

A joint color image encryption and compression scheme based on hyper-chaotic system

Xiao-Jun Tong · Miao Zhang · Zhu Wang ·
Jing Ma

Received: 12 May 2015 / Accepted: 22 January 2016 / Published online: 4 February 2016
© Springer Science+Business Media Dordrecht 2016

Abstract For the low security and compression performance of the existing joint image encryption and compression technology, an improvement algorithm for joint image compression and encryption is proposed. The algorithm employs the discrete cosine transformation dictionary to sparsely represent the color image and then combines it with the encryption algorithm based on the hyper-chaotic system to achieve image compression and encryption simultaneously. Through the experimental analysis, the algorithm proposed in this paper has a good performance in security and compression.

Keywords Hyper-chaos · Sparse representation · DCT dictionary · Image compression and encryption

1 Introduction

Along with the rapid growth of transmitting images over public networks, the network bandwidth and security have received a lot of attention. Compressing

images to save bandwidth and encrypting images to protect privacy have become hot research areas.

The traditional method is compressing the image before the encryption is done. Encryption and compression are unrelated. However, such traditional method will sacrifice flexibility and calculation simplicity. A better way is the joint operation of compression and encryption in a single process. However, the contradiction between the image compression and encryption makes the joint operation difficult. Researches have been working on the joint operation of image compression and encryption to achieve better performance on compression and encryption [1–11].

For the joint operation of compression and encryption, the most common method is embedding the encryption into compression [1, 2] thus to get a balance in image compression and encryption. For the different compression methods, the joint algorithms of compression and encryption are divided into different situations at the present stage. (1) Using the compression method of Joint Photographic Experts Group (JPEG) or related discrete cosine transformation (DCT), the encryption is embedded into the compression process. By means of encrypting DCT coefficient, some joint encryption and compression schemes [3–5] are proposed. As the perceptual information concentrates at low-frequency DCT coefficients, some selective encryption schemes are proposed. Reference [3] proposed to encrypt the DC coefficient and the sign bit of all AC coefficients using a spatiotemporal chaotic system. However, Wu

X.-J. Tong (✉) · M. Zhang
Harbin Institute of Technology, School of Computer
Science and Technology, Weihai 264209, China
e-mail: tong_xiaojun@163.com

Z. Wang
Harbin Institute of Technology, School of Information and
Electrical Engineering, Weihai 264209, China

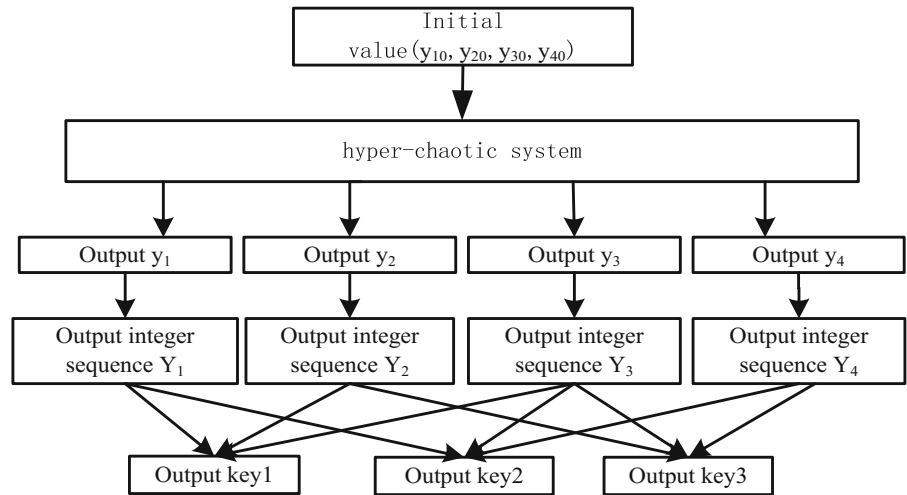
J. Ma
Information Assurance Laboratory, Beijing 100072, China

and Kuo stated that selective encryption is not suitable for DCT-based compression algorithm [6]. Moreover, they proposed some perceptual attacks to restore perceptual image. In full encryption, Yuen and Wong proposed chaos-based joint image compression and encryption algorithm using DCT [5]. In their scheme, the DCT coefficients of the whole image are separated to encrypt. However, compression performance is not high. (2) Compared with the JPEG, JPEG2000 based on wavelet transformation has a better compression performance. So there are a lot of image compression and encryption algorithms based on JPEG2000 and related wavelet transformation [7–9]. In Ref. [7], only low-frequency region of the wavelet transform is encrypted. The security need to be improved. Reference [8] presented a new private key cryptosystem based on the finite-field wavelet. However, the compression performance was not analyzed. Yang et al. [9] encrypted the image between wavelet transformation and SPIHT coding. However, the compression performance is lower than that of only compression using wavelet transformation. (3) Due to the certain resemblance in the sense of secrecy between cryptographic ciphers and entropy coders, some joint encryption and compression schemes based on entropy coding compression method are proposed [10–12]. In these methods, encryption is embedded in entropy coding. Reference [10] proposed to embed encryption in Huffman coding based on multiple Huffman tables. References [11, 12] proposed to embed encryption in arithmetic coding. However, in these schemes, encryption is an additional feature but not the primary concern. Therefore, the security may not be satisfactory. References [10, 11] suffers known-plaintext attack [13, 14]. (4) Using compression technique based on sparse decomposition, the joint encryption and compression algorithms are proposed. Image sparse representation provides a new way to solve the distortion problem of reconstructed image for low bit rate image compression. Wu [15] proposed the joint encryption and compression algorithm using sparse decomposition, Secure Hash Algorithm-1 (SHA-1) and chaotic map. However, the scheme needs a large amount of calculation in the generation of over-complete dictionary. Therefore, the scheme does not suitable for the real-time transmission of images. Rebollo-Neira [16, 17] proposed the idea of encryption image folding in which orthogonal projection matrix is used to transform the coefficients into orthogonal complementary space of the host image, and then the coeffi-

cients are folded into the host image. The algorithm can obtain reconstructed image with better quality at high compression ratio. However, the experimental results presented in the paper are just an ideal state of the data, and there are some problems for the algorithm. The algorithm does not compress the additional data required in decompression and decryption, so it takes up a lot of network resources in transmission. In addition, the compressed and encrypted images carry some information of the original images. When using the error key to decompress and decrypt the image, we can obtain the original outline information. That will create a serious security hole in the algorithm. In order to solve these problems, according to the idea of encryption image folding proposed in Refs. [16, 17], a joint color image encryption and compression scheme based on hyper-chaotic system is proposed by this paper.

In order to improve the security of the joint color image encryption and compression scheme, we design the hyper-chaos-based scrambling and diffusion scheme. Conventional cryptographic techniques, such as International Data Encryption Algorithm (IDEA) or Advanced Encryption Standard (AES), are not suitable for image encryption due to the bulky data capacity and high correlation among pixels in image files [18, 19]. Owing to simplicity in implementation, high speed and good cryptography properties of chaos, the chaotic encryption algorithm has caused widespread concern since it was first proposed by the UK mathematician Matthews in 1989 [20]. With the chaotic system used in the image encryption, high-dimensional chaotic system with high complexity is subjected to the value more and more [21–24]. As high-dimensional chaotic system, hyper-chaotic system has greater complexity. Its pseudorandomness and long-term unpredictability is stronger. In several existing hyper-chaos-based encryption algorithms, due to their inappropriate cryptographic structure, the security is not good. Reference [25] proposed a fast color image encryption algorithm based on hyper-chaotic systems. In Ref. [26], a novel image encryption algorithm with only one round diffusion process based on hyper-chaotic system was proposed. By applying known-plaintext and chosen-plaintext attacks, Ref. [27] broke it. In Ref. [28], a novel image encryption scheme based on improved hyper-chaotic sequences was proposed. Reference [29] evaluated the security of Ref. [28] and broke it. In this paper, the security is improved based on hyper-chaotic system in three aspects: the hyper-chaos-based scram-

Fig. 1 The structure of pseudorandom sequence generator



bling method is used to scramble the effective coefficients' position to remove the relativity among pixels within the block; the sparse coefficients are controlled by the hyper-chaotic system to encrypt the coefficients; the hyper-chaos-based diffusion method is used in the folded image to encrypt the final data. Moreover, Huffman coding is used to compress the additional position information required in decompression and decryption to save network bandwidth.

The rest of paper is organized as follows. Section 2 introduces the scrambling and diffusion encryption algorithms based on hyper-chaotic system. Section 3 describes the joint image encryption and compression scheme of color image based on hyper-chaotic system. Section 4 gives the test results and analysis of our improved scheme. Finally, Sect. 5 makes a conclusion for this paper.

2 The encryption algorithm based on hyper-chaotic system

In this paper we design the system of differential equations like Eq. (1), the Lyapunov exponents are 0.81, 0.31, 0, -24.11. The state of the system is hyper-chaos which has a high complexity.

$$\begin{cases} \frac{dy_1}{dt} = 16(y_2 - y_1) \\ \frac{dy_2}{dt} = 45y_4 - 2y_2 + 47.6y_1 - y_1y_3 - y_3y_4 \\ \frac{dy_3}{dt} = -4y_3 + y_2y_4 + y_1y_2 \\ \frac{dy_4}{dt} = -y_2 - y_4 - 0.05y_1y_3 \end{cases} \quad (1)$$

2.1 The generation of pseudorandom sequence

In order to make a full use of the chaotic sequence generated by hyper-chaotic system, different bits of different dimension of chaotic sequence are selected as a key sequence. The structure of pseudorandom sequence generator is shown in Fig. 1.

Assuming the color image size is $M \times N$, Fig. 1 illustrates the concrete steps of generating pseudorandom sequence. The steps are as follows.

- (1) Generate the hyper-chaotic sequence.

Set the initial value $(y_{10}, y_{20}, y_{30}, y_{40})$ of the hyper-chaotic system Eq. (1) to iterate $(M \times N)/3 + 1000$ times. In order to eliminate transient effect of the chaotic sequence and enhance their sensitivity to initial conditions, we get rid the first 1000 groups value of the chaotic sequence then get the sequences y_1, y_2, y_3, y_4 .

- (2) Discretize the hyper-chaotic sequence.

For each value $y_{ij}, i = 1, 2, 3, 4, j = 1, 2, \dots, M \times N \div 3$, we get the decimal part of chaotic sequence by Eq. (2).

$$\Delta y_{ij} = y_{ij} - \lfloor y_{ij} \rfloor \quad (2)$$

$\lfloor x \rfloor$ returns the value of x to the nearest integers, less than or equal to x . Use Eq. (3) to discretize the decimal part and get the integers sequence Y_1, Y_2, Y_3, Y_4 .

$$Y_{ij} = \text{mod} \left(\left\lfloor \Delta y_{ij} \times 10^{14} \right\rfloor, 65536 \right) \quad (3)$$

- (3) Select different bits of each hyper-chaotic sequence to generate key sequence.

The integer Y_{ij} can be represented as the form shown in Eq. (4).

$$Y_{ij} = w_{ij_{15}} \times 2^{15} + w_{ij_{14}} \times 2^{14} + \dots + w_{ij_1} \times 2^1 + w_{ij_0}, (w_{ij_0} \dots w_{ij_{15}} \in \{0, 1\}) \tag{4}$$

In order to reduce the correlation among the three pseudorandom sequences and enhance the rate of reusing, we select different bits of sequences Y_1, Y_2, Y_3 and Y_4 .

The key sequence $Key1$ comes from the 0th to the 7th bit of Y_{1j}, Y_{2j}, Y_{3j} , and its expression is shown as Eq. (5).

$$\left\{ \begin{aligned} key1(k) &= w_{1j_7} \times 2^7 + w_{1j_6} \times 2^6 + w_{1j_5} \times 2^5 \\ &+ w_{1j_4} \times 2^4 + w_{1j_3} \times 2^3 + w_{1j_2} \times 2^2 + w_{1j_1} \\ &\times 2^1 + w_{1j_0} \\ key1(k+1) &= w_{2j_7} \times 2^7 + w_{2j_6} \times 2^6 + w_{2j_5} \times 2^5 \\ &+ w_{2j_4} \times 2^4 + w_{2j_3} \times 2^3 + w_{2j_2} \times 2^2 + w_{2j_1} \\ &\times 2^1 + w_{2j_0} \\ key1(k+2) &= w_{3j_7} \times 2^7 + w_{3j_6} \times 2^6 + w_{3j_5} \times 2^5 \\ &+ w_{3j_4} \times 2^4 + w_{3j_3} \times 2^3 + w_{3j_2} \times 2^2 + w_{3j_1} \\ &\times 2^1 + w_{3j_0} \end{aligned} \right. \tag{5}$$

The key sequence $Key2$ comes from the 4th to the 11th bit of Y_{3j}, Y_{4j}, Y_{1j} , and its expression is shown as Eq. (6).

$$\left\{ \begin{aligned} key2(k) &= w_{3j_{11}} \times 2^7 + w_{3j_{10}} \times 2^6 + w_{3j_9} \times 2^5 \\ &+ w_{3j_8} \times 2^4 + w_{3j_7} \times 2^3 + w_{3j_6} \times 2^2 + w_{3j_5} \\ &\times 2^1 + w_{3j_4} \\ key2(k+1) &= w_{4j_{11}} \times 2^7 + w_{4j_{10}} \times 2^6 + w_{4j_9} \\ &\times 2^5 + w_{4j_8} \times 2^4 + w_{4j_7} \times 2^3 + w_{4j_6} \times 2^2 \\ &+ w_{4j_5} \times 2^1 + w_{4j_4} \\ key2(k+2) &= w_{1j_{11}} \times 2^7 + w_{1j_{10}} \times 2^6 + w_{1j_9} \\ &\times 2^5 + w_{1j_8} \times 2^4 + w_{1j_7} \times 2^3 + w_{1j_6} \times 2^2 \\ &+ w_{1j_5} \times 2^1 + w_{1j_4} \end{aligned} \right. \tag{6}$$

The key sequence $Key3$ comes from the 8th to the 15th bit of Y_{2j}, Y_{3j}, Y_{4j} , and its expression is shown as Eq. (7).

$$\left\{ \begin{aligned} key3(k) &= w_{2j_{15}} \times 2^7 + w_{2j_{14}} \times 2^6 + w_{2j_{13}} \times 2^5 \\ &+ w_{2j_{12}} \times 2^4 + w_{2j_{11}} \times 2^3 + w_{2j_{10}} \times 2^2 + w_{2j_9} \\ &\times 2^1 + w_{2j_8} \\ key3(k+1) &= w_{3j_{15}} \times 2^7 + w_{3j_{14}} \times 2^6 + w_{3j_{13}} \\ &\times 2^5 + w_{3j_{12}} \times 2^4 + w_{3j_{11}} \times 2^3 + w_{3j_{10}} \times 2^2 \\ &+ w_{3j_9} \times 2^1 + w_{3j_8} \\ key3(k+2) &= w_{4j_{15}} \times 2^7 + w_{4j_{14}} \times 2^6 + w_{4j_{13}} \\ &\times 2^5 + w_{4j_{12}} \times 2^4 + w_{4j_{11}} \times 2^3 + w_{4j_{10}} \times 2^2 \\ &+ w_{4j_9} \times 2^1 + w_{4j_8} \end{aligned} \right. \tag{7}$$

The relation of j and k in Eqs. (5), (6) and (7) is $k = j \times 3 - 2$. While finishing the above three steps, we can get the pseudorandom sequence $key1, key2$ and $key3$ which can be used to encrypt the image.

NIST SP800-22 tests [30] are used to detect the deviation of a sequence from a true random sequence. For each test, if the P value is greater than a predefined threshold α , then the sequence is considered to pass the test. Commonly, the value of α is 0.01. For the three pseudorandom sequences $key1, key2$ and $key3$, we take 100 groups key stream with the length of 10^6 bits to count the pass rate of SP800 test, and the result is shown in Table 1.

From Table 1 we can know that the pseudorandom sequence generator proposed in this paper can generate the key stream with good randomness.

2.2 The diffusion algorithm of image encryption based on hyper-chaos

According to the independence of color image’s channel, this paper processes the three channels of image pixel respectively. The specific structure of image encryption is shown in Fig. 2:

The pseudorandom sequence generator proposed in Sect. 2.1 is used to generate three key streams $key1, key2$ and $key3$. The concrete steps of encryption are as follows:

- (1) Divide the channel.

Load the original image P and divide it into three channels P_R, P_G and P_B . In the encryption, the pixel values of three channels are processed separately.

- (2) Perform diffusion algorithm with row-major.

Diffuse image P using the row-major diffusion. Take an example of P_R , the process of row-major diffusion is shown in Fig. 3a. We use $key1$ to diffuse P_R and get the cipher image C'_R , the equation of diffusion

Table 1 The SP800 test on key stream

Statistical test	Key1	Key2	Key3
Frequency	100/100	100/100	100/100
Block frequency ($m=12280$)	98/100	100/100	100/100
Cumulative sums (reverse)	100/100	100/100	100/100
Runs	98/100	100/100	98/100
Longest run ($M=10000$, $N=100$)	100/100	99/100	99/100
Rank	100/100	100/100	99/100
FFT	100/100	99/100	99/100
Non-overlapping template ($m=9, B=000010011$)	100/100	100/100	100/100
Overlapping template ($m=9$, $M=1032, N=968$)	98/100	99/100	100/100
Universal ($L=7$, $Q=1280, K=141577$)	100/100	100/100	100/100
Approximate entropy ($m=10$)	100/100	100/100	100/100
Random excursions ($x=-3$)	67/69	67/69	67/69
Random excursions variant ($x=5$)	69/69	69/69	69/69
Serial ($m=16, P$ value1)	100/100	100/100	99/100
Serial ($m=16, P$ value2)	100/100	100/100	100/100
Linear complexity ($M=1000$)	98/100	99/100	99/100

$$\begin{cases} C'_R(i, j) = C'_R(i, j - 1) \oplus \text{mod}(P_R(i, j) \\ + \text{key1}((i - 1) \times N + j), 256), j! = 1 \\ C'_R(i, j) = C'_R(i - 1, N) \oplus \text{mod}(P_R(i, j) \\ + \text{key1}((i - 1) \times N + j), 256), i! = 1, j = 1 \\ C'_R(i, j) = P_R(M, N) \oplus \text{mod}(P_R(i, j) \\ + \text{key1}((i - 1) \times N + j), 256), i = 1, j = 1 \end{cases} \quad (8)$$

(3) Perform the diffusion algorithm with column major.

Diffuse image C' using the column major diffusion. Take an example of C'_R , the process of column major diffusion is shown in Fig. 3b. In order to save the time of generating key streams, we reuse $key1, key2, key3$. Meanwhile, we need ensure one key stream is not reused in each channel. Therefore, we assign $key2$ to diffuse C'_R , $key3$ to diffuse C'_G and $key1$ to diffuse C'_B . Use Eq. (9) to diffuse C'_R and get the cipher image C''_R . We take the same method for C'_G and C'_B to get the cipher image C''_G and C''_B . The cipher image after column major diffusion is denoted as C'' .

$$\begin{cases} C''_R(i, j) = C''_R(i + 1, j) \oplus \text{mod}(C'_R(i, j) \\ + \text{key2}((i - 1) \times N + j), 256), i! = M \\ C''_R(i, j) = C''_R(1, j + 1) \oplus \text{mod}(C'_R(i, j) \\ + \text{key2}((M - i + 1) + (N - j) \times M), 256), \\ i = M, j! = N \\ C''_R(i, j) = C'_R(1, 1) \oplus \text{mod}(C'_R(i, j) \\ + \text{key2}(1), 256), i = M, j = N \end{cases} \quad (9)$$

is shown in Eq. (8). We take the same method to P_G and P_B with the $key2$ and $key3$. The cipher image after row-major diffusion is denoted as C' .

After finishing all the steps, we can finally get the encrypted image C'' .

Fig. 2 The structure of image encryption

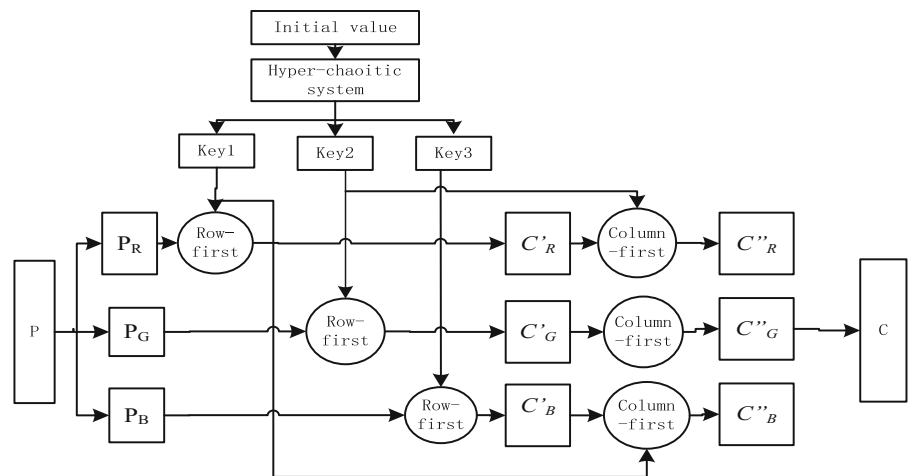
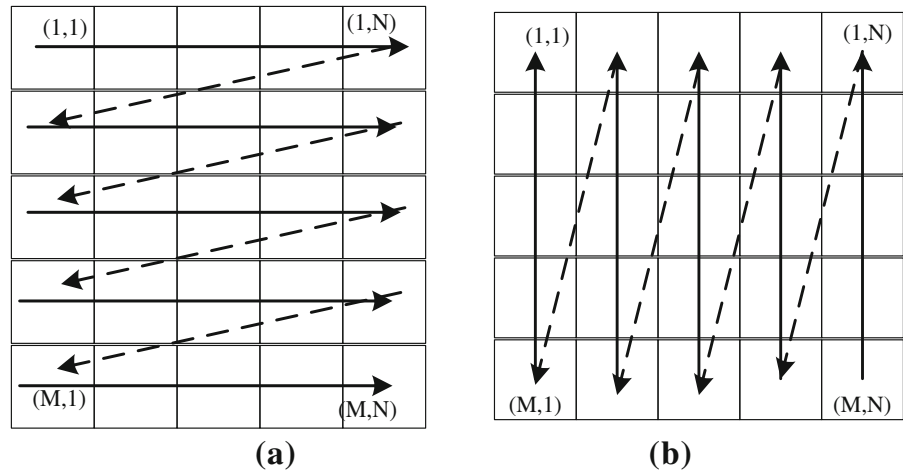


Fig. 3 The diffusion process of encryption **a** the diffusion process of row-major and **b** the diffusion process of column major



2.3 The scrambling algorithm based on the cat map with parameters

In this paper, we use the cat map with parameters to scramble the image. The equation of cat map with parameters is shown in Eq. (10), (x_n, y_n) represents the position of the pixel before scrambling, and (x_{n+1}, y_{n+1}) represents the position of the pixel after scrambling.

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = A \begin{pmatrix} x_n \\ y_n \end{pmatrix} \pmod N$$

where $A = \begin{pmatrix} 1 & a \\ b & ab + 1 \end{pmatrix}$ (10)

Take an example of R Channel, the parameters a_R, b_R used in cat map are shown in Eq. (11). Using the same equation like Eq. (11), we can get the parameters of cat map in G and B Channel scrambling.

$$\begin{cases} a_R = (C''_R(1, 1) \& 0x F0) \gg 4 \\ b_R = C''_R(1, 1) \& 0x 0F \end{cases} \quad (11)$$

One characteristic of cat map is that the value of the first position on the top left corner will not be changed during the scrambling. In order to increase the security of encryption system, we need to hide the value of first pixel of each channel in the cipher image by using the key stream. Take an example of R Channel, the position (i, j) used to hide the first pixel can be calculated by Eq. (12), and then the pixel value of position $(1, 1)$ is interchanged with (i, j) .

$$\begin{cases} i = \text{mod}(\text{key}1(M \times N / (3 \times 2)), M) \\ j = \text{mod}(\text{key}2(M \times N / (3 \times 2)), M) \end{cases} \quad (12)$$

This section is the basis of the third section. In the third section, the encryption algorithm proposed in this section is embedded into the compression, so as to realize compression and encryption for image simultaneously.

3 The design of joint image encryption and compression scheme

The structure of the joint color image encryption and compression algorithm is shown in Fig. 4. The main steps for algorithm are obtaining R, G, B channel of the original image, encrypting and compressing data of each channel and adjusting the data length of each channel. Because the pixel distribution is different for each channel of color image, the compression ratio of each channel maybe is different. Therefore, we need adjust the data length of each channel. The adjusting method is as follows: get the average length of the three-channel data; if the channel length is greater than the average length, we store the extra data in the channel whose length is less than the average height. That realizes the sharing of the channels' space, so as to increase the compression ratio of color image.

For the module of compression and encryption of each channel, the structure is shown in Fig. 5. From Fig. 5, the algorithm in the module of compression and encryption of each channel takes the DCT dictionary to sparsely represent the image, embeds the encryption algorithm into the process of the compression from three aspects: (1) scramble the position information of the effective coefficients, which can destroy the rela-

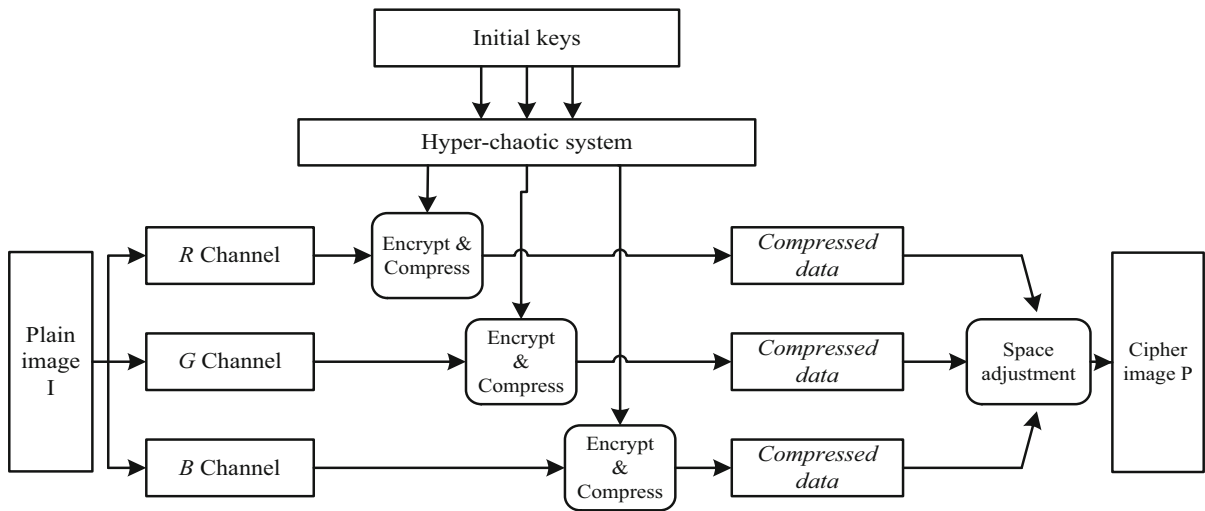
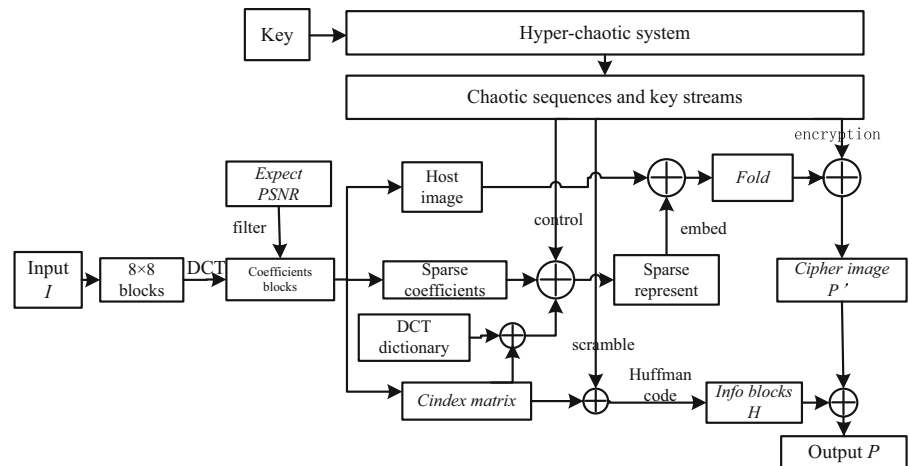


Fig. 4 The structure of color image compression and encryption

Fig. 5 The structure of image compression and encryption on single channel



relationship between atoms in DCT dictionary and effective coefficients. So the algorithm can eliminate relativity among pixels in the wrong reconstructed image blocks. It can also eliminate the vulnerability by which we can get the outline information with a wrong key. (2) Use hyper-chaotic system to control the coefficients transformation to encrypt the coefficients. (3) Use the diffusion algorithm to encrypt the folded image, but not encrypt the information blocks, which can guarantee the security of image and do not destroy the data structure of the information block. Through three ways of encryption, the cipher image has high security.

3.1 The construction of DCT dictionary

Supposing the size of the image is $M \times N$, the concrete steps to generate DCT dictionary are as follows:

- (1) Use the cosine transformation to get the initial orthogonal matrix V , the element $v_{i,j}$ in matrix V can be expressed as Eq. (13). The symbol i and j denote the row and column of elements.

$$v_{i,j} = \cos\left(\frac{\pi(2i-1)(j-1)}{2M}\right) \tag{13}$$

- (2) Normalize the matrix V to get the matrix V' . The element $v'_{i,j}$ can be calculated by Eq. (14).

$$v'_{i,j} = p_j v_{i,j}, \text{ where } p_j = \frac{1}{\|v_{:,j}\|} \tag{14}$$

The symbol p_j , ($j = 1, 2, \dots, N$) is the normalization factor, and its value is the reciprocal of the length of the j th column vector in matrix V .

- (3) Take Kronecker product of matrix V' to get the matrix D whose size is $M^2 \times N^2$, $D = V \otimes V'$, \otimes represents the Kronecker product. Take the matrix A of size $m \times n$ and matrix B of size $p \times q$ as an example, its calculation process is as follows.

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \dots & \dots & \dots & \dots \\ a_{m1}B & a_{m2}B & \dots & a_{mn}B \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} & \dots & a_{11}b_{1q} & a_{12}b_{11} & \dots & a_{12}b_{1q} & \dots & a_{1n}b_{1q} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{11}b_{p1} & \dots & a_{11}b_{pq} & a_{12}b_{p1} & \dots & a_{12}b_{pq} & \dots & a_{1n}b_{pq} \\ a_{21}b_{11} & \dots & a_{21}b_{1q} & a_{22}b_{11} & \dots & a_{22}b_{1q} & \dots & a_{2n}b_{1q} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{21}b_{p1} & \dots & a_{21}b_{pq} & a_{22}b_{p1} & \dots & a_{22}b_{pq} & \dots & a_{2n}b_{pq} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{m1}b_{p1} & \dots & a_{m1}b_{pq} & a_{m2}b_{p1} & \dots & a_{m2}b_{pq} & \dots & a_{mn}b_{pq} \end{pmatrix} \tag{15}$$

Finally we can get the over-complete dictionary D which is used to sparsely represent the image, each column d_l , $l = 1, 2, \dots, N^2$ of D is called atom.

3.2 The relationship between DCT dictionary and DCT

By definition, there is a certain relationship between the DCT dictionary and DCT. Equation (16) is the definition of two-dimensional DCT, and $f(x, y)$ denotes the image pixel value in position (x, y) .

$$F(u, v) = \frac{2}{\sqrt{MN}} c(u)c(v) \sum_{x=1}^M \sum_{y=1}^N f(x, y) \times \cos \frac{(2x-1)u\pi}{2M} \cos \frac{(2y-1)v\pi}{2N} \tag{16}$$

$$\text{where } c(u), c(v) = \begin{cases} 1/\sqrt{2}, & u, v = 1 \\ 1, & \text{others} \end{cases}$$

The inverse DCT is shown in Eq. (17).

$$f(x, y) = \frac{2}{\sqrt{MN}} \sum_{u=1}^{M-1} \sum_{v=1}^{N-1} F(u, v)c(u)c(v) \times \cos \frac{(2x-1)u\pi}{2M} \cos \frac{(2y-1)v\pi}{2N} \tag{17}$$

We represent the two-dimensional image data as a one-dimensional vector $f(\cdot)$. The relationship among DCT coefficients, the original image pixels, and the DCT dictionary D is shown in Eq. (18):

$$f(\cdot) = \sum_{u=1}^M \sum_{v=1}^N F(u, v)d_{(u-1) \times N + v} \tag{18}$$

From Eq.(18) we can know that the DCT coefficients of image are the projection coefficients of image or residuals in the over-complete DCT dictionary. There is a certain relationship between the position of the DCT coefficients and the DCT dictionary atoms column. Combining sparse representation with compression feature of DCT, DCT coefficients carrying less information can be omitted. Finally, we can compress the image by coding some coefficients.

3.3 Properties of orthogonal projection matrix

Using the properties of orthogonal projection matrix, this paper embeds the effective coefficients into the host image to compress the image. In this section, we describe the definition and properties of the orthogonal projection matrix.

Supposing $\varphi \in C^n$ is a sub space, $P \in C^{n,n}$ denotes the orthogonal projection of C^n to φ , when P satisfies the following properties.

(1) $\psi(P) = \varphi$; (2) $P^2 = P$; (3) $P^H = P$. P^H represents the transpose of P matrix. $\psi(P)$ is the value of P space.

When a subspace $\varphi \in C^n$ is given, for any vector $v \in C^n$, there is an orthogonal decomposition on the subspace φ .

$$v = v_1 + v_2 \equiv Pv + (I - P)v \tag{19}$$

where $v_1 \equiv Pv \in \varphi$, $v_2 \equiv (I - P)v \in \varphi^\perp$, φ^\perp represents the orthogonal complement space of φ , I is the unit matrix. We do the following deformation for Eq. (19):

$$Pv = P(v_1 + v_2) \equiv P(Pv + (I - P)v) = P^2v + P(I - P)v = Pv + 0$$

Then we can get the equation like Eq. (20).

$$P(v_1 + v_2) = v_1 \tag{20}$$

From Eq. (19) we know that the projection on $\varphi \in C^n$ of the sum of v_1 and v_2 is v_1 , vector v_1 is in space of $\varphi \in C^n$ and vector v_2 is in space of φ^\perp .

Set φ be r -dimensional subspace, the matrix $D = [d_1, \dots, d_r]$ be orthonormal basis vectors of φ in space of $C^{n \times n}$. Equation (21) is the equation to get the orthogonal projection matrix P .

$$P = DD^H \tag{21}$$

This paper takes the host image as subspace φ , forms the matrix D using the atoms of the DCT dictionary and then uses the matrix $(I - P)$ to transform the sparse coefficients into the orthogonal complement space of host image. Finally the host image can be recovered by Eq. (20).

3.4 The image compression and encryption algorithm

Taking an example of the image I , its size is $M \times N$, the concrete steps of image compression and encryption algorithm are as follows. Figure 6 shows the main procedures of compression and encryption in our proposed scheme.

(1) Divide the image into blocks and make discrete cosine transform on each block. In order to facilitate storing the position information of coefficient, we take 8×8 as the size of block. If the width and length of the image are not a multiple of 8, they should be made up with 0. We mark each block as $I_{x,y}$, $x = 1, \dots, M/8$, $y = 1, \dots, N/8$. For each block $I_{x,y}$, we can get the coefficient block $c_{x,y}$, $x = 1, \dots, M/8$, $y = 1, \dots, N/8$ after discrete cosine transform.

(2) Choose the effective coefficients according to the expected PSNR value. The expected PSNR value represents reconstructed image quality of the user's expectation. The principles of selecting coefficients are as follows.

1) Take the absolute value of the matrix $c_{x,y}$ and sort the elements by column in ascending order. We can get the sorted matrix $c_sort_{x,y}$, and the index matrix $i_sort_{x,y}$ which records the corresponding elements positions of matrix $c_sort_{x,y}$ in $c_{x,y}$.

2) Get the matrix $c_sum_sort_{x,y}$ by calculating the square of the each element in the $c_sort_{x,y}$. Then make the element value of (i, j) equal to the sum of values

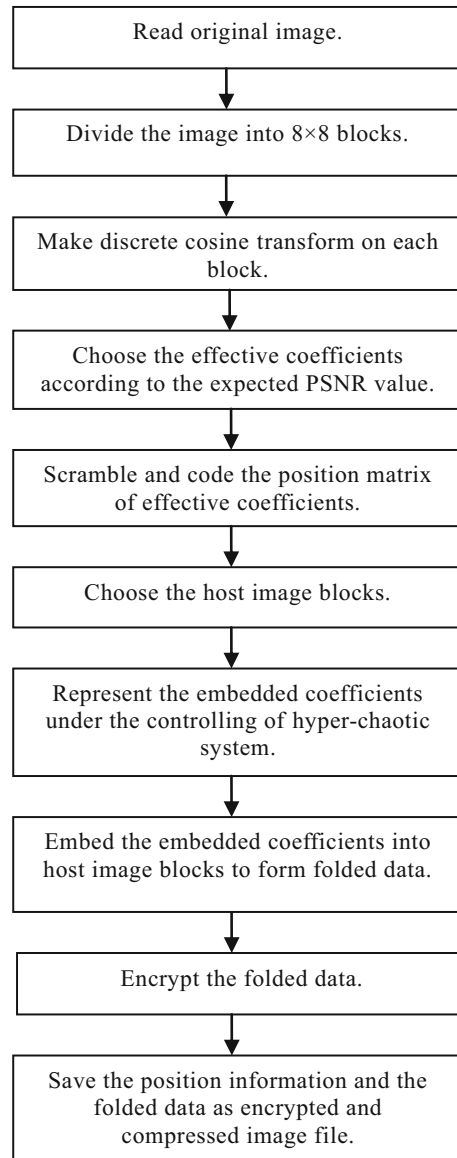


Fig. 6 Procedures of image compression and encryption

of elements contained in the row number from 1 to i of the j th column in the matrix $c_sum_sort_{x,y}$.

3) Compare the elements in the matrix $c_sum_sort_{x,y}$ with the threshold $\Theta = 64 \times (255^2 / (10^{\text{PSNR}} / 10))$. The value PSNR represents the expected PSNR value. If the value is greater than the threshold Θ , reserve the corresponding coefficient in $c_{x,y}$ according to the index matrix $i_sort_{x,y}$. Otherwise, the coefficient is set to 0. Finally we get the matrix $c'_{x,y}$.

(3) Read the nonzero coefficients in $c'_{x,y}$ into an array named as *coefficients*, and calculate the length

content	flag	Row coordinate	Column coordinate	
bits	1bit	3bit	3bit	

Fig. 7 Code rule for the position of nonzero coefficient

of coefficients as *len_coefficients*. We use matrix *CIndex_{x,y}*, $x = 1, \dots, M/8, y = 1, \dots, N/8$ to store the position of the nonzero coefficients.

(4) Scramble the *CIndex* matrix using scrambling algorithm proposed in Sect. 2.3. As we all known, DCT has the property that the outline information of image can be obtained by the high frequency coefficients. According to this property of DCT and its relationship with the DCT dictionary, we can use the position of nonzero coefficients and the DCT dictionary to get outline information of image. In order to improve the security, the *CIndex* matrix is scrambled.

(5) Code the *CIndex* matrix after scrambling. For the 8×8 block matrix, its position can be represented by three bits. So a coefficient’s position can be represented by one byte. The specific code rule for the position of nonzero coefficient is shown in Fig. 7. The seventh bit is the flag used to represent whether the position information belongs to the same block. The fourth to sixth bits are used to store the row coordinate of the coefficients in the block. The first to third bits are used to store the column coordinate of the coefficients in the block. The zeroth bit is idle. Finally, we use Huffman coding to compress the position information.

(6) Choose the host image blocks. The coefficients get by step (2) usually are the large float number. It will need a large space if we store it directly. In this paper we use the method proposed in EIF algorithm¹⁰, and embed the sparse coefficient into the host image blocks through the matrix transformation. The required number of host image blocks is $n = \text{ceil}(\frac{\text{len_coefficients}}{64})$, $\text{ceil}(x)$ returns the value of x to the nearest integers less than or equal to x . Get n $c'_{x,y}$ blocks in row-major order, and do the inverse discrete cosine transform on the n chosen $c'_{x,y}$ blocks to get the host image blocks *Host_{p,q}* ($p = 1, \dots, M/8, q = 1, \dots, 8n/M$) used to embed coefficients.

(7) Represent the embedded coefficients under the controlling of hyper-chaotic system. The following is the detail.

1) For the *Host_{p,q}*, get the related atomic space *D_{p,q}* by *CIndex_{p,q}*, and the number of atoms is *num_{d_{p,q}}*. We can obtain the orthogonal projection matrix *P_{matrix_{p,q}}* by using Eq. (20).

2) Generate a random matrix *Random_{p,q}* with size of $64 \times (64 - \text{num}_{d_{p,q}})$ by the hyper-chaotic system in Eq. (1). In order to reduce the error, the values of the matrix are controlled in $[-0.5, 0.5]$.

3) Use Eq. (22) to represent the embedded coefficients *F_{p,q}*.

$$F_{p,q} = (I - P_{matrix_{p,q}}) \times Random_{p,q} \times coefficients(x + 1 : x + 64 - num_{d_{p,q}}) \tag{22}$$

where $x = (p - 1) \times N \times 8 + (q - 1) \times 64$.

(8) Embed the embedded coefficients into host image blocks. Add the data *F_{p,q}* generated from step (7) to *Host_{p,q}*, and we will get the folded data *Fold_{p,q}*.

(9) Encrypt the folded data. Use Eq. (23) to convert the folded data into integers between 0 and 255 forming *F'*. *F'* is encrypted using the encryption algorithm proposed in Sect. 2.2, and then the cipher data *P'* can be obtained.

$$F'_{p,q}(i, j) = \frac{Fold_{p,q}(i, j)}{Max - Min} \times 255 \tag{23}$$

where the symbol *Max* and *Min* represent the maximum and the minimum value in *Fold* respectively, i, j are row and column value.

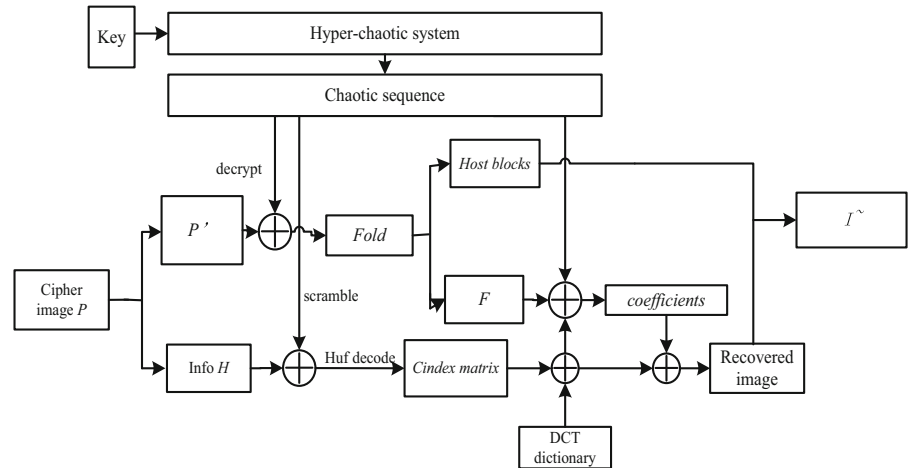
(10) Combine cipher data *P'* and information block, the final encrypted and compressed image *P* can be obtained. The information block includes some information used in decryption, such as position information. The structure of information block is as shown in Fig. 8.

From Fig. 8, *P_{height}* is used to store the length of single channel image data which can be used to recover the data of each channel in color image;

Info data	P_{height}	Len-coefficients	Max	flag	Min	Len_huf	Huf_code	I_{height}	I_{width}	...
bytes	2	3	2	1	2	3	Len_huf	2	2	...

Fig. 8 The coding format for information block

Fig. 9 The structure of image decryption and decompression



$Len_coefficients$ represents length of $coefficients$ in step (3) and the Max and Min are the values in step (9). $Flag$ is used to mark the positive or negative of Min . len_huf represents the length of encoded position information using Huffman coding; Huf_code is the coded position information using Huffman coding; I_height and I_width are used to store the length and width of original plaintext image, and used to judge the correct reconstructed image size in decompression; “...” represents the random number added for data alignment.

3.5 The decompression and decryption algorithm of cipher image

Decryption and decompression algorithm is the reverse process of encryption and compression. The structure of decompression and decryption of single channel data is shown in Fig. 9.

From Fig. 9, the concrete steps for decompression the decryption algorithm are as follows. Figure 10 shows the main procedures of decompression and decryption in our proposed scheme.

(1) Read the header information of P and then separate the information block and the folded data from cipher image.

(2) From the information block, get the information Max , Min and Huf_code required in the decompression process. After decoding the Huffman coding data, we can get the position index matrix, and take reverse scrambling operation to obtain position information $CIndex$ matrix .

(3) Decrypt the folded image. The decryption process is the reverse of diffusion algorithm. Then use

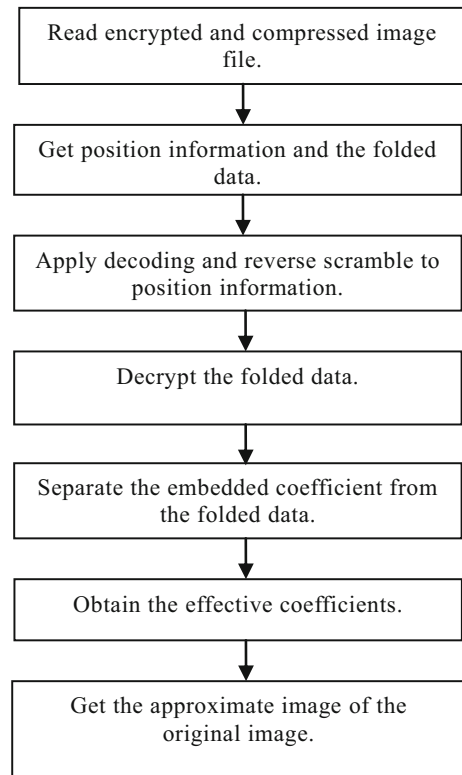


Fig. 10 Procedures of image decompression and decryption algorithm

Eq. (24) to transform the data into the original range, and get the folded data $Fold$.

$$Fold_{p,q}(i, j) = \frac{P'_{p,q}(i, j)}{255} \times (Max - Min) \quad (24)$$

where i, j are row and column value.

(4) Separate the embedded coefficient F from the folded image $Fold$ by Eq. (25).

$$\begin{cases} Host_{p,q} = P_matrix_{p,q} \times Fold_{p,q} \\ F_{p,q} = Fold_{p,q} - Host_{p,q} \end{cases} \quad (25)$$

(5) Obtain the original *coefficients* by Eq. (26).

$$\begin{aligned} &coefficients(x + 1 : x + 64 - num_d_{p,q}) \\ &= ((I - P_matrix_{p,q}) \times Random_{p,q})^{-1} \\ &\quad \times F_{p,q} \end{aligned} \quad (26)$$

The symbol $(.)^{-1}$ represents the inverse matrix.

(6) Through the position matrix $CIndex$, DCT dictionary and the array *coefficients*, we can get the approximate image block $I_{x,y}^{\sim}$, and the calculation equation is shown in Eq. (27):

$$I_{x,y}^{\sim} = \sum_{k=1}^{numel(CIndex_{x,y})} d_l \times coefficients_z \quad (27)$$

The symbol $numel(.)$ denotes the number of the elements in the matrix, the relationship between l and z is shown in Eq. (28).

$$\begin{cases} z = \sum_{i=1}^{x-1} \sum_{j=1}^{N/8} numel(CIndex_{i,j}) \\ \quad + \sum_{j=1}^{y-1} numel(CIndex_{x,j}) \\ l = (CIndex_{(x,y)_k.i_value} - 1) \times N \\ \quad + CIndex_{(x,y)_k.j_value} \end{cases} \quad (28)$$

(7) Finally, we get the approximate image I^{\sim} of the original I .

4 The experimental results and performance analysis

For the new compression and encryption scheme, we conducted the security test and compression performance test. The compression and encryption is implemented in MATLAB 2012b running on a personal computer with 2.8 GHz CPU and 2 GB memory.

4.1 Experimental results

Due to the different distribution of the image pixels, the compression performances of different images are

different. In this paper, we take several standard test images to illustrate the performance of compression and encryption algorithm. The expected signal-to-noise ratio (PSNR) represents the quality of reconstructed image users expect. Take an example of color image Lena with size of 1024×1024 , we set the initial values to be $[0,1,0,2]$, and the results of image compression and encryption under different parameters are shown in Fig. 11.

Figure 11 shows cipher images with different expected PSNR values of the color image Lena with size 1024×1024 . Taking an example of the cipher image with expected PSNR 30, Fig. 12 shows the decompression and decryption image at the initial values $[0,1,0,2]$ and $[0,1,0,2+1-14e]$. The PSNR value of Fig. 12a is 35.27, and the PSNR value of Fig. 10b is -3.66 .

Figure 13 shows cipher images with different expected PSNR values of the color image pepper with size 512×512 . Taking an example of the cipher image with expected PSNR 30, Fig. 14 shows the decompression and decryption image at the initial values $[0,1,0,2]$ and $[0,1,0,2+1-14e]$. The PSNR value of Fig. 14a is 31.79, and the PSNR value of Fig. 14b is -4.49 .

4.2 Performance analysis of image compression and encryption algorithm

4.2.1 Compression ratio and PSNR test

The compression ratio CR is computed by Eq. (29), I_height and I_width denote the height and width of the original image, respectively. And C_height and C_width denote the height and width of the cipher image, respectively.

$$CR = \frac{I_height \times I_width}{C_height \times C_width} \quad (29)$$

The peak PSNR is calculated by Eq. (30), and it is usually used to judge the quality of reconstructed image after image processing. The higher the quality is, the larger PSNR value is. $f_{i,j}$ in Eq. (30) denotes the original pixel value in position (i, j) , and $\hat{f}_{i,j}$ denotes the processed pixel value in position (i, j) .

Fig. 11 The cipher images of Lena 1024×1024 with different parameters: **a** the original image of Lena and **b** the cipher image with expected PSNR=45 and **c** the cipher image with expected PSNR=40 and **d** the cipher image with expected PSNR=35 and **e** the cipher image with expected PSNR=30 and **f** the cipher image with expected PSNR=25

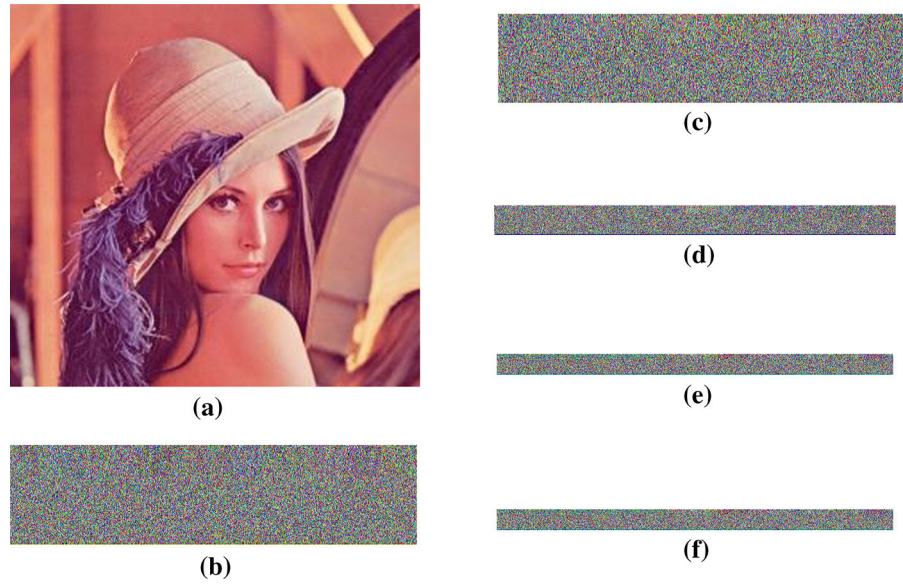
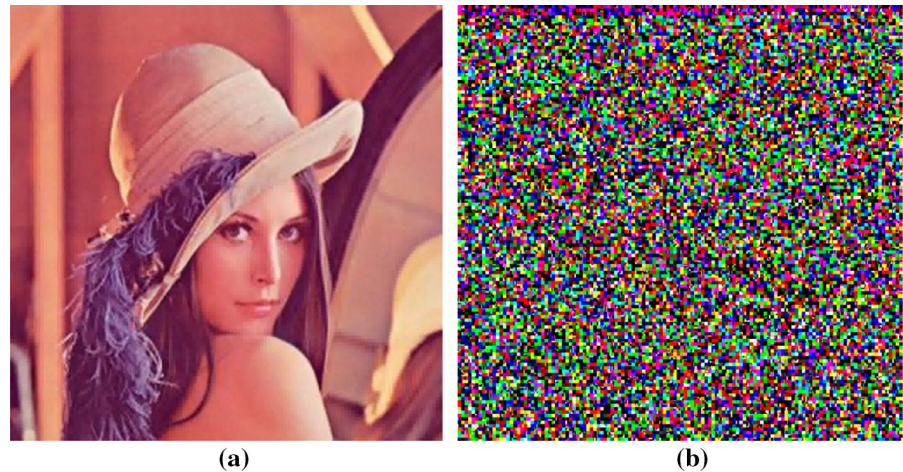


Fig. 12 The decompression and decryption images of Lena cipher image: **a** the decompression and decryption image with [0, 1, 0, 2] and **b** the decompression and decryption image with [0, 1, 0, 2+1e-14]



$$PSNR = 10 \times \log_{10}$$

$$\left[\left(\frac{255^2}{\frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (f_{i,j} - \hat{f}_{i,j})^2} \right) \right] \quad (30)$$

Table 2 lists the compression ratio and the PSNR values for different images with different parameters.

From Table 2, we can know that the compression ratio of images is different for different pixels distribution. We can also know that, if the compression ratio is higher, the quality is lower. Table 2 results reflect the flexibility of the algorithm proposed in this paper. And

we can compress and encrypt image according to the different requirement.

4.2.2 MSSIM test

Structural similarity (SSIM) [31] index is a novel method for measuring the similarity between the two images aiming at improving the traditional metrics such as PSNR by considering human visual system (HVS). SSIM index is defined in Eq. (31).

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (31)$$

Fig. 13 The cipher images of pepper 512×512 with different parameters: **a** the original image of pepper and **b** the cipher image with expected PSNR=45 and **c** the cipher image with expected PSNR=40 and **d** the cipher image with expected PSNR=35 and **e** the cipher image with expected PSNR=30 and **f** the cipher image with expected PSNR=25

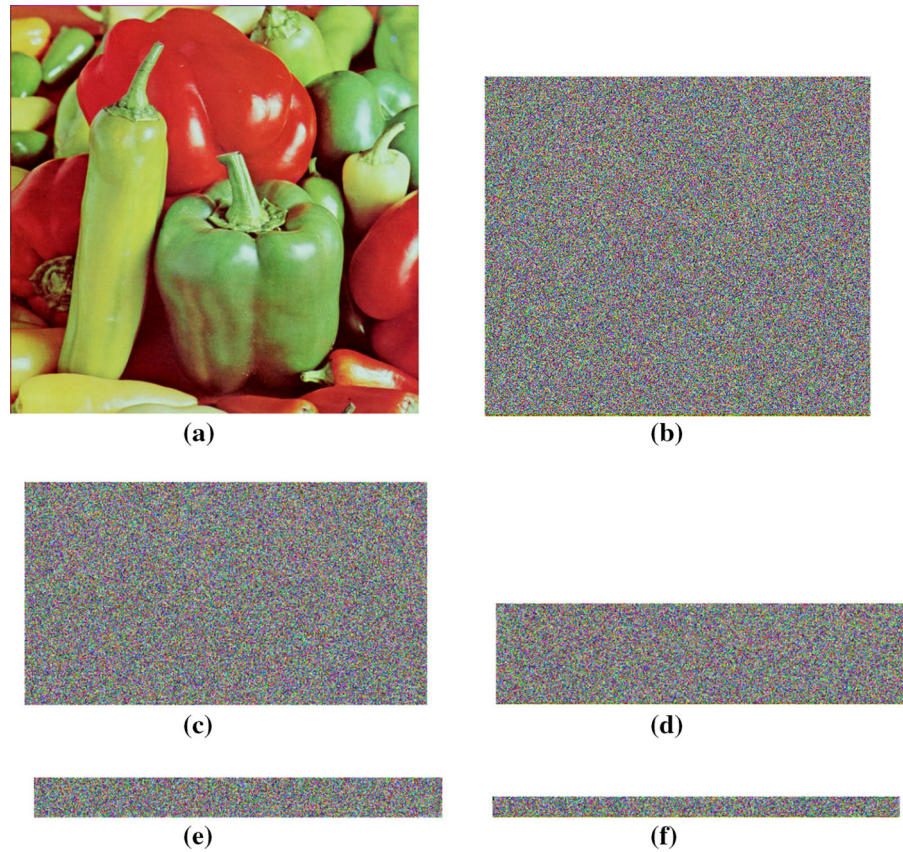
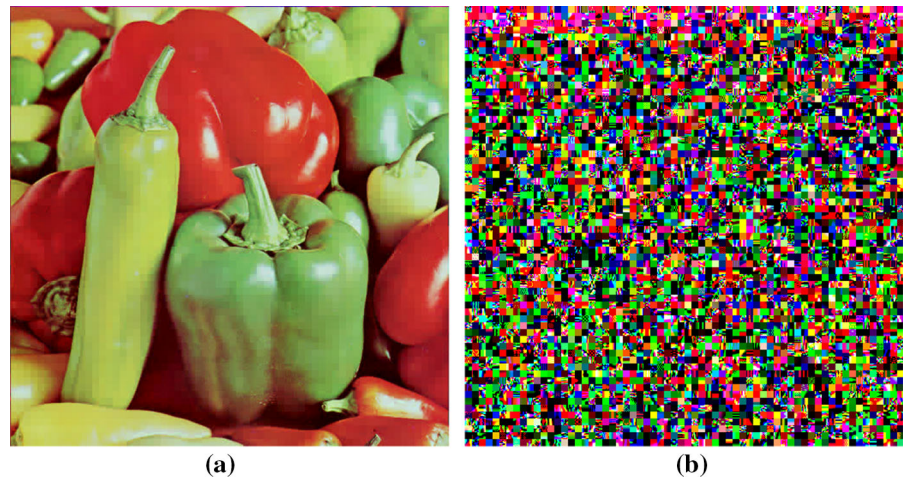


Fig. 14 The decompression and decryption image of pepper cipher image: **a** the decompression and decryption image with $[0, 1, 0, 2]$ and **b** the decompression and decryption image with $[0, 1, 0, 2+1e-14]$



In Eq. (31), x and y represent the windows of two images of size $m \times m$, μ_x and μ_y represent the average values of x and y , σ_x^2 and σ_y^2 represent the variance of x and y , respectively, σ_{xy} represents the covariance of x and y . From Ref. [16], $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$, $k_1 = 0.01$, $k_2 = 0.03$, L is the gray level of pixel value.

In this paper, we use the mean SSIM (MSSIM) as a supplement to evaluate the decompressed image quality. MSSIM is defined in Eq. (32). Table 3 lists the MSSIM results for different images.

$$\text{MSSIM}(X, Y) = \frac{1}{n} \sum_{i=1}^n \text{SSIM}(x_i, y_i) \quad (32)$$

Table 2 The CR and PSNR values for different images with different parameters

Expected PSNR	Lena 1024 × 1024		Pepper 512 × 512		Baboon 512 × 512		Barbara 720 × 576	
	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR
45	10.34	40.72	1.13	41.32	0.81	43.12	1.35	42.23
40	15.05	39.73	1.81	36.56	0.98	38.48	2.18	38.16
35	23.27	38.03	4.06	32.54	1.32	31.57	3.6	34.77
30	33.03	35.27	10.44	31.79	2.14	29.95	7.32	31.53
25	36.57	27.56	19.69	29.05	4.53	28.72	12	27.69

Table 3 The MSSIM for different images with different parameters

Expected PSNR	MSSIM											
	Lena 1024 × 1024			Pepper 512 × 512			Baboon 512 × 512			Barbara 720 × 576		
	R	G	B	R	G	B	R	G	B	R	G	B
45	0.954	0.97	0.972	0.955	0.961	0.976	0.979	0.992	0.992	0.979	0.98	0.98
40	0.956	0.964	0.965	0.913	0.910	0.944	0.965	0.981	0.982	0.95	0.957	0.951
35	0.958	0.964	0.965	0.851	0.868	0.873	0.928	0.952	0.957	0.92	0.93	0.917
30	0.91	0.908	0.911	0.744	0.802	0.777	0.861	0.882	0.902	0.873	0.883	0.863
25	0.866	0.865	0.874	0.675	0.731	0.701	0.704	0.752	0.767	0.778	0.794	0.770

Table 4 The compression performance of different algorithms

	Lena 1024 × 1024		Pepper 512 × 512		Baboon 512 × 512		Barbara 720 × 576	
	CR	PSNR	CR	PSNR	CR	PSNR	CR	PSNR
Ref. [1]	4	–	–	–	4	–	4	–
Ref. [8]	32	31.988	8	34.096	8	27.172	–	–
Ref. [10]	13.8	31.46	–	–	4.33	28.56	6.49	31.87
Our scheme	33.0	35.27	4.06	32.54	4.53	28.72	7.32	31.53

where X and Y represent the original image and the reconstructed image, respectively; x_i and y_i are the image contents at the i th local window; n is the number of local windows. The smaller the MSSIM is, the greater the difference of two images is.

From Tables 2 and 3, with the increasing demand for compression ratio, the variation trends of PSNR and MSSIM are identical. This shows that the higher the compression ratio is, the lower the quality of the reconstructed image is.

4.2.3 Compression performance comparison

Table 4 lists the compression performance of different algorithms, where ‘–’ indicates that the compression

performance of images were not analyzed in the references.

The results of the Refs. [1,8] and [10] in Table 4 are all based on gray image compression and encryption algorithm. The algorithm proposed in this paper is based on the color image. For each channel of a color image, the pixels distribution is different. So the image size in each channel of a color image after compression and encryption may not be the same, which will affect the compression performance of the color image. From Table 4, we can know that the compression and encryption algorithm proposed in this paper has a good performance in compression ratio and image quality.

Table 5 The time for different images with different parameters

Image	Time (s)	Expected PSNR				
		45	40	35	30	25
Lena 1024 × 1024	Encryption and compression	38.45	29.78	24.03	20.89	19.00
	Decryption and decompression	28.14	19.69	13.58	9.97	7.26
Pepper 512 × 512	Encryption and compression	40.22	24.12	11.67	4.79	2.39
	Decryption and decompression	30.56	18.14	6.72	3.88	2.05
Baboon 512 × 512	Encryption and compression	67.44	53.29	36.31	25.03	14.67
	Decryption and decompression	61.93	48.66	29.69	20.40	9.32
Barbara 720 × 576	Encryption and compression	63.88	39.05	21.56	19.25	10.27
	Decryption and decompression	52.64	34.98	17.42	12.05	5.33

4.2.4 The performance test in time

Table 5 shows the compression and encryption time, decompression and decryption time for different images at different expected PSNR.

As shown in Table 5, the lower the quality of reconstructed image is, the less the time required is. Image compression and encryption technology not only ensures the security of the image information, but also reduces the amount of network resources during transmission. The time taken in encryption and compression is acceptable.

4.3 Security analysis of image compression and encryption algorithm

(1) The analysis of key space

The key space of the proposed algorithm depends on the chosen chaotic system for encryption. In our algorithm, if the precision of initial values is 10^{-14} , the size of the key space is $10^{14 \times 4}$, which is greater than 2^{186} . It is enough to resist brute-force attack on the basis of existing computing power.

(2) Resistance to statistical attack

In order to prevent the attacker to obtain the characteristics pixels of the image, resistance to statistical analysis is the most basic principle of image encryption algorithm. Being strong against statistical attack means that the cipher must require some random properties [25]. This is shown by a test on the histograms of the cipher image, the correlations of adjacent pixels in the cipher image and information entropy analysis.

1) The histogram analysis

Take color images Lena and pepper for example; when expected PSNR is 30, three-channel pixel histograms are shown in Figs. 15, 16.

Figures 15 and 16 show that the pixels of cipher images can be uniformly distributed in $[0, 255]$. The pixels distribution has nothing to do with the original image, which can effectively resist the statistical analysis.

2) Correlation of two adjacent pixels

For image data, there is strong correlation between adjacent pixels. Therefore, a good encryption algorithm should remove it to improve the resistance against statistical analysis. The calculation of correlation is as follows.

$$C_r = \frac{N \sum_{j=1}^N (x_j y_j) - \left(\sum_{j=1}^N x_j \right) \left(\sum_{j=1}^N y_j \right)}{\left(N \sum_{j=1}^N x_j^2 - \left(\sum_{j=1}^N x_j \right)^2 \right) \left(N \sum_{j=1}^N y_j^2 - \left(\sum_{j=1}^N y_j \right)^2 \right)} \quad (33)$$

When expected PSNR is 40, the test results are shown in Table 6.

Table 6 shows that the adjacent pixels of plain image are highly correlated, but the cipher pixels are little correlated with each other. The data illustrates that our scheme can effectively destroy the correlation of pixels.

Table 7 gives the comparison of correlation coefficient of adjacent pixels.

From Table 7, compared with the results in Refs. [16, 17], our scheme shows superior performance.

Take color images Lena and pepper for example, when expected PSNR is 40, three-channel pixels distributions are shown in Figs. 17, 18.

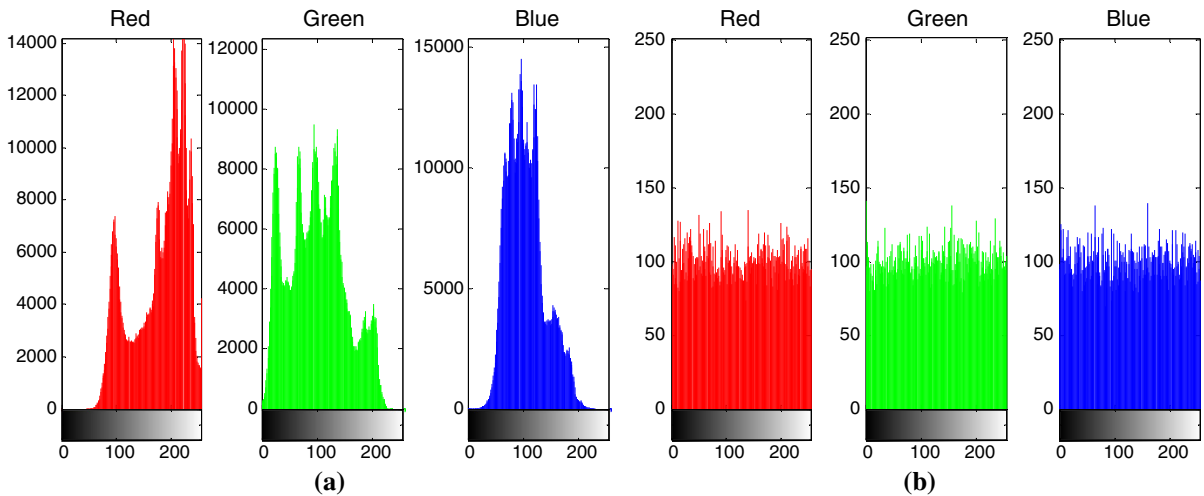


Fig. 15 The histogram of Lena plain image and cipher image: **a** the histogram of Lena and **b** the histogram of Lena cipher image

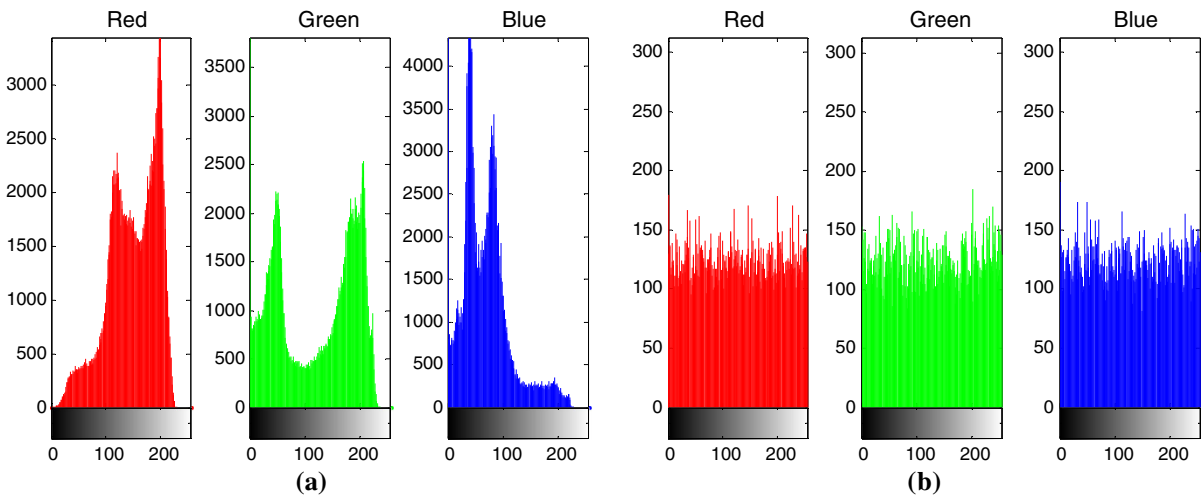


Fig. 16 The histogram of pepper plain image and cipher image: **a** the histogram of pepper and **b** the histogram of pepper cipher image

From Figs. 17, 18, we can know that the pixels distribution of original image is uneven, and there is a strong correlation between pixels. But the correlation between pixels is completely destroyed in the cipher image and we cannot get the pixels information by the adjacent pixels.

3) Information entropy test

The information entropy is an indicator of measuring the random system’s complexity. The higher the entropy is, the more complexity of the system is. The image pixels are treated as random events, we use Eq.(34) to calculate the information entropy of the image.

$$H(S) = \sum_s P(S_i) \log_2 \frac{1}{P(S_i)} \tag{34}$$

From Eq. (34), we can know that the ideal entropy value is 8 if the image pixels can be expressed by 8 bits. Table 8 lists the entropies of cipher images with different parameters.

As is apparent from Table 8, the information entropy of cipher images are closed to the ideal value, which can illustrate that the cipher image has a good randomness. With the compression ratio increasing, the information entropies are decreasing, which accords with the nature of compression and encryption.

Table 6 Correlation coefficient of adjacent pixels

Image		Orientation		
		Horizontal	Vertical	Diagonal
Original image (Lena 1024 × 1024)	R	0.966310	0.998094	0.994114
	G	0.997140	0.998603	0.994749
	B	0.993237	0.995739	0.989763
Encrypted image (Lena 1024 × 1024)	R	−0.010434	0.009557	0.021571
	G	−0.002934	0.012654	0.013535
	B	0.012361	0.011674	0.030442
Original image (Pepper 512 × 512)	R	0.970651	0.965298	0.967087
	G	0.980277	0.983751	0.965162
	B	0.970976	0.967224	0.935720
Encrypted image (Pepper 512 × 512)	R	0.0183476	−0.0255143	0.036132
	G	0.0232030	−0.0137551	0.001106
	B	0.0132531	0.0010002	0.014118
Original image (Baboon 512 × 512)	R	0.9275207	0.864395	0.872386
	G	0.864313	0.782201	0.743004
	B	0.908023	0.889281	0.827124
Encrypted image (Baboon 512 × 512)	R	0.025002	0.016003	0.032376
	G	−0.018205	0.003137	0.017081
	B	−0.026541	0.01543	−0.032159
Original image (Barbara 720 × 576)	R	0.914498	0.969828	0.893957
	G	0.900423	0.963171	0.868453
	B	0.911278	0.961961	0.898419
Encrypted image (Barbara 720 × 576)	R	−0.027834	0.005459	0.012109
	G	0.005493	0.021029	−0.023121
	B	0.020647	0.016501	0.022433

Table 7 Comparison of correlation coefficient of adjacent pixels

Image		Orientation		
		Horizontal	Vertical	Diagonal
Encrypted Lena image (our scheme)	R	−0.010434	0.009557	0.021571
	G	−0.002934	0.012654	0.013535
	B	0.012361	0.011674	0.030442
Encrypted Lena image (Ref. [16])	R	0.040610	−0.013284	0.021483
	G	0.081573	0.033412	−0.017889
	B	0.061240	0.033868	0.018600
Encrypted Lena image (Ref. [17])	R	0.748041	0.771516	0.731441
	G	0.813901	0.727205	0.744201
	B	0.752901	0.716257	0.769612

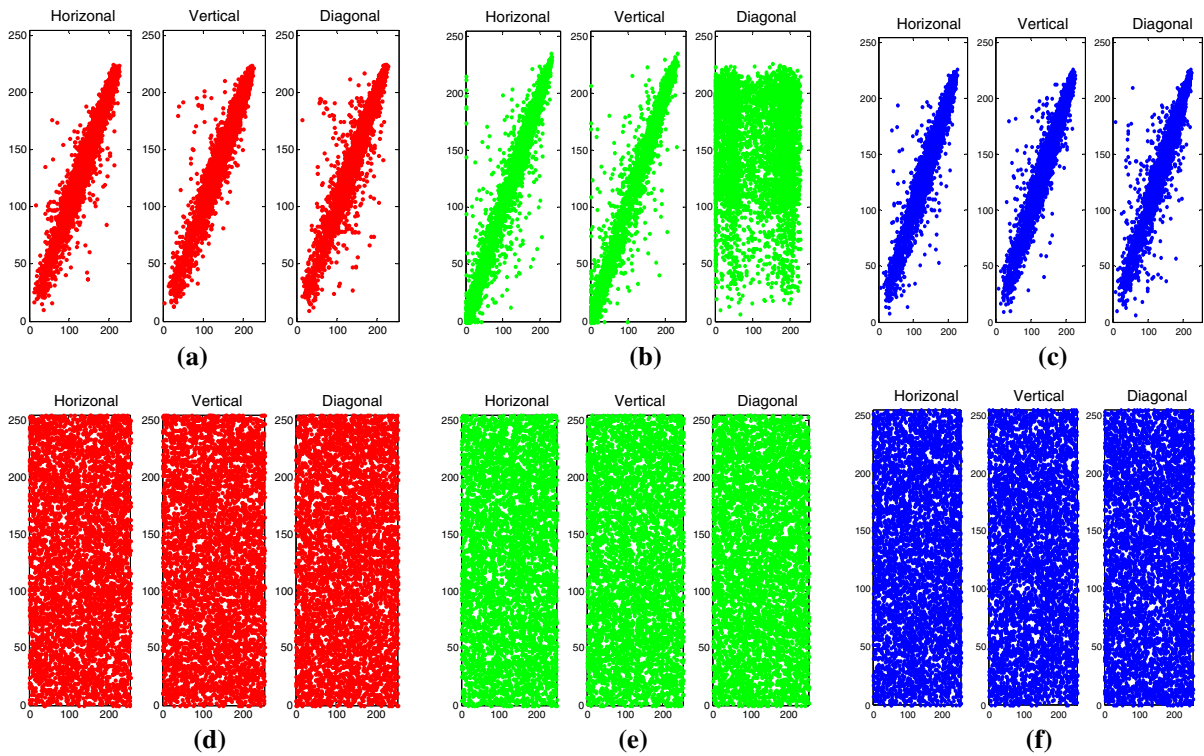


Fig. 17 Pixels distribution of Lena image and the corresponding cipher image: **a** the distribution of original R channel, **b** the distribution of original G channel, **c** the distribution of original B

channel, **d** the distribution of encrypted R channel, **e** the distribution of encrypted G channel and **f** the distribution of encrypted B channel

Table 9 shows the entropy comparison.

From Table 9, the information entropy of our scheme is better than those of the existing algorithms mentioned in Refs. [16, 17].

(3) Avalanche effect

The avalanche effect for image compression and encryption algorithm means the proportion of the bits changed in the cipher image when the initial keys are changed a little. Table 10 lists the avalanche effect of different images. The initial values are [0,1,0,2] and [0,1,0, 2+1e−14]. The symbol ‘-’ represents that the image sizes do not match when the initial keys have a little change, which better explains little change on initial keys has more influence on cipher image.

The ideal value of avalanche effect is 0.5. From Table 10, we can know that the image compression algorithm proposed in this paper has a good avalanche effect.

(4) Resistance to differential attack

Differential attack is one type of classical cryptanalysis methods based on strong characteristics. The

attacker may observe the change of decryption by the tiny change of plaintext to find the correlation between plain image and cipher image. If tiny change of original image can bring great changes to cipher image, the effect of differential attack will be reduced. Generally, we use number of pixels change rate (NPCR) and unified average changing intensity (UACI) to describe this variation. NPCR defined by formula (35) is to obtain the difference between two images by evaluating the number of pixel difference. UACI defined by formula (37) is to obtain the difference between two images by evaluating the change of visual effect.

$$NPCR = \frac{\sum_{i,j} D(i, j)}{W \times H} \times 100 \% \tag{35}$$

$D(i, j)$ is the difference value of the corresponding pixel of the two images. It is defined as:

$$D(i, j) = \begin{cases} 1 & C_1(i, j) \neq C_2(i, j) \\ 0 & C_1(i, j) = C_2(i, j) \end{cases} \tag{36}$$

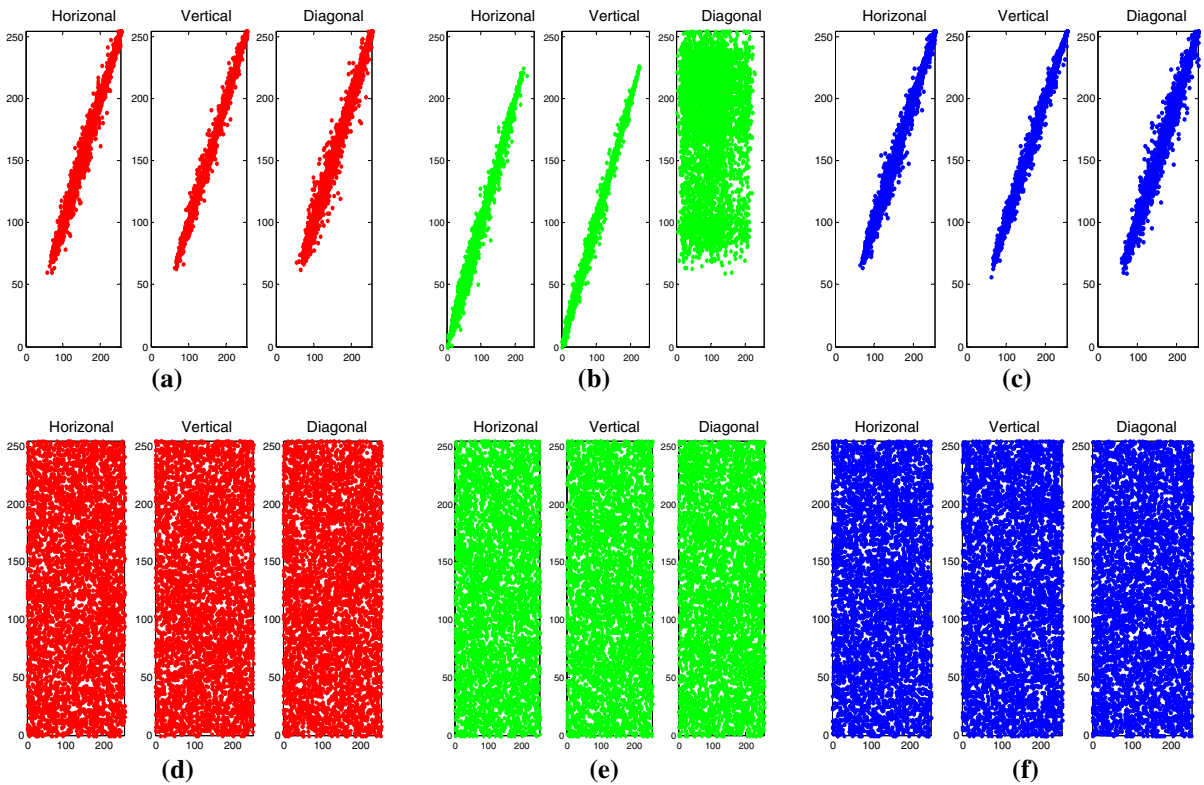


Fig. 18 Pixels distribution of peppers image and the corresponding cipher image: **a** the distribution of original R channel, **b** the distribution of original G channel, **c** the distribution of original B

channel, **d** the distribution of encrypted R channel and **e** the distribution of encrypted G channel, **f** the distribution of encrypted B channel

Table 8 Entropy test

Expected PSNR	Lena 1024 × 1024	Pepper 512 × 512	Baboon 512 × 512	Barbara 720 × 576
Original image	7.03	6.27	7.05	7.58
45	7.997	7.999	7.998	7.998
40	7.997	7.998	7.998	7.997
35	7.980	7.994	7.998	7.998
30	7.982	7.992	7.998	7.998
25	7.975	7.990	7.996	7.995

Table 9 Entropy comparison

Expected PSNR (Lena image)	Our scheme	Ref. [16]	Ref. [17]
45	7.997	6.2731	6.3955
40	7.997	6.1323	6.1757
35	7.980	6.5397	6.5670
30	7.982	6.7740	6.6621
25	7.975	5.9724	6.0465

Table 10 Avalanche effect test

Expected PSNR	Lena 1024 × 1024	Pepper 512 × 512	Baboon 512 × 512	Barbara 720 × 576
45	–	0.4994	–	–
40	0.5001	0.5008	–	–
35	–	–	–	0.4999
30	0.4997	0.4993	–	0.4994
25	–	0.4947	–	–

Table 11 NPCR and UACI value

Image	Lena 1024 × 1024	Pepper 512 × 512	Baboon 512 × 512	Barbara 720 × 576
NPCR	0.9960	0.9933	0.9940	0.9958
UACI	0.3351	0.3315	0.3332	0.3346

Table 12 Sensitivity to key

Image		Lena 1024 × 1024	Pepper 512 × 512	Baboon 512 × 512	Barbara 720 × 576
[1, 1, 0, 1] and [1+1e−14, 1, 0, 1]	NPCR	0.9962	0.9964	0.9943	0.9960
	UACI	0.3377	0.3353	0.3330	0.3341
[1, 1, 0, 1] and [1, 1+1e−14, 0, 1]	NPCR	0.9960	0.9962	0.9954	–
	UACI	0.3349	0.3350	0.3330	–
[1, 1, 0, 1] and [1, 1, 0+1e−14, 1]	NPCR	–	0.9961	0.9961	–
	UACI	–	0.3346	0.3342	–
[1, 1, 0, 1] and [1, 1, 0, 1+1e−14]	NPCR	–	0.9962	0.9959	0.9960
	UACI	–	0.3352	0.3323	0.3334

where C_1 and C_2 are two encrypted images, whose corresponding original images have only one-pixel difference.

$$UACI = \frac{1}{W \times H} \left[\sum_{i,j} \frac{|C_1(i, j) - C_2(i, j)|}{255} \right] \times 100 \% \tag{37}$$

The theoretical values of NPCR and UACI are 99.6 and 33 %, respectively.

In our test, one bit change is introduced with the other values unchanged to get another image. We encrypt two images when expected PSNR is 40. The NPCR and UACI of the two cipher image are evaluated, and the result is shown in Table 11.

From Table 11, NPCR and UACI values are close to the theoretical values. The algorithm can resist to differential attack.

(5) Sensitivity analysis of key

A good encryption scheme should be sensitive to the key. If it is sensitive to the key, a small change of the key will lead to be very different of the cipher text. Figures 12b and 14b show that the decrypted image with a tiny change (10^{-14}) to the right key. We can find that the tiny modification of 10^{-14} to the key will lead to the failure of the decryption. So our scheme is extremely sensitive to the key.

The sensitivity to the key can also be quantified by NPCR and UACI defined by Eqs. (35) and (37).

The tiny modification of 10^{-14} to the key is used in encryption to get another image. We change four initial values respectively. So there are four key pairs: [1, 1, 0, 1] and [1+1e−14, 1, 0, 1], [1, 1, 0, 1] and [1, 1+1e−14, 0, 1], [1, 1, 0, 1] and [1, 1, 0+1e−14, 1], [1, 1, 0, 1] and [1, 1, 0, 1+1e−14]. Table 12 shows the NPCR and UACI value with a tiny change in keys when expected PSNR is 40.

Table 13 Comparison between Ref. [37] and our pseudorandom sequence generator

Item	Ref. [37] (n represents the length of sequence)	Our pseudorandom sequence generator
Key space	$48n$	2^{186}
Approximate entropy	0.6934	0.6937
Kolmogorov entropy	0.6938	0.6940

The symbol ‘-’ represents that the image sizes do not match when the initial keys have a little change, which better explains little change on initial keys has more influence on cipher image.

Table 12 results show our new schemes have a satisfied performance on sensitivity to key.

(6) Cryptanalysis related to hyper-chaos-based schemes

Recently, many hyper-chaos-based image encryption schemes are proposed. However, due to the failed design of the encryption process, such as permutation or diffusion, many of these algorithms are broken [27, 29, 32–36]. This weakness in the design of image encryption process, not hyper-chaos itself, leads to the insecurity of the cryptosystems. For the design of image encryption process in our proposed scheme, the test results show it has high security.

Reference [37] proposed pseudorandom sequence generator based on the Chen chaotic system. However, it is insecure and Ref. [38] analyzed the security weakness of the proposed generator. The discretization process of the proposed generator, not the chaos system itself, leads to the dependence among output values of a chaotic system that yielded a smaller key space. Moreover, Chen chaotic system is not a hyper-chaotic system. For pseudorandom sequence generator of our proposed scheme, we discretize the hyper-chaotic sequence by getting the decimal part of chaotic sequence and select different bits of sequences to obtain the key sequence, which avoids the dependence among output values of a chaotic system. The comparison of the key space between our pseudorandom sequence generator and the pseudorandom sequence generator of Ref. [37] after broken by Ref. [38] is shown in Table 13. From Table 13, the key space of our pseudorandom sequence generator is larger than that of Ref. [37] after broken by Ref. [38]. The key space of our pseudorandom sequence generator is enough to resist brute-

Table 14 Approximate entropy

Map and system	Logistic map	Chen chaotic system	Our hyper-chaotic system
Approximate entropy	0.6604	0.7402	1.3188

force attack on the basis of existing computing power. In addition, the kolmogorov entropy and approximate entropy of our pseudorandom sequence generator and the pseudorandom sequence generator of Ref. [37] are shown in Table 13. Kolmogorov entropy can be used to measure the randomness of binary sequence. The higher the kolmogorov entropy value, the better the randomness. Approximate entropy measures the probability of the key sequence to produce the new pattern. The greater probability means the greater approximate entropy and more random sequence.

From Table 13, the kolmogorov entropy and approximate entropy of our pseudorandom sequence generator are better than these of Ref. [37].

The characteristics of hyper-chaos such as nonlinearity and random-like behaviors make hyper-chaotic systems suited to generate pseudorandom sequences. In our scheme, hyper-chaos is used to construct pseudorandom sequence generator. The characteristics of hyper-chaos such as nonlinearity and random-like behaviors make hyper-chaotic systems suited to generate pseudorandom sequences. Compared with the general chaos system, hyper-chaotic system has more complex chaotic behaviors, its pseudorandomness and long-term unpredictability is stronger. The ability to resist many attacks such as phase-space reconstruction attack is better. Table 14 shows the approximate entropy of chaotic sequence about Logistic map, Chen chaotic system and our hyper-chaotic system.

From Table 14, the approximate entropy of our hyper-chaotic system is larger than that of Logistic map and Chen chaotic system. It shows the superiority of hyper-chaos.

In the same conditions, the hyper-chaos-based cryptosystem has bigger key space. Therefore, in general, hyper-chaos can contribute to the improvement of the security of the key stream produced by the pseudorandom sequence generators. If hyper-chaos-based key stream can be used effectively, it can help to improve the security of image encryption algorithm.

5 Conclusion

In this paper, the image is compressed by folding the color image using the nature of the orthogonal projection matrix and the relationship among image data, DCT dictionary and DCT. The diffusion and scrambling image encryption algorithm based on the hyper-chaotic system is designed to encrypt the image. In the process of the folded compression, the image is encrypted from three aspects: Firstly, a scrambling method for the position of effective coefficients is proposed to reach the goal of eliminating the correlation between pixels in one block. Then in order to encrypt the coefficients, the sparse coefficients are controlled by the hyper-chaotic system, and finally, a diffusion method for the folded image is proposed to encrypt the final data. The proposed algorithm can compress and encrypt the image based on the requirements of the quality of reconstructed images, with some flexibility. The key space is large enough to resist brute-force attacks. The test results show that the proposed compression and encryption algorithm has high security and good performance in compression.

Acknowledgments This work was supported by the National Natural Science Foundation of China (60973162), the Natural Science Foundation of Shandong Province of China (ZR2009GM037, ZR2014FM026), the Science and Technology of Shandong Province, China (2013GGX10129, 2010GGX10132, 2012GGX10110), the Soft Science of Shandong Province, China (2012RKA10009), the National Cryptology Development Foundation of China (No. MMJJ201301006), Foundation of Science and Technology on Information Assurance Laboratory (No. KJ-14-005), and the Engineering Technology and Research Center of Weihai Information Security.

References

- Zhu, H., Zhao, C., Zhang, X.: A novel image encryption-compression scheme using hyper-chaos and Chinese remainder theorem. *Signal Process. Image Commun.* **28**(6), 670–680 (2013)
- Al-Maadeed, S., Al-Ali, A., Abdalla, T.: A new chaos-based image-encryption and compression algorithm. *J. Electr. Comput. Eng.* **2012**, 1–11 (2012)
- Lian, S.G.: Efficient image or video encryption based on spatiotemporal chaos system. *Chaos Solitons Fractals* **40**(5), 2509–2519 (2009)
- Vancea, F., Vancea, C., Borda, M.: Improving an encryption scheme adapted to JPEG stream encoding. In: *The 7th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services 1*, pp. 201–204 (2005)
- Yuen, C.H., Wong, K.W.: A chaos-based joint image compression and encryption scheme using DCT and SHA-1. *Appl. Soft Comput.* **11**(8), 5092–5098 (2011)
- Wu, C.P., Kuo, C.C.J.: Design of integrated multimedia compression and encryption systems. *IEEE Trans. Multimed.* **7**(5), 828–839 (2005)
- Tong, X.J., Wang, Z., Zhang, M., et al.: A new algorithm of the combination of image compression and encryption technology based on cross chaotic map. *Nonlinear Dyn.* **72**(1–2), 229–241 (2013)
- Chan, K.S., Fekri, F.: A block cipher cryptosystem using wavelet transforms over finite fields. *IEEE Trans. Signal Process.* **52**(10), 2975–2991 (2004)
- Yang, H.Q., Liao, X.F., Wong, K.W.: SPIHT-based joint image compression and encryption. *Acta Phys. Sin.* **61**(4), 29–36 (2012)
- Wu, C.P., Kuo, C.C.J.: Design of integrated multimedia compression and encryption systems. *IEEE Trans. Multimed.* **7**(5), 828–839 (2005)
- Wen, J., Kim, H., Villasenor, J.: Binary arithmetic coding with key based interval splitting. *IEEE Signal Process. Lett.* **13**(2), 69–72 (2006)
- Caili, D., Wanyu, B.: Logistic map controlled secure arithmetic coding and its application in image encryption. *J. Chongqing Univ.* **35**(8), 87–91 (2012)
- Zhou, J., Liang, Z., Chen, Y., Au, O.C.: Security analysis of multimedia encryption schemes based on multiple Huffman table. *IEEE Signal Process. Lett.* **14**(3), 20–20 (2007)
- Jakimoski, G., Subbalakshmi, K.: Cryptanalysis of some multimedia encryption schemes. *IEEE Trans. Multimed.* **10**(3), 330–338 (2008)
- Wu, Y.: Research on joint image compression and encryption method. Master thesis, Nanchang University, pp. 25–67 (2013)
- Bowley, J., Rebollo-Neira, L.: Sparsity and something else: an approach to encrypted image folding. *IEEE Signal Process. Lett.* **18**(3), 189–192 (2011)
- Rebollo-Neira, L., Bowley, J., Constantinides, A.G., et al.: Self contained encrypted image folding. *Phys. A Stat. Mech. Its Appl.* **391**(23), 5858–5870 (2012)
- Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. *Int. J. Bifurcat. Chaos* **8**(6), 1259–1284 (1998)
- Wang, X.Y., Teng, L., Qin, X.: A novel colour image encryption algorithm based on chaos. *Signal Process.* **92**, 1101–1108 (2012)
- Matthews, R.: On the derivation of a chaotic encryption algorithm. *Cryptologia* **13**(1), 29–42 (1989)
- Liang-Rui, T., Jing, L., Bing, F.: A new four-dimensional hyperchaotic system and its circuit simulation. *Acta Phys. Sin.* **58**(3), 1446–1455 (2009)
- Liu, H., Wang, X., Kadir, A.: Color image encryption using Choquet fuzzy integral and hyper chaotic system. *Opt.-Int. J. Light Electron Opt.* **124**(18), 3527–3533 (2013)
- Cang, S., Qi, G., Chen, Z.: A four-wing hyper-chaotic attractor and transient chaos generated from a new 4-D quadratic autonomous system. *Nonlinear Dyn.* **59**(3), 515–527 (2010)
- Hong-Yan, J., Zeng-Qiang, C., Zhu-Zhi, Y.: Generation and circuit implementation of a large range hyper-chaotic system. *Acta Phys. Sin.* **58**(7), 4469–4476 (2009)

25. Norouzi, B., Mirzakuchaki, S.: A fast color image encryption algorithm based on hyper-chaotic systems. *Nonlinear Dyn.* **78**, 995–1015 (2014)
26. Norouzi, B., Mirzakuchaki, S., Seyedzadeh, S.M., Mosavi, M.R.: A simple, sensitive and secure image encryption algorithm based on hyper-chaotic system with only one round diffusion. *Multimed. Tools Appl.* **73**(3), 1469–1497 (2014)
27. Zhang, Y., Xiao, D., Wen, W., Li, M.: Breaking an image encryption algorithm based on hyper-chaotic system with only one round diffusion process. *Nonlinear Dyn.* **76**, 1645–1650 (2014)
28. Zhu, C.: A novel image encryption scheme based on improved hyperchaotic sequences. *Opt. Commun.* **285**(1), 29–37 (2012)
29. Li, C., Liu, Y., Xie, T., Chen, M.Z.Q.: Breaking a novel image encryption scheme based on improved hyperchaotic sequences. *Nonlinear Dyn.* **73**, 2083–2089 (2013)
30. <http://csrc.nist.gov/publications/nistpubs/800-22-rev1a/SP800-22rev1a.pdf>. Accessed Jan 2015
31. Wang, Z., Bovik, A.C., Sheikh, H.R., et al.: Image quality assessment: from error visibility to structural similarity. *Image Process. IEEE Trans.* **13**(4), 600–612 (2004)
32. Özkaynak, F., Özer, A.B., Yavuz, S.: Cryptanalysis of a novel image encryption scheme based on improved hyperchaotic sequences. *Opt. Commun.* **285**(24), 4946–4948 (2012)
33. Zhang, Y., Wen, W., Su, M., Li, M.: Cryptanalyzing a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Opt.-Int. J. Light Electron Opt.* **125**(4), 1562–1564 (2014)
34. Xie, T., Liu, Y.S., Tang, J.: Breaking a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Opt.-Int. J. Light Electron Opt.* **125**(24), 7166–7169 (2014)
35. Rhouma, R., Belghith, S.: Cryptanalysis of a new image encryption algorithm based on hyper-chaos. *Phys. Lett. A.* **372**(38), 5973–5978 (2008)
36. Jeng, F.G., Huang, W.L., Chen, T.H.: Cryptanalysis and improvement of two hyper-chaos-based image encryption schemes. *Signal Process.-Image.* **34**, 45–51 (2015)
37. HanPing, H., Liu, L.F., Ding, N.D.: Pseudorandom sequence generator based on the Chen chaotic system. *Comput. Phys. Commun.* **184**, 765–768 (2013)
38. Özkaynaka, F., Yavuzb, S.: Security problems for a pseudorandom sequence generator based on the Chen chaotic system. *Comput. Phys. Commun.* **184**, 2178–2181 (2013)