

Enhancing visual security: An image encryption scheme based on parallel compressive sensing and edge detection embedding

Yiming Wang(王一铭), Shufeng Huang(黄树锋), Huang Chen(陈煌), Jian Yang(杨健), and Shuting Cai(蔡述庭)
Citation: Chin. Phys. B, 2024, 33 (**1**): 010502. DOI: 10.1088/1674-1056/ad01a1
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

Rucklidge-based memristive chaotic system: Dynamic analysis and image encryption

Can-Ling Jian(蹇璨岭), Ze-An Tian(田泽安), Bo Liang(梁波), Chen-Yang Hu(胡晨阳), Qiao Wang(王桥), and Jing-Xi Chen(陈靖翕)
Chin. Phys. B, 2023, 32 (**10**): 100503. DOI: 10.1088/1674-1056/acdac3

Asymmetric image encryption algorithm based on a new three-dimensional improved logistic chaotic map

Guo-Dong Ye(叶国栋), Hui-Shan Wu(吴惠山), Xiao-Ling Huang(黄小玲), and Syh-Yuan Tan
Chin. Phys. B, 2023, 32 (**3**): 030504. DOI: 10.1088/1674-1056/ac7dbb

A color image encryption algorithm based on hyperchaotic map and DNA mutation

Xinyu Gao(高昕瑜), Bo Sun(孙博), Yinghong Cao(曹颖鸿), Santo Banerjee, and Jun Mou(牟俊)
Chin. Phys. B, 2023, 32 (**3**): 030501. DOI: 10.1088/1674-1056/ac8cdf

Lossless embedding: A visually meaningful image encryption algorithm based on hyperchaos and compressive sensing

Xing-Yuan Wang(王兴元), Xiao-Li Wang(王晓丽), Lin Teng(滕琳), Dong-Hua Jiang(蒋东华), and Yongjin Xian(咸永锦)
Chin. Phys. B, 2023, 32 (**2**): 020503. DOI: 10.1088/1674-1056/aca149

Data encryption based on a 9D complex chaotic system with quaternion for smart grid

Fangfang Zhang(张芳芳), Zhe Huang(黄哲), Lei Kou(寇磊), Yang Li(李扬), Maoyong Cao(曹茂永), and Fengying Ma(马凤英)
Chin. Phys. B, 2023, 32 (**1**): 010502. DOI: 10.1088/1674-1056/ac76b2

Enhancing visual security: An image encryption scheme based on parallel compressive sensing and edge detection embedding

Yiming Wang(王一铭)[1,2], Shufeng Huang(黄树锋)[1], Huang Chen(陈煌)[2], Jian Yang(杨健)[2,†], and Shuting Cai(蔡述庭)[1,‡]

[1] *School of Integrated Circuits, Guangdong University of Technology, Guangzhou* 510006, *China*
[2] *School of Automation, Guangdong University of Technology, Guangzhou* 510006, *China*

(Received 7 July 2023; revised manuscript received 20 September 2023; accepted manuscript online 10 October 2023)

A novel image encryption scheme based on parallel compressive sensing and edge detection embedding technology is proposed to improve visual security. Firstly, the plain image is sparsely represented using the discrete wavelet transform. Then, the coefficient matrix is scrambled and compressed to obtain a size-reduced image using the Fisher–Yates shuffle and parallel compressive sensing. Subsequently, to increase the security of the proposed algorithm, the compressed image is re-encrypted through permutation and diffusion to obtain a noise-like secret image. Finally, an adaptive embedding method based on edge detection for different carrier images is proposed to generate a visually meaningful cipher image. To improve the plaintext sensitivity of the algorithm, the counter mode is combined with the hash function to generate keys for chaotic systems. Additionally, an effective permutation method is designed to scramble the pixels of the compressed image in the re-encryption stage. The simulation results and analyses demonstrate that the proposed algorithm performs well in terms of visual security and decryption quality.

Keywords: visual security, image encryption, parallel compressive sensing, edge detection embedding

PACS: 05.45.Gg, 05.45.–a **DOI:** 10.1088/1674-1056/ad01a1

1. Introduction

Digital images have become a primary means of information storage and transmission in the modern era, playing a critical role in various aspects of daily life. As carriers for information, certain images may contain personal data, trade secrets, or military proprietary information. These images are susceptible to unauthorized tampering and interception when transmitted in an insecure network environment. Therefore, image encryption technology has considerable attention in the field of information security.

Numerous methods have been proposed to protect image security. The most traditional method is to use some encryption techniques to encrypt a plaintext image into cipher images of the same size, such as chaos systems,[1–5] DNA sequences,[6–8] neural networks,[9,10] optical technologies,[11,12] quantum computing,[13,14] and tensor theory.[15,16] In particular, chaotic systems are widely used due to their inherent uncertainty, unpredictability, and sensitivity to initial values. However, existing one-dimensional chaotic systems have drawbacks such as small key space and low complexity, researchers have proposed high-dimensional chaotic systems with better security properties.[17] Moreover, the scrambling-diffusion structure is the most common framework in encryption schemes based on chaotic systems.[18–22]

In recent years, compressive sensing (CS) techniques have been applied to image encryption.[23–27] The CS-based encryption method has the advantages of low computational cost, high security, and versatility. For instance, Xu *et al.*[28] combined the hyperchaotic system with CS. They took row-column scrambling before compression to enhance the compression effect, and obtained cipher images with smaller size and higher security. In addition, some encryption schemes adopt two-dimensional compressive sensing to further reduce the size of the cipher image.[29–31]

The above-mentioned encryption techniques can encrypt images into noise-like images, so the attackers cannot obtain valuable information by analyzing the cipher images. Nevertheless, these encrypted images only provide content security and lack visual security, making them susceptible to interception by hackers when they are transmitted on open networks. In order to tackle this problem, Bao *et al.*[32] first proposed a visually meaningful image encryption algorithm, which encrypted the plain image and then embedded it into the carrier image to generate a visually secure cipher image. However, this method requires the carrier image to be four times the size of the plain image, resulting in higher storage requirement. Later, Chai *et al.*[33] introduced CS to compress the image in the encryption stage before embedding, and obtained the visually secure encrypted image with the same size as the original image. This approach reduces the storage cost while maintaining the image quality.

According to the method of embedding encrypted image into carrier image, visual security image encryption schemes can be classified into two categories. The first method involves frequency domain analysis, such as integer wavelet transform,[34] discrete cosine transform,[35,36] singular value

†Corresponding author. E-mail: yj@gdut.edu.cn
‡Corresponding author. E-mail: shutingcai@gdut.edu.cn

© 2024 Chinese Physical Society and IOP Publishing Ltd

decomposition (SVD),[37,38] and so on. For example, Zhu *et al.*[38] proposed a robust meaningful encryption scheme which uses block CS to encrypt the plain image and embeds into carrier image through SVD. The method of embedding information in the frequency domain has strong robustness and high security. However, the embedding method utilized in this scheme is lossy, which means that the encrypted information cannot be entirely extracted from the cipher image. Additionally, its high computational complexity is unsuitable for practical applications. The second method is to operate in the spatial domain, the information is embedded by approaches such as the least significant bit replacement method[39–41] and pixel value difference.[42] In Ref. [41], Chai *et al.* designed a dynamic LSB embedding method that can completely extract secret information from a cipher image. Although this method can achieve lossless embedding, it needs to be further improved in terms of robustness and visual quality. Moreover, many algorithms rely on fixed-location embedding, which means that the encryption system is not absolutely secure. When multiple images are encrypted by using the same method and one of them is decrypted, the security of the other images is compromised.

To solve the aforementioned issues, a visually secure image encryption scheme based on parallel compressive sensing (PCS) and edge detection embedding is proposed. PCS is a technique that integrates CS with parallel signal acquisition method, which is suitable for sampling by columns in image compression. PCS contributes to efficient compression of large-scale image data by simultaneously collecting data from each column. It not only reduces storage and transmission expenses but also ensures high-quality reconstruction. The edge detector is employed to identify the edges of the carrier image, which is crucial to divide pixels into edge and non-edge areas, enabling adaptive embedding for different carrier images with varying edge distributions.

The general process of encryption is as follows: the compressed image obtained after PCS is permuted and diffused to generate a secret image to resist attacks effectively. Subsequently, guided by the edge-detected carrier image, the pixels of the secret image are shuffled and embedded into the edge and non-edge pixels of the carrier image with various embedding bits. Due to variations in the edges of different carrier images, the embedding positions and bit numbers also differ. This adaptive embedding method enhances the visual security of the cipher image while maintaining high visual quality. The simulation results provide compelling proof of the security and reliability of the proposed scheme. The main contributions are as follows:

1. To effectively resist chosen plaintext attacks (CPA), the counter mode is used to update the keys to generate the measurement matrices. The initial vector of the counter derived from the hash values of the plain image.
2. To thoroughly scramble the pixels of the compressed image, a method of permutation is designed to enhance the randomness of the embedding process.
3. Based on edge detection, an adaptive lossless embedding method is devised, aiming to improve the embedding capacity of the carrier image and the visual security of the cipher image.
4. The simulation analysis demonstrates that the proposed encryption scheme can achieve excellent results in improving compression performance, visual security, and decryption quality under the framework of PCS and edge detection.

The rest of the paper is organized as follows: Section 2 introduces the relevant knowledge. Section 3 describes the proposed image encryption algorithm and its corresponding decryption algorithm in detail. The simulation experiments and performance analysis are conducted in Sections 4 and 5. Furthermore, in Section 6, the proposed algorithm is compared with existing visual image encryption algorithms. Finally, we summarize this paper and prospects for future research in Section 7.

2. Preliminaries

2.1. Chaotic systems

In our proposed algorithm, the chaotic sequence generated by the logistic-tent-sine chaotic system is used to construct the measurement matrix. In addition, the cascading infinite collapse system and the Chebyshev system generate two chaotic sequences to scramble and diffuse the compressed image, and then embed it into the carrier image.

2.1.1. Logistic-tent-sine chaotic system

In Ref. [43], the logistic-tent-sine (LTS) chaotic system is proposed to improve the performance of one-dimensional chaotic systems by providing a larger chaotic range and better chaotic behavior. The definition is as follows:

$$X_{n+1} = \begin{cases} ((rx_n(1-x_n) + rx_n/2) \bmod 1 \\ \quad +(4-r)\sin(\pi x_n)/2) \bmod 1, & x_n < 0.5, \\ ((rx_n(1-x_n) + r(1-x_n)/2) \bmod 1 \\ \quad +(4-r)\sin(\pi x_n)/2) \bmod 1, & x_n \geq 0.5. \end{cases} \tag{1}$$

The values of chaotic sequences generated by the LTS chaotic system are within the range of (0, 1).

2.1.2. Cascading Chebyshev system and infinite collapse system

As is well known, the Chebyshev system and the infinite collapse system are commonly used chaotic systems. However, their chaotic properties and traversal distributions are uneven, making them vulnerable to attacks. To address this

problem, Jiang *et al.*[44] proposed a cascading Chebyshev and infinite collapse system (2D-ICCS), defined as follows:

$$\begin{aligned} x_{n+1} &= \sin\{\pi(\cos[(4-\theta)\cos^{-1}x_n] + \sin\tfrac{\theta}{y_n})\}, \\ y_{n+1} &= \sin\{\pi(\cos[(4-\theta)\cos^{-1}y_n] + \sin\tfrac{\theta}{x_n})\}, \end{aligned} \tag{2}$$

where the control parameters $\theta \in [0,4]$. The outputs of the chaotic system are within the range of $[-1,1]$.

2.2. Parallel compressive sensing

In CS, when a sparse signal is sampled at a rate lower than the Nyquist rate, the original signal experiences minimal distortion during the reconstruction of the sampled data. CS can simultaneously sample, compress, and encrypt images, which has attracted great attention from researchers in image security. To reduce computational complexity and storage space, parallel compressive sensing (PCS) samples and reconstructs a two-dimensional image in a column-wise manner.

In general, natural signals are non-sparse. Therefore, before further processing a signal with CS, it is necessary to transform the signal into the frequency domain to obtain its sparse representation. Suppose that there is a signal $x \in R^{N\times 1}$, under the orthogonal basis $\Psi \in R^{N\times N}$, which can be expressed as

$$x = \sum_{i=1}^{N} \psi_i s_i = \Psi s, \tag{3}$$

where $s_i = \{s_1, s_2, \ldots, s_N\}$ is the coefficient vector. If the number of non-zero coefficients in the vector s is $K \ll N$, that is, the coefficient vector is K-sparse, then the signal x is sparse and compressible under the orthogonal basis Ψ.

For a two-dimensional image $X \in R^{M\times N}$, denoted by $x_i \in R^{N\times 1}$ the i-th column of image X. The image is sampled in a column-wise manner using measurement matrices $\Phi \in R^{m\times N}$, where $m = CR \times M$, CR is the compression ratio. The measurement process of PCS can be expressed as

$$y_i = \Phi_i x_i = \Phi_i \Psi s_i = \Theta_i s_i, \tag{4}$$

where $i = 1,2,\ldots,N$, y_i and Φ_i represent the measurement vector and measurement matrix of the i-th column, respectively, $\Theta = \Phi\Psi$ is the sensor matrix.

Signal reconstruction is also an important link of CS. To recover the original signal x from the observed value y, the sensor matrix Θ needs to satisfy the restricted isometric property (RIP). The reconstruction process of signal x can be expressed as a l_1-norm problem:

$$\min \|s\|_1 \quad \text{s.t.} \quad y = \Theta s, \tag{5}$$

where $\|s\|_1$ represents the l_1-norm of vector s. Several reconstruction algorithms have been proposed to solve the reconstruction problem, such as orthogonal matching tracking (OMP), matching tracking (MP), and smoothing l_0 parametric (SL0). In this paper, we employ the OMP algorithm for reconstructing the image.

2.3. Fisher–Yates shuffle

The Fisher–Yates shuffle algorithm, also known as the Knuth random permutation algorithm, is an effective method for scrambling pseudorandom sequences. The scrambling process can be described as follows: Given a sequence Z of length n, we start from the last element Z_n, and randomly select an element Z_i from the 1st to the n-th element, then swap them. Next, repeat this process for the element Z_{n-1}. We select an element from the 1st to the $n-1$-th elements and swap their positions. This operation is performed iteraively until all elements in this sequence are scrambled.

To provide a clearer illustration of the scrambling process, the sequence 21754386 is used as an example, as shown in Fig. 1.

Scrambled	Select range	Random	Scrambling result
$Z_8 = 6$	1–7	$Z_3 = 7$	21654387
$Z_7 = 8$	1–6	$Z_5 = 4$	21658347
$Z_6 = 3$	1–5	$Z_4 = 5$	21638547
$Z_5 = 8$	1–4	$Z_1 = 2$	81632547
$Z_4 = 3$	1–3	$Z_2 = 1$	83612547
$Z_3 = 6$	1–2	$Z_2 = 3$	86312547
$Z_2 = 6$	1	$Z_1 = 8$	68312547

Fig. 1. Fisher–Yates shuffle process.

2.4. Counter mode

In cryptography, the counter mode is commonly used in the process of packet encryption due to its effectiveness in resisting CPA, which can be described by

$$Ctr_{i+1} = (Ctr_i + 1) \bmod 10^{\lambda}, \tag{6}$$

where $i = 1,2,\ldots,N$, $l \cdot N < 10^{\lambda}$, λ is a security parameter, Ctr_0 is the initial vector, N is the number of counters, and l indicates the number of encrypted images.

2.5. Edge detector

Edge detection is one of the fundamental methods in the field of image processing and computer vision. The purpose of edge detection is to identify points with significant brightness changes in digital images. These obvious image changes are attributed to crucial events or transitions. Through edge detection, the mutation of image intensity is identified, and each pixel is classified as a non-edge pixel or an edge pixel. Several basic edge detectors have been developed, including Sobel, Roberts, Laplacian, Prewitt, and Canny operators. The general criteria for edge detection include: (1) Low error rate. The objective is to identify all edges and minimize the occurrence of false responses. (2) High accuracy. The detected edges should closely match the actual edges. (3) Single edge point response.

Each edge within an image should be marked only once to ensure that the potential noise in this image does not generate false edges.

Among the commonly used edge detection, the Canny edge detection algorithm stands out for its precise definition and reliable performance. It has become one of the most popular algorithms for edge detection due to its ability to satisfy the above three criteria. The Canny edge detection algorithm can be divided into the following steps.

Step 1 Use a two-dimensional Gaussian filter to smooth the image and filter out the noise. The choice of Gaussian kernel size has a significant effect on the performance of the edge detector. A larger kernel size results in less sensitivity to noise and may increase the positioning error in the edge detection. A kernel size of 5×5 is typically suitable for most applications, although the optimal size may vary depending on the specific case.

Step 2 Use the Sobel operator to calculate the magnitude and direction of the image intensity gradient. The Sobel operator is implemented through convolution with two-dimensional space gradient, where S_x represents the Sobel operator in x direction, used to detect edges in y direction, S_y represents the Sobel operator in y direction, used to detect edges in x direction and is defined as follows:

$$S_x=\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix},\quad S_y=\begin{bmatrix}1&2&1\\0&0&0\\-1&-2&1\end{bmatrix}. \tag{7}$$

For a 3×3 window A in the image, the gradient of pixel e can be calculated after convolution calculation with the Sobel operator. The gradients of pixel e in the x and y directions are expressed as follows:

$$\begin{aligned}G_x&=S_x*A=\mathrm{sum}\left(\begin{bmatrix}-a&0&c\\-2d&0&2f\\-g&0&i\end{bmatrix}\right),\\G_y&=S_y*A=\mathrm{sum}\left(\begin{bmatrix}a&2b&c\\0&0&0\\-g&-2h&-i\end{bmatrix}\right),\end{aligned} \tag{8}$$

where $*$ is the convolution symbol, and the summations are taken over all elements in the matrix. Then we can calculate the gradient and direction of pixel e according to

$$G=\sqrt{G_x^2+G_y^2},\quad \theta=\arctan\left(\frac{G_y}{G_x}\right). \tag{9}$$

Step 3 To eliminate spurious responses in edge detection, non-maximum suppression techniques are employed. Non-maximum suppression is an edge refinement technique that refines blurred edges after gradient computation. It effectively suppresses all gradient values except local maximum to ensure that each edge has an accurate response. This method compares the gradient intensity of each pixel with its positive and negative gradient direction pixels. When the current pixel has the most significant gradient intensity, it is marked as an edge pixel. Otherwise, it is marked as a non-edge pixel.

Step 4 Perform dual-threshold technique to identify potential edges. After non-maximum suppression, the remaining pixels can better represent the actual edges in the image. However, there may still be edge pixels caused by noise and color variations. To address these false responses, a dual-threshold technique is applied. It involves filtering the edge pixels using a high threshold Th_{h} and a low threshold Th_{l}. Pixels with gradient values higher than Th_{h} are labeled as strong edge pixels, and pixels within the threshold range $[Th_{\mathrm{l}},Th_{\mathrm{h}}]$ are identified as weak edge pixels. Pixels with gradient values lower than Th_{l} are suppressed. The choice of thresholds depends on the input image.

Step 5 Suppress isolated low threshold points. Up to this point, the strong edge pixels are easily recognized, while the weak edge pixels may be mistaken for noise or color changes. The weak edge pixels and their eight adjacent pixels are examined to identify actual edges. If at least one of the adjacent pixels is a strong edge pixel, the weak edge pixel can be identified as an authentic edge.

The above five steps can complete the image edge detection based on the Canny algorithm, and Fig. 2 shows the detection effect of the algorithm.

3. Visual security image encryption scheme

3.1. Keys generation

In our scheme, the SHA-256 hash function values of the plain image are used with the external keys $K=(x',r',y',z',\mu')$ to calculate the internal keys. Specifically, the initial values and parameters of the LTS chaotic system are updated by the counters with x_0 and r_0. The 2D-ICCS is iterated with initial values y_0, z_0, and parameter μ to generate chaotic sequences for permutation, diffusion, and embedding. The specific steps are as follows.

Step 1 Obtain the SHA-256 hash values of the plain image. Subsequently, segment the hash values into 8-bit blocks and convert them to decimal digits $h_1,h_2,\ldots,h_{32}$.

Step 2 The internal keys are computed as follows:

$$\begin{cases}x_0=\frac{\mathrm{sum}(h(1:8))}{256}\bmod1+x',\\r_0=\frac{\mathrm{sum}(h(9:16))}{256}\bmod4+r',\\y_0=\frac{\mathrm{sum}(h(17:24))}{256}\bmod1+y',\\z_0=\frac{\mathrm{sum}(h(25:32))}{256}\bmod1+z',\\\mu=\frac{\mathrm{sum}(h(4:4:32))}{256}\bmod4+\mu'.\end{cases} \tag{10}$$

Step 3 To construct the measurement matrices, the counter code is used to update the initial values x_i and parameter r_i of the chaotic system, according to the following

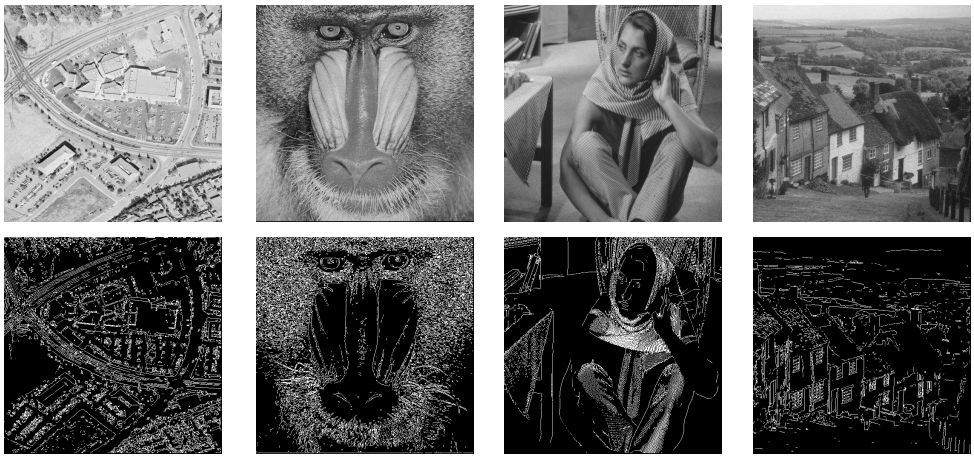
equations:

$$\begin{cases} r_i = (Ctr_i \times 10^{-\lambda} + r_0) \bmod 4, \\ x_i = (Ctr_i \times 10^{-\lambda} + x_0) \bmod 1. \end{cases} \tag{11}$$

3.2. Encryption process

The flow chart of the proposed visual image encryption algorithm is shown in Fig. 3. First, the plain image is pre-encrypted and compressed by the Fisher–Yates shuffle and PCS. Then, perform the permutation and diffusion operation to obtain the secret image. Finally, the secret image is embedded into the carrier image by using the edge detection technique.

Assume that the plain image is $P \in R^{M\times N}$, and the carrier image is $CI \in R^{M\times N}$, the detailed encryption process is shown in the following.



(a) (b) (c) (d)

Fig. 2. Plain images and the corresponding edge images: (a)–(d) are plain images of Aerial, Baboon, Barbara and Goldhill, and their corresponding edge images.

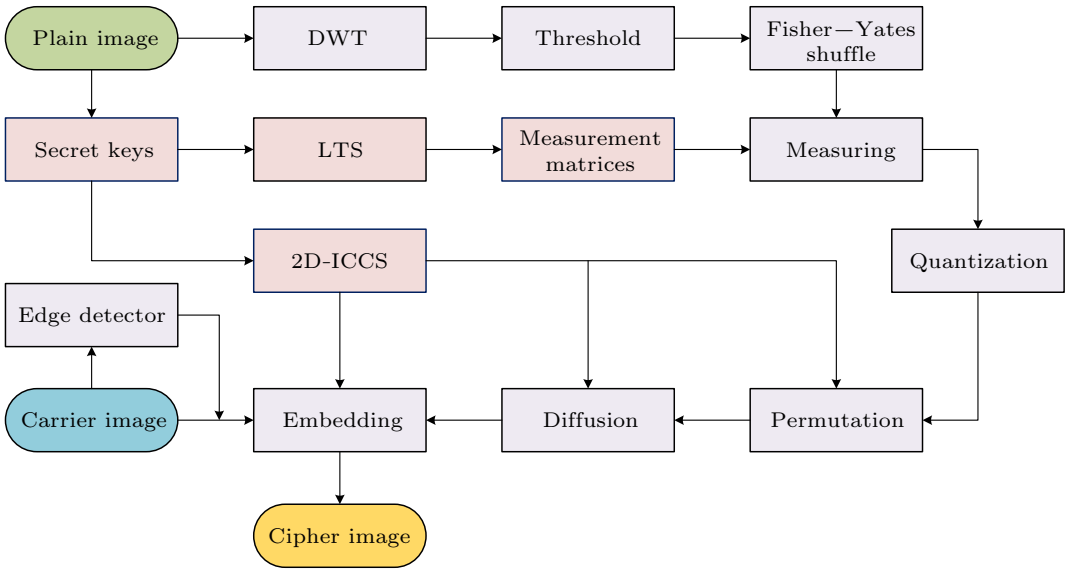


Fig. 3. The flow chart of the proposed image encryption algorithm.

3.2.1. Compression process

Step 1 The sparse basis $\Psi \in R^{N\times M}$ is obtained via the discrete wavelet function (DWT), and transform the plain image P to obtain the sparse coefficients matrix $P1$:

$$P1 = \Psi \times P \times \Psi^{\mathrm{T}}. \tag{12}$$

Step 2 Use Fisher–Yates shuffle to scramble the pixels of the image $P1$. First, arrange $P1$ into a one-dimensional sequence, then use the method described in Subsection 2.3 to scramble it, and the subscripts of randomly selected elements are recorded in the index sequence *order*. Finally, the scrambled sequence is rearranged into a two-dimensional matrix $P2$.

Step 3 Assign the threshold TS to improve the quality of the reconstructed image. Set the values in matrix $P2$ with absolute values less than or equal to TS to zero, resulting in matrix $P3$.

Step 4 Construct measurement matrices $\Phi_i \in R^{m\times N}$, where $i = 1,2,\ldots,N$. The LTS chaotic system is iterated $m \times N + t_0$ times using the initial value x_i and parameter r_i, and discard the former t_0 elements to obtain the chaotic sequence

$W_i = \{w_j\}_1^{mN}$. The measurement matrices are obtained by executing the following equation:

$$\Phi_i = \sqrt{\frac{2}{M}} \begin{bmatrix} w_1 & w_{m+1} & \cdots & w_{mN-m+1} \\ w_2 & w_{m+2} & \cdots & w_{mN-m+2} \\ \vdots & \vdots & \ddots & \vdots \\ w_m & w_{2m} & \cdots & w_{mN} \end{bmatrix}. \tag{13}$$

Step 5 Apply the measurement matrix Φ_i to the corresponding columns of $P3$ to generate the measurement value sequence, and combine all the sequences to obtain the matrix $P4 \in R^{m\times N}$, the process can be expressed as

$$P4(:,i) = \Phi_i \times P3(:,i). \tag{14}$$

Step 6 For the subsequent embedding operation, the image $P4$ is quantified and mapped to the range [0,255] to obtain image $P5$,

$$P5 = \text{floor}\left(\frac{255 \times (P4 - \min)}{\max - \min}\right), \tag{15}$$

where min and max are the minimum and maximum of the matrix $P4$.

3.2.2. Permutation and diffusion

To augment the confidentiality and randomness of the scheme, the compressed image is permuted and diffused. The specific process is as follows:

Step 1 Generate two chaotic sequences by iterating the 2D-ICCS with the initial values y_0, z_0 and parameter μ, taking $m \times N$ elements after discarding the former t_0 elements, as follows:

$$\begin{cases} Y = \{y_1, y_2, y_3, \ldots, y_{m\times N}\}, \\ Z = \{z_1, z_2, z_3, \ldots, z_{m\times N}\}. \end{cases} \tag{16}$$

Step 2 Arrange the sequences Y and Z into two-dimensional matrices $PY \in R^{m\times N}$, $PZ \in R^{m\times N}$. Sort the elements in each column of in ascending order to obtain the index sort matrix PY'.

Step 3 Combine the elements of the indexed sort matrix PY' and their corresponding column coordinate to obtain the coordinate matrix $O_{i,j} = O(PY'_{i,j}, j)$, where $i = 1,2,\ldots,m$, $j = 1,2,\ldots,N$.

Step 4 Set the current row number $r = 1$.

Step 5 According to the coordinates in the r-th row of the coordinates matrix $O_{i,j}$, seek $\{PY_{O_{r,1}}, PY_{O_{r,2}}, \ldots, PY_{O_{r,N}}\}$ from the two-dimensional matrix PZ, ascending order to get index sequence $I_{O_{r,j}}$. The process can be expressed as

$$\left\{[PY'_{O_{r,j}}, I_{O_{r,j}}] = \text{sort}(PY_{O_{r,j}})\right\}_{j=1}^{N}. \tag{17}$$

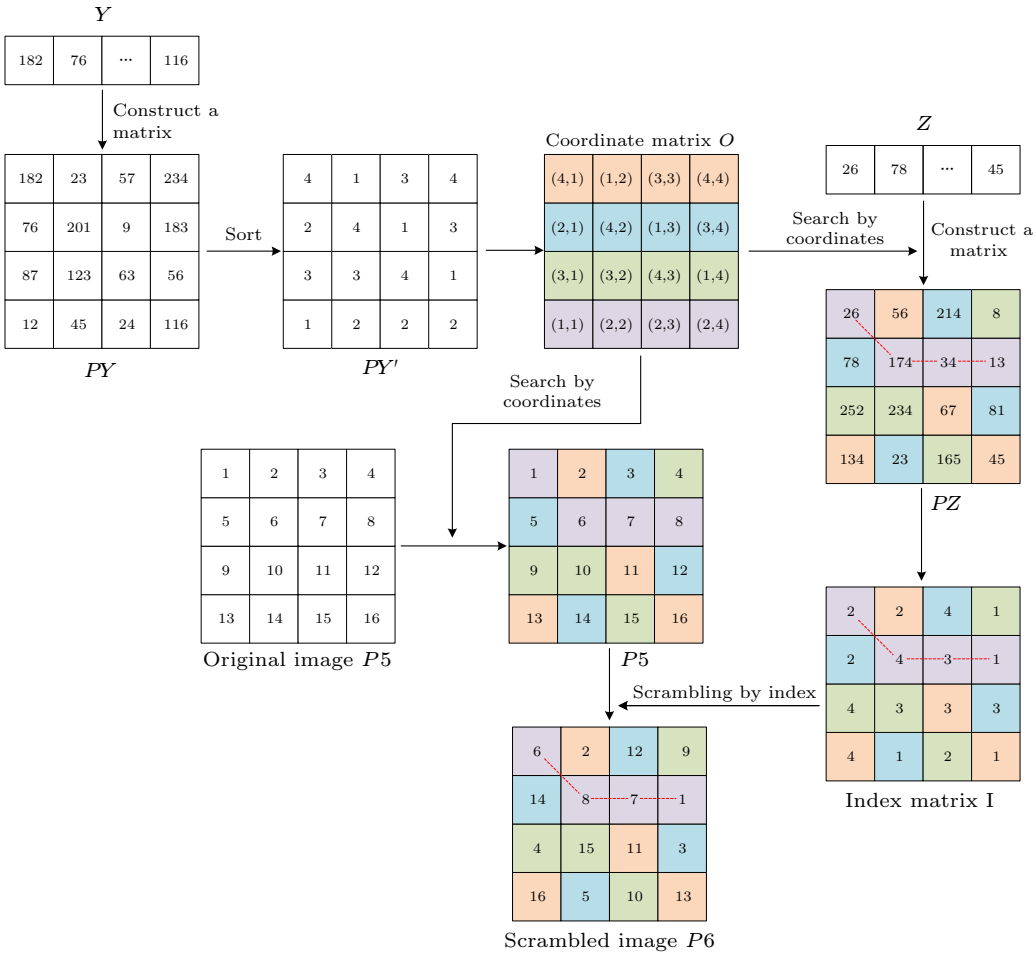


Fig. 4. The flow chart of the designed permutation.

Step 6 According to the coordinates in the r-th row of the coordinates matrix $O_{i,j}$, seek $\{P5_{O_{r,1}}, P5_{O_{r,2}}, \ldots, P5_{O_{r,N}}\}$ from the two-dimensional matrix $P5$, sort them according to index sequence $I_{O_{r,j}}$.

Step 7 Set the row number $r = r+1$. Repeat Steps 5 and 6 until $r = m$ to obtain the scrambled image $P6$. To clearly illustrate the permutation process described above, we use a 4×4 matrix as an example, as shown in Fig. 4.

Step 8 The chaotic sequences Y and Z are transformed as follows:

$$\begin{cases} T1(i) = \text{floor}(Y(i) \times 2^{16}) \bmod 256, \\ T2(i) = \text{floor}(Z(i) \times 2^{16}) \bmod 256. \end{cases} \tag{18}$$

The sequences $T1$ and $T2$ are applied for forward and reverse diffusion, respectively, and the diffusion process can be described as

$$\begin{cases} P7(i) = P7(i-1) \oplus T1(i) \oplus P6(i), \\ SI(i) = SI(i+1) \oplus T2(i) \oplus P7(i), \end{cases} \tag{19}$$

where $\oplus$ denotes the XOR operation. We can get the secret image SI.

3.2.3. Embedding process

Step 1 Use the Canny edge detector to detect the edges of the carrier image through the process described in Subsection 2.5, and the edge image $EI \in R^{M \times N}$ is obtained.

Step 2 Aiming to avoid possible overflow, the grayscale value of the carrier image CI is adjusted to the range [15, 240] to obtain the image CI' by the following equation:

$$CI' = \left\lceil a + \frac{b-a}{255} \times CI \right\rceil, \tag{20}$$

where $a = 15$ and $b = 240$.

Step 3 Based on the edge image EI, the pixels of image CI' are divided into edge and non-edge parts. Then, arrange them into a one-dimensional array $C_1 = [U, V]$ with edge pixels first, followed by non-edge pixels. Suppose that the number of edge pixels U is n, and the number of non-edge pixels V is $M \times N - n$, denoted as $U = \{u_1, u_2, \ldots, u_n\}$ and $V = \{v_1, v_2, \ldots, v_{M \times N - n}\}$.

Step 4 Take the former n elements of the chaotic sequence Y and the former $M \times N - n$ elements of sequence Z, and sort them in ascending order to obtain index sequences sy and sz. Scramble Y and Z in the order of sy and sz to get $U' = \{u'_1, u'_2, \ldots, u'_n\}$ and $V' = \{v'_1, v'_2, \ldots, v'_{M \times N - n}\}$, and then combine them into a scrambled array $C_2 = [U', V']$.

Step 5 All elements of the secret image SI are converted to eight-bit binary and stretched into a one-dimensional secret bits array $S = \{s_1, s_2, \ldots, s_{8mN}\}$ of length $8 \times m \times N$. Each element of the array C_2 is also represented by an eight-bit binary. Then, each group of four secret bits is embedded in the lowest four bits of each edge pixel u'_i, and the remaining $8mN - 4n$ secret bits are embedded in the lowest two bits of non-edge pixels v'_i sequentially. Finally, convert pixels in binary format to decimal. It is worth noting that certain non-edge pixels are devoid of any embedded secret bits.

Step 6 After embedding operation, scramble the U' and V' according to the reverse order of the index sequence sy and sz to obtain U'' and V'', and then get the array $C_3 = [U'', V'']$.

Step 7 The array C_3 is rearranged into a two-dimensional matrix to obtain the final visually meaningful cipher image VI. The encryption process is completed.

3.3. Decryption process

To decrypt the cipher image, the keys are transferred to the decryption side, including the compression ratio CR, the initial vector Ctr_0 of the counter, the parameters and initial values $\{x_0, r_0, y_0, z_0, \mu_0\}$ of chaotic systems. The flow chart of the decryption algorithm is shown in Fig. 5. The specific decryption process is as follows:

Step 1 Perform edge detection on the carrier image CI. Then, the pixel positions of the cipher image VI correspond to the detection result, divided into edge and non-edge segments. Extract the edge pixels U'' are arranged in the front and the non-edge pixels V'' are arranged in the back to form a one-dimensional array.

Step 2 Iterate the 2D-ICCS to obtain two chaotic sequences Y and Z, and take the first n and the first $M \times N - n$ elements, respectively. Arrange them in ascending order to obtain the index sequences sy and sz, scrambling U'' and V'' to obtain U' and V'.

Step 3 The elements in U' and V' are converted to eight-bit binary, then the lowest four bits of each element u'_i in U' and the lowest two bits of the element v'_i in V' are extracted and arranged into an array in turn. Take the first $8 \times m \times N$ elements to construct the secret bit array S, and convert it into decimal in a group of each eight elements. Finally, the array of length $m \times N$ is rearranged into a two-dimensional matrix to acquire the secret image SI.

Step 4 The chaotic sequences $T1$ and $T2$ are used for the inverse forward and backward diffusion according to the following equations:

$$\begin{cases} P7(i) = SI(i+1) \oplus T2(i) \oplus SI(i), \\ P6(i) = P7(i-1) \oplus T1(i) \oplus P7(i). \end{cases} \tag{21}$$

Step 5 The index matrix I is obtained according to Section 3.2.2 for implementing the inverse permutation to generate the matrix $P5$.

Step 6 Execute the inverse quantization on the matrix $P5$.

$$P4 = \frac{P5 \times (\max - \min)}{255} + \min. \tag{22}$$

Step 7 Calculate the measurement matrix Φ_i, and then use OMP to reconstruct the matrix $P3$, the process is shown as follows:

$$P3(:, i) = \text{OMP}(P4(:, i), \Phi_i). \tag{23}$$

Step 8 According the index sequence *order*, perform the inverse Fisher–Yates shuffle to obtain the sparse coefficient matrix $P2$.

Step 9 The decrypted image P is obtained by applying inverse discrete wavelet transform (IDWT) on $P2$, and can be expressed as

$$P = \Psi^{\mathrm{T}} \times P2 \times \Psi. \tag{24}$$

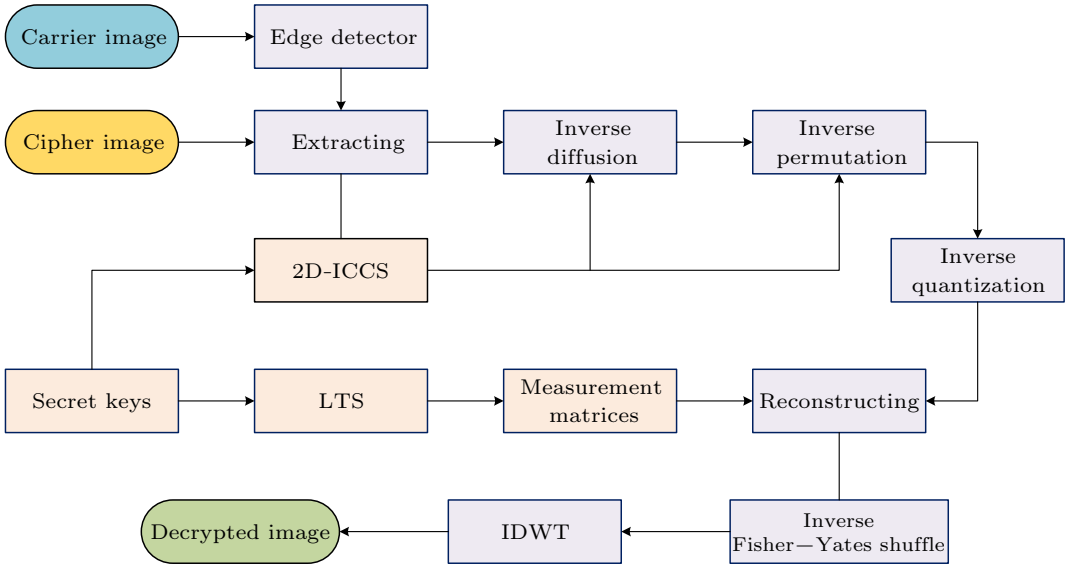


Fig. 5. The flow chart of the proposed image decryption algorithm.

4. Simulation results

4.1. Encryption and decryption results

To verify the effectiveness of the proposed encryption algorithm, the simulation experiments are implemented on Matlab R2018a. The hardware condition is Intel(R) Core (TM)i5-12500 CPU @ 3.00 GHz with 16.00 GB RAM, and the operating system is Microsoft Windows 10. The 512×512 sized images including Lena, Girl, Peppers, and Jet are selected as the plain images, and the same size images including Baboon, Goldhill, Aerial, and Bridge are selected as carrier images for testing. The keys are set as $x' = 0.6107$, $r' = 0.8915$, $y' = 0.3109$, $z' = 0.1681$, $\mu' = 0.3057$, and other parameters are $CR = 0.25$, $TS = 30$. In the decryption process, OMP is chosen as the reconstruction method.

The simulation results are shown in Fig. 6, where from the first to the fifth column are listed the plain image, secret image, carrier image, secret image, cipher image, and decrypted image. Notably, the noise-like secret image is one-fourth the size of the plain image and cannot provide any helpful information, proving the effectiveness of the encryption and compression of the scheme. Meanwhile, the generated steganographic image has the same visual effect as the carrier image, which indicates that the proposed encryption scheme has visual security and can protect the secret information effectively. Additionally, the decrypted image is similar to the plain image, and since the embedding process is entirely reversible, it is demonstrated that the reconstruction quality is satisfactory.

To quantitatively assess the performance of the proposed encryption scheme, this paper applies the peak signal-to-noise ratio (PSNR) and mean structural similarity index measure (MSSIM) to compare the similarity between the carrier image and the cipher image as well as to measure the quality of the decrypted image. Their equations are given as follows:

$$\mathrm{PSNR} = 10\log_{10}\left[\frac{255^2}{\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}[X(i,j)-Y(i,j)]^2}\right], \tag{25}$$

$$\mathrm{SSIM} = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2)(\sigma_X^2 + \sigma_Y^2 + C_2)}, \tag{26}$$

$$\mathrm{MSSIM} = \frac{1}{N}\sum_{k=1}^{N}\mathrm{SSIM}(X_k - Y_k), \tag{27}$$

where $C_1 = (k_1 \times L)$, $C_2 = (k_2 \times L)$, $k_1 = 0.01$, $k_2 = 0.01$, $L = 255$; σ_X, σ_Y, μ_X, μ_Y, and σ_{XY} denote the variance, average and covariance values of the plain image and the decrypted image, respectively, and N is the number of the image blocks.

The test results of PSNR and MSSIM are listed in Table 1. Numerically, all $\mathrm{PSNR}_{\mathrm{cip}}$ values are greater than 40 dB, the values of $\mathrm{PSNR}_{\mathrm{dec}}$ are about 35 dB, and the values of MSSIM are close to 1. The above results demonstrate that our scheme can encrypt the plain image into a visually meaningful steganographic image, and the decrypted image maintains high quality.

4.2. The relationship between decrypted images and different carrier images

Considering the effect of different carrier images on the quality of cipher image and decrypted image is helpful to evaluate the performance of encryption schemes. The 512×512 sized image is embedded into five different carrier images, i.e.,

Goldhill, Barbara, Baboon, Aerial, and Bridge. The simulation results are shown in Fig. 7. The influence of the carrier image on the decryption results is assessed by calculating the PSNR and MSSIM values of the decrypted image. It can be seen from Table 2, all the PSNR_{dec} are about 36 dB, indicating that the different carrier images have almost no effect on the quality of the decrypted images. It needs to be explained that the slight numerical differences are due to the randomness of the Fisher–Yates shuffling process, even though the embedding process is reversible.

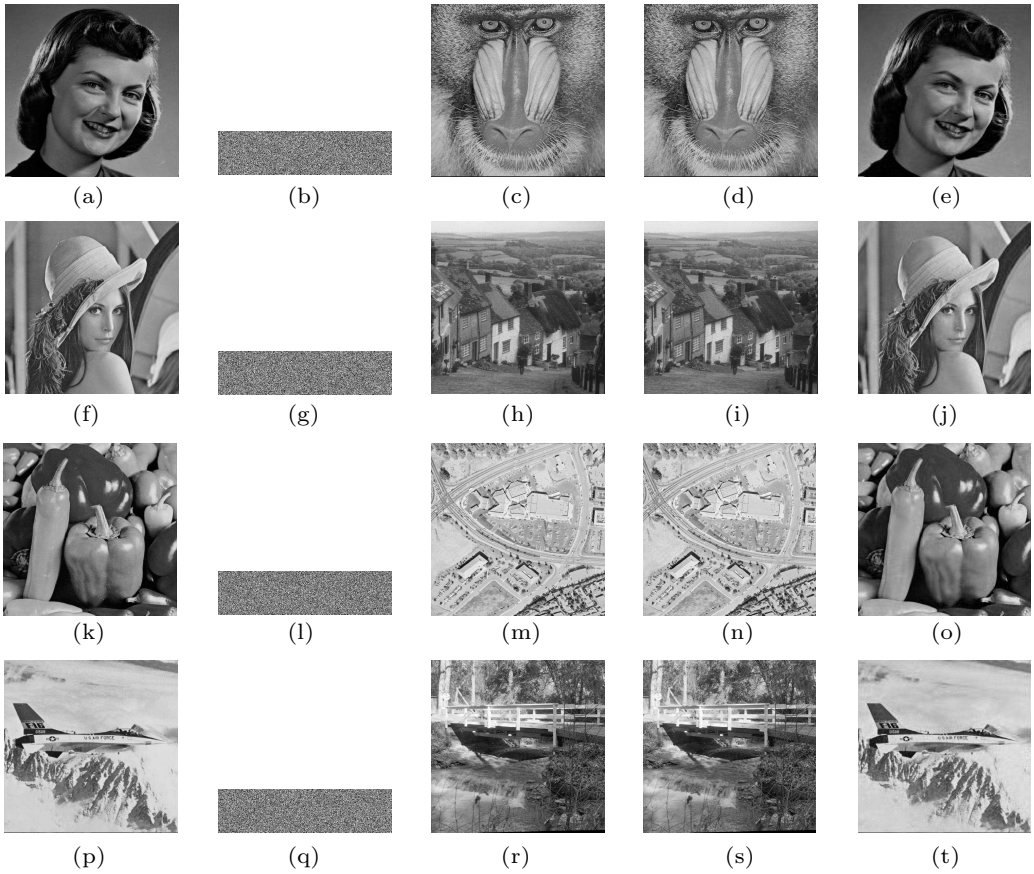


Fig. 6. Simulation results: [(a), (f), (k), (p)] the plain images Girl, Lena, Peppers and Jet; [(b), (g), (l), (q)] the corresponding secret images; [(c), (h), (m), (r)] the carrier images of Baboon, Goldhill, Aerial and Bridge; [(d), (i), (n), (s)] the corresponding cipher images; [(e), (j), (o), (t)] the corresponding decrypted images.

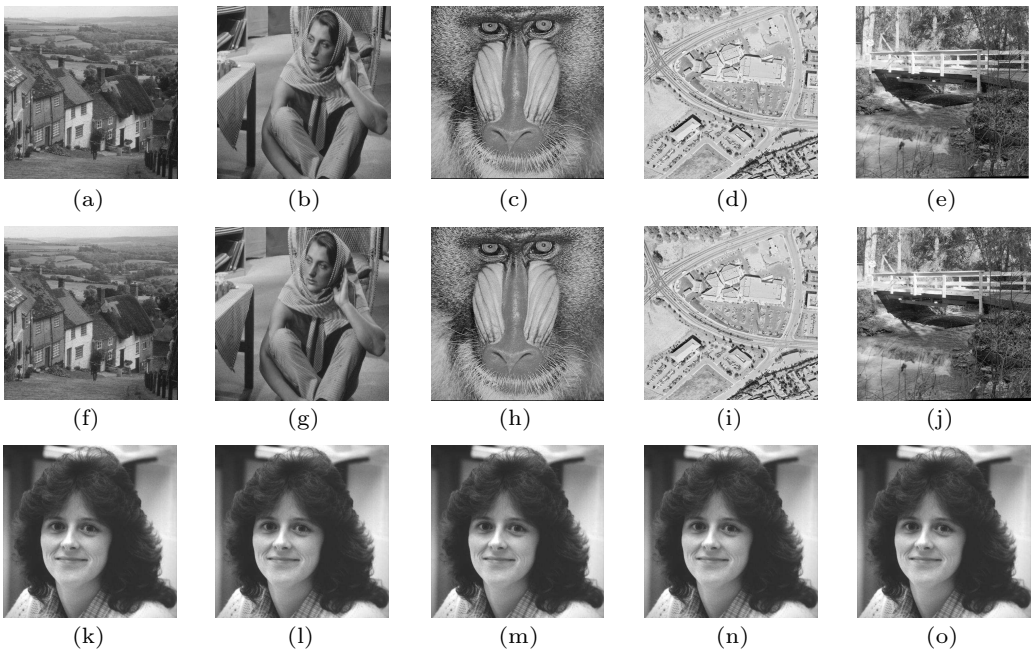


Fig. 7. Simulation results with the different carrier images: (a)–(e) the carrier images of Goldhill, Barbara, Baboon, Aerial, and Bridge; (f)–(j) the corresponding cipher images; (k)–(o) the corresponding decrypted images.

Table 1. PSNR and MSSIM values of simulation results. $PSNR_{cip}$ and MSSIM are used to compare the similarity between the cipher image and the carrier image, and $PSNR_{dec}$ is employed to quantify the quality of the decrypted image.

Plain image	Carrier image	$PSNR_{dec}$	$PSNR_{cip}$	MSSIM
Girl	Baboon	36.4741	41.0339	0.9896
Lena	Goldhill	35.9406	44.0768	0.9862
Peppers	Aerial	35.7072	42.1647	0.9867
Jet	Bridge	34.2136	41.9024	0.9906

Table 2. PSNR and MSSIM values of the different carrier images.

Plain image	Carrier image	$PSNR_{dec}$	$PSNR_{cip}$	MSSIM
Woman	Goldhill	36.2437	44.0634	0.9839
	Barbara	36.0166	42.7982	0.9837
	Baboon	35.9616	41.0531	0.9895
	Aerial	36.3135	42.2263	0.9867
	Bridge	36.3691	41.7492	0.9906

4.3. The influence of TS and CR

In the proposed encryption scheme, the choice of threshold value TS has a direct impact on the quality of the decrypted image. We conduct an evaluation using four 512×512 sized plain images, including Peppers, Barbara, Lena, and Boat, as depicted in Fig. 8. As can be observed, when TS is less than 30, the quality of the decrypted image is improved as TS increases. However, when TS exceeds 30, the quality starts to degrade. Therefore, the threshold value should be set to 30 for optimal decryption performance.

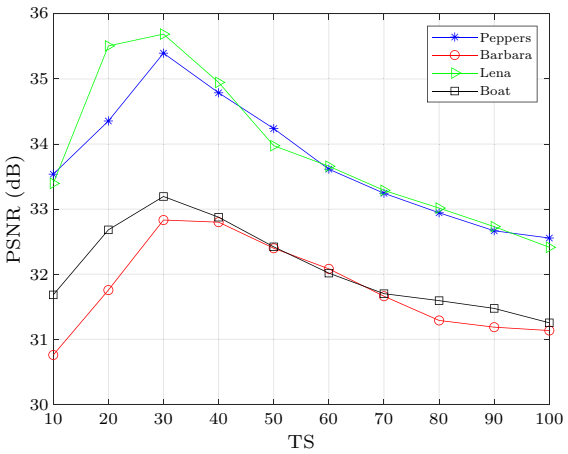


Fig. 8. Influence of TS changes on $PSNR_{dec}$.

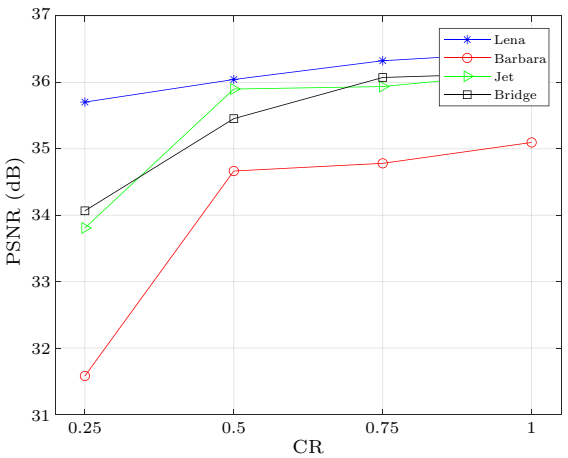


Fig. 9. Influence of CR changes on $PSNR_{dec}$.

Meanwhile, the quality of the decrypted image is also related to the compression ratio CR. In the PCS-based compression-encryption process, to assess the impact of CR on the decryption results, different compression ratios are set to compress the plain images. Subsequently, the generated secret images are decrypted and reconstructed for analysis. The values of CR are set to 0.25, 0.5, and 0.75, respectively. Figure 9 shows the experimental results.

5. Performance analysis

5.1. Key space analysis

The security of an image encryption system is proportional to its key space. Expanding key space enhances the ability of the system to resist brute-force attacks. Typically, the key space for the image encryption scheme should exceed 2^{100}. The keys of the proposed encryption scheme consist of the hash values and the initial keys $\{x', r', y', z', \mu'\}$. Assume that the calculation accuracy of the computer is 10^{-14}, and the size of the key space is $2^{256} \times (10^{14})^5 > 2^{100}$. Moreover, if parameters such as Ctr_0 and TS are also considered as keys, the value of key space will become larger. The results indicate that the key space of the scheme is sufficiently large to resist brute-force attacks.

5.2. Key sensitivity analysis

Key sensitivity is one of the crucial parameters for evaluating the effectiveness of an image encryption system against differential attacks. When slight modifications to the decryption key result in a decrypted image that resembles noise, it indicates that the scheme has good key sensitivity. The differences among decrypted images can be quantified by using the number of pixel change rate (NPCR) and unified average change intensity (UACI), defined as

$$\mathrm{NPCR} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j1}^{N} D(i,j) \times 100\%, \tag{28}$$

$$\mathrm{UACI} = \frac{1}{M \times N} \left(\sum_{i=1}^{M} \sum_{j=1}^{N} \frac{|C_1(i,j) - C_2(i,j)|}{255} \right) \times 100\%, \tag{29}$$

where $D(i,j) = \begin{cases} 0, & C_1(i,j) = C_2(i,j), \\ 1, & C_1(i,j) \neq C_2(i,j), \end{cases}$ C_1 and C_2 are two encrypted images with a size of $M \times N$.

We utilize the 512×512 sized image Peppers as plain image and Barbara as the carrier image for test. The initial keys are set to $k = (x', r', y', z', \mu')$, and then each parameter is slightly modified to generate the incorrect keys as $k_1 = (x' + 10^{-14}, r', y', z', \mu')$, $k_2 = (x', r' + 10^{-14}, y', z', \mu')$, $k_3 = (x', r', y' + 10^{-14}, z', \mu')$, $k_4 = (x', r', y', z' + 10^{-14}, \mu')$, $k_5 = (x', r', y', z', \mu' + 10^{-14})$. As shown in Fig. 10, after decrypting the image using the correct keys and the modified keys respectively, it is observed that the decrypted image with the modified key resembles a noisy image. Table 3 lists the NPCR and NACI values of the cipher images with the correct key and the modified keys.

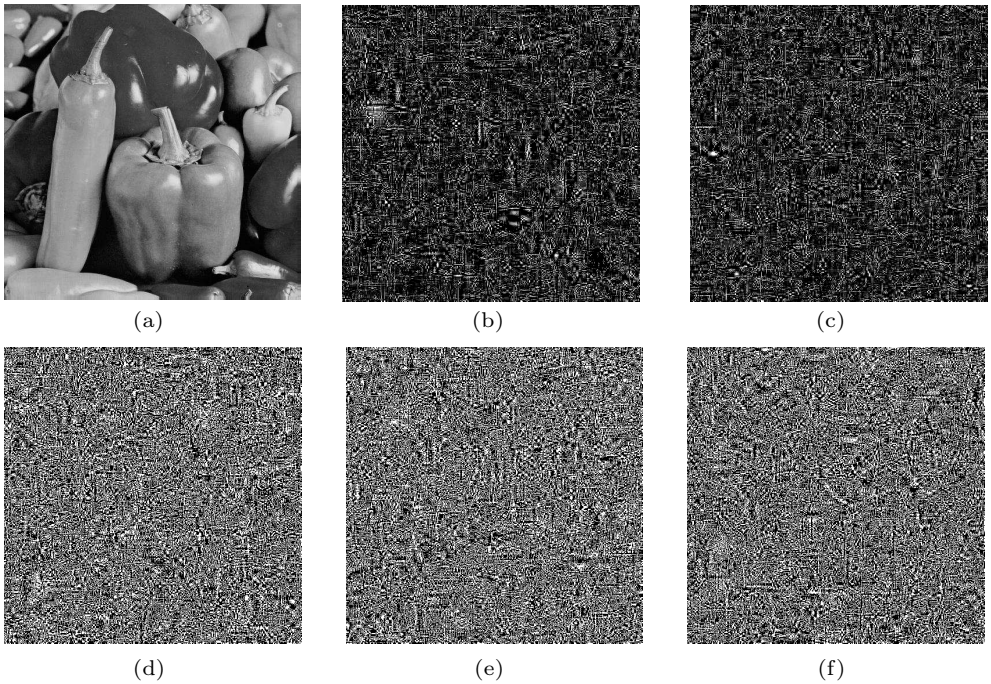


Fig. 10. Key sensitivity analysis: (a) the decrypted image with the correct key; (b) the decrypted image with k_1; (c) the decrypted image with k_2; (d) the decrypted image with k_3; (e) the decrypted image with k_4; (f) the decrypted image with k_5.

Table 3. The NPCR and UACI of the modified keys.

Modified key	NPCR(%)	UACI(%)
$x' + 10^{-14}$	99.8497	33.5068
$r' + 10^{-14}$	99.8417	33.4676
$y' + 10^{-14}$	99.9348	33.5028
$z' + 10^{-14}$	99.9344	33.4728
$\mu' + 10^{-14}$	99.8313	33.4565

5.3. Plaintext sensitivity analysis

The strength of plaintext sensitivity measures the impact of minor changes in the original image on the encrypted image. When the image encryption algorithm exhibits strong plaintext sensitivity, small changes in the original image can lead to significant alterations in the encrypted image, which makes it challenging to predict and analyze the cipher image. To assess plaintext sensitivity, the original image is encrypted to obtain the secret image. Then, according to the equation below, we select and modify the pixel at a random position (x, y) in the original image. Subsequently, the slightly modified image is encrypted using the same key. Four different 256×256 images, four different 512×512 images, and three different 1024×1024 images are tested for 200 times and the average values are shown in Table 4. Clearly, all test values are very close to the ideal values. The proposed method is compared with others listed in Table 5, demonstrating its effectiveness in resisting various attacks.

$$\begin{cases} x = \text{floor}(\text{rand} \times 10^8) \bmod 256 + 1, \\ y = \text{floor}(\text{rand} \times 10^8) \bmod 256 + 1, \end{cases} \tag{30}$$

$$P(x,y) = P(x,y) + 1 \bmod 256. \tag{31}$$

Table 4. The NPCR and UACI values of secret images encrypted from images of different sizes.

Image size	Image	NPCR(%)	UACI(%)
256×256	Lena	99.6321	33.5112
	Peppers	99.6077	33.3046
	Woman	99.6399	33.4297
	Cameraman	99.6015	33.5550
512×512	Lena	99.6037	33.3968
	Peppers	99.6229	33.4736
	Woman	99.6155	33.3805
	Cameraman	99.6078	33.4220
1024×1024	Man	99.6254	33.6121
	Airplane	99.6159	33.5170
	Airport	99.6078	33.4220

5.4. Histogram analysis

The pixel intensity values in an image can be intuitively visualized by using the histogram, which illustrates the fre-

quency of each pixel value. An ideal image encryption algorithm should generate a secret image with a uniform histogram to effectively hide image information. Moreover, the proposed algorithm generates a cipher image with visual security, so the difference between the carrier image and the cipher image should be minimized. As shown in Fig. 11, we can observe that the histograms of the secret images are uniformly distributed, and the histograms of the carrier images and the cipher images are indeed similar. This further confirms the effectiveness and robustness of the algorithm.

5.5. Information entropy

Information entropy reflects the uncertainty of image information. For a 256 grayscale level, the ideal information entropy value is 8, indicating low information visibility and superior encryption performance. The information entropy is calculated according to

$$H(m) = -\sum_{i=1}^{N} p(m_i) \log p(m_i), \tag{32}$$

where $p(m_i)$ represents the probability of m_i, and N represents the gray scale level. In the test, four images are chosen for testing, including Lena, Peppers, Baboon, and Boat. Table 6 shows the information entropy values of the plain images and the corresponding secret images. As can be seen, the values of information entropy are very close to 8, demonstrating the excellent randomness of the proposed scheme.

Table 5. NPCRs and UACIs of Lena(512×512) for different algorithms.

Index	Ideal value	Proposed	Ref. [28]	Ref. [38]	Ref. [45]
NPCR(%)	99.6094	99.6037	99.6066	99.5699	99.6154
UACI(%)	33.4035	33.3968	33.4690	33.4558	33.3526

Table 6. Information entropy of plain images and secret images.

		Secret image			
Images	Plain image	Proposed	Ref. [26]	Ref. [31]	Ref. [46]
Lena	7.4455	7.9978	7.9970	7.9968	7.9931
Peppers	7.5961	7.9973	7.9970	7.9963	7.9941
Baboon	7.5770	7.9973	7.9972	7.9965	7.9949
Boat	7.5360	7.9972	7.9973	7.9964	7.9944

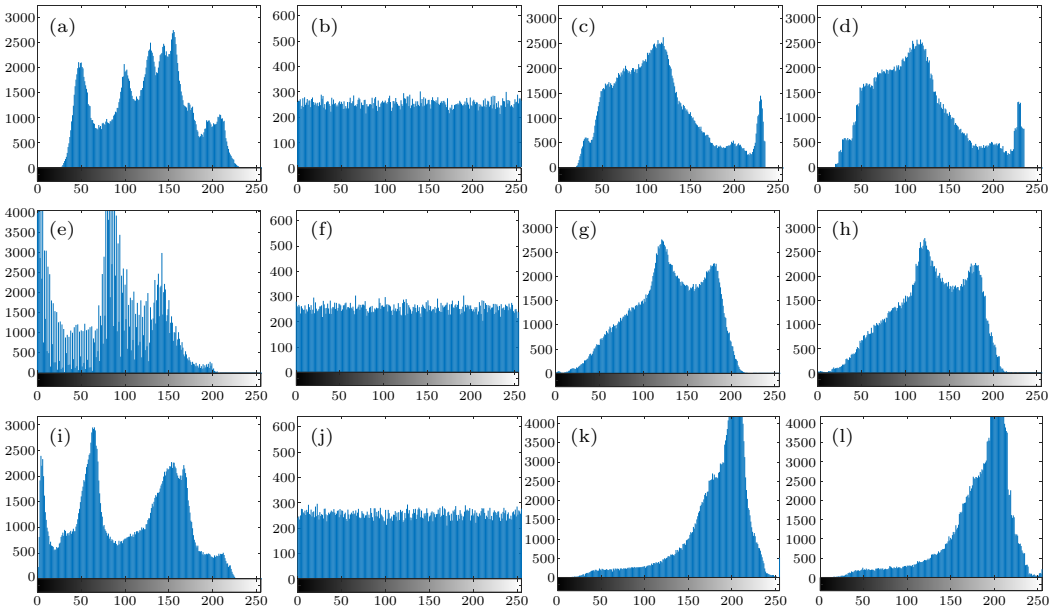


Fig. 11. Histogram analysis: [(a), (e), (i)] histograms of the plain images of Lena, Girl, and Peppers; [(b), (f), (j)] histograms of the corresponding secret images; [(c), (g), (k)] histograms of the carrier images of Goldhill, Baboon, and Aerial; [(d), (h), (l)] histograms of the corresponding cipher images.

5.6. Correlation coefficients

The correlation between adjacent pixels in the original image is strong, making it susceptible to be exploited by attackers. In this experiment, the correlation coefficients are used to quantify the level of correlation between adjacent pixels in an image. It is advisable to minimize the correlation coefficients in the secret image and make them close to 1 in the steganographic image. We choose Lena and Baboon as plain image and carrier image, and 3000 pairs of adjacent pixels are randomly selected to calculate the correlation coefficients in horizontal, vertical, and diagonal directions. The equation of

the correlation coefficient is defined as follows:

$$CC_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}}, \tag{33}$$

where x and y are adjacent pixels. Figure 12 and Table 7 show the simulation and numerical results. In addition, the comparison with other algorithms is shown in Table 8. One can observe that there is a strong correlation between adjacent pixels in the plain image, carrier image and cipher image. Conversely, the secret image exhibits weak correlation. It is worth noting that the correlation coefficients of the carrier image and the cipher image are very close. Therefore, it is difficult for attackers to extract useful information through correlation analysis.

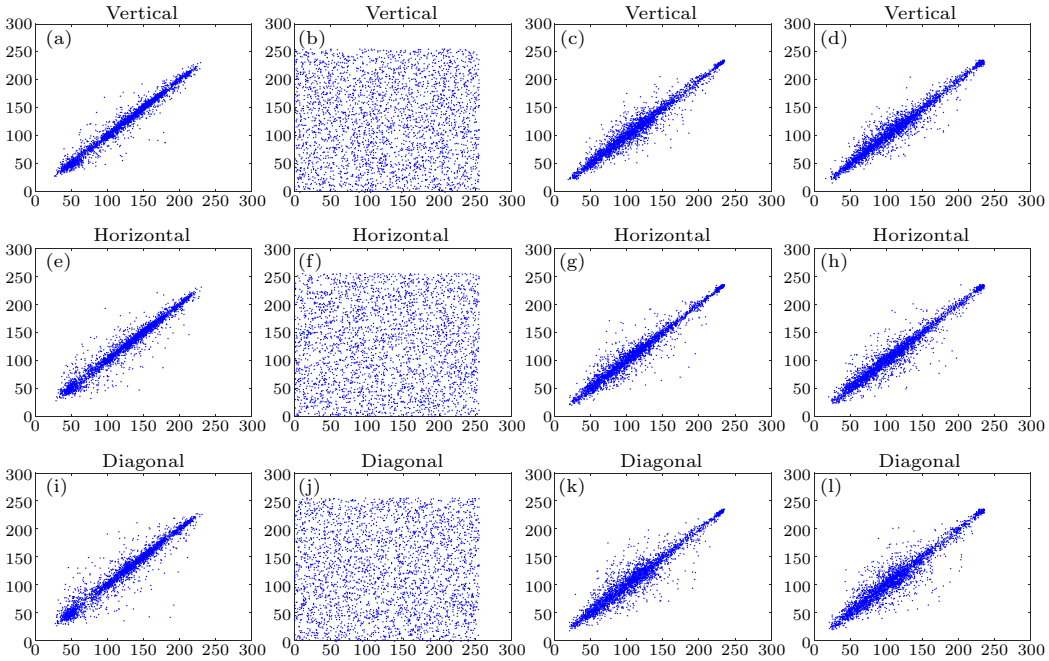


Fig. 12. Correlation coefficients analysis: [(a), (e), (i)] correlation coefficients of plain image Peppers in horizontal, vertical, and diagonal directions; [(b), (f), (j)] correlation coefficients of the secret image in horizontal, vertical and diagonal directions; [(c), (g), (k)] correlation coefficients of carrier image Baboon in horizontal, vertical and diagonal directions; [(d), (h), (l)] correlation coefficients of cipher image in horizontal, vertical and diagonal directions.

Table 7. Correlation coefficients of adjacent pixels in three directions.

Images	Horizontal	Vertical	Diagonal
Plain image	0.9719	0.9850	0.9593
Secret image	−0.0017	0.0020	0.0043
Carrier image	0.8665	0.7586	0.7261
Cipher image	0.8616	0.7543	0.7218

Table 8. Correlation coefficients of encrypted Lena images with different algorithms.

Direction	Plain image	Proposed	Ref. [26]	Ref. [27]	Ref. [38]
Horizontal	0.9719	−0.0017	0.0047	−0.0032	−0.0022
Vertical	0.9850	0.0020	0.0022	0.0123	0.0023
Diagonal	0.9593	0.0043	0.0014	−0.0071	0.0034

Table 9. PSNR and MSSIM values with different noise attack.

SPN			SN		
Intensity(%)	$PSNR_{dec}$	MSSIM	Intensity(%)	$PSNR_{dec}$	MSSIM
0.001	34.3021	0.9863	0.00012	34.2896	0.9932
0.005	29.7415	0.9907	0.00014	31.6053	0.9932
0.01	26.3647	0.9839	0.00016	22.5902	0.9921
0.03	22.0968	0.9788	0.00018	16.1327	0.9932

5.7. Noise attack

In the process of image transmission, the encrypted images are vulnerable to various noises that can degrade the quality of the decrypted images. In the test, we analyze the ability of the scheme to resist noise attacks. The images Lena and Bridge are selected as the plain image and the carrier image. Subsequently, the salt and pepper noise (SPN) with intensities of 0.001%, 0.005%, 0.01%, 0.03%, and the

speckle noise (SN) with intensities of 0.00012%, 0.00014%, 0.00016%, 0.00018% are added to the cipher image, separately. The results are shown in Fig. 13. Furthermore, the PSNR and MSSIM values of the decrypted image and the plain image are listed in Table 9. From the results, we can identify the main features of the decrypted image after being attacked by the noise. Therefore, the proposed algorithm has the ability to resist the noise attack.



(a) 0.001% SPN (b) 0.005% SPN (c) 0.01% SPN (d) 0.03% SPN

(e) 0.00012% SN (f) 0.00014% SN (g) 0.00016% SN (h) 0.00018% SN

Fig. 13. Noise attack analysis: (a)–(d) encrypted images with 0.001%, 0.005%, 0.01% and 0.03% SPN, respectively; (e)–(h) encrypted images with 0.00012%, 0.00014%, 0.00016% and 0.00018% SN, respectively.

6. Comparison with other schemes

In this section, we analyze the proposed scheme in comparison with other existing schemes, including visual security, decryption quality, and running efficiency. To ensure a fair comparison, the plain images and carrier images used in this test are identical, and the result data of the compared schemes are directly cited from the respective references.

6.1. Visual security

It is crucial to minimize the difference between the carrier image and the cipher image, which indicates that the concealment of secret information and the visual security of the algorithm are superior. The algorithm embeds secret information in the lower bits of the carrier image pixels. Since edge regions usually consist of longer pixel sequences, and such pixel sequences can hold more bits of information. Therefore, we embed more bits in the edge pixels relative to the non-edge pixels, ensuring that some non-edge pixels do not need to be embedded with any information, which can significantly improve the visual quality.

To verify the visual security of the image encryption scheme proposed in this paper, we conduct a comparison with Refs. [24,41], and evaluate the results based on PSNR and MISSIM values shown in Table 10. The proposed scheme demonstrates superior performance in these metrics. Specifically, our PSNR values are all greater than 40 dB, which is superior to other algorithms. Although the MSSIM values are slightly different from the references, the average value is around 0.99. These results indicate that the proposed scheme has excellent visual security and high-quality visual effects.

6.2. Decryption quality

In the proposed image encryption algorithm, both the threshold and quantization inherently impact the decryption quality. However, it is crucial that our scheme consistently maintains superior decryption quality. This algorithm can adequately extract secret information from the steganographic image. Moreover, the use of PCS effectively utilizes the sparsity of the signal, ensuring high reconstruction quality. In this experiment, we measured the PSNR value of the decrypted image and compared it with Refs. [24,41], as shown in Table 11. Obviously, the average PSNR value computed by our proposed scheme is greater than 35 dB, which means that our proposed scheme has outstanding decryption quality.

Table 10. Comparison of visual security.

Plain image	Carrier image	$PSNR_{cip}$			MSSIM		
		Ref. [24]	Ref. [41]	Proposed	Ref. [24]	Ref. [41]	Proposed
Lena	Baboon	35.4068	35.1636	41.0311	0.9973	0.9941	0.9915
Peppers	Woman	38.0258	36.5561	46.3969	0.9933	0.9870	0.9892
House	Sailboat	36.9968	36.8192	43.5021	0.9965	0.9904	0.9875
Boat	Barbara	35.0977	35.9176	42.7790	0.9945	0.9872	0.9876
Average		36.3818	36.1141	43.4273	0.9954	0.9897	0.9890

Table 11. Comparison of decryption quality.

Plain image	Carrier image	$PSNR_{dec}$		
		Ref. [24]	Ref. [41]	Proposed
Lena	Baboon	32.9538	33.0984	35.9139
Peppers	Woman	33.6672	32.6477	35.6233
House	Sailboat	31.9479	32.9378	36.4796
Boat	Barbara	31.7007	32.2659	33.1358
Average		32.5674	32.7375	35.2882

Table 12. Comparison of running time in units of second.

Image	Algorithm	Encryption			Decryption		
		Compression	Embedding	Total	Extraction	Reconstruction	Total
Lena (256×256)	Ref. [41]	0.10	0.24	0.35	0.28	1.25	1.53
	Proposed	0.52	0.10	0.62	1.07	0.10	1.17
Lena (512×512)	Ref. [41]	0.53	2.00	2.53	1.70	6.68	8.38
	Proposed	2.10	0.31	2.41	0.34	3.06	4.40
Peppers(512×512)	Ref. [41]	0.54	2.03	2.57	1.72	5.81	7.53
	Proposed	2.14	0.33	2.47	0.34	3.07	4.41

6.3. Running efficiency

For real-time applications, the running time of the encryption algorithm is an important performance index. In this section, we evaluate the running time required for the encryption, embedding, extraction, and reconstruction phases of the algorithm for images of different sizes, including 256×256 and 512×512. Table 12 presents the results in comparison with Ref. [41]. It can be seen that in the encryption phase, the compression process takes up the majority of the time, about 87% of the total encryption time. Moreover, the running time increases with the increase of image size. In the decryption process, the extraction phase consumes less time, while the reconstruction phase needs more time. Overall, our proposed scheme requires less encryption and decryption time than the comparative literature, which runs more efficiently.

7. Conclusion

This paper introduces a novel image encryption algorithm that combines PCS and edge detection technology. The proposed algorithm utilizes PCS to optimize compression performance and reconstruction quality. It effectively scrambles the compressed image using a specific permutation method to enhance security and randomness. The edge detection embedding capitalizes on the property that image edges can accommodate more embedding bits to design a lossless embedding method to improve the visual quality of encrypted images. Moreover, the method has ability to different carrier images, which further improves security. The encryption process is closely related to the plain image and is highly sensitive to the plaintext. The simulation results and performance comparisons show that the encrypted and decrypted images perform well regarding PSNR values, indicating that the proposed image encryption scheme has excellent visual security and decryption quality. Future research will focus on improving the encryption algorithm to enhance its robustness.

Acknowledgments

This work was supported by the Key Area R&D Program of Guangdong Province (Grant No. 2022B0701180001), the National Natural Science Foundation of China (Grant No. 61801127), the Science Technology Planning Project of Guangdong Province, China (Grant Nos. 2019B010140002 and 2020B111110002), and the Guangdong-Hong Kong-Macao Joint Innovation Field Project (Grant No. 2021A0505080006).

References

[1] Gui X Q, Huang J, Li L, Li S L and Cao J 2022 *Multimedia Tools Appl.* **81** 21975
[2] Wang X Y and Liu P B 2022 *IEEE Trans. Circuits Syst. I Regul. Pap.* **69** 1291
[3] Wang X Y, Gu S X and Zhang Y Q 2015 *Opt. Lasers Eng.* **68** 126
[4] Alawida M, Samsudin A, Teh J S and Alkhawaldeh R S 2019 *Signal Process.* **160** 45
[5] Wu J H, Liao X F and Yang B 2017 *Signal Process.* **141** 109
[6] Chai X L, Chen Y R and Broyde L 2017 *Opt. Lasers Eng.* **88** 197
[7] Hu T, Liu Y, Gong L H, Guo S B and Yuan HM 2017 *Signal Process* **134** 234
[8] Chai X L, Fu X L, Gan Z H, Lu Y and Chen Y R 2019 *Signal Process.* **155** 44
[9] Yang F F, Mou J, Cao Y H and Chu R 2020 *China Commun.* **17** 21
[10] Ma Z L, Li J Q, Di X Q, Sheng Y H and Liu Z F 2021 *Chaos Solitons Fractals* **152** 111318
[11] Wang X G, Zhou G Q, Dai C Q and Chen J L 2017 *IEEE Photon. J.* **9** 1
[12] Hu G Q, Xiao D, Wang Y, Xiang T and Zhou Q 2017 *Opt. Lasers Eng.* **98** 123
[13] Zhou N R, Yan X Y, Liang H R, Tao X Y and Li G Y 2018 *Quantum Inf. Process.* **17** 338
[14] Wu W Q and Wang Q 2022 *Int. J. Theor. Phys.* **61** 64
[15] Wang X Y and Gao S 2020 *Multimedia Tools Appl.* **80** 10301
[16] Chai X L, Zhi X C, Gan Z H, Zhang Y S, Chen Y R and Fu J Y 2021 *Signal Process.* **183** 108041
[17] Zhang L M, Sun K H, Liu W H and He S B 2017 *Chin. Phys. B* **26** 100504
[18] Chen J X, Zhu Z L, Zhang L B, Zhang Y S and Yang B Q 2018 *Signal Process.* **142** 340
[19] Hua Z Y, Yi S and Zhou Y C 2018 *Signal Process.* **144** 134
[20] Li C L, Zhang Y, Li H D and Zhou Y 2023 *Vis. Comput.* (in press)

[21] Zhou Y, Li C L, Li W, Li H M, Feng W and Qian K 2021 *Nonlinear Dyn.* **103** 2043
[22] Li C L, Zhou Y, Li H M, Feng W and Du J R 2021 *Multimedia Tools Appl.* **80** 18479
[23] Huang W, Jiang D H, An Y S, Liu L D and Wang X Y 2021 *IEEE Access* **9** 41704
[24] Jiang D H, Liu L D, Zhu L Y, Wang X Y, Rong X W and Chai H X 2021 *Signal Process.* **188** 108220
[25] Xiao D, Cai H K and Zheng H Y 2015 *Chin. Phys. B* **24** 060505
[26] Huang S F, Huang L Q, Cai S T, Xiong X M and Liu Y 2022 *IET Image Process.* **16** 1544
[27] Yang C, Pan P and Ding Q 2022 *Entropy* **24** 273
[28] Xu Q Y, Sun K H, He S B and Zhu C X 2020 *Opt. Lasers Eng.* **134** 106178
[29] Huo D M, Zhu Z L, Wei L S, Han C and Zhou X 2021 *Opt. Commun.* **492** 126976
[30] Zhou N R, Li H L, Wang D, Pan S M and Zhou Z H 2015 *Opt. Commun.* **343** 10
[31] Yang Y G, Guan B W, Li J, Li D, Zhou Y H and Shi W M 2019 *Opt. Laser Technol.* **119** 105661
[32] Bao L and Zhou Y C 2015 *Inf. Sci.* **324** 197
[33] Chai X L, Gan Z H, Chen Y R and Zhang Y S 2017 *Signal Process.* **134** 35
[34] Wang H, Xiao D, Li M, Xiang Y P and Li X Y 2019 *Signal Process.* **155** 218
[35] Thanki R, Borra S, Dwivedi V and Borisagar K 2017 *Imaging Sci. J.* **65** 457
[36] Wen H P, Ma L C, Liu L H, Huang Y M, Chen Z F, Li R, Liu Z, Lin W X, Wu J H, Li Y Q and Zhang C F 2022 *Sci. Rep.* **12** 16523
[37] Musanna F, Dangwal D and Kumar S 2020 *Multimedia Tools Appl.* **79** 25115
[38] Zhu L Y, Song H S, Zhang X, Yan M D, Zhang T, Wang X Y and Xu J 2020 *Signal Process.* **175** 107629
[39] Luo W Q, Huang F J and Huang J W 2010 *IEEE Trans. Inf. Forensics Secur.* **5** 201
[40] Rajendran S and Doraipandian M 2017 *Int. J. Netw. Secur.* **19** 593
[41] Chai X L, Wu H Y, Gan Z H, Zhang Y S, Chen Y R and Nixon K W 2020 *Opt. Lasers Eng.* **124** 105837
[42] Wu D C and Tsai W H 2003 *Pattern Recognit. Lett.* **24** 1613
[43] Li H J, Yu C Z and Wang X Y 2020 *Multimedia Tools Appl.* **80** 8721
[44] Jiang D H, Liu L D, Wang X Y and Rong X W 2021 *IET Image Process.* **15** 3698
[45] Xie Y Q, Yu J Y, Guo S Y, Ding Q and Wang E F 2019 *Entropy* **21** 819
[46] Ponuma R and Amutha R 2017 *Multimedia Tools Appl.* **77** 19209