**−0.00062639**	**0.0018238**	**−0.0010242**
This work	4.2.05	Vertical	**0.00017306**	**0.00065355**	**−0.0004688**
This work	4.2.05	Diagonal	0.0026312	**0.00047658**	**−0.0013144**
Ref. [15]	4.2.05	Horizontal	0.0017	0.0026	0.0017
Ref. [15]	4.2.05	Vertical	0.0019	0.0019	0.0015
Ref. [15]	4.2.05	Diagonal	**0.0017**	0.0021	0.0020
This work	4.2.03	Horizontal	**−0.00011509**	**0.00056796**	0.0013133
This work	4.2.03	Vertical	0.0017407	−0.00075841	−0.0016187
This work	4.2.03	Diagonal	**0.00054237**	0.001598	**−0.000471**
Ref. [29]	4.2.03	Horizontal	−0.0003	0.0008	**0.0008**
Ref. [29]	4.2.03	Vertical	**−0.0002**	**−0.0002**	**0.0008**
Ref. [29]	4.2.03	Diagonal	−0.0011	**0.0005**	0.0025
Ref. [15]	4.2.03	Horizontal	0.0015	0.0018	0.0023
Ref. [15]	4.2.03	Vertical	0.0026	0.0031	0.0021
Ref. [15]	4.2.03	Diagonal	0.0017	0.0022	0.0016
This work	4.2.07	Horizontal	0.0031349	−0.0038061	0.0024694
This work	4.2.07	Vertical	**−0.0013014**	**0.0012009**	0.0025589
This work	4.2.07	Diagonal	$\mathbf{-2.7096 \times 10^{-5}}$	**0.0011369**	−0.0030308
Ref. [15]	4.2.07	Horizontal	**0.0017**	**0.0016**	**0.0017**
Ref. [15]	4.2.07	Vertical	0.0022	0.0016	**0.0022**
Ref. [15]	4.2.07	Diagonal	0.0020	0.0026	**0.0022**

The above test results indicate that the P_{cc} values of images fluctuate in different directions. The P_{cc} values of the proposed algorithm are relatively smaller than those of the comparator. The average value is closer to 0, which means that the distribution of adjacent pixels in the encrypted image is random. The proposed algorithm can more effectively reduce the correlation between adjacent pixels, indicating its stronger ability to resist statistical attacks.

5.4. Information Entropy Analysis

Information entropy can measure the randomness or uncertainty of an information source. Greater information entropy means that the more random or uncertain the information source is, the more difficult it will be to predict or decipher. The information entropy of an information source can be calculated using the following formula:

$$H(S) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{8}$$

where $S = \{s_1, s_2, \ldots, s_n\}$ represents the information source, and p_i represents the probability of s_i occurrence. For the information source of an 8-bit image, there are 256 gray levels, and n = 256. Therefore, the maximum information entropy that can be reached by the 8-bit image is $\log_2 256 = 8$. The information entropy of various standard test images encrypted using this algorithm and some recently published algorithms are listed in Table 3. The results show that the information entropy of ciphertext images is very close to 8. Compared with other algorithms, the images encrypted using the proposed algorithm have a larger information entropy in many cases.

Table 3. Information entropy of encrypted images for several different algorithms.

Image Name	Channels	Ours	Ref. [15]	Ref. [30]	Ref. [31]	Ref. [29]
2.2.01/San Diego	R	7.9998	\	\	\	7.9998
2.2.01/San Diego	G	7.9998	\	\	\	7.9998
2.2.01/San Diego	B	7.9998	\	\	\	7.9998
4.2.03/Baboon	R	7.9993	7.9993	\	7.9992	7.9992
4.2.03/Baboon	G	7.9993	7.9994	\	7.9993	7.9993
4.2.03/Baboon	B	7.9993	7.9993	\	7.9993	7.9991
4.2.04/Lena	R	7.9994	\	7.9994	7.9993	7.9976
4.2.04/Lena	G	7.9994	\	7.9994	7.9994	7.9973
4.2.04/Lena	B	7.9993	\	7.9994	7.9993	7.9971
4.2.05/Airplane	R	7.9993	7.9993	7.9993	7.9992	\
4.2.05/Airplane	G	7.9994	7.9993	7.9992	7.9993	\
4.2.05/Airplane	B	7.9994	7.9992	7.9993	7.9993	\
4.2.07/Peppers	R	7.9993	7.9993	7.9993	7.9993	7.9991
4.2.07/Peppers	G	7.9992	7.9993	7.9994	7.9993	7.9992
4.2.07/Peppers	B	7.9993	7.9993	7.9993	7.9993	7.9992

5.5. Sensitivity Analysis

Sensitivity includes the sensitivity of encrypted images to key changes and plaintext changes, which, respectively, reflect the degree to which the algorithm's actual key space approaches the theoretical key space and its ability to resist differential attacks.

5.5.1. Secret Key Sensitivity

Whether a cryptographic system is sensitive to keys is related to whether the actual total number of keys available can reach the theoretical value of the key space. Testing whether a cryptographic system is sensitive to keys can usually be performed separately during the encryption and decryption processes.

During the encryption process, if two slightly different keys are used to encrypt the same image, resulting in two completely different ciphertext images, it can be verified from the encryption process that the cryptographic algorithm is sensitive to the key. It is more intuitive to use data to reflect the difference between two ciphertext images. The mean squared error (MSE) can reflect the difference between two images, which can be calculated using

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} [D_K(i,j) - D_{K'}(i,j)]^2 \tag{9}$$

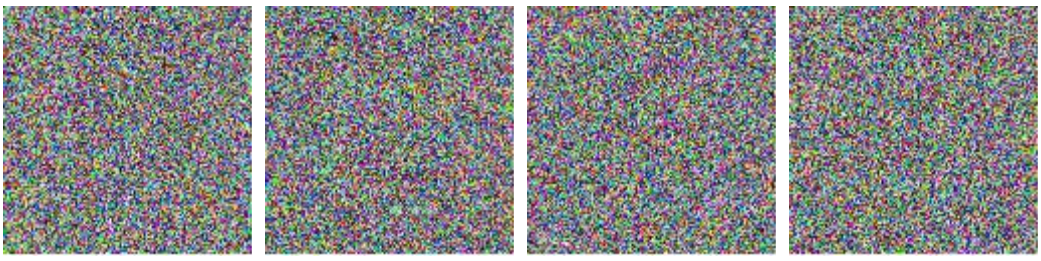
where $D_K(i,j)$ is the pixel value of the cipher image encrypted using the secret key K, $D_{K'}(i,j)$ is the pixel value of the cipher image encrypted using the secret key K', M is the row number and N is the column number. Table 4 lists the mean squared error (MSE) results of the ciphertext obtained by encrypting the same image Airplane using different keys.

Where the original key is of $K = \{a = 30, b = 40; x_0 = 0.134, y_0 = 0.987\}$, MSE_1 corresponds to the key of $K_1 = \{a = 30 + 10^{-14}, b = 40; x_0 = 0.134, y_0 = 0.987\}$, MSE_2 corresponds to the key of $K_2 = \{a = 30, b = 40 + 10^{-14}; x_0 = 0.134, y_0 = 0.987\}$, MSE_3 corresponds to the key of $K_3 = \{a = 30, b = 40; x_0 = 0.134 + 10^{-14}, y_0 = 0.987\}$ and MSE_4 corresponds to the key of $K_4 = \{a = 30, b = 40; x_0 = 0.134, y_0 = 0.987 + 10^{-14}\}$. The results indicate that the algorithm is sensitive to keys.

Table 4. The *MSE* values of encrypted images using slightly different keys.

Channels	MSE_1	MSE_2	MSE_3	MSE_4
R	10,937	10,920	10,951	10,929
G	10,966	10,870	10,955	10,967
B	10,911	10,940	10,878	10,907

During the decryption process, if there is a slight difference between the actual key used in the decryption process and the right key, and there is a significant difference between the decrypted image and the original plaintext image, the cryptographic system is said to be sensitive to the key. The more sensitive the cryptosystem is to keys, the greater the difficulty for attackers to crack the cryptosystem by using exhaustive secret keys, and the stronger the security of the cryptosystem. In order to test the sensitivity of the proposed algorithm to secret keys, we used Baboon images as the test image, and decrypted its encrypted images using another set of slightly changed keys. Each change of key only changed the initial value or one of the system parameters of the chaotic system by 10^{-14}. The decrypted images are shown in Figure 13a,b,c,d, respectively. From these decryption results, it can be seen that all the decrypted images are still unrecognizable and meaningless images similar to noise signals. This indicates that the small differences in keys lead to incorrect decryption. Therefore, the algorithm is very sensitive to secret keys.



(**a**) (**b**) (**c**) (**d**)

Figure 13. Baboon test results of key sensitivity in decryption. (**a**) Modified x_0 with 10^{-14}, (**b**) Modified y_0 with 10^{-14}, (**c**) Modified a with 10^{-14}, (**d**) Modified b with 10^{-14}.

5.5.2. Plain Image Sensitivity

Differential analysis is a chosen-plaintext attack. Its basic idea is to obtain the largest key possible by analyzing the influence of a specific plaintext difference on the corresponding ciphertext difference. In order for the cryptographic system to resist differential attacks, encryption algorithms must be very sensitive to changes in plaintext. The commonly used indicators for measuring the sensitivity of algorithms to plaintext are the NPCR (number of pixel change rate) and UACI (unified average change of intensity). The formulas for calculating the NPCR and UACI are as follows:

$$D(i,j) = \begin{cases} 1, & if\ C(i,j) \neq C'(i,j) \\ 0, & if\ C(i,j) = C'^{(i,j)} \end{cases} \tag{10}$$

$$\text{NPCR} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} D(i,j) \times 100\% \tag{11}$$

$$\text{UACI} = \frac{1}{M \times N}\left(\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{\left|C(i,j) - C'^{(i,j)}\right|}{255}\right) \times 100\% \quad (12)$$

where M and N denote the row and column numbers of the image. The ideal values of the NPCR and UACI are 99.6094% and 33.4635%, respectively. The larger the NPCR is and the closer the UACI is to the center of the expected value range (i.e., 33.4636%), the better the anti-difference performance of the algorithm is. We adopted the image Lena ($512 \times 512 \times 3$) as the test image in the experiment, and selected the pixel at the middle position point (i.e., P(512,512,2)) to increase its pixel value by 1. Table 5 lists the NPCR and UACI results obtained using our algorithm and several published algorithms. Compared with the results provided by the literature in the table, our scheme NPCR and UACI are both close to the ideal values of 99.6094% and 33.4635%, and the NPCR values of our scheme have a certain competitive advantage. Therefore, the proposed scheme performs better in resisting differential attacks.

Table 5. NPCR and UACI test results of different image encryption schemes.

Algorithms	NPCR (%)			UACI (%)		
	R	**G**	**B**	**R**	**G**	**B**
Ours	99.6136	99.6059	99.6143	33.4562	33.4078	33.4610
Ref. [30]	99.6167	99.6046	99.6158	33.4395	33.4587	33.4566
Ref. [32]	99.6109	99.6208	99.6067	33.4782	33.4580	33.4228
Ref. [29]	99.6140	99.6017	99.6140	33.5627	33.5218	33.4339

5.6. Robustness Analysis

Images inevitably suffer from noise pollution or data loss during channel transmission. If the encryption algorithm is not robust, the decryption process cannot obtain recognizable plaintext information. A good encryption algorithm should be able to resist noise pollution or data loss attacks. Figure 14 shows the decryption result of the Lena cipher image after being polluted by noise, where (a1), (a2), (a3) and (a4) are noisy cipher images polluted by salt-and-pepper noise (SPN) with densities = 0.005 and 0.05, and Gaussian noise (GN) with densities = 0.0005 and 0.005, respectively. (b1), (b2), (b3) and (b4) are decrypted images of (a1), (a2), (a3) and (a4), respectively. Figure 15 shows the decryption results of the Lena cipher images after data loss, with (a1), (a2), (a3) and (a4) being cropped by 32×32, 64×64, 128×128 and 256×256. (b1), (b2), (b3) and (b4) are the decrypted images of (a1), (a2), (a3) and (a4), respectively. It can be seen that the cipher images that underwent noise pollution or cropping processing can still obtain recognizable decrypted images. Therefore, the proposed encryption algorithm has a strong resistance to noise and cropping attacks.

5.7. Security Analysis for Classical Attacks

According to the intensity of attacks, there are four classical types of attacks, namely, ciphertext-only attack, known plaintext attacks, chosen-plaintext attack, and chosen-ciphertext attack. Chosen-plaintext and chosen-ciphertext attacks are the most powerful attacks. If a cryptosystem can resist these two attacks, it can resist other types of attacks. For chosen-plaintext and chosen-ciphertext attacks, the commonly used methods are to obtain the corresponding ciphertext (plaintext) image by selecting a special plaintext (ciphertext) image through an encryption machine (decryption machine) in order to crack the intermediate equivalent key of the cryptographic system.

At present, most image encryption algorithms separate the permutation process and diffusion process, which provides convenience for attackers to crack the equivalent key of the diffusion process and the permutation process step by step and greatly reduces the difficulty of cracking. The equivalent keys of the proposed cryptosystem in this paper include sequences **r**, **s**, **t** and **K**. The proposed cryptosystem combines permutation and

diffusion processing in parallel, and there are two encryption rounds. Therefore, attackers cannot individually crack some of the keys in a step-by-step manner, but can only break all {**r**, **s**, **t**, **K**} keys at the same time. The difficulty is equivalent to an exhaustive attack on the ciphers set of {**r**, **s**, **t**, **K**}. For images of size $M \times N$, the sequence **r** has $M!$ possible forms, and the sequence **s** has $N!$ possible forms. There are three possible forms; sequence t has $3^{M \times N \times 3}$ possible forms, and sequence K has $256^{M \times N}$ possible forms, so the equivalent key space size is $(M!) \times (N!) \times (3^{M \times N \times 3}) \times (256^{M \times N})$. Such a large key space is sufficient to resist exhaustive attacks.

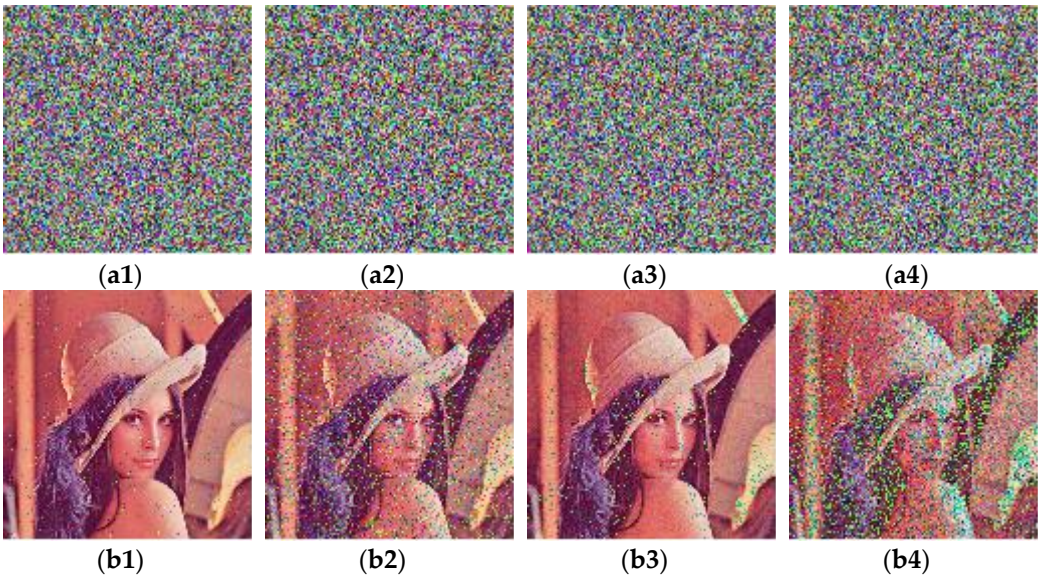


Figure 14. The resistance to noise attack: (**a1**) noisy images via SPN with density = 0.005; (**a2**) noisy images via SPN with density = 0.05; (**a3**) noisy images via GN with density = 0.0005; (**a4**) noisy images via GN with density = 0.005; (**b1**) decrypted image of (**a1**); (**b2**) decrypted image of (**a2**); (**b3**) decrypted image of (**a3**); (**b4**) decrypted image of (**a4**).

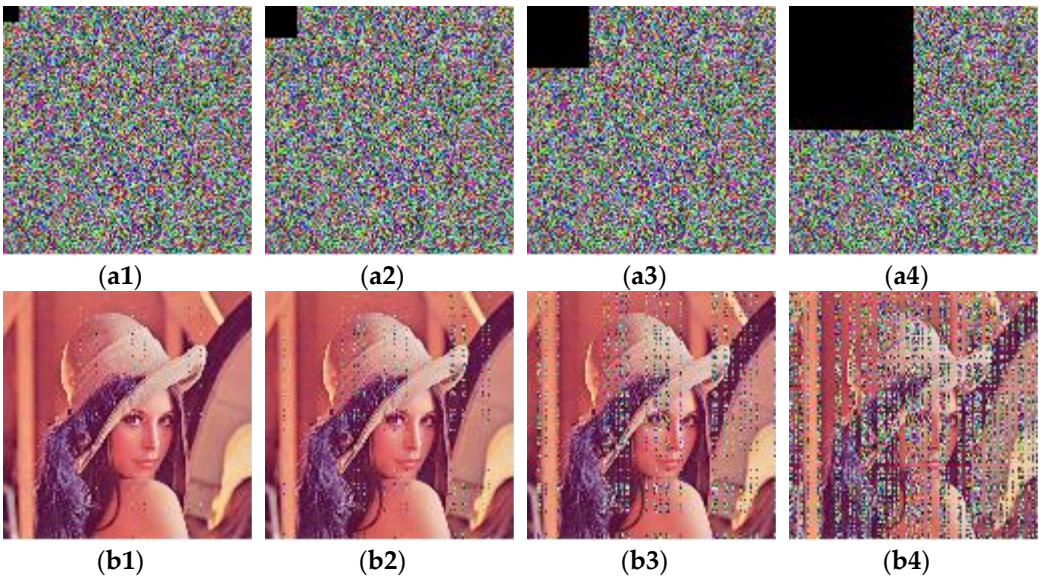


Figure 15. The resistance to cut attack: (**a1**) clipping image with size of 32 × 32; (**a2**) clipping image with size of 64 × 64; (**a3**) clipping image with size of 128 × 128; (**a4**) clipping image with size of 256 × 256; (**b1**) decrypted image of (**a1**); (**b2**) decrypted image of (**a2**); (**b3**) decrypted image of (**a3**); (**b4**) decrypted image of (**a4**).

5.8. Time Complexity Analysis

In addition to security, the speed of encryption and decryption is also an important aspect of measuring the performance of image encryption algorithms. Table 6 lists the time required to encrypt and decrypt different images using our algorithm.

Table 6. Running time of different images.

Images	Size	Encryption Time (s)	Decryption Time (s)
House	$256 \times 256 \times 3$	0.0602	0.0157
Airplane	$512 \times 512 \times 3$	0.0854	0.0539
Peppers	$512 \times 512 \times 3$	0.0862	0.0555
San Diego	$1024 \times 1024 \times 3$	0.3498	0.2088

Table 7 compares our algorithm with other algorithms in Lena ($512 \times 512 \times 3$). The running time of image encryption was compared. From the values in the table, it can be seen that our encryption algorithm has excellent running speed compared to other encryption algorithms.

Table 7. Running time of different algorithms for $512 \times 512 \times 3$ Lena image.

Algorithms	Encryption Time (s)	Decryption Time (s)
Ours	0.1027126	0.0614000
Ref. [29]	0.9220030	0.8414570
Ref. [33]	1.7961050	0.8475750
Ref. [34]	1.3053220	1.0264940
Ref. [35]	1.4933000	7.8065000

6. Conclusions

In this paper, a new two-dimensional discrete hyperchaotic system is proposed. The system has very complex hyperchaotic properties and can produce highly random chaotic sequences, which are confirmed via bifurcation diagrams, chaotic attractor, Lyapunov exponents, correlation analysis, approximate entropy and permutation entropy. As an application, a pseudo-random number generator (PRNG) and an efficient color image encryption algorithm were designed based on the hyperchaotic system. The bit sequence generated by the proposed PRNG can completely pass all the NIST test items. The proposed color image encryption algorithm can implement cross-channel scrambling and pixel value diffusion encryption in parallel, which not only greatly improves the encryption speed of color images, but also improves the security level of cipher images via cross-channel simultaneous scrambling and the diffusion of plain images. The proposed algorithm is robust to differential attacks, statistical attacks and interference attacks, and its overall performance is better than some existing algorithms. Chaos-based image encryption is an interesting and meaningful research topic.

For future works, we will further optimize the encryption structure to improve security and efficiency, including further optimization of the scrambling strategy design and the diffusion operation design.

Author Contributions: Conceptualization, S.Z. and C.Z.; methodology, W.Z. and X.D.; software, S.Z.; validation, S.Z., W.Z., X.D. and C.Z.; formal analysis, S.Z.; investigation, C.Z.; resources, W.Z.; data curation, S.Z.; writing—original draft preparation, S.Z.; writing—review and editing, C.Z.; visualization, X.D.; supervision, X.D. and W.Z.; project administration, W.Z.; funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 62172441), the Local Science and Technology Developing Foundation guided by the Central Government (free exploration project 2021Szvup166), the Xinjiang Uygur Autonomous Region

Key R&D Project (2021B01002) and the Doctoral Research Fund Project of Xinjiang University, China (202112120001).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Jain, A.K.; Sahoo, S.R.; Kaubiyal, J. Online social networks security and privacy: Comprehensive review and analysis. *Complex Intell. Syst.* **2021**, *7*, 2157–2177. [CrossRef]
2. Zhang, Q.; Han, J. A novel color image encryption algorithm based on image hashing, 6D hyperchaotic and DNA coding. *Multimed. Tools Appl.* **2021**, *80*, 13841–13864. [CrossRef]
3. Moafimadani, S.S.; Chen, Y.; Tang, C. A New Algorithm for Medical Color Images Encryption Using Chaotic Systems. *Entropy* **2019**, *21*, 577. [CrossRef] [PubMed]
4. Dou, Y.; Li, M. Cryptanalysis of a New Color Image Encryption Using Combination of the 1D Chaotic Map. *Appl. Sci.* **2020**, *10*, 2187. [CrossRef]
5. Zhang, X.; Yan, X. Adaptive Chaotic Image Encryption Algorithm Based on RNA and Pixel Depth. *Electronics* **2021**, *10*, 1770. [CrossRef]
6. Zhang, S.; Liu, L.; Xiang, H. A Novel Plain-Text Related Image Encryption Algorithm Based on LB Compound Chaotic Map. *Mathematics* **2021**, *9*, 2778. [CrossRef]
7. Zhao, R.; Zhang, Y.; Nan, Y.; Wen, W.; Chai, X.; Lan, R. Primitively visually meaningful image encryption: A new paradigm. *Inf. Sci.* **2022**, *613*, 628–648. [CrossRef]
8. Shi, Y.; Hu, Y.; Wang, B. Image Encryption Scheme Based on Multiscale Block Compressed Sensing and Markov Model. *Entropy* **2021**, *23*, 1297. [CrossRef]
9. Bao, W.; Zhu, C. A secure and robust image encryption algorithm based on compressive sensing and DNA coding. *Multimed. Tools Appl.* **2022**, *81*, 15977–15996. [CrossRef]
10. Liang, Q.; Zhu, C. A new one-dimensional chaotic map for image encryption scheme based on random DNA coding. *Opt. Laser Technol.* **2023**, *160*, 109033. [CrossRef]
11. Zhang, B.; Liu, L. Chaos-Based Image Encryption Review, Application, and Challenges. *Mathematics* **2023**, *11*, 2585. [CrossRef]
12. Peng, J.; Zhu, C.; Jiang, D. A new 2D-ASC chaotic system and its image encryption applications. *Mod. Phys. Lett. B* **2023**, *37*, 2350009. [CrossRef]
13. Ali, W.; Zhu, C.; Latif, R.; Asim, M.; Tariq, M.U. Image Encryption Scheme Based on Orbital Shift Pixels Shuffling with ILM Chaotic System. *Entropy* **2023**, *25*, 787. [CrossRef] [PubMed]
14. Elghandour, A.; Salah, A.; Karawia, A. A new cryptographic algorithm via a two-dimensional chaotic map. *Ain Shams Eng. J.* **2022**, *13*, 101489. [CrossRef]
15. Yang, S.; Tong, X.; Wang, Z.; Zhang, M. Efficient color image encryption algorithm based on 2D coupled chaos and multi-objective optimized S-box. *Phys. Scr.* **2022**, *97*, 045204. [CrossRef]
16. Lai, Q.; Liu, Y. A cross-channel color image encryption algorithm using two-dimensional hyperchaotic map. *Expert Syst. Appl.* **2023**, *223*, 119923. [CrossRef]
17. Lai, Q.; Wan, Z.; Zhang, H.; Chen, G. Design and Analysis of Multiscroll Memristive Hopfield Neural Network With Adjustable Memductance and Application to Image Encryption. In *IEEE Transactions on Neural Networks and Learning Systems*; IEEE: Piscataway, NJ, USA, 2022; pp. 1–14. [CrossRef]
18. Zhou, S.; Qiu, Y.; Wang, X.; Zhang, Y. Novel image cryptosystem based on new 2D hyperchaotic map and dynamical chaotic S-box. *Nonlinear Dynam.* **2023**, *111*, 9571–9589. [CrossRef]
19. Zhu, S.; Deng, X.; Zhang, W.; Zhu, C. Secure image encryption scheme based on a new robust chaotic map and strong S-box. *Math. Comput. Simul.* **2023**, *207*, 322–346. [CrossRef]
20. Sun, K.H.; He, S.B.; Yin, L.Z.; Li-Kun, A. Application of FuzzyEn algorithm to the analysis of complexity of chaotic sequence. *Acta Phys. Sin.* **2012**, *61*, 130507.
21. Sun, K.H.; He, S.B.; He, Y.; Yin, L.Z. Complexity analysis of chaotic pseudo-random sequences based on spectral entropy algorithm. *Acta Phys. Sin.* **2013**, *62*, 010501. [CrossRef]
22. Zhu, S.; Deng, X.; Zhang, W.; Zhu, C. A New One-Dimensional Compound Chaotic System and Its Application in High-Speed Image Encryption. *Appl. Sci.* **2021**, *11*, 11206. [CrossRef]
23. Erkan, U.; Toktas, A.; Toktas, F.; Alenezi, F. 2D eπ-map for image encryption. *Inf. Sci.* **2022**, *589*, 770–789. [CrossRef]
24. He, J.; Yu, S.; Cai, J. Numerical Analysis and Improved Algorithms for Lyapunov-Exponent Calculation of Discrete-Time Chaotic Systems. *Int. J. Bifurc. Chaos* **2016**, *26*, 1650219. [CrossRef]
25. Zhu, S.; Deng, X.; Zhang, W.; Zhu, C. Image Encryption Scheme Based on Newly Designed Chaotic Map and Parallel DNA Coding. *Mathematics* **2023**, *11*, 231. [CrossRef]

26. Paar, C.; Pelzl, J. *Understanding Cryptography—A Textbook for Students and Practitioner*; Springer: Berlin/Heidelberg, Germany, 2010.
27. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
28. Hu, Y.; Wu, H.; Zhou, L. Color image encryption base on a 2D hyperchaotic enhanced Henon map and cross diffusion. *Alex. Eng. J.* **2023**, *73*, 385–402. [CrossRef]
29. Xin, J.; Hu, H.; Zheng, J. 3D variable-structure chaotic system and its application in color image encryption with new Rubik's Cube-like permutation. *Nonlinear Dynam.* **2023**, *111*, 7859–7882. [CrossRef]
30. Sha, Y.; Sun, B.; Cheng, X.; Mou, J.; Wang, L. Cross-plane colour image encryption scheme based on BST model and chaotic map. *Eur. Phys. J. Spec. Top.* **2022**, *231*, 3249–3263. [CrossRef]
31. Khalil, N.; Sarhan, A.; Alshewimy, M.A.M. An efficient color/grayscale image encryption scheme based on hybrid chaotic maps. *Opt. Laser Technol.* **2021**, *143*, 107326. [CrossRef]
32. Wang, S.; Peng, Q.; Du, B. Chaotic color image encryption based on 4D chaotic maps and DNA sequence. *Opt. Laser Technol.* **2022**, *148*, 107753. [CrossRef]
33. Teng, L.; Wang, X.; Yang, F.; Xian, Y. Color image encryption based on cross 2D hyperchaotic map using combined cycle shift scrambling and selecting diffusion. *Nonlinear Dynam.* **2021**, *105*, 1859–1876. [CrossRef]
34. Qiu, H.; Xu, X.; Jiang, Z.; Sun, K.; Xiao, C. A color image encryption algorithm based on hyperchaotic map and Rubik's Cube scrambling. *Nonlinear Dynam.* **2022**, *110*, 2869–2887. [CrossRef]
35. Ahmad, I.; Shin, S. A novel hybrid image encryption-compression scheme by combining chaos theory and number theory. *Signal Process.-Image Commun.* **2021**, *98*, 116418. [CrossRef]