



# Cloud-decryption-assisted image compression and encryption based on compressed sensing

Jiangyu Fu<sup>1</sup> · Zhihua Gan<sup>2</sup> · Xiuli Chai<sup>1,3</sup> · Yang Lu<sup>1,3</sup>

Received: 14 December 2020 / Revised: 22 March 2021 / Accepted: 9 February 2022 /  
Published online: 7 March 2022

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

## Abstract

In this paper, we propose a new image compression and encryption scheme (ICES) based on compressed sensing (CS) and double random phase encoding (DRPE), as well as a joint decryption on the cloud and user side. In the encryption, the plain image is firstly decomposed into approximate component and detail components by discrete wavelet transform (DWT), then the approximate component is scrambled, and detail components are compressed using a measurement matrix constructed by the Logistic-tent system. Secondly, the scrambled approximate component and compressed detail components are combined into a complex matrix, and then it is subjected to DRPE to get the amplitude and phase matrices. Subsequently, the resulting matrices are performed random pixel scrambling and diffusion to obtain the final cipher image. During the decryption, the cipher image is first partially decrypted on the cloud, and then fully decrypted to recover the original image on the user side, and we can judge whether the cloud has cheated us by comparing the contour similarity between the approximate component and the detail component returned by the cloud to the user, which not only significantly shortens the decryption time, but also effectively prevents malicious deception of the cloud. Experimental results and performance analyses demonstrate its effectiveness and efficiency.

**Keywords** Compressed sensing · Cloud · Image compression and encryption schemes · Double random phase encoding

---

✉ Zhihua Gan  
gzh@henu.edu.cn

✉ Yang Lu  
lyhenu@126.com

<sup>1</sup> School of Artificial Intelligence, Henan University, Kaifeng 475004, China

<sup>2</sup> School of Software, Intelligent Data Processing Engineering Research Center of Henan Province, Institute of Intelligent Network System, Henan University, Kaifeng 475004, China

<sup>3</sup> Henan Key Laboratory of Big Data Analysis and Processing, Henan University, Kaifeng 475004, China

## 1 Introduction

With the rapid development of the Internet and information processing technology, more and more video and image information is transmitted and stored over the network [57]. However, the Internet is an open and shared platform. The transmission and storage of multimedia information through the Internet may have many security risks, especially in military, medical, and commercial fields. Encrypting images is an effective way to protect them [53].

In early times, data encryption was mainly proposed for texts. Such as Anand et al. [1] proposed a symmetric key cryptography encryption algorithm for data security and Jha et al. [26] presented a data security encryption algorithm based on matrix property. When encrypting images, most scholars also use these text encryption techniques to encrypt images. Patel et al. [34] designed a multilevel data encryption algorithm for image and textual information based on AES and RAS, and Arab et al. [3] suggested an image encryption scheme based on chaotic system and AES algorithm. However, many encryption algorithms for texts are not suitable for images, because there are also many differences between images and text. First, the amount of information contained in images is very large. The second is that there is a strong correlation between adjacent pixels of the image. Third, the cipher image allows a certain amount of distortion during decryption, while the text encryption technology does not consider the issue of distortion.

Therefore, more and more scholars put forward new schemes to achieve image encryption. Hua et al. [25] put forward a new color image encryption scheme based on the new two-dimensional logistic tent chaotic system, and Mou et al. [44] proposed an image encryption scheme based on the fractional order hyper-chaotic complex system and Galois field (GF). Based on the good pseudo-randomness and ergodicity of the proposed chaotic system, Zhou et al. [47] suggested an image encryption scheme based on phase truncated short-time fractional Fourier transform and hyper-chaotic system, Ghazvini et al. [19] designed an image encryption scheme based on chaotic mapping and genetic algorithm, and Anwar et al. [2] makes use of the chaos theory to propose an image encryption technique that is based on pixel permutation. The above algorithm effectively protects the security of the image. However, with the increasing size of the image transmitted on the Internet, it is essential to compress the image to improve the efficiency of image transmission and storage while protecting the image security. Therefore, image compression and encryption schemes have attracted more and more attention.

In 2006, D. L. Donoho [17] and E. Candès [5] formally proposed the theory of compressed sensing, and then CS was quickly used in the field of image encryption, which may compress and encrypt the images simultaneously. For example, Chen et al. [11] proposed an ICES based on Kronecker compressed sensing (KCS) and elementary cellular automata (ECA), ECA was used to scramble the sparse image, and KCS was utilized to encrypt and compress the scrambled image. Yao et al. [45] presented a CS-based color image encryption scheme, the plain image was split into R, G, and B three components, and then the obtained components are measured by Gaussian random matrices. Next, the resulting measurement value matrices are scrambled by the multi-image cross pixel scrambling. Chai et al. [7] combined CS and least significant bit (LSB) embedding to propose an effective visual meaningful ICES. After the plain image was compressed and shuffled by CS and zigzag path, the obtained cipher image was embedded into an independent carrier image by a dynamic LSB embedding technology, and finally a visual meaning cipher image was generated, which had the same size as the original image. Luo et al. [31] proposed an image encryption scheme based on Chua's circuit and CS.

After performing CS on the plain image, the obtained cipher image has less dimension than the corresponding original image. When it is transmitted and stored through Internet, transmission bandwidth and storage space may be effectively reduced. However, when recovering the original image from the compressed cipher images on the decryption side, convex optimization computation process of CS reconstruction may require a lot of time, which seriously affects the application of the cryptosystem in real-time fields. In order to solve this problem, some scholars apply the cloud platform with high computing power to manipulate the CS reconstruction [23, 49, 51, 54]. However, since the cloud is semi-trusted, the data information stored in the cloud may be stolen by some criminals, and the cloud sometimes deceives users maliciously. Therefore, some necessary measures should be taken to ensure that our privacy is not leaked and tampered with [28, 30]. For instance, Zhang et al. designed a large image data security transmission and data sharing scheme based on hybrid cloud [49], and then they proposed a CS reconstruction outsourcing scheme based on cloud platform [51]. Besides, Xiao et al. [23] proposed a CS-based image outsourcing reconstruction and identity authentication service in a cloud environment. These schemes ensure the efficiency and secrecy of image data transmission.

In a word, the current CS-based ICES can provide a good compression performance for images, but there also exist some problems. For example, the key used by the algorithms [11, 20, 36, 40, 58] has nothing to do with the plain image, which makes the algorithm vulnerable to chosen-plaintext attacks. Additionally, the schemes [14, 35, 42, 45] have no diffusion operation in the encryption process, resulting in low information entropy of the cipher images. Since CS is a lossy encryption process, the methods [12, 16, 31, 33, 48, 59] lose more data in the compression stage, resulting in poor reconstruction quality. At the same time, the algorithm [51] needs two different samplings to ensure the security of cloud reconstruction, and the methods [4, 23, 49] cannot prevent the malicious deception of the cloud.

To solve the above problems, combining CS with DRPE, we design a secure and efficient image compression-encryption system with cloud-assisted reconstruction. In our scheme, the SHA-256 function of the plain image is utilized to disturb the generation of chaotic key sequences, ensuring that the algorithm and plain image have high correlation. Our main contributions are as follows:

- 1) An ICES based on CS and DRPE is presented. Firstly, the plain image is decomposed by DWT to obtain the approximate component and the detail components. Secondly, the approximate component is scrambled, and the detail components is compressed by CS. Finally, DRPE, scrambling and diffusion are performed on the scrambled approximation component and the compressed measurement values to obtain the cipher image. The combination of CS, DRPE, confusion and diffusion may improve the security level of the proposed ICES while effectively compressing the image. Besides, through scrambling the approximate component containing important information and compressing the detail components containing unimportant information, the volume of the cipher image is less than that of the original image, and the image reconstruction quality is guaranteed.
- 2) A random pixel scrambling method is proposed to shuffle the phase and amplitude matrices of the plain image. The summation of the two matrices is used to decide which elements of these two matrices need to be exchanged, and the index sequence generated by sorting the chaotic sequence is utilized to give the new position for scrambling. The adaptive scrambling process is controlled by the plaintext image and chaotic sequence, which makes the proposed algorithm resistant to statistical attacks and plaintext attacks.

- 3) A joint decryption scheme of cloud platform and user side is designed to shorten the decryption time. In the decryption process, the cipher image is firstly partially decrypted on the cloud. Secondly, the measurement matrix after scrambling is also transmitted to the cloud by the user, and the scrambled sparse coefficient matrix is reconstructed on the cloud. Then it is transmitted to the user and further decrypted to recover the plain image. At the same time, by comparing the contour similarity between the approximate component and the detail component returned by the cloud to the client, we can judge whether the cloud has cheated us. The proposed decryption method may not only enhance decryption efficiency by assisting powerful cloud platform in CS reconstruction, but also prevent the malicious deception of the cloud and protect the security of the image data.

The rest of the paper is organized as follows. The compressed sensing and chaotic system are described in Section 2. Section 3 presents the proposed image encryption system and decryption system. In Section 4, we present the simulation results. The performance and security analysis of the proposed image encryption algorithm is demonstrated in Section 5. Finally, Section 6 concludes the paper.

## 2 Preliminaries

This section mainly introduces the basic knowledge of compressed sensing, and the used Logistic-tent chaotic system and 2D Logistic-adjusted-Sine map.

### 2.1 Compressed sensing

The CS architecture includes the sampling process in the encryption side and reconstruction process in the decryption side. In CS theory, the low-dimensional signal obtained by compression contains enough information of the original signal, and thus the signal can be accurately reconstructed by solving the convex optimization problem [6]. Meanwhile, the number of samples required for signal reconstruction is less than the Nyquist sampling theorem, so that CS has high efficiency.

Assuming that the signal  $x \in \mathcal{R}^n \times 1$  is  $K$ -sparse, it is transformed under a standard orthogonal basis  $\Psi$  and expressed as

$$x = \Psi s \quad (1)$$

where the vector  $s$  has  $K$  ( $K < \langle n \rangle$ ) non-zero coefficients. The signal  $x$  is sampled and compressed by a measurement matrix  $\Phi \in \mathcal{R}^m \times n$  independent of  $\Psi$  and represented as

$$y = \Phi x = \Phi \Psi s = \Theta s \quad (2)$$

where  $y \in \mathcal{R}^m \times 1$  ( $m < \langle n \rangle$ ) is a measurement value vector, and  $\Theta \in \mathcal{R}^m \times n$  is a sensing matrix.

The signal  $x$  may be reconstructed from  $y$  by solving an  $l_0$  optimization problem, illustrated as,

$$\left\{ \min_s \|s\|_0 \quad s.t. \quad \Theta s = y \right. \quad (3)$$

Solving Eq. (3) is a NP-hard problem, and it may be converted to solve the following  $l_1$  norm problem,

$$\left\{ \min_s \|s\|_1 \quad s.t. \quad \Theta s = y \right. \tag{4}$$

The basic model of CS theory mainly includes three major aspects: the sparse representation of original signals, compression measurement and signal reconstruction [55]. The natural image is not sparse in time domain, but it is sparse after it is transformed by DWT or discrete cosine transform (DCT). The measurement matrix needs to meet the restricted isometry property (RIP), which is not related to sparse basis, and Gaussian random matrix and cyclic matrix are mostly utilized [6]. As for reconstruction methods, there are two categories. The first one is greedy iterative algorithms, such as orthogonal matching pursuit (OMP) algorithm, block orthogonal matching pursuit method (BOMP). The second one is convex optimization algorithms, such as base pursuit (BP) algorithm, gradient projection algorithm (GP) and others.

In this paper, the DWT is utilized to sparsity the original image, the random sequence generated by the chaotic system is applied for constructing the measurement matrix, and OMP is used to reconstruct the plain image.

### 2.2 Logistic-tent system

Based on Logistic map and tent map, Logistic-tent system (LTS) [37] is designed and derived by,

$$z_{n+1} = \begin{cases} \left[ \mu z_n(1-z_n) + \frac{(4-\mu)z_n}{2} \right] \text{mod}1, & z_n < 0.5 \\ \left[ \mu z_n(1-z_n) + \frac{(4-\mu)(1-z_n)}{2} \right] \text{mod}1, & z_n \geq 0.5 \end{cases} \tag{5}$$

where control parameter  $\mu \in (0,4]$  and state values  $z_n \in [0, 1]$ . Performance analyses verify that the LTS has better chaotic performance than the Logistic and Tent maps [37]. In the proposed method, the chaotic sequences generated by LTS are applied for constructing the measurement matrix for CS.

### 2.3 2D logistic-adjusted-sine map

The 2D Logistic-adjusted-Sine map (2D-LASM) has extremely complex dynamic behavior and it is defined as [40],

$$\begin{cases} x_{i+1} = \sin(\pi\mu(y_i + 3)x_i(1-x_i)) \\ y_{i+1} = \sin(\pi\mu(x_{i+1} + 3)y_i(1-y_i)) \end{cases} \tag{6}$$

When the system parameter  $\mu \in [0.37, 0.38] \cup [0.4, 0.42] \cup [0.44, 0.93] \cup \{1\}$ , 2D-LASM maps to a chaotic state, and the resulting sequences  $\{(x_n, y_n), n = 0, 1, 2, 3, \dots\}$  are aperiodic, non-convergent and sensitive to initial values. In this article, the chaotic sequences generated by 2D-LASM are utilized to scramble and diffuse the image, and produce the mask matrix used in the DRPE.

## 3 The proposed image encryption and decryption scheme

The proposed image cryptosystem includes the encryption process and decryption process, as illustrated in Fig. 1. In the encryption stage, the plain image is firstly transmitted to the client side, and then compressed and encrypted to the cipher image. Next, the client side sends the

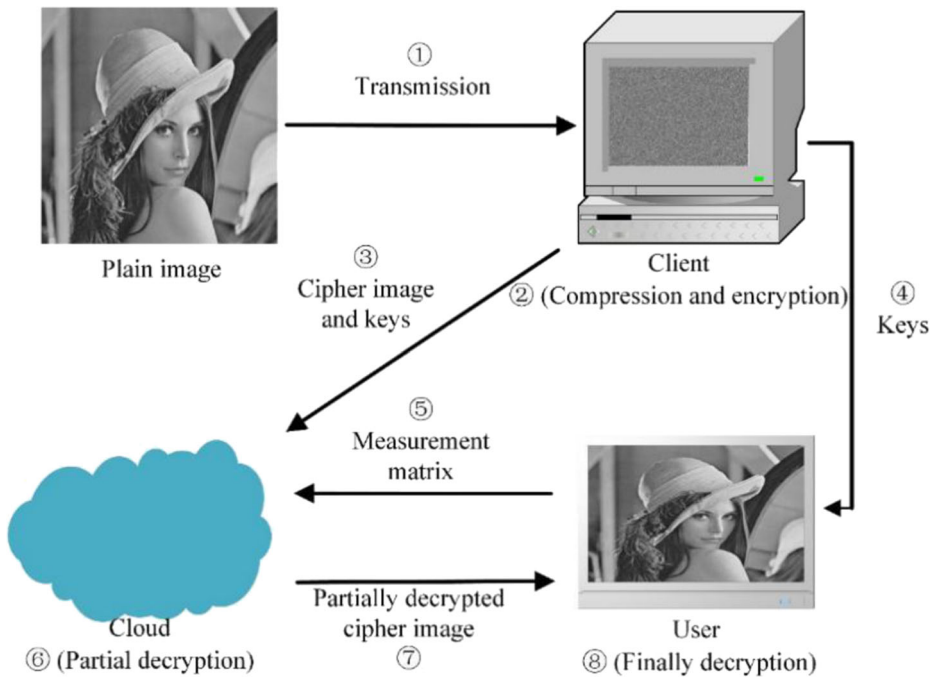


Fig. 1 The framework of the proposed image cryptosystem

cipher image and some keys to the cloud for storage, and also sends other keys to the user for decryption.

In the decryption stage, the user side transmits the measurement matrix to the cloud, and then the cipher image is partially decrypted and reconstructed in the cloud. Subsequently, the resulting image is sent to the user and decrypted by use of the keys to obtain the final plain image. The detailed encryption and decryption steps are as follows.

### 3.1 The proposed ICES

In this section, the proposed ICES will be described in detail, and it is composed of five stages. As shown in Fig. 2, the first stage is to generate the chaotic sequences, some sequences are used for constructing measurement matrices, and others are utilized for DRPE, confusion and diffusion. The second stage is to confuse the approximate component of plain image and to compress detail components by CS. Next, in the third stage, after recombining scrambled approximate component and compressed detail components to a complex matrix, DRPE is performed on the obtained matrix to find the phase and amplitude matrices. Subsequently, the phase and amplitude matrices are quantified and scrambled in the fourth stage, and diffused in the fifth stage to obtain the final cipher image.

Before encryption, the plain image  $P$  with size  $M \times M$  is decomposed into an approximate component  $LL$  and three detail components  $HL$ ,  $LH$ ,  $HH$  by using DWT, the size of each component is  $m \times m$ , and  $m = M/2$ . The detailed encryption steps are given as follows.

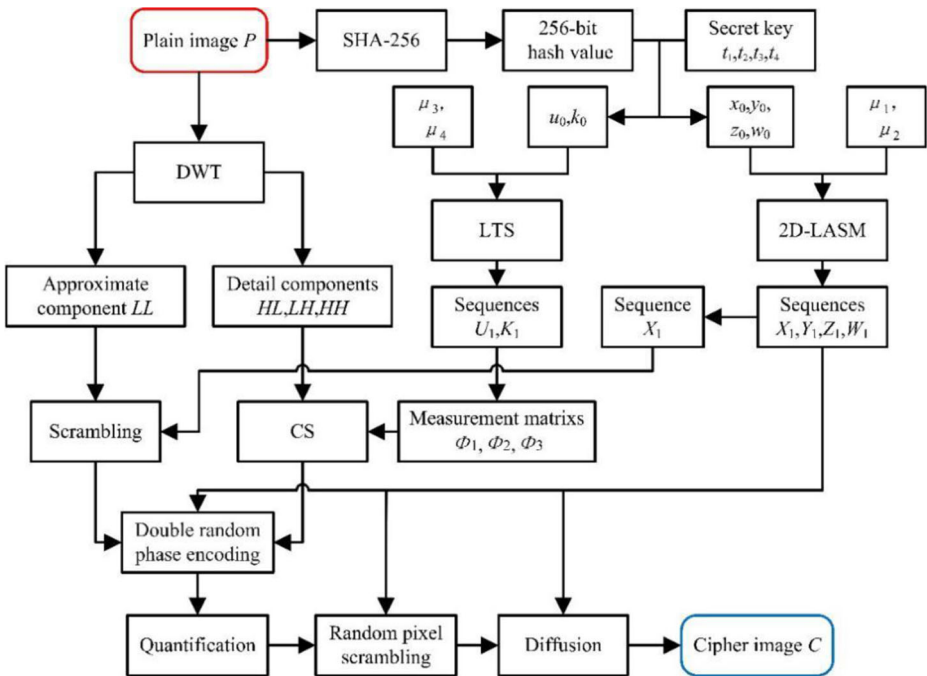


Fig. 2 The flow chart of the proposed ICES

### 3.1.1 The generation of the chaotic sequences

In this paper, the SHA-256 function is used to generate a 256-bit hash key for the plain image  $P (M \times M)$ , which is combined in groups of 8 bits to obtain  $k_i$ , where  $i = 1, 2, \dots, 32$ . Then,  $k_i$  and the external keys  $t_1, t_2, t_3, t_4$  are utilized to generate the initial values  $x_0, y_0, z_0, w_0, u_0, k_0$  of the used chaotic system by the following Eq. (7),

$$\begin{cases} x_0 = ((t_1 \times t_2)/(256 \times t_3 \times t_4))((k_1|k_5) \oplus (k_9|k_{13}) \oplus (k_{17} \& k_{21}) \oplus (k_{25} \& k_{29})) \\ y_0 = ((t_1 \times t_3)/(256 \times t_2 \times t_4))((k_2|k_6 \oplus k_{10} \& k_{14}) \oplus (k_{18}|k_{22} \oplus k_{26} \& k_{30})) \\ z_0 = ((t_1 \times t_2)/256(t_3 \times t_4))((k_3 \oplus k_7 \oplus k_{11} \oplus k_{15})|(k_{19} \oplus k_{23} \oplus k_{27} \oplus k_{31})) \\ w_0 = ((t_1 \times t_3)/256(t_2 \times t_4))(k_4 \oplus k_8 \oplus k_{12} \oplus k_{16} \oplus k_{20} \oplus k_{24} \oplus k_{28} \oplus k_{32}) \\ u_0 = (x_0 + y_0 + z_0 + w_0)/4 \\ k_0 = (x_0 + y_0 + z_0 + w_0 + u_0)/5 \end{cases} \quad (7)$$

where  $x \oplus y$  is the XOR operation of  $x$  and  $y$ ,  $x | y$  is the OR operation of  $x$  and  $y$ ,  $x \& y$  is the AND operation of  $x$  and  $y$ .

Next, the initial values  $x_0, y_0$  and parameter  $\mu_1$  are brought into the 2D-LASM chaotic system to iterate  $m^2 + n_0$  times, wherein,  $n_0 \geq 500$  and  $m = M/2$ . In order to eliminate transient effect, the first  $n_0$  values are discarded to obtain two chaotic sequence  $X_1, Y_1$ . In the same way, bringing the initial values  $z_0, w_0$  and the parameter  $\mu_2$  into the 2D-LASM system to get the chaotic sequences  $Z_1, W_1$ .

Subsequently, iterate the LTS chaotic system for  $m^2 + n_0$  times with the initial values  $u_0$  and parameter  $\mu_3$ , then obtain chaos sequence  $U_1$  after removing the previous  $n_0$  values.

Similarly, the chaotic sequence  $K_1$  is gotten by bringing the initial value  $k_0$  and parameter  $\mu_4$  into the LTS system.

### 3.1.2 Scrambling of approximate component and compression of detail components

The approximate component  $LL$  is shuffled by the chaotic sequence  $X_1$ . In detail,  $LL$  is transformed into 1D sequence  $l_1$  sized of  $1 \times m^2$ , the chaotic sequence  $X_1$  with the size of  $1 \times m^2$  is selected and sorted in ascending order according to Eq. (8), and the index sequence  $s_1$  is obtained. Subsequently, as shown in Eq. (9), the sequence  $l_2$  is obtained by scrambling  $l_1$  with  $s_1$ .

$$\left[ X'_1, s_1 \right] = \text{sort}(X_1) \quad (8)$$

$$l_2(s_1(t)) = l_1(t) \quad (9)$$

where  $1 \leq t \leq m^2$ . Finally,  $l_2$  is converted to a matrix  $LL_1$  with the size of  $m \times m$ .

Next, the measurement matrices are produced by chaotic sequences, and then the detail components of the plain image are compressed and encrypted by CS. The specific steps are as below.

Step 1: The discrete wavelet transform sparse basis  $\psi$  is used to sparsity the three detail components  $HL$ ,  $LH$  and  $HH$ , and three sparse coefficient matrices  $HL_1$ ,  $LH_1$  and  $HH_1$  is obtained by,

$$\begin{cases} HL_1 = \psi HL \psi' \\ LH_1 = \psi LH \psi' \\ HH_1 = \psi HH \psi' \end{cases} \quad (10)$$

where  $\psi'$  denotes the transposition of matrix  $\psi$ .

Step 2: Use two chaotic sequences  $U_1$  and  $K_1$  to construct two random matrices  $\Phi_{01}$  and  $\Phi_{01}$  of size  $m \times m$ , and then recombine them into a non-overlapping random matrix  $\Phi_0$  sized of  $2m \times m$ . At the same time, pick up the sequence  $U_m$  with a length of  $1 \times 2m$  from the sequence  $U_1$ , and then sort  $U_m$  in ascending order to obtain the index sequence  $s_2$ . Next, scramble  $\Phi_0$  by row according to  $s_2$ , and select the first  $m$  rows of the obtained shuffled matrix to produce the measurement matrix  $\Phi$  sized of  $m \times m$ .

Step 3: Divide the measurement matrix  $\Phi$  into three measurement matrices  $\Phi_1$ ,  $\Phi_2$ ,  $\Phi_3$ , their size is  $m_1 \times m$ ,  $m_2 \times m$ ,  $m_3 \times m$ , where  $m_1 = m/2$ ,  $m_2 = m/4$ ,  $m_3 = m/4$ . Then, measure the three sparse coefficient matrices  $HL_1$ ,  $LH_1$  and  $HH_1$  by use of these three measurement matrices to get three measured value matrices  $HL_2$ ,  $LH_2$ ,  $HH_2$ , respectively. The specific process is as follows:

$$\begin{cases} HL_2 = \Phi_1 HL_1 \\ LH_2 = \Phi_2 LH_1 \\ HH_2 = \Phi_3 HH_1 \end{cases} \quad (11)$$



Step 4: The matrices  $HL_2$ ,  $LH_2$  and  $HH_2$  are recombined into a non-overlapping matrix  $H_1$  through the following Eq. (12).

$$H_1 = \begin{bmatrix} HL_2 \\ LH_2 \\ HH_2 \end{bmatrix} \tag{12}$$

### 3.1.3 Double random phase encoding

DRPE has been widely utilized in image encryption for its parallel processing and easy configuration [59]. In this subsection, DRPE is applied for encrypting the approximate component and three detail components of the plain image.

- Step 1: The matrix  $LL_1$  and  $H_1$  are merged into a complex matrix  $LLH_1$  with the size of  $m \times m$ , where  $LLH_1 = LL_1 + j H_1$  and  $j$  is an imaginary unit.
- Step 2: Chaotic sequences  $X_1, Y_1, Z_1, W_1$  are utilized to generate two  $1 \times m^2$  sequences  $XY_0$  and  $ZW_0$ , where  $XY_0 = (X_1 + Y_1) / 2$ ,  $ZW_0 = (Z_1 + W_1) / 2$ .
- Step 3: The sequences  $XY_0$  and  $ZW_0$  are converted to two random matrices  $q_1$  and  $q_2$  with the size of  $m \times m$ , respectively. Then, two random phase mask matrices  $M_1, M_2$  are gotten by

$$M_i = e^{j2\pi q_i(x,y)} \tag{13}$$

where  $i = 1, 2$ , and the size of matrix  $M_i$  is  $m \times m$ .

Step 4: Manipulate the DRPE transform on the matrix  $LLH_1$  to obtain the matrix  $LLH_2$  via,

$$LLH_2 = IFFT(FFT(LLH_1 \cdot M_1) \cdot M_2) \tag{14}$$

where  $FFT()$  and  $IFFT()$  indicate the Fourier transform and inverse Fourier transform, respectively.

Step 5: Extract the phase and amplitude from the matrix  $LLH_2$  to obtain matrices  $LL_2$  and  $H_2$  sized of  $m \times m$ , respectively. The specific operations are as follows.

$$\begin{cases} LL_2 = \text{real}(LLH_2) \\ H_2 = \text{imag}(LLH_2) \end{cases} \tag{15}$$

### 3.1.4 Random pixel scrambling

Before scrambling, the obtained matrices  $LL_2$  and  $H_2$  are quantified to get the matrices  $LL_3$  and  $H_3$ , respectively. Taking  $LL_2$  as an example, its quantization process is as follows:

$$LL_3 = \text{round}\left(\frac{255 \times (LL_2 - \min)}{\max - \min}\right) \tag{16}$$

where  $\min$  is the minimum value of matrix  $LL_2$ ,  $\max$  is the maximum value of matrix  $LL_2$ , and  $\text{round}(x)$  is the integer closest to  $x$ .

Subsequently, the random pixel scrambling is manipulated on the matrices  $LL_3$  and  $H_3$ , and the detailed steps are as below.

- Step 1: Pixel exchanging. Calculate the summation of the two matrices  $LL_3$  and  $H_3$ , judge the parity of the obtained matrix to find a new matrix, when the element is even, it is marked with 0, otherwise it is marked with 1. Next, the resulting matrix is utilized to determine whether the elements at the same position of the  $LL_3$  and  $H_3$  need to be exchanged.
- Step 2: Pixel scrambling. Firstly, select the sequence  $Z_1$  generated by the chaotic system, rearrange the elements of  $Z_1$  in ascending order to obtain a  $1 \times m^2$  index sequence  $s_3$ , and then transform the obtained sequence into an  $m \times m$  index matrix  $S(i, j)$ . Secondly, the element value of the index matrix is used to provide the new position of the pixels to be shuffled.

Through the above operations, the scrambled matrix  $LL_4$  and  $H_4$  are obtained. The specific confusion process may be represented as follows:

$$\text{When } (LL_3(i, j) + H_3(i, j)) \bmod 2 = 1, \text{ then } H_4(s, t) = LL_3(i, j), LL_4(s, t) = H_3(i, j);$$

$$\text{When } (LL_3(i, j) + H_3(i, j)) \bmod 2 = 0, \text{ then } H_4(s, t) = H_3(i, j), LL_4(s, t) = LL_3(i, j).$$

where

$$\begin{cases} s = S(i, j) \bmod m + 1 \\ t = \text{ceil}(S(i, j)/n) \end{cases} \quad (17)$$

$\text{ceil}(x)$  means to return the smallest integer greater than or equal to  $x$ , and  $1 \leq i, j, s, t \leq m$ . Figure 3 shows an example of the random pixel scrambling.

### 3.1.5 Diffusion

This subsection is provided to modify the pixel values of the scrambled matrices  $LL_4$  and  $H_4$  to improve the encryption effect.

- Step 1: Recombine the scrambled matrices  $LL_4$  and  $H_4$  into a non-overlapping matrix  $B_{LH}$  of size  $m \times 2m$ , where  $B_{LH} = [LL_4, H_4]$ , and then convert the  $B_{LH}$  to a  $1 \times 2m^2$  sequence  $B$ .
- Step 2: The chaotic sequences  $X_1, Y_1, Z_1, W_1$  are utilized to generate two random sequences  $XY$  and  $ZW$  with size of  $1 \times 2m^2$ , where  $XY = [X_1, Y_1], ZW = [Z_1, W_1]$ .
- Step 3: According to the following Eq. (18), the sequence  $B$  is diffused using the sequences  $XY$  and  $ZW$  to obtain an encrypted sequence  $C$ , and then the final cipher image is obtained by transforming  $C$  to the matrix.

The first pixel  $C_1$  of the sequence  $C$  is generated by using the first element of the random sequence  $XY, ZW$  and  $B$ . The diffusion process is as follows:

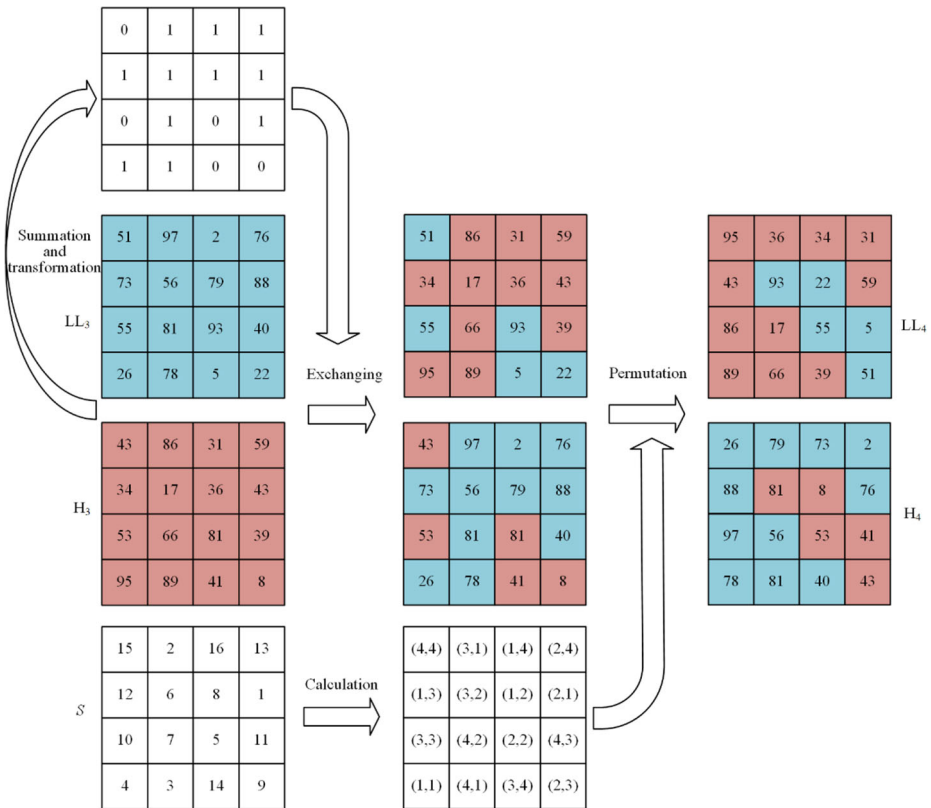


Fig. 3 Illustration of the random pixel scrambling

$$\begin{cases} C_1 = B_1 \oplus XY_1 \oplus ZW_1 \\ C_i = B_i \oplus C_{i-1} \oplus XY_i \oplus ZW_i \end{cases} \quad (18)$$

where  $XY_1, ZW_1, B_1$  and  $C_1$  represent the first element of random sequences  $XY, ZW, B$  and  $C$ , respectively.  $XY_i, ZW_i, B_i$ , and  $C_i$  represent the  $i$ -th element of random sequences  $XY, ZW, B$  and  $C$ , respectively.  $C_{i-1}$  represents the  $(i-1)$ -th element of the sequence  $C$ , and  $i = 2, 3, \dots, 2m^2$ .

### 3.2 The image decryption process

The image decryption is the inverse process of image encryption, and the flow chart is shown in Fig. 4. Before decryption, the client needs to transmit the SHA-256 hash value of the plain image and the external keys  $t_1, t_2, t_3, t_4, \mu_1, \mu_2$  to the cloud, and sends the hash value and external keys  $t_1, t_2, t_3, t_4, \mu_3, \mu_4$  to the user. As described in Section 3.1.1, the chaotic sequences  $X_1, Y_1, Z_1, W_1, U_1$  and  $K_1$  are firstly generated by the secret keys. The specific decryption process consists of the following ten steps.

Step 1: Inverse diffusion. As described in Section 3.1.5, the sequences  $XY, ZW$  are produced by use of the sequences  $X_1, Y_1, Z_1, W_1$ . Then, the sequence  $C$  transformed from the

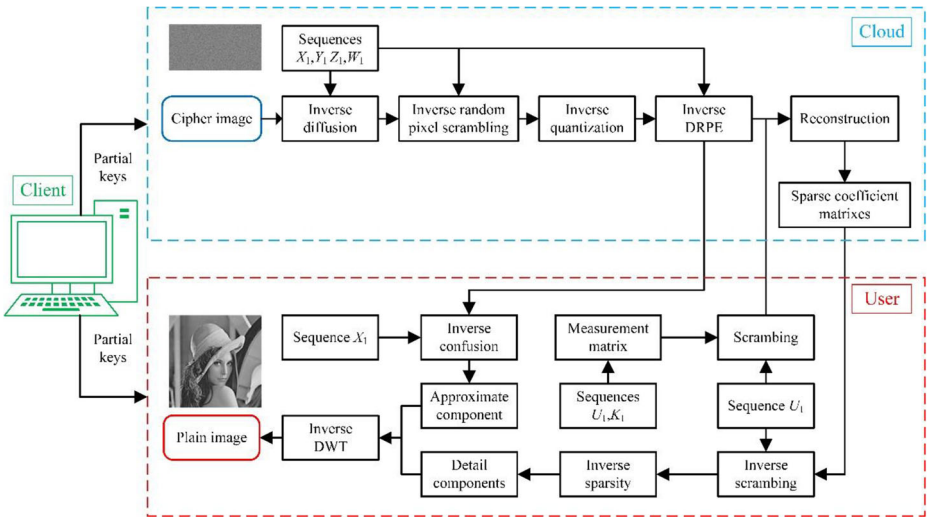


Fig. 4 The flow chart of the image decryption algorithm

cipher image sized of  $m \times 2m$  is diffused by the sequences  $XY$  and  $ZW$  to obtain the sequence  $B$ . The inverse diffusion process is as follows:

$$\begin{cases} B_1 = C_1 \oplus XY_1 \oplus ZW_1 \\ B_i = C_i \oplus B_{i-1} \oplus XY_i \oplus ZW_i \end{cases} \quad (19)$$

where  $i = 2, 3, \dots, 2m^2$ . Subsequently, the sequence  $B$  is converted into a matrix and divided into two non-overlapping matrices  $LL_4$  and  $H_4$  with the size of  $m \times m$ .

Step 2: Inverse random pixel scrambling. According to the following rules, the matrices  $LL_4$  and  $H_4$  are inversely scrambled by the chaotic sequence  $Z_1$  to obtain the matrices  $LL_3$  and  $H_3$ .

When  $(LL_4(i, j) + H_4(i, j)) \bmod 2 = 1$ , then  $H_3(s, t) = LL_4(i, j)$ ,  $LL_3(s, t) = H_4(i, j)$ ;

When  $(LL_4(i, j) + H_4(i, j)) \bmod 2 = 0$ , then  $H_3(s, t) = H_4(i, j)$ ,  $LL_3(s, t) = LL_4(i, j)$ , where

$$\begin{cases} s = S(i, j) \bmod m + 1 \\ t = \text{ceil}(S(i, j)/n) \end{cases} \quad (20)$$

where the  $\text{ceil}(x)$  means to return the smallest integer greater than or equal to  $x$ , and  $1 \leq i, j, s, t \leq m$ .

Step 3: Inverse quantization. The matrices  $LL_3$  and  $H_3$  are inversely quantized to obtain  $LL_2$  and  $H_2$ . Taking  $LL_3$  as an example, its inverse quantization process is as follows:

$$LL_2 = \frac{LL_3 \times (\text{max} - \text{min})}{255} + \text{min} \quad (21)$$

where  $\text{min}$ ,  $\text{max}$  are the minimum and maximum values of the matrix  $LL_2$ , respectively.

Step 4: Inverse double random phase encoding (IDRPE).

- (1) The matrices  $LL_2$  and  $H_2$  are combined into a complex matrix  $LLH_2$  with the size of  $m \times m$ , where  $LLH_2 = LL_2 + j \times H_2$  and  $j$  is an imaginary unit.
- (2) As described in Section 3.1.3, two random sequences  $XY_0$  and  $ZW_0$  are generated by the chaotic sequences  $X_1, Y_1, Z_1, W_1$  and then converted to two random matrices  $q_1$  and  $q_2$ , respectively. Then, two random phase mask matrices  $M'_1, M'_2$  are gotten by

$$M'_i = e^{-j2\pi q_i(x,y)} \tag{22}$$

where  $i = 1, 2$ .

- (3) Perform the IDRPE transform on the matrix  $LLH_2$  to obtain the matrix  $LLH_1$  via Eq. (23),

$$LLH_1 = FFT\left(\left(IFFFT(LLH_2)\right) \cdot M'_2\right) \cdot M'_1 \tag{23}$$

After the above operation, the phase and amplitude are picked up from the matrix  $LLH_1$  to find the matrices  $LL_1$  and  $H_1$  with the size of  $m \times m$  via Eq. (15), respectively.

Step 5: Generation and scrambling of measurement matrix. As described in Section 3.1.2, three measurement matrices  $\Phi_1, \Phi_2, \Phi_3$  are constructed by the chaotic sequences  $U_1$  and  $K_1$ . To protect the security of data on the cloud, the user selects three sequences  $U_{S1}, U_{S2}$  and  $U_{S3}$  sized of  $1 \times m$  from the chaotic sequence  $U_1$ , then sorts them in ascending order to obtain index sequences  $s_4, s_5$  and  $s_6$ , respectively. Subsequently, the matrices  $M_{a1}, M_{a2}, M_{a3}$  are obtained by shuffling the measurement matrices  $\Phi_1, \Phi_2, \Phi_3$  via,

$$\begin{cases} M_{a1}(:, s_4(t)) = \Phi_1(:, t) \\ M_{a2}(:, s_5(t)) = \Phi_2(:, t) \\ M_{a3}(:, s_6(t)) = \Phi_3(:, t) \end{cases} \tag{24}$$

where  $1 \leq t \leq m$ . Afterwards, the three scrambled measurement matrices are transmitted to the cloud for CS reconstruction.

Step 6: CS reconstruction on cloud. The matrix  $H_1$  is divided into three non-overlapping matrices  $y_1, y_2, y_3$ , where the size of  $y_1$  is  $m/2 \times m$ , the size of  $y_2$  and  $y_3$  are both  $m/4 \times m$ . When the cloud receives the measurement matrices from the user, it uses its strong computing power to get the sparse coefficient matrices  $HL_{11}, LH_{11}, HH_{11}$  sized of  $m \times m$ , and then transmits them to the user.

Step 7: Inverse scrambling of sparse coefficient matrices. After the user receives the sparse coefficient matrices  $HL_{11}, LH_{11}, HH_{11}$  from the cloud, it uses the index sequence  $s_4, s_5, s_6$  to scramble them to obtain three sparse coefficient matrices  $HL_1, LH_1, HH_1$ . The detailed inverse scrambling operation is as,

$$\begin{cases} HL_1(t, :) = HL_{11}(s_4(t), :) \\ LH_1(t, :) = LH_{11}(s_5(t), :) \\ HH_1(t, :) = HH_{11}(s_6(t), :) \end{cases} \tag{25}$$

where  $1 \leq t \leq m$ .

Step 8: Inverse sparsity. According to the following Eq. (26), three detail components  $HL$ ,  $LH$  and  $HH$  with the size of  $m \times m$  are obtained by inversely processed the sparse coefficient matrices  $HL_1$ ,  $LH_1$ ,  $HH_1$ .

$$\begin{cases} HL = \psi' HL_1 \psi \\ LH = \psi' LH_1 \psi \\ HH = \psi' HH_1 \psi \end{cases} \quad (26)$$

where  $\psi'$  is the transposition of matrix  $\psi$ .

Step 9: Inverse scrambling of approximate component. The index sequence  $s_1$  is obtained by sorting the chaotic sequence  $X_1$  in ascending order, the matrix  $LL_1$  is transformed to one-dimensional sequence  $l_2$ , and then  $l_1$  is gotten by inversely shuffling  $l_2$  by use of  $s_1$ , and converted to the approximate component  $LL$  with the size of  $m \times m$ .

$$l_2(t) = l_1(s_1(t)) \quad (27)$$

Step 100: Inverse discrete wavelet transform (IDWT). By performing IDWT on the approximate component  $LL$  and detail components  $HL$ ,  $LH$ ,  $HH$ , the plain image  $P$  is obtained. The decryption process is finished.

Among the above decryption process, step 1, step 2, step 3, step 4 and step 6 are operated on the cloud platform, and the remaining steps are completed on the user side. In general, a cloud is curious about the computation content and attempts to get some sensitive information from it, even if it may faithfully perform its computation duties. Except for the cloud's curiosity, malicious behaviors should be considered. To tackle this type of semi-trusted cloud, input/output privacy designs are essential. In the proposed method, the measurement matrices are shuffled by the user and sent to the cloud for CS construction in step 5, and the gotten sparse coefficient matrices are the output information. When they are gotten by the hackers, the correct image may not be obtained without the index sequences  $s_4$ ,  $s_5$  and  $s_6$ , indicating the input/output information is secure. Besides, before decrypting the plain image, we can compare the contour information of the approximate component and detail components to determine whether the user is cheated by the cloud.

## 4 Simulation results

In this section, MATLAB R2016a is employed to verify the encryption and decryption effects of the proposed algorithm in a personal computer with an Intel(R) Core(TM) i7-6700 CPU 3.40 GHz and 8 GB memory, and the operating system is Microsoft Windows 10. The parameters used in the algorithm are shown in Table 1.  $t_1$ ,  $t_2$ ,  $t_3$  and  $t_4$  are the parameters of the external key, and  $\mu_1$ ,  $\mu_2$ ,  $\mu_3$ ,  $\mu_4$  are the control parameters of the chaotic system. The four different  $512 \times 512$  images “Lena”, “Baboon”, “Cameraman”, “Peppers” are all used as the plain images. Moreover, in order to fully illustrate the effectiveness of the algorithm, the related test images are collected from USC-SIPI image database [39].

**Table 1** Experimental parameters

| Item                                  | Parameter value  |
|---------------------------------------|--|
| Additional key parameters             | $t_1=0.354564, t_2=0.154354, t_3=0.254168, t_4=0.445635$ |
| Control parameters of chaotic systems | $\mu_1=0.93, \mu_2=0.92889, \mu_3=3.99988, \mu_4=4$      |

## 4.1 Encryption and decryption results

The encryption and decryption images of “Lena”, “Baboon”, “Cameraman” and “Peppers” with the size of  $512 \times 512$  are shown in Fig. 5. The simulation results display that when the compression ratio (CR) is 0.5, the cipher images shown in the second column cannot be recognized, thus protecting the information of plaintext images, and their volumes have been compressed half. In addition, the reconstructed images shown in the third column have good visual effect and are just like their corresponding plain images.

## 4.2 Compression performance

This section mainly shows the compression performance of the algorithm from peak signal-to-noise ratio (PSNR) and structural similarity index measurement (SSIM), and compares with the existing algorithms to show that the proposed algorithm has good reconstruction quality.

### 4.2.1 Peak signal-to-noise ratio (PSNR)

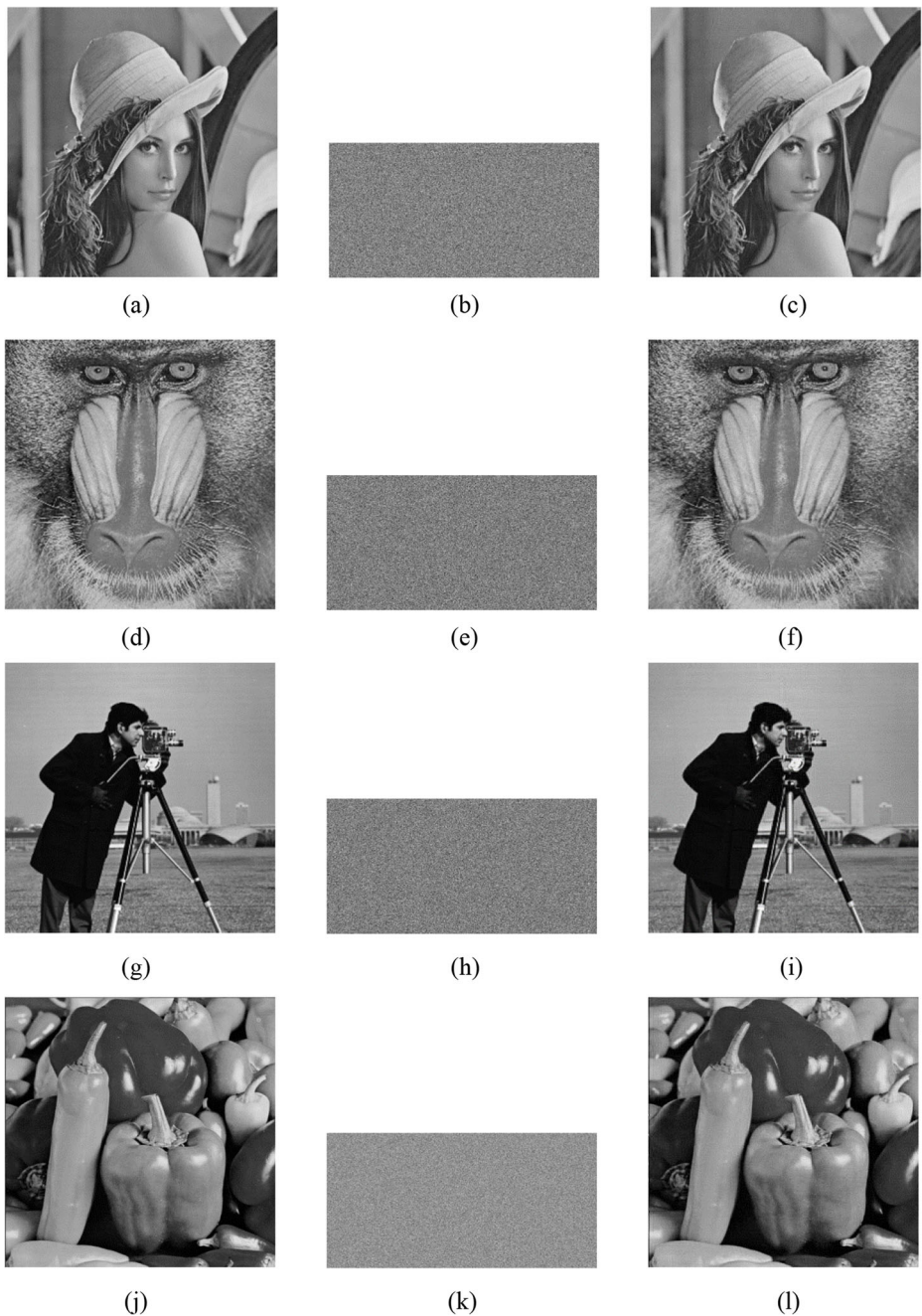
The PSNR is usually employed to estimate the quality of the decrypted image, and it can be calculated by [46].

$$PSNR = 10 \log \frac{255^2}{(1/N^2) \sum_{i=1}^N \sum_{j=1}^N [D(i, j) - I(i, j)]^2} \quad (28)$$

where  $D(i, j)$  and  $I(i, j)$  represent the reconstructed image and the plain image, respectively. And the PSNR test results of different images are shown in Tables 2 and 3. In Table 2, the plain images are all  $512 \times 512$ , the PSNR values between the reconstructed image and plain image are more than 30 dB, and the maximum value is more than 36 dB for “Cameraman”, when the compression ratio is 0.5. For  $256 \times 256$  images in Table 3, the PSNR values are about 30 dB. These results both demonstrate that the decrypted images gotten by our algorithm have good visual quality and can be well recognized. Additionally, it may be found from Table 4 that our algorithm has higher PSNR values than those in [12, 31, 59], which means that the proposed ICES has better reconstruction quality.

### 4.2.2 Structural similarity index measurement (SSIM)


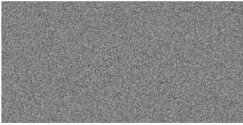

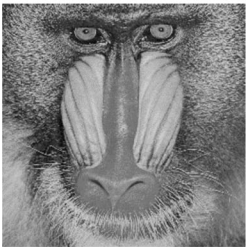
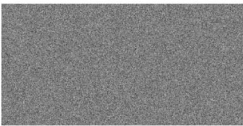
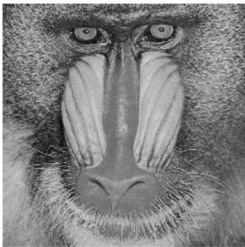

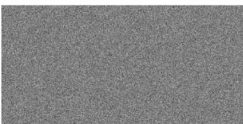

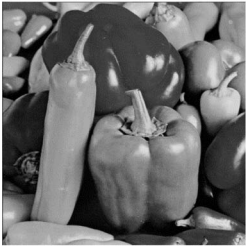
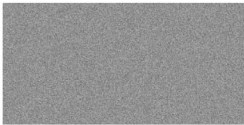

Besides, SSIM is employed to measure the similarity of the decrypted image and plain image. Its value range is  $[0, 1]$ , and the larger the value is, the higher the similarity between these two images is. SSIM is defined by [43].



**Fig. 5** Plain images, cipher images and decrypted images. (a), (b), (c) represent the original, cipher and decrypted images of “Lena”, respectively. (d), (e), (f) represent the original, cipher and decrypted images of “Baboon”, respectively. (g), (h), (i) represent the original, cipher and decrypted images of “Cameraman”, respectively. (j), (k), (l) represent the original, cipher and decrypted images of “Peppers”, respectively



**Table 2** PSNR values of different plain images sized of 512 × 512.

| Original image<br>(512×512)   | Cipher image<br>(CR=0.5)  | Decrypted images<br>(512×512)   | PSNR<br>(dB) |
|---|---|---|--------------|
|    |    |    | 35.3844      |
|    |    |    | 31.6223      |
|   |    |   | 36.4611      |
|  |  |  | 34.5971      |

$$SSIM = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)} \tag{29}$$

where  $\mu_X$  and  $\mu_Y$  respectively represent the mean value of the plain image  $X$  and decrypted

**Table 3** PSNR values of different images sized of  $256 \times 256$ .


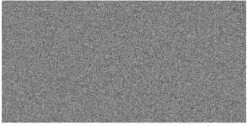

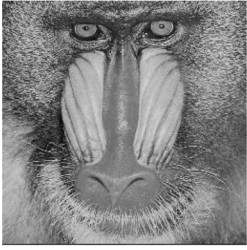
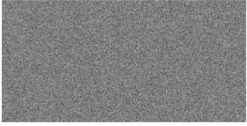
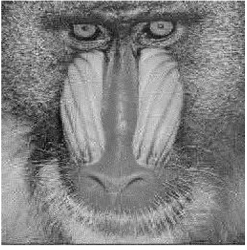

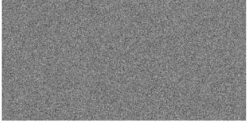



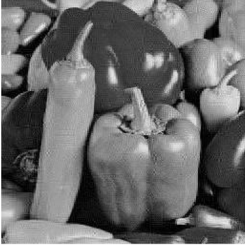
| Original image<br>(256×256)   | Cipher image<br>(CR=0.5)  | Decrypted images<br>(256×256)   | PSNR<br>(dB) |
|---|---|---|--------------|
|    |    |    | 32.5049      |
|    |    |    | 29.2406      |
|   |    |   | 32.2973      |
|  |  |  | 32.3807      |

image  $Y$ ,  $\sigma_X$  and  $\sigma_Y$  respectively denote the variance of  $X$  and  $Y$ ,  $\sigma_{XY}$  is the covariance of  $X$  and  $Y$ ,  $C_1 = (k_1 \times L)^2$ ,  $C_2 = (k_2 \times L)^2$ ,  $k_1 = 0.01$ ,  $k_2 = 0.03$ , and  $L = 255$ .

The SSIM values of different images are listed in Table 5. When the compression ratio of the original images is 0.5, the SSIM values of the decrypted images and the original images are all above 0.95 for  $256 \times 256$  images; in the meantime, the SSIM values of the decrypted images and original images are all more than 0.97 for  $512 \times 512$  images, indicating that the original images and the reconstructed images are very similar, and the proposed method has good compression and reconstruction effect.

**Table 4** Comparison of different algorithms for PSNR values

| Image     | Ours       | Ref. [31]  | Ref. [59] | Ref. [12] |
|-----------|------------|------------|-----------|-----------|
| Lena      | 35.3844 dB | 34.5560 dB | <34 dB    | <35 dB    |
| Baboon    | 31.6223 dB | –          | –         | –         |
| Cameraman | 36.4611 dB | 34.6995 dB | –         | –         |
| Peppers   | 34.5971 dB | 31.5132 dB | –         | <34 dB    |

## 5 Performance analyses

This section assesses the performance of the proposed cipher from the key space, key sensitivity, information entropy, histogram, adjacent pixel correlation, cropping attack and noise attack.

### 5.1 Key space analysis

The secret key of the proposed algorithm is as follows: ① the external keys  $t_1, t_2, t_3, t_4$ ; ② the control parameters  $\mu_1, \mu_2, \mu_3, \mu_4$  of chaotic systems; ③ the 256-bit hash value from the SHA-256 hash function of the original image. If the computing accuracy of the computer is  $10^{-14}$ , the key space is about  $(10^{14})^4 \times (10^{14})^4 = 10^{112} > 2^{372}$ . If the 256-bit hash value is considered, the overall key space is much larger than  $2^{100}$ , which can resist all kinds of brute-force attacks [8]. In addition, as shown in Table 6, the proposed scheme has the largest key space than encryption schemes in [12, 21, 31, 59], so it has higher security.

### 5.2 Key sensitivity analysis

A good image encryption algorithm should have a high sensitivity to secret keys [15]. It means that a tiny change in the keys would cause a great distortion [24]. In this subsection, the Lena image shown in Fig. 5(a) is utilized as test image, the correct cipher image and decrypted image are displayed in Fig. 5(b) and (c).

**Table 5** SSIM values of different images

| Image     | SSIM    |         |
|-----------|---------|---------|
|           | 256×256 | 512×512 |
| Lena      | 0.9906  | 0.9964  |
| Baboon    | 0.9714  | 0.9924  |
| Cameraman | 0.9501  | 0.9796  |
| Peppers   | 0.9609  | 0.9737  |
| 7.1.01    | –       | 0.9960  |
| 7.1.02    | –       | 0.9994  |
| 7.1.03    | –       | 0.9977  |
| 7.1.04    | –       | 0.9883  |
| 7.1.05    | –       | 0.9928  |
| 7.1.06    | –       | 0.9878  |
| 7.1.07    | –       | 0.9934  |
| 7.1.08    | –       | 0.9978  |

**Table 6** Comparison of key space of the proposed method with other methods

| Algorithm | Ours       | Ref. [31] | Ref. [59] | Ref. [12] | Ref. [21] |
|-----------|------------|-----------|-----------|-----------|-----------|
| Key Space | $>2^{372}$ | $2^{360}$ | $2^{199}$ | $2^{149}$ | $2^{176}$ |

The number of pixel change rate (NPCR) is an important criterion to measure the pixel consistency between two images. The NPCR is computed by [22].

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \quad (30)$$

$$D(i,j) = \begin{cases} 1, & C_1(i,j) \neq C_2(i,j) \\ 0, & otherwise \end{cases} \quad (31)$$

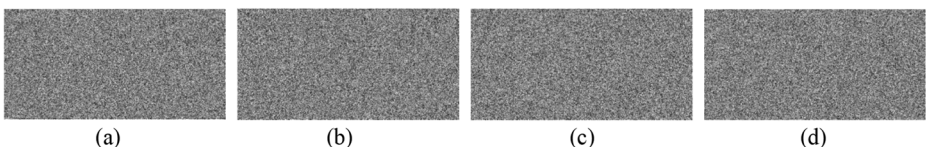
where  $C_1$  and  $C_2$  represent two images, respectively, and  $M$  and  $N$  are the sizes of the image.

In the simulation, the key sensitivity analyses may be performed in encryption process and decryption process. Firstly, the key parameters  $t_1$ ,  $t_2$ ,  $t_3$ , and  $t_4$  are chosen and a change of  $10^{-14}$  is added on them to obtain new parameters  $t_1 + 10^{-14}$ ,  $t_2 + 10^{-14}$ ,  $t_3 + 10^{-14}$ , and  $t_4 + 10^{-14}$ . Next, the changed parameters are utilized to encrypt the plain image Fig. 5(a), and the obtained cipher images are shown in Fig. 6. The subtraction images and NPCR values between them and Fig. 5(b) are given in Fig. 7 and Table 7, respectively. Obviously, when the key changes slightly, the pixel change is about 99.60%, which means that our algorithm has high key sensitivity in the encryption stage.

Besides, the trivially changed parameters  $t_1 + 10^{-14}$ ,  $t_2 + 10^{-14}$ ,  $t_3 + 10^{-14}$ , and  $t_4 + 10^{-14}$  are utilized to decrypt the Lena cipher image shown in Fig. 5(b), and each time only one parameter is modified. The results are illustrated in Fig. 8(a)-(d). As shown from Fig. 8, the correct plain image may not be obtained even if the parameters change slightly. In addition, the NPCR between the correctly decrypted image and the decrypted image with changed key is tested and listed in Table 8. As can be observed from Table 8, the values of NPCR are about 99.5%, which means that the proposed scheme has high key sensitivity to make brute-force attacks invalid.

### 5.3 Histogram analysis

In order to prevent attackers from statistically analyzing the gray value distribution to crack the image, it is required that the histogram of the cipher image is smooth and uniform [10, 50]. The histograms of plain images “Lena”, “Baboon”, “Cameraman”, “Peppers” and corresponding cipher images are plotted respectively, as shown in Fig. 9. It can be seen from Fig. 9 that the histogram of the plain image is steep and fluctuating, histogram of the cipher image is very



**Fig. 6** Encrypted “Lena” using incorrect keys (a)  $t_1 + 10^{-14}$ , (b)  $t_2 + 10^{-14}$ , (c)  $t_3 + 10^{-14}$ , (d)  $t_4 + 10^{-14}$

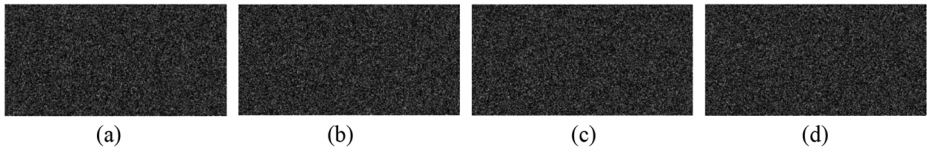


Fig. 7 The subtraction images between Fig. 5(b) and (a) Fig. 6(a), (b) Fig. 6(b), (c) Fig. 6(c), (d) Fig. 6(d)

uniform and significantly different from that of the plain image, so it can effectively stand statistical analysis attacks.

In addition, the chi-square test ( $\chi^2$ ) and histogram variance are used to quantitatively evaluate the uniformity of the cipher image. The chi-square test is calculated by [41].

$$\chi^2_{test} = \sum_{i=0}^{255} \frac{(o_i - o)^2}{o} \tag{32}$$

$$o = \frac{M \times N}{256} \tag{33}$$

where  $o_i$  is the number of times of the pixel value  $i$  in the image sized of  $M \times N$ . Under a significance level of 0.05, the chi-square test is shown in Table 9. Table 10 gives the variance of histogram of plain images and cipher images. The results indicate that all cipher images have past the chi-square test, and the histogram variance of cipher image is much smaller than that of plain image, which indicates that the cipher image has a random distribution and consequently does not provide any clue for statistical analysis attacks.

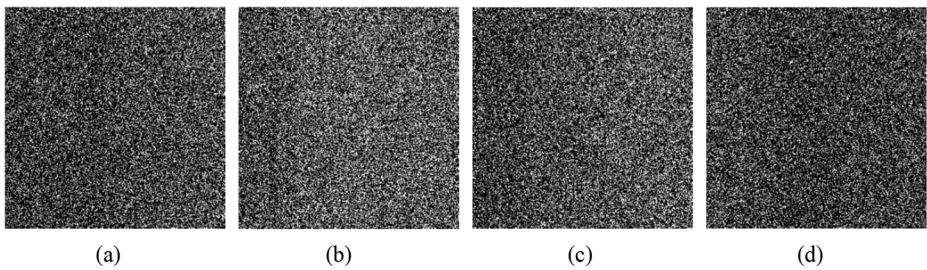
### 5.4 Correlation coefficient analysis

Correlation coefficient analyses of plain images and cipher images are utilized to qualitatively evaluate the encryption scheme. The correlation coefficient may be defined as [59],

Table 7 Sensitivity analysis of encryption keys

| Image     | Encryption keys  |                  |                  |                  |
|-----------|------------------|------------------|------------------|------------------|
|           | $t_1 + 10^{-14}$ | $t_2 + 10^{-14}$ | $t_3 + 10^{-14}$ | $t_4 + 10^{-14}$ |
| Lena      | 0.9962           | 0.9959           | 0.9961           | 0.9961           |
| Baboon    | 0.9961           | 0.9960           | 0.9959           | 0.9964           |
| Cameraman | 0.9958           | 0.9960           | 0.9961           | 0.9962           |
| Peppers   | 0.9960           | 0.9961           | 0.9962           | 0.9961           |
| 7.1.01    | 0.9961           | 0.9961           | 0.9961           | 0.9959           |
| 7.1.02    | 0.9961           | 0.9961           | 0.9958           | 0.9959           |
| 7.1.03    | 0.9962           | 0.9960           | 0.9961           | 0.9957           |
| 7.1.04    | 0.9958           | 0.9960           | 0.9960           | 0.9960           |
| 7.1.05    | 0.9965           | 0.9961           | 0.9962           | 0.9959           |
| 7.1.06    | 0.9958           | 0.9961           | 0.9963           | 0.9963           |
| 7.1.07    | 0.9957           | 0.9962           | 0.9958           | 0.9957           |
| 7.1.08    | 0.9961           | 0.9962           | 0.9961           | 0.9961           |





**Fig. 8** Decrypted “Lena” using incorrect keys (a)  $t_1 + 10^{-14}$ , (b)  $t_2 + 10^{-14}$ , (c)  $t_3 + 10^{-14}$ , (d)  $t_4 + 10^{-14}$

**Table 8** Sensitivity analysis of decryption keys

| Image     | Decryption keys  |                  |                  |                  |
|-----------|------------------|------------------|------------------|------------------|
|           | $t_1 + 10^{-14}$ | $t_2 + 10^{-14}$ | $t_3 + 10^{-14}$ | $t_4 + 10^{-14}$ |
| Lena      | 0.9985           | 0.9983           | 0.9985           | 0.9985           |
| Baboon    | 0.9952           | 0.9951           | 0.9951           | 0.9951           |
| Cameraman | 0.9952           | 0.9951           | 0.9951           | 0.9951           |
| Peppers   | 0.9927           | 0.9924           | 0.9919           | 0.9928           |
| 7.1.01    | 0.9983           | 0.9981           | 0.9981           | 0.9984           |
| 7.1.02    | 0.9989           | 0.9987           | 0.9988           | 0.9988           |
| 7.1.03    | 0.9986           | 0.9984           | 0.9984           | 0.9985           |
| 7.1.04    | 0.9940           | 0.9939           | 0.9939           | 0.9935           |
| 7.1.05    | 0.9984           | 0.9980           | 0.9984           | 0.9984           |
| 7.1.06    | 0.9977           | 0.9976           | 0.9976           | 0.9977           |
| 7.1.07    | 0.9980           | 0.9979           | 0.9980           | 0.9981           |
| 7.1.08    | 0.9985           | 0.9982           | 0.9983           | 0.9987           |

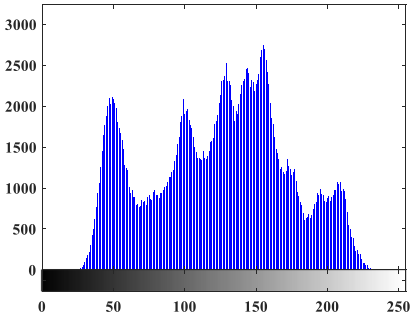
$$CC = \frac{Cov(v_{ij}, v_{xy})}{\sqrt{D(v_{ij}) \cdot D(v_{xy})}} \tag{34}$$

$$Cov(v_{ij}, v_{xy}) = \frac{1}{MN} \sum_{i,x=1}^M \sum_{j,y=1}^N ([v_{ij} - E(v)] \cdot [v_{xy} - E(v)]) \tag{35}$$

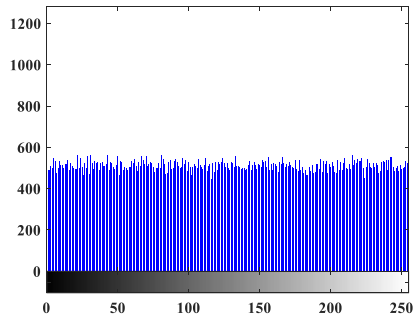
where  $v_{ij}$  and  $v_{xy}$  are the values of two adjacent pixels located at  $(i, j)$  and  $(x, y)$  in the image, respectively,  $E(\cdot)$  represents the mean value of image pixels, and  $D(\cdot)$  expresses the variance of the values,  $M$  and  $N$  are the sizes of the image.

In this section, we randomly choose 5000 pairs of adjacent pixels from the plain image and the cipher image of “Lena”, and analyze the correlations from horizontal, vertical and diagonal directions. Figure 10 plots the correlation distribution among adjacent pixels of “Lena” before and after encryption. It can be observed from Fig. 10 that the adjacent pixels in the horizontal,

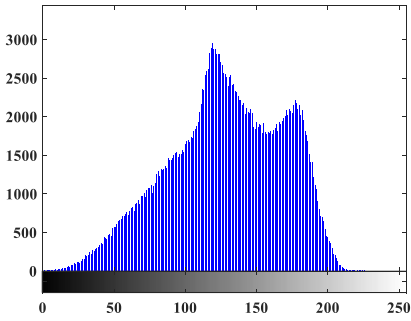
**Fig. 9** Histogram of different plain images ( $512 \times 512$ ) and cipher images ( $256 \times 512$ ): (a) “Lena”, (b) encrypted “Lena”, (c) “Baboon”, (d) encrypted “Baboon”, (e) “Cameraman”, (f) encrypted “Cameraman”, (h) “Peppers”, (i) encrypted “Peppers”



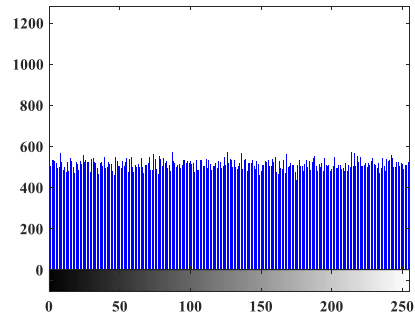
(a)



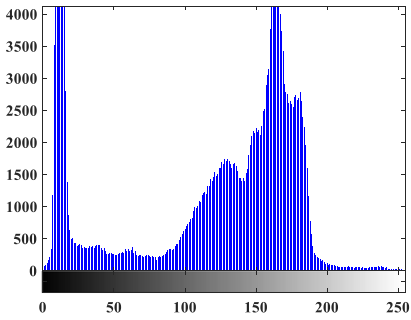
(b)



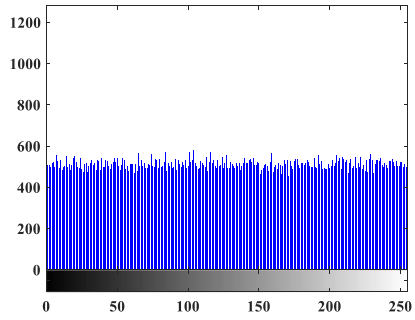
(c)



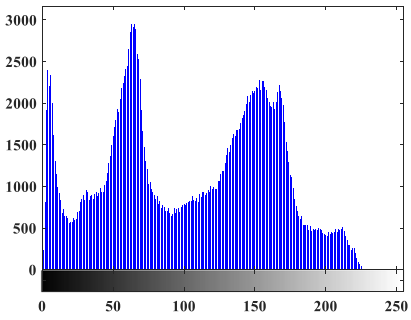
(d)



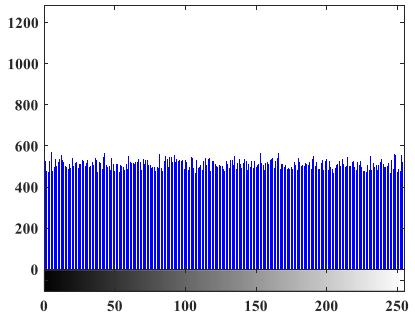
(e)



(f)



(g)



(h)

**Table 9** Chi-square test of different images

| Image     | $\chi_{test}^2$ | $\chi_{256,0.05}^2$ | Result |
|-----------|-----------------|---------------------|--------|
| Lena      | 246.0469        | 293                 | Pass   |
| Baboon    | 242.2695        | 293                 | Pass   |
| Cameraman | 234.7383        | 293                 | Pass   |
| Peppers   | 235.3125        | 293                 | Pass   |
| 7.1.01    | 250.3672        | 293                 | Pass   |
| 7.1.02    | 262.6289        | 293                 | Pass   |
| 7.1.03    | 243.3594        | 293                 | Pass   |
| 7.1.04    | 221.6055        | 293                 | Pass   |
| 7.1.05    | 228.3359        | 293                 | Pass   |
| 7.1.06    | 250.8906        | 293                 | Pass   |
| 7.1.07    | 265.6055        | 293                 | Pass   |
| 7.1.08    | 247.4258        | 293                 | Pass   |

vertical and diagonal directions of the plain image are linearly distributed, while the pixels of the cipher image are uniformly distributed, indicating that the correlation between adjacent pixels of the plaintext image can be effectively eliminated.

In addition, the correlation coefficients in the horizontal, vertical, and diagonal directions of test images are calculated and shown in Table 11. From Table 11, one may watch that the correlation coefficients of the plain images are close to 1, while the correlation coefficients of the corresponding cipher images tend to 0, which means that the strong correlation of the plain image has been effectively removed. Table 12 shows the comparison results of correlation coefficients. By comparing the data in Table 12, the correlation coefficients obtained by the proposed algorithm are similar to [12, 13, 29, 59], demonstrating that our method has good confusion effect and it may resist statistical analysis attacks.

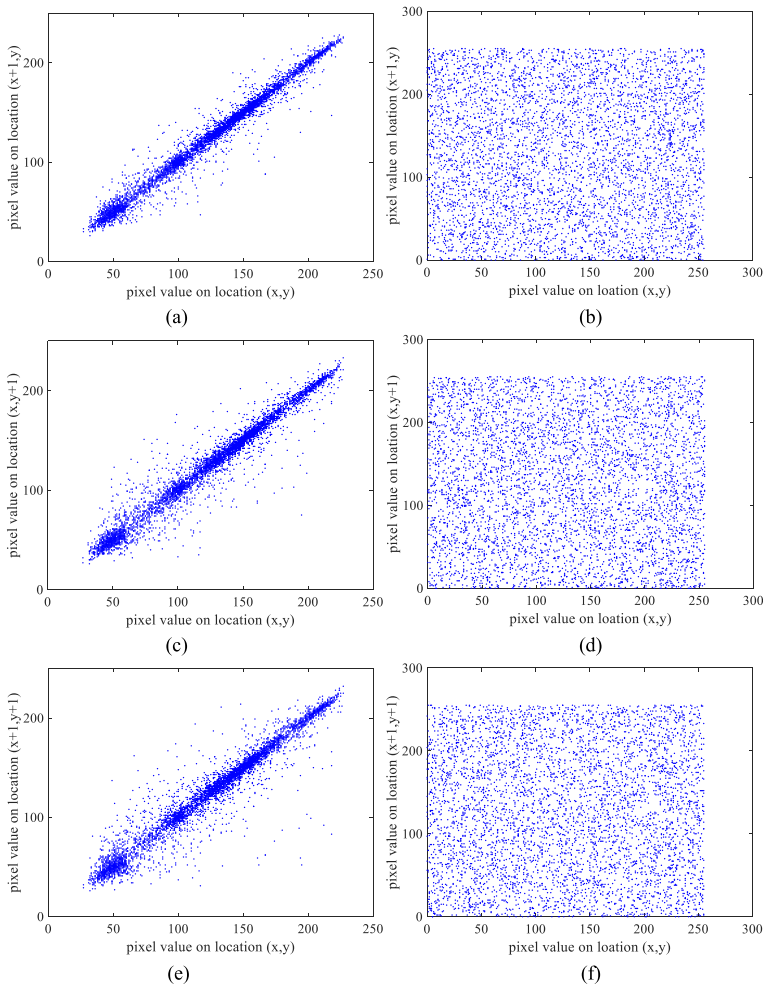
## 5.5 Information entropy analysis

To a certain extent, the information entropy of the image can represent the degree of disturbance of the image [32]. Moreover, the image is divided into non overlapping

**Table 10** Variance of histogram of different images

| Image     | Plain image | Cipher image         |
|-----------|-------------|----------------------|
| Lena      | 494.0235    | $6.3586 \times 10^5$ |
| Baboon    | 486.4392    | $8.4878 \times 10^5$ |
| Cameraman | 471.3176    | $1.6807 \times 10^6$ |
| Peppers   | 472.4706    | $5.5752 \times 10^5$ |
| 7.1.01    | 502.6980    | $4.6324 \times 10^6$ |
| 7.1.02    | 527.3176    | $2.1689 \times 10^7$ |
| 7.1.03    | 488.6275    | $8.1354 \times 10^6$ |
| 7.1.04    | 444.9490    | $4.2281 \times 10^6$ |
| 7.1.05    | 458.4627    | $2.2382 \times 10^6$ |
| 7.1.06    | 503.7490    | $1.8589 \times 10^6$ |
| 7.1.07    | 533.2941    | $4.2200 \times 10^6$ |
| 7.1.08    | 496.7922    | $1.1596 \times 10^7$ |





**Fig. 10** Correlation of two adjacent pixels in the plain and cipher images. (a) and (b) are the horizontal correlation of the plain image “Lena” and its cipher image, respectively; (c) and (d) are the vertical correlation of the plain image “Lena” and its cipher image, respectively; (e) and (f) are the diagonal correlation of the plain image “Lena” and its cipher image, respectively

blocks of the same size. The average value of the information entropy of each block is the local entropy of the image. The local entropy can better represent the randomness of the pixel values in the image, and its ideal value is 8. At the same time, for 8-bit grayscale image, the image entropy divided by 8 is called relative entropy, which can be used to indicate whether the image can be compressed and its ideal value is 1. The difference between 1 and relative entropy is called redundancy [56]. Redundancy indicates whether the image has information redundancy, and its ideal value is 0. The information entropy, local entropy, relative entropy and redundancy are calculated by [18],

**Table 11** Correlation coefficient between the plain image and cipher image

| Image     | Plain image |          |          | Cipher image |          |          |
|-----------|-------------|----------|----------|--------------|----------|----------|
|           | Horizontal  | Vertical | Diagonal | Horizontal   | Vertical | Diagonal |
| Lena      | 0.9846      | 0.9717   | 0.9578   | -0.0018      | -0.0070  | -0.0024  |
| Baboon    | 0.9125      | 0.9367   | 0.8697   | 0.0040       | 0.0178   | 0.0071   |
| Cameraman | 0.9895      | 0.9814   | 0.9708   | -0.0096      | 0.0082   | -0.0043  |
| Peppers   | 0.9829      | 0.9804   | 0.9702   | 0.0134       | -0.0062  | -0.0024  |
| 7.1.01    | 0.9223      | 0.9640   | 0.9073   | -0.0030      | -0.0168  | -0.0237  |
| 7.1.02    | 0.9397      | 0.9371   | 0.8865   | 0.0139       | -0.0074  | -0.0230  |
| 7.1.03    | 0.9407      | 0.9497   | 0.9077   | 0.0006       | -0.0108  | 0.0021   |
| 7.1.04    | 0.9650      | 0.9780   | 0.9521   | -0.0080      | -0.0217  | 0.0148   |
| 7.1.05    | 0.9108      | 0.9459   | 0.8945   | 0.0513       | -0.0042  | -0.0253  |
| 7.1.06    | 0.9038      | 0.9430   | 0.8853   | 0.0003       | 0.0101   | -0.0028  |
| 7.1.07    | 0.8758      | 0.8793   | 0.8370   | -0.0477      | -0.0001  | 0.0123   |
| 7.1.08    | 0.9305      | 0.9602   | 0.9215   | -0.0189      | 0.0052   | 0.0070   |

$$\begin{cases} H(m) = \sum_{i=0}^{255} p(m_i) \log_2 \frac{1}{p(m_i)} \\ \bar{H}_{k,T_B}(m) = \sum_{i=1}^k \frac{H(M_i)}{k} \\ H_r = \frac{H}{8} \\ R_{ed} = 1 - H_r \end{cases} \quad (36)$$

where  $H(m)$  is information entropy,  $p(m_i)$  denotes the occurrence probability of symbol  $m_i$ .  $\bar{H}_{k,T_B}(m)$  is the local entropy, the image is divided into  $k$  sub-blocks, and the

**Table 12** Comparison with other algorithms for Lena image

| Algorithm  | Ours    | Ref. [59] | Ref. [12] | Ref. [13] | Ref. [29] |
|------------|---------|-----------|-----------|-----------|-----------|
| Horizontal | -0.0018 | 0.0075    | 0.0018    | 0.0003    | 0.0053    |
| Vertical   | -0.0070 | -0.0015   | 0.0014    | -0.0064   | 0.0193    |
| Diagonal   | -0.0024 | -0.0044   | 0.0034    | 0.0110    | 0.0034    |

**Table 13** Information entropy, local entropy, relative entropy and redundancy of plain images

| Image     | Information entropy | Local entropy | Relative entropy | Redundancy |
|-----------|---------------------|---------------|------------------|------------|
| Lena      | 7.4451              | 4.9885        | 0.9306           | 6.9363%    |
| Baboon    | 7.2925              | 5.9121        | 0.9115           | 8.8437%    |
| Cameraman | 7.5715              | 5.0681        | 0.9464           | 5.3563%    |
| Peppers   | 7.0480              | 4.1954        | 0.8810           | 11.9000%   |
| 7.1.01    | 6.0274              | 4.6268        | 0.7534           | 24.6575%   |
| 7.1.02    | 4.0044              | 2.1996        | 0.5005           | 49.9450%   |
| 7.1.03    | 5.4957              | 4.3101        | 0.6869           | 31.3037%   |
| 7.1.04    | 6.1074              | 4.5749        | 0.7634           | 23.6575%   |
| 7.1.05    | 6.5631              | 5.2560        | 0.8203           | 17.9612%   |
| 7.1.06    | 6.6952              | 5.4694        | 0.8369           | 16.3100%   |
| 7.1.07    | 5.9915              | 5.1704        | 0.7489           | 25.1062%   |
| 7.1.08    | 5.0534              | 3.7650        | 0.6316           | 36.8325%   |

information entropy of each sub-block  $M_i$  is  $H(M_i)$ , and pixel number of every sub-block is  $T_B$ .  $H_r$  represents the correlation entropy, and  $R_{ed}$  represents the redundancy.

In the simulation, the number of sub-blocks is  $k = 256$ . For the plain image, the pixel number of every sub-block is  $T_B = 1024$ ; For the cipher image, the pixel number of every sub-block is  $T_B = 512$ , and the information entropy, local entropy, relative entropy and redundancy of different plain images are tested, as shown in Table 13. The corresponding results of encrypted images are given in Table 14.

Comparing Tables 13 and 14, one can conclude that: (1) the information entropy of the plain image is significantly less than the ideal value 8, while the information entropy of the cipher image is very close to 8; (2) the local entropy of every plain image is less than its cipher image, and the cipher image is more random; (3) the relative entropy of each plain image is less than 1, indicating that the plain image can be compressed, and the relative entropy of the cipher image is very close to 1, demonstrating that the cipher image is difficult to compress; (4) the redundancy of each plain image is greater than 5%, and that of each cipher image is less than 0.02%, which means that the cipher image has little information redundancy.

Furthermore, the information entropy values of different cipher images gotten by different cryptosystems are shown in Table 15. It may be observed from Table 15 that the information entropy of our algorithm is slightly higher than that of [29, 59], and trivially lower than that of [12, 13], which shows that the proposed scheme is robust against entropy attack and has higher security level.

## 5.6 Noise attack analysis

In the simulation, the “Lena” image (shown in Fig. 5(a)) is used as test image, Salt & Pepper noise (SPN), Speckle noise (SN), and Gaussian noise (GN) with different intensities are added to the “Lena” cipher image (shown in Fig. 5(b)), and then the obtained noisy images are decrypted and shown in Fig. 11. The PSNR values between the decrypted images and the plain image are calculated and displayed in Fig. 12.

From Figs. 11 and 12, it can be seen that when the noise intensity increases from 0.000001 to 0.000007, the proposed encryption algorithm has strong resistance to SPN, and the PSNR values keep to 35.3844 dB; besides, our method has general resistance to SN and GN, under the same noise level, the PSNR values vary from 35.3844 dB to 26.0047 dB for SN, and those

**Table 14** Information entropy, local entropy, relative entropy and redundancy of cipher images

| Image     | Information entropy | Local entropy | Relative entropy | Redundancy |
|-----------|---------------------|---------------|------------------|------------|
| Lena      | 7.9986              | 7.1775        | 0.999812         | 00.0187%   |
| Baboon    | 7.9985              | 7.1761        | 0.999812         | 00.0187%   |
| Cameraman | 7.9987              | 7.1758        | 0.999837         | 00.0162%   |
| Peppers   | 7.9986              | 7.1776        | 0.999825         | 00.0175%   |
| 7.1.01    | 7.9986              | 7.1760        | 0.999825         | 00.0175%   |
| 7.1.02    | 7.9985              | 7.1715        | 0.999812         | 00.0187%   |
| 7.1.03    | 7.9986              | 7.1728        | 0.999825         | 00.0175%   |
| 7.1.04    | 7.9987              | 7.1792        | 0.999837         | 00.0162%   |
| 7.1.05    | 7.9987              | 7.1738        | 0.999837         | 00.0162%   |
| 7.1.06    | 7.9986              | 7.1707        | 0.999825         | 00.0175%   |
| 7.1.07    | 7.9985              | 7.1726        | 0.999812         | 00.0187%   |
| 7.1.08    | 7.9986              | 7.1765        | 0.999825         | 00.0175%   |

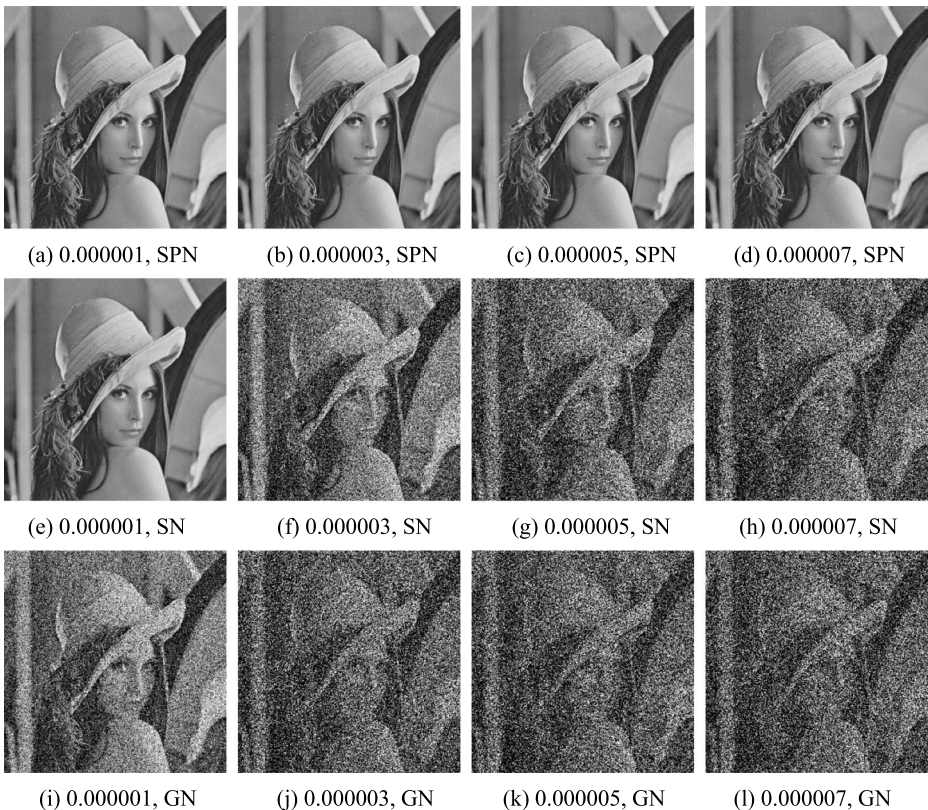
**Table 15** Comparison of information entropy for different images

| Algorithm | Ours   | Ref. [59] | Ref. [12] | Ref. [13] | Ref. [29] |
|-----------|--------|-----------|-----------|-----------|-----------|
| Lena      | 7.9986 | 7.9973    | 7.9986    | 7.9993    | 7.9984    |
| Baboon    | 7.9985 | 7.9972    | –         | –         | –         |
| Cameraman | 7.9987 | –         | –         | –         | 7.9983    |
| Peppers   | 7.9986 | 7.9972    | 7.9987    | –         | 7.9986    |

modify from 26.8283 dB to 25.8966 dB for GN. In summary, the proposed scheme is robust to noise attacks.

### 5.7 Cropping attack analysis

During the transmission of cipher images, some information may be lost. Therefore, assessing the ability to resist cropping attack is necessary for a cryptosystem. Here, the “Lena” cipher image (shown in Fig. 5(b)) is selected for testing, the data of  $16 \times 16$ ,  $32 \times 32$ ,  $64 \times 64$ ,  $128 \times 128$  are cropped to obtain images shown in Fig. 13(a)–(d), and the corresponding decrypted images are illustrated in Fig. 13(e)–(h). The PSNR values of the decrypted images and plain images are computed and listed in Fig. 14.

**Fig. 11** The reconstructed images under different noise intensities

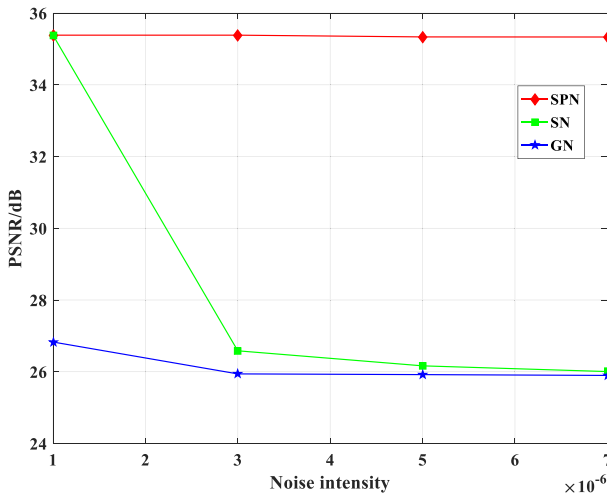


Fig. 12 PSNR values under different noise intensities

It can be observed from Figs. 13 and 14 that when the lost data increases, the decrypted image becomes blurred, but the outline of the plain image can still be recognized. In addition, the PSNR value is another indicator to evaluate the performance of cropping attacks. When 1/64 of the encrypted image is lost, the PSNR value between the decrypted image and plain image drops to 30.9005 dB. When the encrypted image loses 1/8 of data, the PSNR is 26.8660 dB, which shows that the proposed algorithm can resist certain cropping attacks.

### 5.8 Running time analysis

The encryption process mainly includes scrambling of approximate component and compression of detail components, double random phase encoding, random pixel scrambling and diffusion. The process of decryption on cloud mainly includes the generation and scrambling

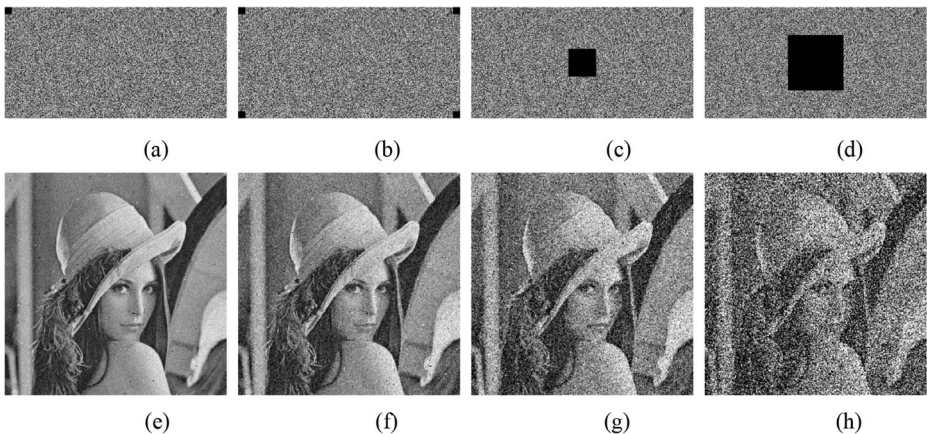


Fig. 13 Cropping attack test images. (a), (b), (c), (d) are the cipher images with the cropping size of  $16 \times 16$ ,  $32 \times 32$ ,  $64 \times 64$ ,  $128 \times 128$  data, respectively; (e), (f), (g), (h) are the corresponding decrypted images of (a), (b), (c), (d), respectively

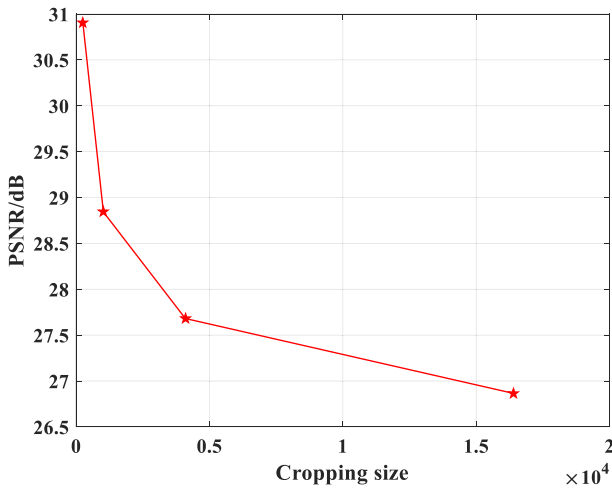


Fig. 14 PSNR values under different cropping sizes

of measurement matrix, inverse scrambling, inverse sparsity, inverse scrambling of approximate component and inverse discrete wavelet transform (IDWT). It can be seen from Table 16 that for images with the size of  $512 \times 512$ , the average encryption time of the proposed algorithm is  $0.582940 s$ , the average decryption time is  $8.623362 s$  without cloud, the average decryption time is  $0.157799 s$  using cloud, and the decryption time with cloud is  $8.465562 s$  less than that without cloud.

At the same time, the specific process of encryption and decryption of Lena image is analyzed. As shown in Fig. 15, in the encryption process, scrambling and diffusion represent the scrambling of approximate components and compression of detail components. The others mainly include the generation of key and chaotic sequences, and they take a long time, accounting for about 85% of the total encryption time. In the decryption process on the cloud, the most time-consuming convex optimization process is carried out on the cloud, so the

Table 16 Encryption and decryption times of different images (Unit:  $s$ )

| Image     | Encryption time | Decryption time |          |            |
|-----------|-----------------|-----------------|----------|------------|
|           |                 | No cloud        | Cloud    | Difference |
| Lena      | 0.582683        | 8.735512        | 0.156431 | 8.579081   |
| Baboon    | 0.576587        | 8.627581        | 0.149664 | 8.477917   |
| Peppers   | 0.583076        | 8.849963        | 0.147897 | 8.702066   |
| Cameraman | 0.577763        | 8.633264        | 0.151211 | 8.482053   |
| 7.1.01    | 0.584659        | 8.541292        | 0.161846 | 8.379446   |
| 7.1.02    | 0.588407        | 8.464594        | 0.152367 | 8.312227   |
| 7.1.03    | 0.585400        | 8.720073        | 0.157279 | 8.562794   |
| 7.1.04    | 0.585653        | 8.617070        | 0.165187 | 8.451883   |
| 7.1.05    | 0.580739        | 8.585542        | 0.165273 | 8.420269   |
| 7.1.06    | 0.583628        | 8.518862        | 0.161792 | 8.357070   |
| 7.1.07    | 0.581410        | 8.598539        | 0.162598 | 8.435941   |
| 7.1.08    | 0.585283        | 8.588053        | 0.162049 | 8.426004   |
| Average   | 0.582940        | 8.623362        | 0.157799 | 8.465562   |



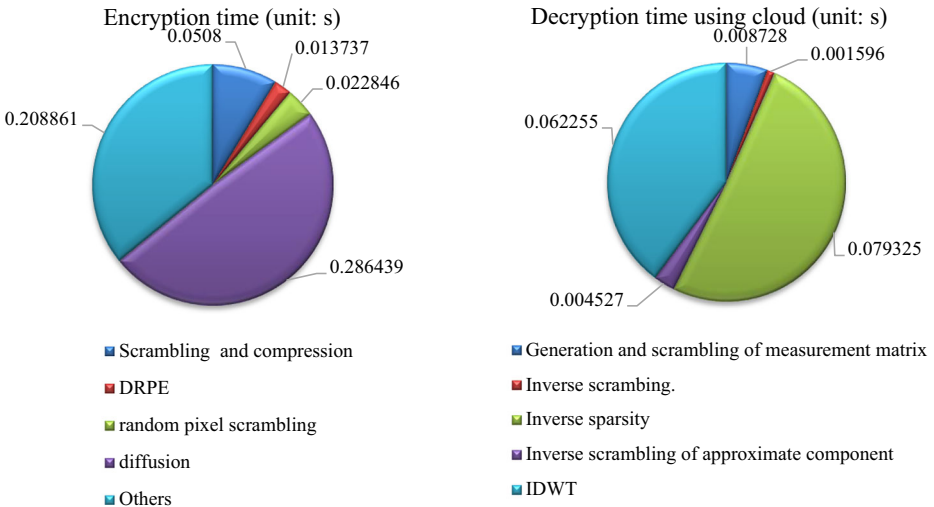


Fig. 15 The specific running time of encryption and decryption processes for Lena image

decryption time on the cloud is very short, in which the inverse discrete wavelet transformation and inverse sparsity account for 90% of the total decryption time.

In addition, without using cloud-assisted decryption, the encryption and decryption times of our proposed algorithm and other methods are shown in Table 17. It can be seen from this table that even if the cloud platform is not used to assist decryption, for  $256 \times 256$  images, the encryption time of our algorithm is shorter than that of the algorithms [6, 9, 31, 38], and the decryption time is lower than that of the algorithm [6]; For  $512 \times 512$  images, the encryption time of our algorithm is shorter than that of the algorithms [6, 9, 31, 38, 43], and the decryption time is lower than that of the algorithm [6]. To sum up, the encryption time and decryption time of the proposed algorithm are short, which shows that the algorithm is very effective and can be applied to real-time secure image communication.

### 5.9 Security analysis of cloud-assisted decryption strategy

In the proposed image encryption, the client directly transmits the cipher image to the cloud after encryption. In the decryption process, the cipher image is decrypted directly on the cloud, and then the partially decrypted cipher image is transmitted to the authorized users for the

Table 17 Comparison results with other algorithms in running times for Lena images (Unit: s)

| Image     | Encryption time |          | Decryption time |          |
|-----------|-----------------|----------|-----------------|----------|
|           | 256×256         | 512×512  | 256×256         | 512×512  |
| Ours      | 0.278052        | 0.582683 | 1.416537        | 8.735512 |
| Ref. [31] | 0.39            | >1       | –               | 6.7      |
| Ref. [6]  | 0.58            | 0.7742   | >1.5            | 14.0266  |
| Ref. [43] | 0.18486         | 0.72900  | –               | –        |
| Ref. [9]  | 0.44            | 2.76     | –               | –        |
| Ref. [38] | 0.701           | 1.363    | 0.701           | 1.363    |

second decryption to get the original image. The cipher images transmitted to the cloud platform are noise-like images, and the output partially decrypted images on the cloud are also noisy images. The original image information cannot be obtained from these images from the naked eyes. Besides, before the user transmits the measurement matrix to the cloud platform for CS reconstruction, it is scrambled by use of the index sequences generated by the chaotic sequences. Thus, without the correct index sequences, the correct detail components of the plain image may not be recovered by manipulating the inverse sparse operation on the sparse coefficient matrices reconstructed by the cloud platform. Therefore, the cloud cannot obtain the original image, ensuring the privacy and security of the image.

Moreover, decryption on the cloud also needs to consider whether the cloud deceives us. Therefore, in order to prevent cloud deception, after the users get three detail components and the approximate component using inverse scrambling, the user firstly constructs a zero matrix of the same size as the detail component. Then the reconstructed three detail components and the zero matrix are performed the IDWT to obtain a new image. Because the authorized user has the approximate component of the decrypted image, it can compare the new image with the approximate component. If the contour of the two images is similar, it shows that the detail components gotten from the cloud are correct, and the plain image can be obtained by using the IDWT of the detail components and the approximate component. If the contour is completely different, it means that the cloud deceives the user, and the user can reject the reconstruction results from the cloud.

In short, the proposed cloud-assisted decryption method can not only ensure the authenticity and security of the decrypted image, but also prevent the user from being deceived. Even if other illegal users obtain the reconstructed results from the cloud, they cannot obtain the original image without the index sequences stored in the user.

## 5.10 Discussion

This part shows the advantages of our algorithm comparison with the existing algorithms from two aspects: qualitative analysis and quantitative analysis. The qualitative analysis mainly shows the advantages of the proposed algorithm from two aspects of the security and efficiency of image data transmission, while the quantitative analysis shows the comparison results between our algorithm and the state-of-the-art methods from several indicators of adjacent pixel correlation, information entropy, key space, PSNR and NPCR.

### 5.10.1 Qualitative analysis with other algorithms

The proposed image security transmission algorithm has some advantages.

First of all, some existing image data security algorithms [7, 11, 12, 20, 31, 35, 42, 45, 48, 58, 59] only involve the encryption and decryption of the client and the user, wherein the image is encrypted on the client side, and the decryption is performed on the user side. Although they protect the security of image data, convex optimization computation process of CS reconstruction may require a lot of time, which seriously affects the application of the cryptosystem in real-time fields. In addition, some image security transmission algorithms [51, 52, 54, 57] based on cloud do not consider the work of CS reconstruction on the cloud. In order to solve this problem, in this paper, we designed a secure and efficient cloud-decryption-assisted image compression-encryption based on CS, our scheme not only protects the security



of image data, but also effectively relieves the burden of data storage and calculation, and is suitable for the transmission and storage of large amounts of images on the Internet.

Secondly, there are some shortcomings in some existing image security transmission algorithms based on the cloud. For example, although the algorithms [4, 23, 49] effectively encrypts the image, it lacks the authenticity authentication of the image, which makes it impossible to prevent the malicious tampering of the cloud. That's because when data is stored on the cloud, the cloud may distort the stored image data or return calculation results at will because of laziness or curiosity. In this paper, we judge the authenticity of the image data by the contour information of the detail components and the approximate component obtained by the decryption scheme, which can effectively prevent the tampering of image data stored on the cloud.

Finally, algorithms [51, 52, 54, 57] require two different sampling and reconstruction to ensure the security of CS reconstruction. In this paper, we only use one sampling and reconstruction to achieve the security authentication of image data stored on the cloud, and a lot of time-consuming CS reconstruction process is completed on the cloud, which effectively improves the transmission efficiency of image data. To sum up, compared with other existing algorithms, the proposed algorithms have some predominance.

### 5.10.2 Quantitative analysis with other algorithms

In order to compare with other algorithms conveniently, the image Lena ( $512 \times 512$ ) is chosen as the test image, the information entropy, the correlation of adjacent pixels, PSNR, key space and NPCR are all computed and listed in Table 18.

As depicted from Table 18 that the information entropy gotten by our algorithm is close to the theoretical value of 8, and it is more than the information entropy of algorithms [12, 27, 29, 59], which illustrates the effectiveness of our algorithm. Meanwhile, in addition to the correlation coefficient of adjacent pixels in the vertical direction, those in the other two directions of cipher image are better than those in algorithms [12, 27, 29, 59], which shows that our method can effectively reduce the correlation coefficient between adjacent pixels. The PSNR of our algorithm is better than algorithms [12, 29, 59], which shows that our method has good reconstruction quality. In addition, the key space of the proposed method is better than most algorithms [12, 27, 59], and the NPCR value is higher than algorithm [12], close to algorithms [27, 29, 59]. It shows that the proposed algorithm has a good ability to resist differential attacks. Conclusively, the comparative analyses between our method and other algorithms verify the effectiveness and advantage of our algorithm and it can be applied to the field of image security.

**Table 18** Comparison results with other studies for Lena image

| Algorithm | Correlation coefficients in |          |          | PSNR       | NPCR   | Entropy | Key space  |
|-----------|-----------------------------|----------|----------|------------|--------|---------|------------|
|           | Horizontal                  | Vertical | Diagonal |            |        |         |            |
| Ours      | -0.0018                     | -0.0070  | -0.0024  | 35.3844 dB | 0.9961 | 7.9986  | $2^{372}$  |
| Ref. [59] | 0.0075                      | -0.0015  | -0.0044  | <34 dB     | 0.9963 | 7.9973  | $2^{199}$  |
| Ref. [12] | 0.0018                      | 0.0014   | 0.0034   | <35 dB     | 0.9959 | 7.9986  | $2^{149}$  |
| Ref. [29] | 0.0053                      | 0.0193   | 0.0034   | <34 dB     | 0.9961 | 7.9984  | $10^{135}$ |
| Ref. [27] | 0.0039                      | -0.0027  | 0.0024   | –          | 0.9961 | 7.9969  | $2^{256}$  |

## 6 Conclusion

In this paper, we designed a secure and efficient cloud-decryption-assisted image compression-encryption based on CS and DRPE. Specifically, the original image is firstly transformed into an approximate component and three detail components by DWT, and then the important approximate component is shuffled, and the unimportant detail components are compressed by CS. Subsequently, perform DRPE, random pixel scrambling and diffusion to obtain the final cipher image. In addition, the SHA-256 function of plain image is utilized to generate the initial values of the chaotic systems, the obtained chaotic sequences are used in the encryption stages, ensuring the correlation between the algorithm and plain image. During the decryption process, the partial decryption is firstly manipulated on the cloud to obtain the intermediate cipher image, and then the final decryption is performed on the user to obtain the plain image, which not only saves a lot of time, but also protects user privacy and prevents cloud deception.

Simulation results demonstrate that the proposed algorithm not only reduces the amount of image data transmitted and stored, but also has good security performance, and at the same time can achieve fast decryption, which can be applied in medical, commercial and other fields that require real-time processing of big data images. However, the algorithm designed in this paper also has shortcomings. Among them, it is artificial to judge whether the image stored on the cloud has been tampered by comparing the contour information of the detail component and the approximate component. In the future, it will be more effective to design a contour comparison algorithm to realize this process.

**Acknowledgments** All the authors are deeply grateful to the editors for smooth and fast handling of the manuscript. The authors would also like to thank the anonymous referees for their valuable suggestions to improve the quality of this paper. This work is supported by the National Natural Science Foundation of China (Grant No. 61802111, 61872125, 61871175), Science and Technology Foundation of Henan Province of China (Grant No. 182102210027, 182102410051), China Postdoctoral Science Foundation (Grant No. 2018 T110723), Key Scientific Research Projects for Colleges and Universities of Henan Province (Grant No. 19A413001), Natural Science Foundation of Henan Province (Grant No. 182300410164), Graduate Education Innovation and Quality Improvement Project of Henan University (Grant No. SYL18020105), Henan Higher Education Teaching Reform Research and Practice Project (Graduate Education) (Grant No. 2019SJJGLX080Y), and the Key Science and Technology Project of Henan Province (Grant No. 201300210400, 212102210094).

## Declarations

**Conflict of interest** The authors declare that they have no conflict of interest.

## References

1. Anand A, Raj A, Kohli R, Bibhu V (2016) Proposed symmetric key cryptography algorithm for data security. 2016 international conference on innovation and challenges in cyber security (ICICCS-INBUSH), Noida, pp. 159–162
2. Anwar S, Meghana S (2019) A pixel permutation based image encryption technique using chaotic map. *Multimed Tools Appl* 78(19):27569–27590
3. Arab A, Rostami MJ, Ghavami B (2019) An image encryption method based on chaos system and AES algorithm. *J Supercomput* 75:6663–6682
4. Ashwini K, Amutha R (2018) Fast and secured cloud assisted recovery scheme for compressively sensed signals using new chaotic system. *Multimed Tools Appl* 77(31):31581–31606

5. Candès EJ, Romberg J (2006) Quantitative robust uncertainty principles and optimally sparse decompositions. *Found Comput Math* 6(2):227–254
6. Chai XL, Zheng XY, Gan ZH, Han DJ, Chen YR (2018) An image encryption algorithm based on chaotic system and compressive sensing. *Signal Process* 148:124–144
7. Chai XL, Wu HY, Gan ZH, Zhang YS, Chen YR, Nixon KW (2020) An efficient visually meaningful image compression and encryption scheme based on compressive sensing and dynamic LSB embedding. *Opt lasers Eng* 124: UNSP 105837
8. Chai XL, Bi JQ, Gan ZH, Liu XX, Zhang YS, Chen YR (2020) Color image compression and encryption scheme based on compressive sensing and double random encryption strategy. *Signal Process* 176:107684
9. Chai XL, Zheng XY, Gan ZH, Chen YR (2020) Exploiting plaintext-related mechanism for secure color image encryption. *Neural Comput Applic* 32(12):8065–8088
10. Chai XL, Zhi XC, Gan ZH, Zhang YS, Chen YR, Fu JY (2021) Combining improved genetic algorithm and matrix semi-tensor product (STP) in color image encryption. *Signal Process* 183:108041
11. Chen TH, Zhang M, Wu J, Yuen C, Tong Y (2016) Image encryption and compression based on kronecker compressed sensing and elementary cellular automata scrambling. *Opt Laser Technol* 84:118–133
12. Chen JX, Zhang Y, Qi L, Fu C, Xu LS (2018) Exploiting chaos-based compressed sensing and cryptographic algorithm for image encryption and compression. *Opt Laser Technol* 99:238–248
13. Chen JX, Zhu ZL, Zhang LB, Zhang YS, Yang BQ (2018) Exploiting self-adaptive permutation-diffusion and DNA random encoding for secure and efficient image encryption. *Signal Process* 142:340–353
14. Chen MJ, He XT, Gong LH, Chen RL (2019) Image compression and encryption scheme with hyper-chaotic system and mean error control. *J Mod Opt* 66(13):1416–1424
15. Chen JX, Chen L, Zhang LY, Zhu ZL (2019) Medical image cipher using hierarchical diffusion and non-sequential encryption. *Nonlinear Dynam* 96(1):301–322
16. Darwish SM (2019) A modified image selective encryption-compression technique based on 3D chaotic maps and arithmetic coding. *Multimed Tools Appl* 78(14):19229–19252
17. Donoho DL (2006) Compressed sensing. *IEEE Trans Inform Theory* 52(4):1289–1306
18. Gan ZH, Chai XL, Han DJ, Chen YR (2019) A chaotic image encryption algorithm based on 3-D bit-plane permutation. *Neural Comput Applic* 31(11):7111–7130
19. Ghazvini M, Mirzadi M, Parvar N (2020) A modified method for image encryption based on chaotic map and genetic algorithm. *Multimed Tools Appl* 79(37–38):26927–26950
20. Gong LH, Deng CZ, Pan SM, Zhou NR (2018) Image compression-encryption algorithms by combining hyper-chaotic system with discrete fractional random transform. *Opt Laser Technol* 103:48–58
21. Gong LH, Qiu KD, Deng CZ, Zhou NR (2019) An image compression and encryption algorithm based on chaotic system and compressive sensing. *Opt Laser Technol* 115:257–267
22. Gong LH, Qiu KD, Deng CZ, Zhou NR (2019) An optical image compression and encryption scheme based on compressive sensing and RSA algorithm. *Opt Lasers Eng* 121:169–180
23. Hu GQ, Xiao D, Xiang T, Bai S, Zhang YS (2017) A compressive sensing based privacy preserving outsourcing of image storage and identity authentication service in cloud. *Inf Sci* 387:132–145
24. Hua ZY, Xu BX, Jin F, Huang HJ (2019) Image encryption using Josephus problem and filtering diffusion. *IEEE Access* 7:8660–8674
25. Hua ZY, Zhu ZH, Yi S, Zhang Z, Huang HJ (2021) Cross-plane colour image encryption using a two-dimensional logistic tent modular map. *Inf Sci* 546:1063–1083
26. Jha DP, Kohli R, Gupta A (2016) Proposed encryption algorithm for data security using matrix properties. 2016 international conference on innovation and challenges in cyber security (ICICCS-INBUSH), Noida, pp. 86–90
27. Khedr WI (2020) A new efficient and configurable image encryption structure for secure transmission. *Multimed Tools Appl* 79(23–24):16797–16821
28. Li LL, Xie YY, Liu YZ (2019) Exploiting optical chaos for color image encryption and secure resource sharing in cloud. *IEEE Photonics J* 11(3):1503112
29. Liang YR, Xiao ZY (2020) Image encryption algorithm based on compressive sensing and fractional DCT via polynomial interpolation. *Int J Autom Comput* 17(2):292–304
30. Liu L, Wang LF, Shi YQ, Chang CC (2019) Separable data-hiding scheme for encrypted image to protect privacy of user in cloud. *Symmetry-Basel* 11(1):82
31. Luo YL, Lin J, Liu JX, Wei DQ, Cao LC, Zhou RL, Cao Y, Ding XM (2019) A robust image encryption algorithm based on Chua's circuit and compressive sensing. *Signal Process* 161:227–247
32. Ma S, Zhang Y, Yang ZG, Hu JH, Lei X (2019) A new plaintext-related image encryption scheme based on chaotic sequence. *IEEE Access* 7:30344–30360
33. Pan C, Ye GD, Huang XL, Zhou JW (2019) Novel meaningful image encryption based on block compressive sensing. *Secur Commun Netw* 2019:6572105–6572112

34. Patel U, Dadhanian P (2019) Multilevel data encryption using AES and RSA for image and textual information data. 2019 innovations in power and advanced computing technologies (I-PACT), Vellore
35. Ponuma R, Amutha R (2019) Encryption of image data using compressive sensing and chaotic system. *Multimed Tools Appl* 78(9):11857–11881
36. Som S, Mitra A, Palit S, Chaudhuri BB (2018) A selective bitplane image encryption scheme using chaotic maps. *Multimed Tools Appl* 78(8):10373–10400
37. Song YJ, Zhu ZL, Zhang W, Guo L, Yang X, Yu H (2019) Joint image compression-encryption scheme using entropy coding and compressive sensing. *Nonlinear Dynam* 95(3):2235–2261
38. Telem ANK, Fotsin HB, Kengne J (2021) Image encryption algorithm based on dynamic DNA coding operations and 3D chaotic systems. *Multimed Tools Appl* 80:19011–19041. <https://doi.org/10.1007/s11042-021-10549-0>
39. USC-SIPI (1977) The USC-SIPI Image Database. Available: <http://sipi.usc.edu/database/database.php?volume=misc>. Accessed 20 Mar 2020
40. Wang XY, Gao S (2020) Image encryption algorithm for synchronously updating Boolean networks based on matrix semi-tensor product theory. *Inf Sci* 507:16–36
41. Xian YJ, Wang XY, Yan XP, Li Q, Wang XY (2020) Image encryption based on chaotic sub-block scrambling and chaotic digit selection diffusion. *Opt Lasers Eng* 134:106202
42. Xie YQ, Yu JY, Guo SY, Ding Q, Wang E (2019) Image encryption scheme with compressed sensing based on new three-dimensional chaotic system. *Entropy-Switz* 21(9):819
43. Xu QY, Sun KH, He SB, Zhu CX (2020) An effective image encryption algorithm based on compressive sensing and 2D-SLIM. *Opt Lasers Eng* 134:106178
44. Yang FF, Mou J, Liu J, Ma CG, Yan HZ (2020) Characteristic analysis of the fractional-order hyperchaotic complex system and its image encryption application. *Signal Process* 169:107373
45. Yao SY, Chen LF, Zhong Y (2019) An encryption system for color image based on compressive sensing. *Opt Laser Technol* 120:105703
46. Ye GD, Pan C, Dong YX, Shi Y, Huang XL (2020) Image encryption and hiding algorithm based on compressive sensing and random numbers insertion. *Signal Process* 172:107563
47. Yu SS, Zhou NR, Gong LH, Nie Z (2020) Optical image encryption algorithm based on phase-truncated short-time fractional Fourier transform and hyper-chaotic system. *Opt Lasers Eng* 124:105816
48. Zhang R, Xiao D (2020) A secure image permutation-substitution framework based on chaos and compressive sensing. *Int J Distrib Sens Netw* 16(3) 1550147720912949
49. Zhang YS, Huang H, Xiang Y, Zhang LY, He X (2017) Harnessing the hybrid cloud for secure big image data service. *IEEE Internet Things* 4(5):1380–1388
50. Zhang YS, He Q, Xiang Y, Zhang LY, Liu B, Chen JX, Xie YY (2018) Low-cost and confidentiality-preserving data acquisition for internet of multimedia things. *IEEE Internet Things* 5(5):3442–3451
51. Zhang YS, Xiang Y, Zhang LY, Yang LX, Zhou JT (2019) Efficiently and securely outsourcing compressed sensing reconstruction to a cloud. *Inf Sci* 496:150–160
52. Zhang YS, He Q, Chen G, Zhang XP, Xiang Y (2019) A low-overhead, confidentiality-assured, and authenticated data acquisition framework for IoT. *IEEE Trans Ind Inform* 16(12):7566–7578
53. Zhang H, Wang XQ, Sun YJ, Wang XY (2020) A novel method for lossless image compression and encryption based on LWT, SPIHT and cellular automata. *Signal Process-Image* 84:115829
54. Zhang YS, Wang P, Fang LM, He X, Han H, Chen B (2020) Secure transmission of compressed sampling data using edge clouds. *IEEE Trans Ind Inform* 16(10):6641–6651
55. Zhang Z, Bi HB, Kong XX, Li N, Lu D (2020) Adaptive compressed sensing of color images based on salient region detection. *Multimed Tools Appl* 79(21–22):14777–14791
56. Zhang Y, Chen AG, Tang YJ, Dang JY, Wang GP (2020) Plaintext-related image encryption algorithm based on perceptron-like network. *Inf Sci* 526:180–202
57. Zhang YS, Wang P, Huang H, Zhu YW, Xiao D, Xiang Y (2021) Privacy-assured FogCS: chaotic compressive sensing for secure industrial big image data processing in fog computing. *IEEE Trans Ind Inform* 17(5):3401–3411
58. Zhou NR, Jiang H, Gong LH, Xie XW (2018) Double-image compression and encryption algorithm based on co-sparse representation and random pixel exchanging. *Opt Lasers Eng* 110:72–79
59. Zhou KL, Fan JJ, Fan HJ, Li M (2020) Secure image encryption scheme using double random-phase encoding and compressed sensing. *Opt Laser Technol* 121:105769