

A plaintext-related image encryption algorithm based on chaos

Yong Zhang¹  · Yingjun Tang¹

Received: 28 August 2016 / Revised: 30 December 2016 / Accepted: 6 March 2017 /

Published online: 12 April 2017

© Springer Science+Business Media New York 2017

Abstract A symmetric key image cryptosystem based on the piecewise linear map is presented in this paper. In this cryptosystem, the encryption process and the decryption process are exactly same. They both include the same operations of plaintext-related scrambling once, diffusion twice and matrix rotating of 180 degrees four times. The length of secret key in the system is $64d$ where d is a positive integer. The proposed system can fight against the chosen/known plaintext attacks due to the using of plaintext-related scrambling. The simulate results and comparison analysis show that the proposed system has many merits such as high encryption/decryption speed, large key space, strong key sensitivity, strong plaintext sensitivity, strong cipher-text sensitivity, good statistical properties of cipher images, and large cipher-text information entropy. So the proposed system can be applied to actual communications.

Keywords Image encryption · Piecewise linear chaotic map · Plaintext-related scrambling · Identical encryption and decryption algorithm · Security analysis

1 Introduction

With the rapid development of communication technology, the security of image information has drawn increasingly extensive attention, and the image encryption is the most effective means to ensure the confidential image transmission in public information channel. However, the classic text-based encryption methods, such as DES and AES, are no longer suitable for image encryption because the image possesses the characteristics of huge volume, high data redundancy and strong adjacent correlation. Meantime, the chaotic systems firstly discovered in the 1970s

✉ Yong Zhang
zhangyong@jxufe.edu.cn; zhnyong@21cn.com

Yingjun Tang
Yingjun.T@gmail.com; 07112047@bjtu.edu.cn

¹ School of Software and Communication Engineering, Jiangxi University of Finance and Economics, Nanchang, People's Republic of China

by E. N. Lorenz have the intrinsic properties of sensitivity to initial conditions and parameters, non-periodicity and ergodicity, and these properties correspond in a way to the properties of image encryption system such as secret key sensitivity, plaintext/cipher-text sensitivity, cipher-text balance, confusion and so on. Hence, chaotic systems are widely employed in digital image encryption systems to generate secret code streams for encryption [2, 5, 9, 10, 12, 16, 17, 25, 30].

Some of the existing chaotic-system-based image encryption systems use chaotic systems to produce pseudo-random numbers with excellent statistical properties to be used as the secret code streams, but their image encryption algorithms [11, 14, 15, 20, 32, 34] essentially employ the same secret code streams to encrypt the different plain images. Therefore, these image encryption systems cannot resist the chosen/known plaintext attacks [4, 19, 22, 26, 29, 31]. However, as a frequently used attack method, chosen/known plaintext attacks can be used by eavesdroppers to crack some not well-designed image encryption systems in a short time. In order to secure plain images against these types of attack methods, the scholars suggested the plaintext-related image encryption algorithms [1, 24, 27, 28, 33, 30].

The main features of the plaintext-related image encryption algorithm are that the plaintext information is directly or indirectly employed to change the secret code streams during the encryption process, which makes different plain images encrypted with different secret code streams even when their secret key or equivalent keys are same. In [1], an image encryption algorithm named *BES-w/r/b* was proposed, where *BES* means a block encryption scheme, *w* represents the number of bits per pixel, *r* represents the number of rounds, and *b* represents the length of secret key (in bytes). This scheme used the initial secret key and the extension of previous cipher block to produce the secret key of encrypting the current image block, and to make the diffusion plaintext-related. However, this scheme is poor on security when $r = 1$ while slow on encryption speed when $r \geq 2$.

In [24], the generalized Logistic map is used to produce three groups of secret code streams for encrypting. In order to fight against the chosen plaintext attack, the value of current pixel in plain image is positively diffused then oppositely diffused to influence the encryption of next pixel in the plain image. In [33], an image encryption scheme based on feedback method was proposed, which used the plaintext information as feedback in the confusion process so that different plain images have different confusion results. In [27], random pixels in the intermediate cipher images generated in the first round were selected to produce the new secret key (named by the second level key), and in this manner, the scheme indirectly realized the plaintext-related image encryption. But the encryption speed of the scheme is relatively slow.

In 2016, we proposed a method which added plaintext information in the confusion process, and the structure of the encryption scheme is ‘diffusion - plaintext related confusion- diffusion’. Its encryption speed is much faster than those schemes with plaintext related diffusion algorithms [28, 30]. However, this scheme requires three secret code matrices with the same size as the plain image. Due to the huge volume of image data, the scheme has high performance requirements for pseudo-random number generator.

In the foresaid plaintext-related image encryption schemes based on chaotic systems, the encryption algorithms are different from the decryption algorithms, and the latter are the reverse of the former. So the communication parties both need two sets of equipment, for encryption and decryption algorithms respectively. This paper attempts to study an image cryptosystem whose encryption and decryption algorithms are the same, which makes the communication parties possess only one set of equipment and one algorithm to achieve encryption and decryption. Meanwhile, the proposed image cryptosystem needs only one secret code matrix and adopt the plaintext-related scrambling algorithm to resist the chosen/known plaintext attacks effectively.

The remains of the paper are organized as follows: Section 2 details the structure and algorithms of the proposed image encryption scheme; Section 3 presents some simulation results of encryption and decryption applications; Section 4 analyzes the security performance of the scheme from various aspects, such as encryption/decryption speed, key space, statistical properties of the cipher-text, key sensitivity, plaintext sensitivity, cipher-text sensitivity, information entropy and so on; Section 5 summarizes the paper.

2 Image encryption scheme

2.1 Used chaotic system

The piecewise linear chaotic map (PWLCM) is used in this paper, and its equation is as follows [3]:

$$x_{i+1} = F(x_i, p) = \begin{cases} \frac{x_i}{p}, & 0 \leq x_i < p \\ \frac{x_i - p}{0.5 - p}, & p \leq x_i < 0.5 \\ F(1 - x_i, p), & 0.5 \leq x_i < 1 \end{cases} \quad (1)$$

Where, $x_i \in (0, 1)$, and $p \in (0, 0.5)$.

In [13], the chaotic characteristics of PWLCM were studied thoroughly, and also the PWLCM used to generate pseudo-random sequences with good statistical properties was verified. The PWLCM is widely used in variety of image encryption systems [8, 21].

2.2 Typical image cryptosystem

The typical image encryption system based on chaotic system includes both confusion module and diffusion module, and it converts plain image into noise-like cipher image through executing the two modules for multiple rounds, as shown in Fig. 1. In this image encryption system, chaotic system is used to generate the pseudo-random secret code stream for encryption. The number of rounds is not less than 2. And the decryption process is different from the encryption process, it is the reverse of the encryption process [5, 17].

On the basis of traditional chaotic image cryptosystem, we try to explore a new image cryptosystem, in which the pseudo-random secret code streams are still produced by the chaotic system, but the encryption process and the decryption process are exactly the same, and no round operation is needed in the two processes. Also, it uses the plaintext-related confusion to frustrate the chosen/known plaintext attacks.

2.3 Proposed encryption scheme

The proposed image encryption scheme is as shown in Fig. 2. In the proposed scheme, the encryption process and the decryption process are exactly the same. They both consist of the operations of secret code streams generation, forward diffusions once, plaintext-related scrambling once, backward diffusion once and matrix rotating 180 degrees for four times. The algorithms will be discussed in detail in the following subsections.

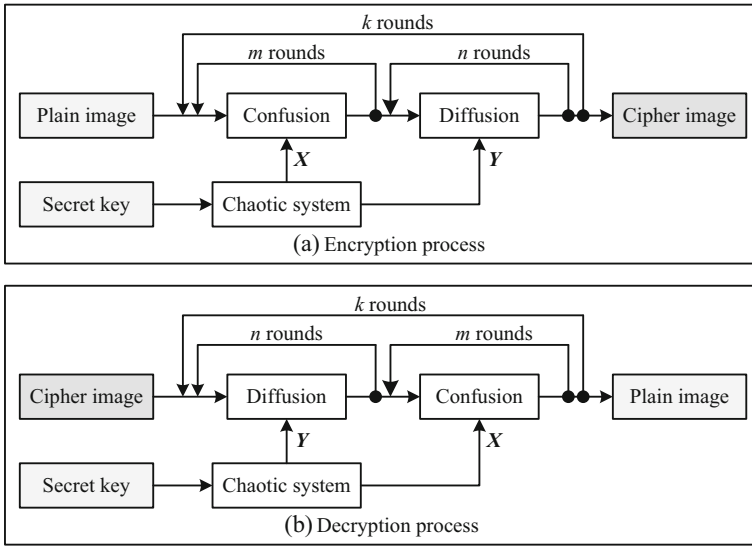


Fig. 1 Typical image cryptosystem

Assume that the plain image is denoted by P with size of $M \times N$, where M and N are the height and width of image, respectively. Note that N must be an even number in the proposed scheme, i.e. $N \bmod 2 = 0$. Otherwise, a zero-column vector of length M need to be added to the image P as its last column to make P a matrix of size $M \times (N + 1)$.

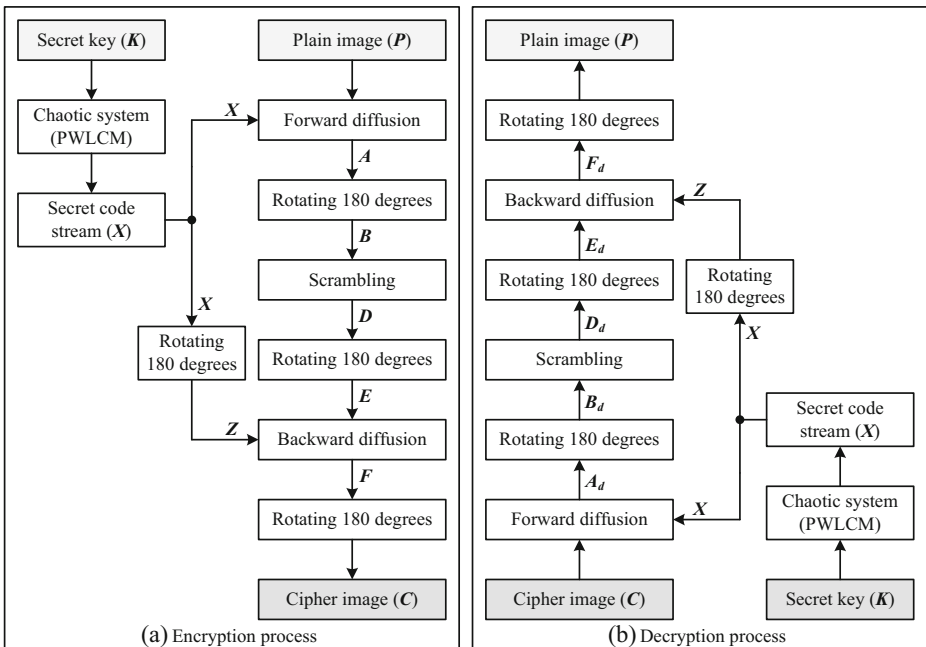


Fig. 2 Proposed encryption scheme

2.3.1 Secret code stream generating algorithm

The secret key of the proposed scheme is denoted by \mathbf{K} with length of $64 \times d$, where, d is a natural number. And the secret key \mathbf{K} can be represented in hexadecimal format as ‘ $k_1k_2k_3k_4k_5\dots k_{16d-3}k_{16d-2}k_{16d-1}k_{16d}$ ’, where, each k_i , $i = 1, 2, \dots, 16d$, is a decimal integer in range of 0 to 15.

Generating the initial value x_0 and the parameter p of PWLCM with the secret key \mathbf{K} using the following algorithms:

Step 1. Fetch ‘ $k_1k_2k_3k_4k_5k_6k_7k_8$ ’ from the secret key \mathbf{K} , and calculate the value of x_{00} according to the following formula:

$$x_{00} = 0.8 \times \sum_{i=1}^8 \frac{k_i}{2^{4i}} + 0.1 \quad (2)$$

Here, the calculated x_{00} is in the range (0.1,0.9).

Step 2. Fetch ‘ $k_9k_{10}k_{11}k_{12}k_{13}k_{14}k_{15}k_{16}$ ’ from the secret key \mathbf{K} , and calculate the value of p_0 with the following formula:

$$p_0 = 0.4 \times \sum_{i=1}^8 \frac{k_{8+i}}{2^{4i}} + 0.1 \quad (3)$$

Here, the calculated p_0 is in the range(0.1,0.5).

Step 3. Substitute x_{00} and p_0 into Eq. (1) as its initial value and parameter, respectively. Then iterate Eq. (1) for 64 times to get the state variable denoted by x_{64} . Renew x_{00} with x_{64} , i.e. new $x_{00} = x_{64}$. If $d = 1$, then let $x_0 = x_{00}$, $p = p_0$, and end.

Step 4. If $d > 1$, then execute Steps. 5–9 in a loop for $d-1$ times. Denote the loop variable by j , and let $j = 1$.

Step 5. If $j > d-1$, then jump to Step 10; Else, continue to Step 6.

Step 6. Fetch ‘ $k_{16j+1}k_{16j+2}k_{16j+3}k_{16j+4}k_{16j+5}k_{16j+6}k_{16j+7}k_{16j+8}$ ’ from \mathbf{K} , and assign them to ‘ $k_1k_2k_3k_4k_5k_6k_7k_8$ ’. And use Eq. (2) to calculate a value named by x_{0j} . Then renew the value x_{0j} with the following formula:

$$x_{0j} = 0.382 \times x_{0j} + 0.618 \times x_{0,j-1} \quad (4)$$

Step 7. Fetch ‘ $k_{16j+9}k_{16j+10}k_{16j+11}k_{16j+12}k_{16j+13}k_{16j+14}k_{16j+15}k_{16j+16}$ ’ from \mathbf{K} and assign them to ‘ $k_9k_{10}k_{11}k_{12}k_{13}k_{14}k_{15}k_{16}$ ’. And use Eq. (3) to calculate a value named by p_j . Then update the value of p_j with the following formula:

$$p_j = 0.382 \times p_j + 0.618 \times p_{j-1} \quad (5)$$

Step 8. Substitute x_{0j} and p_j into Eq. (1) as the initial value and parameter, respectively. Iterate Eq. (1) for 64 times to get the new state variable denoted by x_{64} . Then renew x_{0j} with x_{64} , i.e. new $x_{0j} = x_{64}$.

Step 9. $j = j + 1$ and go to Step 5.

Step 10. Let $x_0 = x_{0,j-1}$, and $p = p_{j-1}$. And x_0 and p are the initial value and parameter of Eq. (1), respectively.

In the above algorithm, the bigger the value of d is, the longer the length of the secret key \mathbf{K} is. For example, when $d = 4$, the length of \mathbf{K} is 256 bits; and when $d = 8$, the length of \mathbf{K} is 512 bits. Consequently, the longer the secret key is, the longer the time consumed for generating x_0 and p is. Now, substitute the obtained x_0 and p into Eq. (1) and iterate Eq. (1) for $M \times N$ times to get a float matrix named by \mathbf{Y} with size of $M \times N$. Then generate the secret code matrix \mathbf{X} from \mathbf{Y} by converting each element of $\mathbf{Y}(i,j)$ into $\mathbf{X}(i,j)$ using the following formula:

$$\mathbf{X}(i,j) = \text{floor}(\mathbf{Y}(i,j) \times 10^{14}) \bmod 256, i = 1, 2, \dots, M, j = 1, 2, \dots, N \quad (6)$$

Rotate matrix \mathbf{X} by 180 degrees to generate another matrix named by \mathbf{Z} .

2.3.2 Forward diffusion

The forward diffusion operation is used to convert plain image \mathbf{P} into a new matrix named by \mathbf{A} , with the secret code matrix \mathbf{X} . The steps of this operation are as follows:

- Step 1. Let $i = 1, j = 1$.
- Step 2. Let $\mathbf{A}(i,j) = \mathbf{P}(i,j) + \mathbf{X}(i,j) \pmod{256}$.
- Step 3. $j = j + 1$. If $j > N$, then go to Step 5; Else, continue to Step 4.
- Step 4. $\mathbf{A}(i,j) = \mathbf{P}(i,j) + \mathbf{A}(i,j-1) + \mathbf{X}(i,j) \pmod{256}$, then go to Step 3.
- Step 5. Let $j = 1, i = i + 1$.
- Step 6. $\mathbf{A}(i,j) = \mathbf{P}(i,j) + \mathbf{A}(i-1,j) + \mathbf{A}(i-1,N) + \mathbf{X}(i,j) \pmod{256}$.
- Step 7. $j = j + 1$. If $j > N$, then go to Step 5; otherwise, continue to Step 8.
- Step 8. $\mathbf{A}(i,j) = \mathbf{P}(i,j) + \mathbf{A}(i-1,j) + \mathbf{A}(i,j-1) + \mathbf{X}(i,j) \pmod{256}$.
- Step 9. If $i = M$ and $j = N$, then stop; otherwise, go to Step 7.

The obtained matrix \mathbf{A} is rotated by 180 degrees to get an image matrix named by \mathbf{B} , which is the input of the plaintext-related scrambling. The scrambling operation will be discussed in the next subsection.

2.3.3 Plaintext-related scrambling

The plaintext-related scrambling operation is used to transform matrix \mathbf{B} into a new matrix named by \mathbf{D} . The steps of this operation are as follows:

- Step 1. For each coordinate point (i,j) of matrix \mathbf{B} , $i = 1, 2, \dots, M, j = 1, 2, \dots, N$, calculate the sum of the i -th row (except $\mathbf{B}(i,j)$) and then calculate the sum of the j -th column (except $\mathbf{B}(i,j)$), denoted by \mathbf{R}_i and \mathbf{H}_j , respectively. After that, calculate a new coordinate point (m,n) with the following methods:

If $j \bmod 2 = 1$, then $m = (\mathbf{H}_j \bmod M) + 1, n = (\mathbf{R}_i \bmod N) + 1$.

If $j \bmod 2 = 0$, then $m = M - (\mathbf{H}_j \bmod M), n = N - (\mathbf{R}_i \bmod N)$.

If the calculated $m = i$ or $n = j$, then do nothing; otherwise, swap $\mathbf{B}(i,j)$ and $\mathbf{B}(m,n)$.

Step 2. Scramble matrix B by looping the Step 1 for each pixel in B from the left to right and then top to bottom, to get a matrix named by D .

The above steps are shown in Fig. 3. Matrix E is generated by rotating matrix D by 180 degrees, and it will be used in backward diffusion described in the next subsection.

2.3.4 Backward diffusion

The backward diffusion operation is used to convert the matrix E into a new matrix named by F , with the help of pseudo-random secret code matrix Z . The steps of this operation are as follows:

- Step 1. Let $i = 1, j = 1$.
- Step 2. Calculate $F(i,j) = (256 \times 3 + E(i,j) - E(i,j + 1) - E(i + 1,j) - Z(i,j)) \pmod{256}$.
- Step 3. Let $j = j + 1$. If $j = N$, then continue to Step 4; otherwise, go to Step 2.
- Step 4. Calculate $F(i,j) = (256 \times 3 + E(i,j) - E(i + 1,N) - E(i + 1,1) - Z(i,j)) \pmod{256}$.
- Step 5. Let $j = 1, i = i + 1$. If $i = M$, then continue to Step 6; otherwise, go to Step 2.
- Step 6. Calculate $F(i,j) = (256 \times 2 + E(i,j) - E(M,j + 1) - Z(i,j)) \pmod{256}$.
- Step 7. $j = j + 1$. If $j = N$, continue to Step 8; otherwise, go to Step 6.
- Step 8. Calculate $F(i,j) = (256 + E(i,j) - Z(i,j)) \pmod{256}$.
- Step 9. End

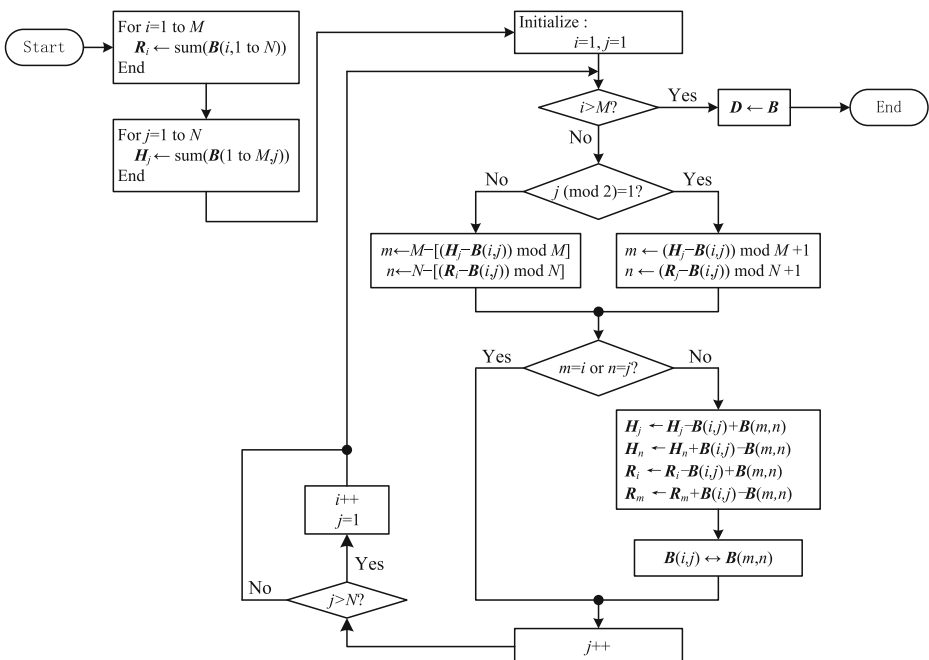


Fig. 3 Flowchart of plaintext-related scrambling

Rotating the obtained matrix F by 180 degrees, we get a matrix named by C , and it is the expected cipher image.

3 Simulation results

In our proposed scheme, the encryption algorithm and the decryption algorithm share the same function. Based on C language, we programmed the encryption/decryption function ' $[I_2] = \text{EncDecAlg}(K, I_1)$ '. For the encryption process, the inputs of the function 'EncDecAlg' are the secret key K and the plain image P , while the output is the cipher image C , i.e. $C = \text{EncDecAlg}(K, P)$; while for the decryption process, the inputs are K and C , while the output is the recovered image P , i.e. $P = \text{EncDecAlg}(K, C)$.

The computer used is configured with Intel Core I7-6700 k@4.00GHz CPU, 8GB DDR4 Memory, Windows 10 (64-bit) and Eclipse C/C++ with MinGW GCC Build Tool Chain. Without loss of generality, with the help of encryption/decryption function 'EncDecAlg' and the secret key of length 256-bit ($d = 4$) $K = \text{'FEDCBA98765432100123456789ABCDEF02468ACE13579BDFF0E1D2C3B4A59687'}$ (in hexadecimal), we encrypted the plain images Lena, Baboon, Pepper and Plane (all of size 353×398 , as shown in Fig. 4a–d, respectively) to obtain the cipher images as shown in Fig. 4e–h, respectively. Then we used the same function 'EncDecAlg' and the same key K to decrypt the Fig. 4e–h, and the recovered images are as shown in Fig. 4i–l. The histograms of the plain images Lena, Baboon, Pepper and Plane are as shown in Fig. 4m–p, respectively. The histograms of the cipher images (as shown in Fig. 4e–h respectively) are as shown in Fig. 4q–t, respectively.

As can be seen from Fig. 4, (i) the proposed scheme encrypted the plain images (as shown in Fig. 4a–d, respectively) into the noise-like cipher images (as shown in Fig. 4e–h, respectively) with no visual information leakage; (ii) the decrypted images (as shown in Fig. 4i–l, respectively) are identical to the original plain images (as shown in Fig. 4a–d, respectively); (iii) the histograms of cipher images (as shown in Fig. 4q–t, respectively) are fairly flat and largely different from those of plain images (as shown in Fig. 4m–p, respectively).

With the key $K = \text{'4E176D626DDBA9D65247F262F49FFF9BAF3A518EE31D4B89FE955 D51C7B52B61'}$ (in hexadecimal), we encrypted all-0 s image and all-255 s image (both of size 353×398 , as shown in Fig. 5a–b, respectively) into cipher images (as shown in Fig. 5c–d, respectively). The histograms of cipher images are as shown in Fig. 5e–f, respectively. Meanwhile, when the all-black image is used as P in Fig. 2a, the intermediate cipher images A, B, D, E and F are as shown in Fig. 6a–e, respectively; when the image of Fig. 5c is used as the cipher image C in Fig. 2b, the decrypted intermediate images A_d, B_d, D_d, E_d and F_d are as shown in Fig. 6f–j, respectively. Where, $A_d = D$, and $E_d = B$. This indicates that one forward diffusion is equivalent to the combination of one 180-degree matrix rotation, one backward diffusion and another 180-degree matrix rotation, and the combination of one 180-degree matrix rotation and the scrambling operation is equivalent to the combination of the same scrambling operation and one 180-degree matrix rotation.

It can be seen from Fig. 5 that the proposed cryptosystem can encrypt the all-0 s image and all-255 s image into noise-like images without any visual information leakage, and the histograms of cipher images are fairly flat.

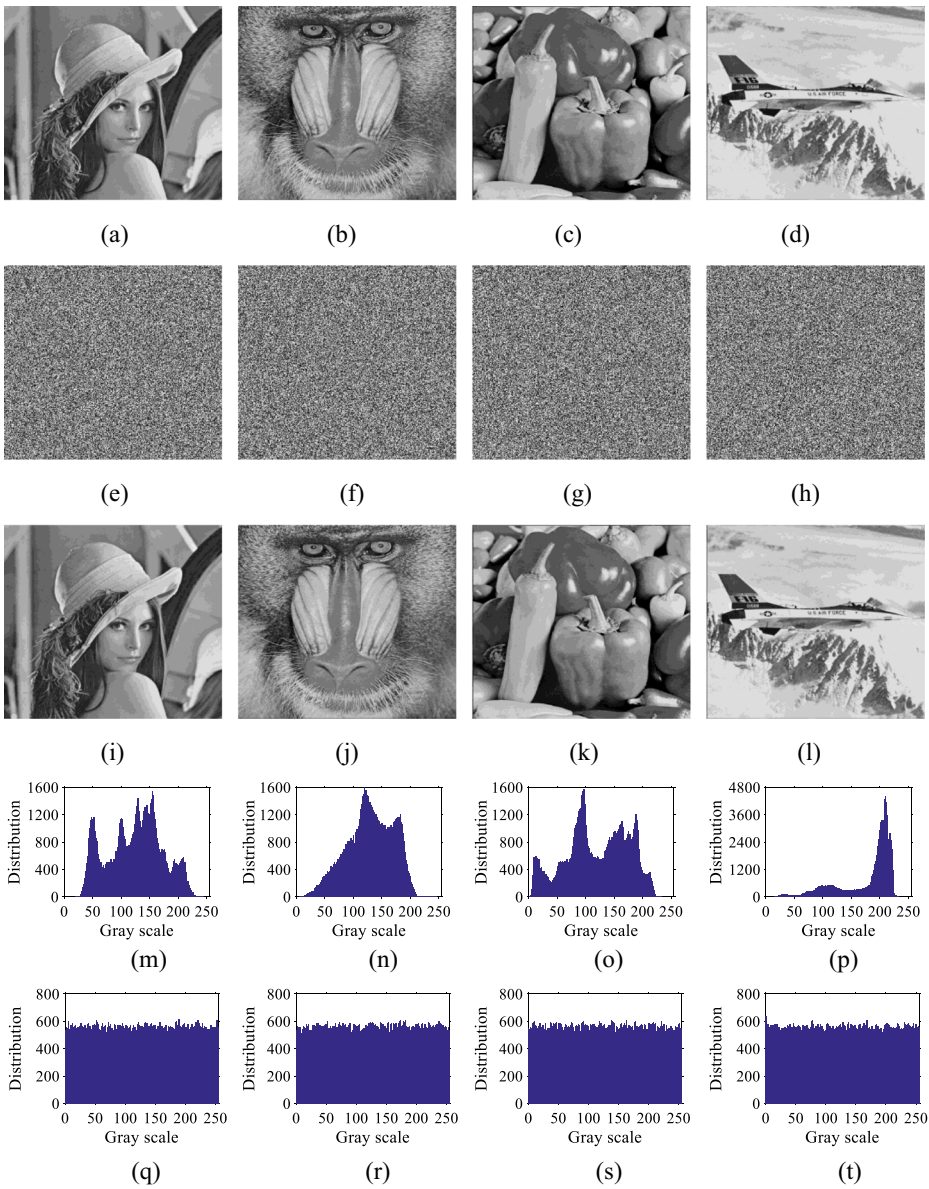


Fig. 4 Simulation results-I. **a–d** Plain images of Lena, Baboon, Pepper and Plane, respectively; **e–h** Cipher images of (**a–d**), respectively; **i–l** Recovered images of (**e–h**), respectively; **(m–p)** Histograms of (**a–d**), respectively; **q–t** Histograms of (**e–h**), respectively

4 Security analysis

This section will discuss the security performance of the proposed scheme from various aspects, such as encryption/decryption speed, key space, statistical properties of the cipher-text, key sensitivity, plaintext sensitivity, cipher-text sensitivity, information entropy, and so on.

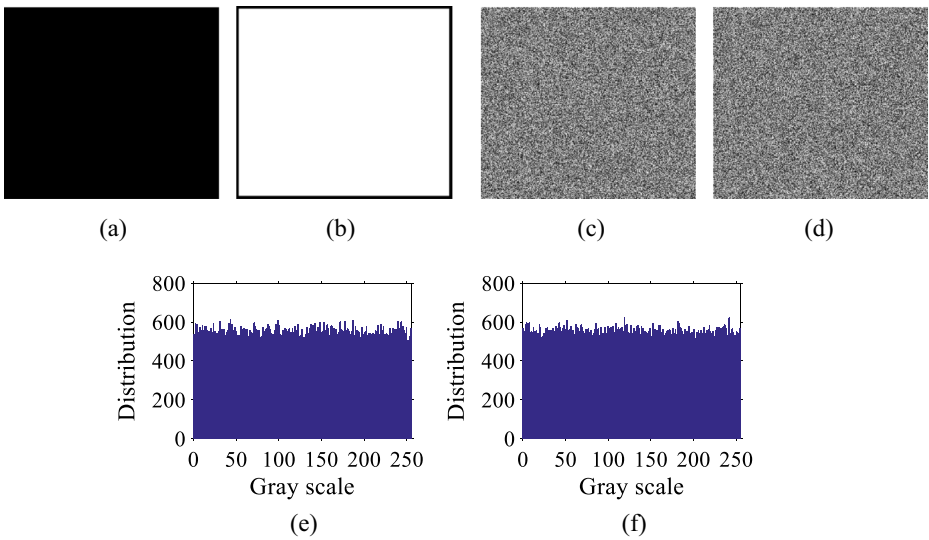


Fig. 5 Simulation results-II. **a** All-0 s image; **b** All-255 s image; **c** Cipher image of (a); **d** Cipher image of (b); **e–f** Histograms of (c–d), respectively

4.1 Encryption and decryption speed

In the proposed scheme, the encryption and decryption processes share the same function, so the encryption speed and the decryption speed are the same. The encryption/decryption speed is related to the length of secret key and the size of image. The larger the image size and the longer the secret key length are, the longer the encryption/decryption time is. Table 1 lists the encryption/decryption time under the condition of different secret key lengths and different image sizes, where the length of the secret key is $64d$ bits, $d = 1, 2, 4, 8, 16,$ and 32 , the size of image is $128 \times 128, 256 \times 256, 353 \times 398, 384 \times 384$ and 512×512 , and the unit of encryption/decryption time is second.

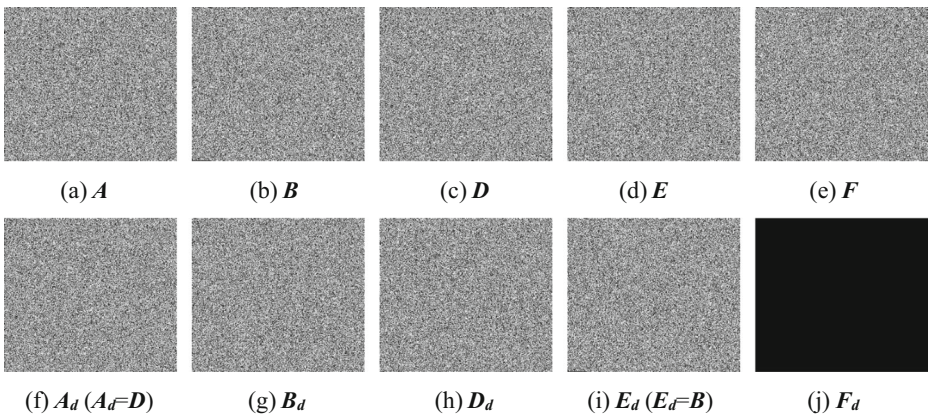


Fig. 6 The intermediate cipher images

Table 1 Encryption/decryption time for different image sizes and secret key lengths (unit:s)

Image size	$d = 1$	$d = 2$	$d = 4$	$d = 8$	$d = 16$	$d = 32$
128×128	0.0020	0.0020	0.0020	0.0020	0.0020	0.0020
256×256	0.0070	0.0070	0.0070	0.0070	0.0070	0.0070
353×398	0.0140	0.0140	0.0140	0.0140	0.0150	0.0150
384×384	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
512×512	0.0270	0.0270	0.0270	0.0270	0.0270	0.0270

In Eclipse C/C++ program, the minimum time resolution of the computer is 1 millisecond

As can be seen from Table 1, for the same image, the encryption/decryption time with key length of 2048 bits ($d = 32$) is almost equal to the encryption/decryption time with key length of 64 bits ($d = 1$). These demonstrate that the length of secret key imposes fairly small impact on the encryption/decryption time. Therefore, the proposed scheme can use longer secret key to enhance the security.

Take the key length of 256 bits ($d = 4$) as an example, the relationship between the image size and the encryption/decryption time is as shown in Fig. 7.

It can be seen from Fig. 7 that the encryption/decryption time is approximately linear with the image size by the fitting curve of $y = (1.0149 \times 10^{-7})x + 2.7907 \times 10^{-4}$. When the image size is $x = 353 \times 398$, the calculated encryption/decryption time is about $y = 0.0145$ s which is close to value 0.0140 s in Table 1, and their relative error is about 3.45%.

4.2 Key space

With the development of computer, it's required for the security system that the length of secret key is at least 128 bits [18]. In the proposed scheme, the length of secret key is $64d$ bits, where d is a positive integer. As for the image of size 256×256 , according to the encryption/decryption time as shown in Table 1, the consumed time for cracking the system with the brute-force attack method is listed in Table 2 while the used computer is the same as mentioned in Section 3. The unit of time is year in Table 2.

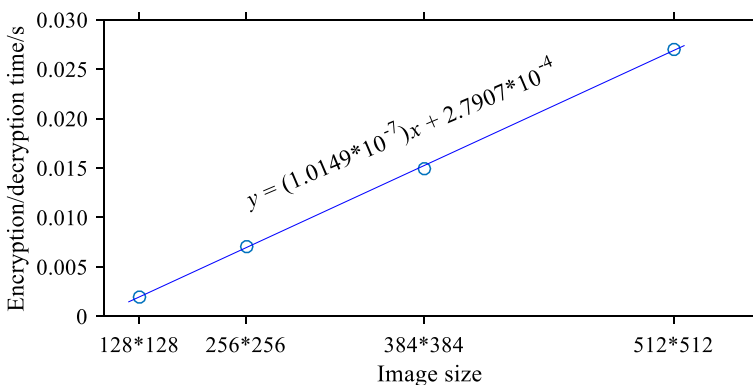


Fig. 7 Relationship between the encryption/decryption time and the image size

Table 2 Consumed time for cracking the proposed encryption system

	$d = 1$	$d = 2$	$d = 4$	$d = 8$	$d = 16$	$d = 32$
Time (year)	4.0946×10^9	7.5532×10^{28}	2.5702×10^{67}	2.9761×10^{144}	3.9903×10^{298}	7.1734×10^{606}

As can be seen from Table 2, exhaustively attacking the image of size 256×256 in the proposed scheme, the consumed time is about 7.5532×10^{28} years for the key length of 128 bits ($d = 2$) and about 2.5702×10^{67} years for the key length of 256 bits ($d = 4$). In fact when the key length is growing to 2048 bits ($d = 32$), the exhaustive attack time is about 7.1734×10^{606} years. And for most applications, key of 128 bits ($d = 2$) or 256 bits ($d = 4$) is enough, and then the size of key space is 2^{128} or 2^{256} .

4.3 Statistical characters of cipher images

Without loss of generality, we consider the statistical properties of cipher images as shown in Fig. 4e–h. These cipher images have extremely flat histograms as shown in Fig. 4q–t, indicating that the occurrence probability of each gray scale value is approximately equal, which is the reason that the cipher image is noise-like. The relationship between adjacent pixels in the cipher images will be discussed in the following.

Select N pairs of adjacent pixels randomly from the image, and denote the values of the i -th pair by (x_i, y_i) , $i = 1, 2, \dots, N$ (Labeling $\mathbf{x} = \{x_i\}$, $\mathbf{y} = \{y_i\}$). Then the correlation coefficient r_{xy} between \mathbf{x} and \mathbf{y} can be calculated by the following formulas:

$$r_{xy} = \frac{\text{cov}(\mathbf{x}, \mathbf{y})}{\sqrt{D(\mathbf{x})}\sqrt{D(\mathbf{y})}} \quad (7)$$

$$\text{cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N (x_i - E(\mathbf{x}))(y_i - E(\mathbf{y})) \quad (8)$$

$$D(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N (x_i - E(\mathbf{x}))^2 \quad (9)$$

$$E(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^N x_i \quad (10)$$

Table 3 Correlation coefficients of the plain and cipher images

	Lena		Baboon		Pepper		Plane	
	Fig. 3a	Fig. 3e	Fig. 3b	Fig. 3f	Fig. 3c	Fig. 3g	Fig. 3d	Fig. 3h
Horizontal	0.96457	-0.02457	0.89509	-0.01427	0.97824	0.01218	0.96001	-0.01040
Vertical	0.97864	-0.02264	0.81079	-0.04469	0.97900	0.03890	0.94769	-0.01708
Diagonal	0.95098	-0.01930	0.76255	0.01061	0.96189	0.04578	0.92159	0.01095

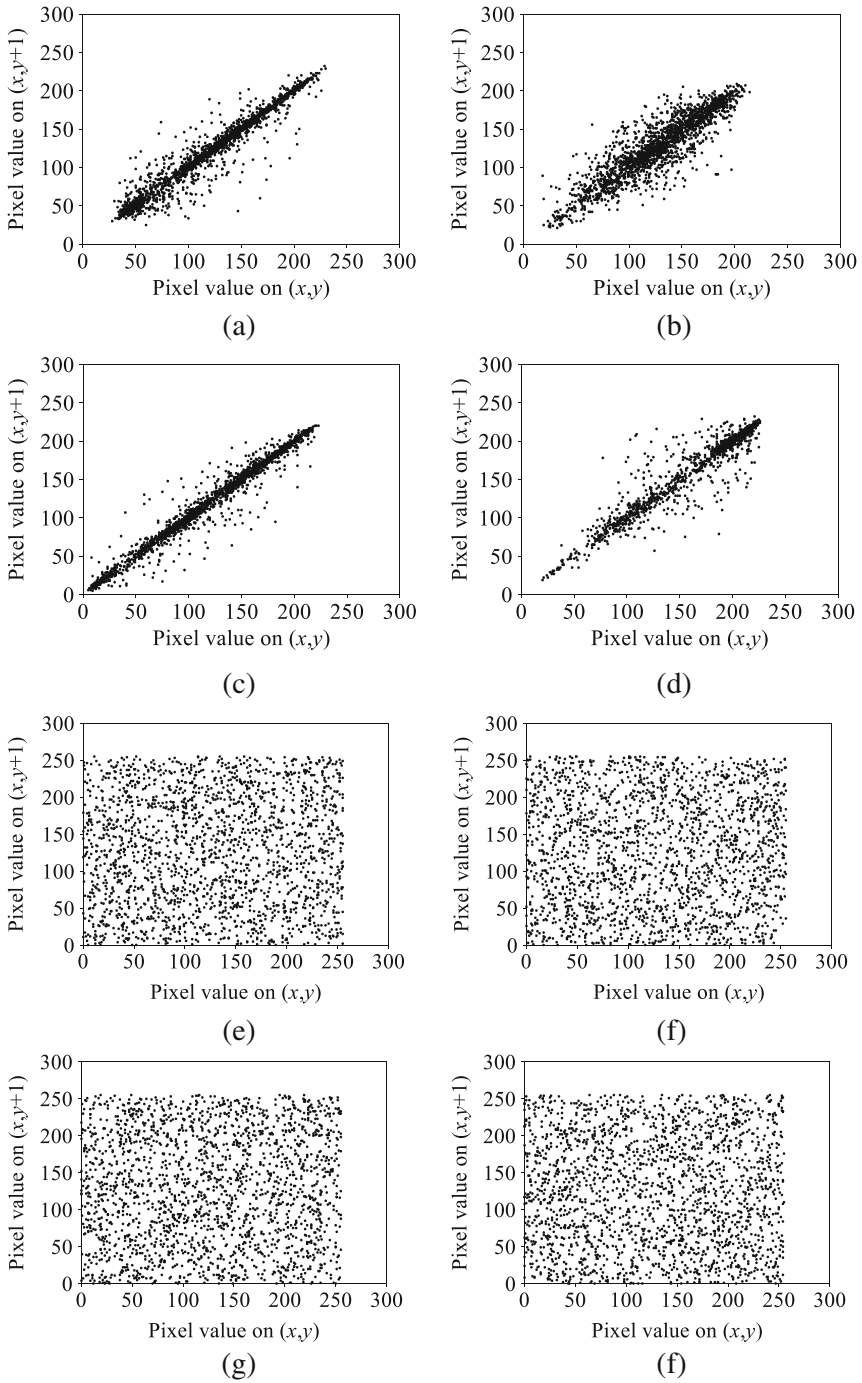


Fig. 8 Results of correlation analysis. **a–d** Correlations in horizontal direction for the plain images of Lena, Baboon, Pepper and Plane, respectively; **e–h** Correlations in horizontal direction for the cipher images of Lena, Baboon, Pepper and Plane, respectively

Table 4 Results of key sensitivity tests ($d = 1$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6100%	33.4646%	99.6080%	28.6229% (28.6181%)
Baboon	99.6093%	33.4722%	99.6087%	27.5729%(27.5702%)
Pepper	99.6057%	33.4591%	99.6096%	29.5946%(29.5959%)
Plane	99.6105%	33.4529%	99.6115%	32.4702%(32.4698%)

Where, $cov(x,y)$ represents the covariance between x and y , $D(x)$ represents the variance of x , $E(x)$ returns the mean value of x , and N is the length of vector x .

Here, let $N = 2000$, and then calculate the correlation coefficients of the plain images (as shown in Fig. 4a–d) and their corresponding cipher images (as shown in Fig. 4e–h) in the horizontal, vertical and diagonal directions, respectively. List the calculated results in Table 3. Meanwhile, illustrate the correlation of each image in the horizontal direction in Fig. 8.

As can be seen from Table 3 and Fig. 8, the plain images have strong correlation between adjacent pixels with the correlation coefficient being close to 1; while the cipher images hardly have correlation between adjacent pixels due to their correlation coefficients being close to 0. These indicate that the proposed scheme can frustrate the attacks based on statistical properties.

4.4 Sensitivity analysis

The indicators of NPCR (number of pixels change rate) and UACI (unified average changing rate) are usually used to quantitatively measure the system sensitivity [5]. Assume that the images C_1 and C_2 are both of size $M \times N$. Then define NPCR and UACI as follows:

$$NPCR = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N |\text{Sign}(C_1(i, j) - C_2(i, j))| \times 100\% \tag{11}$$

$$UACI = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N \frac{|C_1(i, j) - C_2(i, j)|}{255} \times 100\% \tag{12}$$

Table 5 Results of key sensitivity tests ($d = 2$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6093%	33.4714%	99.6077%	28.6076%(28.6181%)
Baboon	99.6106%	33.4683%	99.6075%	27.5657%(27.5702%)
Pepper	99.6078%	33.4596%	99.6099%	29.5980%(29.5959%)
Plane	99.6104%	33.4648%	99.6100%	32.4820%(32.4698%)

Table 6 Results of key sensitivity tests ($d = 4$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6073%	33.4805%	99.6075%	28.6237%(28.6181%)
Baboon	99.6073%	33.4651%	99.6093%	27.5705%(27.5702%)
Pepper	99.6096%	33.4748%	99.6079%	29.6014%(29.5959%)
Plane	99.6089%	33.4643%	99.6101%	32.4750%(32.4698%)

For two random images, the theoretical values of NPCR and UACI are $255/256 \approx 99.6094\%$ and $257/768 \approx 33.4635\%$, respectively [27].

4.4.1 Key sensitivity analysis

Assume the secret keys K_1 and K_2 are only 1-bit different, the key sensitivity can be checked from the following two cases:

Case 1: Encrypt the plain image P by the proposed scheme with secret keys K_1 and K_2 to obtain two cipher images, denoted by C_1 and C_2 respectively. Then calculate the values of NPCR and UACI between C_1 and C_2 .

Case 2: Encrypt the plain image P_1 by the proposed scheme with the secret key K_1 to get the cipher image, denoted by C . Then decrypt the cipher image C by the proposed scheme with the secret key K_2 to get the recovered image, denoted by P_2 . Then calculate the values of NPCR and UACI between P_1 and P_2 .

When P_1 takes Lena, Baboon, Pepper and Plane (as shown in Fig. 4a–d, respectively) in turn and P_2 is a random image, the theoretical values of NPCR are all $255/256 \approx 99.6094\%$, while the theoretical values of UACI are approximately 28.6181, 27.5702, 29.5959 and 32.4698%, respectively [27].

Without loss of generality, we take the plain images as shown in Fig. 4a–d (all of size 353×398) as examples to check the key sensitivity with the key length of 64 bits ($d = 1$), 128 bits ($d = 2$), 256 bits ($d = 4$), 512 bits ($d = 8$), 1028 bits ($d = 16$) and 2048 bits ($d = 32$). For the each selected image and key length, we do 100 trials to calculate the average values of NPCR and UACI, respectively. The calculated results are listed in Tables 4, 5, 6, 7, 8 and 9, where the

Table 7 Results of key sensitivity tests ($d = 8$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6067%	33.4598%	99.6071%	28.6112%(28.6181%)
Baboon	99.6110%	33.4602%	99.6091%	27.5682%(27.5702%)
Pepper	99.6111%	33.4582%	99.6080%	29.6036%(29.5959%)
Plane	99.6085%	33.4627%	99.6072%	32.4661%(32.4698%)

Table 8 Results of key sensitivity tests ($d = 16$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6068%	33.4689%	99.6107%	28.6190%(28.6181%)
Baboon	99.6105%	33.4679%	99.6096%	27.5719%(27.5702%)
Pepper	99.6094%	33.4649%	99.6087%	29.5898%(29.5959%)
Plane	99.6101%	33.4598%	99.6079%	32.4708%(32.4698%)

values in brackets are the theoretical values of corresponding indicators. In each test, the secret keys K_1 and K_2 are only different on the k -th bit and are randomly generated by the following formulas:

$$K_1 = \text{floor}(\text{rand}(1, 16 \times d) \times 1000) \bmod 16 \quad (13)$$

$$K_2 = K_1 \quad (14)$$

$$k = \text{floor}(\text{rand} \times 1000) \bmod (64 \times d) + 1 \quad (15)$$

$$K_2(\text{floor}((k-1)/4) + 1) = \text{bitxor}(K_2(\text{floor}((k-1)/4) + 1), \text{pow2}((k-1) \bmod 4)) \quad (16)$$

Where, $\text{rand}(a,b)$ produces a 0–1 uniformly distributed random number matrix with size of $a \times b$, $\text{floor}(x)$ returns the largest integer less than or equal to x , $\text{pow2}(x)$ means 2^x , and $\text{bitxor}(x,y)$ means x bitwise XOR y .

The results in Tables 4, 5, 6, 7, 8 and 9 show that the calculated values of NPCR and UACI are very close to their corresponding theoretical values. This indicates that one bit change of any secret key with length of 64 bits ($d = 1$), 128 bits ($d = 2$), 256 bits ($d = 4$), 512 bits ($d = 8$), 1024 bits ($d = 16$) or 2048 bits ($d = 32$) will lead to dramatic changes in cipher images or recovered images, which demonstrates that the proposed scheme has strong key sensitivity, and each secret key in key space is valid.

4.4.2 Plaintext sensitivity analysis

Assume that the plain images P_1 and P_2 are only one pixel different. Encrypt them by the proposed scheme with the same key K to get two cipher images, denoted by

Table 9 Results of key sensitivity tests ($d = 32$)

	Case 1		Case 2	
	NPCR(99.6094%)	UACI(33.4635%)	NPCR(99.6094%)	UACI
Lena	99.6075%	33.4655%	99.6126%	28.6172%(28.6181%)
Baboon	99.6129%	33.4735%	99.6111%	27.5710%(27.5702%)
Pepper	99.6088%	33.4690%	99.6096%	29.5998%(29.5959%)
Plane	99.6101%	33.4580%	99.6080%	32.4618%(32.4698%)

Table 10 Results of plaintext sensitivity tests ($d = 4$, unit: %)

Image size	NPCR (99.6094%)				UACI (33.4635%)			
	Lena	Baboon	Pepper	Plane	Lena	Baboon	Pepper	Plane
128×128	99.6044	99.6121	99.6066	99.6088	33.5194	33.3706	33.4740	33.4788
256×256	99.6113	99.6096	99.6072	99.6084	33.4643	33.4921	33.4356	33.3900
353×398	99.6076	99.6091	99.6111	99.6090	33.4573	33.4656	33.4375	33.4658
384×384	99.6102	99.6092	99.6119	99.6082	33.4725	33.4813	33.4961	33.4729
512×512	99.6072	99.6100	99.6079	99.6096	33.4203	33.4414	33.4747	33.4598

C_1 and C_2 , respectively. Then calculate the values of NPCR and UACI between C_1 and C_2 .

Here taking the plain images Lena, Baboon, Pepper and Plane with size of 128×128 , 256×256 , 353×398 , 384×384 and 512×512 as examples to explore the proposed scheme with key length of 256 bits ($d = 4$) and 2048 bits ($d = 32$). For each image we do 100 trials to calculate the average values of NPCR and UACI, respectively. The results are listed in Tables 10 and 11. Note that for each test, the secret key is randomly produced by the Eq. (13).

As can be seen from Tables 10 and 11, the calculated values of NPCR and UACI are very close to their corresponding theoretical values, which indicates that the slight change in plain images will lead to the dramatic changes in cipher images even with the same secret key. So, the proposed scheme possesses strong plaintext sensitivity. Hence, the proposed scheme can in a way frustrate the differential attack base on the plain images.

4.4.3 Cipher-text sensitivity analysis

Assume that the plain image P_1 was encrypted by the proposed scheme with secret key K_1 to generate the cipher image, denoted by C_1 . Change C_1 by only one pixel to get the image named by C_2 , i.e. $C_2 = C_1$ except that $C_2(i,j) = C_1(i,j) + 1 \pmod{256}$ on a certain pixel position (i,j) . Assume that K_1 is changed by only one bit to get another key named K_2 . We will check the cipher image sensitivity from the following two cases:

Case 1: Decrypt cipher image C_2 by the proposed scheme with secret key K_1 to get the recovered image named P_2 . Then use indicators of NPCR and UACI to analyze the

Table 11 Results of plaintext sensitivity tests ($d = 32$, unit: %)

Image size	NPCR (99.6094%)				UACI (33.4635%)			
	Lena	Baboon	Pepper	Plane	Lena	Baboon	Pepper	Plane
128×128	99.6108	99.6067	99.6117	99.6048	33.5236	33.4001	33.5023	33.5063
256×256	99.6090	99.6065	99.6058	99.6114	33.4658	33.4966	33.4562	33.3997
353×398	99.6121	99.6117	99.6092	99.6087	33.4771	33.4619	33.4522	33.4594
384×384	99.6090	99.6113	99.6082	99.6106	33.4734	33.4891	33.4974	33.4783
512×512	99.6102	99.6090	99.6103	99.6084	33.4338	33.4371	33.4741	33.4588

Table 12 Results of cipher image sensitivity tests ($d = 4$)

Image	Using correct secret key K_1		Using wrong secret key K_2	
	NPCR(99.6094%)	UACI	NPCR(99.6094%)	UACI(33.4635%)
Lena	99.6078%	28.6203%(28.6181%)	99.6115%	33.4622%
Baboon	99.6101%	27.5663%(27.5702%)	99.6104%	33.4639%
Pepper	99.6088%	29.5912%(29.5959%)	99.6089%	33.4710%
Plane	99.6100%	32.4616%(32.4698%)	99.6069%	33.4716%

level of difference between P_1 and P_2 . When P_1 takes the plain images as shown in Fig. 4a–d in turn and P_2 is a random image, the theoretical values of NPCR are all $255/256 \approx 99.6094\%$, while the theoretical values of UACI are 28.6181, 27.5702, 29.5959 and 32.4698%, respectively [27].

Case 2: Decrypt C_1 and C_2 by the proposed scheme with the slightly changed key K_2 to get the recovered images, denoted by P_3 and P_4 , respectively. Then use indicators of NPCR and UACI to analyze the degree of difference between P_3 and P_4 . The theoretical values of NPCR and UACI are $255/256 \approx 99.6094\%$ and $257/768 \approx 33.4635\%$, respectively [27].

Without loss of generality, take the plain images Lena, Baboon, Pepper and Plane (as shown in Fig. 4a–d all of size 353×398) as examples to test the cipher-text sensitivity of proposed scheme with the key length of 256 bits ($d = 4$) and 2048 bits ($d = 32$). For given plain image P_1 , firstly, randomly generate two secret keys K_1 and K_2 with Eqs. (13)–(16). Secondly, encrypt P_1 by the proposed scheme with the secret key K_1 to get the cipher image C_1 . Thirdly, randomly change one pixel of C_1 by one bit to get the image C_2 , and then decrypt C_2 by the proposed scheme with the secret key K_1 to get P_2 . Fourthly, decrypt C_1 and C_2 by the proposed scheme with the secret key K_2 to get their corresponding P_3 and P_4 . Finally, calculate the values of NPCR and UACI with Eqs. (11)–(12). Repeat the above trails for 100 times, and calculate the average values of NPCR and UACI, respectively. The results are listed in Tables 12 and 13, where, the values in brackets are theoretical values of corresponding indicators. In Table 12 the length of secret key is 256 bits ($d = 4$), while in Table 13 the length of secret key is 2048 bits ($d = 32$).

As can be seen from Tables 12 and 13, the test value of NPCR and UACI are very close to their corresponding theoretical values, indicating that the proposed scheme is extremely sensitive to the tiny change of cipher images, which is helpful for the scheme to resist the differential attack based on cipher images.

Table 13 Results of cipher image sensitivity tests ($d = 32$)

Image	Using correct secret key K_1		Using wrong secret key K_2	
	NPCR(99.6094%)	UACI	NPCR(99.6094%)	UACI(33.4635%)
Lena	99.6083%	28.6226%(28.6181%)	99.6123%	33.4620%
Baboon	99.6113%	27.5724%(27.5702%)	99.6078%	33.4468%
Pepper	99.6083%	29.5985%(29.5959%)	99.6071%	33.4593%
Plane	99.6112%	32.4731%(32.4698%)	99.6112%	33.4641%

Table 14 Results of information entropy tests for all-white and all-black images ($d = 2$ and $d = 4$)

Image size	Entropy of white($d = 2$)		Entropy of black($d = 2$)		Entropy of white($d = 4$)		Entropy of black($d = 4$)	
	Plain	Cipher	Plain	Cipher	Plain	Cipher	Plain	Cipher
128×128	0	7.98810	0	7.98994	0	7.98912	0	7.98912
256×256	0	7.99726	0	7.99706	0	7.99752	0	7.99705
353×398	0	7.99871	0	7.99860	0	7.99857	0	7.99862
384×384	0	7.99845	0	7.99875	0	7.99884	0	7.99863
512×512	0	7.99932	0	7.99926	0	7.99937	0	7.99926

4.5 Information entropy

Information entropy reflects the uncertainty of the image information. For a grey image of L -level, when the occurrence probability of grey value i is $p(m_i)$, the information entropy of this image is formulated as follows:

$$H(m) = -\sum_{i=0}^{L-1} p(m_i) \log_2(p(m_i)) \quad (17)$$

The larger the information entropy is, the more uncertainty the image information is, and the less intelligible the image is. For an 8-bit random image, the theoretical value of information entropy is 8.

Here, $L = 256$. We used the plain images all-zeros (pure black), all-255 s (pure white), Lena, Baboon, Pepper and Plane (all with size of 128×128 , 256×256 , 353×398 , 384×384 and 512×512) and their corresponding cipher images. The secret keys used in each test are randomly generated by Eq. (13) with the length of 128 bits ($d = 2$) and 256 bits ($d = 4$). The calculated values of information entropy for these plain and cipher images are listed in Tables 14, 15 and 16.

According to the results in Tables 14, 15 and 16, we can see that the values of information entropy for plain images are largely different from the theoretical value of noise image (i.e. 8), while the values of information entropy for cipher images are very close to 8. For the plain images of all-black and all-white whose information entropies are 0, their corresponding cipher images still have excellent information entropies (see Table 14). These demonstrate that the proposed scheme can resist the attacks based on information entropy.

Table 15 Results of information entropy tests ($d = 2$)

Image size	Entropy of lena		Entropy of baboon		Entropy of pepper		Entropy of plane	
	Plain	Cipher	Plain	Cipher	Plain	Cipher	Plain	Cipher
128×128	7.43087	7.98859	7.16100	7.98894	7.58236	7.98928	6.79748	7.98724
256×256	7.45304	7.99754	7.24331	7.99766	7.58757	7.99725	6.76401	7.99693
353×398	7.44407	7.99875	7.29470	7.99869	7.58566	7.99870	6.72195	7.99862
384×384	7.44493	7.99864	7.30047	7.99895	7.58477	7.99881	6.72128	7.99856
512×512	7.44506	7.99915	7.35834	7.99933	7.59365	7.99927	6.70246	7.99931

Table 16 Results of information entropy tests ($d = 4$)

Image size	Entropy of lena		Entropy of baboon		Entropy of pepper		Entropy of plane	
	Plain	Cipher	Plain	Cipher	Plain	Cipher	Plain	Cipher
128×128	7.43087	7.98736	7.16100	7.98791	7.58236	7.98846	6.79748	7.98863
256×256	7.45304	7.99721	7.24331	7.99745	7.58757	7.99713	6.76401	7.99733
353×398	7.44407	7.99878	7.29470	7.99870	7.58566	7.99876	6.72195	7.99876
384×384	7.44493	7.99868	7.30047	7.99873	7.58477	7.99877	6.72128	7.99857
512×512	7.44506	7.99928	7.35834	7.99926	7.59365	7.99937	6.70246	7.99928

4.6 Comparison analysis

Without losing generality, take image Lena of size 512×512 as an example. Compare the proposed scheme with the AES in CBC mode and the schemes in [6, 7, 23] in aspects of encryption/decryption speed and system sensitivity. The computer used has the same configuration as the one in Section 3. The system sensitivities of AES and the schemes in [6, 7, 23] are listed in Table 17. Here, AES uses a key of 128-bit length. In [23], the initial value and parameter of Logistic map are directly used as the secret key, so the key length is about 96 bits. The scheme in [7] uses the initial value and parameter of Tent map as the secret key, so its key length is about 100 bits. In [6], the initial values and parameters of two Tent map are directly used as the secret key in the encryption scheme, so the key length is about 200 bits. Here, our proposed system with the key of length 512 bits is used to make the comparison.

According to Table 17, and the security analyses of our proposed system and the systems in [6, 7, 23], the results of the performance comparison are listed in Table 18.

From Table 18, the encryption/decryption speed of AES is about 7.5709Mbps, and this speed can be used as a threshold to measure the speed of newly proposed image encryption algorithms. The image encryption algorithms having higher processing speed than AES are considered to be practically significant. The scheme in [23] is slower in encryption/decryption speed than AES. The system sensitivities of AES and

Table 17 Sensitivity analysis results of several cryptosystems (Unit: %)

		Key sensitivity		Plaintext sensitivity	Cipher-text sensitivity with correct key
		Encryption process	Decryption process		
Theoretical	NPCR	99.6094	99.6094	99.6094	99.6094
	UACI	33.4635	28.6242	33.4635	28.6242
AES	NPCR	99.6110	99.6121	47.0554	0.006447
	UACI	33.4498	28.6131	15.8099	0.001764
Ref. [23]	NPCR	95.9857	96.8443	47.4753	48.2031
	UACI	-	39.2263	-	20.3384
Ref. [7]	NPCR	99.6091	99.6089	45.5548	49.2057
	UACI	33.4616	28.6290	15.3028	14.1885
Ref. [6]	NPCR	99.6099	99.6096	99.6090	99.6057
	UACI	33.4624	28.6321	33.4631	28.6223

In [23], the ciphered image consists of the numbers of iterations of chaotic system, so the values of UACIs in the key and plaintext sensitivities cannot be calculated. In [6], the rounds of encryption processing are 2

Table 18 Performance comparison analysis results

Scheme	Encryption speed (Mbps)	Decryption speed (Mbps)	Statistical properties of cipher images	Key sensitivity	Plaintext sensitivity	Cipher-text sensitivity	Information entropy
AES	7.5709	7.5709	Excellent	Excellent	Average	Poor	Excellent
Ref. [23]	0.8247	1.2243	Excellent	Good	Average	Average	Excellent
Ref. [7]	20.9715	20.9715	Excellent	Excellent	Average	Average	Excellent
Ref. [6]	42.7990	42.7990	Excellent	Excellent	Excellent	Excellent	Excellent
Our proposed	77.6723	77.6723	Excellent	Excellent	Excellent	Excellent	Excellent

Excellent > Good > Average > Fair > Poor

the schemes in [7, 23] are not good. However, both our proposed scheme and the scheme in [6] possess fast encryption/decryption speed and excellent system sensitivities. Furthermore, our proposed scheme is the fastest one. So, our scheme is superior to the above schemes.

5 Conclusion

This paper presents a new plaintext-related image encryption system. In the proposed system, the encryption algorithm and the decryption algorithm are exactly the same. They both include the operations of plaintext-related scrambling once, diffusion twice, and matrix rotating 180 degrees four times. The key length of proposed system can be extended to support $64d$ -bit length, where d is a positive integer. Since the encryption and decryption procedures are identical, the proposed scheme can use one device to fulfill both encryption and decryption to save hardware resources. And due to the usage of plaintext-related scrambling algorithm, the proposed scheme can frustrate the chosen/known plaintext attacks.

For the 512-bit long key, the encryption/decryption speed is up to 77.6723Mbps in the used computer. Since the secret code matrix (i.e. X in Fig. 2) can be generated separately from the confusion and diffusion process, the average encryption/decryption speed will be far higher than the speed of processing one single image when encrypting/decrypting multiple images. The simulation results and comparison analysis show that the proposed scheme has some merits, such as fast encryption speed, large key space, strong key sensitivity, strong plaintext sensitivity, good statistical properties of cipher images, strong cipher-text sensitivity, large information entropy and so on. Therefore, the proposed encryption system can be used in actual communications.

In the future, on the basis of the image cryptosystem with identical encryption and decryption process, further researches on new confusion and diffusion algorithms based on the finite field integer operations will be carried out to improve the processing speed without loss of security. Also, the scheme's application to color images and videos will be studied.

Acknowledgements Thanks go to the anonymous reviewers for their valuable comments. This work was fully supported by the National Natural Science Foundation of China (Grant No. 61562035), the Natural Science Foundation of Jiangxi Province (Grant No. 20161BAB202058), and the Science and Technology Project of Education Department of Jiangxi Province (Grant No. GJJ160426).

References

1. AdbEl-Latif AA, Li L, Zhang T, Wang N, Song X, Niu X (2012) Digital image encryption scheme based on multiple chaotic systems. *Sens Imaging Int J* 13(2):67–88
2. Alvarez G, Li SJ (2006) Some basic cryptographic requirements for chaos-based cryptosystems. *Int J Bifurcation Chaos* 16(8):2129–2151
3. Baranovsky A, Daems D (1995) Design of one-dimensional chaotic maps with pre-scribed statistical properties. *Int J Bifurcation and Chaos* 5(6):1585–1598
4. Belazi A, Hermassi H, Rhouma R, Belghith S (2014) Algebraic analysis of a RGB image encryption algorithm based on DNA encoding and chaotic map. *Nonlinear Dynam* 76(4):1989–2004
5. Chen GR, Mao Y, Chui CK (2004) A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons Fractals* 21(3):749–761
6. Cheng P, Yang H, Wei P, Zhang W (2015) A fast encryption algorithm based on chaotic and lookup table. *Nonlinear Dynam* 79(3):2121–2131
7. Eslami Z, Bakhshandeh A (2013) An improvement over an image encryption method based on total shuffling. *Opt Commun* 286(1):51–55
8. EyebeFouda JSA, Effa JY, Sabat SL, Ali M (2014) A fast chaotic block cipher for image encryption. *Commun Nonlinear Sci Numer Simul* 19(3):578–588
9. Fridrich J (1998) Symmetric ciphers based on two-dimensional chaotic maps. *Int J Bifurcation Chaos* 8(6):1259–1284
10. Guesmi R, Farah MAB, Kachouri A, Samet M (2016) Hash key-based image encryption using crossover operator and chaos. *Multimed Tools Appl* 75(8):4753–4769
11. Huang XL (2012) Image encryption using chaotic Chebyshev generator. *NonlinearDynam* 67(4):2411–2417
12. Kulsoom A, Xiao D, Aqeel-ur-Rehman S, Abbas A (2016) An efficient and noise resistive selective image encryption scheme for gray images based on chaotic maps and DNA complementary rules. *Multimed Tools Appl* 75(1):1–23
13. Li S, Li Q, Li W, Mou X, Cai Y (2001) Statistical properties of digital piecewise linear chaotic maps and their roles in cryptography and pseudo-random coding. *Lect Notes Comput Sci* 2260:205–221
14. Liu L, Zhang Q, Wei X (2012) A RGB image encryption algorithm based on DNA encoding and chaos map. *Comput Electr Eng* 38(5):1240–1248
15. Liu H, Wang X, Kadir A (2013) Color image encryption using Choquet fuzzy integral and hyper chaotic system. *Optik* 124(18):3527–3533
16. Matthews R (1989) On the derivation of a “chaotic encryption algorithm”. *Cryptologia* 13(1):29–42
17. Pareek NK, Patidar V, Sud KK (2005) Cryptography using multiple one-dimensional chaotic maps. *Commun Nonlinear Sci Numer Simul* 10(7):715–723
18. Stallings W (2012) Cryptography and network security: principles and practice. *Int Annals of Criminology* 46(4):121–136
19. Su M, Wen W, Zhang Y (2014) Security evaluation of bilateral-diffusion based image encryption algorithm. *Nonlinear Dynam* 77(1–2):243–246
20. Tong X (2012) The novel bilateral-diffusion image encryption algorithm with dynamical compound chaos. *J Syst Softw* 85(4):850–858
21. Wang X, Xu D (2014) A novel image encryption scheme based on Brownian motion and PWLCM chaotic system. *Nonlinear Dynam* 75(1–2):345–353
22. Wang X, Luan D, Bao X (2014) Cryptanalysis of an image encryption algorithm using Chebyshev generator. *Digital Signal Process* 25:244–247
23. Wong KW (2002) A fast chaotic cryptographic scheme with dynamic look-up table. *Phys Lett A* 298(4):238–242
24. Ye G (2013) Chaotic image encryption algorithm using multi-generalized logistic maps. *J Comput Theor Nanosci* 10(11):2789–2795
25. Ye G, Zhao H, Chai H (2016) Chaotic image encryption algorithm using wave-line permutation and block diffusion. *Nonlinear Dynam* 83(4):2067–2077
26. Zhang Y (2014a) Comments on “color image encryption using Choquet fuzzy integral and hyper chaotic System”. *Optik* 125(19):5560–5565
27. Zhang Y (2014b) Plaintext related image encryption scheme using chaotic map. *Telkomnika* 12(1):635–643
28. Zhang Y (2014c) A chaotic system based image encryption algorithm using plaintext-related confusion. *TELKOMNIKA* 12(11):7952–7962
29. Zhang Y (2015) Cryptanalysis of a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Optik* 126(2):223–229
30. Zhang Y (2016) The image encryption algorithm with plaintext-related shuffling. *IETE Tech Rev* 33(3):310–322

31. Zhang Y, Wang X (2014) Analysis and improvement of a chaos-based symmetric image encryption scheme using a bit-level permutation. *Nonlinear Dynam* 77(3):687–698
32. Zhang Q, Guo L, Wei X (2013) A novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Optik* 124(18):3596–3600
33. Zhang L, Hu X, Liu Y, Wong K-W, Gan J (2014) A chaotic image encryption scheme owning temp-value feedback. *Commun Nonlinear Sci Numer Simul* 19(10):3653–3659
34. Zhu Z, Zhang W, Wong K, Yu H (2011) A chaos-based symmetric image encryption scheme using a bitlevel permutation. *Inf Sci* 181(6):1171–1186



Yong Zhang received the Bachelor's degree in petroleum engineering from Chengdu University of Technology (CDUT) in 1998, the MS degree in communication and information systems and the PhD degree in circuits and systems 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 chaotic signal processing, cryptology and quantum information.



Yingjun Tang is the lecturer of Jiangxi University of Finance and Economics (JXUFE). She received her MS and PhD degrees both in computer science from Beijing Jiaotong University (BJTU). Her research fields include digital image processing and information security.