The fast image encryption algorithm based on substitution and diffusion

Yong Zhang[1] and Xiaoyang Jia[2]

[1]School of Software and Communication Engineering, Jiangxi University of Finance and Economics

Nanchang, P. R. China

[e-mail: zhangyong@jxufe.edu.cn]

[2]Changchun China Optical Science & Technology Museum

Changchun, P.R. China

[e-mail: jiaxiaoyang_cc@126.com]

*Corresponding author: Yong Zhang

Received December 27, 2017; revised February 19, 2018; accepted May 8, 2018; published September 30, 2018

Abstract

A fast image encryption system based on substitution and diffusion was proposed, which includes one covering process, one substitution process and two diffusion processes. At first, Chen's chaotic system together with an external 256-bit long secret key was used to generate the key streams for image encryption, in which the initial values of Chen's chaotic system were regarded as the public key. Then the plain image was masked by the covering process. After that the resulting image was substituted with the disturbed S-Box of AES. Finally, the substituted image was diffused twice with the add-modulo operations as the core to obtain the cipher image. Simulation analysis and comparison results with AES and some existing image cryptosystems show that the proposed image cryptosystem possesses the merits of fast encryption/decryption speed, good statistical characteristics, strong sensitivity and etc., and can be used as a candidate system of network security communication.

Keywords: Information security; image encryption; chaos; substitution; diffusion

1. Introduction

Chaotic image cryptology research has become a hotspot in the field of information security. And that includes three main research directions: First is to study the new chaotic systems which can produce cryptographic key streams [1-5]. The direction serves as an important application of chaos theory to give birth to the study of new chaotic systems. Second is to study the new image encryption schemes [6-10]. The research results of this direction show that the variety of novel image confusion and diffusion algorithms has emerged in recent decades. Third is to study the crypto-analysis of image cryptosystem [11-15]. Accompanied by the emergence of a large number of image encryption systems, quantities of crypto-analysis systems have been proposed, which are simulating the further research on new high-intensity image cryptography.

Recently, Zhang and Tang suggested an image cryptosystem with identical encryption and decryption algorithm [16]. The length of secret key is 64d bits (where, d is a positive integer). The encryption/decryption process consists of two diffusion operations, two matrix rotations and one plaintext-related scrambling. Due to skillful using the matrix rotations, the encryption process and decryption process in their scheme share the same algorithm. Chai et al. proposed an image encryption system using the memristive hyper-chaotic system, DNA encoding and cellular automaton [17]. The system employed the SHA-256 Hash value of plain image as the secret key of size 256 bits. The secret key was then converted into four initial values of memristive hyper-chaotic system to iteratively generate the key streams. Finally, the key streams were used in confusion and diffusion processes under the rules of DNA encoding and cellular automaton. From the point of view of cryptography, Chai et al.'s system is essentially a complex one-time pad system, which has certain theoretical research value, but almost no application value. Diaconu presented a new image encryption system using bit-level permutation [18]. In this system each row of image was converted into a bit sequence, and the bit-shift operation was based on the whole row of bit sequence. This algorithm is very suitable for hardware implementation, but not suitable for computer software implementation. At the same time, Fan et al. cracked this image encryption system by means of chosen plaintext attack [19]. Zhang et al. delivered an image encryption algorithm with typical confusion-diffusion architecture based on Chen's chaotic system and 3D Cat map [20]. In order to improve the processing speed of encryption, their scheme used 3D bit permutation algorithm based on sorting method and 3D Cat transform matrix. However, Zhang et al. pointed out that the system in [20] cannot frustrate the chosen plaintext attack [21].

All of the aforesaid image encryption systems used the confusion-diffusion architecture. In this architecture the diffusion algorithms usually employ add-then-modulo operation or XOR operation, while the confusion algorithms are the main innovation points of cryptographic scheme [22]. However, some of the research results have obvious security

vulnerabilities, so as to fail in confronting the chosen/known plaintext attacks [19,21]. Aiming at reducing the above security problems as much as possible, this paper proposed a novel image cryptosystem based on the combination of substitution and diffusion algorithms. The substitution technology instead of confusion is used for image information hiding, hoping to get better information security performance. This paper starts with the research on key streams generator with an external key and the Chen's chaotic system. The initial values of Chen's system are taken as the public key. Then, the key streams are used to cover the plain image. After that the covered image is encrypted into cipher image by means of substitution-diffusion-diffusion architecture. Finally, the simulation test and theoretical analysis are performed to verify that the proposed image cryptosystem possesses good security characteristics and fast encryption/decryption speed.

The rest of this paper is organized as follows: Section 2 introduces the chaotic pseudo random sequence generator; Section 3 details the structure of proposed image cryptosystem and its algorithm implementation steps; Section 4 gives the typical representatives of simulation results; Section 5 performs the security performance analyses of proposed image cryptosystem, and compares them with some of existing image cryptosystems; Finally, Section 6 summarizes the full paper.

2. Pseudo random sequence generator

Chaotic systems are widely used to generate pseudo random sequences due to their extreme sensitivity to initial values and parameters, and ergodicity of state space. This section presents a method for generating pseudo random sequences by multiple initial states acting on a chaotic system. Without losing generality, take a one-dimensional chaotic system as an example. Assume that the discrete state evolution equation of one-dimensional chaotic system is described by

$$x(n)=F(x(n\text{-}1);\boldsymbol{u}) \tag{1}$$

Where, $\boldsymbol{u}$ is the parameter vector, and $x(n)$ is the n-th state.

Now, let there be a total of k pieces of initial states, denoted by $\{s_i\}$, i=1,2,...,k. The steps of generating pseudo random sequences are as follows:

Step 1. Randomly take a state value x_0 from the state space of Eq. (1) as the initial value of Eq. (1), then iterate it for l times (l>10). Denote the last 10 iteration values as $\{x_{l\text{-}9},x_{l\text{-}8},x_{l\text{-}7},...,x_{l\text{-}1},x_l\}$, and then calculate λ by the following formula

$$\lambda = \frac{|x_l|}{\Sigma_{j=0}^{9}|x_{l-j}|} \tag{2}$$

Where, $|x|$ returns the absolute value of x.

Step 2. Let i=1.

Step 3. Let new $x_l=(1\text{-}\lambda)x_l+\lambda s_i$. Take the new x_l as the initial value x_0 of Eq. (1), and then iterate it for l times. Denote the last 10 iteration values as $\{x_{l\text{-}9},x_{l\text{-}8},x_{l\text{-}7},...,x_{l\text{-}1},x_l\}$, and then

calculate the new λ also by Eq. (2).

Step 4. Let $i=i+1$. If $i<k$, then jump to Step 3; otherwise, continue to Step 5.

Step 5. Let new $x_l=(1-\lambda)x_l+\lambda s_k$. Take the new x_l as the initial value x_0 of Eq. (1), and then iterate it for l times to bypass the transient state. Then, continue to iterate Eq. (1) for L times to get a floating-point state sequence of length L, denoted by $\boldsymbol{x}=\{x_j,j=1,2,...,L\}$.

Step 6. Convert the sequence $\boldsymbol{x}$ into an integer sequence, denoted by $\boldsymbol{t}=\{t_j,j=1,2,...,L\}$, using the following formula.

$$t_j=(|x_j|\times 2^{32}) \bmod 256,\ j=1,2,...,L \tag{3}$$

And the sequence $\boldsymbol{t}$ is the key stream for image encryption.

For a multidimensional chaotic system, one can choose an optional state form the state vector of discretized state equation as the role of the above x of one-dimensional chaotic system. Then, one can use the above method to generate pseudo random sequence by multiple initial states acting on the multidimensional chaotic system.

The proposed image cryptosystem is as shown in **Fig. 1**. In **Fig. 1**, three pseudo random matrices $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$, and three pseudo random numbers r_1, r_2 and r_3, are generated by the secret key $\boldsymbol{K}$ with the help of key stream generator. Here, the secret key $\boldsymbol{K}$ used is of length 256 bits and consists of 32 pieces of 8-bit unsigned bytes, namely, $\boldsymbol{K}=\{K_i,\ i=1,2,...,32\}$, where, each K_i is a byte. The $\boldsymbol{K}$ is transformed into 32 pieces of initial states, denoted by $\boldsymbol{s}=\{s_i,\ i=1,2,...,32\}$, using the following formula.

$$s_i=K_i/256,\ i=1,2,...,32 \tag{4}$$

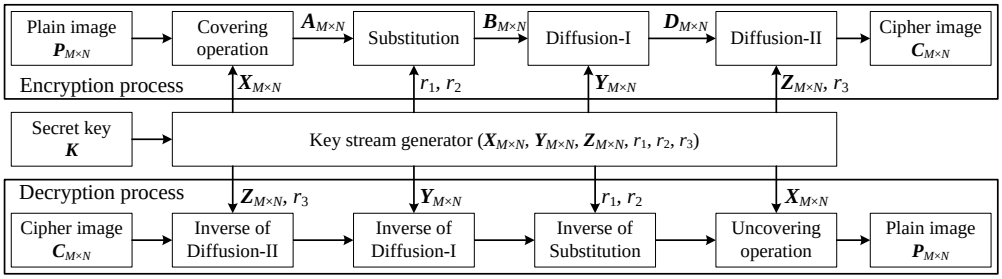
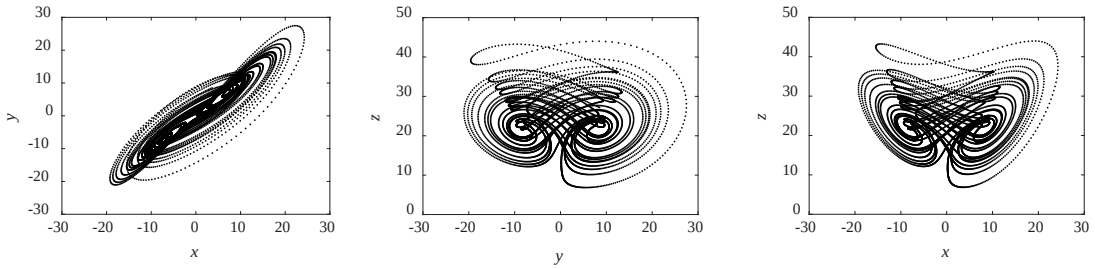


Fig. 1. Proposed image cryptosystem

Any chaotic system that can generate cryptographic pseudo random numbers can be chosen as the mapping F in Eq. (1). Here, Chen's chaotic system [23] is selected, and its formula is as follows.

$$\begin{cases} \dot{x}=a(y-x) \\ \dot{y}=(c-a)x-xz+cy \\ \dot{z}=xy-bz \end{cases} \tag{5}$$

Where, a=35, b=3, $c\in$[20,28.4]. When c=28 (used in this paper), the phase portrait of Chen's system is as shown in **Fig. 2** (with the size of discretized step being 0.002).



(a) x vs. y (b) y vs. z (c) x vs. z

Fig. 2. Phase portraits of Chen's system.

Randomly select a group of state values (x_0,y_0,z_0) from the state space of Eq. (5) as the initial values of iteration (which are open and serve as public key). Here, let l=100. Assume that the plain image is of size $M\times N$, and the state x is selected as the disturbed state. Employ the above method with multiple initial states $\boldsymbol{s}$ and (x_0,y_0,z_0) acting on Chen's system to iteratively generate three chaotic state sequences $\{x_i\}$, $\{y_i\}$ and $\{z_i\}$ all of length $L/3$, where, $L=3MN+3$. Then, convert these three sequences into an integer sequence $\boldsymbol{t}=\{t_i\}$ of length $L=3MN+3$ using the following formulae.

$$t_{3j-2}=[|x_j|\times 2^{32}] \bmod 256,\ t_{3j-1}=[|y_j|\times 2^{32}] \bmod 256,\ t_{3j}=[|z_j|\times 2^{32}] \bmod 256 \quad (6)$$

Where, j=1,2,...,$L/3$, and $[x]$ returns the integer part of x.

Then, generate $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$, r_1, r_2 and r_3 form the sequence $\boldsymbol{t}$ with the following formulae.

$$\boldsymbol{X}(i,j)=t_{N(i-1)+j},\ \boldsymbol{Y}(i,j)=t_{MN+N(i-1)+j},\ \boldsymbol{Z}(i,j)=t_{2MN+N(i-1)+j},\ i=1,2,...,M,\ j=1,2,...,N \quad (7)$$

$$r_1=t_{3MN+1},\ r_2=t_{3MN+2},\ r_3=t_{3MN+3} \quad (8)$$

The matrices $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ and the pseudo random number r_1, r_2 and r_3 are the key streams of proposed image cryptosystem.

To prove that the sequence $\boldsymbol{t}$ possesses good cryptographic characteristics, the test items of FIPS140-2 [24], such as Monobit test, Poker test, Runs test and Long run test, are performed on the sequence $\boldsymbol{t}$. Without loss of generality, assume that the initial values of Chen's system are (x_0,y_0,z_0) = (0.56, 0.809, 12.347), the step size is 0.002, l=100, and the secret key $\boldsymbol{K}$={65, 212, 155, 202, 4, 27, 202, 161, 7, 233, 146, 36, 229, 154, 157, 109, 193, 182, 117, 206, 39, 244, 177, 216, 34, 255, 74, 55, 12, 223, 205, 112}(in decimal format). By the above method a sequence $\boldsymbol{t}$ of length 2500 bytes is obtained. Then, the sequence $\boldsymbol{t}$ is converted into a bit sequence of length 20000 bits, whose test results are listed in **Table 1**.

Table 1. FIPS140-2 test results of sequence $\boldsymbol{t}$

Test item	Monobit test	Poker test	Runs test						Long run test
			Length of runs						
			1	2	3	4	5	>5	(>25)
bit 0	9878	18.37	2538	1206	614	327	153	145	0
bit 1	10122		2419	1291	617	320	170	166	0

Theoretical value	9725 ~ 10725	2.16 ~ 46.17	2315 ~ 2685	1114 ~ 1386	527 ~ 723	240 ~ 384	103 ~ 209	103 ~ 209	0
Results	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass

As can be seen from **Table 1**, the sequence ***t*** passes all the test items of FIPS140-2, which shows that the proposed pseudo random number generator can produce cryptographic pseudo random sequences with good statistical characteristics.

The following analyzes the autocorrelation of the sequence ***t***. Convert ***t*** into a sequence of zero mean, denoted by ***s***, with the following formula.

$$\boldsymbol{s}(i)=2\boldsymbol{t}(i)-255,\ i=1,2,...,L \tag{9}$$

Where, L=12000, and ***t*** is generated by the above method with the same initial values and secret key ***K***. Then, calculate the cyclic autocorrelation of ***s*** and illustrate the result of correlation coefficient in **Fig. 3**.

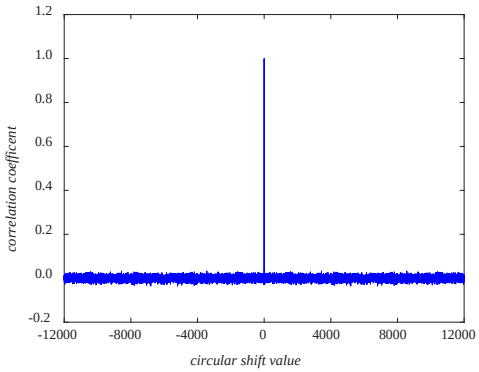


Fig. 3. Cyclic autocorrelation of sequence ***s***

It can be seen from **Fig. 3** that the correlation coefficient curve of the sequence ***s*** is thumbtack-like, which shows that the sequence ***s*** (or its predecessor sequence ***t***) has no correlation between the cyclic-shift sequences. So the adjacent data in the sequence ***t*** are not predictable, i.e. the sequence ***t*** can be used as the key stream for image encryption.

3. Image cryptosystem

In the proposed image cryptosystem shown in **Fig. 1**, the encryption process includes one covering operation, one substitution process and two diffusion processes. The decryption process is the inverse of the encryption process. Both the encryption process and the decryption process have no permutation or confusion process.

In **Fig. 1**, the plain image $\boldsymbol{P}$ is assumed to be an 8-bit grayscale image of size $M \times N$, and the secret key $\boldsymbol{K}$ is of length 256 bits. The matrices $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\boldsymbol{Z}$ and the three pseudo random numbers r_1, r_2 and r_3 used as the key streams in encryption/decryption process are generated by the method described in Section 2 with the secret key $\boldsymbol{K}$. The details of encryption and decryption processes are as follows.

(1) Covering operation and its inverse

From **Fig. 1**, the covering operation converts the plain image $\boldsymbol{P}$ into a matrix, denoted by $\boldsymbol{A}$, using the following formula.

$$\boldsymbol{A}(i,j)=\boldsymbol{P}(i,j) \oplus \boldsymbol{X}(i,j),\ i=1,2,...,M,\ j=1,2,...,N \tag{10}$$

Where, $\oplus$ represents bitwise XOR (exclusive or) operation (the same meaning thereinafter). The covering operation uses the matrix $\boldsymbol{X}$ to mask the plain image $\boldsymbol{P}$ with the XOR operation. Because the XOR operation is invertible, the inverse of Eq. (10) is as follows.

$$\boldsymbol{P}(i,j)=\boldsymbol{A}(i,j) \oplus \boldsymbol{X}(i,j),\ i=1,2,...,M,\ j=1,2,...,N \tag{11}$$

From Eq. (11), one can recover $\boldsymbol{P}$ from $\boldsymbol{A}$ with the help of $\boldsymbol{X}$.

(2) Substitution process and its inverse

The S-Box of AES [25] shown in **Table 2** is used in the substitution process.

For any 8-bit byte a, its higher 4 bits are considered as x, and the lower 4 bits as y, then one can get a unique 8-bit byte b by looking up the **Table 2** with the coordinates (x,y), which is denoted by

$$a \xrightarrow{\text{S-Box}} b \tag{12}$$

The concrete steps of substitution are as follows.

Step 1. XOR each element of S-Box in **Table 2** with r_1 to get a new S-Box. The new S-Box is employed in the following steps.

Step 2. Let $\boldsymbol{T}(1,1)=[\boldsymbol{A}(1,1)+r_2] \bmod 256$, and obtain $\boldsymbol{B}(1,1)$ from $\boldsymbol{T}(1,1)$ by looking up the S-Box. This process is denoted by

$$[\boldsymbol{A}(1,1)+r_2] \bmod 256 = \boldsymbol{T}(1,1) \xrightarrow{\text{S-Box}} \boldsymbol{B}(1,1) \tag{13}$$

Step 3. Let $j=2$ to N, then perform the following for each j.

$$[\boldsymbol{A}(1,j)+\boldsymbol{A}(1,j\text{-}1)+\boldsymbol{T}(1,j\text{-}1)] \bmod 256 = \boldsymbol{T}(1,j) \xrightarrow{\text{S-Box}} \boldsymbol{B}(1,j) \tag{14}$$

Step 4. Replace $\boldsymbol{A}(2,1)$ by $\boldsymbol{B}(2,1)$ with the following formula.

$$[\boldsymbol{A}(2,1)+\boldsymbol{A}(1,1)+\boldsymbol{A}(1,\mathrm{N})+\boldsymbol{T}(1,1)+\boldsymbol{T}(1,N)] \bmod 256 = \boldsymbol{T}(2,1) \xrightarrow{\text{S-Box}} \boldsymbol{B}(2,1) \tag{15}$$

Step 5. Let $i=3$ to M, then do the following for each i.

$$[\boldsymbol{A}(i,1)+\boldsymbol{A}(i\text{-}1,1)+\boldsymbol{T}(i\text{-}1,1)] \bmod 256 = \boldsymbol{T}(i,1) \xrightarrow{\text{S-Box}} \boldsymbol{B}(i,1) \tag{16}$$

Table 2. S-Box of AES

S-Box		*y*															
		0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
x	*0*	99	124	119	123	242	107	111	197	48	1	103	43	254	215	171	118
	1	202	130	201	125	250	89	71	240	173	212	162	175	156	164	114	192
	2	183	253	147	38	54	63	247	204	52	165	229	241	113	216	49	21
	3	4	199	35	195	24	150	5	154	7	18	128	226	235	39	178	117
	4	9	131	44	26	27	110	90	160	82	59	214	179	41	227	47	132
	5	83	209	0	237	32	252	177	91	106	203	190	57	74	76	88	207
	6	208	239	170	251	67	77	51	133	69	249	2	127	80	60	159	168
	7	81	163	64	143	146	157	56	245	188	182	218	33	16	255	243	210
	8	205	12	19	236	95	151	68	23	196	167	126	61	100	93	25	115
	9	96	129	79	220	34	42	144	136	70	238	184	20	222	94	11	219
	10	224	50	58	10	73	6	36	92	194	211	172	98	145	149	228	121
	11	231	200	55	109	141	213	78	169	108	86	244	234	101	122	174	8
	12	186	120	37	46	28	166	180	198	232	221	116	31	75	189	139	138
	13	112	62	181	102	72	3	246	14	97	53	87	185	134	193	29	158
	14	225	248	152	17	105	217	142	148	155	30	135	233	206	85	40	223
	15	140	161	137	13	191	230	66	104	65	153	45	15	176	84	187	22

Step 6. Let i=2 to M and j=2 to N, then do the following for each combination of i and j.

$$[\boldsymbol{A}(i,j)+\boldsymbol{A}(i,j\text{-}1)+\boldsymbol{A}(i\text{-}1,j)+\boldsymbol{T}(i,j\text{-}1)+\boldsymbol{T}(i\text{-}1,j)] \bmod 256 = \boldsymbol{T}(i,j) \xrightarrow{\text{S-Box}} \boldsymbol{B}(i,j) \tag{17}$$

After the above processing, the matrix $\boldsymbol{A}$ is replaced by the matrix $\boldsymbol{B}$ with the help of r_1, r_2 and S-Box.

The above substitution process is reversible, and the matrix $\boldsymbol{A}$ can be recovered from the matrix $\boldsymbol{B}$ with the help of r_1, r_2 and S-Box. The inverse of the substitution is as follows:

Step 1. Denote the inverse of S-Box in **Table 2** by S-Box^{-1}.

Step 2. XOR each element of $\boldsymbol{B}$ with r_1 to get a new matrix, denoted by $\boldsymbol{G}$.

Step 3. Obtain $\boldsymbol{T}(1,1)$ from $\boldsymbol{G}(1,1)$ by looking up S-Box^{-1}, then get $\boldsymbol{A}(1,1)$ by calculating $[\boldsymbol{T}(1,1)\text{-}r_2\text{+}256] \bmod 256$. This process is denoted by

$$\boldsymbol{G}(1,1) \xrightarrow{\text{S-Box}^{-1}} \boldsymbol{T}(1,1), \text{ then } [\boldsymbol{T}(1,1) + 256 - r_2] \bmod 256 = \boldsymbol{A}(1,1) \tag{18}$$

Step 4. Let j=2 to N, then carry out the following formula for each j.

$$\boldsymbol{G}(1,j) \xrightarrow{\text{S-Box}^{-1}} \boldsymbol{T}(1,j), \text{ then } [\boldsymbol{T}(1,j)+256\times2-\boldsymbol{T}(1,j\text{-}1)-\boldsymbol{A}(1,j\text{-}1)] \bmod 256 = \boldsymbol{A}(1,j) \tag{19}$$

Step 5. Recover $\boldsymbol{A}(2,1)$ from $\boldsymbol{G}(2,1)$ by the following formula.

$$\boldsymbol{G}(2,1) \xrightarrow{\text{S-Box}^{-1}} \boldsymbol{T}(2,1), \text{ then } [\boldsymbol{T}(2,1)+256\times4-\boldsymbol{T}(1,1)-\boldsymbol{T}(1,N)-\boldsymbol{A}(1,1)-\boldsymbol{A}(1,N)] \bmod 256 = \boldsymbol{A}(2,1) \tag{20}$$

Step 6. Let i=3 to M, then do the following formula for each i.

$$\boldsymbol{G}(i,1) \xrightarrow{\text{S-Box}^{-1}} \boldsymbol{T}(i,1), \text{ then } [\boldsymbol{T}(i,1) + 256\times 2 - \boldsymbol{T}(i-1,1) - \boldsymbol{A}(i-1,1)] \bmod 256 = \boldsymbol{A}(i,1) \quad (21)$$

Step 7. Let i=2 to M, and j=2 to N, then do the following formula for each combination of i and j.

$$\boldsymbol{G}(i,j) \xrightarrow{\text{S-Box}^{-1}} \boldsymbol{T}(i,j), \text{ then } [\boldsymbol{T}(i,j) + 256\times 4 - \boldsymbol{T}(i,j-1) - \boldsymbol{T}(i-1,j) - \boldsymbol{A}(i,j-1) - \boldsymbol{A}(i-1,j)] \bmod 256 = \boldsymbol{A}(i,j) \quad (22)$$

After the above processing, the matrix $\boldsymbol{A}$ is recovered from the matrix $\boldsymbol{B}$ with the help of r_1, r_2 and S-Box.

(3) Diffusion-I process and its inverse

The diffusion process disperses the information of each pixel in the original image to as many pixels as possible to get the resultant image. The multiplication based on Galois field GF(2^8) is used in the diffusion process with the primitive polynomial being $m(x)=x^8+x^4+x^3+x+1$. In **Fig. 1**, the diffusion process includes two stages, namely, the diffusion-I and the diffusion-II. The latter will be discussed in the next part labeled with (4) of subhead. The specific steps of diffusion-I process are as follows:

Step 1. Obtain $\boldsymbol{D}(M,N)$ from $\boldsymbol{B}(M,N)$ and $\boldsymbol{Y}(M,N)$ using the following

$$\boldsymbol{D}(M,N) = [\boldsymbol{B}(M,N) + \boldsymbol{Y}(M,N)] \bmod 256 \quad (23)$$

Step 2. Let j=N-1 to 1, then calculate $\boldsymbol{D}(M,j)$ for each j with the following formula.

$$\boldsymbol{D}(M,j) = [\boldsymbol{B}(M,j) + \boldsymbol{B}(M,j+1) + \boldsymbol{D}(M,j+1) + \boldsymbol{Y}(M,j)] \bmod 256 \quad (24)$$

Step 3. Generate $\boldsymbol{D}(M\text{-}1,N)$ from $\boldsymbol{B}(M\text{-}1,N)$, $\boldsymbol{Y}(M\text{-}1,N)$, $\boldsymbol{B}(M,1)$ and $\boldsymbol{D}(M,1)$ with the following formula.

$$\boldsymbol{D}(M\text{-}1,N) = [\boldsymbol{B}(M\text{-}1,N) + \mathrm{B}(\mathrm{M},1) + \boldsymbol{D}(M,1) + \boldsymbol{Y}(M\text{-}1,N)] \bmod 256 \quad (25)$$

Step 4. Let i=M-2 to 1, then calculate $\boldsymbol{D}(i,N)$ for each i with the following formula.

$$\boldsymbol{D}(i,N) = [\boldsymbol{B}(i,N) + \boldsymbol{B}(i+1,N) + \boldsymbol{D}(i+1,N) + \boldsymbol{Y}(i,N)] \bmod 256 \quad (26)$$

Step 5. Let i=M-1 to 1, and j=N-1 to 1, then calculate $\boldsymbol{D}(i,j)$ for each combination of i and j with the following formula.

$$\boldsymbol{D}(i,j) = \{\boldsymbol{B}(i,j) + \boldsymbol{Y}(i,j) \bullet [\boldsymbol{D}(i,j+1) \oplus \boldsymbol{B}(i+1,j)] + [\boldsymbol{Y}(i,j) \oplus 128] \bullet [\boldsymbol{D}(i+1,j) \oplus \boldsymbol{B}(i,j+1)]\} \bmod 256 \quad (27)$$

After the above processing, one can get the matrix $\boldsymbol{D}$ from the matrices $\boldsymbol{B}$ and $\boldsymbol{Y}$.

The diffusion-I process is reversible, and its inverse process recovers $\boldsymbol{B}$ from $\boldsymbol{D}$ and $\boldsymbol{Y}$ with the following steps:

Step 1. Obtain $\boldsymbol{B}(M,N)$ from $\boldsymbol{D}(M,N)$ and $\boldsymbol{Y}(M,N)$ with the following formula.

$$\boldsymbol{B}(M,N) = [\boldsymbol{D}(M,N) + 256 - \boldsymbol{Y}(M,N)] \bmod 256 \quad (28)$$

Step 2. Let j=N-1 to 1, then calculate $\boldsymbol{B}(M,j)$ for each j with the following formula.

$$\boldsymbol{B}(M,j) = [\boldsymbol{D}(M,j) + 256\times 3 - \boldsymbol{D}(M,j+1) - \mathrm{B}(\mathrm{M},\mathrm{j}+1) - \boldsymbol{Y}(M,j)] \bmod 256 \quad (29)$$

Step 3. Generate $\boldsymbol{B}(M\text{-}1,N)$ from $\boldsymbol{D}(M\text{-}1,N)$, $\boldsymbol{D}(M,1)$, $\boldsymbol{Y}(M\text{-}1,N)$ and $\boldsymbol{B}(M,1)$ with the following formula.

$$\boldsymbol{B}(M\text{-}1,N) = [\boldsymbol{D}(M\text{-}1,N) + 256\times 3 - \boldsymbol{D}(M,1) - \mathrm{B}(\mathrm{M},1) - \boldsymbol{Y}(M\text{-}1,N)] \bmod 256 \quad (30)$$

Step 4. Let $i=M$-2 to 1, then calculate $\boldsymbol{B}(i,N)$ for each i with the following formula.

$$\boldsymbol{B}(i,N)=[\boldsymbol{D}(i,N) + 256\times3 - \boldsymbol{D}(i+1,N) - \mathrm{B}(\mathrm{i}+1,\mathrm{N}) - \boldsymbol{Y}(i,N)] \bmod 256 \quad (31)$$

Step 5. Let $i=M$-1 to 1, and $j=N$-1 to 1, then calculate $\boldsymbol{B}(i,j)$ for each combination of i and j with the following formula.

$$\boldsymbol{B}(i,j)= \{\boldsymbol{D}(i,j)+256\times4-\boldsymbol{Y}(i,j)\bullet[\boldsymbol{D}(i,j+1)\oplus\boldsymbol{B}(i+1,j)]-[\boldsymbol{Y}(i,j)\oplus128]\bullet[\boldsymbol{D}(i+1,j)\oplus\boldsymbol{B}(i,j+1)]\} \bmod 256 \quad (32)$$

After the above processing, one can recover $\boldsymbol{B}$ from $\boldsymbol{D}$ and $\boldsymbol{Y}$.

(4) Diffusion-II process and its inverse

From **Fig. 1**, the diffusion-II produces the cipher image $\boldsymbol{C}$ from the matrices $\boldsymbol{D}$, $\boldsymbol{Z}$ and r_3. The concrete steps of diffusion-II are as follows.

Step 1. Obtain $\boldsymbol{C}(1,1)$ from $\boldsymbol{D}(1,1)$, $\boldsymbol{Z}(1,1)$ and r_3 with the following formula.

$$\boldsymbol{C}(1,1)=[\boldsymbol{D}(1,1)+\boldsymbol{Z}(1,1) + r_3] \bmod 256 \quad (33)$$

Step 2. Let j=2 to N, then calculate $\boldsymbol{C}(1,j)$ for each j with the following formula.

$$\boldsymbol{C}(1,j)=[\boldsymbol{D}(1,j)+\boldsymbol{D}(1,j-1)+\boldsymbol{Z}(1,j)] \bmod 256 \quad (34)$$

Step 3. Generate $\boldsymbol{C}(2,1)$ from $\boldsymbol{D}(2,1)$, $\boldsymbol{Z}(2,1)$ and $\boldsymbol{D}(1,N)$ with the following formula.

$$\boldsymbol{C}(2,1)=[\boldsymbol{D}(2,1)+\boldsymbol{D}(1,N)+\boldsymbol{Z}(2,1)] \bmod 256 \quad (35)$$

Step 4. Let i=3 to M, then calculate $\boldsymbol{C}(i,1)$ for each i with the following formula.

$$\boldsymbol{C}(i,1)=[\boldsymbol{D}(i,1)+\boldsymbol{D}(i-1,1)+\boldsymbol{Z}(i,1)] \bmod 256 \quad (36)$$

Step 5. Let i=2 to M, and j=2 to N, then calculate $\boldsymbol{C}(i,j)$ for each combination of i and j with the following formula.

$$\boldsymbol{C}(i,j)=[\boldsymbol{D}(i,j)+\boldsymbol{D}(i,j-1)+\boldsymbol{D}(i-1,j)+\boldsymbol{Z}(i,j)] \bmod 256 \quad (37)$$

After the above processing, one can get the cipher image $\boldsymbol{C}$ from $\boldsymbol{D}$, $\boldsymbol{Z}$ and r_3.

The above diffusion-II is reversible, and its inverse process recovers $\boldsymbol{D}$ from $\boldsymbol{C}$, $\boldsymbol{Z}$ and r_3. And the concrete steps of the inverse process of diffusion-II are as follows:

Step 1. Obtain $\boldsymbol{D}(1,1)$ from $\boldsymbol{C}(1,1)$, $\boldsymbol{Z}(1,1)$ and r_3 with the following formula.

$$\boldsymbol{D}(1,1)= [\boldsymbol{C}(1,1) + 256\times2 - \boldsymbol{Z}(1,1) - r_3] \bmod 256 \quad (38)$$

Step 2. Let j=2 to N, then calculate $\boldsymbol{D}(1,j)$ for each j with the following formula.

$$\boldsymbol{D}(1,j)=[\boldsymbol{C}(1,j) + 256\times2 - \boldsymbol{D}(1,j-1) - \boldsymbol{Z}(1,j)] \bmod 256 \quad (39)$$

Step 3. Produce $\boldsymbol{D}(2,1)$ from $\boldsymbol{C}(2,1)$, $\boldsymbol{Z}(2,1)$ and $\boldsymbol{D}(1,N)$ with the following formula.

$$\boldsymbol{D}(2,1)=[\boldsymbol{C}(2,1)+256\times2-\boldsymbol{D}(1,N)-\boldsymbol{Z}(2,1)] \bmod 256 \quad (40)$$

Step 4. Let i=3 to M, then calculate $\boldsymbol{D}(i,1)$ for each i with the following formula.

$$\boldsymbol{D}(i,1)=[\boldsymbol{C}(i,1) + 256\times2 - \boldsymbol{D}(i-1,1) - \boldsymbol{Z}(i,1)] \bmod 256 \quad (41)$$

Step 5. Let i=2 to M, and j=2 to N, then calculate $\boldsymbol{D}(i,j)$ for each combination of i and j with the following formula.

$$\boldsymbol{D}(i,j)=[\boldsymbol{C}(i,j) + 256\times3 - \boldsymbol{D}(i,j-1) - \boldsymbol{D}(i-1,j) - \boldsymbol{Z}(i,j)] \bmod 256 \quad (42)$$

After the above processing, the matrix $\boldsymbol{D}$ is recovered from $\boldsymbol{C}$ with the help of $\boldsymbol{Z}$.

4. Simulation results

The computer used is configured with Intel Core i7-4720HQ CPU and 8GB DDR3L memory. And C# language based on Visual Studio 2017 Community Edition is used to program the proposed algorithm. Without losing generality, the 8-bit grayscale images of Lena, Baboon, Pepper, All-black image and All-white image are used in the simulation test (where, the images of Lena, Baboon and Pepper are from the USC-SIPI Image Database which is freely available at http://sipi.usc.edu/database/). And these images are all of size 256×256 pixels, and are shown in **Figs. 4a-4e**, respectively. In the test, the secret key $\boldsymbol{K}$={255, 138, 130, 97, 14, 17, 172, 88, 194, 103, 0, 137, 234, 71, 208, 26, 26, 243, 207, 231, 15, 113, 130, 61, 252, 217, 9, 243, 21, 254, 80, 143} (in decimal format), the parameters of Chen's system (a, b, c) = (25, 3, 28), and the public key, i.e. the initial values of Chen's system, (x_0, y_0, z_0) = (-8.319, 12.0456, 36.789). Now, the proposed image encryption system is used to encrypt the plain images (as shown in **Figs. 4a-4e**, respectively), and the resultant cipher images are as shown in **Figs. 4f-4j**, respectively. Then the proposed image decryption system is used to decrypt the cipher images (as shown in **Figs. 4f-4j**, respectively), and the recovered images are as shown in **Figs. 4k-4o**, respectively. The images in **Figs. 4k-4o** are exactly the same as the original plain images of **Figs. 4a-4e**, respectively, indicating that the proposed image cryptosystem works properly.

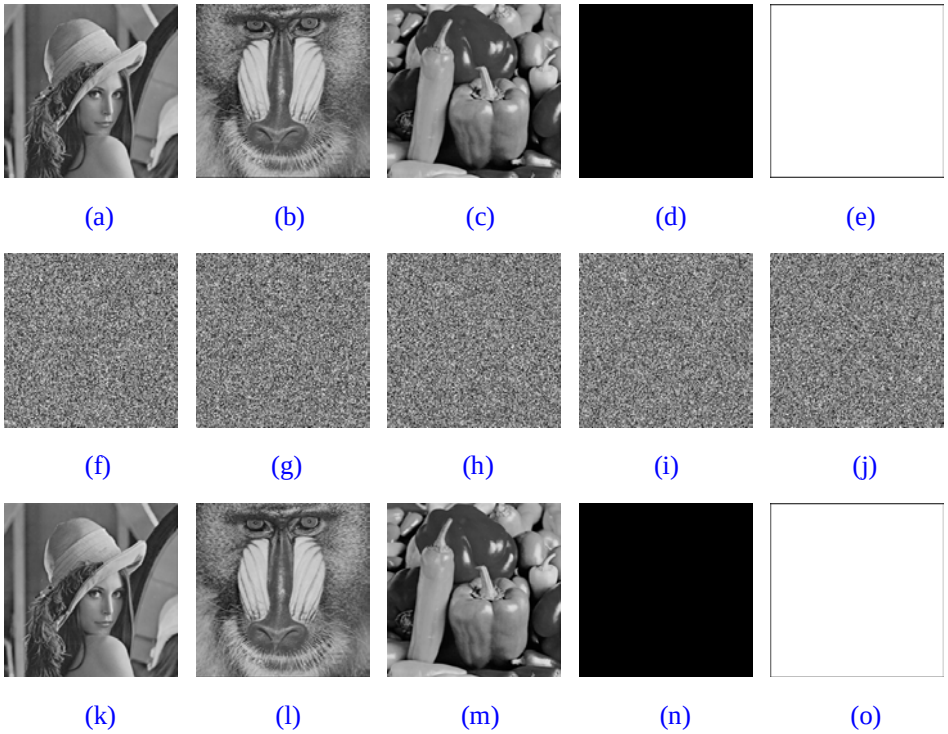


Fig. 4. Simulation results. (a)-(e) Plain images of Lena, Baboon, Pepper, All-black and All-white, respectively; (f)-(j) Cipher images of (a)-(e), respectively; (k)-(o) Recovered images of (f)-(j), respectively.

5. Security performance analysis

5.1 Key space

Currently, the information security standard issued by NIST is AES, and its key length is 128, 192 or 256 bits. And NSA (National Security Agency) claimed that AES with key length of 192 or 256 bits can be used to encrypt the top secret level documents. Referring to the key length of AES, the key length of proposed image cryptosystem is 256 bits. Therefore, the key space of proposed cryptosystem is of size 2^{256}, about 1.1579×10^{77}. Combining with the encryption/decryption speed in Section 5.4, the ability of proposed cryptosystem against the exhaustive attack will be discussed there.

In the proposed image cryptosystem, the initial values of Chen's system, i.e. x_0, y_0 and z_0, are open data and serve as the public key. Where, $x_0\in[-19.23,24.27]$, $y_0\in[-21.03,27.47]$ and $z_0\in[6.86,44.00]$. And the step size respecting x_0, y_0 and z_0 is 10^{-13}. Therefore, the size of public key space is about 7.8356×10^{43}, equivalent to the public key length being about 146 bits. The sensitivity of public key will be discussed in Section 5.3.4.

5.2 Statistical characteristics analysis

5.2.1 Histogram analysis

The histogram directly reflects the distribution of pixels of each gray value in the image. Without lose of generality, here take the plain images of **Figs. 4a-4e** and their corresponding cipher images of **Figs. 4f-4j** as examples to analyze the histogram difference between plain and cipher images. The histograms of plain images **Figs. 4a-4e** are shown in **Figs. 5a-5e**, respectively. And the histograms of cipher images **Figs. 4f-4j** are shown in **Figs. 5f-5j**, respectively. Intuitively, the histograms of plain images have obvious fluctuations, while the histograms of cipher images are fairly flat. Therefore, there is distinguishable difference between them.

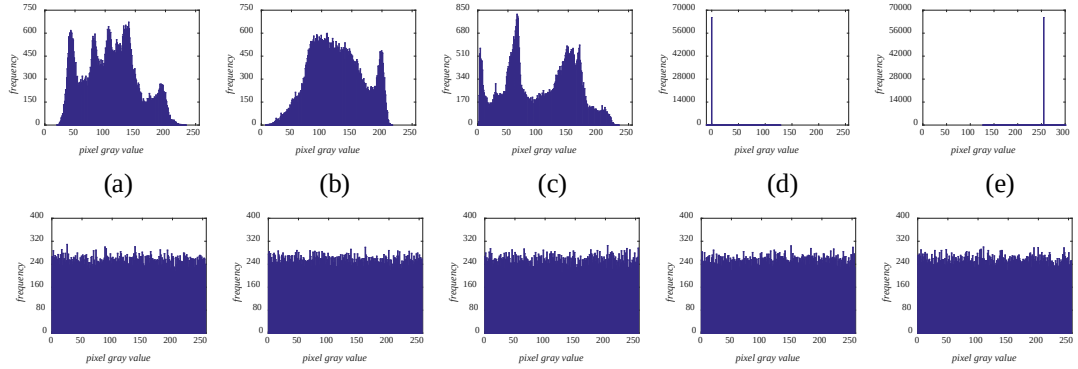


Fig. 5. Histograms. (a)-(e) Histograms of plain images **Figs. 4a-4e**, respectively; (f)-(j) Histograms of cipher images **Figs. 4f-4j**, respectively.

The quantitative analysis on the difference between the histograms of plain and cipher images is described below.

It is well known that an 8-bit $M\times N$ sized grayscale image $\boldsymbol{I}$ characterized with uniform distribution has the mean E($\boldsymbol{I}$) and variance Var($\boldsymbol{I}$) as follows.

$$\mathrm{E}(\boldsymbol{I})=\frac{1}{256}\sum_{i=0}^{255} i = 127.5 \tag{43}$$

$$\mathrm{Var}(\boldsymbol{I})=\frac{1}{256}\sum_{i=0}^{255} i^2 - 127.5^2 = 5.4613\times 10^3 \tag{44}$$

However, for a certain 8-bit grayscale image $\boldsymbol{J}$ of size $M\times N$, whose pixel values are in unknown distribution, one can calculate its mean E($\boldsymbol{J}$) and variance Var($\boldsymbol{J}$) as follows.

$$\mathrm{E}(\boldsymbol{J}) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\boldsymbol{J}(\mathrm{i},\mathrm{j}) \tag{45}$$

$$\mathrm{Var}(\boldsymbol{J}) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[\boldsymbol{J}(\mathrm{i},\mathrm{j}) - \mathrm{E}(\boldsymbol{J})]^{\mathbf{2}} \tag{46}$$

According to Eqs. (45)-(46), the mean and variance of each histogram in **Fig. 5** (i.e. the mean and variance of each image in **Figs. 4a-4j**) are calculated, and the results are listed in **Tables 3-4**.

Table 3. The mean and variance of each histogram in **Figs. 5a-5e**

	Uniform distribution	**Fig. 5a**	**Fig. 5b**	**Fig. 5c**	**Fig. 5d**	**Fig. 5e**
mean	127.5	109.5	124.5	104.2	0	255
variance	5.4613e3	2.0162e3	1.8339e3	3.2526e3	0	0

Table 4. The mean and variance of each histogram in **Figs. 5f-5j**

	Uniform distribution	**Fig. 5f**	**Fig. 5g**	**Fig. 5h**	**Fig. 5i**	**Fig. 5j**
mean	127.5	127.4	126.9	127.5	127.8	127.1
variance	5.4613e3	5.4587e3	5.4860e3	5.4768e3	5.4399e3	5.4585e3

In **Table 3**, the mean and variance of each histogram in **Figs. 5a-5e** correspond sequentially to the mean and variance of each plain image in **Figs. 4a-4e**. **Table 3** shows the differences between the plain images and uniformly distributed image in both mean and variance are significantly large, especially the minimum relative error of variance is greater than 40.44%. However in **Table 4**, the mean and variance of each histogram in **Figs. 5f-5j** correspond sequentially to the mean and variance of each plain image in **Figs. 4f-4j**. And **Table 4** shows the differences between the cipher images and uniformly distributed image in both mean and variance are fairly small, and the maximum relative error of variance is less than 0.39%. These demonstrate that the histograms of cipher images are very close to the uniform distribution, so the cipher images can resist the attack based on histogram analysis.

5.2.2 Correlation analysis

In general, plain images have a strong correlation between adjacent pixels. While the image encryption algorithm will destroy the correlation seriously to make the adjacent pixels in ciphered images completely irrelevant. The correlation coefficient is usually used to measure the correlation between adjacent pixels of images. Assume that randomly select K pairs of adjacent pixels from the image, and denote their values by (u_i,v_i), i=1,2,...,K, then the correlation coefficient r can be calculated with the following formula.

$$r = \frac{\sum_{i=1}^{K}(u_i-\mathrm{E}(\boldsymbol{u}))(v_i-\mathrm{E}(\boldsymbol{v}))}{\sqrt{\sum_{i=1}^{K}(u_i-\mathrm{E}(\boldsymbol{u}))^2\sum_{i=1}^{K}(v_i-\mathrm{E}(\boldsymbol{v}))^2}} \tag{47}$$

Where, E($\boldsymbol{u}$) and E($\boldsymbol{v}$) are separately the mean values of sequences $\{u_i\}$ and $\{v_i\}$, i=1,2,...,K. From Eq. (47), when r gets close to 1 or -1, the correlation gets strong; when r gets close to 0, the correlation gets weak; when r=0, there is no correlation between $\boldsymbol{u}$ and $\boldsymbol{v}$.

Without loss of generality, take the plain images **Figs. 4a-4e** and their corresponding cipher images **Figs. 4f-4j** as examples to discuss the correlation issue. Here, let K=2000, and then from the tested images randomly select K pairs of adjacent pixels in horizontal, vertical, diagonal and counter-diagonal directions respectively to calculate the correlation coefficients, and list the results in **Table 5**.

Table 5. Results of correlation coefficients.

Image		Horizontal	Vertical	Diagonal	Counter-diagonal
Lena	Plain (**Fig. 4a**)	0.96572	0.93161	0.91500	0.93761
	Cipher (**Fig. 4f**)	0.00070	-0.01385	0.00020	0.01186
Baboon	Plain (**Fig. 4b**)	0.84340	0.87903	0.78298	0.80757
	Cipher (**Fig. 4g**)	0.00438	-0.00261	0.00135	0.00883
Pepper	Plain (**Fig. 4c**)	0.97261	0.96512	0.92994	0.93890
	Cipher (**Fig. 4h**)	-0.01168	-0.00033	0.01072	-0.01327
All-black	Plain (**Fig. 4d**)	1.00000	1.00000	1.00000	1.00000
	Cipher (**Fig. 4i**)	-0.00592	0.03098	-0.01783	-0.04287
All-white	Plain (**Fig. 4e**)	1.00000	1.00000	1.00000	1.00000
	Cipher (**Fig. 4j**)	0.00142	0.01309	-0.00976	-0.00590

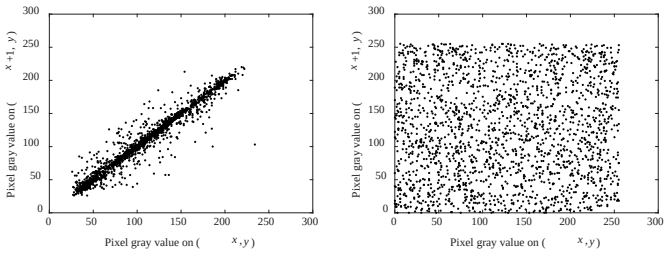
In **Table 5**, the correlation coefficient of each plain image is close to 1 (and both All-black and All-white images are linearly correlated, and their correlation coefficients are both 1), i.e. the adjacent pixels in plain images have a strong positive correlation. However, the correlation coefficient of each cipher image is close to 0, i.e. the adjacent pixels in cipher images are approximately uncorrelated. Therefore, **Table 5** shows that the proposed image encryption algorithm completely removes the correlation between adjacent pixels in the original images.

To visually compare the correlation characteristics between plain and cipher images, **Fig. 6** illustrates the correlations of **Fig. 4a** (i.e. Lena) and **Fig. 4f** (i.e. Lena's cipher image) both in horizontal direction. In **Fig. 6a**, 2000 pairs of selected adjacent pixels of Lena in horizontal direction are concentrated nearby the line of $y=x$, while, in **Fig. 6b**, 2000 pairs of selected adjacent pixels of Lena's cipher image in horizontal direction are dispersed in the whole phase portrait. Thereby, **Fig. 6** shows that the proposed image encryption algorithm effectively eliminates the correlation between adjacent pixels in the original images.

5.2.3 Information entropy analysis

The information entropy of an image reflects the uncertainty of the image information. For an 8-bit grayscale image with occurrence frequency $p(i)$ for gray value i, its information entropy H is described as follows.

$$H = -\sum_{i=0}^{255} p(i)\log_2 p(i) \tag{48}$$



(a) (b)

Fig. 6. The correlation between adjacent pixels in horizontal direction. (a) Case of Lena (**Fig. 4a**); (b) Case of Lena's cipher image (**Fig. 4f**)

Obviously, an 8-bit grayscale random image has the maximum uncertainty, and its information entropy gets to the maximum value of 8 bits. Generally, the information entropies of images with visual information are obviously less than 8 bits. Whereas the information entropies of cipher images are expected to be close to 8 bits. Here, the information entropies of the plain images Lena, Baboon, Pepper, All-black and All-white (see **Figs. 4a-4e**, respectively) and their corresponding cipher images (see **Figs. 4f-4j**, respectively) are separately calculated with Eq. (48), and the results are listed in **Table 6**.

Table 6. Information entropies of plain and cipher images (unit: bit)

	Lena	Baboon	Pepper	All-black	All-white
Plain image	7.36848	7.35572	7.56456	0	0
Cipher image	7.99734	7.99732	7.99663	7.99673	7.99735

In **Table 6**, the information entropies of Lena, Baboon and Pepper are all less than 8 bits with the minimum relative error of 5.44%, and the information entropies of All-black and All-white are both 0. Whereas the information entropies of cipher images are all close to 8 bits with the maximum relative error of 0.042%. These demonstrate that the cipher images are similar to noise-like images hiding the visual information perfectly.

5.3 Sensitivity analysis

In general, the sensitivity analysis of image cryptosystem includes three aspects, i.e. secret key sensitivity analysis, plaintext sensitivity analysis and cipher-text sensitivity analysis. Moreover, in the proposed cryptosystem, the initial values of Chen's system are regarded as public key, so the public key sensitivity analysis is also discussed in this part.

Two important indicators, known as NPCR and UACI [26], are widely used in the sensitivity analysis. NPCR (number of pixels change rate) measures the number of different pixels between two same sized images. And UACI (unified average changing intensity) measures the degree of difference between two same sized images. Assume that two $M\times N$ sized images are denoted by $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$, respectively, then NPCR and UACI can be calculated by the following formulae.

$$\mathrm{NPCR}(\boldsymbol{I}_1,\boldsymbol{I}_2)=\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}|\mathrm{Sign}(\boldsymbol{I}_1(i,j)-\boldsymbol{I}_2(i,j))|\times 100\% \tag{49}$$

Where, Sign(x) is the sign function, which returns 1 if x>0, 0 if x=0, or -1 if x<0.

$$\mathrm{UACI}(\boldsymbol{I}_1,\boldsymbol{I}_2)=\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{|\boldsymbol{I}_1(i,j)-\boldsymbol{I}_2(i,j)|}{255-0}\times 100\% \tag{50}$$

For two random images of the same size, the expected values of NPCR and UACI are 99.6094% and 33.4635%, respectively. If an image is one of the **Figs. 4a-4e** and the other is a random image of the same size, the expected values of NPCR and UACI are as shown in **Table 7**.

Table 7. The expected values of NPCR and UACI between a random image ***R*** and a plain image

Index	***R*** and Lana (**Fig. 4a**)	***R*** and Baboon (**Fig. 4b**)	***R*** and Pepper (**Fig. 4c**)	***R*** and All-black (**Fig. 4d**)	***R*** and All-white (**Fig. 4e**)
NPCR	99.6094%	99.6094%	99.6094%	99.6094%	99.6094%
UACI	28.6850%	27.9209%	30.9134%	50.0000%	50.0000%

5.3.1 Secret key sensitivity

Secret key sensitivity investigates that the influence of tiny changes of secret keys on the ciphered images in encryption system or on the recovered images in decryption system. Excellent image cryptosystem must have a good sensitivity on secret keys, i.e. the tiny changes of each secret key will lead to completely different resultant images both in

encryption and decryption systems.

The secret key sensitivity analysis includes two cases, i.e. the key sensitivity analysis of the encryption system and the key sensitivity analysis of the decryption system.

The process of key sensitivity analysis of the encryption system is as follows: Firstly, randomly generate a secret key of length 256 bits, and then randomly change its value by 1 bit to get a new key. Secondly, encrypt one plain image with these two keys to get two corresponding cipher images. Thirdly, calculate the values of NPCR and UACI between these two cipher images. Finally, repeat the test for 100 times to calculate the average values of NPCR and UACI, and then list the results in **Table 8**.

The process of key sensitivity analysis of the decryption system is as follows: Firstly, randomly generate a secret key of length 256 bits, denoted by $\boldsymbol{K}_1$. And then use $\boldsymbol{K}_1$ to encrypt a plain image $\boldsymbol{P}_1$ to get its corresponding cipher image $\boldsymbol{C}_1$. Secondly, randomly change the key $\boldsymbol{K}_1$ by 1 bit to get a new key, denoted by $\boldsymbol{K}_2$. Thirdly, use the key $\boldsymbol{K}_2$ to decrypt the cipher image $\boldsymbol{C}_1$ to get the recovered image, denoted by $\boldsymbol{P}_2$. Fourthly, calculate the values of NPCR and UACI between $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$. Finally, repeat the test for 100 times to calculate the average values of NPCR and UACI, and then list the results in **Table 9**.

Here, without loss of generality, the images in **Figs. 4a-4e** (i.e. Lena, Baboon, Pepper, All-black and All-white images, respectively) are used as the tested images, and the initial values of Chen's system (i.e. public key) are set to (x_0, y_0, z_0)=(-8.319, 12.0456, 36.789).

Table 8. Secret key sensitivity of the encryption system (unit: %).

Index	Lean	Baboon	Pepper	All-black	All-white	Theoretical
NPCR	99.6067	99.6100	99.6108	99.6081	99.6044	99.6094
UACI	33.4542	33.4541	33.4702	33.4598	33.4750	33.4635

Table 9. Secret key sensitivity of the decryption system (unit: %).

Index	Lena		Baboon		Pepper		All-black		All-white	
	Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.
NPCR	99.6097	99.6094	99.6107	99.6094	99.6087	99.6094	99.6082	99.6094	99.6071	99.6094
UACI	28.6907	28.6850	27.9199	27.9209	30.9108	30.9134	50.0086	50.0000	49.9978	50.0000

Tables 8-9 show the tested values of NPCR and UACI are both close to their theoretical values with the maximum relative error of 0.034% in both encryption and decryption systems, thereby indicating that the proposed cryptosystem possesses a strong sensitivity on secret keys.

5.3.2 Plaintext sensitivity

The plaintext sensitivity measures the degree of influence of tiny changes in plain images on the image encryption system. Excellent image encryption system must have a good plaintext

sensitivity, i.e. the tiny changes of each plain image inputted into the image encryption system will cause two completely different encrypted images.

Here, the images in **Figs. 4a-4e** (i.e. Lena, Baboon, Pepper, All-black and All-white images, respectively) are taken as examples in the plaintext sensitivity test. Firstly, make slight change for each plain image (i.e. change its one random pixel's value by 1) to obtain one pair of images with only one different pixel. Secondly, encrypt these two images with the proposed image encryption system (using the identical secret key) to obtain their corresponding cipher images. Thirdly, use these two resultant images to calculate the values of NPCR and UACI. Finally, repeat this test for 100 times to calculate the average values of NPCR and UACI, which are listed in **Table 10**. In each test, the secret key is randomly selected, and the initial values of Chen's system (i.e. public key) are set to (x_0, y_0, z_0)=(-8.319, 12.0456, 36.789).

Table 10. Results of plaintext sensitivity test (unit: %).

Index	Lean	Baboon	Pepper	All-black	All-white	Theoretical
NPCR	99.6078	99.6017	99.6092	99.6075	99.6087	99.6094
UACI	33.4476	33.4395	33.4552	33.4653	33.4598	33.4635

As can be seen in **Table 10**, the calculated results of NPCR and UACI are fairly close to their theoretical values with the maximum relative error of 0.072%, thereby showing that the proposed image encryption system has a strong plaintext sensitivity.

5.3.3 Cipher-text sensitivity

The plaintext sensitivity measures the sensitivity of image encryption system, while the cipher-text sensitivity measures the sensitivity of image decryption system. Excellent image decryption system must have a strong cipher-text sensitivity, i.e. the tiny changes in cipher images will lead to that the decrypted images completely different form the original plain images.

Here, take the images of **Figs. 4a-4e** (i.e. Lena, Baboon, Pepper, All-black and All-white images, respectively) as the tested images. Firstly, encrypt the images of **Figs. 4a-4e** with the proposed image encryption system to get their corresponding cipher images. Secondly, make slight change for each cipher image (i.e. change its one random pixel's value by 1) to obtain one pair of images with only one different pixel. Thirdly, decrypt this pair of images with the proposed image decryption system (using the same secret key as that used in the encryption system) to obtain the original plain image and one decrypted image. Finally, use these two resultant images to calculate the values of NPCR and UACI. Repeat this test for 100 times to calculate the average values of NPCR and UACI, and then list the results in **Table 11**. Note that in each test the secret key is randomly selected, and the initial values of Chen's system (i.e. public key) are set to (x_0, y_0, z_0)=(-8.319, 12.0456, 36.789).

Table 11. Results of cipher-text sensitivity test (unit: %).

Index	Lena		Baboon		Pepper		All-black		All-white	
	Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.
NPCR	99.6069	99.6094	99.6083	99.6094	99.6107	99.6094	99.6130	99.6094	99.6062	99.6094
UACI	28.6877	28.6850	27.9256	27.9209	30.9091	30.9134	50.0209	50.0000	49.9903	50.0000

In **Table 11**, the calculated values of NPCR and UACI are approximately equal to their theoretical values with the maximum relative error of 0.072%, which shows that the proposed image decryption system has a strong cipher-text sensitivity.

5.3.4 Public key sensitivity

In the proposed image cryptosystem, Chen's system is used to generate the key streams. The three initial values of Chen's system (x_0, y_0, z_0) are open and serve as the public key. The ranges of (x_0, y_0, z_0) are slightly smaller than the state space of Chen's system. Here, let $x_0 \in [-19.23,24.27]$, $y_0 \in [-21.03,27.47]$, $z_0 \in [6.86,44.00]$, and their step sizes be all 10^{-13}.

In the public key sensitivity test, randomly generate a public key (i.e. (x_0, y_0, z_0)), and change one element of the public key by 10^{-13} while keeping the other two elements unchanged to obtain the slightly changed public key. Then, use these two slightly different public keys separately as the initial values of Chen's system to generate two key streams (i.e. two sets of {$\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$, r_1, r_2, r_3}). Employ the two key streams in the encryption/decryption system to get two resultant images, and calculate the values of NPCR and UACI between the resultant images. Finally, repeat the test for 100 times to get the average values of NPCR and UACI, and list the results in **Tables 12-13**. Note that in each test a random secret key is used.

Table 12. Public key sensitivity of image encryption system (unit: %).

Index		Lean	Baboon	Pepper	All-black	All-white	Theoretical
x_0	NPCR	99.6107	99.6117	99.6095	99.6117	99.6096	99.6094
	UACI	33.4678	33.4638	33.4670	33.4538	33.4639	33.4635
y_0	NPCR	99.6085	99.6071	99.6105	99.6048	99.6100	99.6094
	UACI	33.4735	33.4506	33.4627	33.4738	33.4638	33.4635
z_0	NPCR	99.6125	99.6118	99.6100	99.6102	99.6098	99.6094
	UACI	33.4568	33.4704	33.4637	33.4534	33.4597	33.4635

Table 13. Public key sensitivity of image decryption system (unit: %).

Index		Lena		Baboon		Pepper		All-black		All-white	
		Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.	Tested	Theor.
x_0	NPCR	99.6147	99.6094	99.6068	99.6094	99.6133	99.6094	99.6122	99.6094	99.6104	99.6094
	UACI	28.6842	28.6850	27.9178	27.9209	30.8929	30.9134	50.0157	50.0000	50.0110	50.0000
y_0	NPCR	99.6022	99.6094	99.6120	99.6094	99.6051	99.6094	99.6052	99.6094	99.6077	99.6094
	UACI	28.6860	28.6850	27.9270	27.9209	30.9177	30.9134	49.9780	50.0000	49.9840	50.0000
z_0	NPCR	99.6106	99.6094	99.6075	99.6094	99.6061	99.6094	99.6120	99.6094	99.6052	99.6094
	UACI	28.6843	28.6850	27.9226	27.9209	30.9039	30.9134	49.9843	50.0000	49.9942	50.0000

Tables 12-13 reflect the degree of influence of small changes of public key on the encryption and decryption systems. In **Tables 12-13**, the calculated values of NPCR and UACI are very close to their theoretical values, i.e. the proposed image cryptosystem have a strong sensitivity on public keys. These also say that one can arbitrarily choose a public key in the public key space.

5.4 Encryption and decryption speed

The computer in Section 4 and C# program are used to test the encryption and decryption speed of proposed image cryptosystem. If the time of encrypting/decrypting an image of size $M\times N$ is denoted by t_1 or t_2 seconds respectively, then the encryption/decryption speed is $8MN/t_1$ or $8MN/t_2$ bps respectively. If the time of generating key streams for $M\times N$ sized image is denoted by t_3 seconds, then the speed of key stream generating is $8MN/t_3$ bps. Here, take the 8-bit grayscale image Lena of size 256×256 pixels as an example, and list the tested results in **Table 14**.

Table 14. Encryption/Decryption speed of proposed system.

	Key stream generator	Encryption process	Decryption process	Key stream generator + encryption process	Key stream generator + decryption process
Time (ms)	27.309	14.972	16.026	42.281	43.335
Speed (Mbps)	-	35.018	32.715	12.400	12.098

In general, once the secret key is established between two communication sides, it will be used for a period of time, so the encryption/decryption speed of proposed cryptosystem in **Table 14** is referred to 35.018 Mbps or 32.715 Mbps, respectively. While the encryption/decryption speed of proposed system containing the key stream generator is separately 12.400 Mbps or 12.098 Mbps and is equivalent to the speed of one-time pad,

which is the speed for the brute-force attacking the system once. Therefore, if the computer used in Section 4 is employed to crack the proposed encryption/decryption system by the brute-force attack method, half of the key space will be tried out and at least 7.7622×10^{67} or 7.9557×10^{67} years will be required respectively. So, the proposed system can combat the exhaustive attack.

5.5 Comparative analysis

In order to prove that the proposed image cryptosystem has excellent performance, the proposed system is compared with AES (in CBC mode) and those systems described in Refs. [27-29]. All these systems can produce the cipher images with excellent statistical characteristics, so the compared items are limited to the encryption/decryption speed, secret key sensitivity, plaintext sensitivity and cipher-text sensitivity. And the comparison results are listed in **Table 15**. Note that (1) the 8-bit grayscale image Lena of size 256×256 pixels is used; (2) the speed of each image encryption/decryption system excludes that of key stream generator; (3) the scheme of AES uses a secret key of length 256 bits [25]; (4) the scheme in [27] uses a secret key of length 256 bits; (5) the scheme in [28] uses three double-precision floating-point numbers as the secret key (equivalent to the key length of 140 bits); (6) the scheme in [29] uses four double-precision floating-point numbers as the secret key (equivalent to the key length of 186 bits); (7) the values of NPCR and UACI are the average values of 100 tests.

Table 15. Results of comparative analysis

Image cryptosystem	Enc. speed (Mbps)	Dec. speed (Mbps)	Secret key sensitivity (%)				Plaintext sensitivity (%)		Cipher-text sensitivity (%)	
			Enc. process		Dec. process					
			NPCR	UACI	NPCR	UACI	NPCR	UACI	NPCR	UACI
Theoretical	-	-	99.6094	33.4635	99.6094	28.6850	99.6094	33.4635	99.6094	28.6850
AES	7.2496	6.6223	99.6095	33.4564	99.5996	28.6531	99.6049	33.4798	99.6079	28.6853
Ref. [27]	36.587	35.223	75.7019	25.4295	86.6635	24.8754	99.5988	33.0621	0.0245	0.0001
Ref. [28]	137.862	2.967	98.2652	33.0074	98.2919	27.8255	0.0015	0.0005	0.0015	0.0004
Ref. [29]	0.068	0.068	99.6170	33.4531	99.4099	22.7641	99.5879	33.4334	99.6132	28.5904
Proposed	35.018	32.715	99.6067	33.4542	99.6097	28.6907	99.6078	33.4476	99.6069	28.6877

(Note: In **Table 15**, 'Enc.' and 'Dec.' are separately the abbreviations of 'Encryption' and 'Decryption'.)

From **Table 15**, the encryption and decryption speeds of AES are separately 7.2496 and 6.6223 Mbps. If the encryption and decryption speeds of AES are taken as two thresholds, the speed of scheme in [29] is too slow, and the decryption speed of scheme in [28] is also slow (although its encryption speed is very fast). The speeds of both proposed scheme and the scheme in [27] are faster than the thresholds so as to meet the speed requirement.

In fact, the scheme in [28] is essentially a one-time pad system using RSA technology to complete the secret key exchange for each secure communication. So the scheme in [28] has no plaintext and cipher-text sensitivities as shown in **Table 15**, and cannot resist the known/chosen plaintext attacks. From **Table 15**, the scheme in [27] has no cipher-text sensitivity, which is a defect in resisting the known/chosen cipher-text attacks. The scheme in [29] uses SHA-3 technology and possesses excellent sensitivities as shown in **Table 15**, but with very slow encryption/decryption speed, which limits its application. Like the scheme in [29], the proposed scheme also possesses excellent sensitivities.

So in the five schemes listed in **Table 15**, only the proposed scheme has the best comprehensive performance.

6. Conclusion

Unlike most of recent image cryptosystems based on confusion-diffusion architecture, this paper proposed an image cryptosystem based on substitution-diffusion architecture. The substitution algorithm "removes" the information of original image from the encrypted images, so that the useful information of plain image can be hidden better. Combining the substitution algorithm with the diffusion processes, the proposed scheme makes the changes of each pixel in the plain image affect the whole cipher image. Simulation results show that the proposed image cryptosystem has the characteristics of fast encryption/decryption speed and strong sensitivities. So the proposed system can be used as an alternative scheme for the real network communication.

The substitution technology is widely used in data cryptography. This paper used the S-Box of AES to carry out substitution operation and achieved satisfactory security results. In the proposed scheme, two diffusion processes are used to get the excellent plain/cipher-text sensitivity, which cost a lot of time. The future work will focus on studying the image scheme with new substitution algorithm and only one diffusion process, so as to further improve the encryption/decryption speed without affecting the system security.

Acknowledgement

This work was fully supported by the National Science Foundation of China (Grant Nos. 61762043, 61562035 and 61702238), the Natural Science Foundation of Jiangxi Province, China (Grant No. 20161BAB202058), and the Science and Technology Project of Education Department of Jiangxi Province, China (Grant No. GJJ160426).

References

[1] Y. Zhou, L. Bao and C. L. P. Chen, "A new 1D chaotic system for image encryption," *Signal Processing*, vol. 97, no. 7, pp. 172-182, April 2014. Article (CrossRef Link).

[2] X. Li, G. Zhang and X. Zhang, "Image encryption algorithm with compound chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 5, pp. 563-570, May 2015. Article (CrossRef Link).

[3] A. Akif, M. Irene, P. Ihsan and S. Vaidyanathan, "A new four-scroll chaotic attractor and its engineering applications," *Optik*, vol. 127, no. 13, pp. 5491-5499, July 2016. Article (CrossRef Link).

[4] E. Chen, L. Min and G. Chen, "Discrete chaotic systems with one-line equilibria and their application to image encryption," *International Journal of Bifurcation and Chaos*, vol. 27, no. 3, pp. 1750046 (17 pages), March 2017. Article (CrossRef Link).

[5] N. B. Slimane, K. Bouallegue and M. Machhout, "Designing a multi-scroll chaotic system by operating Logistic map with fractal process," *Nonlinear Dynamics*, vol. 88, no. 3, pp. 1655-1675, March 2017. Article (CrossRef Link).

[6] T. Sivakumar and R. Venkatesan, "A novel image encryption using calligraphy based scan method and random number," *KSII Transactions on Internet & Information Systems*, vol. 9, no. 6, pp. 2317-2337, June 2015. Article (CrossRef Link).

[7] J. S. A. E. Fouda, J. Y. Effa, S. L. SamratL and M. Ali, "A fast chaotic block cipher for image encryption," *Communications in Nonlinear Science & Numerical Simulation*, vol. 19, no. 3, pp. 578-588, March 2014. Article (CrossRef Link).

[8] 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, March 2015. Article (CrossRef Link).

[9] Y. Zhang, "A chaotic system based image encryption scheme with identical encryption and decryption algorithm," *Chinese Journal of Electronics*, vol. 26, no. 5, pp. 1022-1031, October 2017. Article (CrossRef Link).

[10] P. Li and Y. Zhao, "A simple encryption algorithm for quantum color image," *International Journal of Theoretical Physics*, vol. 56, no. 6, pp. 1961-1982, June 2017. Article (CrossRef Link).

[11] C. Li, "Cracking a hierarchical chaotic image encryption algorithm based on permutation," *Signal Processing*, vol. 118, no. C, pp. 203-210, January 2016. Article (CrossRef Link).

[12] J. A. Jolfaei, X. Wu and V. Muthukkumarasamy, "On the security of permutation-only image encryption schemes," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 235-246, February 2016. Article (CrossRef Link).

[13] D. S. Laiphrakpam and M. S. Khumanthem, "Cryptanalysis of symmetric key image encryption using chaotic Rossler system," *Optik*, vol. 135, no. 4, pp. 200-209, April 2017. Article (CrossRef Link).

[14] B. Norouzi and S. Mirzkuchaki, "Breaking a novel image encryption scheme based on an improper fractional order chaotic system," *Multimedia Tools & Applications*, vol. 76, no. 2, pp. 1817-1826, February 2017. Article (CrossRef Link).

[15] W. Wen, Y. Zhang, M. Su, R. Zhang, J. Chen and M. Li, "Differential attack on a hyper-chaos-based image cryptosystem with a classic bi-modular architecture," *Nonlinear Dynamics*, vol. 87, no. 1, pp. 383-390, January 2017. Article (CrossRef Link).

[16] Y. Zhang and Y. Tang, "A plaintext-related image encryption algorithm based on chaos," *Multimedia Tools and Applications*, vol. 77, no. 6, pp. 6647-6669, March 2018. Article (CrossRef Link).

[17] X. Chai, Z. Gan, K. Yang, Y. Chen and X. Liu, "An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and DNA sequence operations," *Signal Processing: Image Communication*, vol. 52, no. C, pp. 6-19, March 2017. Article (CrossRef Link).

[18] A. V. Diaconu, "Circular inter-intra pixels bit-level permutation and chaos-based image encryption," *Information Sciences*, vol. 355-356, pp. 314-327, August 2016. Article (CrossRef Link).

[19] H. Fan and M. Li, "Cryptanalysis and improvement of chaos-based image encryption scheme with circular inter-intra-pixels bit-level permutation," *Mathematical Problems in Engineering*, vol. 2017, pp. 1-11, July 2017. Article (CrossRef Link).

[20] W. Zhang, H. Yu, Y. Zhao and Z. Zhu, "Image encryption based on three-dimensional bit matrix permutation," *Signal Processing*, vol. 118, no. 1, pp. 36-50, January 2016. Article (CrossRef Link).

[21] J. Wu, X. Liao and B. Yang, "Cryptanalysis and enhancements of image encryption based on three-dimensional bit matrix permutation," *Signal Processing*, vol. 142, no. 1, pp. 292-300, January 2018. Article (CrossRef Link).

[22] X. Wang and H. Zhang, "A novel image encryption algorithm based on genetic recombination and hyper-chaotic systems," *Nonlinear Dynamics*, vol. 83, no. 1-2, pp. 333-346, February 2016. Article (CrossRef Link).

[23] T. Ueta and G. Chen, "Bifurcation analysis of Chen's equation," *International Journal of Bifurcation & Chaos*, vol. 10, no. 8, pp. 1917-1931, August 2000. Article (CrossRef Link).

[24] National Institute of Standards and Technology, "Federal Information Processing Standard FIPS PUB 140-2, Security requirements for cryptographic modules," 2002. Article (CrossRef Link)

[25] J. Daemen and V. Rijmen, "The design of Rijndael: AES - the Advanced Encryption Standard," Springer-Verlag, Berlin, 2002.

[26] 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, July 2004. Article (CrossRef Link).

[27] Z. Hua, Y. Zhou, C. Pun and C. L. P. Chen, "2D Sine Logistic modulation map for image encryption," *Information Sciences*, vol. 297, no. C, pp. 80-94, March 2015. Article (CrossRef Link).

[28] Ünal Çavuşoğlu, S. Kaçar, I. Pehlivan I and A. Zengin, "Secure image encryption algorithm design using a novel chaos based S-Box," *Chaos Solitons & Fractals*, vol. 95, no. 2, pp. 92-101, February 2017. Article (CrossRef Link).

[29] G. Ye, H. Zhao and H. Chai, "Chaotic image encryption algorithm using wave-line permutation and block diffusion," *Nonlinear Dynamics*, vol. 83, no. 4, pp. 2067-2077, April 2016. Article (CrossRef Link).

Yong Zhang received the MS degree in communication and information systems and the PhD degree in circuits and systems both from University of Electronic Science and Technology of China (UESTC), in 2003 and in 2006 respectively. Currently, he is an associate professor at School of Software and Communication Engineering in Jiangxi University of Finance and Economics (JXUFE) in China. His research interests focus on the area of information security and quantum information.
E-mail: zhangyong@jxufe.edu.cn

Xiaoyang Jia received the Bachelor's degree in detection guidance and control technology and the MS degree in optical engineering both from Changchun University of Science and Technology (CUST) in China, in 2014 and in 2017 respectively. She serves as an associate lecturer in Changchun China Optical Science & Technology Museum. Her research interests focus on the area of information security and space laser communication.
E-mail: jiaxiaoyang_cc@126.com