Hindawi
Complexity
Volume 2022, Article ID 8765268, 15 pages
https://doi.org/10.1155/2022/8765268

WILEY | Hindawi

Research Article

Development of a Novel Hyperchaos-Based Image Encryption Algorithm Consisting of Two Scrambling-Diffusion Operations

Yongzhong Huang,[1] Xueguang Bi,[1] Yucheng Liu ,[2] and Yuxia Li[1]

[1]*Department of Physics and Telecommunication Engineering, Yulin Normal University, Yulin, Guangxi 537000, China*
[2]*Department of Mechanical Engineering, South Dakota State University, Brookings, SD 57007, USA*

Correspondence should be addressed to Yucheng Liu; yucheng.liu@sdstate.edu

Received 22 June 2022; Revised 24 July 2022; Accepted 12 August 2022; Published 4 October 2022

Academic Editor: Jesus M. Munoz-Pacheco

This paper proposes a hyperchaos-based image encryption algorithm, which consists of two scrambling-diffusion operations and one scrambling operation. In the proposed algorithm, the initial conditions of a hyperchaotic Chen system are first generated using the Message Digest 5 (MD5) value of digital images and given initial values, and then the images will be encrypted using the keystream created by the system. Since the initial values of hyperchaotic Chen systems are related to plaintext and the encryption process is related to the images, this algorithm is able to effectively protect images against selective plaintext attacks. Simulation results demonstrate that the present algorithm offers enhanced encryption performance, high security, and strong resistance to known attacks. Therefore, it may find wide application in image encryption transmission. Compared to other image encryption algorithms, the proposed algorithm uses different keystreams when encrypting different images and is capable of effectively resisting various plain image and differential attacks faster.

1. Introduction

Digital images have been widely used as a major carrier of network information. The past decade has seen the rapid development of cloud computing, big data, and communication technology. However, the security of image transmission has raised noticeable awareness and has become a high-priority topic in the field of information security. Neighboring pixels in an image exhibit a high level of correlation, and the image itself contains a large amount of pictorial data. Because of that, conventional text-based encryption schemes like the data encryption standard (DES) or advanced encryption standard (AES) are not applicable to image encryption. It is therefore a critical task to design new algorithms that can be used for image encryption.

Chaotic systems are nonlinear and have inherent characteristics like high sensitivity to control parameters' initial conditions, internal randomness, unpredictability, and ergodicity. Therefore, pseudo-random sequences generated from chaotic systems are random, are computationally unpredictable, have a weak correlation, and possess a large parameter space. This property makes chaotic systems particularly competent for image encryption. Since Fridrich put forward a symmetric block encryption algorithm on the basis of invertible 2D chaotic maps in 1989 [1], many chaotic-based image encryption algorithms have been developed.

For instance, [2] proposed a real-time secure symmetric encryption algorithm that used a 3D cat map to rearrange all the pixels of the original images and another chaotic map to mystify the relationship between the plain image and the cipher image, thereby increasing the immunity against statistical and differential attacks. The authors of [3] developed an image encryption approach based on chaotic logistic maps, which employed an 80-bit external secret key and two chaotic logistic maps. The authors of [4] designed a modified image encryption using coupled map lattices and substitution box transformation. In that approach, the positions of image pixels were mixed up by using a chaotic tent map; coupled map lattices and the S-box transformation

were also employed to confuse the correlation between the original and the encrypted images. The authors of [5] put forward an image encryption scheme based on a quantum chaotic map and diffusion-permutation architecture. In their approach, the quantum chaotic map and the 2D logistic map were separately coupled with nearest-neighboring coupled-map lattices to achieve high complexity and randomness in the generated keystreams. The authors of [6] developed a lossless encryption method for color images by means of a 6D hyperchaotic system and the 2D discrete wavelet transform (DWT). Their scheme was established based on the 2D DWT and 6D hyperchaotic systems in both the spatial and frequency domains, where the key streams hinge on both the hyperchaotic system and the plain image.

The authors of [7] presented an encryption scheme of linear-nonlinear-linear structure that employed total shuffling. Their system was able to generate a 1D chaotic system with enhanced chaotic performances and broader chaotic ranges by comparison with previous chaotic maps. The authors of [8] presented an image encryption method that enhanced the link between value changing for grayness and position shuffling for pixels. The authors of [9] developed a new image encryption algorithm, SPRING, applying lightweight chaotic maps and simple arithmetic and logical operations. The authors of [10] put forward a digital image encryption scheme via dynamic deoxyribonucleic acid coding and chaotic operations, making use of hyper-digital chaos in the frequency domain. In that algorithm, both the phase and amplitude components in the frequency domain were scrambled and diffused. The authors of [11] presented a 2D logistic-modulated-sine-coupling-logistic chaotic map (LSMCL) and used that map to design an image encryption scheme consisting of two rounds of permutation and diffusion operation. The authors of [12] put forward a composite chaotic map and a way of utilizing optimization approach to enhance the performance of encryption schemes. The authors of [13] applied three chaotic sequences to acquire a high level of encryption in their scheme to conduct both permutation and substitution processes of image encryption. The authors of [14] first obtained a memristive neuron model by coupling a memristor into an advanced neuron model with key chaotic characteristics. After that, they proposed a new encryption algorithm to apply the memristive neuron for image encryption. Lai, Zhang, and other coworkers [15] also designed a novel Hopfield neural network (HNN) that can yield multiscroll attractors by using a new memristor as a synapse in the HNN. They finally developed a 3D image encryption scheme based on the proposed memristive and confirmed its outstanding encryption performance.

However, lots of chaos-based image encryption schemes are in fact not as secure as their developers claimed and cannot resist selective and known plaintext attacks. For example, the images encrypted using the schemes developed by [16, 17], and [18] were decoded by [19, 20], and [21] through plaintext attacks, respectively. Those encrypted images were deciphered because the secret keys were not related to the original plain images and the generated chaotic sequences to encrypt image data were constant. Those security defects made the aforementioned algorithms [16, 17]; and [18] as well as other encryption algorithms designed in similar manners [22, 23] vulnerable to plaintext attacks.

To overcome the abovementioned shortcomings, this paper presents a hyperchaos-based image encryption scheme that has a scrambling-diffusion structure. In the present algorithm, the generated secret key is related to the original plain image and the encryption process is related to the image. Simulation results have verified that the presented scheme possesses a high degree of resistance against various plaintext attacks. The remainder of this paper is structured as follows. Section 2 reviews the hyperchaotic Chen system adopted in this study. Section 3 illustrates the proposed algorithm, its structure, and presents the simulation results to verify its encryption performance. Section 4 discusses the characteristics of the algorithm by means of key space, key sensitivity, histograms, correlation coefficient, information entropy, and resistance to differential attacks. Section 5 concludes the paper.

2. Hyperchaotic Chen System

The hyperchaotic Chen system was presented by [24] from the Chen system via a dynamical controller. This system was chosen for designing the image encryption algorithm because it has notable dynamical properties and can generate pseudo-random sequences with excellent statistical properties. The equations describing the dynamics of the hyperchaotic Chen system can be written as follows [25]:

$$\begin{cases} \dot{x} = a(y - x), \\ \dot{y} = -xz + dx + cy - w, \\ \dot{z} = xy - bz, \\ \dot{w} = x + k. \end{cases} \tag{1}$$

When the control parameters $a = 36$, $b = 3$, $c = 28$, $d = 16$, and $-0.7 \leq k \leq 0.7$, the system is in hyperchaotic status. In this study, k is set as 0.2. Figure 1 shows the attractor of this hyperchaotic system.

3. Image Encryption Algorithm

The image encryption process is illustrated in Figure 2. At first, the Message Digest 5 (MD5) hash value of the plain image is acquired, based on which the initial conditions of the hyperchaotic Chen system are determined. Next, the system generates chaotic sequences in an image encryption process. The generated sequences are then used for scrambling and diffusing the pixels or bits of the original image, and the entire encryption process is completed after two rounds of high-speed scrambling and pixel adaptive diffusion and a bit-plane scrambling step.

3.1. Generation and Testing of Key Stream. We assume that the size of a plain image P is $M \times N$, the steps to generate a key stream are as follows:

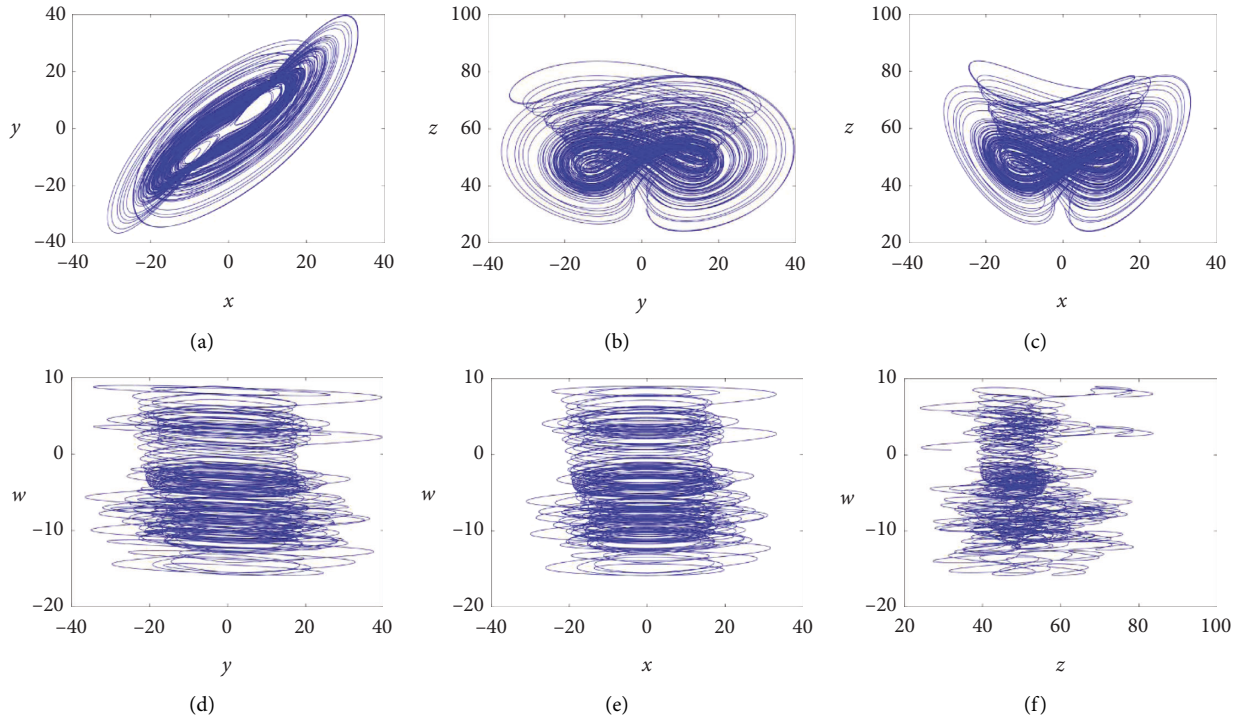


Figure 1: Attractor of the hyperchaotic chen system: (a) x-y plane; (b) y-z plane; (c) x-z plane; (d) y-w plane; (e) x-w plane; (f) z-w plane.

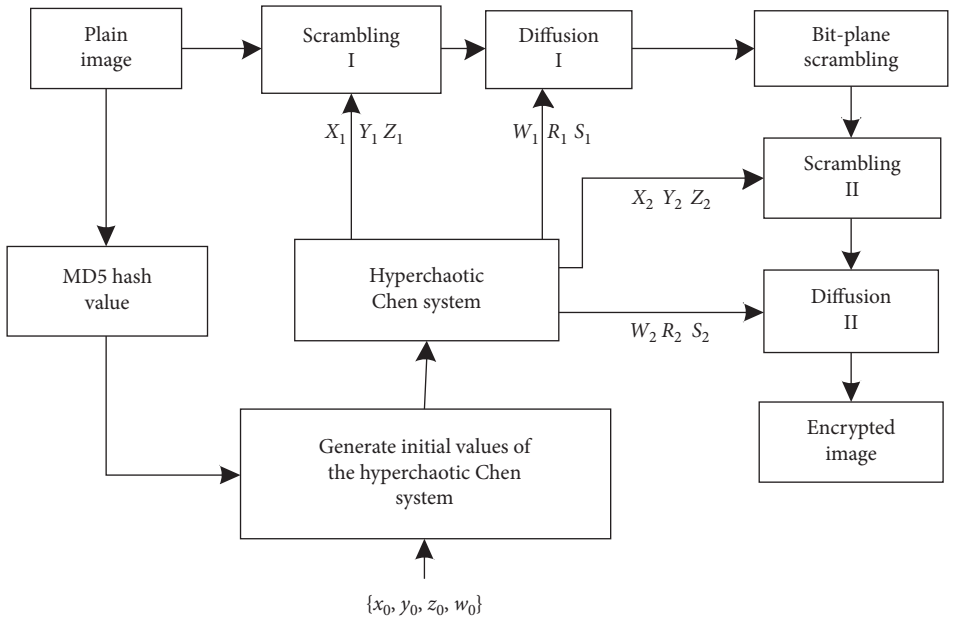


Figure 2: Image encryption scheme.

Step 1: Generate a 128 bit secret key I from the plain image P using the MD5 algorithm.

Step 2: Divide I into 32 4-bit-long groups, which is $I = i_1, i_2, i_3, \ldots, i_{32}$.

Step 3: Determine the initial conditions of the hyperchaotic Chen system $\{x_1, y_1, z_1, w_1\}$ based on given initial values $\{x_0, y_0, z_0, w_0\}$ using equation (2). This set of initial conditions will be used for scrambling and diffusion operation I.

$$\begin{cases} x_1 = x_0 + \dfrac{i_1 \oplus i_2 \oplus i_3 \oplus i_4}{16}, \\ y_1 = y_0 + \dfrac{i_5 \oplus i_6 \oplus i_7 \oplus i_8}{16}, \\ z_1 = z_0 + \dfrac{i_9 \oplus i_{10} \oplus i_{11} \oplus i_{12}}{16}, \\ \omega_1 = \omega_0 + \dfrac{i_{13} \oplus i_{14} \oplus i_{15} \oplus i_{16}}{16}. \end{cases} \tag{2}$$

Step 3: Substitute $\{x_1, y_1, z_1, w_1\}$ into the hyperchaotic Chen system (equation (1)), iterate 500 times, and then continue to iterate other $M \times N$ times to obtain a sequence $x_1(i)$, $y_1(i)$, $z_1(i)$, and $w_1(i)$, where $i = 1, 2, 3, \ldots, M \times N$.

Step 4: Processe the generated sequence using equation (3) to obtain $M \times N$ matrices X_1, Y_1, Z_1, W_1, R_1, and S_1.

$$\begin{cases} X_1(i) = \text{floor}\left(\text{mod}\left(200 + x_{1i} \times 10^{14}, M\right)\right), \\ Y_1(i) = \text{floor}\left(\text{mod}\left(200 + y_{1i} \times 10^{14}, N\right)\right), \\ Z_1(i) = \text{floor}\left(\text{mod}\left(200 + z_{1i} \times 10^{14}, 256\right)\right), \\ W_1(i) = \text{floor}\left(\text{mod}\left(200 + x_{1i} + z_{1i} \times 10^{14}, 256\right)\right), \\ R_1(i) = \text{floor}\left(\text{mod}\left(200 + x_{1i} + y_{1i} \times 10^{14}, 256\right)\right), \\ S_1(i) = \text{floor}\left(\text{mod}\left(200 + w_{1i} \times 10^{14}, M + N\right)\right). \end{cases} \tag{3}$$

In equation (3), floor(x) returns the largest integer that is smaller than or equal to x, while mod(x, y) returns the remainder of x divided by y.

Step 5: Determine another set of initial conditions $\{x_2, y_2, z_2, w_2\}$ for the Chen system based on $\{x_0, y_0, z_0, w_0\}$ using equation (4). This set of initial conditions will be used for scrambling and diffusion operation II.

$$\begin{cases} x_2 = x_0 + \dfrac{i_{17} \oplus i_{18} \oplus i_{19} \oplus i_{20}}{16}, \\ y_2 = y_0 + \dfrac{i_{21} \oplus i_{22} \oplus i_{23} \oplus i_{24}}{16}, \\ z_2 = z_0 + \dfrac{i_{25} \oplus i_{26} \oplus i_{27} \oplus i_{28}}{16}, \\ \omega_2 = \omega_0 + \dfrac{i_{29} \oplus i_{30} \oplus i_{31} \oplus i_{32}}{16}. \end{cases} \tag{4}$$

Step 6: Repeat step 3 to generate another sequence obtain a sequence $x_2(i)$, $y_2(i)$, $z_2(i)$, and $w_2(i)$, where $i = 1, 2, 3, \ldots, M \times N$.

Step 7: Repeat step 4 to obtain another set of $M \times N$ matrices X_2, Y_2, Z_2, W_2, R_2, and S_2 using equation (5).

$$\begin{aligned} X_1(i) &= \text{floor}\left(\text{mod}\left(200 + x_{2i} \times 10^{14}, M\right)\right), \\ Y_1(i) &= \text{floor}\left(\text{mod}\left(200 + y_{2i} \times 10^{14}, N\right)\right), \\ Z_1(i) &= \text{floor}\left(\text{mod}\left(200 + z_{2i} \times 10^{14}, 256\right)\right), \\ W_1(i) &= \text{floor}\left(\text{mod}\left(200 + x_{2i} + z_{2i} \times 10^{14}, 256\right)\right), \\ R_1(i) &= \text{floor}\left(\text{mod}\left(200 + x_{2i} + y_{2i} \times 10^{14}, 256\right)\right), \\ S_1(i) &= \text{floor}\left(\text{mod}\left(200 + w_{2i} \times 10^{14}, M + N\right)\right). \end{aligned} \tag{5}$$

Next, a statistical test suite put forward by [26] was applied to test a key stream generated by the hyperchaotic Chen system following the above steps to determine its randomness. The plain image chosen for this experiment is "Lena", whose size is 512×512. The initial values of this hyperchaotic Chen system are: $x_0 = 0.3838$, $y_0 = 0.9876$, $z_0 = 32.1234$, and $w_0 = 0.6565$. Test results are displayed in Table 1, from which it can be determined that the key stream generated by this system possesses excellent randomness.

3.2. Scrambling Operation I. The scrambling operation I is pixel-related, which scrambles the pixels in the original image through two scrambling processes. $M + N$ pixels are selected from the image and the remaining pixels are scrambled in the first process, while the $M + N$ selected pixels will be scrambled in the second process.

The steps of the first scrambling process are:

Step 1: We randomly select $M + N$ pixels from the image P, which will not be scrambled in this process; then use matrices X_1 and Y_1 to determine the locations of the selected pixels $P(b(i), c(i))$, $i = 1, 2, 3, \ldots, M + N$. The values of b and c are calculated using equations (6) and (7), respectively.

$$\begin{aligned} b(i) &= X_1(i, 1), \quad i = 1, 2, 3, \ldots, M, \\ b(j + M) &= X_1(1, j), \quad j = 1, 2, 3, \ldots, N, \end{aligned} \tag{6}$$

$$\begin{aligned} c(i) &= Y_1(i, 1), \quad i = 1, 2, 3, \ldots, M, \\ c(j + M) &= Y_1(1, j), \quad j = 1, 2, 3, \ldots, N. \end{aligned} \tag{7}$$

Step 2: Solve the information entropy (H_1) of the remaining pixels of the image (excluding the selected $M + N$ pixels $P(b(i), c(i))$ using equation (8); then find h_1 from H_1 based on equation (9).

$$H_1(m) = \sum_{i=0}^{2^N - 1} p(m_i) \log \frac{1}{p(m_i)}. \tag{8}$$

In (8), m_i represents the grayscale value of each pixel of the image; $p(m_i)$ represents the probability of the occurrence of the value m_i in the image; and N denotes the value's number of bits.

Table 1: Results of a randomness test.

Items	P-values	Results
Approximate entropy	0.684325	Success
Block frequency	0.201642	Success
Cumulative sums (forward)	0.448463	Success
Fast Fourier transform (FFT)	0.742726	Success
Frequency	0.640642	Success
Linear complexity	0.998936	Success
Longest run	0.503063	Success
Nonoverlapping template	0.829020	Success
Overlapping template	0.946675	Success
Rank	0.903588	Success
Runs	0.749820	Success
Serial P-value 1	0.517025	Success
Serial P-value 2	0.824740	Success
Universal	0.355340	Success
Random excursions $X = -4$	0.292475	Success
Random excursions $X = -3$	0.665211	Success
Random excursions $X = -2$	0.201510	Success
Random excursions $X = -1$	0.785998	Success
Random excursions $X = 1$	0.056035	Success
Random excursions $X = 2$	0.567424	Success
Random excursions $X = 3$	0.375432	Success
Random excursions $X = 4$	0.802170	Success
Random excursions variant $X = -9$	0.993779	Success
Random excursions variant $X = -8$	0.874679	Success
Random excursions variant $X = -7$	0.820132	Success
Random excursions variant $X = -6$	0.564102	Success
Random excursions variant $X = -5$	0.301069	Success
Random excursions variant $X = -4$	0.144795	Success
Random excursions variant $X = -3$	0.078193	Success
Random excursions variant $X = -2$	0.150287	Success
Random excursions variant $X = -1$	0.296088	Success
Random excursions variant $X = 1$	0.074375	Success
Random excursions variant $X = 2$	0.135122	Success
Random excursions variant $X = 3$	0.284107	Success
Random excursions variant $X = 4$	0.841091	Success
Random excursions variant $X = 5$	0.588381	Success
Random excursions variant $X = 6$	0.415502	Success
Random excursions variant $X = 7$	0.284619	Success
Random excursions variant $X = 8$	0.233579	Success
Random excursions variant $X = 9$	0.167543	Success

$$h_1 = \text{floor}\left(H_1 \times 10^5\right). \tag{9}$$

Step 3: Shift the location of pixels $P(i, j)$ and $P(u, v)$, determine the location u and v using equations (10) and (11).

$$u = X_1(i, j) + P(b(i), c(i)) + h_1 \text{mod} M, \quad i = 1, 2, 3, \ldots M; \quad j = 1, 2, 3, \ldots N, \tag{10}$$

$$v = Y_1(i, j) + P(b(j + M), c(j + M)) + h_1 \text{mod} N, \quad i = 1, 2, 3, \ldots M; \quad j = 1, 2, 3, \ldots N. \tag{11}$$

If u is solved as 0 based on (10), then we set $u = X_1(i, j)$; likewise, if v is solved as 0 from (11), it will be set as $v = Y_1(i, j)$.

Step 4: Complete the location change if both $P(i, j)$ and $P(u, v)$ do not belong to the randomly selected $M + N$ pixels $P(b, c)$. In another word, if $i = b(k)$, $j = c(k)$ or

$u = b(k), v = c(k)$, $k = 1, 2, 3, \ldots, M+N$, then the positions of $P(i, j)$ and $P(u, v)$ will not be changed.

The image obtained after the first scrambling process is denoted as E.

The second scrambling process is to change the locations of the $M+N$ pixels $E(b(i), c(i))$ that have not been scrambled in the first process, which includes the following steps:

Step 1: Form a vector F with the $M+N$ pixels $F(i) = \{E(b(i), c(i))\}$, $i = 1, 2, 3, \ldots, M+N$.

Step 2: Find the information entropy of $\{F\}$, H_2, using equation (8), and obtain h_2 from H_2 following the same approach described in equation (9): $h_2 = \text{floor}(H_2 \times 10^2)$.

Step 3: Chang the location of $F(i)$ to $F(t)$ by means of matrix S_1 (established in equation (3)). t can be determined using equation (12)

$$t = S_1(i) + h_2 \bmod (M+N), \quad i = 1, 2, 3, \ldots, M+N. \tag{12}$$

If t is found to be 0 from (12), then let $t = S_1(i)$.

Step 4: Plac the scrambled $F(i)$ back into the image E and update the $M+N$ pixels in E as $E(b(i), c(i)) = F(i)$, $i = 1, 2, 3, \ldots, M+N$.

3.3. Diffusion Operation I. The diffusion operation I mixes modulo addition and bitwise exclusive or (XOR) operations as suggested by [2]. Its main steps are:

Step 1: Obtain an image matrix G by conducting an XOR diffusion operation with the help of matrix Z_1 (equation (3))

$$G(i, j) = Z_1(i, j) \oplus E(i, j), \quad i = 1, 2, 3, \ldots, M, \quad j = 1, 2, 3, \ldots, N. \tag{13}$$

Step 2: Represent G as a row vector GV, and matrix R_1 (established in equation (3)) as another row vector RV, then conduct a modulo operation to obtain J (equation (14)).

$$J(1) = (RV(1) + GV(1)) \bmod 256, \; J(i) = (J(i-1) + RV(i) + GV(i)) \bmod 256, \quad i = 2, 3, \ldots, MN. \tag{14}$$

Step 3: Represent matrix W_1 (equation (3) as a row vector WV and use it for another modulo operation to obtain diffused image data K, as illustrated in equation (15):

$$K(MN) = (WV(MN) + J(MN)) \bmod 256, \quad K(i) = (K(i+1) + WV(i) + J(i)) \bmod 256, \quad i = MN-1, \; MN-2, \; MN-3, \ldots, 2, 1. \tag{15}$$

3.4. Bit-Plane Scrambling Operation. The image's pixel values are represented in an 8 bit binary format. We first obtain the 8 bit binary value of each pixel in K and rearrange those binary values to obtain the position-scrambled image data (Figure 3).

The method of acquiring the 8-bit binary value of the i^{th} pixel involves the following steps: $K(i) = \{b_{i8}, b_{i7}, b_{i6}, b_{i5}, b_{i4}, b_{i3}, b_{i2}, b_{i1}\}$, $i = 1, 2, 3, \ldots$, MN. Next, a new set of image data $L_1(i)$ can be obtained by scrambling $K(i)$ via the process illustrated in Figure 2 as $L_1(i) = \{b_{i4}, b_{i3}, b_{i2}, b_{i1}, b_{i7}, b_{i6}, b_{i5}, b_{i8}\}$, $i = 1, 2, 3, \ldots$, MN. Decimal values of $L_1(i)$ can then be easily obtained as

$$L(i) = b_{i4} \times 2^7 + b_{i3} \times 2^6 + b_{i2} \times 2^5 + b_{i1} \times 2^4 + b_{i7} \times 2^3 + b_{i6} \times 2^2 + b_{i5} \times 2^1 + b_{i8} \times 2^0 \quad i = 1, 2, 3, \ldots, MN. \tag{16}$$

The $L(i)$ is a $M \times N$ matrix and the image K is converted to image A after this operation.

3.5. Scrambling Operation II. The scrambling operation II is related to the plaintext. All the pixels in the image A will be scrambled in this operation through two processes. Similar to the scrambling operation I, in the first scrambling process, $M+N$ pixels are selected from A and the remaining pixels will be scrambled. In the second process, the $M+N$ selected pixels will be scrambled.

The first scrambling process consists of following steps:

Step 1: Randomly select $M+N$ pixels from A and the selected pixels will not be scrambled in this process. The locations of those pixels $(d(i), f(i))$ are determined by means of matrices X_2 and Y_2 (equation (5)) using the following equations:

$$\begin{aligned} d(i) &= X_2(i, 1), \quad i = 1, 2, 3, \ldots, M, \\ d(j+M) &= X_2(1, j), \quad j = 1, 2, 3, \ldots, N, \\ f(i) &= Y_2(i, 1), \quad i = 1, 2, 3, \ldots, M, \\ f(j+M) &= Y_2(1, j), \quad j = 1, 2, 3, \ldots, N. \end{aligned} \tag{17}$$

The selected pixels $A(d(i), f(i))$, $i = 1, 2, 3, \ldots, M+N$ will not be scrambled in the first scrambling process.

Step 2: Find the mean value and mix for image A and obtain mixA using equation (19).

$$\text{mix} = \frac{1}{MN} \sum_{i=0}^{MN} A(i), \quad \text{mix}A = \text{mix} \times 10^5. \tag{18}$$

Step 3: Shift the locations of the pixel $A(i, j)$ and $A(g, h)$, where g and h are determined by the following equations:

$$\begin{aligned} g &= X_2(i, j) + A(d(i), f(i)) + \text{mix}A \bmod M, \quad i = 1, 2, 3, \ldots M; \quad j = 1, 2, 3, \ldots N, \\ h &= Y_2(i, j) + A(d(j), f(j)) + \text{mix}A \bmod N, \quad i = 1, 2, 3, \ldots M; \quad j = 1, 2, 3, \ldots N. \end{aligned} \tag{19}$$

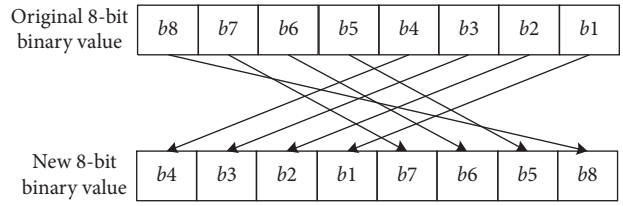


Figure 3: The process diagram of the bit-plain scrambling operation.

If g is found to be zero, then let $g = X_2(i, j)$; likewise, if h is found to be zero then $h = Y_2(i, j)$.

Step 4: Complete the location change if both $A(i, j)$ and $A(g, h)$ do not belong to the randomly selected $M + N$ pixels $A(d, f)$. In another word, if $i = d(k)$, $j = f(k)$ or $u = d(k), v = f(k)$, $k = 1, 2, 3, \ldots, M + N$, then the positions of $A(i, j)$ and $A(g, h)$ will not be shifted.

The image obtained after the first scrambling process is denoted as B.

The second scrambling process is to scramble the $M + N$ pixels $B(d(i), f(i))$ that have not been scrambled in the first process. The steps of this process are listed below:

Step 1: Form a vector C with the $M + N$ pixels $C(i) = \{B(d(i), f(i))\}$, $i = 1, 2, 3, \ldots, M + N$.

Step 2: Find the mean value and mix of C, and obtain mixC following the same approach depicted in

$$\text{mix} = \frac{1}{MN}\sum_{i=0}^{MN} C(i), \quad \text{mix}C = \text{mix} \times 10^5. \tag{20}$$

Step 3: Shift the locations of $C(i)$ and $C(q)$ by means of matrix S_2 (established in equation (5)). q can be determined based on

$$q = S_2(i) + \text{mix}C \bmod (M + N), \quad i = 1, 2, 3, \ldots M + N. \tag{21}$$

If q is found to be 0 from (21), then let $q = S_2(i)$.

Step 4: Place the scrambled $C(i)$ back into the image and update the $M + N$ pixels in B as $B(d(i), f(i)) = C(i)$, $i = 1, 2, 3, \ldots, M + N$.

3.6. Diffusion Operation II. The diffusion operation II is similar to operation I (Section 3.3) and consists of the following steps:

Step 1: Obtain an image matrix D through a modulo operation using matrix Z_2 (equation (5)):

$$D(i,j) = Z_2(i, j) + B(i, j) \bmod 256, \quad i = 1, 2, 3, \ldots M, \quad j = 1, 2, 3, \ldots N. \tag{22}$$

Step 2: Represent D as a row vector DK, matrix R_2 (established in equation (5)) as another row vector RK and perform an XOR diffusion operation to obtain Q.

$$Q(1) = RK(1) \oplus DK(1), \quad Q(i) = Q(i-1) \oplus RK(i) \oplus DK(i), \quad i = 2, 3, \ldots, MN. \tag{23}$$

Step 3: Represent matrix W_2 (equation (5) as a row vector WK and use it for another XOR operation to obtain T:

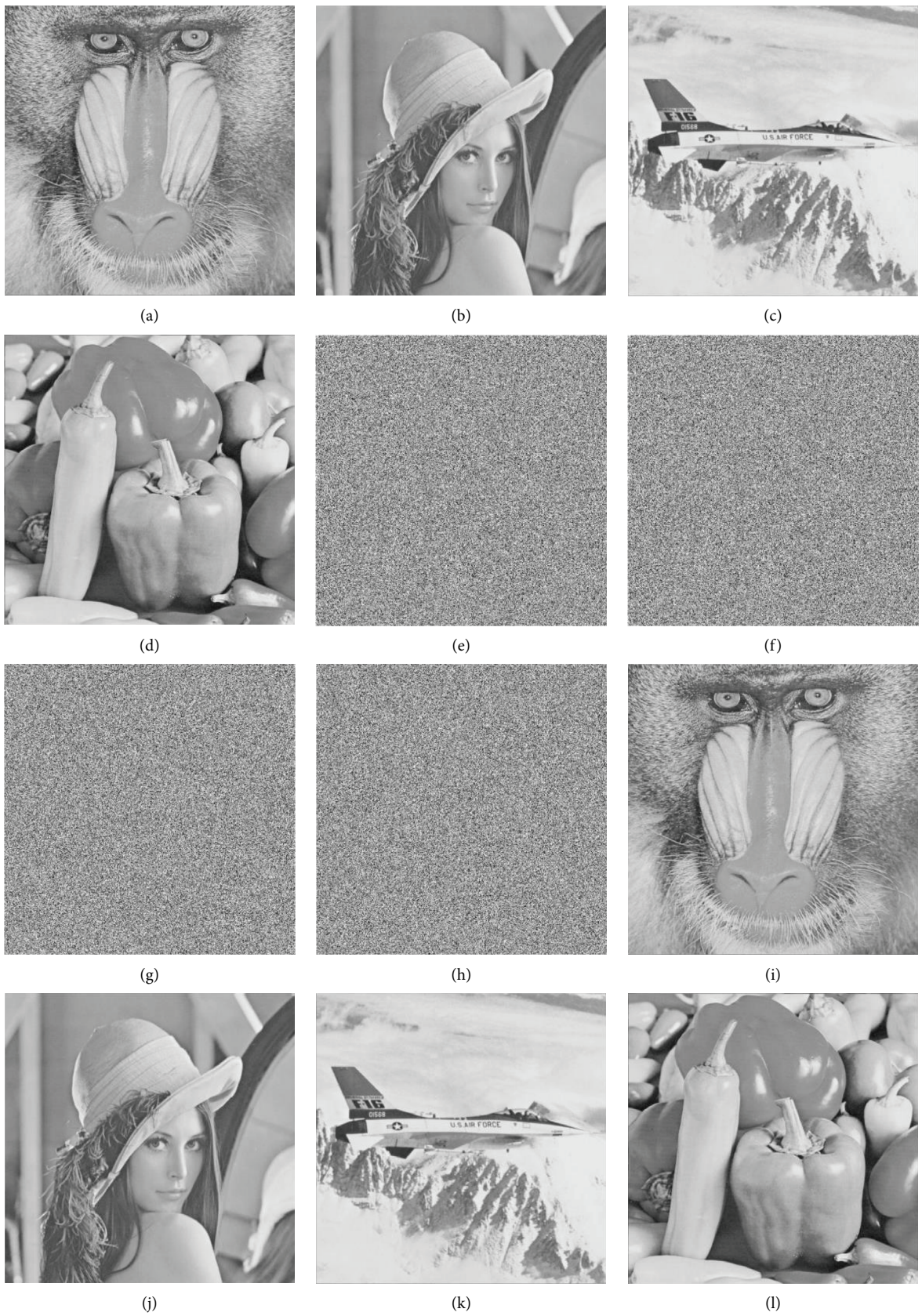
$$T(MN) = WK(MN) \oplus Q(MN), \quad K(i) = (K(i+1) + WV(i) + J(i)) \bmod 256, \quad i = MN - 1, \quad MN - 2, \quad MN - 3, \ldots, 2, 1. \tag{24}$$

The result T is then converted to a $M \times N$ matrix TL.

3.7. Simulation Results. The proposed image encryption algorithm was then applied to encrypt and decrypt four grayscale images (baboon, Lena, airplane, and pepper) to assess its encryption performance. The size of the selected images is 512×512. The secret key of each image consists of initial values x_0, y_0, z_0, w_0, as well as the 128 bit MD5 hash value of the image. The encryption and decryption processes were simulated on professional graphic workstations (graphics card: NVDIA Tesla K80, processor: Intel Core i7-6700K, and the maximum RAM capacity was 8 GB) using Matlab. Matlab is a powerful software package which has been widely used for solving engineering problems and developing graphical illustrations [27, 28]. Simulation results and the original plain images are displayed in Figure 4. From that figure, we can find that the encrypted images are noise-like images, which are completely different from the original images. However, the decrypted images are identical to the original images.

4. Performance Analysis

A good encryption algorithm should guarantee security and exhibit an excellent encryption performance. An assessment of the developed image encryption algorithm is presented in this section. As mentioned in 3.7, Matlab was used for this computational analysis. The running platform was an Intel Core i7-6700K processor (8M Cache, 4 GHz) with a maximum RAM capacity of 8 GB and the operating system was Windows 7.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figure 4: Simulation results: (a, b, c, d) plain images; (e, f, g, h) encrypted images; (i, j, k, l) decrypted images.

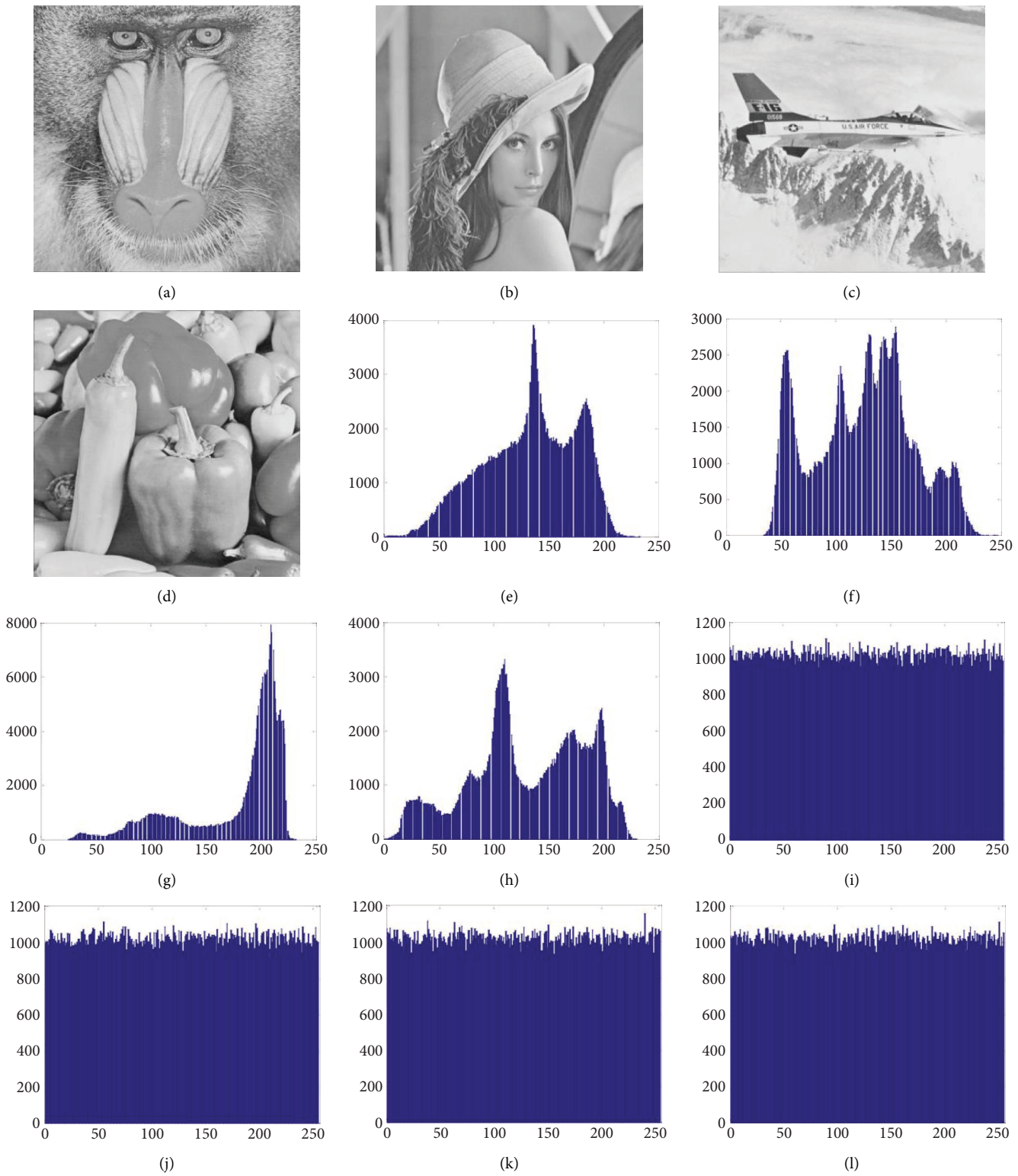


Figure 5: Histograms of original plain images (e)-(h) and encrypted images (i)-(l); (a)-(d) the original images.

4.1. Key Space. An encryption system needs to have a large key space to resist exhaustive attacks. The key space size has to be larger than 2^{128} to effectively protect the images from exhaustive attacks. The secret key generated from the proposed encryption algorithm includes four initial values x_0, y_0, z_0, w_0, and a 128 bit MD5 value. The initial values are double precision values. The space of the key consists of those first four values is about 2^{192}, which is far greater than 2^{128}. Therefore, the present encryption algorithm can effectively resist the exhaustive attacks.

Table 2: Correlation coefficients of selected adjacent pixels.

Algorithms	Images	Horizontal	Vertical	Diagonal
Present in this paper	Baboon (plain)	0.753386	0.864758	0.718634
Present in this paper	Lena (plain)	0.985316	0.977914	0.958496
Present in this paper	Airplane (plain)	0.970116	0.966374	0.940405
Present in this paper	Pepper (plain)	0.980926	0.975215	0.965311
Present in this paper	Baboon (encrypted)	0.011226	0.016652	0.004684
Present in this paper	Lena (encrypted)	0.016603	−0.007193	0.001036
Present in this paper	Airplane (encrypted)	0.014916	−0.000845	0.004081
Present in this paper	Pepper (encrypted)	0.010380	0.006469	−0.010354
[30]	Lena (encrypted)	−0.028872	0.014593	0.036587
[31]	Lena (encrypted)	−0.0285	0.0014	0.0013
[32]	Lena (encrypted)	0.0039	−0.0314	0.0158
[12]	Lena (encrypted)	0.011816	−0.017311	0.022785

4.2. Histograms. The histogram of an image reflects the image's statistical properties. Figure 4 displays histograms of the original plain images and the encrypted images. A secure encryption system can make an encrypted image with a uniform histogram to resist any statistical attacks [29]. As shown in Figure 5, the histograms of the encrypted images are evidently different from those of the plain images, which reveals that the pixel values of the encrypted images are evenly distributed. Therefore, the displayed histograms proved that the present scheme is well able to withstand the statistical attacks.

4.3. Correlation Coefficient Analysis. Neighboring pixels of plain images have strong correlations. An effective image encryption scheme should be able to reduce these correlations to almost zero. The correlation coefficient (r_{uv}, u and v are two adjacent pixels) is calculated as

$$r_{uv} = \frac{\operatorname{cov}(u, v)}{\sqrt{D(u)}\sqrt{D(v)}}, \tag{25}$$

where

$$\begin{aligned} \operatorname{cov}(u, v) &= \frac{1}{N}\sum_{i=1}^{N}\big(u_i - E(u)\big)\big(v_i - E(v)\big), \\ D(u) &= \frac{1}{N}\sum_{i=1}^{N}\big(u_i - E(u)\big)^2, \\ E(u) &= \frac{1}{N}\sum_{i=1}^{N}u_i. \end{aligned} \tag{26}$$

In this study, we selected 5,000 pairs of adjacent pixels from both the plain and encrypted images and calculated the correlation coefficients along vertical, horizontal, and diagonal directions. Calculation results are listed in Table 2 and Figure 6 compares the correlation coefficients calculated from the original and encrypted image of pepper. It can be seen from Table 2 that the horizontal, vertical, and diagonal correlation coefficients of the original plain images are close to 1 and the corresponding correlation coefficients of the encrypted images are nearly zero. This means that the present algorithm can significantly reduce the correlation coefficients and show desirable encryption performance.

4.4. Key Sensitivity Analysis. Next, we use two similar key codes (with slight differences) to encrypt the same image to obtain two encrypted images. If these two images differ markedly from each other, then it means the present algorithm is very sensitive to small changes in key because such small changes would produce great differences between cipher images. Otherwise, the key sensitivity of the algorithm is low. The initial values of the first key code are $x_0 = 0.4$, $y_0 = 0.9$, $z_0 = 30$, and $\omega_0 = 0.6$; and those of the second key code are set to be $x_0 = 0.4 + 10^{-15}$, $y_0 = 0.9$, $z_0 = 30$, and $\omega_0 = 0.6$. These two keys were used to encrypt the image for the airplane, and the two cipher images are displayed in Figure 7. From that figure, it can be found that the two encrypted images show obvious differences. The key sensitivity of the proposed encryption algorithm is therefore confirmed.

4.5. Information Entropy. Information entropy is a measure of the randomness or uncertainty of image information. Under ideal conditions, the information entropy of a grayscale image should be 8 if the image consists of randomly distributed pixels. Therefore, the information entropy of a cipher image should be close to eight if the cipher image is encrypted using an effective image encryption algorithm. The information entropies of the four plain images and the corresponding encrypted images were calculated, and Table 3 lists the calculation results.

From Table 3 it can be found that the information entropies of the encrypted images are close to the optimal value 8, which verifies that the presented scheme is secure enough to be used for image transmission. Furthermore, the information entropy of the image of Lena processed using other encryption schemes was calculated as 7.9972 [33], 7.9992 [34], 7.9993 [31], and 7.9991 [35]. Compared to those values, the information entropy of the cipher image encrypted utilizing the present algorithm is closer to 8.

4.6. Resistance to Differential Attacks. An effective image encryption scheme should have excellent plain-image sensitivity and be capable of defending against differential attacks. It signifies that any slight change in the original plain image will cause its cipher image to change

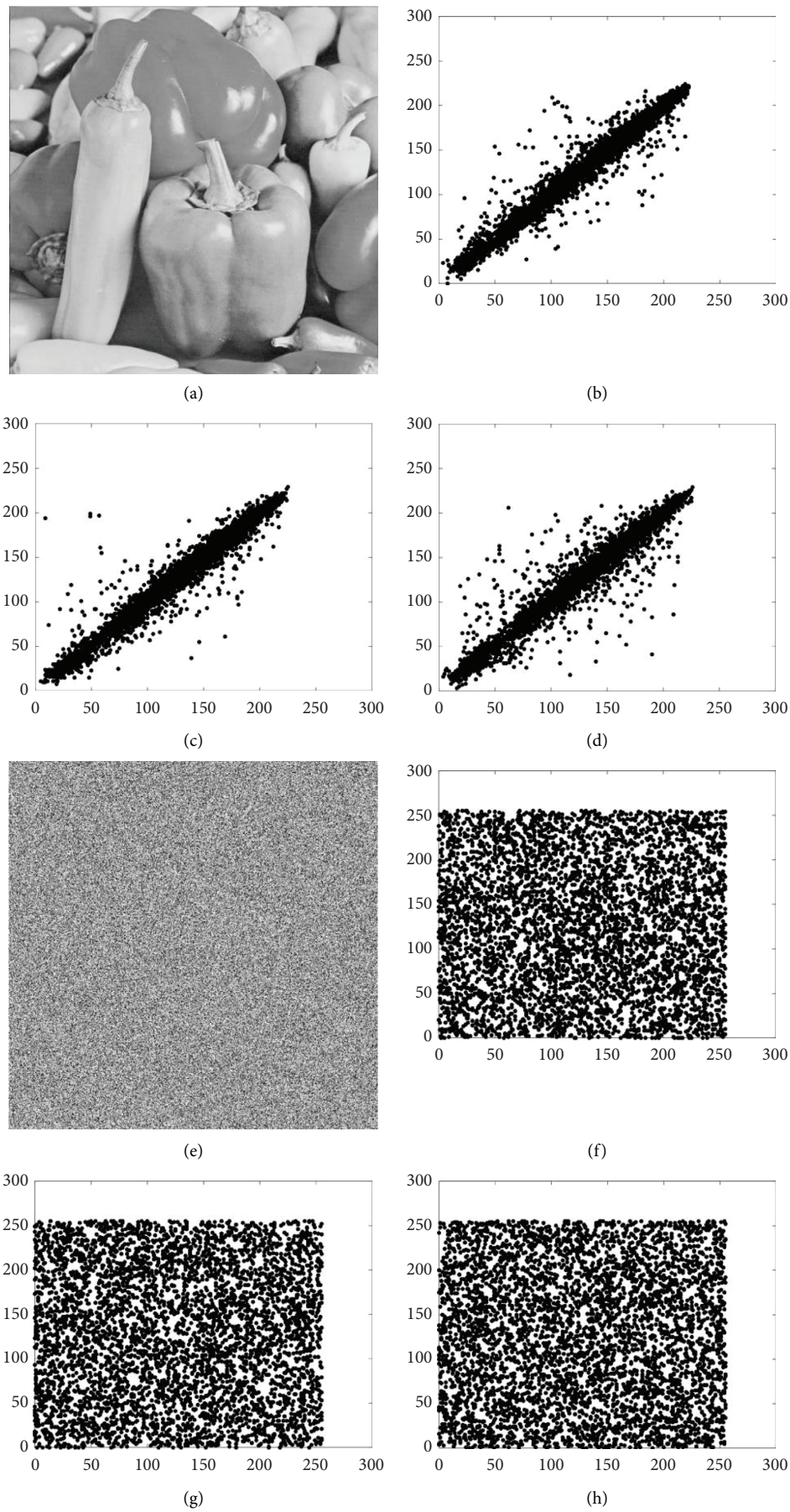
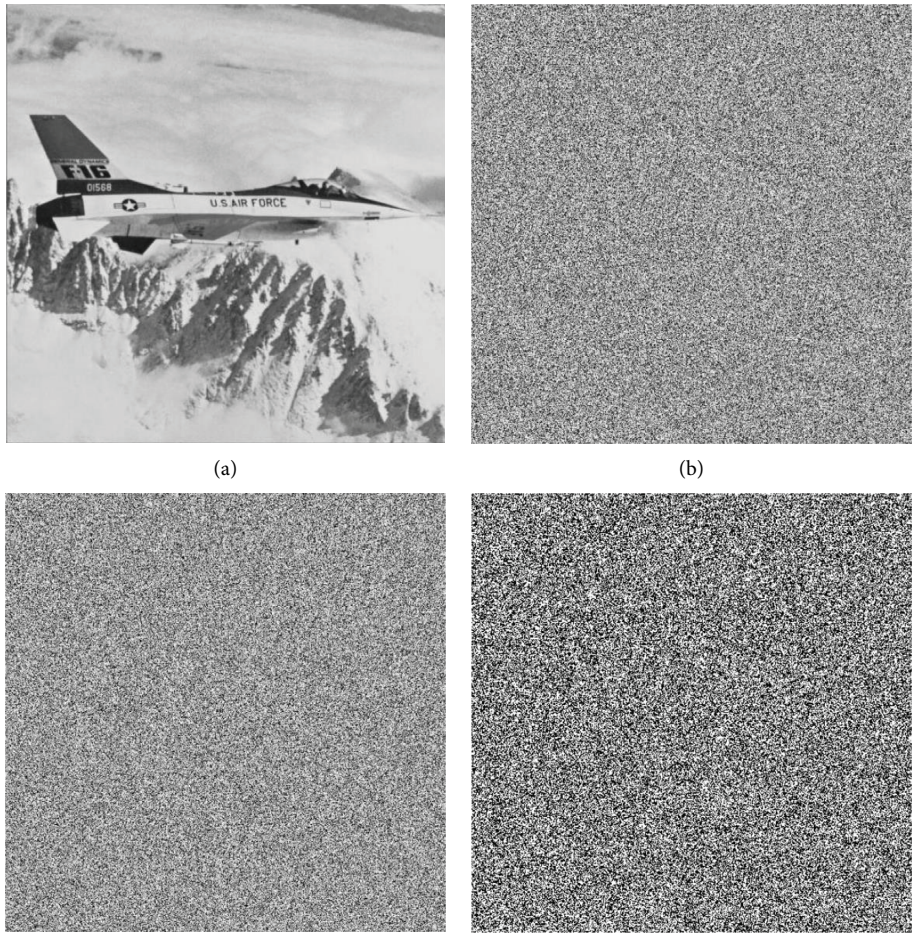


Figure 6: Correlation coefficients of the image of pepper (a) plain image; (b)-(d) horizontal, vertical, and diagonal correlation coefficients of the plain image; (e) encrypted image; (f)-(h) horizontal, vertical, and diagonal correlation coefficients of the encrypted image.



(a) (b)

(c) (d)

Figure 7: Key sensitivity analysis results: (a) a plain image for airplane; (b) a cipher image encrypted using the first key code; (c) a cipher image encrypted using the second key code; and (d) the difference between (b) and (c).

Table 3: Information entropies of the plain and encrypted figures.

Image	Information entropy for plain image	Information entropy for encrypted image
Baboon	7.315499	7.999338
Lena	7.347902	7.999303
Airplane	6.716945	7.999211
Pepper	7.517146	7.999275

Table 4: NPCR and UACI of cipher images.

Encryption algorithm	Image	NPCR (%)	UACI (%)
Present in this study	Baboon	99.593734	33.544325
Present in this study	Lena	99.590302	33.462442
Present in this study	Airplane	99.600983	33.481864
Present in this study	Pepper	99.602127	33.453822
[31]	Lena	99.62	33.46
[37]	Lena	99.63	33.56
[38]	Lena	99.6227	33.51964

significantly. The plain image sensitivity of the present image encryption algorithm is determined through the following steps: Assuming a plain image P_1, we first change the value of a randomly selected pixel of that image to obtain another plain image P_2. Next, both P_1 and P_2 are encrypted, applying the present encryption algorithm to acquire two cipher images, C_1 and C_2. Finally, the plain-image sensitivity of the proposed algorithm is determined by comparing C_1 with C_2. The difference between C_1 and C_2 is indicated using net pixel change rate

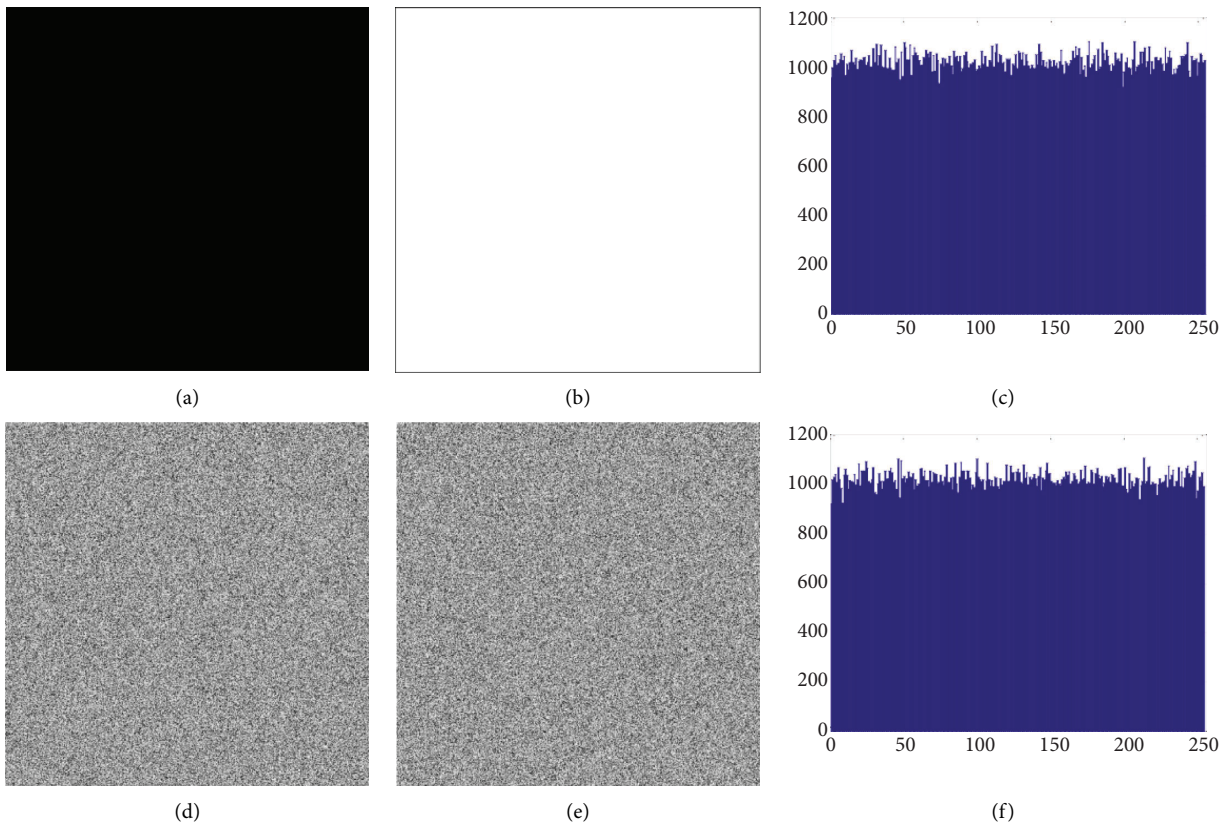


Figure 8: Plain and cipher histograms and images: (a) plain black image; (b) plain white image; (c) histogram of the cipher image of the black image; (d) cipher image of the black image; (e) cipher image of the white image; and (f) histogram of the cipher image of the white image.

Table 5: Information entropies and correlation coefficients of the cipher images.

Image		Information entropy	Correlation coefficient		
			Horizontal	Vertical	Diagonal
White	Plain image	0	0	0	0
	Cipher image	7.999292	−0.0272079	0.028597	−0.0085912
Black	Plain image	0	0	0	0
	Cipher image	7.999257	0.023715	−0.007927	0.042870

(NPCR) (27) and unified average changing intensity (UACI) (28) [36].

$$\text{NPCR} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} |D(i,j)| \times 1, \tag{27}$$

$$\text{UACI} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{\left|C_1(i,j) - C_2(i,j)\right|}{255}, \tag{28}$$

where

$$D(i,j) = \begin{cases} 1, & C_1(i,j) \neq C_2(i,j), \\ 0, & C_1(i), j = C_2(i,j). \end{cases} \tag{29}$$

In equation (29), (C1) (i, j) and $C_2(i, j)$ denote pixel values of C_1 and C_2. The NPCR and UACI of the encrypted images of baboon, Lena, airplane, and pepper are displayed in Table 4. For comparison, that table also contains the NPCR and UACI of the cipher images of Lena encrypted using existing image encryption algorithms.

Table 6: Speed analysis results.

Encryption algorithm	Image (size)	Time (s)
Present in this study	Lena (256 × 256)	0.547
[39]	Lena (256 × 256)	1.532
[40]	Lena (256 × 256)	0.580
[12]	Lena (256 × 256)	2.443

From Table 4, it is found that compared with other schemes, the UACI and NPCR values yielded from the proposed algorithm are closer to their theoretical values of 99.61% and 33.46%, respectively. In other words, if we use the same key code generated from our algorithm to encrypt

Table 7: Encryption times and information entropies of the encrypted images.

Encryption algorithms	Images (size)	Times (s)	Information entropies
Proposed one with two scrambling-diffusion operations	Baboon (256×256)	0.559	7.999
An existing algorithm with one scrambling-diffusion operation	Baboon (256×256)	0.484	7.997

two very similar plain images, the obtained cipher images will differ markedly. Hence, it is verified that the developed algorithm is very sensitive to plain images and possesses a strong capacity to resist the differential attacks.

4.7. Resistance to Chosen Plain Image Attacks. Our algorithm can effectively defend against chosen plain image attacks. This is because the generated key code is related to the plain image, so different key codes will be acquired for different images. With a chosen plain image attack, the attacker can encrypt a plain image of his or her choice and has access to the resulting cipher image. This information is used to derive the encryption key and break the encryption algorithm. To assess the resistance of the designed algorithm to the chosen plain image attacks, the algorithm was applied to encrypt a 512×512 black image and a white image of the same size. The obtained cipher images were analyzed, and the results are displayed in Figure 8 and Table 5.

As shown in Table 5, the information entropies of the cipher images are very close to 8 and the correlation coefficients are close to 0. Therefore, almost no useful information can be extracted from the cipher images encrypted using the developed algorithm.

4.8. Speed Analysis. A good image encryption scheme should possess not only high security but also fast encryption speed. Using the image of Lena as an example, the speed analysis data (Table 6) show that compared with other image encryption algorithms, our algorithm is faster in encrypting images.

4.9. Discussion. The proposed image encryption algorithm consists of two scrambling-diffusion operations. To verify the superiority of two scrambling-diffusion operations over one scrambling-diffusion operation in image encryption, we encrypted the image "Baboon" using the proposed algorithm and an existing image encryption algorithm that only includes one scrambling-diffusion operation separately, and compared the encryption times and information entropies of these two encryptions (Table 7). As displayed in Table 7, the times needed for two scrambling-diffusion operations are quite close to those needed for one operation. However, the information entropy of the image encrypted after two scrambling-diffusion operations is better than the information entropy of the image encrypted with one operation. The comparison shows that the proposed encryption algorithm has high efficiency and exhibits greater performance than the algorithm with only one scrambling-diffusion operation.

5. Conclusion

This paper presents an image encryption algorithm based on the hyperchaotic Chen system. In this algorithm, the key code and the diffusion process are directly related to plain images. The present algorithm was employed to encrypt several plain images. Important parameters such as the key space, histograms, correlation coefficients, key and plain-image sensitivity, and information entropy of the cipher images were calculated to validate the encryption performance of the designed algorithm. Simulation results indicate that the proposed algorithm has high security and can effectively defend against the chosen plain image attacks and differential attacks. Therefore, the algorithm possesses high potential for practical applications. The present encryption algorithm will be further improved to enhance its security and encryption speed through the introduction of an image compression algorithm and machine learning techniques. [41, 42].

Data Availability

The data that support the findings of this study are available upon request.

Conflicts of Interest

All authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the High-Level Talent Research Start-Up Fund of Yulin Normal University under grant no. G2019ZK24. An earlier version of this paper has been presented as a preprint available at SSRN [42].

References

[1] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and Chaos*, vol. 08, no. 06, pp. 1259–1284, 1998.

[2] G. Chen, Y. Mao, and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps," *Chaos, Solitons & Fractals*, vol. 21, no. 3, pp. 749–761, 2004.

[3] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and Vision Computing*, vol. 24, no. 9, pp. 926–934, 2006.

[4] I. Hussain and M. A. Gondal, "An extended image encryption using chaotic coupled map and S-box transformation," *Nonlinear Dynamics*, vol. 76, no. 2, pp. 1355–1363, 2014.

[5] S. M. Seyedzadeh, B. Norouzi, M. R. Mosavi, and S. Mirzakuchaki, "A novel color image encryption algorithm based on spatial permutation and quantum chaotic map," *Nonlinear Dynamics*, vol. 81, no. 1-2, pp. 511–529, 2015.

[6] X. Wu, D. Wang, J. Kurths, and H. Kan, "A novel lossless color image encryption scheme using 2D DWT and 6D hyperchaotic system," *Information Sciences*, vol. 349-350, pp. 137–153, 2016.

[7] C. Pak and L. Huang, "A new color image encryption using combination of the 1D chaotic map," *Signal Processing*, vol. 138, pp. 129–137, 2017.

[8] G. Ye, C. Pan, X. Huang, and Q. Mei, "An efficient pixel-level chaotic image encryption algorithm," *Nonlinear Dynamics*, vol. 94, no. 1, pp. 745–756, 2018.

[9] W. K. Lee, R. C. W. Phan, W. S. Yap, and B. M. Goi, "SPRING: a novel parallel chaos-based image encryption scheme," *Nonlinear Dynamics*, vol. 92, no. 2, pp. 575–593, 2018.

[10] M. Guan, X. Yang, and W. Hu, "Chaotic image encryption algorithm using frequency-domain DNA encoding," *IET Image Processing*, vol. 13, no. 9, pp. 1535–1539, 2019.

[11] H. Zhu, Y. Zhao, and Y. Song, "2D logistic-modulated-sine-coupling-logistic chaotic map for image encryption," *IEEE Access*, vol. 7, Article ID 14081, 2019.

[12] M. A. B. Farah, A. Farah, and T. Farah, "An image encryption scheme based on a new hybrid chaotic map and optimized substitution box," *Nonlinear Dynamics*, vol. 99, no. 4, pp. 3041–3064, 2020.

[13] A. Girdhar, H. Kapur, and V. Kumar, "A novel grayscale image encryption approach based on chaotic maps and image blocks," *Applied Physics B*, vol. 127, no. 3, pp. 39–12, 2021.

[14] Q. Lai, C. Lai, H. Zhang, and C. Li, "Hidden coexisting hyperchaos of new memristive neuron model and its application in image encryption," *Chaos, Solitons & Fractals*, vol. 158, Article ID 112017, 2022.

[15] Q. Lai, Z.-Q. Wan, H. Zhang, and G.-R. Chen, "Design and analysis of multiscroll memristive Hopfield neural network with adjustable memductance and application to image encryption," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[16] Z. Hua and Y. Zhou, "Image encryption using 2D logistic-adjusted-sine map," *Information Sciences*, vol. 339, pp. 237–253, 2016.

[17] P. Zhen, G. Zhao, L. Min, and X. Jin, "Chaos-based image encryption scheme combining DNA coding and entropy," *Multimedia Tools and Applications*, vol. 75, no. 11, pp. 6303–6319, 2016.

[18] A. Jain and N. Rajpal, "A robust image encryption algorithm resistant to attacks using DNA and chaotic logistic maps," *Multimedia Tools and Applications*, vol. 75, no. 10, pp. 5455–5472, 2016.

[19] W. Feng, Y. He, H. Li, and C. Li, "Cryptanalysis and improvement of the image encryption scheme based on 2D logistic-adjusted-sine map," *IEEE Access*, vol. 7, Article ID 12584, 2019.

[20] X. Su, W. Li, and H. Hu, "Cryptanalysis of a chaos-based image encryption scheme combining DNA coding and entropy," *Multimedia Tools and Applications*, vol. 76, no. 12, Article ID 14021, 2017.

[21] Y. Dou, X. Liu, H. Fan, and M. Li, "Cryptanalysis of a DNA and chaos based image encryption algorithm," *Optik*, vol. 145, pp. 456–464, 2017.

[22] A. Kadir, A. Hamdulla, and W. Q. Guo, "Color image encryption using skew tent map and hyper chaotic system of 6^{th}-order CNN," *Optik*, vol. 125, no. 5, pp. 1671–1675, 2014.

[23] H. Zhu, C. Zhao, X. Zhang, and L. Yang, "An image encryption scheme using generalized Arnold map and affine cipher," *Optik*, vol. 125, no. 22, pp. 6672–6677, 2014.

[24] Y. Li, W. K. S. Tang, and G. Chen, "Generating hyperchaos via state feedback control," *International Journal of Bifurcation and Chaos*, vol. 15, no. 10, pp. 3367–3375, 2005.

[25] T. Gao, Z. Chen, Z. Yuan, and G. Chen, "A hyperchaos generated from Chen's system," *International Journal of Modern Physics C*, vol. 17, no. 04, pp. 471–478, 2006.

[26] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, p. 800, NIST Special Publication, Gaithersburg, MD, USA, 2002.

[27] Y C. Liu, "A programming course including C# and MATLAB for mechanical engineering students," *ASEE Computers in Education Journal*, vol. 2, no. 3, pp. 106–112, 2011.

[28] Y C. Liu, "Implementation of MATLAB/Simulink into a Vibration and Control Course for Mechanical Engineering Students," in *Proceedings of the ASEE SE Section Annual Conference*, Auburn University, Alabama, AL, USA, March, 2020.

[29] Y. Li, C. Wang, and H. Chen, "A hyper-chaos-based image encryption algorithm using pixel-level permutation and bit-level permutation," *Optics and Lasers in Engineering*, vol. 90, pp. 238–246, 2017.

[30] E. Yavuz, R. Yazici, M. C. Kasapbaşı, and E. Yamac, "A chaos-based image encryption algorithm with simple logical functions," *Computers & Electrical Engineering*, vol. 54, pp. 471–483, 2016.

[31] X. Chai, "An image encryption algorithm based on bit level Brownian motion and new chaotic systems," *Multimedia Tools and Applications*, vol. 76, no. 1, pp. 1159–1175, 2017.

[32] X. Zhang, Z. Zhou, and Y. Niu, "An image encryption method based on the Feistel network and dynamic DNA encoding," *IEEE Photonics Journal*, vol. 10, no. 4, pp. 1–14, 2018.

[33] A. Kulsoom, D. Xiao, A. Rehman, and S. A. Abbas, "An efficient and noise resistive selective image encryption scheme for gray images based on chaotic maps and DNA complementary rules," *Multimedia Tools and Applications*, vol. 75, pp. 1–23, 2016.

[34] X. Wang and D. Xu, "A novel image encryption scheme based on Brownian motion and PWLCM chaotic system," *Nonlinear Dynamics*, vol. 75, no. 1-2, pp. 345–353, 2014.

[35] Y. Mao, G. Chen, and S. Lian, "A novel fast image encryption scheme based on 3D chaotic baker maps," *International Journal of Bifurcation and Chaos*, vol. 14, no. 10, pp. 3613–3624, 2004.

[36] M. A. Murillo-Escobar, C. Cruz-Hernandez, F. Abundiz-Perez, R. M. Lopez-Gutierrez, and O. R. Acosta Del Campo, "A RGB image encryption algorithm based on total plain image characteristics and chaos," *Signal Processing*, vol. 109, pp. 119–131, 2015.

[37] X. Wang and D. Xu, "A novel image encryption scheme using chaos and Langton's Ant cellular automaton," *Nonlinear Dynamics*, vol. 79, no. 4, pp. 2449–2456, 2015.

[38] S. F. Raza and V. Satpute, "A novel bit permutation-based image encryption algorithm," *Nonlinear Dynamics*, vol. 95, no. 2, pp. 859–873, 2019.

[39] H. Zhu, X. Zhang, H. Yu, C. Zhao, and Z. Zhu, "An image encryption algorithm based on compound homogeneous hyper-chaotic system," *Nonlinear Dynamics*, vol. 89, no. 1, pp. 61–79, 2017.

[40] X. Chai, X. Zheng, Z. Gan, D. Han, and Y. Chen, "An image encryption algorithm based on chaotic system and compressive sensing," *Signal Processing*, vol. 148, pp. 124–144, 2018.

[41] G. Chen and T. Ueta, "Yet another chaotic attractor," *International Journal of Bifurcation and Chaos*, vol. 09, no. 07, pp. 1465-1466, 1999.

[42] Y.-Z. Huang, X.-G. Bi, Y.-X. Li, Y.-C. Liu, and [, "A novel hyperchaos-based image encryption algorithm with two scrambling-diffusion operations," 2022, https://ssrn.com/abstract=3994132.