Hindawi
Security and Communication Networks
Volume 2019, Article ID 8694678, 15 pages
https://doi.org/10.1155/2019/8694678

Research Article

Image Encryption with Double Spiral Scans and Chaotic Maps

Zhenjun Tang,[1] **Ye Yang**,[1] **Shijie Xu**,[1] **Chunqiang Yu**,[2] **and Xianquan Zhang**[1,2]

[1]*Guangxi Key Lab of Multi-Source Information Mining & Security, and Department of Computer Science, Guangxi Normal University, Guilin 541004, China*
[2]*Network Information Center, Guangxi Normal University, Guilin, 541004, China*

Correspondence should be addressed to Zhenjun Tang; tangzj230@163.com

Received 19 August 2018; Revised 9 December 2018; Accepted 24 December 2018; Published 15 January 2019

Academic Editor: Angel M. Del Rey

Image encryption is a useful technique of image content protection. In this paper, we propose a novel image encryption algorithm by jointly exploiting random overlapping block partition, double spiral scans, Henon chaotic map, and Lü chaotic map. Specifically, the input image is first divided into overlapping blocks and pixels of every block are scrambled via double spiral scans. During spiral scans, the start-point is randomly selected under the control of Henon chaotic map. Next, image content based secret keys are generated and used to control the Lü chaotic map for calculating a secret matrix with the same size of input image. Finally, the encrypted image is obtained by calculating XOR operation between the corresponding elements of the scrambled image and the secret matrix. Experimental result shows that the proposed algorithm has good encrypted results and outperforms some popular encryption algorithms.

1. Introduction

Some well-known security events, such as PRISM and Xkeyscore, make people pay much attention to information security. Since digital images are widely used in the Internet, how to protect image content [1, 2] has become an issue to be urgently solved. Image encryption is a useful technique of image content protection [3]. It converts images into noise-like encrypted images by disrupting pixel positions or changing pixel values. In recent years, researchers have developed many useful image encryption algorithms. These algorithms can be roughly divided into two categories as follows.

The first direction is to manipulate image pixels in the spatial domain. This kind of algorithm usually scrambles pixel positions through matrix transformation and destroys spatial correlation between pixels of the original image, so as to convert input image into chaotic image. For example, Tang et al. [4] proposed an encryption algorithm based on Arnold transform and three random strategies. This encryption technique is a secure algorithm and can overcome size limitation of the Arnold transform. In another study, Tang et al. [5] divided input image into overlapping blocks, conducted random block shuffling, and exploited Arnold transform and a chaotic map to generate secure matrix for block-wise encryption. In [6], Zhang and Liu used skew-tent chaotic map to achieve permutation and diffusion without changing pixel information. This method has high efficiency and a large key space, but it is not secure enough from the viewpoint of histogram [5]. In [7], Li et al. exploited reversible data hiding (RDH) and compressive sensing to design a meaningful image encryption algorithm. This algorithm encrypts a secret image into a meaningful image by RDH and reaches a high embedding rate. Recently, Wang et al. [8] designed an efficient image encryption algorithm based on two-dimensional partitioned cellular automaton. This algorithm supports parallel computing and is easy for VLSI implementation. In another work, Wang et al. [9] exploited multiple mixed hash functions, cyclic-shift function, and piece-wise linear chaotic maps to achieve image encryption. This scheme can overcome security flaw of the well-known chaotic image encryption called Baptista's algorithm and its improved versions. In [10], Hayat and Azam proposed a useful image encryption technique using a dynamic S-box and pseudo-random numbers over an elliptic curve. This technique can resist known plaintext attack and chosen plaintext attack.

Since chaotic systems have many sensitive properties (e.g., sensitivity to initial conditions and system parameters) and show better performance than traditional encryption techniques (e.g., AES [11] and DES [12]), many researchers have tried to design image encryption with chaotic maps [13–19]. In general, chaos-based image encryption algorithms consist of two steps: pixel permutation and pixel diffusion. The pixel permutation changes pixel position, while the pixel diffusion alters pixel values where a change in a pixel will spread almost to other pixels of entire image. Contributed by the sensitivity properties of chaotic system, chaos-based image encryption algorithms generally achieve good security performance. Some representative chaos-based encryption algorithms are introduced here. Amin et al. [20] proposed a new image encryption algorithm based on chaotic block cipher. This algorithm jointly uses cryptographic primitive operations, nonlinear transformation functions, and chaotic tent map to achieve encryption. It is secure against brute-force attack. Abd El-Latif et al. [21] presented a novel image encryption with linear feedback shift register and chaotic maps in time and frequency domains. This method can resist differential attack. In another study, Abd El-Latif and Niu [22] proposed a hybrid image encryption by using chaotic system and cyclic elliptic curve. This method reaches good security. In [23], Tang et al. exploited Henon map, logistic map, and bit-plane decomposition to design an algorithm for multiple-image encryption. This algorithm can convert four gray-scale images into an encrypted PNG image. In another work, Wang et al. [24] used two chaotic systems to develop a hybrid color image encryption scheme. In [25], Abanda and Tiedeu proposed a fast and simple image encryption algorithm by combining two kinds of chaotic maps to meet real-time application. Belazi et al. [26] presented a novel selective image encryption by using DWT with AES s-box and chaotic permutation. This method can resist differential and statistical attacks. In another study, Belazi et al. [27] designed an efficient image encryption with substitution-permutation network and chaotic systems. This algorithm has good performances in security and speed.

Recently, Tang et al. [28] proposed an image encryption algorithm by using random projection partition and chaotic system. This algorithm is secure and has a fast speed. Li et al. [29] introduced a quantum color image encryption scheme. This scheme exploits quantum controlled-NOT image generated by multiple chaotic maps to control the XOR operation in the encryption process. It can resist the attack of histogram analysis. Parvaz and Zarebnia [30] defined a combination chaotic system with logistic, sine, and tent systems and applied it to image encryption. To improve security, Liu et al. [31] proposed to use a randomly sampled noise signal as initial value of chaotic map. Chen and Hu [32] designed an adaptive encryption algorithm based on improved chaotic mapping for medical images. However, the encryption results of this algorithm have obvious block effect. In [33], Chai et al. jointly used chaotic system, elementary cellular automata, and compressive sensing to design efficient image encryption. This algorithm can resist known-plaintext attack and chosen-plaintext attack. Wu et al. [34] proposed a new image encryption algorithm by pixel diffusion with DNA approach and pixel permutation by a two-dimensional Hénon-Sine map. This algorithm can resist statistical attack, differential attack, and noise attack, but has limitation in encrypting color images.

The other direction is to conduct encryption in the transform domain [35, 36]. Generally, this kind of encryption algorithms firstly transforms input image from spatial domain to transform domain, then modifies those coefficients in the transform domain with some well-defined rules, and finally converts coefficients to spatial domain. For example, Singh et al. [35] exploited Arnold transform and singular value decomposition to make phase image encryption in the fractional Hartley domain. Vashisth et al. [36] conducted image encryption in the fractional Mellin transform domain by using structured phase filters and phase retrieval. Naeem et al. [37] presented novel image encryption algorithms with a cyclic shift and the 2D chaotic Baker map in transform domains, such as the Integer Wavelet Transform (IWT) domain, the Discrete Wavelet Transform (DWT) domain, and the Discrete Cosine Transform (DCT) domain. The algorithm in DWT domain shows better performance than those in other transform domains. In another work, Belazia et al. [38] proposed a novel partial image encryption approach based on permutation-substitution-diffusion (PSD) network and multiple chaotic maps in wavelet transform domain. Recently, Annaby et al. [39] integrated random fractional Fourier transforms, phase retrieval, and chaotic maps to design a scheme for color image encryption. Zhang and Tong [40] exploited IWT and set partitioning in hierarchical trees (SPIHT) to make image encryption and compression. This method has good performances in security and compression. Li et al. [41] exploited two-dimensional DWT to decompose original images and conducted encryption in DWT domain by Arnold transform and robust chaotic map. This algorithm can convert four grayscale images with the same size into an encrypted image. Wu et al. [42] introduced an asymmetric multiple-image encryption method via compressed sensing and nonlinear operations in cylindrical diffraction domain. This method can encrypt eight images and resist ciphertext-only attack.

In this paper, we propose an image encryption algorithm based on double spiral scans and chaotic maps. The proposed algorithm jointly exploits random overlapping block partition, double spiral scans, Henon chaotic map, and Lü chaotic map to calculate encrypted image. Compared with previous image encryption algorithms based on chaotic map, a key contribution of our algorithm is the double spiral scans, which can efficiently scramble pixels of image block. Many experiments are conducted and the results show that the proposed algorithm is effective and outperforms some popular encryption algorithms. The rest of this paper is organized as follows. In Section 2, we introduce the proposed algorithm. In Section 3, we present key space analysis. In Section 4, we discuss experimental results. Finally, conclusions of this paper are made in Section 5.

2. Proposed Algorithm

Figure 1 shows the block diagram of our image encryption. The main steps of our algorithm include random overlapping

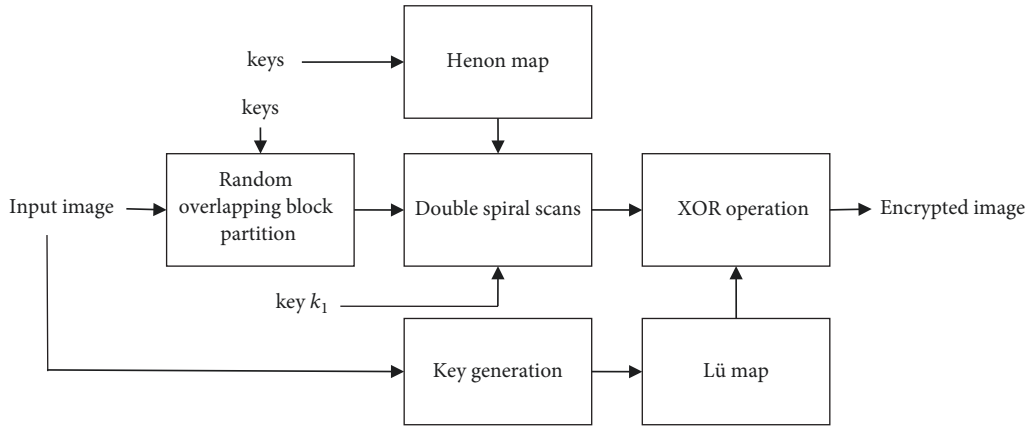


Figure 1: Block diagram of our image encryption.

block partition, double spiral scans, and XOR operation. In the first step, the input image is randomly divided into overlapping blocks under the control of secret keys. In the second step, we shuffle image pixels of every overlapping block by double spiral scans, where a secret key k_1 is used to determine the order of encrypting image blocks. To improve security, the Henon chaotic map is exploited to generate random start-points for double spiral scans. In the third step, we generate keys based on the content of input image and use the keys to control the Lü chaotic map for generating a secret matrix. Finally, we calculate the XOR operation between the shuffled image and the secret matrix, and the result is the final encrypted image. In the following sections, we first introduce the random overlapping-block partition, then describe the double spiral scans and the used chaotic maps in our algorithm, and finally explain the key generation and the detailed steps of our encryption scheme and decryption scheme.

2.1. Random Overlapping Block Partition. We exploit random overlapping-block partition scheme [5] to divide input image into overlapping blocks. And then we shuffle every image block by double spiral scans to complete the scrambling operation. The detailed process of the random overlapping-block partition is explained as follows. Assume that the size of input image is $M \times N$, the selected block size is $S \times S$, and t_x and t_y are the overlapping sizes between adjacent blocks along the x-axis and the y-axis, respectively, where $t_x \in [1, S)$ and $t_y \in [1, S)$. Let n_x and n_y be the numbers of image blocks in the x-axis and the y-axis, respectively. Thus, n_x and n_y can be calculated as follows.

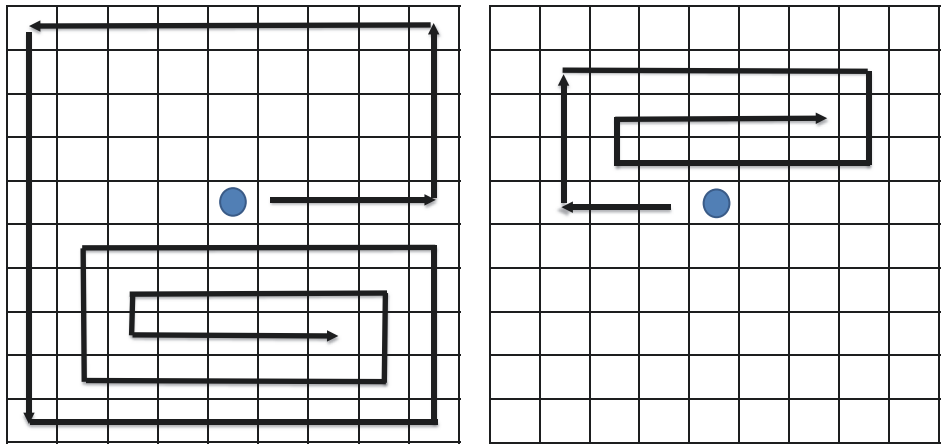
$$n_x = \begin{cases} \dfrac{N - t_x}{S - t_x}, & \text{If } \mod(N - t_x, S - t_x) = 0 \\ \left\lfloor \dfrac{N - t_x}{S - t_x} \right\rfloor + 1, & \text{Otherwise} \end{cases} \quad (1)$$

$$n_y = \begin{cases} \dfrac{M - t_y}{S - t_y}, & \text{If } \mod(M - t_y, S - t_y) = 0 \\ \left\lfloor \dfrac{M - t_y}{S - t_y} \right\rfloor + 1, & \text{Otherwise} \end{cases} \quad (2)$$

where $\lfloor \cdot \rfloor$ means rounding down operation and $\mod(\cdot, \cdot)$ means modulo operation. Therefore, the total number of random overlapping blocks is $N_{\text{total}} = n_x \times n_y$. The coordinate of the i-th image block in the x-axis direction is denoted by $X[i]$, and the coordinate of the j-th image block in the y-axis direction is denoted by $Y[i]$, where $i = 1, 2, \ldots, n_x$ and $j = 1, 2, \ldots, n_y$. Then, $X[i]$ and $Y[j]$ can be determined as follows. If $\mod(N - t_x, S - t_x) = 0$, $X[i] = (i - 1)(S - t_x)$, where $i = 1, 2, \ldots, n_x$. Otherwise, the x-coordinates of the first $n_x - 1$ blocks are calculated by $X[i] = (i - 1)(S - t_x)$, where $i = 1, 2, \ldots, n_x - 1$, and the x-coordinate of the last block is $X[n_x] = N - S + 1$. Similarly, if $\mod(M - t_y, S - t_y) = 0$, $Y[j] = (j - 1)(S - t_y)$, where $y = 1, 2, \ldots, n_y$. Otherwise, the y-coordinates of the first $n_y - 1$ blocks are $Y[j] = (j - 1)(S - t_y)$, where $y = 1, 2, \ldots, n_y - 1$. And the y-coordinate of the last block is $Y[n_y] = M - S + 1$. Here, image blocks are numbered from left to right and top to bottom, and the coordinates of the i-th image block are represented by $(X[u_i], Y[v_j])$. The random block pattern depends on the block size S and the overlapping sizes t_x and t_y, where the theoretical range of S is $(1, \min(M, N)]$. In the experiment, it is found that a small S value will make more image blocks and thus lead to a slow speed. Therefore, we randomly select the S value from the range $[32, \min(M, N))$ in this study. As the ranges of t_x and t_y are both $[1, S)$, the total number of our random block patterns is $(\min(M, N) - 31)(S - 1)^2$. Note that the parameters S, t_x, and t_y are determined by the user and thus they can be taken as secret keys in practice.

2.2. Double Spiral Scans. The scheme of double spiral scans proposed in this paper is used to scramble position of every pixel in an overlapping block. Details of our double spiral scans for pixel scrambling in a block are explained as follows.

As shown in Figure 2, our double spiral scans consist of two parts. Firstly, a start-point is randomly selected. Then, we visit block pixels from the start-point with a spiral scan as shown in Figure 2(a) and obtain a pixel sequence according to the order of visit. Similarly, we visit block pixels from the start-point with another spiral scan as shown in Figure 2(b) and obtain another pixel sequence according to the order of the visit. Next, we concatenate the first pixel sequence and the



(a) The first scanning direction (b) The second scanning direction

Figure 2: Diagram of double spiral scans.

A	B	C	D	E
F	G	H	I	J
K	L	M	N	O
P	Q	R	S	T
U	V	W	X	Y

(a) Original block

N	O	J	E	D
C	B	A	F	K
P	U	V	W	X
Y	T	S	R	Q
M	L	G	H	I

(b) Encrypted block started from 'N'

Q	R	S	T	O
J	E	D	C	B
A	F	K	P	U
V	W	X	Y	L
G	H	I	N	M

(c) Encrypted block started from 'Q'

Figure 3: Original block and different encrypted blocks.

Pixel sequence 1: A-B-C-D-E-F-G-H-I-J-K-L-M-N-O-P-Q-R-S-T-U-V-W-X-Y.
Pixel sequence 2: N-O-J-E-D-C-B-A-F-K-P-U-V-W-X-Y-T-S-R-Q-M-L-G-H-I.
Pixel sequence 3: Q-R-S-T-O-J-E-D-C-B-A-F-K-P-U-V-W-X-Y-L-G-H-I-N-M.

Figure 4: Original pixel sequence and the pixel sequences generated by double spiral scans with different start-points.

second pixel sequence to make a new pixel sequence. Finally, we can generate the encrypted image block by filling image block from left to right and top to bottom via picking pixel from the new sequence one by one. Note that all block pixels are visited, and every pixel is scanned only once.

More specifically, we first randomly select start-point for the double spiral scans. Here, coordinates of the random start-point are randomly generated by the Henon chaotic map. The detailed calculation will be described in Section 2.3. As shown in Figure 2, the solid circle is the start-point of double spiral scans in a block. The scanning process can be divided into two directions. For the first scanning direction, as shown in Figure 2(a), we scan block pixels starting from the start-point and follow the below scanning direction: right, up, left, and down. When scanning to block border or the scanned pixel, we turn the scanning direction. If there is no pixel for scanning, the first scanning process is finished and then the first pixel sequence is obtained. For the second scanning direction, as shown in Figure 2(b), we scan block pixels starting from the start-point and follow the below scanning direction: left, up, right, and down. Similarly, when scanning to block border or the scanned pixel, we turn the scanning direction. If there is no pixel for scanning, the second scanning process is also finished and then the second pixel sequence is obtained. Next, a new pixel sequence can be generated by concatenating the first and the second pixel sequences. Finally, the encrypted block can be obtained by filling it with the new pixel sequence from left to right and top to bottom. Clearly, the encrypted block can be accurately decrypted once the start-point is known by filling pixels back according to the visiting order of double spiral scans.

An example of our double spiral scans is illustrated here. Figure 3(a) is a 5×5 image block. We scan pixels from left to right and top to bottom and then get a pixel sequence 1, as shown in Figure 4. Suppose that coordinates of the start-point are (3, 4), i.e., the location of 'N' as shown in Figure 3(a). We visit block pixels by the double spiral scans and generate the pixel sequence 2, as shown in Figure 4. Then, we fill pixels back to image block with the pixel sequence 2 and obtain the encrypted block as shown in Figure 3(b). Similarly, suppose that coordinates of the start-point are (5, 2), i.e., the location of 'Q' as shown in Figure 3(a). We visit block pixels by the double spiral scans and generate the pixel sequence 3, as shown in Figure 4. Then, we fill pixels back

to image block with the pixel sequence 3 and obtain the encrypted block as shown in Figure 3(c). Obviously, different start-points lead to different encrypted results. In this study, we choose different start-points for different image blocks by using Henon chaotic map. This strategy can improve security of our algorithm.

2.3. Chaotic Maps. This section describes the chaotic maps used in our algorithm. Henon chaotic map is a typical two-dimensional discrete chaotic map. We use it to generate the start-points for double spiral scans. The Henon chaotic map is defined as follows.

$$\begin{aligned} x(k+1) &= 1 - ax^2(k) + y(k) \\ y(k+1) &= bx(k) \end{aligned} \tag{3}$$

where a and b are control parameters. When $a \in (0.54, 2)$ and $b \in (0, 1)$, the Henon chaotic map will reach chaotic state. In this study, we select $a = 1.4$ and $b = 0.3$ and take the initial values $x(0)$ and $y(0)$ as keys. We repeatedly calculate (3) N_{total} times and then obtain two chaotic sequences: $\mathbf{x} = [x(1), x(2), \ldots, x(N_{\text{total}})]$ and $\mathbf{y} = [y(1), y(2), \ldots, y(N_{\text{total}})]$. Since elements of the chaotic sequences are decimals and pixel coordinates of image block are integers, the two sequences are mapped to integer sequences as follows.

$$D(i) = \left\lfloor \left(\bmod \left(x(i) * 2^{48}, S \right) \right) \right\rfloor, \quad i = 1, 2, \ldots, N_{\text{total}} \tag{4}$$

$$F(i) = \left\lfloor \left(\bmod \left(y(i) * 2^{48}, S \right) \right) \right\rfloor, \quad i = 1, 2, \ldots, N_{\text{total}} \tag{5}$$

where $D(i)$ is the i-th element of the array $\mathbf{D}$ used to record the x-coordinate of the start-point of the i-th image block and $F(i)$ is the i-th element of the array $\mathbf{F}$ used to record the y-coordinate of the start-point of the i-th image block.

Moreover, we exploit Lü chaotic map to generate secret matrix for XOR operation. The classical Lü chaotic map [43] is a three-dimensional discrete chaotic map that characterizes the transition between the Lorenz system [44] and the Chen system. The Lü chaotic map is calculated as follows.

$$\begin{aligned} x' &= a(y - x) \\ y' &= cy - xz \\ z' &= xy - bz \end{aligned} \tag{6}$$

where x_0, y_0, and z_0 are initial values of the Lü chaotic system and a, b, and c are its control parameters. The system is in chaos when $a = 36$, $b = 3$, and $c = 20$. Note that x_0, y_0, and z_0 are also taken as keys.

2.4. Key Generation. We generate content-based keys based on input image and use them to control the Lü chaotic map. This strategy can make our algorithm resistant to differential attack. To do so, the initial values x_0, y_0, and z_0 are calculated by the following equations.

$$\begin{aligned} x_0 &= \frac{\left[I(1,1) \oplus I(1,2) \oplus \cdots \oplus I(i,j) \right]}{255} \\ y_0 &= \frac{\left[(1/MN) \sum_{i=1}^{N} \sum_{j=1}^{M} I(i,j) \right]}{255} \\ z_0 &= x_0 + y_0 \end{aligned} \tag{7}$$

where $M \times N$ is the size of input image and $I(i, j)$ is the pixel value of input image $\mathbf{I}$, where $i \in [1, N]$ and $j \in [1, M]$. Obviously, x_0 is the decimal result of XOR operation between all image pixel values, and y_0 is the decimal result of mean value of all image pixels. Therefore, if an image pixel in the plaintext image is changed, the calculated results of x_0, y_0, and z_0 are also changed and then the chaotic sequences controlled by these keys will be different. This means that a changed input image will lead to a different encrypted result.

2.5. Encryption Scheme. The steps of our encryption scheme are as follows.

Step 1. The input image $\mathbf{I}$ is divided into random overlapping blocks according to the block size S and the overlapping sizes t_x and t_y. Calculate the total number of the overlapping blocks N_{total} and use a pseudo-random generator controlled by a key k_1 to generate N_{total} random numbers. Sort these random numbers and record the original positions of the sorted numbers in an array $P[N_{\text{total}}]$, which is used to determine the order of encrypting image blocks.

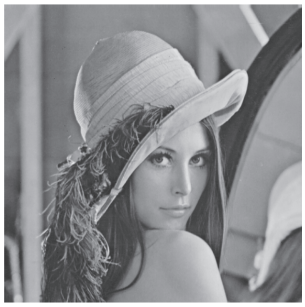
Step 2. Use Henon chaotic map to generate two arrays: $D[N_{\text{total}}]$ and $F[N_{\text{total}}]$. Note that $D(i)$ and $F(i)$ are the x-coordinate and y-coordinate of the start-point of the $P[i]$-th block, respectively, where $1 \le i \le N_{\text{total}}$. Take $(D(i), F(i))$ as the start-point of the $P[i]$-th block, encrypt the $P[i]$-th block by double spiral scans, and write the encrypted result to the image. Repeatedly conduct block encryption starting from $i = 1$ to $i = N_{\text{total}}$. After all blocks are processed, a scrambled image $\mathbf{J}$ is available.

Step 3. Calculate content-based keys x_0, y_0, and z_0 from input image and use them to control the Lü chaotic map to generate a secret matrix $\mathbf{G}$ sized $M \times N$. Then calculate $\mathbf{E} = \mathbf{J} \oplus \mathbf{G}$, where $\oplus$ represents the XOR operation of the corresponding elements of the matrices, and the matrix $\mathbf{E}$ is the encrypted image.

2.6. Decryption Scheme. The decryption scheme is a reverse process of our encryption scheme. Note that the keys of the Lü system are transmitted to the receiver via secure channel. Therefore, calculations of initial chaotic parameters x_0, y_0, and z_0 are not required at the receiver's side. Detailed decryption process is as follows.

Step 1. Use x_0, y_0, and z_0 to control the Lü chaotic map to generate a secret matrix $\mathbf{G}$. Calculate $\mathbf{J} = \mathbf{E} \oplus \mathbf{G}$.

Step 2. According to the block size S and the overlapping sizes t_x and t_y, the image matrix $\mathbf{J}$ is divided into random overlapping blocks. Compute the number of the overlapping blocks N_{total} and use a pseudo-random generator controlled



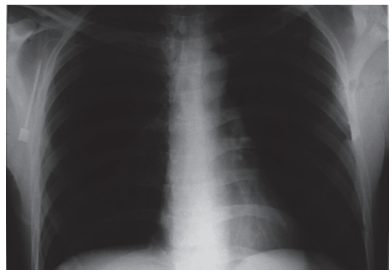
(a) Lena

(b) Fingerprint

(c) Woman

(d) Lake

(e) Goldhill



(f) ChestXray

FIGURE 5: Six original images.

by a key k_1 to generate N_{total} random numbers. Sort these random numbers and record the original positions of the sorted numbers in an array $P[N_{\text{total}}]$.

Step 3. Use Henon map to calculate the arrays $D[N_{\text{total}}]$ and $F[N_{\text{total}}]$. Take $(D(i), F(i))$ as the start-point of the $P[i]$-th block, decrypt the $P[i]$-th block by double spiral scans, and write the decrypted result to the image. Repeatedly conduct block decryption starting from $i = N_{\text{total}}$ to $i = 1$. After all blocks are processed, the decrypted image $\mathbf{I}$ is obtained. Note that this step is similar to the second step of encryption scheme. The major difference is the order of processing image blocks.

3. Key Space Analysis

Kerckhoffs's principle is a basic principle of the modern cryptography. It illustrates that "*A cryptographic system should be secure even if everything about the system, except the key, is public knowledge*" [45]. This implies that security of a cryptographic system is only dependent on secret keys, not the algorithm itself. In other words, security of an encryption algorithm mainly depends on the size of key space. In general, the larger the key space, the more secure the algorithm. The key space of our algorithm includes three parts. The first part is the initial values of the Henon chaotic map $x(0)$ and $y(0)$, and the random block pattern determined by the block size S and the overlapping sizes t_x and t_y. As $x(0)$ and $y(0)$ are floating numbers and require 64 bits' storage, their key space is $2^{64\times2} = 2^{128}$. The space of random block pattern is $(\min(M, N) - 31)(S - 1)^2$. The second part is the random key used to control the pseudo-random generator for block selection. The precision of the key is 64 bits. Since the key is used to randomly select block and the permutation number of blocks is $N_{\text{total}}!$, the valid key space is $\min(2^{64}, N_{\text{total}}!)$. When the block number $N_{\text{total}} \geq 21$, the following expression $N_{\text{total}}! > 2^{64}$ holds. In this case, the key space is 2^{64}. Otherwise, the key space is $N_{\text{total}}!$. The third part is the initial values of the Lü chaotic map. These three parameters are all floating numbers. Therefore, the key space of this part is $2^{64\times3} = 2^{192}$.

In summary, the key space of our algorithm is $2^{128}\times (\min(M, N) - 31)(S - 1)^2\times \min(2^{64}, N_{\text{total}}!) \times 2^{192} = (\min(M, N) - 31)(S - 1)^2\times \min(2^{64}, N_{\text{total}}!) \times 2^{320}$. For example, for a 512×512 image, if $S = 150$, $t_x = 70$, and $t_y = 70$, the total number of image blocks is 36. Consequently, our key space is $481 \times 149^2\times 2^{64}\times 2^{320} = 481 \times 149^2\times 2^{384} \approx 4.2 \times 10^{122}$, which is large enough to resist brute-force attacks [6]. For reference, the key spaces of the encryption algorithms reported in [6, 18, 19, 31] are 2^{104}, 2^{265}, 10^{117}, and 10^{59}, respectively, which are much smaller than our key space.

4. Experimental Results

In the experiment, the parameters of our algorithm are set as follows. The initial values of Henon map are $x(0) = 0.1$ and $y(0) = 0.3$. The block size is $S = 150$ and the overlapping sizes are $t_x = 70$ and $t_y = 70$. The key of the pseudo-random generator is $k_1 = 2$.

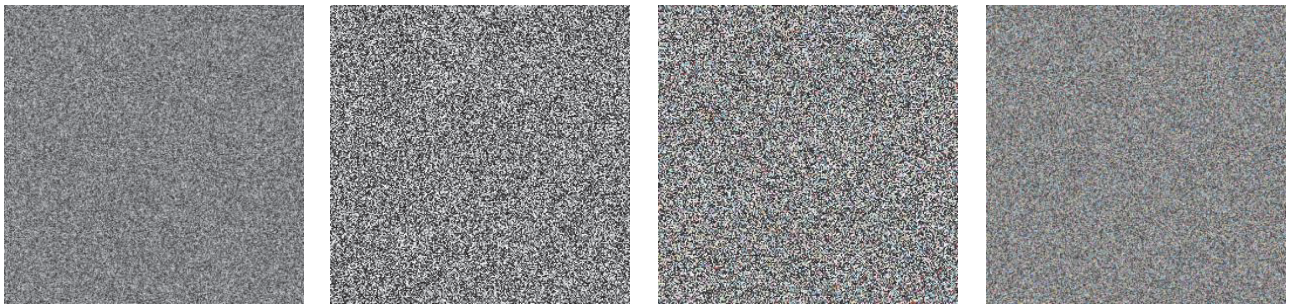
4.1. Encrypted Results. To validate our algorithm, some gray-scale images and color images are selected as test images. Figure 5 presents these test images and their detailed information is listed in Table 1. We apply our encryption scheme to these images and find that all encrypted images are chaotic images. Figures 6(a)–6(f) are the encrypted versions of the

Table 1: Test images.

Image	Lena	Fingerprint	Woman	Lake	Goldhill	ChestXray
Size	512×512	256×256	256×256	512×512	576×720	418×602
Type	Grayscale	Grayscale	Color	Color	Grayscale	Grayscale

Table 2: Correlation coefficients of the original images and their encrypted images.

Test image	Horizontal direction		Vertical direction		Diagonal direction	
	Original image	Encrypted image	Original image	Encrypted image	Original image	Encrypted image
Lena	0.9757	−0.0685	0.9692	0.0857	0.8820	0.0059
Fingerprint	0.9142	0.0933	0.9738	0.0616	0.8050	−0.0101
Woman	0.9655	0.0334	0.9758	0.1030	0.8825	−0.0007
Lake	0.9677	−0.0062	0.9638	−0.0137	0.9144	−0.0214
Goldhill	0.9780	−0.0351	0.9625	0.0556	0.6915	0.0330
ChestXray	0.9974	0.1183	0.9969	−0.0403	0.9364	−0.0059



(a) Encrypted Lena (b) Encrypted Fingerprint (c) Encrypted Woman (d) Encrypted Lake

(e) Encrypted Goldhill (f) Encrypted ChestXray

Figure 6: Encrypted images.

six test images generated by our encryption scheme. From the encryption results, it is observed that our encryption scheme can encrypt different size images, and all the encrypted images are noise-like images and meaningless. This means that our algorithm can effectively encrypt images.

4.2. Correlation Analysis. The pixel correlation is the degree of association of the gray values between pixels. Generally, the smaller the correlation between adjacent pixels of the encrypted image is, the better the performance of the encryption algorithm is. When calculating the pixel correlation in a certain direction (horizontal, vertical, or diagonal direction), several adjacent pixel pairs are randomly selected, and then the correlation coefficient is calculated. The formula of correlation coefficient is defined as follows.

$$\operatorname{corr}(\mathbf{x},\mathbf{y}) = \frac{E\left[\left(x-\mu_x\right)\left(y-\mu_y\right)\right]}{\sigma_x\sigma_y} \tag{8}$$

where μ_x and μ_y represent mean values of $\mathbf{x}$ and $\mathbf{y}$, σ_x and σ_y are the standard deviations of $\mathbf{x}$ and $\mathbf{y}$, and $E[\cdot]$ is the expectation function. The correlation coefficient is ranging from −1 to 1. The larger the correlation coefficient, the stronger the correlation between two pixel sequences. For a plaintext image, any two adjacent pairs of pixels usually have a strong correlation. A good performance encryption algorithm should break such correlation.

In the experiments, we randomly select 3000 pairs of adjacent pixels in horizontal, vertical, and diagonal directions, respectively, and calculate their correlation coefficients to verify the performance of our algorithm. Table 2 presents

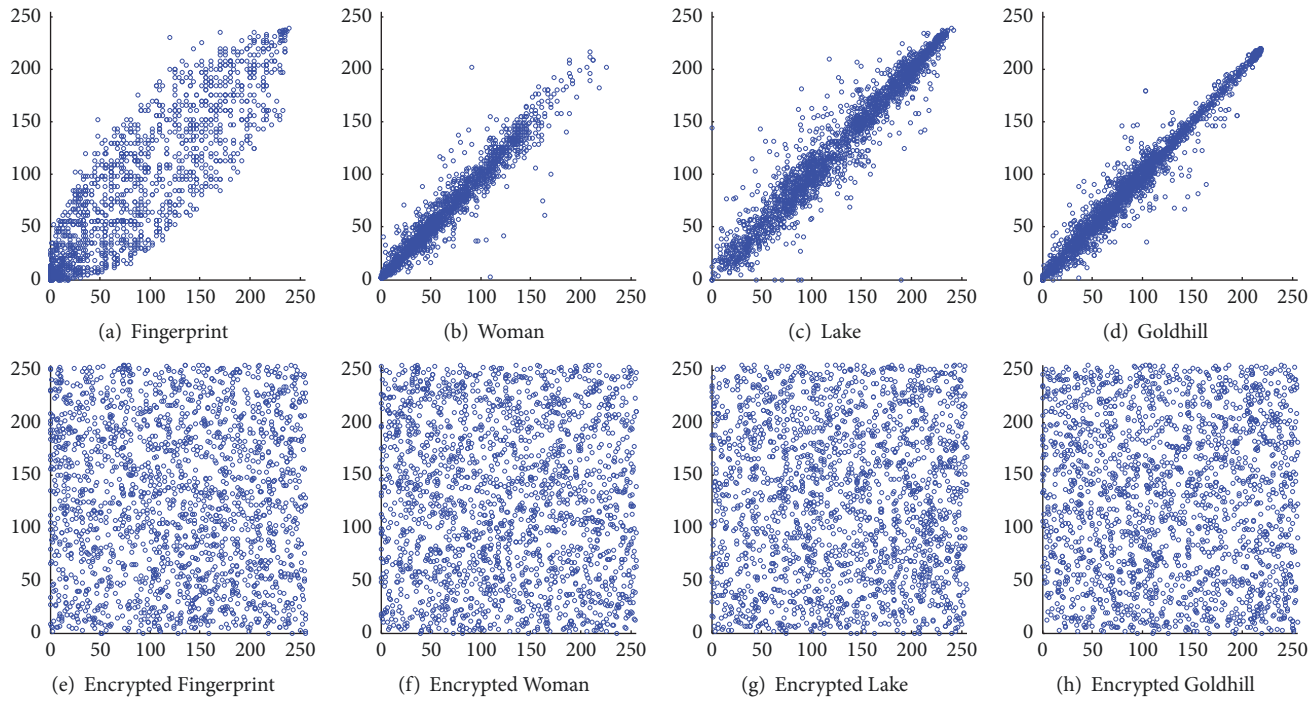


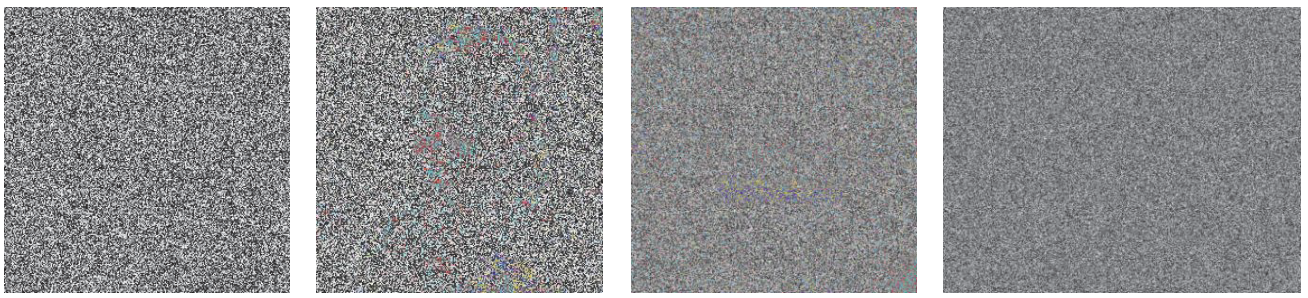
Figure 7: Distribution of adjacent pixels in horizontal direction.

Figure 8: Decrypted images with a wrong key: $x(0) = 0.100001$.

the correlation coefficients of original images and their encrypted versions. It can be found that the correlation coefficients of the original image are close to 1, while the correlation coefficients of the encrypted image are near 0. For space limitation, some typical visual results of distribution of adjacent pixels are illustrated in Figure 7. Figures 7(a)–7(d) are the pixel pair distribution of the test images (i.e., Fingerprint, Woman, Lake, and Goldhill) in horizontal direction, and Figures 7(e)–7(h) are the pixel pair distribution of their encrypted versions in the horizontal direction. The comparison shows that the pixel pairs of original images are concentrated around the diagonal with 45 degrees, showing high correlation coefficients, while the pixel pairs of the encrypted image are uniformly distributed over the entire value interval. This illustrates that our encryption scheme can effectively break the correlation between adjacent pixels.

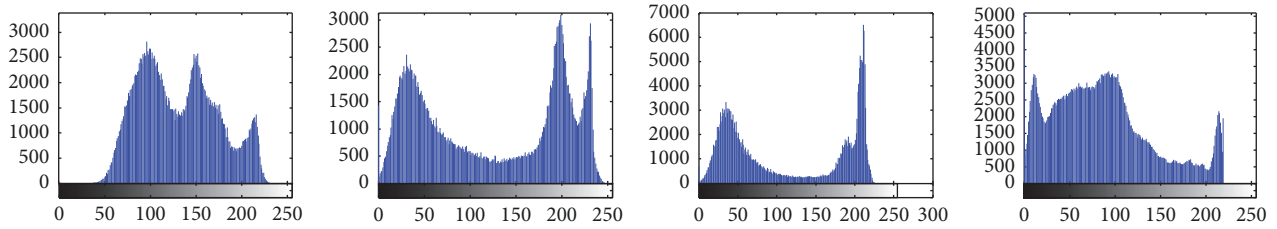
4.3. Key Sensitivity Analysis. A good encryption algorithm should be sensitive to the change of secret keys. This means that a slight difference of the keys should result in a great change in the decrypted image. If the difference between two encrypted images is very large, it is very difficult for attackers to break the algorithm through differential attacks. In the experiment, we change the initial values $x(0)$ or $y(0)$ of the Henon chaos system slightly and keep other decryption keys unchanged. We use these wrong keys to decrypt Figures 6(c)–6(f) and obtain the decrypted images as shown in Figure 8 and Figure 9, where Figure 8 uses a wrong key $x(0) = 0.100001$ (10^{-6} added) and Figure 9 uses a wrong key $y(0) = 0.300001$ (10^{-6} added). Obviously, the decrypted results with a wrong key are still noise-like images. This indicates that our algorithm is key-sensitive.

4.4. Histogram Analysis. Histogram is an important statistical feature used to show distribution of pixel values. It is



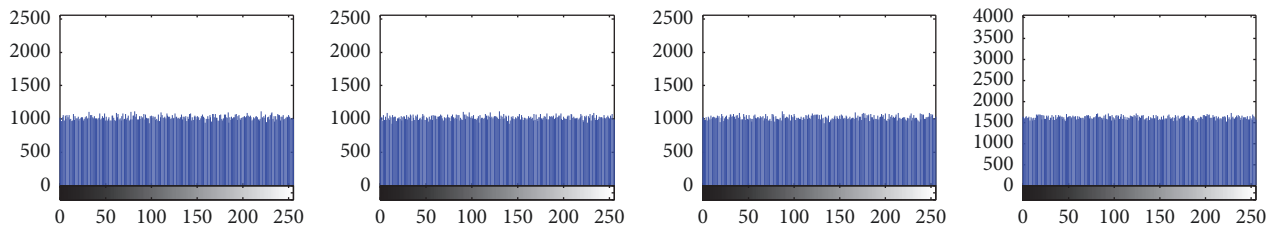
(a) Decrypted Woman (b) Decrypted Lake (c) Decrypted Goldhill (d) Decrypted ChestXray

Figure 9: Decrypted images with a wrong key: $y(0) = 0.300001$.



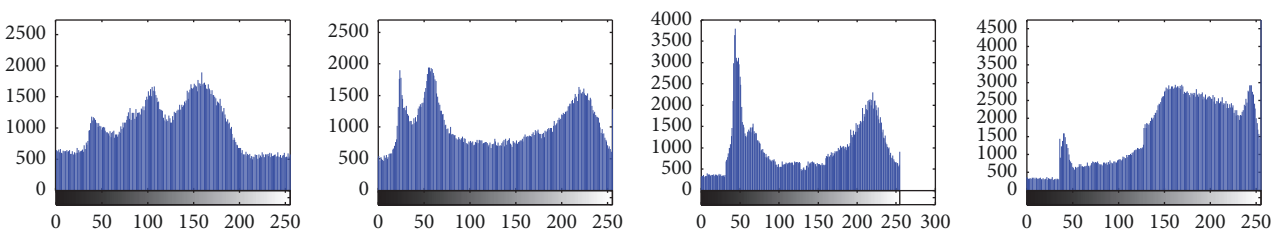
(a) Red component of Lake (b) Green component of Lake (c) Blue component of Lake (d) Goldhill

Figure 10: Histograms of original images.



(a) Red component of Lake (b) Green component of Lake (c) Blue component of Lake (d) Goldhill

Figure 11: Histograms of our encrypted images.



(a) Red component of Lake (b) Green component of Lake (c) Blue component of Lake (d) Goldhill

Figure 12: Histograms of the encrypted images generated by [6].

often exploited to measure performance of image encryption algorithms. In general, an efficient encryption algorithm is expected to generate encrypted image with uniformly distributed histogram. Figures 10(a)–10(c) are the histograms of red, green, and blue components of the color image Lake (Figure 5(d)), and Figure 10(d) is the histogram of the grayscale image Goldhill (Figure 5(e)). Figure 11 presents the histograms of the encrypted images generated by our algorithm, and Figure 12 shows the histograms of the encrypted results generated by the encryption algorithm [6]. It can be observed that our histograms are almost uniformly distributed and those histograms generated by the encryption algorithm [6] are rugged. Therefore, from the viewpoint of histogram, our algorithm is also secure.

Table 3: Comparison of the variance of histogram among different algorithms.

Image	Original variance	Variance of histogram of the encrypted result			
		[6]	[25]	[31]	Our
Lena	0.9725×10^6	0.1681×10^6	607.3	1339	1052.4
Fingerprint	1.5573×10^6	0.6517×10^6	34.96	277.51	329.11
Woman	0.1696×10^6	0.0793×10^6	369.33	260.1	245.37
Lake	0.7899×10^6	0.1514×10^6	3515.0	2146.5	1030.3
Goldhill	1.5506×10^6	0.9906×10^6	5538.7	2519.7	1737.4
ChestXray	6.1454×10^6	3.5074×10^6	1164.6	1164.1	1356.9
Average	1.8643×10^6	0.9248×10^6	2421.6	1284.5	958.58

Table 4: Entropy comparisons among different algorithms.

Image	Original	[6]	[25]	[31]	Our
Lena	7.2185	7.3634	7.9974	7.9991	7.9992
Fingerprint	5.1141	6.9960	7.9963	7.9963	7.9964
Woman	6.8981	7.4099	7.9972	7.9986	7.9991
Lake	7.7610	7.9427	7.9985	7.9993	7.9997
Goldhill	7.5195	7.6788	7.9984	7.9993	7.9995
ChestXray	5.8733	7.3865	7.9991	7.9985	7.9989
Average	6.7308	7.4629	7.9978	7.9985	7.9988

To quantitatively analyze histograms, the variance of histogram [46] is exploited to analyze performance, which is defined as follows.

$$V(Z) = \frac{1}{L^2}\sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\frac{1}{2}\left(z_i - z_j\right)^2 \tag{9}$$

where $Z = \{z_1, z_2, \ldots, z_{L\text{-}1}\}$, z_i $(0 \le i \le L-1)$, z_i is the total number of pixels with gray value equal to i, and $L = 256$ for the grayscale image. In general, the smaller the histogram variance, the more secure the encrypted image. Table 3 is histogram variance comparison between our algorithm and other algorithms [6, 25, 31]. From the results, it is observed that our results are all smaller than those of the compared algorithms, except two cases. Specifically, our results of Fingerprint and ChestXray are bigger than those of [25, 31]. However, for the average variance of histogram, our result is much smaller than those of the compared algorithms. It means that our algorithm has better performance than the compared algorithms [6, 25, 31] in terms of variance of histogram.

4.5. Entropy Analysis. Entropy [47] is often used to describe the uncertainty or randomness of an image. It is a useful metric for measuring security of image encryption. It is defined as follows.

$$H(\mathbf{E}) = -\sum_{i=0}^{L-1} P\left(e_i\right)\log_2 P\left(e_i\right) \tag{10}$$

where $\mathbf{E} = \{e_0, e_1, \ldots, e_{\mathrm{L}-1}\}$ and $P(e_i)$ is the possibility of the occurrence of e_i. For an image with 256 gray-level (e.g., $L = 256$), the theoretical maximum of the entropy is 8. In general, a bigger entropy means a more secure encryption algorithm. Table 4 lists entropy comparisons between our algorithm and the compared encryption algorithms [6, 25, 31]. It can be seen that the entropies of our algorithm are all close to 8 and are bigger than those of the compared algorithms [6, 25, 31]. Therefore, our algorithm is more secure than the compared algorithms [6, 25, 31] from the viewpoint of entropy.

4.6. Differential Attack. Differential attack is an effective method for analyzing security of cryptographic system. Generally, attacker slightly changes pixels of a plaintext image, generates a slightly altered encrypted image, and then analyzes the relationship between the generated encrypted image and the normal encrypted image. In practice, NPCR (Number of Pixels Change Rate) and UACI (Unified Average Changing Intensity) [48, 49] are often used to evaluate the capability of resisting differential attack. UACI and NPCR are defined as follows.

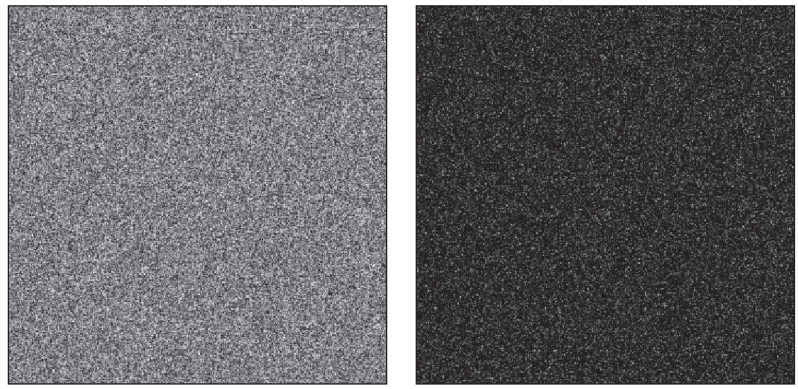
$$\mathrm{UACI}_{\mathrm{R,G,B}} = \frac{1}{W\times H}\left[\sum_{i,j}\frac{\left|C_{\mathrm{R,G,B}}(i,j) - C'_{\mathrm{R,G,B}}(i,j)\right|}{255}\right] \times 100\% \tag{11}$$

$$\mathrm{NPCR}_{\mathrm{R,G,B}} = \frac{\sum_{i,j} D_{\mathrm{R,G,B}}(i,j)}{W\times H}\times 100\% \tag{12}$$

where W and H are the image width and height and $C_{\mathrm{R,G,B}}(i,j)$ and $C'_{\mathrm{R,G,B}}(i,j)$ are the pixel values in the i-th

Table 5: Comparison results of NPCR and UACI (unit: %).

Algorithm	Image	(1, 1) NPCR/UACI	(128, 128) NPCR/UACI	(360, 360) NPCR/UACI	(512, 512) NPCR/UACI
Our	Lake	99.61/33.42	99.60/33.37	99.61/33.37	99.61/33.37
	Goldhill	99.61/33.75	99.61/33.48	99.62/33.75	99.62/33.71
	Lena	99.63/33.47	99.60/33.39	99.62/33.39	99.60/33.39
[6]	Lake	99.52/34.40	99.51/34.41	99.51/34.41	99.51/34.41
	Goldhill	99.56/34.31	99.59/34.36	99.59/34.36	99.56/34.36
	Lena	99.54/34.67	99.55/34.64	99.54/34.68	99.54/34.66
[31]	Lake	99.61/33.48	99.60/33.53	99.59/33.55	99.61/33.48
	Goldhill	99.60/33.53	99.58/33.52	99.60/33.55	99.61/33.49
	Lena	99.61/33.51	99.60/33.48	99.60/33.61	99.58/33.58



(a) Encryption result of the modified Lena

(b) Difference between (a) and the original encrypted Lena

Figure 13: Visual result of differential attack.

row and j-th column of two encrypted images, respectively. If $C_{R,G,B}(i, j) \neq C'_{R,G,B}(i, j)$, then $D_{R,G,B}(i, j) = 1$. Otherwise, $D_{R,G,B}(i, j) = 0$. Note that the theoretical values of NPCR and UACI are 100% and 33.33%, respectively. Generally, the closer to the theoretical values the calculated results, the more secure the encryption algorithm.

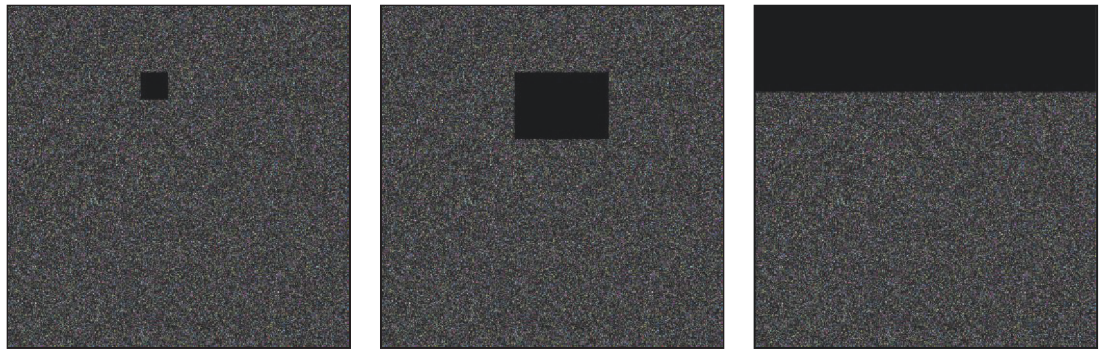
In the experiments, the standard images Lake, Goldhill, and Lena are selected as the test images, where their red components are used. Four locations (i.e., (1, 1), (128, 128), (360, 360), and (512, 512)) are selected in the test images for changing pixel values. For each test image, we just alter one pixel to generate a modified original image and then conduct encryption. Table 5 presents NPCR and UACI comparison results among our algorithm and the compared algorithms [6, 31]. It can be found that all NPCR values of our algorithm are greater than or equal to 99.60%, which is a little greater than or equal to those of the compared algorithms [6, 31]. For UACI, all our values are close to the theoretical value 33.33%, which is almost equal to the results of [31] but a little better than those of [6].

For space limitation, we only present a typical visual result. In the experiment, we change the pixel of Lena in the coordinate (1, 1) (its value is changed from 169 to 170) to generate the modified Lena and encrypt it with the same key. The encryption result is shown in Figure 13(a). The difference image between Figure 13(a) and the original encrypted result of the Lena is shown in Figure 13(b). From the result, it is observed that even if input image is slightly changed, our encryption result will be greatly changed. This indicates that our algorithm is highly sensitive to pixel change. Therefore, our algorithm can resist differential attack.

According to Kerckhoffs's principle [45], a successful cryptanalysis should accurately estimate secret keys (equivalent to recovering plaintext). For chosen plaintext attack, attacker can choose some specific plaintexts to calculate their corresponding ciphertexts. In practice, differential attack analysis is the most common way to achieve the chosen plaintext attack [18]. As well-known, diffusion technique can ensure security of a cryptographic algorithm against the chosen-plaintext attack [18]. To resist this attack, in this paper, we achieve diffusion by using content-based keys to control the Lü chaotic map for changing pixel values. In addition, we exploit double spiral scans to randomly scramble pixel positions. These techniques ensure that it is difficult to observe useful trace between secret keys and plaintext/ciphertext. This means that correct key estimation is almost impossible in practice. Therefore, the chosen-plaintext attack is impractical for our algorithm.

(a) Density is 0.01 (b) Density is 0.05 (c) Density is 0.10

Figure 14: Decrypted images under the attack of salt and pepper noise with different densities.



(a) 40 × 40 block (b) 100 × 140 block (c) 128 × 512 block

Figure 15: Encrypted images with block missing.

(a) 40 × 40 block missing (b) 100 × 140 block missing (c) 128 × 512 block missing

Figure 16: Decrypted images with block missing.

4.7. Robustness Test. To evaluate robustness performance of our algorithm, we attack many encrypted images with operations of salt & pepper noise and block missing, respectively. It is found that our algorithm can efficiently restore the decrypted images from the attacked encrypted images. For space limitation, typical examples are presented here. Firstly, we add salt & pepper noise with different densities (i.e., 0.01, 0.05, and 0.10) to the encrypted version of Lena (as shown in Figure 6(a)), decrypt the attacked encrypted images, and then obtain the decrypted images as shown in Figure 14. It can be observed that image noises are randomly distributed in the decrypted images and image qualities of the decrypted images gradually decrease with the increase of noise density. Secondly, we remove image blocks with different sizes from the encrypted version of Lena to generate the attacked encrypted versions as shown in Figure 15. We decrypt these attacked encrypted images and obtain the recovered results as shown in Figure 16. It is found that when the size of the missing block is small (e.g., 40 × 40), visual quality of the recovered image is good and the

Table 6: Computational time comparison among different algorithms (unit: second).

Algorithm	Lena	Fingerprint	Woman	Lake	Goldhill	ChestXray
[6]	4.601	2.350	4.065	4.059	5.747	4.444
[25]	7.788	2.984	5.838	13.098	12.231	7.782
[31]	4.821	2.718	5.774	6.514	4.382	5.219
Our	2.792	1.755	3.264	3.860	3.759	2.100

recovered image is almost the same as its original image. As the block size becomes large (e.g., 128 × 512), visual quality of the decrypted image decreases. But the appearance of the original image can be easily recognized from the decrypted image. From the above results, it can be concluded that our algorithm is robust against salt & pepper noise attack and block missing.

4.8. Computational Time Evaluation. To compare computational time, we exploit the assessed algorithms to encrypt the six test images, i.e., Lena, Fingerprint, Woman, Lake, Goldhill, and ChestXray, and record the running time of each algorithm. All algorithms are implemented with MATLAB R2014a and run on a computer with 3.4 GHz Intel Core i5-3570 CPU and 4.0 GB RAM. Table 6 presents computational time comparison among different algorithms. It is observed that our algorithm runs faster than the compared algorithms [6, 25, 31]. The fast speed of our algorithm is mainly contributed by the low complexity of double spiral scans.

5. Conclusions

In this paper, we have proposed an image encryption algorithm based on double spiral scans and chaotic maps. A key contribution is the double spiral scans, which can efficiently scramble pixels of image block. Moreover, content-based keys are generated and used to control the Lü chaotic system, so as to ensure our sensitivity to the change of input image. Many experiments have been done and the results have illustrated that our algorithm has good encryption performance and outperforms some popular image encryption algorithms.

Data Availability

The images used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (61562007, 61762017), Guangxi "Bagui Scholar" Teams for Innovation and Research, the Guangxi Natural Science Foundation (2017GXNSFAA198222), the Project of Guangxi Science and Technology (GuiKeAD17195062, GuiKeAD16380008), the Guangxi 1000-Plan of Training Middle-Aged/Young Teachers in Higher Education Institutions, the Guangxi Collaborative Innovation Center of Multi-Source Information Integration and Intelligent Processing, and the Project of the Guangxi Key Lab of Multi-Source Information Mining & Security (16-A-02-02).

References

[1] Z. Tang, X. Zhang, X. Li, and S. Zhang, "Robust image hashing with ring partition and invariant vector distance," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 200–214, 2016.

[2] Z. Tang, X. Zhang, and S. Zhang, "Robust perceptual image hashing based on ring partition and NMF," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 3, pp. 711–724, 2014.

[3] X.-Y. Wang, Y.-Q. Zhang, and X.-M. Bao, "A novel chaotic image encryption scheme using DNA sequence operations," *Optics and Lasers in Engineering*, vol. 73, pp. 53–61, 2015.

[4] Z. Tang and X. Zhang, "Secure image encryption without size limitation using Arnold transform and random strategies," *Journal of Multimedia*, vol. 6, no. 2, pp. 202–206, 2011.

[5] Z. Tang, X. Zhang, and W. Lan, "Efficient image encryption with block shuffling and chaotic map," *Multimedia Tools and Applications*, vol. 74, no. 15, pp. 5429–5448, 2015.

[6] G. Zhang and Q. Liu, "A novel image encryption method based on total shuffling scheme," *Optics Communications*, vol. 284, no. 12, pp. 2775–2780, 2011.

[7] M. Li, H. Fan, H. Ren, D. Lu, D. I. Xiao, and Y. Li, "Meaningful Image Encryption Based on Reversible Data Hiding in Compressive Sensing Domain," *Security and Communication Networks*, vol. 2018, Article ID 9803519, 12 pages, 2018.

[8] Y. Wang, Y. Zhao, Q. Zhou, and Z. Lin, "Image encryption using partitioned cellular automata," *Neurocomputing*, vol. 275, pp. 1318–1332, 2018.

[9] X. Y. Wang, X. Zhu, X. Wu, and Y. Zhang, "Image encryption algorithm based on multiple mixed hash functions and cyclic shift," *Optics and Lasers in Engineering*, vol. 107, pp. 370–379, 2018.

[10] U. Hayat and N. A. Azam, "A novel image encryption scheme based on an elliptic curve," *Signal Processing*, vol. 155, pp. 391–402, 2019.

[11] U.S. National Institute of Standards and Technology (NIST), "Announcing the advanced encryption standard (AES)," vol. 29, no. 8, pp. 2200–2203, Federal Information Processing Standards Publication, 2001.

[12] S. K. Banerjee, "High speed implementation of DES," *Computers & Security*, vol. 1, no. 3, pp. 261–267, 1982.

[13] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and Chaos*, vol. 8, no. 6, pp. 1259–1284, 1998.

[14] F. Sun, S. Liu, Z. Li, and Z. Lü, "A novel image encryption scheme based on spatial chaos map," *Chaos, Solitons & Fractals*, vol. 38, no. 3, pp. 631–640, 2008.

[15] Z. Liu, Q. Guo, L. Xu, M. A. Ahmad, and S. Liu, "Double image encryption by using iterative random binary encoding in gyrator domains," *Optics Express*, vol. 18, no. 11, pp. 12033–12043, 2010.

[16] 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.

[17] J. Wu, X. Liao, and B. Yang, "Cryptanalysis and enhancements of image encryption based on three-dimensional bit matrix permutation," *Signal Processing*, vol. 142, pp. 292–300, 2018.

[18] C. Fu, G. Zhang, M. Zhu, Z. Chen, and W. Lei, "A New Chaos-Based Color Image Encryption Scheme with an Efficient Substitution Keystream Generation Strategy," *Security and Communication Networks*, vol. 2018, Article ID 2708532, 13 pages, 2018.

[19] J. Wu, X. Liao, and B. Yang, "Color image encryption based on chaotic systems and elliptic curve ElGamal scheme," *Signal Processing*, vol. 141, pp. 109–124, 2017.

[20] M. Amin, O. S. Faragallah, and A. A. Abd El-Latif, "A chaotic block cipher algorithm for image cryptosystems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 11, pp. 3484–3497, 2010.

[21] A. A. Abd El-Latif, X. Niu, and M. Amin, "A new image cipher in time and frequency domains," *Optics Communications*, vol. 285, no. 21-22, pp. 4241–4251, 2012.

[22] A. A. Abd El-Latif and X. Niu, "A hybrid chaotic system and cyclic elliptic curve for image encryption," *AEÜ - International Journal of Electronics and Communications*, vol. 67, no. 2, pp. 136–143, 2013.

[23] Z. Tang, J. Song, X. Zhang, and R. Sun, "Multiple-image encryption with bit-plane decomposition and chaotic maps," *Optics and Lasers in Engineering*, vol. 80, pp. 1–11, 2016.

[24] L. Wang, H. Song, and P. Liu, "A novel hybrid color image encryption algorithm using two complex chaotic systems," *Optics and Lasers in Engineering*, vol. 77, pp. 118–125, 2016.

[25] Y. Abanda and A. Tiedeu, "Image encryption by chaos mixing," *IET Image Processing*, vol. 10, no. 10, pp. 742–750, 2016.

[26] A. Belazi, A. A. Abd El-Latif, R. Rhouma, and S. Belghith, "Selective image encryption scheme based on DWT, AES S-box and chaotic permutation," in *Proceedings of the 11th International Wireless Communications and Mobile Computing Conference, IWCMC 2015*, pp. 606–610, Croatia, August 2015.

[27] A. Belazi, A. A. Abd El-Latif, and S. Belghith, "A novel image encryption scheme based on substitution-permutation network and chaos," *Signal Processing*, vol. 128, pp. 155–170, 2016.

[28] Z. Tang, F. Wang, and X. Zhang, "Image encryption based on random projection partition and chaotic system," *Multimedia Tools and Applications*, vol. 76, no. 6, pp. 8257–8283, 2017.

[29] L. Li, B. Abd-El-Atty, A. A. A. El-Latif, and A. Ghoneim, "Quantum color image encryption based on multiple discrete chaotic systems," in *Proceedings of the 2017 Federated Conference on Computer Science and Information Systems, FedCSIS 2017*, pp. 555–559, Czech Republic, September 2017.

[30] R. Parvaz and M. Zarebnia, "A combination chaotic system and application in color image encryption," *Optics & Laser Technology*, vol. 101, pp. 30–41, 2018.

[31] H. Liu, A. Kadir, and X. Sun, "Chaos-based fast colour image encryption scheme with true random number keys from environmental noise," *IET Image Processing*, vol. 11, no. 5, pp. 324–332, 2017.

[32] X. Chen and C.-J. Hu, "Adaptive medical image encryption algorithm based on multiple chaotic mapping," *Saudi Journal of Biological Sciences*, vol. 24, no. 8, pp. 1821–1827, 2017.

[33] 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.

[34] J. Wu, X. Liao, and B. Yang, "Image encryption using 2D Hénon-Sine map and DNA approach," *Signal Processing*, vol. 153, pp. 11–23, 2018.

[35] P. Singh, A. K. Yadav, and K. Singh, "Phase image encryption in the fractional Hartley domain using Arnold transform and singular value decomposition," *Optics and Lasers in Engineering*, vol. 91, pp. 187–195, 2017.

[36] S. Vashisth, H. Singh, A. K. Yadav, and K. Singh, "Image encryption using fractional Mellin transform, structured phase filters, and phase retrieval," *Optik - International Journal for Light and Electron Optics*, vol. 125, no. 18, pp. 5309–5315, 2014.

[37] E. A. Naeem, M. M. Abd Elnaby, N. F. Soliman et al., "Efficient implementation of chaotic image encryption in transform domains," *The Journal of Systems and Software*, vol. 97, pp. 118–127, 2014.

[38] A. Belazi, A. A. Abd El-Latif, A.-V. Diaconu, R. Rhouma, and S. Belghith, "Chaos-based partial image encryption scheme based on linear fractional and lifting wavelet transforms," *Optics and Lasers in Engineering*, vol. 88, pp. 37–50, 2017.

[39] M. H. Annaby, M. A. Rushdi, and E. A. Nehary, "Color image encryption using random transforms, phase retrieval, chaotic maps, and diffusion," *Optics and Lasers in Engineering*, vol. 103, pp. 9–23, 2018.

[40] M. Zhang and X. Tong, "Joint image encryption and compression scheme based on IWT and SPIHT," *Optics and Lasers in Engineering*, vol. 90, pp. 254–274, 2017.

[41] C. Li, H. Li, F. Li, D. Wei, X. Yang, and J. Zhang, "Multiple-image encryption by using robust chaotic map in wavelet transform domain," *Optik - International Journal for Light and Electron Optics*, vol. 171, pp. 277–286, 2018.

[42] C. Wu, Y. Wang, Y. Chen, J. Wang, and Q. Wang, "Asymmetric encryption of multiple-image based on compressed sensing and phase-truncation in cylindrical diffraction domain," *Optics Communications*, vol. 431, pp. 203–209, 2019.

[43] J. Lü and G. Chen, "A new chaotic attractor coined," *International Journal of Bifurcation and Chaos*, vol. 12, no. 3, pp. 659–661, 2002.

[44] E. N. Lorenz, "Deterministic Nonperiodic Flow," *Journal of the Atmospheric Science*, vol. 20, pp. 130–141, 1963.

[45] "Kerckhoffs's principle," 2015, http://crypto-it.net/eng/theory/kerckhoffs.html.

[46] Y.-Q. Zhang and X.-Y. Wang, "A symmetric image encryption algorithm based on mixed linear-nonlinear coupled map lattice," *Information Sciences*, vol. 273, pp. 329–351, 2014.

[47] Z. Tang, X. Zhang, L. Huang, and Y. Dai, "Robust image hashing using ring-based entropies," *Signal Processing*, vol. 93, no. 7, pp. 2061–2069, 2013.

[48] Q. Wang, Q. Guo, and J. Zhou, "Double image encryption based on linear blend operation and random phase encoding in fractional Fourier domain," *Optics Communications*, vol. 285, no. 21-22, pp. 4317–4323, 2012.

[49] Y. Wang, C. Quan, and C. J. Tay, "Nonlinear multiple-image encryption based on mixture retrieval algorithm in Fresnel domain," *Optics Communications*, vol. 330, pp. 91–98, 2014.