entropy

MDPI

Article

A Novel Five-Dimensional Three-Leaf Chaotic Attractor and Its Application in Image Encryption

Tao Wang [1,2], Liwen Song [3], Minghui Wang [1], Shiqiang Chen [2] and Zhiben Zhuang [3,*]

[1] College of Computer Science, Sichuan University, Chengdu 610000, China; 2000012@hbmzu.edu.cn (T.W.); wangminghui@scu.edu.cn (M.W.)
[2] School of Advanced Materials and Mechatronic Engineering, Hubei Mizu University, Enshi 445000, China; 1997013@hbmzu.edu.cn
[3] School of Science, Hubei Minzu University, Enshi 445000, China; 201930081@hbmy.edu.cn
* Correspondence: zbxzzb185898@163.com; Tel.: +86-15871181531

Received: 14 January 2020; Accepted: 19 February 2020; Published: 21 February 2020

Abstract: This paper presents a novel five-dimensional three-leaf chaotic attractor and its application in image encryption. First, a new five-dimensional three-leaf chaotic system is proposed. Some basic dynamics of the chaotic system were analyzed theoretically and numerically, such as the equilibrium point, dissipative, bifurcation diagram, plane phase diagram, and three-dimensional phase diagram. Simultaneously, an analog circuit was designed to implement the chaotic attractor. The circuit simulation experiment results were consistent with the numerical simulation experiment results. Second, a convolution kernel was used to process the five chaotic sequences, respectively, and the plaintext image matrix was divided according to the row and column proportions. Lastly, each of the divided plaintext images was scrambled with five chaotic sequences that were convolved to obtain the final encrypted image. The theoretical analysis and simulation results demonstrated that the key space of the algorithm was larger than 10^{150} that had strong key sensitivity. It effectively resisted the attacks of statistical analysis and gray value analysis, and had a good encryption effect on the encryption of digital images.

Keywords: five-dimensional chaotic system; digital image encryption; proportional block; convolution kernel

1. Introduction

In recent years, chaotic and hyperchaotic systems that can produce various types and are suitable for secure communication have become topics of great interest in the fields of physics, biomathematics, and information security [1]. Compared with the traditional encryption method, the complex structure and dynamic behavior of chaotic attractors have a better encryption effect for digital image encryption [2]. Therefore, it has become more important to construct chaotic attractors with multiple scrolls. In the process of encrypting digital images, the core purpose is to change the position of pixels and the size of pixel values. Therefore, experts and scholars have proposed many encryption algorithms such as using chaotic sequences to perform bit disturb of images [3,4], using the chaotic sequence and image pixel value for the XOR operation [5,6], and scrambling the pixel [7–9]. A nonlinear state feedback controller was proposed in Reference [10] based on the original three-dimensional (3D) autonomous chaotic system to construct a new four-dimensional hyperchaotic system. An image encryption algorithm based on five-dimensional hyper-chaos and bit-level disturbance was proposed in Reference [11]. Chaotic image encryption algorithms based on bit-level scrambling and dynamic DNA coding were proposed in Reference [12]. Liu et al. [13] proposed an image encryption algorithm for bit position chaos on the upper four bits. On the basis of arranging the diffusion structure, to improve safety and

sensitivity, a chaotic image encryption algorithm based on a breadth-first search and dynamic diffusion was proposed in Reference [14]. The cryptosystem in Reference [15] uses a diffusion layer and then positions the image in layers instead of byte permutations to disturb the position of the image pixels. A new chaotic system with hidden attractors and a chaotic-based image encryption algorithm with a random number generator was proposed in Reference [16]. To reduce the processing time, Enayatifar et al. [17] performed simultaneous replacement and diffusion steps for any pixel.

Many chaotic-based image encryption algorithms have been inspired by Fridrich's method. Therefore, this architecture has become most well-known [18]. However, security of an efficient image encryption method is a fundamental issue in an image encryption algorithm. Recent cryptanalytical studies have proven that some chaos-based algorithms are not adequately secure to resist against a common attack such as Reference [19].

In the proposed scheme, the new five-dimensional chaotic system can generate three-leaf chaotic attractors in multiple directions. Simultaneously, its dynamic characteristics are analyzed. In the process of encryption, to increase the key space, we perform convolution operations on five chaotic sequences. Second, we scale the image matrix proportionally and scramble the image matrix for each segment separately. Through these processes, the difficulty of the exhaustive attack is increased. All except for the exhaustive method based on the explicit plaintext ciphertext mapping, the method will be invalid and have high security.

The rest of the paper is organized as follows. In Section 2, the chaotic system model is given. In Section 3, circuit design and experimental results are described. In Section 4, related knowledge is introduced. The encryption scheme is described in Section 5. Simulation results and performance analyses are reported in Section 6. Lastly, the conclusions are drawn in Section 7.

2. New Five-Dimensional Chaotic System

In 1994, Sprott [20] summarized many 3D chaotic systems. After analyzing these 3D chaotic systems, we propose a new 3D chaotic system. The system equations are as follows.

$$\begin{cases} \dot{x} = ax + byz^2 \\ \dot{y} = cx + dz^2 \\ \dot{z} = e + fx \end{cases} \tag{1}$$

when $a = -1$, $b = -3$, $c = 1$, $d = 1$, $e = 1$, $f = 1$, system (1) is in a chaotic state.

Based on system (1), we introduce two controllers, w, v, and feedback w into the original controller y, feedback v into the original controller w, original controller x feedback into the new controller v, original controller y feedback into the new controller w, v, original controller z feedback to the new controller w, and feedback v to w. These six operations make the five controllers of this system interact with each other, which makes the relationship more complicated. The newly constructed five-dimensional chaotic system is as follows.

$$\begin{cases} \dot{x} = ax + byz^2 \\ \dot{y} = x + z^2 - wz \\ \dot{z} = 1 + x \\ \dot{w} = yz + cw + dv \\ \dot{v} = xy + ev \end{cases} \tag{2}$$

when parameters $a = -1$, $b = -3$, $c = -3$, $d = 1.8$, $e = -5$, the system is in a chaotic state.

2.1. Dissipative Analysis

Because of

$$\Delta V = \frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{y}}{\partial y} + \frac{\partial \dot{z}}{\partial z} + \frac{\partial \dot{w}}{\partial w} + \frac{\partial \dot{v}}{\partial v} = a + c + e = -9 < 0 \tag{3}$$

Therefore, system (2) is dissipative, and convergence is in an exponential form of $V_0e^{-(a+c+e)t}$. Clearly, the volume element V_0 shrinks to the volume $V_0e^{-(a+c+e)t}$ at moment t. Now consider when $t \to \infty$. Each volume element that contains the system trajectory shrinks to 0 at an exponential rate $a + c + e = -9$.

2.2. Balance Point Analysis

Let $x = y = z = w = v = 0$, that is,

$$\begin{cases} ax + byz^2 = 0 \\ x + z^2 - wz = 0 \\ 1 + x = 0 \\ yz + cw + dv = 0 \\ xy + ev = 0 \end{cases} \tag{4}$$

when $a = -1$, $b = -3$, $c = -3$, $d = 1.8$, $e = -5$, the three equilibrium points of system (2) obtained by Equation (4) are $S_0 = [0, 0, 0, 0, 0]$, $S_1 = [\frac{4\sqrt{66}}{3}, \frac{4\sqrt{66}}{3}, 32, \frac{128\sqrt{66}}{9}, \frac{128\sqrt{66}}{81}]$, $S_2 = [-\frac{4\sqrt{66}}{3}, -\frac{4\sqrt{66}}{3}, 32, -\frac{128\sqrt{66}}{9}, -\frac{128\sqrt{66}}{81}]$. The balance point is S_0 and the corresponding eigenvalues are $\lambda_1 = 0, \lambda_2 = 0, \lambda_3 = -1, \lambda_4 = -3, \lambda_5 = -5$. Therefore, S_0 is a stable focus, the balance point is S_1, and the corresponding eigenvalues are $\lambda_1 = 25.7790, \lambda_2 = 1.9099, \lambda_3 = -12.5696, \lambda_{4,5} = -12.0596 \pm 81.956i$, where λ_1, λ_2 are positive numbers. Therefore, the balance point S_1 is an unstable saddle focus, the balance point is S_2, and the corresponding eigenvalues are $\lambda_1 = -0.8878, \lambda_2 = -8.7718, \lambda_3 = -73.2231, \lambda_{4,5} = 36.9413 \pm 77.9941i$, where the real part of $\lambda_{4,5}$ is positive, and the balance point S_2 is an unstable saddle focus.

2.3. Time Series Chart

When $a = -1$, $b = -3$, $c = -3$, $d = 1.8$, $e = -5$, the sequence diagram of the values x, y, z, w, v changes over time t, as shown in Figure 1. We can see that when the system parameters $a = -1$, $b = -3$, $c = -3$, $d = 1.8$, $e = -5$, system (2) is in a chaotic state.

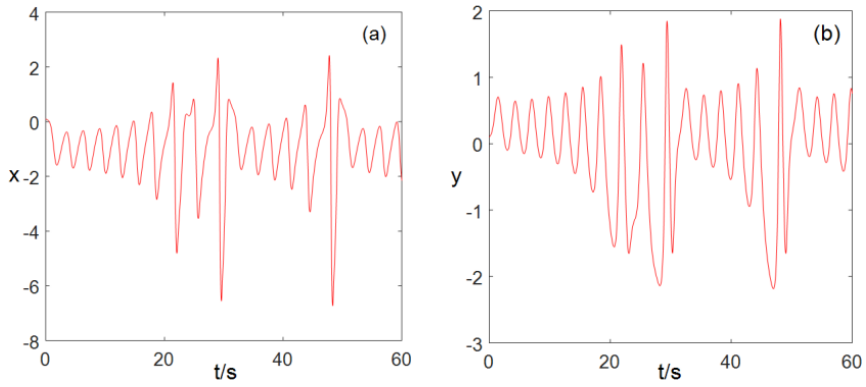

Figure 1. *Cont.*

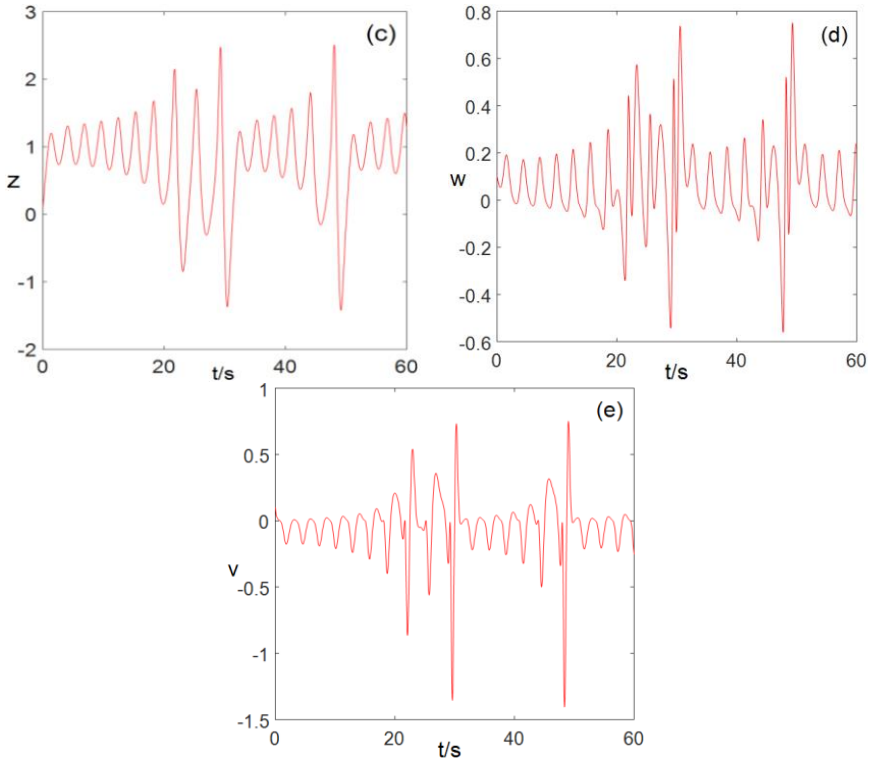

Figure 1. Time series diagram (**a**) $x-t$ time series, (**b**) $y-t$ time series, (**c**) $z-t$ time series, (**d**) $w-t$ time series, (**e**) $v-t$ time series.

2.4. Phase Diagram Analysis

For system parameters $a=-1$, $b=-3$, $c=-3$, $d=1.8$, $e=-5$, the 3D phase diagram generated by system (2) is shown in Figure 2. The resulting planar phase diagram is shown in Figure 3. It can be clearly seen from Figures 2 and 3 that the chaotic system can generate three-leaf chaotic attractors in multiple directions.

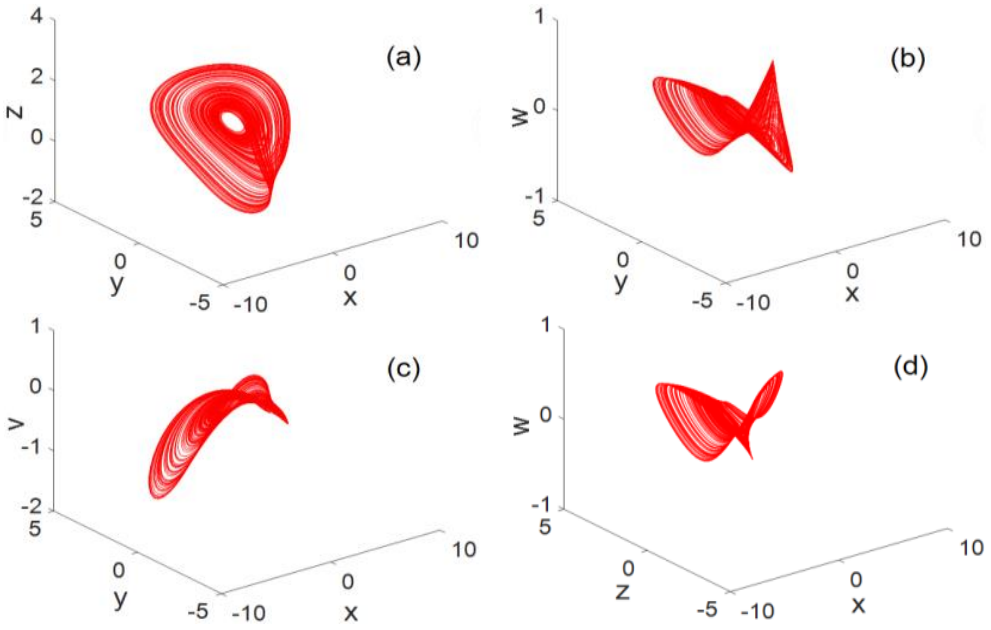

Figure 2. *Cont.*

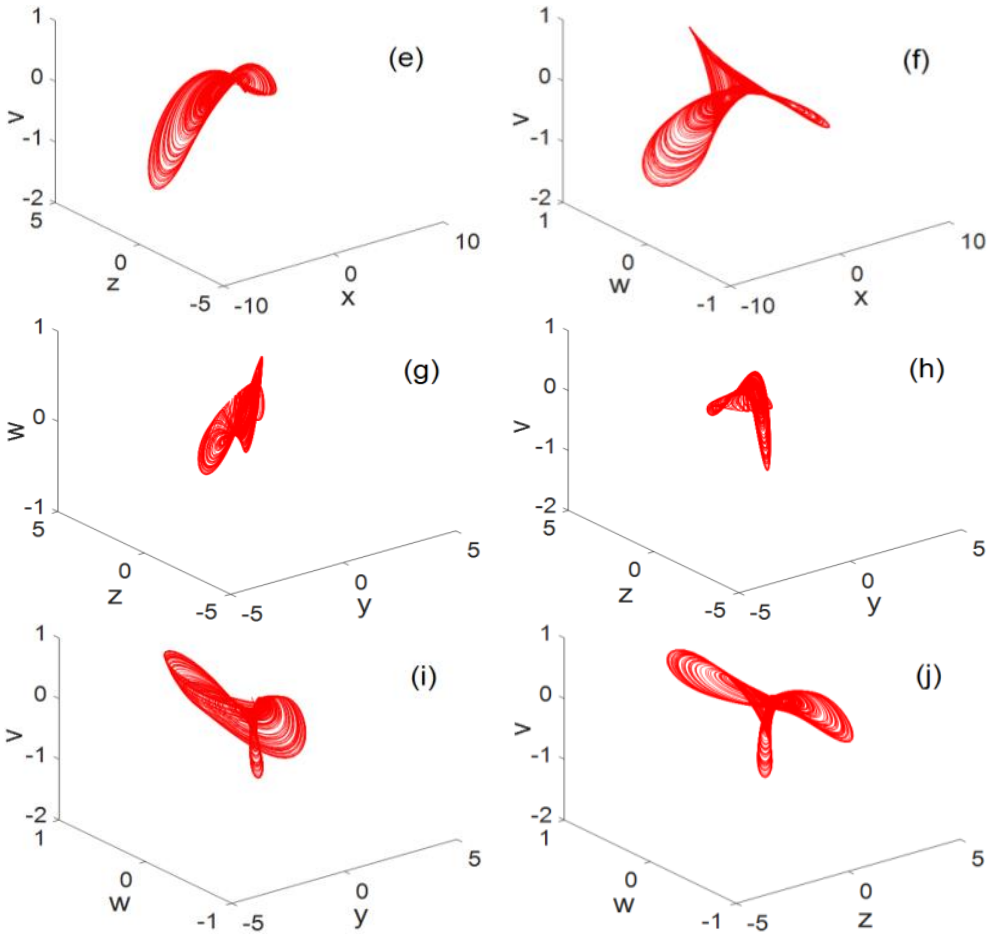

Figure 2. 3D phase diagram (**a**) $x-y-z$ 3D map, (**b**) $x-y-w$ 3D map, (**c**) $x-y-v$ 3D map, (**d**) $x-z-w$ 3D map, (**e**) $x-z-v$ 3D map, (**f**) $x-w-v$ 3D map, (**g**) $y-z-w$ 3D map, (**h**) $y-z-v$ 3D map, (**i**) $y-w-v$ 3D map, (**j**) $z-w-v$ 3D map.

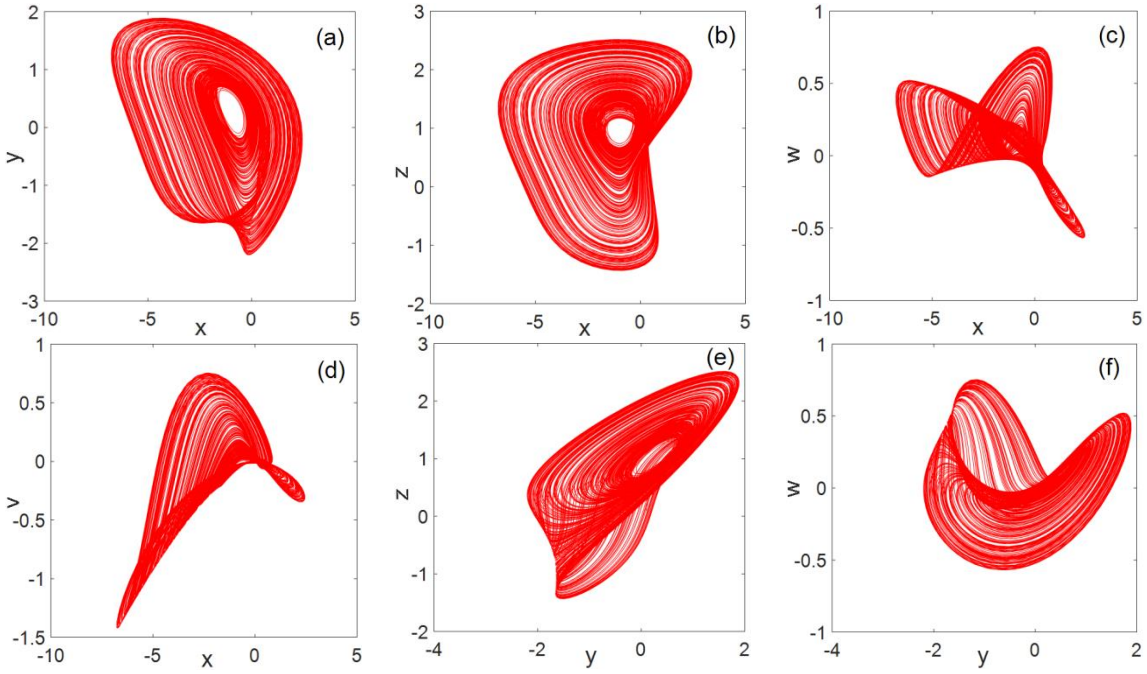

Figure 3. *Cont.*

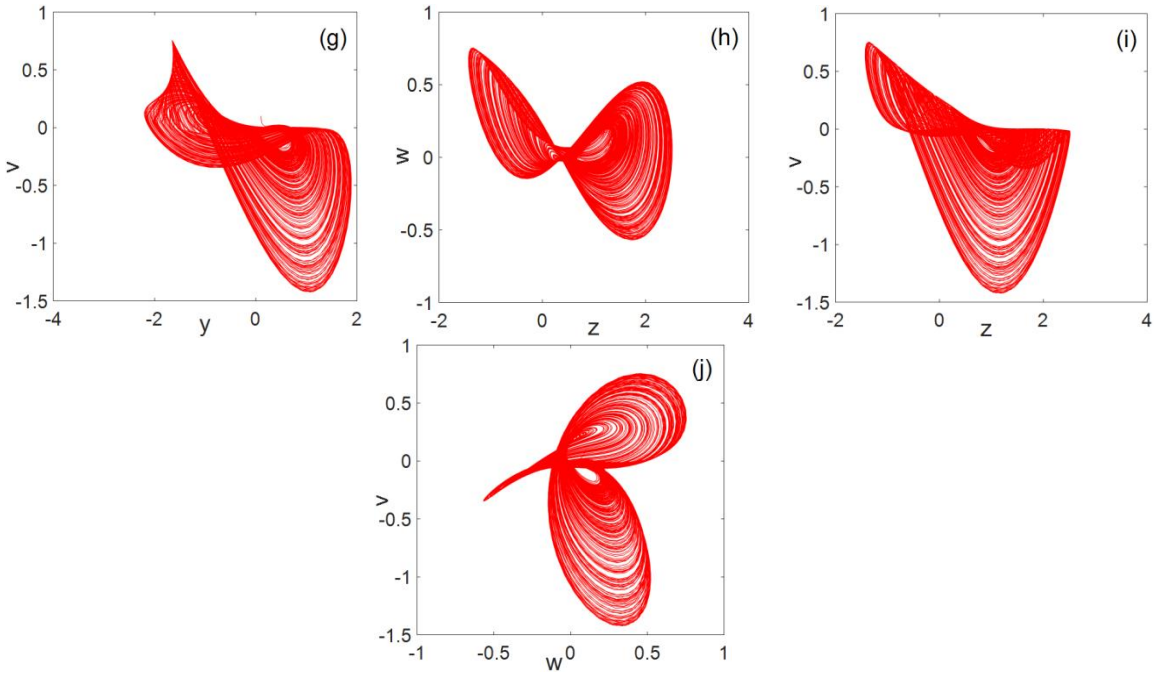

Figure 3. Plane phase diagram (**a**) $x-y$ flat, (**b**) $x-z$ flat, (**c**) $x-w$ flat, (**d**) $x-v$ flat, (**e**) $y-z$ flat, (**f**) $y-w$ flat, (**g**) $y-v$ flat, (**h**) $z-w$ flat, (**i**) $z-v$ flat, and (**j**) $w-v$ flat.

2.5. Bifurcation Diagram

For the equation parameters $b=-3$, $c=-3$, $d=1.8$, $e=-5$, the bifurcation diagram of the change in parameter a is shown in Figure 4. Figure 4 shows that, when the parameter $a\in(-1.1,-1)$, system (2) is in a chaotic state.

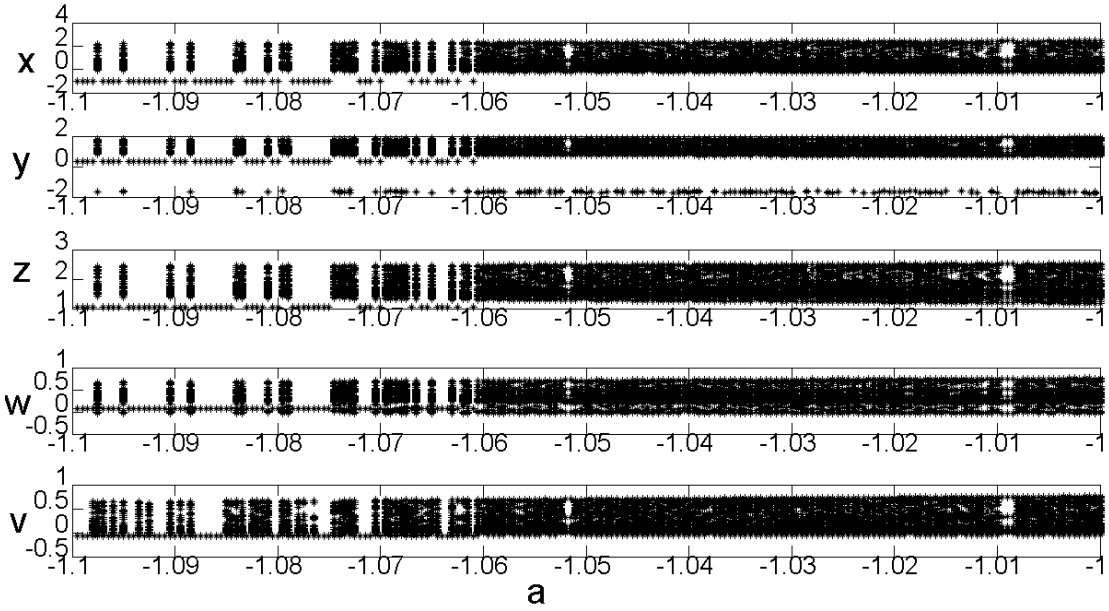

Figure 4. System (2) bifurcation diagram with variable a.

For equation parameters $a=-1$, $b=-3$, $c=-3$, $e=-5$, the bifurcation diagram of the change in parameter d is shown in Figure 5. As can be seen from Figure 5, when the parameter $d\in(1,2)$, system (2) is in a chaotic state.

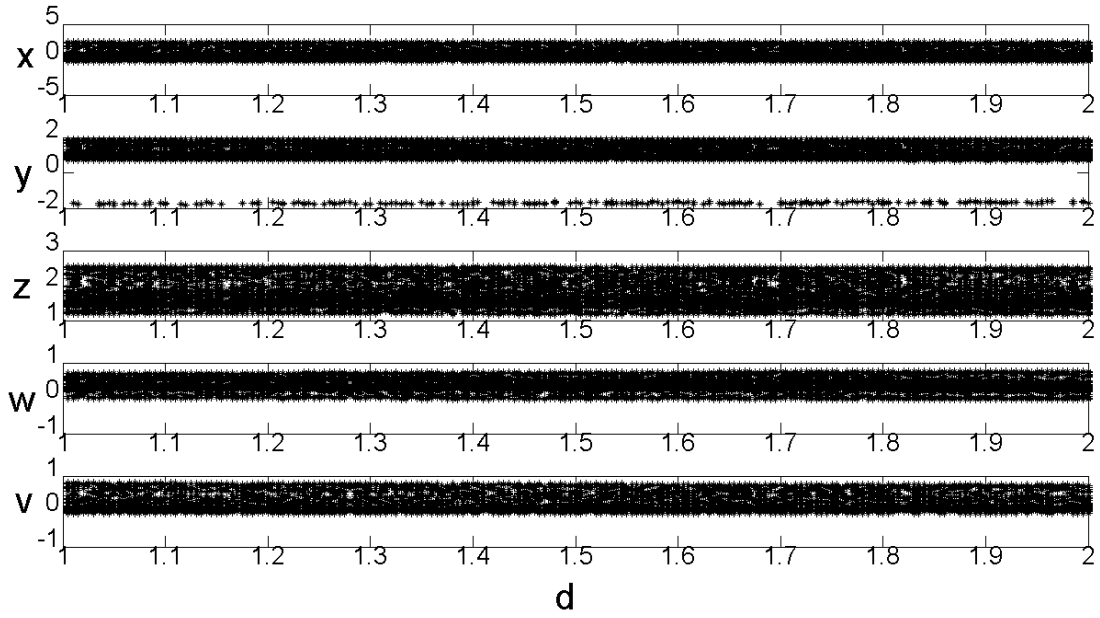

Figure 5. System (2) bifurcation diagram with variable d.

2.6. Power Spectrum Analysis

The power spectrum of the chaotic sequence is a continuum, and the calculation results of the sequence x, y, w, and v power spectrums of system (2) are shown in Figure 6a–d, respectively. It can be seen from Figure 6 that system (2) is in a chaotic state.

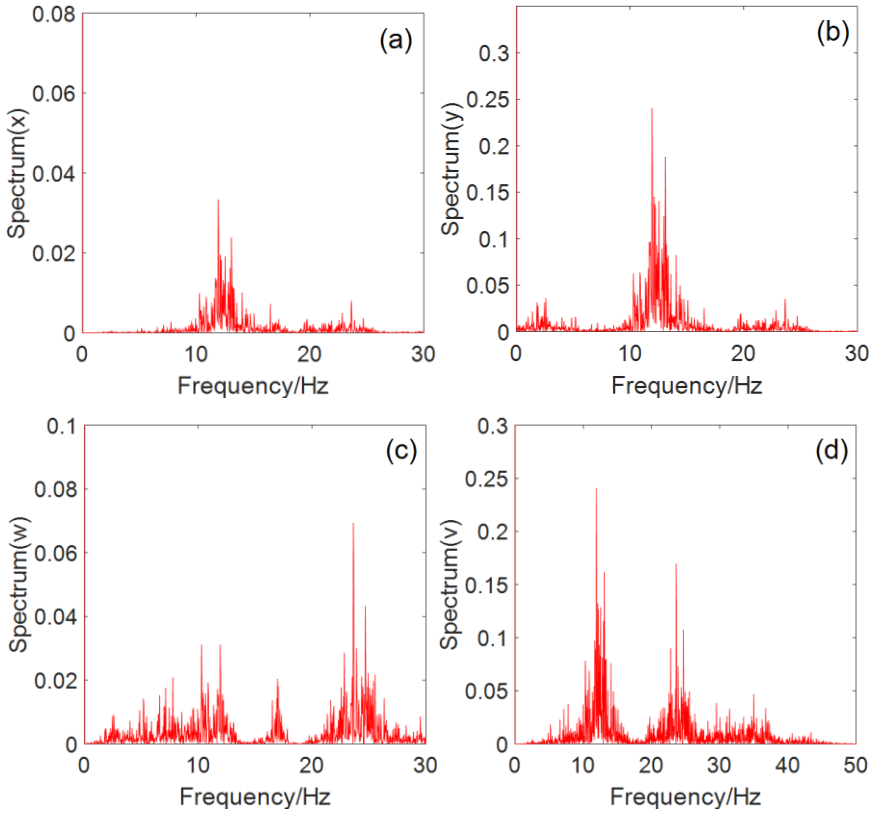

Figure 6. System power spectrum: (**a**) power spectrum of the x sequence, (**b**) power spectrum of the y sequence, (**c**) power spectrum of the w sequence, and (**d**) power spectrum of the v sequence.

3. Circuit Design and Experimental Results

In this section, we design an analog circuit, as shown in Figure 7, which is mainly composed of an inverting adder, integrator, and inverter composed of an operational amplifier TL082CD. The power supply voltage of the operational amplifier TL082CD is $E = \pm 15V$. This circuit has a simple structure and is easy to implement. The experimental results for this circuit are shown in Figure 8. Figure 8a shows the $y - v$ plane, Figure 8b shows the $x - w$ plane, and Figure 8c shows the $w - v$ plane. It can be seen from the oscilloscope that the experimental results for the circuit for the three-leaf chaotic attractor in each plane were consistent with the results of the numerical simulation experiments.

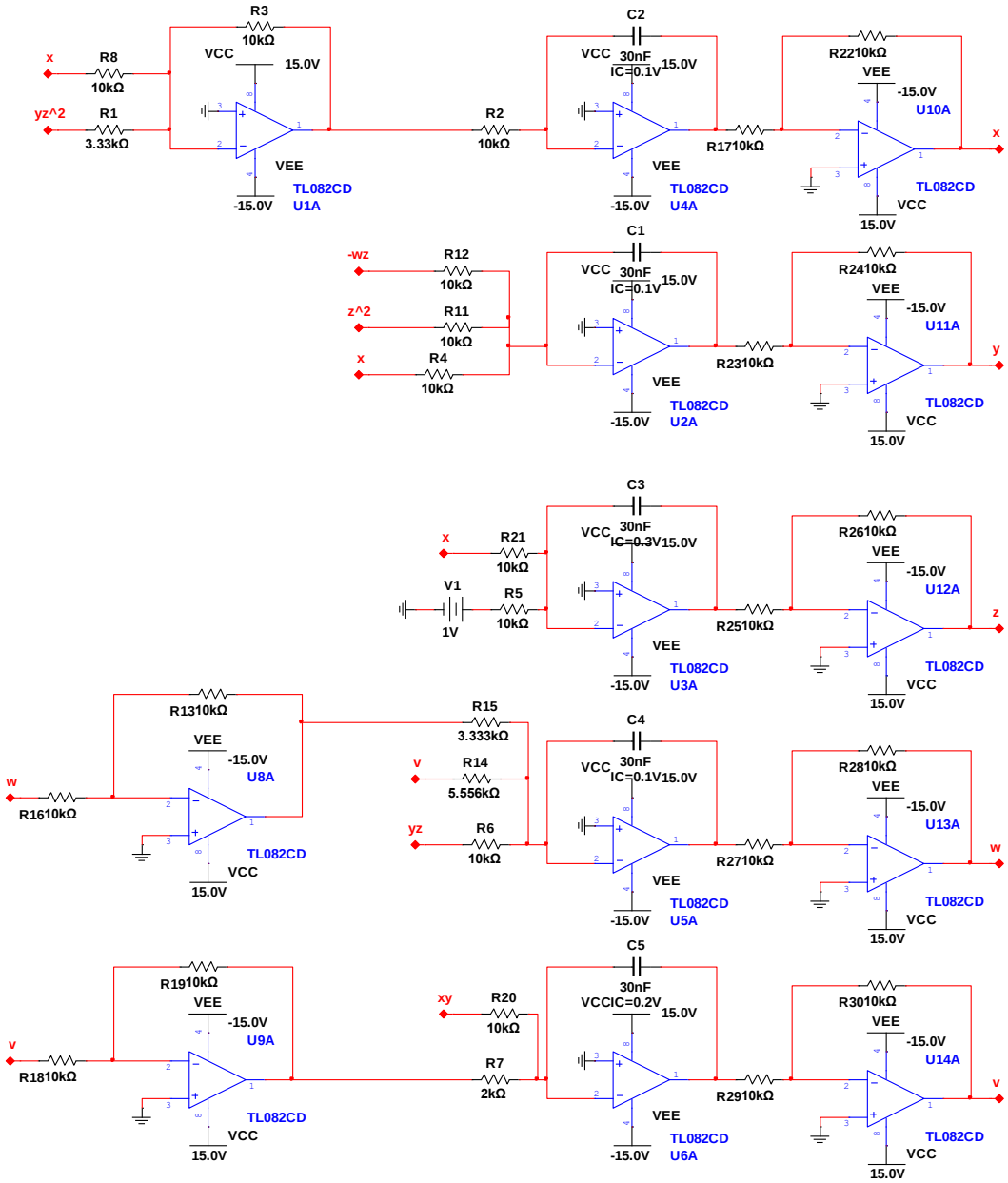

Figure 7. Five-dimensional chaotic system circuit diagrams.

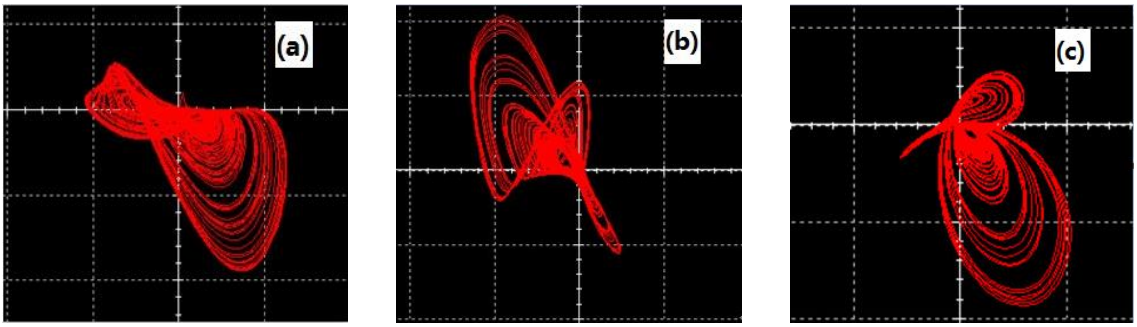

Figure 8. Experimental results for the circuit (**a**) $x-w$ plane, (**b**) $w-v$ plane, and (**c**) $w-v$ plane.

4. Related Information

4.1. Convolution Operation

Let

$$h = \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1m} \\ h_{21} & h_{22} & \dots & h_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ h_{m1} & h_{m2} & \dots & h_{mm} \end{bmatrix}$$

is the convolution kernel of $m \times m$,

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

is an $n \times n$ matrix, where $m < n$. Then a $n \times n$ matrix C is obtained by a convolution operation between matrix A and convolution kernel h.

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{n1} & c_{n2} & \dots & c_{nn} \end{bmatrix}$$

The convolution operation steps are as follows.

Step 1: Extend matrix A to an $(n+2) \times (n+2)$ matrix with 0.

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{11} & a_{12} & \dots & a_{1n} & 0 \\ 0 & a_{21} & a_{22} & \dots & a_{2n} & 0 \\ 0 & \vdots & \vdots & \vdots & \vdots & 0 \\ 0 & a_{n1} & a_{n2} & \dots & a_{nn} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where $B(1, 1:n+2) = B(1:n+2, 1) = B(n+2, 1:n+2) = B(1:n+2, n+2) = 0$, $B(2:n+1, 2:n+1) = A$.

Step2: Obtain matrix $C = (c_{ij})_{nn}$ using a convolution operation between matrix A and convolution kernel h, where

$$c_{ij} = \sum_{p,q=1}^{p,q=m} h_{pq} \times B(p + (i-1), q + (j-1)),\ i = 1, 2, \cdots, n,\ j = 1, 2, \cdots n, \tag{5}$$

4.2. "Same OR" Operation

The "same OR" operation and the "exclusive OR" operation have the same effect. The "same OR" operation is defined as follows: when the input variables are the same, the output is 1, and when the input variables are different, the output is 0. The calculation results are presented in Table 1.

Table 1. Result table for the "same OR" operation.

A(Input)	B(Input)	F(Result)
0	0	1
0	1	0
1	0	0
1	1	1

5. Algorithm Descriptions

5.1. Encryption Algorithm Description

The flow chart of the encryption is shown in Figure 9.

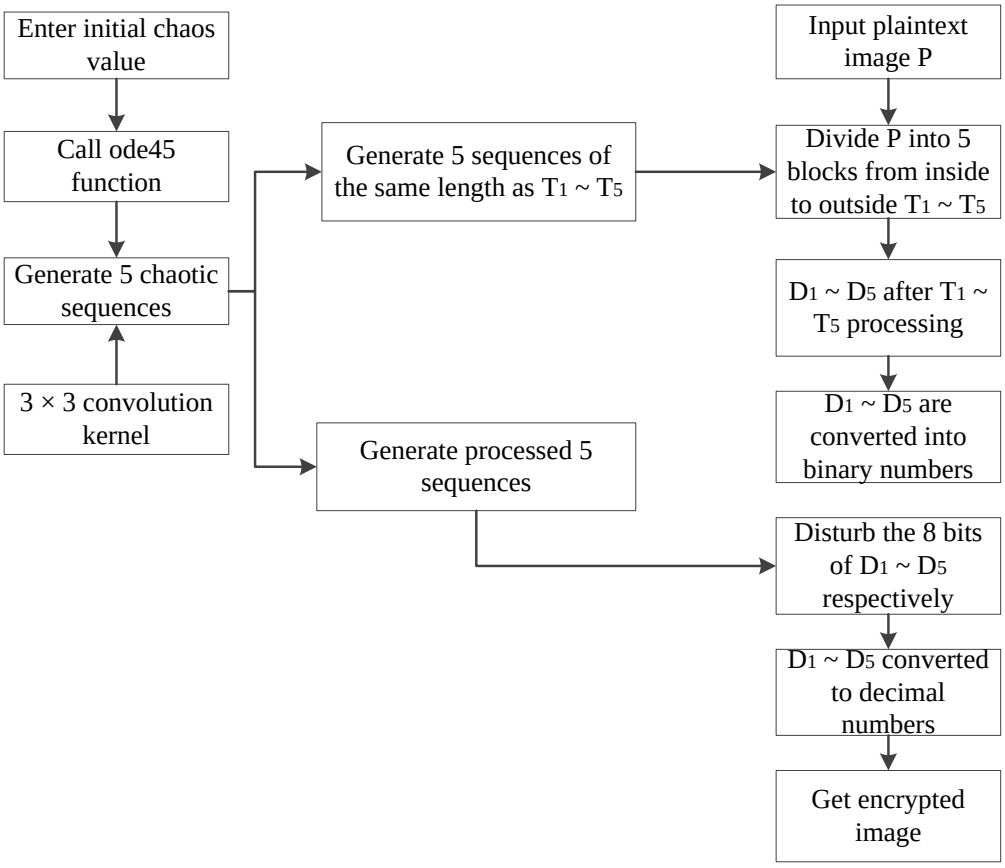

Figure 9. Encryption flow chart.

Given an $M \times N$ grayscale image A, the encryption steps are as follows.

Step 1: Input a grayscale image A, initial value of the chaotic system $y_0 = [0.6, 0.1, 0.2, 0.5, 0.4]$, and the step size $L = 0.01$.

Step 2: Find the total iteration time $T = (250 + P \times P) \times L$, where $P = \max(M, N)$.

Step 3: Call the ode45 function, iterate system (2), and generate five chaotic sequences.

Step 4: The five chaotic sequences are treated separately as follows.

$$A1(:,:,i) = (reshape(y(250 : 249 + M \times N, i), M, N)) \times 10^{11}, i = 1, 2, 3, 4, 5 \tag{6}$$

where $y(250 : (249 + M \times N), i)$ is the $M \times N$ value that starts from the 250th value of the chaotic sequence $y(:, i)$. Use $(reshape(y(250 : (249 + M \times N), i), M, N)) \times 10^{11}$ to convert the fetched $M \times N$ values into an $M \times N$ matrix, and then multiply each element value of the matrix by 10^{11}

Step 5: Given a convolution kernel of 3×3.

Step 6: For the five $M \times N$ matrices $A1(:,:,i), i = 1, 2, 3, 4, 5$ obtained in step 4, apply the convolution operation with the given convolution kernel to obtain five $M \times N$ matrices $Y(:,:,i), i = 1, 2, 3, 4, 5$.

Step 7: Divide grayscale image A into five regions starting from the center area in the proportion $l = M : N$. Input the value of m_1 using $n_1 = m_1 \div l$, and obtain n_1. The formulas for m_i and n_i are as follows.

$$\begin{cases} m_{i+1} = m_i + \frac{1}{4}\left(round\left(\frac{M}{2}\right) - m_1\right) \\ n_{i+1} = n_i + \frac{1}{4}\left(round\left(\frac{N}{2}\right) - n_1\right) \end{cases}, i = 1, 2, 3 \tag{7}$$

There are $T1 = m_1 \times n_1$, $T2 = m_2 \times n_2$, $T3 = m_3 \times n_3$, $T4 = m_4 \times n_4$, $T5 = M \times N$ as shown in Figure 10.

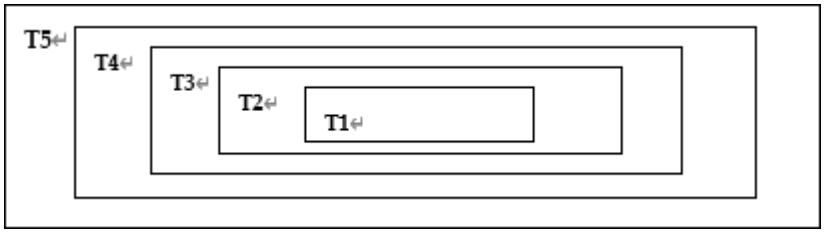

Figure 10. Division of grayscale image A.

Step 8: Divide the region $H1$ in matrix $Y(:,:,1)$ according to the starting point and size of $T1$ in grayscale image A, as shown in Figure 11.

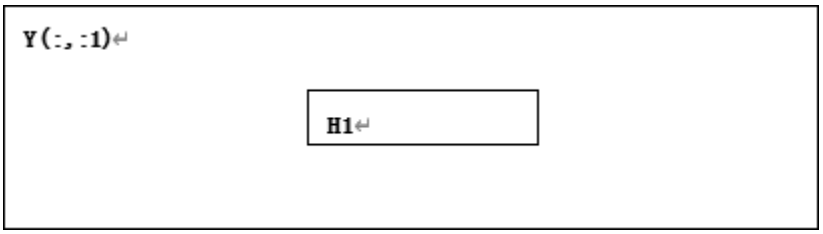

Figure 11. $H1$ divided from $Y(:,:,1)$.

Step 9: Convert matrix $T1$ and matrix $H1$ sums of one row and $(m_1 \times n_1)$ column matrix $T11$ and $H11$, respectively. $H11$ is treated as follows.

$$HH = mod(round(H11 \times 10^{-7}), 256) \tag{8}$$

Step 10: Combine $T11$ and HH, and perform a bitwise "XOR" operation to obtain matrix $B1$.

Step 11: Process $H11$ as follows to obtain $M1$ and $M2$.

$$\begin{cases} M1 = mod(round(H11 \times 10^4), 7) + 1 \\ M2 = mod(round(H11(end-7:end) \times 10^4), m_1 \times n_1 - 1) + 1 \end{cases} \quad (9)$$

Step 12: Convert matrix $B1$ into binary matrix $F1$.

Step 13: Disturb each line of binary numbers in each row of $F1$, and then obtain matrix $C1$. The disturbance formula is as follows.

$$C1(i1,:) = circshift(F1(i1,:), M1(i1), 2), i1 = 1, 2, 3, \ldots, m_1 \times n_1 \quad (10)$$

where $F1(i1,:)$ denotes all the columns of row $i1$ of matrix A and $circshift(A,k,2)$ moves all elements of row vector A clockwise by k units.

Step 14: Disturb all column elements of $C1$, and then obtain $C2$. The disturbance formula is as follows.

$$C2(:,i2) = circshift(C1(:,i2), M2(i2), 1), i2 = 1, 2, 3, 4, 5, 6, 7, 8 \quad (11)$$

where $C1(:,i2)$ denotes all the rows of column $i2$ of matrix $C1$ and *circshift*(A,k,1) moves all the elements of column vector A clockwise by k units.

Step 15: Convert binary number matrix $C2$ to decimal number matrix $D1$.

Step 16: $D1$ is used to replace the area $T1$. The results are shown in Figure 12.

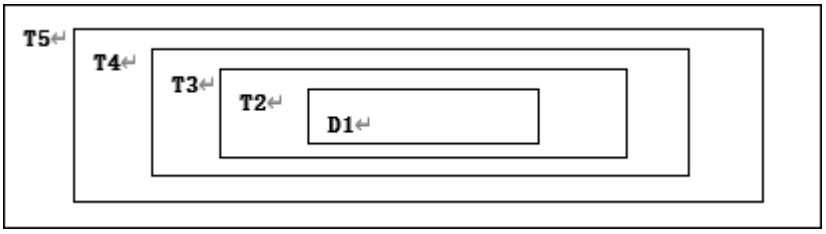

Figure 12. Subtraction after replacing $T1$ with $D1$.

Step 17: The area $H2$ is taken out according to $T2$ in the starting point and size in A from $Y(:,:,2)$, as shown in Figure 13.

Figure 13. $H2$ divided in $Y(:,:,2)$.

Step 18: Convert matrix $T2$ and matrix $H2$ sums of one row and $(m_2 \times n_2)$ column matrix $T22$ and $H22$, respectively. $H22$ is treated as follows.

$$HH1 = mod(round(H22 \times 10^{-7}), 256) \quad (12)$$

Step 19: Combine $T22$ and $HH1$, and perform a bitwise "same OR" operation to obtain matrix $B2$

Step 20: Process $H22$ to obtain $M3$ and $M4$ as follows.

$$\begin{cases} M3 = mod(round(H22 \times 10^4), 7) + 1 \\ M4 = mod(round(H22(end-7:end) \times 10^4), m_2 \times n_2 - 1) + 1 \end{cases} \quad (13)$$

Step 21: Convert matrix $B2$ into binary matrix $F2$.

Step 22: Scramble each row of binary numbers in $F2$ to obtain matrix $C3$. The scrambling formula is as follows.

$$C3(i3,:) = circshift(F2(i3,:), M3(i3), 2),\ i3 = 1, 2, 3, \ldots, m_2 \times n_2 \tag{14}$$

Step 23: Disturb all the column elements of $C3$ to obtain $C4$. The scrambling formula is as follows.

$$C4(:, i4) = circshift(C3(:, i4), M4(i4), 1),\ i4 = 1, 2, 3, 4, 5, 6, 7, 8 \tag{15}$$

Step 24: Convert binary number matrix $C4$ to decimal number matrix $D2$.

Step 25: Replace area $T2$ with $D2$. The result is shown in Figure 14.

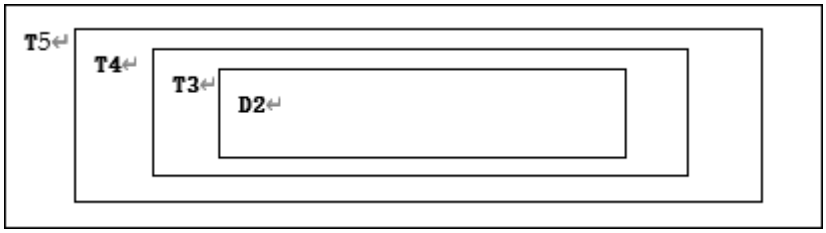

Figure 14. Subtraction after replacing $T2$ with $D2$.

Step26: According to $T3$ from the starting point and size in the A, take the area $H3$ from $Y(:,:,3)$, as shown in Figure 15.

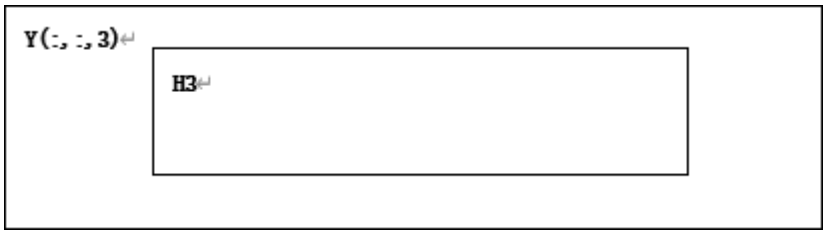

Figure 15. $Y(:,:,3)$ is divided to obtain $H3$.

Step 27: Convert the matrices $T3$ and $H3$ into matrices $T33$ and $H33$ with one row and $(m_3 \times n_3)$ columns, respectively, and process $H33$ as follows.

$$HH2 = mod(round(H33 \times 10^{-7}), 256) \tag{16}$$

Step 28: Combine $T33$ and $HH2$, and perform a bitwise "same OR" operation to obtain matrix $B3$.

Step 29: Process $H33$ as follows to obtain $M5$ and $M6$.

$$\begin{cases} M5 = mod(round(H33 \times 10^4), 7) + 1 \\ M6 = mod(round(H33(end-7:end) \times 10^4), m_3 \times n_3 - 1) + 1 \end{cases} \tag{17}$$

Step 30: Convert matrix $B3$ into binary matrix $F3$.

Step 31: Scramble the binary numbers in each row of $F3$ to obtain matrix $C5$. The scrambling formula is as follows.

$$C5(i5,:) = circshift(F3(i5,:), M5(i5), 2),\ i5 = 1, 2, 3, \ldots, m_3 \times n_3 \tag{18}$$

Step 32: Disturb all the column elements of $C5$ to obtain $C6$. The scrambling formula is as follows.

$$C6(:, i6) = circshift(C5(:, i6), M6(i6), 1),\ i6 = 1, 2, 3, 4, 5, 6, 7, 8 \tag{19}$$

Step 33: Convert binary number matrix $C6$ to decimal number matrix $D3$.
Step 34: Replace area $T3$ with $D3$. The result is shown in Figure 16.

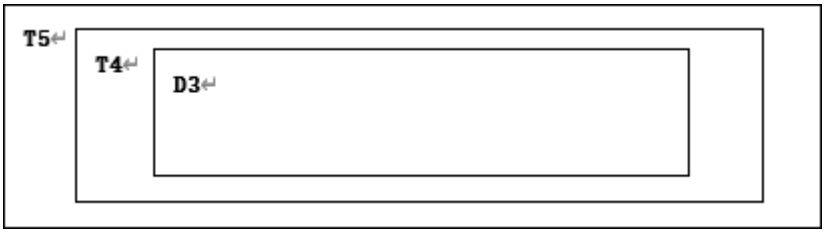

Figure 16. Subtraction after replacing $T3$ with $D3$.

Step 35: Take region $H4$ from $Y(:,:,4)$ according to the starting point and size of $T4$ in A, as shown in Figure 17.

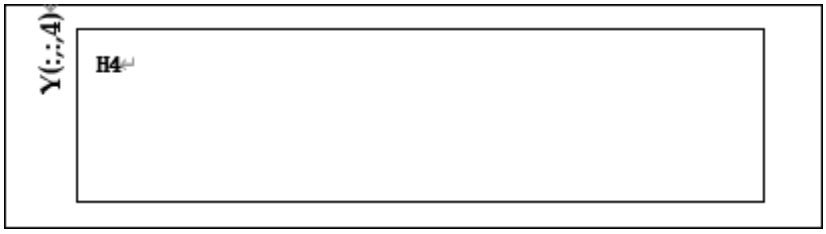

Figure 17. $Y(:,:,4)$ is divided to obtain $H4$.

Step 36: Convert matrices $T4$ and $H4$ into matrices $T44$ and $H44$ with one row and $(m_4 \times n_4)$ columns, respectively, and process $H44$ as follows.

$$HH3 = mod(round(H44 \times 10^{-7}), 256) \tag{20}$$

Step 37: Combine $T44$ and $HH3$, and perform a bitwise "XOR" operation to obtain matrix $B4$.
Step 38: Process $H44$ to obtain $M7$ and $M8$ as follows.

$$\begin{cases} M7 = mod(round(H44 \times 10^4), 7) + 1 \\ M8 = mod(round(H44(end-7:end) \times 10^4), m_4 \times n_4 - 1) + 1 \end{cases} \tag{21}$$

Step 39: Convert matrix $B4$ into binary matrix $F4$.
Step 40: Scramble the binary numbers in each row in $F4$ to obtain matrix $C7$. The scrambling formula is as follows.

$$C7(i7,:) = circshift(F4(i7,:), M7(i7), 2), i7 = 1, 2, 3, \ldots, m_4 \times n_4 \tag{22}$$

Step 41: Disturb all the column elements of $C7$ to obtain $C8$. The scrambling formula is as follows.

$$C8(:,i8) = circshift(C7(:,i8), M8(i8), 1), i8 = 1, 2, 3, 4, 5, 6, 7, 8 \tag{23}$$

Step42: Convert binary number matrix $C8$ to decimal number matrix $D4$.
Step43: Replace area $T4$ with $D4$. The result is shown in Figure 18.

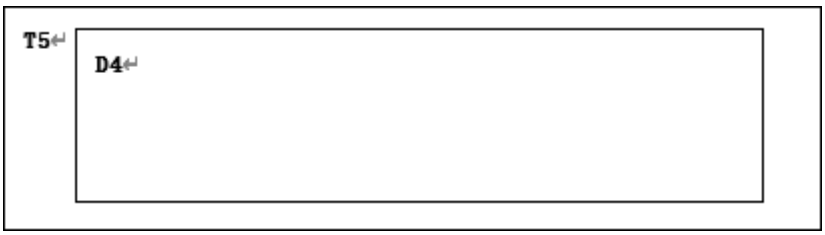

Figure 18. Subtraction after replacing $T4$ with $D4$.

Step 44: Convert matrix $T5$ and $Y(:,:,5)$ to matrix $T55$ and $H55$ with one row and $(M \times N)$ columns, respectively, and $H55$ is treated as follows.

$$HH4 = mod(round(H55 \times 10^{-7}), 256) \tag{24}$$

Step 45: Combine $T55$ and $HH4$, and perform a bitwise "same OR" operation to obtain matrix $B5$.

Step 46: Process $H55$ as follows to obtain $M9$ and $M10$.

$$\begin{cases} M9 = mod(round(H55 \times 10^4), 7) + 1 \\ M10 = mod(round(H55(end-7:end) \times 10^4), M \times N - 1) + 1 \end{cases} \tag{25}$$

Step 47: Convert matrix $B5$ into binary matrix $F5$.

Step 48: Scramble the binary numbers in each row of $F5$ to obtain matrix $C9$. The scrambling formula is as follows.

$$C9(i9,:) = circshift(F5(i9,:), M9(i9), 2), i9 = 1, 2, 3, \ldots, M \times N \tag{26}$$

Step 49: Disturb all the column elements of $C9$. The scrambling formula is as follows.

$$C10(:, i10) = circshift(C9(:, i10), M10(i10), 1), i10 = 1, 2, 3, 4, 5, 6, 7, 8 \tag{27}$$

Step 50: Convert binary number matrix $C10$ to decimal number matrix $D5$, and $D5$ is the final encrypted image. The result is shown in Figure 19.

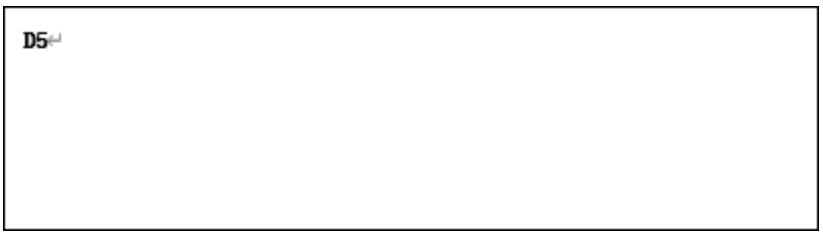

Figure 19. Encrypted image.

5.2. Decryption Algorithm Description

Step 1: Input the initial value of the chaotic system $y_0 = [0.6, 0.1, 0.2, 0.5, 0.4]$ and step size $L = 0.02$, and find the total iteration time $T = (250 + P \times P) \times L$, where $P = \max(M, N)$.

Step 2: Call the ode45 function, iterate system (2), and generate five chaotic sequences.

Step 3: The five chaotic sequences are treated as follows.

$$A1(:,:,i) = (reshape(y(250 : 249 + M \times N, i), M, N)) \times 10^{11}, i = 1, 2, 3, 4, 5 \tag{28}$$

where $y(250 : (249 + M \times N), i)$ is the $M \times N$ value starting from the 250th value of chaotic sequence $y(:, i)$. Use $(reshape(y(250 : (249 + M \times N), i), M, N)) \times 10^{11}$ to convert the fetched $M \times N$ values into an $M \times N$ matrix, and then multiply each element value of the matrix by 10^{11}.

Step 4: Regarding the matrix $A1(:,:,i)$, $i = 1,2,3,4,5$ of $M \times N$, the five obtained in the previous step convolution operation with the given convolution kernel of the encryption algorithm include five matrix $Y1(:,:,i)$, $i = 1,2,3,4,5$ of $M \times N$ are obtained.

Step 5: Divide the encrypted image $D5$ of $M \times N$ into five regions in proportion to the middle region, with $l = M : N$, $E1 = (m_1 \times n_2)$, $E2(m_2 \times n_2)$, $E3(m_3 \times n_3)$, $E4(m_4 \times n_4)$, $E5(M \times N)$, as shown in Figure 20.

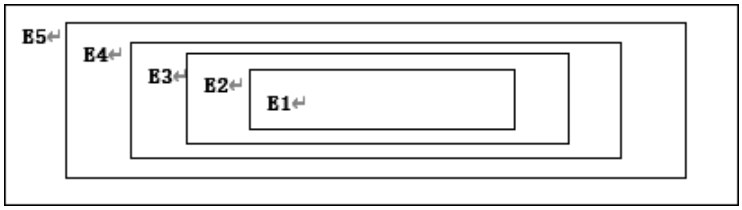

Figure 20. Division diagram for $D5$.

Step6: Restore $E5, E4, E3, E2, E1$ in turn to obtain decrypted image A.

6. Experimental Results and Analysis

6.1. Experiment Platform

The PC configuration was as follows: Intel(R) Core (TM) i5-6500 CPU @ 3.70 GHz 3.70 GHz, memory 8 GB, and Windows 7 64-bit operating system. The above encryption algorithm was implemented in a program in MATLAB R2014a.

6.2. Experimental Result

For the experiment, six types of grayscale images of classic images were selected: Lena, boat, baboon, peppers, couple, and leaf, which were all 256×256. This algorithm is also applicable to grayscale images of any sizes. The plaintext image, encrypted image, and decrypted image are shown in Figure 21.

6.3. Key Space Analysis

The size of the key space is one of the most important factors that determines the strength of the image encryption algorithm. The larger the key space, the stronger the ability to resist brute force attacks. The secret keys of this proposed encryption algorithm include five initial values $y_0 = [0.6, 0.1, 0.2, 0.5, 0.4]$ and five system parameters a, b, c, d, e. Since the precision is 10^{-15} by computer with accuracy, the key space is $(10^{15})^{10} = 10^{150}$. In addition, when the convolution kernel size is 3×3, the secret key also needs to consider nine convolution kernel parameters. Therefore, the total key space of the algorithm is much larger than $10^{150} > 2^{100}$. For a security encryption algorithm, its key space should be larger than 2^{100} [21]. Therefore, this algorithm was sufficiently secure.

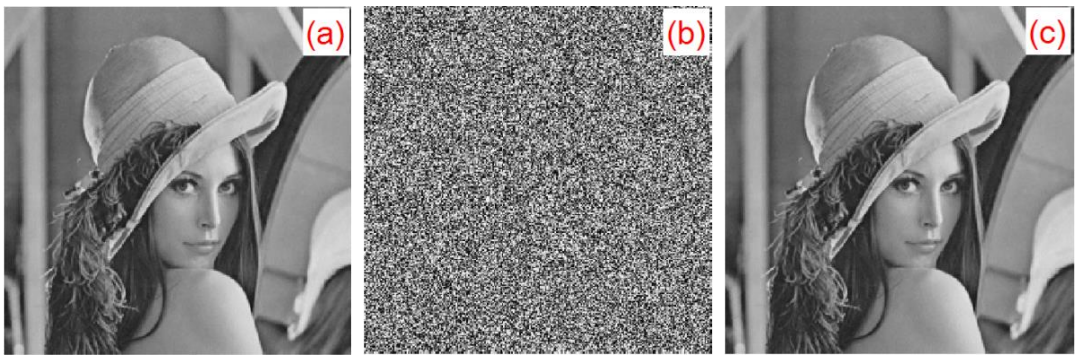

Figure 21. *Cont.*

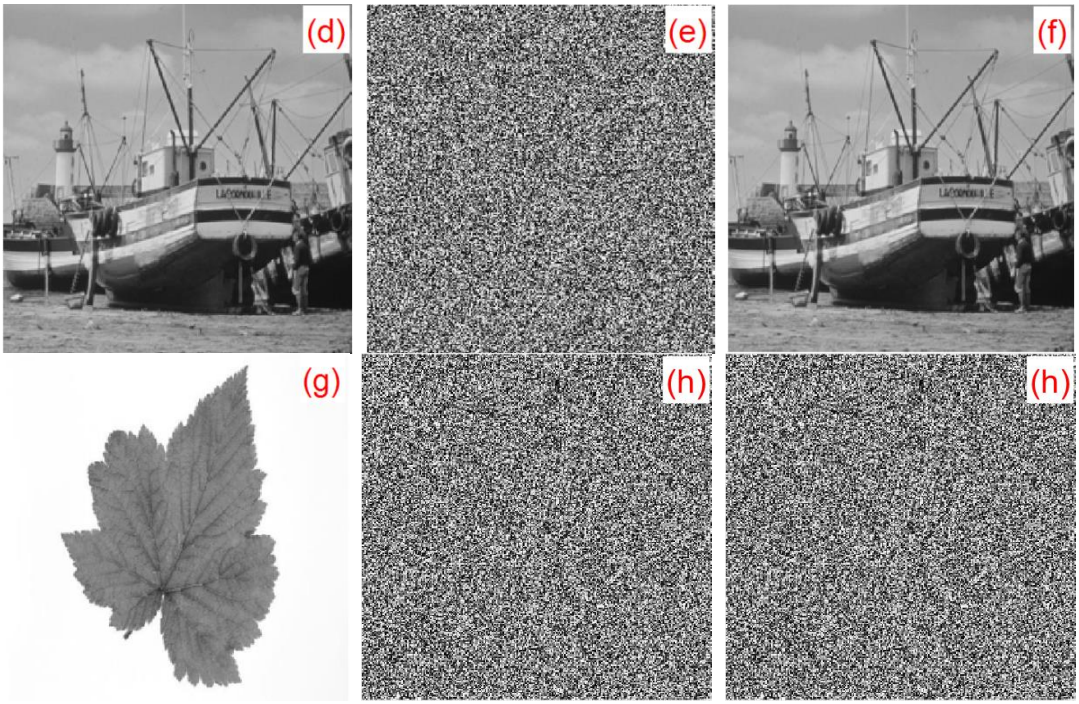

Figure 21. Encrypted/decrypted experimental results: (**a**) Lena original image, (**b**) Lena encrypted image, (**c**) Lena decrypted image, (**d**) boat original image, (**e**) boat encrypted image, (**f**) boat decrypted image, (**g**) leaf original image, (**h**) leaf encrypted image, and (**i**) leaf decrypted image.

6.4. Convolution Nuclear Sensitivity Analysis

In the encryption process, we multiplied each element of the five chaotic sequences produced by 10^{11} and performed convolution operations with the convolution kernel of 3×3. The convolution kernel of 3×3 in this algorithm was $c = [1, 2, 3; 4, 5, 6; 7, 8, 9]$. In the decryption process, when any of the parameters in the convolution kernel were slightly changed, the original image could not be successfully decrypted. When any parameter in the convolution kernel changed slightly with 10^{-15}, the decrypted image was blurred, but the outline could be seen, as shown in Figure 22c. When any of the parameters in the convolution kernel was slightly changed with 10^{-14}, the decrypted image could not substantially display the plaintext image information, as shown in Figure 22d. When any parameter in the convolution kernel changed slightly with 10^{-13}, the plaintext image information could not be solved at all, as shown in Figure 22e. Taking the Lena image as an example, we made a slight change to the parameters of the second row and the second column of the convolution kernel: $c_1 = [1, 2, 3; 4, 5.000000000000001, 6; 7, 8, 9]$, $c_2 = [1, 2, 3; 4, 5.00000000000001, 6; 7, 8, 9]$, $c_3 = [1, 2, 3; 4, 5.0000000000001, 6; 7, 8, 9]$. The plaintext images, ciphertext images, and the corresponding decrypted images of c_1, c_2, and c_3 are shown in Figure 22.

A small change in the convolution kernel can lead to a great change in the ciphertext. In this study, only the Lena image was used as an example. Figure 23 shows the sensitivity of this algorithm to convolution kernels. Figure 23a is a plaintext image. Figure 23b,c are convolution kernels, $c_0 = [1, 2, 3; 4, 5, 6; 7, 8, 9]$ and $c_1 = [1, 2, 3; 4, 5.00000000000001, 6; 7, 8, 9]$ for encrypted ciphertext image C_0 and C_1, respectively. Figure 23d is the result of the correct decryption for C_0 using c_0. Figure 23e,f show the results of C_0 and C_1 using the wrong convolution kernel c_1 and c_0 decryption, respectively. Figure 23 illustrates that, despite only minor changes between the convolution kernels c_0 and c_1, the ciphertext images C_0 and C_1 could not be properly decrypted with the convolution kernels.

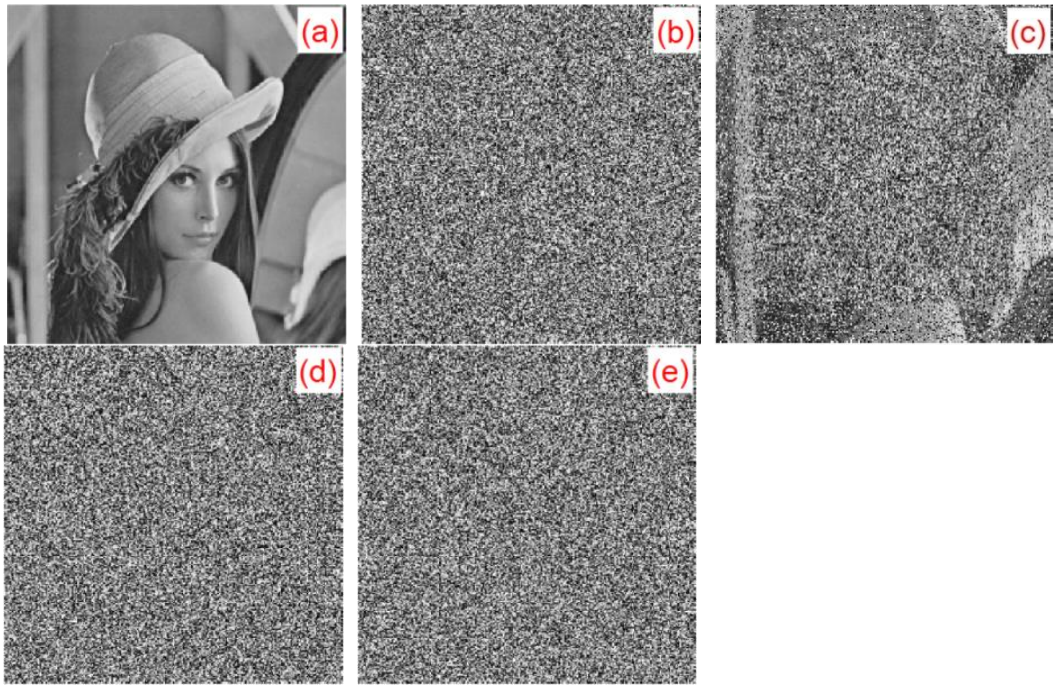

Figure 22. c_1, c_2, c_3 decryption map: (**a**) Lena plaintext image, (**b**) c corresponding decrypted image, (**c**) c_1 corresponding decrypted image, (**d**) c_2 corresponding decrypted image, and (**e**) c_3 corresponding decrypted image.

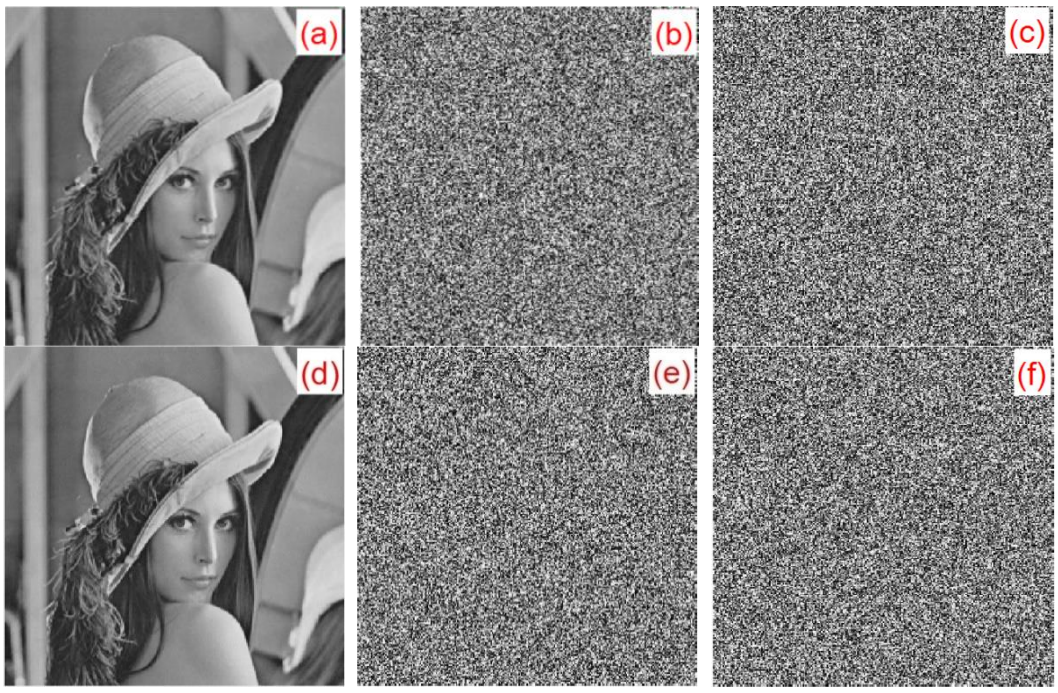

Figure 23. Convolution nuclear sensitivity experiment analysis chart: (**a**) Plaintext image, (**b**) Ciphertext C_0 (convolution kernel c_0), (**c**) Ciphertext C_1 (convolution kernel c_1), (**d**) C_0 correct decryption result, (**e**) C_0 error decryption result using c_1, and (**f**) C_1 error decryption result using c_0.

The difference between the two images can also be measured by the pixel change rate (NPCR) and the normalized mean change intensity (UACI), which are described as Equations (29) and (30).

$$NPCR = \frac{\sum_{i,j}^{M,N} D(i,j)}{M \times N} \times 100\% \tag{29}$$

$$UACI = \frac{\sum_{i,j}^{M,N} |C_1(i,j) - C_2(i,j)|}{M \times N \times 255} \times 100\% \tag{30}$$

where $D(i,j) = \begin{cases} 1, C_1(i,j) \neq C_2(i,j) \\ 0, C_1(i,j) = C_2(i,j) \end{cases}$; M,N is the size of the image. $C_1(i,j)$ and $C_2(i,j)$ are the pixel values at position (i,j). The larger the values of NPCR and UACI, the greater the difference between the two images. To better evaluate the sensitivity of the convolution kernel, we use Equations (29) and (30) to calculate the *NPCR* and *UACI* of the encrypted images obtained by the slightly changed convolution kernels c_1, c_2, c_3, c_4, c_5 and the encrypted image obtained by the original convolution kernel c_0. There are $c_1 = [1,2,3;4,5.000000000000001,6;7,8,9]$, $c_2 = [1,2,\ 3;4,5.00000000000001,6;7,8,9]$, $c_3 = [1,2,3;4,5.0000000000001,6;7,8,9]$, $c_4 = [1,2,3;4,5.00000000001,6;7,8,9]$, $c_5 = [1,2,3;4,5.00000000001,6;7,8,9]$, $c_0 = [1,2,3;4,5,6;7,8,9]$. The *NPCR* and *UACI* values between the obtained encrypted images are shown in Table 2. Table 2 shows that using the convolution kernel operation can greatly increase the key space of the algorithm.

Table 2. Convolution nuclear sensitivity test results.

Lena Image		Kernel c_1	Kernel c_2	Kernel c_3	Kernel c_4	Kernel c_5
Index	Pixel Change Rate (NPCR)	0.9716	0.9962	0.9962	0.9963	0.996
	Normalized Mean Change Intensity (UACI)	0.2271	0.3341	0.3335	0.3352	0.3346

6.5. Key Sensitivity Analysis

A small change in the key results in a great change in the ciphertext, which is the key sensitivity. In the experiment, the Lena image was considered as an example. Figure 24 shows the sensitivity of the algorithm to the initial key. Figure 24a is a plaintext image. Figure 24b,c show the results of using keys $y_0 = [0.6, 0.1, 0.2, 0.5, 0.4]$ and $y_1 = [0.600000000000001, 0.1, 0.2, 0.5, 0.4]$ for encrypted ciphertext image Y_0 and Y_1, respectively. Figure 24d shows the result of the correct decryption of Y_0 using y_0. Figure 24e,f show the results of Y_0 and Y_1 using the wrong decryption keys y_1 and y_0, respectively. Figure 24 illustrates that, despite only minor changes between the keys y_0 and y_1, the ciphertext images Y_0 and Y_1 could be decrypted correctly using the corresponding keys y_1 and y_0.

To better evaluate the key sensitivity of the algorithm, we tested the *NPCR* and *UACI* values between the keys $y_0 = [0.6, 0.1, 0.2, 0.5, 0.4]$ and $y = [0.600000000000001, 0.1, 0.2, 0.5, 0.4]$ for the encrypted image using Equations (29) and (30). The test values in this study and those in the literature [4,14,17] are shown in Table 3. Table 3 shows that the algorithm had good test results, so the encryption algorithm proposed in this paper has good key sensitivity.

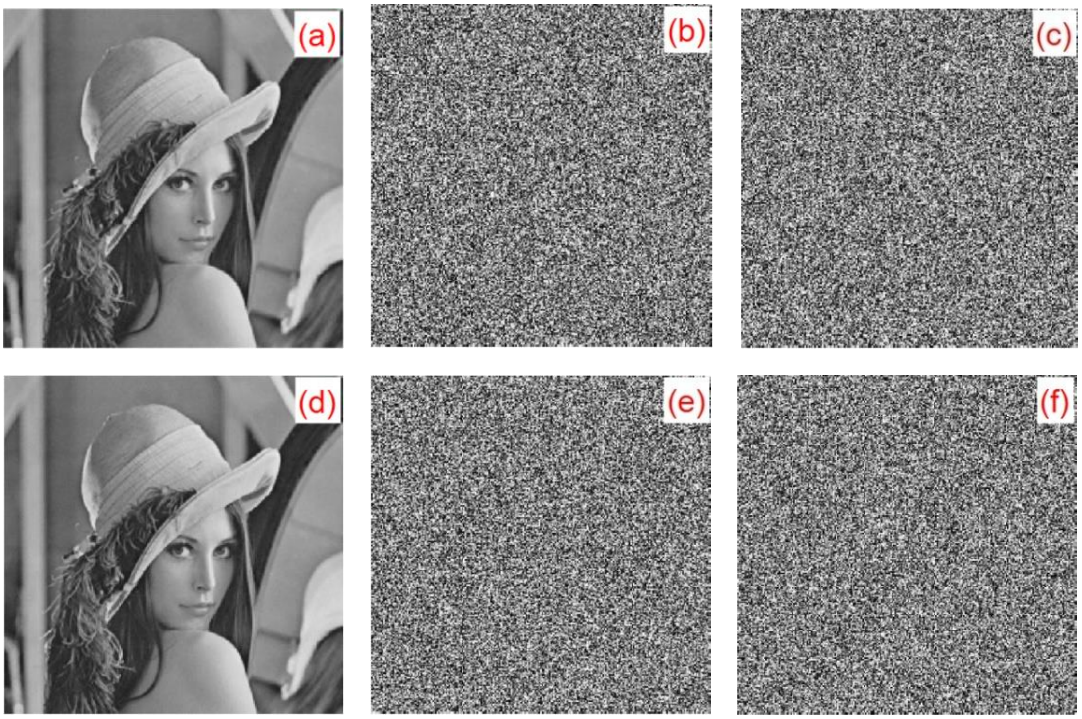

Figure 24. Key sensitivity experiment analysis chart: (**a**) plaintext image, (**b**) ciphertext Y_0 (key is y_0), (**c**) ciphertext Y_1 (key is y_1), (**d**) Y_0 Correct decryption result, (**e**) error decryption result Y_0 using y_1, and (**f**) error decryption result Y_1 using y_0.

Table 3. Key sensitivity test results.

Images	Lena		Baboon		Boat		Peppers	
Index	Pixel Change Rate (NPCR)	Normalized Mean Change Intensity (UACI)	NPCR	UACI	NPCR	UACI	NPCR	UACI
Test value	0.9961	0.3356	0.9962	0.3344	0.996	0.3363	0.9962	0.3357
Reference [4]	0.9961	0.3346	-	-	-	-	-	-
Reference [14]	0.9961	0.3346	-	-	-	-	0.9962	0.3341
Reference [17]	0.9952	0.3359	0.991	0.3325	0.9925	0.3339	0.985	0.3295

6.6. Information Entropy Analysis

Information entropy is an important indicator of randomness, which reflects the distribution of gray values of images. The more uniform the gray value distribution, the larger the information entropy of the image. The information entropy calculation formula of an image is shown below.

$$H(C) = -\sum_{i=1}^{L} p(x_i) \log_2 p(x_i) \tag{31}$$

where $p(x_i)$ is the probability of C and L is the total number of x_i. For grayscale images, the theoretical maximum value of information entropy is 8. The closer the image information entropy is to the theoretical maximum, the more random the image pixel gray value distribution is. The information entropy before and after the encryption of Lena, baboon, boat, peppers, couple, and leaf is shown in Table 4. The simulation results show that the pixel value distribution of the encrypted image was very uniform, and the algorithm had a good encryption effect.

Table 4. Information entropy analysis for plaintext and encrypted images.

Image	Lena	Baboon	Boat	Peppers	Couple
Original Image	7.4832	7.3713	7.1267	7.5715	7.2369
Encrypted Image	7.9978	7.9977	7.997	7.9974	7.9974

Considering the Lena image as an example, the information entropy after encrypting the image was compared with the information entropy in the literature [4,11–13,17]. The experimental results are shown in Table 5. Table 5 shows that the algorithm had a better encryption effect.

Table 5. Information entropy comparison.

Image	Original Image Information Entropy	Encrypted Image Information Entropy					
		Algorithm	Reference [4]	Reference [11]	Reference [12]	Reference [13]	Reference [17]
Lena	7.4832	7.9978	7.9967	7.9972	7.9900	7.9959	7.9975

Global and local entropy [22] values of the encrypted image are tabulated in Table 6. From Table 6, it is clear that both the entropy values are closer to the optimal theoretical values (≈8). Furthermore, the obtained results are compared against the critical values at 5%, 1%, and 0.1% significance. From the above discussion, it can be concluded that the proposed encryption algorithm possesses high randomness and is robust against a statistical attack.

Table 6. Global and local entropy analysis.

Image	Global Entropy	Local Entropy No. of Blocks = 20 (Block Size = 44 × 44)	Local Entropy Critical Values		
			$h^{l*0.05}$left = 7.9019 $h^{l*0.05}$right = 7.9030	$h^{l*0.01}$left = 7.9017; $h^{l*0.01}$right = 7.9032	$h^{l*0.001}$left = 7.9015l $h^{l*0.001}$right = 7.9034
Lena	7.9978	7.9028	Pass	Pass	Pass
Baboon	7.9977	7.9023	Pass	Pass	Pass
Boat	7.997	7.9027	Pass	Pass	Pass
Couple	7.9974	7.9022	Pass	Pass	Pass

6.7. Histogram Analysis

A histogram can reflect the distribution of image pixel values very well. The flatter the histogram is, the more uniform the pixel value distribution is. Figure 25 shows histograms of the original images for Lena, baboon, and boat, and encrypted images.

6.8. Histogram Statistics

The variance and standard deviation are metrics of dispersion implemented to support the results of visual inspection in graphic histograms. They measure how much the elements of a set of data vary with respect to each other around the mean. Two datasets may have the same average value (mean), but the variations can be drastically different [23].

The variance calculates the average difference between each of the values with respect to their central point (mean $\overline{x}$). This average is calculated by squaring each of the differences and calculating its mean again. The squaring process is used to eliminate the negative signs and to increase the variance of dispersed (non-uniform) datasets. On the other hand, the more uniform the graphic histogram is, the lower the histogram variance is, which is determined with the following expression.

$$\alpha = \frac{1}{256}\sum_{i=1}^{256}(x_i - \overline{x})^2 \tag{32}$$

where:

$$\overline{x} = \frac{M \times N}{256} \tag{33}$$

x is the frequency for each intensity value from 0–255 grayscale value of the histogram, α is the histogram variance, and $\overline{x}$ is the mean of the histogram. The standard deviation allows us to know the arithmetic average of fluctuations of the dataset with respect to the mean. It is determined with the square root of the histogram variance as follows.

$$\beta = \sqrt{\alpha} \tag{34}$$

where β is the standard deviation of the histogram. In Table 7, the histogram variance and its standard deviation are presented for plain and encrypted images. Table 7 shows that the pixel values of the image encrypted by this algorithm are more uniform. This algorithm has a better encryption effect.

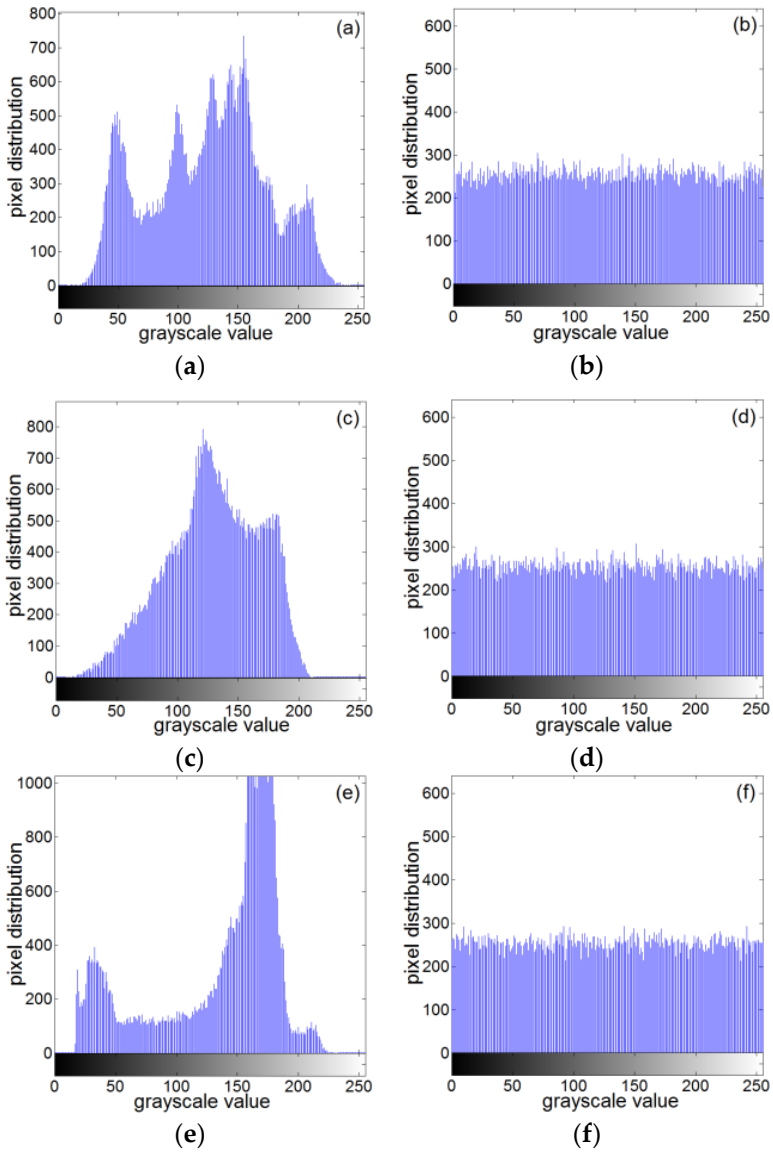

Figure 25. Histogram of the plaintext image and ciphertext image: (**a**) histogram of Lena plaintext, (**b**) histogram of Lena ciphertext, (**c**) histogram of baboon plaintext, (**d**) histogram of baboon ciphertext, (**e**) histogram of the clear text of the boat, and (**f**) histogram of the boat ciphertext.

Table 7. Histogram statistics with the variance and standard deviation of plain and encrypted images.

Plain Image	**Scale**	α	β
Lena 256 × 256	gray	37,963	195
Reference [22] (lena 256 × 256)	gray	38,451	196
Boat 256 × 256	gray	103,380	321.5
Baboon 256 × 256	gray	58,542	241.9
Couple 256 × 256	gray	79,457	281.9
Encrypted Image	**Scale**		
Lena 256 × 256	gray	230	15
Reference [22] (lena 256 × 256)	gray	414	20
Boat 256 × 256	gray	250	15.8
Baboon 256 × 256	gray	260	16.1
Couple 256 × 256	gray	242	15.6

6.9. Correlation Analysis of Adjacent Pixels

A feature of digital images is the strong correlation of adjacent pixels. To calculate the correlation of adjacent pixels before and after encryption, 5000 sets of adjacent pixels were randomly selected in the horizontal, vertical, and diagonal directions of the plaintext and ciphertext images. The horizontal, vertical, and diagonal correlation coefficients were calculated using Equations (35).

$$\begin{cases} E(x) = \frac{1}{K}\sum_{i=1}^{K} x_i \\ D(x) = \frac{1}{K}\sum_{i=1}^{K} (x_i - E(x))^2 \\ Cov(x,y) = \frac{1}{K}\sum_{i=1}^{K} (x_i - E(x))(y_i - E(y)) \\ r_{xy} = \frac{Cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \end{cases} \tag{35}$$

The test results are shown in Table 8. The pixel correlation of the Lena plaintext image and ciphertext image in the horizontal direction, vertical direction, and diagonal direction are shown in Figure 26.

Table 8. Test results for the correlation coefficient between plaintext and ciphertext images.

Images	**Horizontal Correlation Coefficient**		**Vertical Correlation Coefficient**		**Diagonal Direction Correlation Coefficient**	
	Clear Image	**Ciphertext Image**	**Clear Image**	**Ciphertext Image**	**Clear Image**	**Ciphertext Image**
Lena	0.971	0.012	0.9402	0.002	0.9121	−0.0083
Baboon	0.8343	−0.0109	0.8712	0.0043	0.794	−0.0074
Boat	0.9574	−0.0076	0.9533	−0.0137	0.915	0.0036

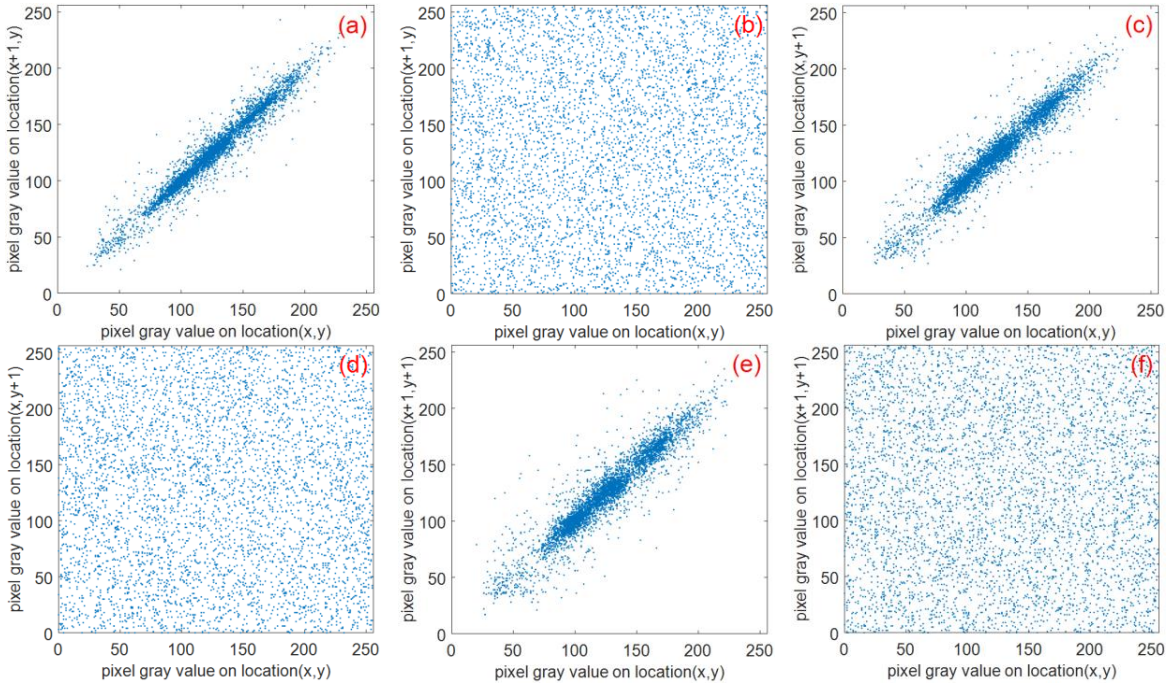

Figure 26. Correlation analysis of the three directions before and after Lena image encryption: (**a**) and (**b**) horizontally adjacent, (**c**) and (**d**) vertically adjacent, (**e**) and (**f**) diagonally adjacent.

6.10. Robustness Analysis

6.10.1. Quality Metrics Analysis

Quality evaluation of digital images can use the Mean Squared Error (*MSE*) and Peak Signal-to-Noise Ratio (*PSNR*) for measurement [23,24]. The *MSE* is a parameter to measure the difference between two images, which is described as Equation (36).

$$MSE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} (X(i,j) - (i,j))^2 \tag{36}$$

where $H \times W$ is the size of original image, $X(i,j)$ is the original image, and (i,j) is the encrypted image. The higher value of *MSE* represents better encryption quality. This *MSE* analysis is a useful test for a plain image and encrypted image with pixel values in the range of [0–255]. The *PSNR* (expressed in logarithmic scale and decibels) determines the ratio between the maximum possible power of a signal and the power of distorting noise that affects the quality of its representation. It is calculated by Equation (37).

$$PSNR = 20 \log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \tag{37}$$

The smaller the *MSE* value is, the larger the *PSNR* value is, which means that there is a high degree of similarity between the tested images. By calculation, the *MSE* between the original image and the decrypted image is 0, and the value of *PSNR* is Inf. In this algorithm, the *MSE* between the original Lena image and the decrypted image is 77,012, and *PSNR* is 9.265. The results show that the quality metrics of the tested images is good.

6.10.2. Occlusion Attack Analysis

In an occlusion attack, we choose 12.5%, 25%, and 50% of occlusion in an encrypted image. In Figure 27, the attack results are shown. For 12.5% of occlusion, the *MSE* value is 7853 and the *PSNR*

value is 9.1802. For 25% of occlusion, the *MSE* value is 10,148 and the *PSNR* value is 8.0672. For 50% of occlusion, the *MSE* value is 13,376 and the *PSNR* value is 6.8676. The results show that the proposed cryptographic algorithm can effectively resist occlusion attack.

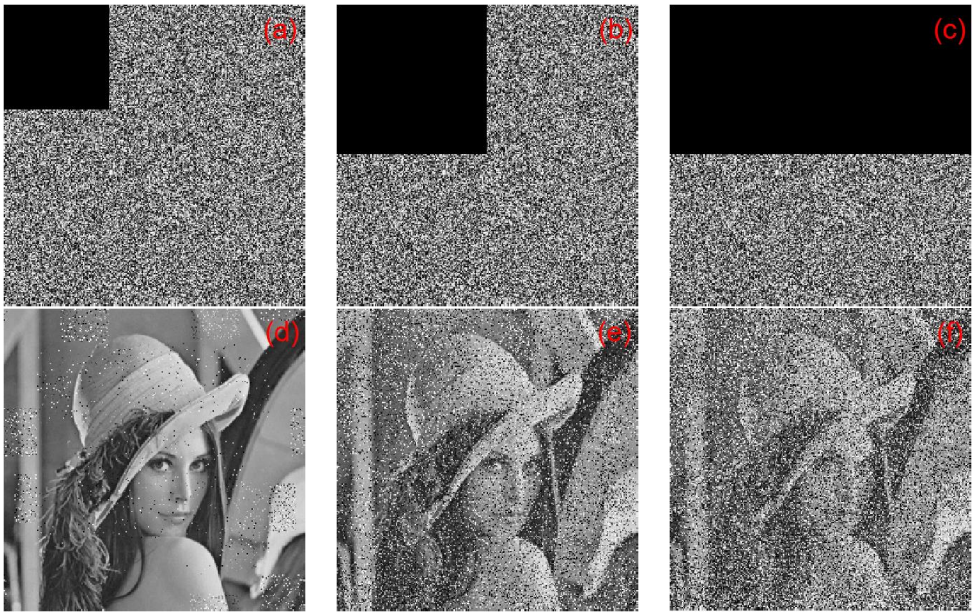

Figure 27. The results of occlusion attack: (**a**) encrypted with 12.5% occlusion, (**b**) encrypted with 25% occlusion, (**c**) encrypted with 50% occlusion, (**d**) decrypted with 12.5% occlusion, (**e**) decrypted with 25% occlusion, and (**f**) decrypted with 50% occlusion.

6.10.3. Noise Attack Analysis

In order to verify the anti-noise performance of the proposed algorithm, Salt and pepper noise with different intensities was added to the encrypted image. The intensities were 10, 15, and 20, respectively, and they were then decrypted. The results are shown in Figure 28. For 10 of intensity, the *MSE* value is 8758.9 and the *PSNR* value is 8.706. For 15 of intensity, the *MSE* value is 9302.9 and the *PSNR* value is 8.446. For 20 of intensity, the *MSE* value is 9866.8 and the *PSNR* value is 8.189. It can be seen that the original image can be basically recovered after the noise image is decrypted. Therefore, the proposed algorithm has a certain anti-noise attack capability.

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

Figure 28. The results of noise attack analysis: (**a**) noise with 10 of intensity, (**b**) noise with 15 of intensity, and (**c**) noise with 20 of intensity.

7. Conclusions

In this paper, a five-dimensional chaotic system was proposed, which had a simple structure and was easy to implement. Basic dynamic analysis of the system was conducted, including the

equilibrium point, phase diagram, bifurcation diagram, and power spectrum. Based on the theoretical analysis, a chaotic circuit was designed using the analog device amplifier TL082CD. The consistency of the numerical simulation results confirmed the feasibility of the method. Simultaneously, five chaotic sequences generated by the system were applied to the hybrid image encryption algorithm for physical chaotic encryption and advanced encryption standard encryption. In the algebraic encryption process, we performed convolution operations on five chaotic sequences, which was followed by convolution operations. The latter sequence was applied to the image scaled block encryption, and a numerical simulation experiment was conducted on the hybrid encryption system. The simulation results verified the correctness of the encryption algorithm. Therefore, the encryption algorithm proposed in this paper has a good application prospect in secure communication, particularly digital image encryption.

Author Contributions: Conceptualization, Z.Z.; Data curation, L.S.; Formal analysis, T.W. and M.W.; Methodology, T.W.; Software, T.W.; Validation, S.C.; Writing—original draft, T.W.; Writing—review & editing, T.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: Special Funds for "Double First-Class" Construction in Hubei Province supported the work.

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

References

1. Peng, Z.P.; Wang, C.H.; Yuan, L.; Luo, X.W. A novel four-dimensional multi-wing hyper-chaotic attractor and its application in image encryption. *Acta Phys. Sin. Chin. Ed.* **2014**, *63*, 506. [CrossRef]
2. Liu, Y. Study on Chaos Based Pseudorandom Sequence Algorithm and Image Encryption Technique. Ph.D. Thesis, Harbin Institute of Technology, Harbin, China, 2015.
3. Rui, L. New Algorithm for Color Image Encryption Using Improved 1D Logistic Chaotic Map. *Open Cybern. Syst. J.* **2015**, *9*, 210–216. [CrossRef]
4. Sun, S. A Novel Hyperchaotic Image Encryption Scheme Based on DNA Encoding, Pixel-Level Scrambling and Bit-Level Scrambling. *IEEE Photonics J.* **2018**, *10*, 1–14. [CrossRef]
5. Chai, X.; Gan, Z.; Zhang, M. A fast chaos-based image encryption scheme with a novel plain image-related swapping block permutation and block diffusion. *Multimed. Tools Appl.* **2016**, *76*, 15561–15585. [CrossRef]
6. Ahmad, J.; Khan, M.A.; Ahmed, F.; Khan, J.S. A novel image encryption scheme based on orthogonal matrix, skew tent map, and XOR operation. *Neural Comput. Appl.* **2017**, *30*, 3847–3857. [CrossRef]
7. Ahmad, J.; Hwang, S.O. Chaos-based diffusion for highly autocorrelated data in encryption algorithms. *Nonlinear Dyn.* **2015**, *82*, 1839–1850. [CrossRef]
8. Gao, T.; Chen, Z. Image encryption based on a new total shuffling algorithm. *Chaos Solitons Fractals* **2018**, *38*, 213–220. [CrossRef]
9. Pareek, N.K.; Patidar, V.; Sud, K.K. Image encryption using chaotic logistic map. *Image Vis. Comput.* **2006**, *24*, 926–934. [CrossRef]
10. Li, Y.; Tang, W.K.S.; Chen, G. Generating hyperchaos via state feedback control. *Int. J. Bifurc. Chaos* **2005**, *15*, 3367–3375. [CrossRef]
11. Li, Y.; Wang, C.; Chen, H. A hyper-chaos-based image encryption algorithm using pixel-level permutation and bit-level permutation. *Opt. Lasers Eng.* **2017**, *90*, 238–246. [CrossRef]
12. Zhang, X.; Feng, H.; Ying, N. Chaotic image encryption algorithm based on bit permutation and dynamic DNA encoding. *Comput. Intell. Neurosci.* **2017**, *2017*. [CrossRef]
13. Liu, J.; Yang, D.; Zhou, H.; Chen, S. A digital image encryption algorithm based on bit-planes and an improved logistic map. *Multimed. Tools Appl.* **2017**, *77*, 10217–10233. [CrossRef]
14. Qi, Y.; Wang, C. A New Chaotic Image Encryption Scheme Using Breadth-First Search and Dynamic Diffusion. *Int. J. Bifurc. Chaos* **2018**, *28*, 475. [CrossRef]
15. Assad, S.E.; Farajallah, M. A new chaos-based image encryption system. *Signal Process. Image Commun.* **2015**, *41*, 144–157. [CrossRef]

16. Çavuşoğlu, Ü.; Panahi, S.; Akgül, A.; Jafari, S.; Kaçar, S. A new chaotic system with hidden attractor and its engineering applications: Analog circuit realization and image encryption. *Analog Integr. Circuits Signal Process.* **2019**, *98*, 85–99. [CrossRef]
17. Enayatifar, R.; Abdullah, A.H.; Isnin, I.F.; Altameem, A.; Lee, M. Image encryption using a synchronous permutation-diffusion technique. *Opt. Lasers Eng.* **2017**, *90*, 146–154. [CrossRef]
18. Chen, J.X.; Zhu, Z.-L.; Fu, C.; Zhang, L.-B.; Zhang, Y. An image encryption scheme using nonlinear inter-pixel computing and swapping based permutation approach. *Commun. Nonlinear Sci. Numer. Simul.* **2015**, *23*, 294–310. [CrossRef]
19. Zhang, Y.; Li, C.; Li, Q.; Zhang, D.; Shu, S. Breaking a chaotic image encryption algorithm based on perceptron model. *Nonlinear Dyn.* **2012**, *69*, 1091–1096. [CrossRef]
20. Sprott, J. Some simple chaotic flows. *Phys. Rev. E* **1994**, *50*, 647–650. [CrossRef]
21. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
22. Ravichandran, D.; Praveenkumar, P.; Rayappan, J.B.B.; Amirtharajan, R. DNA Chaos Blend to Secure Medical Privacy. *IEEE Trans. Nanobiosci.* **2017**, *16*, 850–858. [CrossRef] [PubMed]
23. Murillo-Escobar, M.A.; Meranza-Castillón, M.O.; López-Gutiérrez, R.M.; Cruz-Hernández, C. Suggested Integral Analysis for Chaos-Based Image Cryptosystems. *Entropy* **2019**, *21*, 815. [CrossRef]
24. Li, S.; Ding, W.; Yin, B.; Zhang, T.; Ma, Y. A Novel Delay Linear Coupling Logistics Map Model for Color Image Encryption. *Entropy* **2018**, *20*, 463. [CrossRef]