A Lightweight Image Encryption Algorithm Based on a Dual Chaotic System and Dynamic S-box

Rui-Hong Chen, Qiu-Yu Zhang, Ling-Tao Meng, and Yi-Lin Liu
(Corresponding author: Qiu-yu Zhang)

School of Computer and communication, Lanzhou University of Technology
No. 287, Lan-Gong-Ping Road, Lanzhou 730050, China
Email: c2745168470@163.com,zhangqylz@163.com

(Received Apr. 6, 2023; Revised and Accepted Nov. 22, 2023; First Online Feb. 23, 2024)

Abstract

To solve the problems of long encryption time, low encryption throughput, large memory usage, high energy consumption, and weak encryption performance of existing image encryption algorithms in the resource-constrained multimedia Internet of Things (MIoT) for secure communication, a lightweight image encryption algorithm based on dual chaotic system and dynamic S-box was proposed. Firstly, the digital image is compressed after blocking by discrete cosine transform (DCT) coding technology. Then, the encryption key is generated from the initial key and the grey value of the compressed image. The generated encryption key is used as the initial value and control parameter for two-dimensional (2D) logistic mapping and tent mapping, and the key sequence is generated after each iteration. The generated key sequence is used to permute pixel bits and generate the S-box. Finally, the S-box dynamically generated by the tent mapping is used to diffuse the permuted pixel bits and obtain the encrypted image. The experimental results show that compared with the existing methods, the encryption time of the proposed algorithm is reduced by 3.0230s on average, the throughput of the system is increased by 0.0673Mbps on average, and the memory usage is low, with high encryption efficiency and security strength.

Keywords: Dynamic S-box; Image Compression; Image Encryption; Lightweight Encryption; Logistic-Tent Dual Chaotic System

1 Introduction

With the rapid development of social media networks and the MIoT, the demand for multimedia data sharing has increased significantly, but MIoT devices have resource constraints in terms of energy consumption, computing power, running time, and storage space [21]. Therefore, to enable secure and confidential communication with minimal power consumption in resource-constrained devices, lightweight cryptographic algorithms can meet both security and performance requirements, compared with traditional general cryptographic algorithms, lightweight cryptographic algorithms generally have the characteristics of low throughput, moderate security level, and high performance, and can be applied in MIoT to solve the problems of low system throughput and long response time in the encryption process [11].

In recent years, in resource-constrained IoT application scenarios, have used the SHA (Secure Hash Algorithm) series of hash algorithms, the SM series of national secret algorithms, AES (Advanced Encryption Standard) and other symmetric cryptographic algorithms, RSA (Rivest Shamir Adleman) and ECC (Ellipse Curve Cryptography) and other public key cryptographic algorithms [17,25] traditional cryptographic algorithms represented by traditional cryptography generally have performance limitations, such as relatively low computational efficiency, and it is difficult to obtain better encryption effects. Existing image encryption technologies mainly include chaotic encryption [7,8], DNA coding [23], quantum encryption [27], optical encryption [24], cellular automaton encryption [18] and neural network [26] as well as other encryption methods. These encryption technologies have good encryption performance, but due to the strong correlation between pixels and high data redundancy, the implementation is more complex or difficult to ensure the security of encryption, and the memory consumption is large, the encryption efficiency is not high, and it does not meet the requirements of lightweight cryptographic algorithms.

With the rapid development of chaos theory, chaos encryption technology has gradually become the main direction of image encryption. A chaotic cryptosystem can have excellent characteristics in terms of computing power, security, complexity and other aspects, and the combination of a chaotic system and image compression can not only reduce the amount of data and storage space of encrypted images but also reduce the consumption of space resources. To increase the security strength and encryption efficiency of the whole cryptographic algorithm, the single nonlinear structure S-box in the block cypher algorithm can be used to confuse and spread the image pixel value, and the quality of the indicators of the S-box directly determines the quality of the cryptographic algorithm [28]. Currently, due to the increasing amount of multimedia content, several lightweight cryptographic algorithms have emerged, that enable MIoT nodes to communicate securely with minimal computational complexity and bandwidth. Considering the transmission of images in resource-constrained MIoT, many scholars combine chaotic schemes with related technologies to optimize the performance of image encryption algorithms in constrained environments and achieve better encryption performance. For example, decreasing the encryption speed due to increasing the dimension of chaos and the computation step affects the effect of encryption, which is not suitable for resource-constrained IoT devices.

To solve the above problems, to make the lightweight cryptographic algorithm better take into account the requirements of the throughput, security and performance, this paper presents a lightweight image encryption algorithm based on a dual chaotic system and dynamic S-box, which uses the compression properties of DCT coding to compress the original image, and uses the Logistic-Tent dual chaotic system and dynamic S-box to encrypt the compressed image. The main contributions of this work are as follows:

1) A lightweight image encryption algorithm based on Logistic-Tent dual chaotic system and dynamic S-box is proposed, which takes into account balance between throughput, security level and performance.

2) Using the DCT coding compression feature, according to the statistical properties of the image signal in the frequency domain, the correlation between adjacent pixels is eliminated, the amount of data in the encrypted image is reduced, and the encryption efficiency is improved.

3) The designed Logistic-Tent dual chaotic system and dynamic S-box have stronger traversability and higher key space, and the initial parameters of the chaotic system and the average pixel value of the compressed image are iteratively generated to generate the encryption key, and the random permutation and diffusion of pixel bits are performed, which increases the correlation between the key and the image and improves the security of the image.

The rest of the text is arranged as follows: Section 2 reviews the relevant research. Section 3 describes the lightweight image encryption algorithm, dynamic S-box construction, and the encryption and decryption processes. Section 4 gives the experimental results and analysis of the proposed algorithm, and compares it with the experimental data of the existing related algorithms. Section 5 summarizes the work in this paper.

2 Related Works

In recent years, the hybrid digital image encryption method combining chaotic systems and other methods has been favoured and concerned by many scholars, and the existing results can have the characteristics of better key space and more system parameters, and achieve complementary advantages. Yousaf *et al.* [28] proposed an image encryption method that evolved a highly nonlinear S-box through the action of puzzle subgroups on the set of elements in magic, taking into account both complexity and ease of use. Kumar *et al.* [19] proposed a chaotic image encryption algorithm based on enhanced Thorp scrambling and Zigzag scanning convolution, which has high security. Zheng *et al.* [30] proposed an algorithm for constructing dynamic S-boxes based on chaos mapping and apply the idea of obfuscation to the construction of S-boxes, which has a good key space and can resist common attacks. Arif *et al.* [3] proposed a method based on substitution and substitution, combined with single S-box encryption, which has a high sensitivity to plaintext attacks. Idrees *et al.* [16] proposed an image encryption algorithm using S-box and dynamic Henon position exchange, which improved the security of encrypted images, but occupied more storage resources. Farah *et al.* [5] proposed a cryptographic algorithm based on the obfuscation/diffusion Shannon feature, using the Jaya algorithm to generate S-boxes. Farah *et al.* [4] proposed an algorithm for constructing S-boxes using chaos mapping and genetic algorithms, and the constructed S-boxes have better randomness and anti-attack ability. Ibrahim *et al.* [15] proposed a dynamic S-box construction method based on the key-dependent permutation of elliptic curve points, but the computational overhead of dynamic S-box construction limits the achievable encryption throughput. Hayat *et al.* [13] proposed a method to generate block cyphers using elliptic curves, and the generated S-box has high randomness and security. Zahid *et al.* [29] proposed a method to construct dynamic S-boxes through linear trigonometric transformations, which effectively improved the randomness of trigonometric transformations to generate S-boxes.

At present, most of the communication between MIoT devices is done in the form of images, and MIoT devices are often attacked by illegal elements because of small storage space and large losses. To solve these problems, scholars have proposed a variety of lightweight image encryption algorithms. For example, Alghamdi *et al.* [1] proposed a lightweight image encryption algorithm based on Logistic mapping, permutation and AES, which reduces the time required for encryption while ensuring certain security. Ferdush *et al.* [6] proposed an image encryption scheme based on Arnold and Logistic, which effectively reduced the time required for encryption and improved encryption efficiency. Almalkawi *et al.* [2] proposed a lightweight compressed image encryption scheme with joint chaotic mapping, which can resist most attack types, but the disadvantage of this scheme is that it uses a one-for-time cypher scheme when generating keys. Liu *et al.* [20] proposed a lightweight image encryption algorithm based on a messaging algorithm with chaotic external messages, and the messaging algorithm adopted allows simple messages to be delivered locally to solve global problems, which will make the interaction between neighbouring pixels without additional space cost, thereby reducing resource consumption. Gupta *et al.* [10] proposed a new lightweight image key-based image encryption algorithm using Chebyshev chaotic mapping and cross-blending, which uses a mixture of crossover and Chebyshev mapping to create session keys, which is highly secure but slow in encryption. On this basis, Gupta *et al.* [9] proposed a lightweight image encryption algorithm based on a logistic-tent map and genetic algorithm, which effectively reduced the complexity and improved the encryption efficiency. Hasan *et al.* [12] proposed a scheme for encrypting medical images using two permutation techniques, which are highly secure but also highly complex. Hedayati *et al.* [14] used a scan-based block compression algorithm and selective encryption algorithm to encrypt images, which effectively reduced the time complexity and reduced the amount of encrypted data. Mondal *et al.* [22] proposed a lightweight image encryption scheme based on chaotic mapping and diffusion circuit, which controls the random number sequence used for pixel bit permutation and diffusion through the chaotic sequence, which reduces complexity and computational overhead.

In summary, the existing lightweight image encryption algorithm adopts a low-dimensional chaotic system with a simple structure and easy software and hardware to implement, but there are problems such as short periodicity, limited computing accuracy and low security, while the high-dimensional chaotic system has more control parameters and can better resist various attacks, but it is not suitable for MIoT devices with limited resources due to high complexity. Therefore, this paper uses DCT coding compression, dynamic S-box and Logistic-Tent dual chaos system to present a lightweight image encryption algorithm suitable for resource-constrained application scenarios.

3 The Proposed Scheme

To solve the problems of low system throughput, low encryption efficiency, and high memory usage in the confidential transmission of MIoT devices by existing image encryption algorithms, the algorithm compresses the original images and then encrypts them, thereby reducing the amount of encrypted data and effectively shortening the time required for encryption. In terms of security, a dynamic S-box is developed based on Tent mapping, and the key sequence generated by the 2D Logistic mapping is used to permute and diffuse the image pixel bits. Figure 1 shows the processing flow of the lightweight image encryption algorithm of the proposed algorithm. The proposed algorithm includes four processing steps: DCT coding image compression, key generation, dynamic S-box construction, and image encryption.

3.1 DCT Coding Image Compression Processing

The application of DCT coding for image data compression can reduce the digital information representing the image's luminance (or colour value) and achieve the purpose of data compression. To reduce the processing time and improve the encryption efficiency, the 2D discrete cosine transform is used for image compression. The specific steps are as follows:

Step 1: Divide the original image I with size $M \times N$ into 8×8 image blocks, and use Equation (1) to convert the image block with pixel density $f(x,\ y)$ into a matrix $F(u,\ v)$, which is the same size as the image block of the original image I, and the two conversion parameters u and v point to the spatial frequency.

$$F(u,v) = \frac{2}{\sqrt{M \times N}} C_u C_v \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cos\left(\frac{\prod(2y+1)v}{N}\right) \cos\left(\frac{\prod(2x+1)u}{M}\right) \tag{1}$$

where $u = 0, 1, ...M-1, v = 0, 1, ..., N-1$. When $u = 0, C_u = \sqrt{1/2}$, when $u \neq 0, C_u = 0$, when $v = 0, C_v = \sqrt{1/2}$, when $v \neq 0, C_v = 0$.

Step 2: Obtain the DCT quantization coefficient. First, after 2D-DCT conversion, 64 coefficients are obtained:the important DC coefficient (DC component) and the low-frequency AC coefficient (AC component), located in the upper left corner, while the remaining high-frequency AC coefficient is less important for the human visual system. Then, the other coefficients are discarded by retaining the important coefficients using the quantification step.

Step 3: After quantization, a large part of the data in the matrix has become 0, and the Zigzag is used to convert the quantized two-dimensional matrix into a one-dimensional array. According to the arrangement shown in Figure 2, the DCT coefficients were first arranged from a data series, and then the compressed image $\boldsymbol{I}_c$ was obtained by using Huffman coding.

3.2 Key Generation

First, calculate the sum of all pixels of the compressed image $\boldsymbol{I}_c$ sum to obtain the pixel average K, and then use the sum of pixels *sum* and the initial key (x_0, y_0, z_0, k, u) to calculate the encryption key $(x_0^{'}, y_0^{'}, z_0^{'}, k^{'}, u^{'}, K)$ that is,

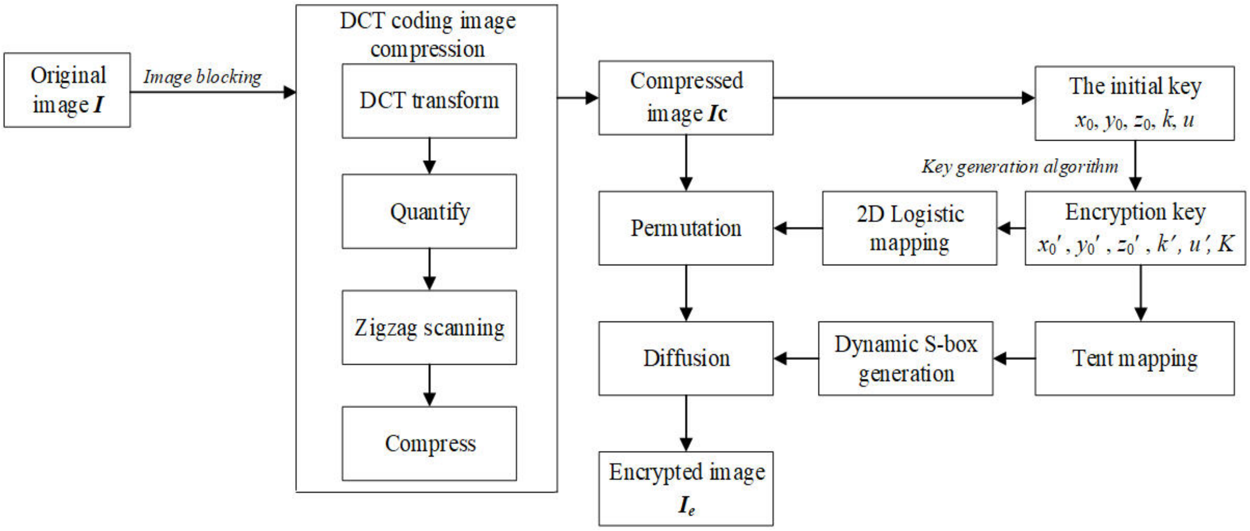
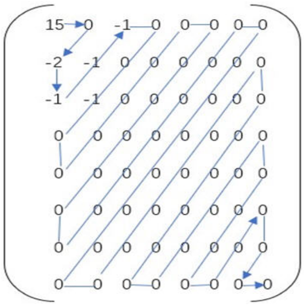


Figure 1: The lightweight image encryption algorithm processing flow.



$$\begin{pmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Figure 2: Arrangement of DCT coefficients.

use the 2D Logistic mapping of Equation (2) to obtain the initial value (x_0, y_0) and control parameter k, and use the tent mapping of Equation (3) to obtain the initial value z_0 and control parameter u.

$$\begin{cases} x(n+1) = y(n) \\ y(n+1) = k * y(n)(1 - x(n)) \end{cases} \tag{2}$$

where the control parameters $k \in (0, 2.28)$, the initial state x, $y \in (0, 1)$.

$$z_{(n+1)} = \begin{cases} (z_n/u) & 0 < z_n < 0.5 \\ (1 - z_n)/(1 - u) & 0.5 \le 0 \le 1 \end{cases} \tag{3}$$

where $z_n \in (0,1)$, and u are the control parameters that control the dynamic properties of the tent mapping. When controlling parameter u>1, the map has a bifurcation phenomenon, that is, the tent mapping has a chaotic phenomenon. When $u = 2$, the chaotic sequence produced by the tent mapping approximately obeys a uniform distribution.

2D Logistic mapping has better randomness and key space and controls the dynamics of chaos mapping. The specific processing process of encryption key generation is shown in Algorithm 1.

```
Algorithm 1 : Key generation algorithm
Input: Compressed image $\boldsymbol{I}_c$, The initial key $x_0$, $y_0$, $z_0$, $k$, $u$
Output: Encryption key $x_0^{'}$, $y_0^{'}$, $z_0^{'}$, $k^{'}$, $u^{'}$, $K$
1: The compressed image $\boldsymbol{I}_c$ is traversed in the order from left to right and top to bottom to obtain a sequence $\boldsymbol{I}_c{'}$ with size $M \times N$
2: Calculates the sum of all pixels:$sum = \sum_{x=0}^{M*N} I_c(i)^{'}$
3: Calculates the average of all pixel sums:
       $K = floor\left(\frac{sum}{M \times N}\right)$
4: $x_0^{'} = (x_0 + x_0 * sum) mod 1$; /* Calculate the encryption key $x_0^{'}$, $y_0^{'}$, $z_0^{'}$, $k^{'}$, $u^{'}$
5: $y_0^{'} = (y_0 + y_0 * sum) mod 1$;
6: $z_0^{'} = (z_0 + z_0 * sum) mod 1$;
7: $u^{'} = (u + u * sum) mod 1$;
8: $k^{'} = (k + k * sum) mod 1$;
9: return $x_0^{'}$, $y_0^{'}$, $z_0^{'}$, $k^{'}$, $u^{'}$, $K$
```

3.3 Dynamic S-box Construction Based on Tent Mapping

The S-box is nonlinear and is the basic structure for performing permutation calculations, providing better security. To design a satisfactory S-box, this paper uses Tent mapping to construct a dynamic S-box. This is because the tent mapping algorithm is simple, it can effectively reduce storage space, encryption and decryption speed, has high efficiency, and high security, suitable for MIoT devices to transfer information.

For grayscale or colour images of size $M\times N$, the processing flow of using Tent mapping to construct a dynamic S-box is shown in Figure 3.

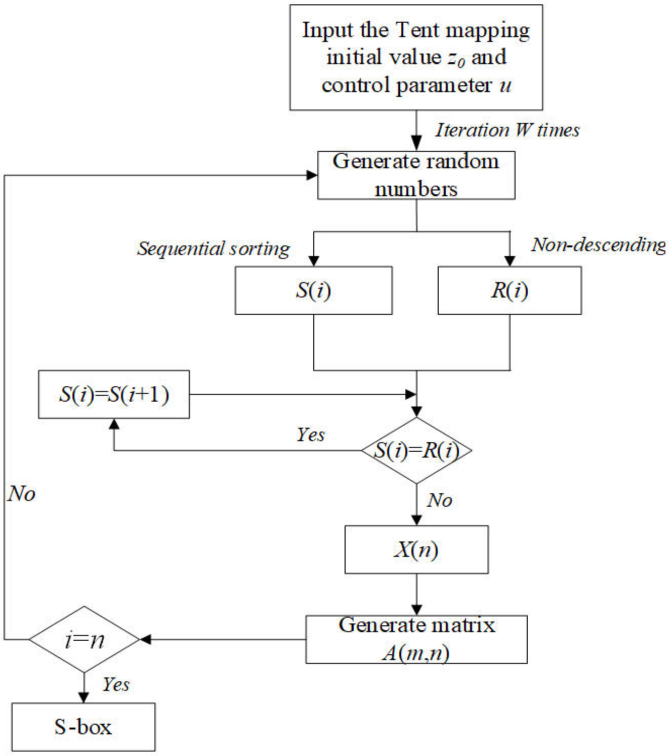


Figure 3: Dynamic S-box construction process.

The specific construction steps of the dynamic S-box are as follows:

Step 1: Take the initial value z_0 of the tent map and the control parameter u iteration W imes to obtain a sequence of random numbers $\boldsymbol{L}$ of length W, where $\boldsymbol{L}=\{L(1), L(2), L(3), ..., L(i)\}, (0 \leq i \leq n, L(i) = 256)$.

Step 2: Record the interval sequence the number of the elements in the subsequence as m, start numbering from 0, when the value of m reaches 255, start the next subsequence number, and so on.

Step 3: The sequence of interval numbers that generate random numbers is denoted as $\boldsymbol{S}$, where $\boldsymbol{S}=\{S(1), S(2), S(3), ..., S(i)\}, (0 \leq i \leq n)$.

Step 4: Sort the corresponding random values in the sequence $\boldsymbol{S}$ in non-descending order to obtain the interval number sequence $\boldsymbol{R}=\{R(1), R(2), R(3), ..., R(i)\}$, if $\boldsymbol{S}$(i)$\neq$ $\boldsymbol{R}$(i), then use the elements in the sequence $\boldsymbol{R}$(i) to form the sequence $\boldsymbol{X}$(n) into a two-dimensional matrix **A** of 16×16 in the order of first left and then right, and then up and down, if $\boldsymbol{S}$(i)=$\boldsymbol{R}$(i), then make $\boldsymbol{S}$(i)=$\boldsymbol{S}$(i+1), and judge again.

Step 5: $\mathbf{A}(x, y)$ represents the value of the x row y column in the matrix, use Equation (4) to convert the value at position $\mathbf{A}(x, y)$ in the matrix to another position $\mathbf{A}(x', y')$, and convert the values in the matrix in the order of row priority to obtain an 8×8 S-box.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \bmod (256) \qquad (4)$$

Step 6: If $i \neq n$, repeat Steps 3 to 5 to generate the S-box again, this process can be cycled multiple times, generating multiple S-boxes.

In the construction of the above S-box, only simple sorting, replacing and remaining operations are involved, which can effectively reduce the memory occupation of the algorithm and help reduce the running time.

3.4 Encryption Process

In Figure 1, the key generation algorithm of Algorithm 1 is used to generate encryption keys to control 2D Logistic mapping and Tent mapping, and the key sequence is generated for image pixel bits to perform permutation and diffusion operations and generate dynamic S-boxes. The specific encryption steps are as follows:

Step 1: Image compression of the original image $\boldsymbol{I}$ through the DCT encoding image compression process given in Section 3.1 to generate the compressed image $\boldsymbol{I}_c$.

Step 2: Bring the initial key $\boldsymbol{I}_c$, The initial key (x_0, y_0, z_0, k, u) and the compressed image Ic into the key generation algorithm of Algorithm 1 to obtain the encryption key $(x_0', y_0', z_0', k', u', K)$.

Step 3: Iterate the tent mapping according to Section 3.3 to generate an 8×8 S-box. To randomly disperse the displaced pixels using a dynamic S-box, the S-box can be represented by a matrix of size 16×16. The matrix of 16×16 is traversed from left to right and top to bottom to obtain a sequence $\boldsymbol{S}$ of length 256. Where $\boldsymbol{S}$(i) represents the i-th element of the sequence $\boldsymbol{S}$, i=0, 1, ..., 255.

Step 4: Substituting x_0, y_0, k into Equation (2) and iterating 1000+$M\times N$, in order to obtain more stable data, the data generated from 1001 iterations to 1000+ $M\times N$ times are used as valid data to generate a sequence list1 with length $M\times N$.

Step 5: Use Equation (5) to sort and calculate the sequence $\boldsymbol{L}_{seq}$ to obtain the sorted index sequence $\boldsymbol{L}_1$.

$$L_1 = \arg sort(L_{seq}) \qquad (5)$$

where argsort() is a function in Python for sorting elements in a sequence of arrays from smallest to largest, and returns the index after the elements are sorted.

Step 6: Traverse the compressed image $\boldsymbol{I}_c$ in order from top to bottom, left to right, and obtain a sequence $\boldsymbol{I}_c{}'$ of size $M\times N$, $\boldsymbol{I}_c{}'(i)$ represents the i-th element in the sequence $\boldsymbol{I}_c{}'$ (i=0, 1, ..., $M\times N$-1).

Step 7: According to the value of sequence $\boldsymbol{L}_1$, the elements in sequence $\boldsymbol{I}_c{}'$ are replaced according to the permutation processing method in Figure 4 to obtain a new sequence $\boldsymbol{I}_{c1}{}'$.

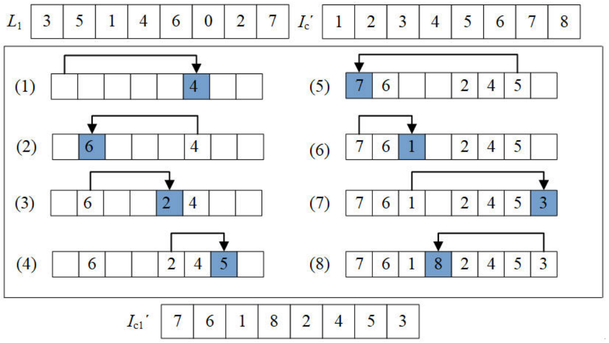


Figure 4: Permutation process.

Step 8: Determine the first-pixel value of the diffusion according to the value of the first element in the sequence $\boldsymbol{L}_1$, use the sequence $\boldsymbol{S}$ and the pixel and the average $\boldsymbol{K}$ to obtain the first grey value of the diffusion, and then store it in the first position of the empty sequence $\boldsymbol{I}_{c2}{}'$. The calculation formula is shown in Equation (6), where $\boldsymbol{L}_1(0)$ represents the first element in the sequence $\boldsymbol{L}_1$.

$$\begin{cases} index = I_{c1}{}'(L_1(0)) \oplus K \oplus L_1(i) \\ I_{c2}{}'(L_1(0)) = S(index) \end{cases} \tag{6}$$

Step 9: Equation (7) is continued to be used to diffuse other pixels, and then the diffusion of each pixel will be affected by the previous pixel, to obtain the final sequence $\boldsymbol{I}_{c2}{}'$.

$$\begin{cases} index = I_{c1}{}'(L_1(0)) \oplus I_{c2}{}'(L_1(i-1)) \oplus L_1(i) \\ I_{c2}{}'(L_1(i)) = S(index) \\ i = 1, 2, ..., M * N \end{cases} \tag{7}$$

Step 10: Convert the sequence $\boldsymbol{I}_{c2}{}'$ in row-first order using Equation (8) to the matrix form $\boldsymbol{P}$ of $M\times N$, which is the final encrypted image $\boldsymbol{I}_e$.

$$P = reshape(I_{c2}{}', M, N) \tag{8}$$

where reshape() is a function in Python that converts elements in an array to matrix form.

3.5 Decryption Process

The decryption process is the reverse of the encryption process. The specific process is as follows:

Step 1: Enter the encryption key ($x_0^{'}$, $y_0^{'}$, $z_0^{'}$, $k^{'}$, $u^{'}$, K), and obtain the sorted sequence $\boldsymbol{L}_1$ by Equation (9).

$$L^{-1} = \arg sort(list) \tag{9}$$

Step 2: Traverse all pixels of the encrypted image $\boldsymbol{I}_e$ in order from left to right, top to bottom to generate a sequence $\boldsymbol{I}_{c2}{}'$.

Step 3: Through Equation (10), the sequence $\boldsymbol{I}_{c1}{}'$ is obtained by using the reverse process of diffusion.

$$\begin{cases} index = S(I_{c2}{}'(L_1(0))) \\ I_{c1}{}'(L_1(0)) = index \oplus K \oplus L_1(0) \\ I_{c1}{}'(L_1(i)) = S(I_{c2}{}'(L_1(\mathrm{i}))) \oplus K \oplus L_1(i) \\ i{=}\ 1, 2,...M{*}N{-}1, \end{cases} \tag{10}$$

Step 4: Through Equation (11), the sequence $\boldsymbol{I}_c{}'$ is obtained by using the reverse process of permutation.

$$\begin{cases} I_c{}'(L_1(0)) = I_{c1}{}'(L_1(M * N - 1)) \\ I_c{}'(L_1(i+1)) = I_c{}'(L_1(i)) \\ i = M * N - 2, ..., 1, 0 \end{cases} \tag{11}$$

Step 5: Convert the sequence $\boldsymbol{I}_c\prime$ through Equation (12) to the matrix $\boldsymbol{I}_c$.

$$I_c = reshape(I_c{}', M, N) \tag{12}$$

Step 6: The obtained compressed image $\boldsymbol{I}_c$ is inversely encoded and quantized by the inverse discrete cosine transform (IDCT) to restore the original image $\boldsymbol{I}$.

4 Experimental Results

Experimental hardware environment: Intel(R) Core(TM) i5-7572U 1.60GHz, 16GB. The software environment is Windows 10, PyCharm (Professional Edition) 2020.3.2 x64. The performance evaluation of the proposed algorithm is mainly evaluated from four aspects: encryption image quality, lightweight algorithm, encryption security level, and chaotic characteristics of S-box. Four images of 256×256 grayscale images are used for experimental test images: Lena, Peppers, Cameraman, and Baboon.

4.1 Encrypted Image Quality Analysis

The peak signal-to-noise ratio (PSNR) [1] is one of the objective evaluation indicators to measure image quality. The lower the PSNR value of the encrypted image and the worse the image quality of the encrypted image, the better the encryption effect. The PSNR is generally expressed through the mean squared error (MSE). For images of size $M \times N$, MSE is calculated as in Equation (13):

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (I(i,j) - I'(i,j))^2 \tag{13}$$

where M and N represent the width and height of the image, I is the original image, and I is the reconstructed image.

PSNR is based on the MSE definition, which is calculated as shown in Equation (14):

$$PSNR = 10\log_{10}(\frac{255^2}{MSE}) \tag{14}$$

The ratio of the data stream length of the image before compression to the data stream length of the compressed image is called the compression ratio. Table 1 shows the PSNR values of Lena, Peppers, Cameraman, and Baboon encryption/decryption when the compression ratio is 16:1. Table 2 compares the PSNR values of the proposed algorithm with the existing methods [3, 6, 20] of Lena encrypted image.

As can be seen from Table 1, the PSNR value of the encrypted image is low, and it is difficult for humans to observe the useful information in the image with the naked eye from the encrypted image, indicating that the proposed algorithm has good encryption performance. The PSNR values of the decrypted images are greater than 38dB, indicating that the proposed algorithm has a high-quality of decrypted image.

As can be seen from Table 2, the PSNR value of the proposed algorithm is lower than that of the comparative Ref. [3, 6, 20] which indicates that the encryption effect of the proposed algorithm is better than that of the comparative Ref. [3, 6, 20]. Therefore, the proposed algorithm has good encryption performance.

4.2 Encrypted Efficiency Analysis

4.2.1 Encryption Time, Encryption Throughput Analysis

Encryption speed is very important for the practicality of image encryption algorithms. At present, encryption time is an important indicator to measure the execution efficiency of MIoT devices when processing massive amounts of data. The longer it takes to encrypt an image, the less efficient the encryption becomes. Conversely, the more efficient the encryption. Table 3 compares the encryption time of Lena's image in the proposed algorithm with existing methods [5, 6, 16, 19, 22].

In addition, encryption throughput (ET) refers to the ability of a system to process data per second, which is an important indicator of system performance. Higher throughput, shorter algorithm execution times and faster encryption. Conversely, the slower the encryption. The calculation formula is shown as Equation (15) (unit: Mbps):

$$ET = \frac{C}{t} \tag{15}$$

where ET represents the encryption throughput, C is the size of the encrypted image, and t is the encryption time.

As can be seen from Table 3, the proposed algorithm takes 0.8625s to encrypt a 256×256 Lena image, and the encryption throughput can reach 0.2174Mbps. Compared with the existing methods [5, 6, 16, 19, 22], the average encryption time is reduced by an average of 3.0320s, and the encryption throughput is increased by an average of 0.0673Mbps. Therefore, the proposed algorithm takes less time, the encryption speed is faster, and it has good encryption efficiency, which can be used for real-time compression and encryption of images in MIoT.

4.2.2 Memory Consumption Analysis

In general, an efficient image encryption algorithm is to minimize communication overhead and occupies less storage space. Therefore, the code size of the individual algorithms involved in the encryption/decryption process store is an important metric for measuring the performance of the algorithm. The larger the code size of the algorithm, the greater the communication overhead, the more resources consumed, the lower the encryption efficiency, and vice versa, the higher the encryption efficiency. Table 4 shows the code size analysis of the proposed algorithm and the existing method [10, 17].

As can be seen from Table 4, the code size of the algorithm generating the key and constructing the S-box in this paper is small (21094Byte), which is lower than [17], indicating that the memory occupation is lower than the [17]. Since the Chebyshev mapping and crossing method is used in [10] to generate session keys, the key size is small, so its code size is small, and the memory footprint is lower than that of the proposed algorithm. Therefore, the proposed algorithm occupies less storage space, effectively reduces the burden of channel transmission, and improves operational efficiency.

4.2.3 Time Complexity Analysis

Time complexity is an important indicator to measure the quality and operation efficiency of algorithms [14]. Assuming that the size of the image is $N \times N$, the time complexity of the encryption algorithm is divided into the following parts: the time complexity of 2D DCT compression encoding is $O((N^2)logN)$, the time complexity of key generation is $O(N^2/CR)$, the time complexity of permutation and diffusion is $O(N/CR)$, and the time complexity of building S-box is $O(N^2)$, then the time complexity of the proposed algorithm is $O((logN + 3/CR + 1)N^2)$. Table 5 compares the time complexity of the proposed algorithm with the existing methods [2, 5, 12, 20, 29].

As can be seen from Table 5, the time complexity of the proposed algorithm is lower than that of the comparative Ref. [2, 5, 12, 20, 29]. Therefore, the proposed algorithm has lower time complexity.

4.2.4 Energy Analysis

The energy [28] is a measure of changes in the colour or brightness of pixels in an image. The total energy value of the constant image is 1. Compared with the original image, the encrypted image tends to have lower energy,

Table 1: PSNR values for encryption/decryption of 4 test images when the compression ratio is 16:1

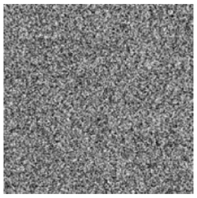
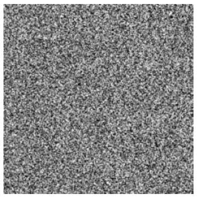
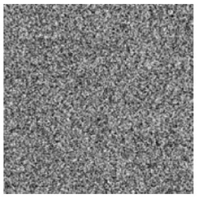
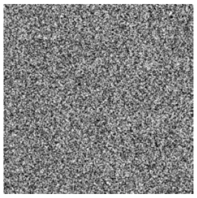
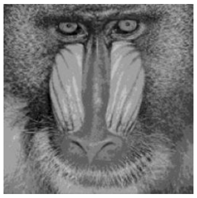
Original image	Encrypted image	PSNR /dB	Decrypted image	PSNR /dB
 Lena		8.4860		41.2280
 Peppers		8.6880		39.6430
 Cameraman		8.2064		38.2500
 Baboon		9.7029		40.2060

Table 2: Comparison of PSNR values of Lena's encrypted image

Method	Proposed	Ref. [3]	Ref. [6]	Ref. [20]
PSNR(dB)	8.4860	9.2376	11.8325	8.5510

and the lower the energy value, the better the security effect of the encryption algorithm. Table 6 compares the energy values of encrypted images between the proposed algorithm and the existing methods [15, 16, 28, 29]. The formula for calculating energy E is as follows:

$$E = \sum_{i,j} p(i,j)^2 \tag{16}$$

where $p(i,j)$ represents the total number of grayscale symbiotic matrices at (i,j).

As can be seen from Table 6, the energy value of the encrypted image of the proposed algorithm is lower than that of Ref. [16, 28, 29] and close to Ref. [15]. Therefore, compared with other encryption algorithms, the proposed algorithm has higher security, better attack resistance and good encryption performance.

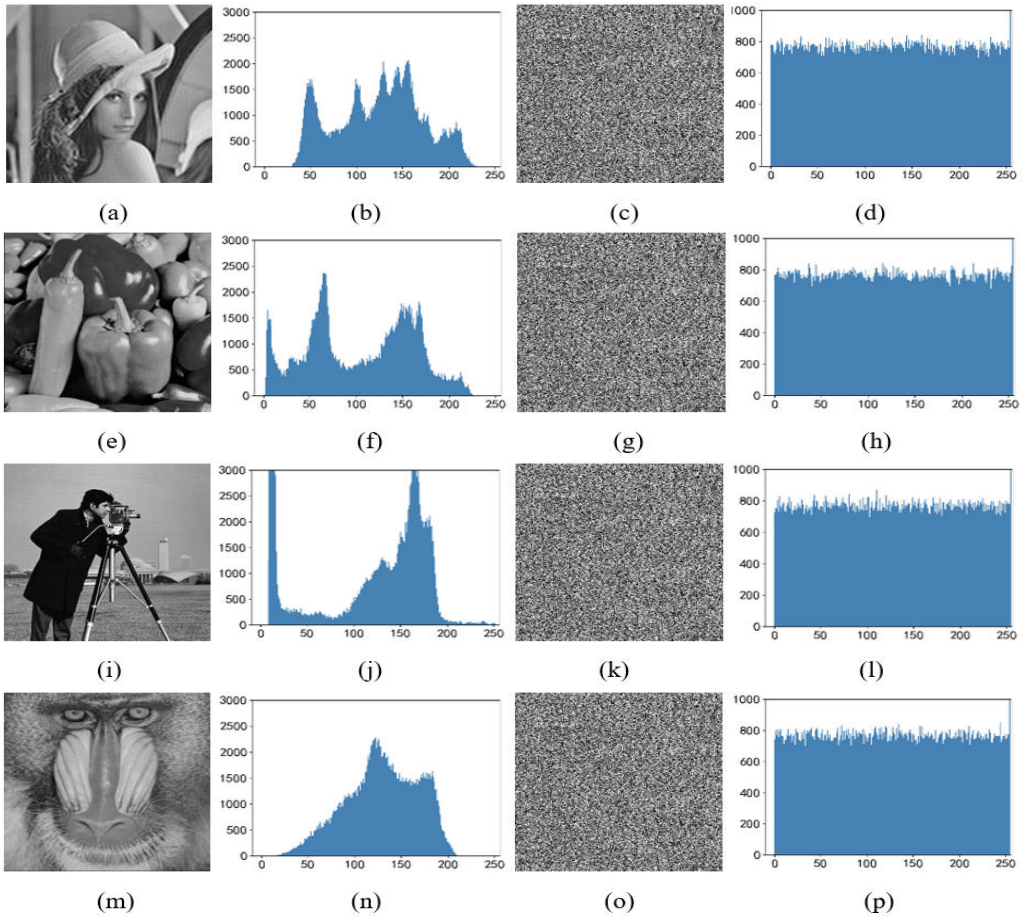
4.3 Histogram Analysis

The histogram [22] is used to determine the distribution of pixels in an image. In general, the ideal encrypted image histogram has a uniform frequency distribution and is highly resistant to statistical attacks. Figure 5 shows the histograms of the original images of the four test images and the corresponding encrypted images.

As can be seen from Figure 5, the histogram grey value distribution of plaintext images is uneven, while the distribution of histogram grey values of encrypted images tends to be flat. Therefore, the proposed algorithm can well hide the pixel distribution information of the original image and can resist the statistical analysis attack using the histogram information.

Table 3: Compared with the average encryption time and encryption throughput of Lena's images

Method	Image size	Encryption time /s	Encryption throughput /Mbps	Hardware environment
Proposed	256×256	0.8625	0.2174	Intel(R) Core(TM) i5-7572U /1.60GHz, 16GB
Ref. [19]	256×256	10.4400	0.0180	Intel Pentium N3540 /2.16GHz, 8GB
Ref. [16]	256×256	2.6264	0.0714	-
Ref. [5]	256×256	2.4433	0.0767	Intel i3 /2.53 GHz, 2 GB
Ref. [6]	256×256	2.2270	0.0842	Intel core i7 /2.90 GHz, 16GB
Ref. [22]	256×256	1.5000	0.5000	Intel(R) Core(TM)-i5 M450 16GB



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

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 5: Histogram analysis of the original image and encrypted image, (b)(d), (f)(h), (j)(i), (n)(p) are histograms of the original image and the encrypted image corresponding to the four test images, respectively.

Table 4: Code size comparison with existing methods

Method	Proposed	Ref. [17]	Ref. [10]
Code size (Byte)	21094	22016	441

4.4 Information Entropy Analysis

The information entropy [16] is a criterion for evaluating the distribution of grey values in images, which reflects the randomness of the distribution of grey values in images. The image information entropy H(s) is shown as

Table 5: Compared with the time complexity

Method	Time complexity
Proposed	$O((logN + 3/CR + 1)N^2)$
Ref. [5]	$O(2N^2(logN))$
Ref. [29]	$O(4^n)$
Ref. [2]	$O((logN + 11/CR + 1)N^2)$
Ref. [20]	$O(3N^2)$
Ref. [12]	$O(I(N^2))$

Table 6: Energy comparison

Method	Image	Energy
Proposed	Lena	0.01562
	Peppers	0.01563
	Camerman	0.01560
	Baboon	0.01562
Ref. [28]	Lena	0.01780
Ref. [16]	Baboon	0.01610
Ref. [15]	Baboon	0.01560
Ref. [29]	Lena	0.01564

Equation (17) (unit bits):

$$H(s) = -\sum_{i=1}^{m} P(s_i)\log_2 \frac{1}{P(s_i)} \tag{17}$$

where m represents the number of symbols (pixel grey levels) emitted by the source, s_i represents the grey value size of the image, and $P(s_i)$ represents the probability size of the symbol s_i.

The information entropy of the encrypted image should be as close as possible to the ideal value of 8. If the entropy value of the ciphertext image is less than 8, the plaintext image is predictable. Table 7 compares the entropy values of the proposed algorithm with the existing methods [2, 3, 6, 11, 19, 22, 29].

As can be seen from Table 7, the average value of the information entropy value of the proposed algorithm is higher than that of Ref. [3, 6, 11, 22, 29] and slightly lower than that of Ref. [2, 19], indicating that the information entropy of the proposed algorithm can be better than the ideal value than the existing methods, and it has strong resistance to entropy attacks.

4.5 Differential Attack Analysis

The differential attack [29] is common in cryptography. To secure the performance of the proposed algorithm against differential attacks, this paper uses the number pixels change rate(NPCR) and unified average changing intensity(UACI) to analyze. When one or more pixels in the plaintext image change, the ciphertext image changes significantly, and when the values of NPCR and UACI are close to the ideal values of 99.6094% and 33.4635%, the encryption is more resistant to differential attacks. Table 8 compares the differential attack performance of the proposed algorithm with the existing methods [2, 3, 6, 11, 19, 22, 29]. NPCR and UACI are defined as follows:

$$NPCR = \frac{\sum_{i,j} E(i,j)}{M \times N} \times 100\% \tag{18}$$

$$E(i,j) = \begin{cases} 0, I_1(i,j) = I_2(i,j) \\ 1, I_1(i,j) \neq I_2(i,j) \end{cases} \tag{19}$$

$$UACI = \frac{1}{M \times N} \frac{\sum\limits_{i,j} (I_1(i,j) - I_2(i,j))}{255} \times 100\% \tag{20}$$

where M and N are the width and height of the encrypted image, and I_1 and I_2 are the pixel values of the ciphertext image corresponding to the pixel value change.

As can be seen from Table 8, the difference between the NPCR mean and the ideal value of the proposed algorithm is lower than that of Ref. [2, 3, 6, 11, 19, 22, 29], which is closer to the theoretical value, and the difference between the mean and the ideal value of UACI is lower than that of Ref. [2,6,11,19,22,29], which is slightly higher than Ref. [3]. Therefore, the proposed algorithm can effectively resist differential attacks compared with existing methods.

4.6 Key Space Analysis

The key space is the ability to evaluate an algorithm's resistance to brute force attacks [5]. The security strength of lightweight encryption algorithms is generally required to be more than 80 bits, which is sufficient to resist brute force attacks. Therefore, to prevent the key from being cracked, from the perspective of security, the key space needs to be $\geq 2^{80}$, and when the key space is $\geq 2^{80} \approx 10^{24}$ can meet the security required by the device. The proposed algorithm encryption key requires five key values x_0, y_0, z_0, k, and u, the total key space of the proposed algorithm obtained by a large number of experimental tests is$10^{15} \times 10^{15} \times 10^{15} \times 10^{15} \times 10^{15} = 10^{75} \approx 2^{250}$. Table 9 compares the key space between the proposed algorithm and the existing methods [5, 9–11, 19, 22].

As can be seen from Table 9, the key space of the proposed algorithm is higher than that of Ref. [5, 9–11, 19]. Due to the use of cascading shadow mapping to generate encryption keys in Ref. [22], the complexity is high, making the key space slightly higher than the key space of the proposed algorithm.

4.7 Key Sensitivity Analysis

The key sensitivity [19] refers to the difference between ciphertext images obtained by encrypting the same plaintext image with the key before and after the change when the key changes slightly. Figure 6 shows the key sensitivity analysis of the Lena image. Where: Figure 6(a) is the original image, the ciphertext image, and the decrypted image obtained with the correct key. Figure 6(b) shows different decrypted images obtained using different error keys. Table 10 shows the comparison results of measuring

Table 7: Comparison with the information entropy of existing methods to encrypted image

Method	Lena	Peppers	Cameraman	Baboon	Average value
Proposed	7.9990	7.9993	7.9992	7.9993	7.9992
Ref. [11]	7.9967	-	-	7.9969	7.9968
Ref. [19]	7.9996	7.9993	7.9970	7.9992	7.9994
Ref. [3]	7.9993	7.9993	7.9995	7.9992	7.9987
Ref. [29]	7.9968	-	-	7.9964	7.9966
Ref. [6]	7.4077	-	-	7.9434	7.6756
Ref. [2]	7.9992	-	7.9993	7.9993	7.9993
Ref. [22]	7.9989	7.9989	-	7.9989	7.9989

Table 8: Comparison with NPCR and UACI values of existing methods

Method	Evaluation indicator	Lena	Peppers	Cameraman	Baboon	Mean value	Difference
Proposed	NPCR	99.6000	-	-	99.6038	99.6019	0.0071
	UACI	33.5550	33.4577	33.4460	33.4670	33.4814	0.0179
Ref. [11]	NPCR	99.5960	99.6020	99.6130	99.5988	99.6023	0.0075
	UACI	28.3875	-	-	27.3288	27.8582	5.6053
Ref. [19]	NPCR	99.6419	99.6196	99.6016	99.6153	99.6196	0.0102
	UACI	33.5582	33.5378	33.5737	33.5786	33.5621	0.0986
Ref. [3]	NPCR	99.6193	99.6414	99.6048	99.6060	99.6179	0.0085
	UACI	33.4860	33.4532	33.4856	33.4375	33.4656	0.0021
Ref. [29]	NPCR	99.6930	-	-	99.6840	99.6885	0.0791
	UACI	33.6100	-	-	33.4300	33.5200	0.0565
Ref. [6]	NPCR	99.3700	-	-	99.4900	99.4300	0.1794
	UACI	20.7500	-	-	23.3700	22.0600	11.4035
Ref. [2]	NPCR	99.6600	-	99.6300	99.6300	99.6400	0.0306
	UACI	33.5600	-	33.3400	33.6900	33.5300	0.0665
Ref. [22]	NPCR	99.5870	99.6010	-	99.6160	99.6013	0.0081
	UACI	30.7010	31.0360	-	27.8660	29.8677	3.5958

Table 9: Comparison with the key space of existing methods

Algorithm	Proposed	Ref. [11]	Ref. [19]	Ref. [5]	Ref. [10]	Ref. [9]	Ref. [22]
Key space	10^{75}	10^{24}	10^{60}	10^{56}	10^{24}	10^{24}	10^{96}

and evaluating the differences between different decrypted images by calculating the values of NPCR and UACI.

As can be seen from Figure 6, in the decryption process, when the decryption key changes slightly, the decrypted images are different, and it is difficult to identify the relevant information of the original image from these images, indicating that the proposed algorithm has a strong sensitivity to the key and can effectively resist brute force attacks.

It can be seen from Table 10 that the average values of NPCR and UACI are close to the theoretical values of 99.6094% and 33.4635%, indicating that the proposed algorithm is very sensitive to keys and can effectively resist exhaustive attacks.

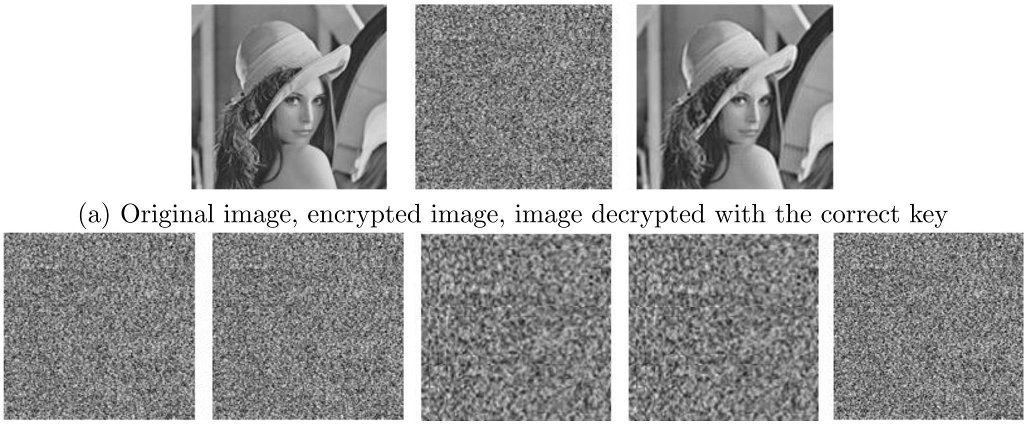
4.8 Security Performance Analysis of S-box

The proposed algorithm uses the characteristics of nonlinearity (NL), strict avalanche criterion (SAC), differential approximation rate (DP), and output bit-to-bit independence (BIC) to verify the performance of the generated S-box.

S-box nonlinearity (NL) [30] is a measure of resistance to linear cryptanalysis. In the security analysis of the S-box, the higher the value of nonlinearity, the stronger the S-box is resistant to attacks in the face of linear cryptanalysis. Suppose $f(x)$ is a Boolean function of n elements.

$$N_f = \underset{l \in L_n}{mid}\, d_H(f,l) \tag{21}$$

where N_f is the nonlinearity of $f(x)$, Ln is the set of linear sums of all n elements, and $d_H(f,\ l)$ is the Hamming



(a) Original image, encrypted image, image decrypted with the correct key

(b) Decrypted images with the key of $x_0 + 10^{-15}, y_0 + 10^{15}, z_0 + 10^{-15}, k + 10^{-15}, andu + 10^{-15}$

Figure 6: Key sensitivity analysis of Lena's image.

Table 10: NPCR and UACI comparison between decrypted images obtained with different decryption keys (%)

Evaluation indicator	x_0	y_0	z_0	k	u
NPCR	99.6146	99.6092	99.6185	99.5898	99.6036
UACI	33.7041	33.4465	32.7200	32.6928	33.1978

distance between f and l.

The proposed algorithm calculates the nonlinearity of the function through the Walsh spectrum, and the expression can be defined as:

$$S_{<f>}(w) = \sum_{x \in GF(2^n)} (-1)^{f(x) \oplus x \cdot w} \tag{22}$$

where $w \in GF(2^n)$, $x \cdot w$ is the dot product of x and w, and $x \cdot w = x_1 w_1 \oplus x_2 w_2 \oplus \ldots \oplus x_n w_n$.

The strict avalanche criterion (SAC) [4] states that when one input bit of a given sequence is modified, more than half of the output bits are altered. In general, the Boolean function is tested whether Boolean function meets the strict avalanche criterion by constructing a correlation matrix, and if the value of each element in the matrix is close to the ideal value of 0.5, it means that the S-box meets the SAC criterion, and Table 11 shows the SAC correlation matrix of the proposed algorithm. Among them, the maximum and minimum values are 0.5812 and 0.4375, respectively, and the average value is 0.4992, which is a 0.0008 difference from 0.5000.

Differential approximation rate (DP) [15] is an evaluation criterion to measure the ability of S-boxes to resist differential cryptanalysis and is usually expressed as DP_f as follows:

$$DP_f = \max_{\Delta x \neq 0, \Delta y} \left(\frac{\#\{x \in GF(2^n)\}}{2^n} \right) \left(\frac{|f(x) \oplus f(x+\Delta x) = \Delta y|}{2^n} \right) \tag{23}$$

where $GF(2^n)$ represents the set of inputs, 2^n represents the number of all set elements, Δx and Δy represent the input difference and output difference, respectively, and DP_f represents the maximum probability of the output. A higher value of DP_f means more resistance to attacks. Conversely, the weaker the ability to resist attack.

For the output inter-bit independence (BIC) performance test of the S-box, the value of any two output bits f_i, f_k $(i \neq k)$ XOR operation can be used to determine whether the value of the XOR operation meets the strict avalanche criterion, if the obtained value is nonlinear and meets the strict avalanche effect, it can be ensured that when a bit is reversed, the correlation coefficient of each output bit pair is close to 0, that is, BIC is satisfied.

Table 12 compares the security performance of the proposed algorithm compared with the existing methods [4,5,13,15,30] S-box.

As can be seen from Table 12, the mean NL of the proposed algorithm is 107.00, which is higher than the average value of Ref. [5,13,15,30], because Ref. [4], uses a chaotic system based on elliptic curve points to construct the S-box, which is highly complex, so its NL value is higher than that of the proposed algorithm, indicating that the S-box constructed by the proposed algorithm has strong resistance in the face of linear cryptanalysis. Compared with Ref. [4,5,13,15,30] , the SAC value of the proposed algorithm is closer to the ideal value of 0.5000, indicating that the S-box constructed by the proposed algorithm meets the strict avalanche criterion and has a good security performance. Compared with Ref. [4,5,13,15,30] , the maximum difference value of the algorithm in Ref. [5,13,15,30] is 10, the maximum difference value of Ref. [4] is 12, and the maximum difference value of S-box of the proposed algorithm is 10, indicating that the difference uniformity of S-box constructed in the proposed algorithm is better and the ability to resist at-

Table 11: SAC correlation matrix

0.5172	0.4688	0.5490	0.5781	0.4823	0.4844	0.5332	0.4844
0.5000	0.4980	0.4688	0.4823	0.4531	0.5333	0.4862	0.5137
0.5176	0.4688	0.5313	0.4705	0.5812	0.4666	0.5000	0.5088
0.4375	0.4667	0.5288	0.5000	0.5220	0.5062	0.4475	0.5315
0.4392	0.4844	0.4896	0.5413	0.4980	0.4549	0.5256	0.4531
0.5156	0.5137	0.5023	0.4688	0.5313	0.5313	0.5000	0.5176
0.5000	0.5156	0.4844	0.5469	0.4844	0.4888	0.4375	0.5019
0.5137	0.4862	0.5356	0.5625	0.4531	0.4844	0.4705	0.4990

Table 12: Comparison of S-box safety performance with existing methods

Method	NL(AVG)	SAC	The difference from 0.5000	DP	BIC-SAC	The difference from 0.5000
Proposed	107.00	0.4992	0.0008	10	0.4992	0.0008
Ref. [30]	104.00	0.4988	0.0012	10	0.5052	0.0052
Ref. [5]	106.25	0.5009	0.0009	10	0.4996	0.0040
Ref. [4]	108.00	0.4988	0.0012	12	0.4969	0.0031
Ref. [15]	106.50	0.5009	0.0009	10	0.4990	0.0010
Ref. [13]	106.25	0.5086	0.0086	10	0.5010	0.0010

tack is strong. The mean values of the Ref. [4,5,13,15,30] schemes were 0.5052, 0.4996, 0.4969, 0.4990, and 0.5010, respectively, and the average value of BIC-SCA in the proposed algorithm was 0.4992, which was only 0.0010 different from 0.5000. Comparative analysis shows that the output of the proposed algorithm is more independent than the inter-cells, indicating that the S-box constructed in the proposed algorithm has obvious advantages in resisting the attacks of differential and linear cryptanalysis.

In summary, the dynamic S-box constructed by the proposed algorithm performs well in nonlinearity, strict avalanche criterion, differential approximation rate and independence between output bits, it can effectively resist linear cryptanalysis attacks and differential cryptanalysis attacks, and has good security, which indicates that the chaotic dynamic S-box generated by the proposed algorithm has good nonlinearity and attack resistance.

5 Conclusions

In this paper, a lightweight image encryption algorithm based on a dual chaotic system and dynamic S-box is proposed, which considers the balance between throughput, security level and performance, and better solves the shortcomings of existing image encryption algorithms in terms of decryption image quality, encryption throughput, time complexity, memory usage and resistance statistics. The proposed algorithm uses DCT coding to compress the original image, effectively reducing the amount of encrypted data and improving the encryption speed. The designed Logistic-Tent dual chaotic system has better key space and randomness, which can generate better random key sequences. The generated random key sequence is XOR with the dynamic S-box to enhance the security of the encrypted image. The experimental results show that the quality of the decrypted image with the proposed algorithm is good, the time complexity is low, the encryption throughput is high, the security is high, and the defense against common attacks is effective. The proposed algorithm can maximize the requirements of device security interaction in MIoT and can be better adapted to constrained environments.

The disadvantage is that the proposed algorithm uses DCT-based compression technology in compression, and there are still some shortcomings in compression efficiency. Therefore, the next improvement direction is to further improve the efficiency of the encryption algorithm while ensuring the security of the image.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 61862041). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

[1] Y. Alghamdi, A. Munir and J. Ahmad, "A Lightweight Image Encryption Algorithm Based on Chaotic Map and Random Substitution," *Entropy*, vol. 24, no. 10, 1344, 2022.

[2] I. T. Almalkawi, R. Halloush, A. Alsarhan, et al., "A lightweight and efficient digital image encryption using hybrid chaotic systems for wireless network applications," *Journal of Information Security and Applications*, vol. 49, 102384, 2019.

[3] J. Arif, M. A. Khan, B. Ghaleb, et al., "A novel chaotic permutation-substitution image encryption

scheme based on logistic map and random substitution," *IEEE Access*, vol. 10, pp. 12966–12982, 2022.

[4] M. B. Farah, N. A. Azam, R. Guesmi, et al., "A new design of cryptosystem based on S-box and chaotic permutation," *Multimedia Tools Applications*, vol. 79, no. 27, pp. 19129–19150, 2020.

[5] M. B. Farah, A. Farah and T. Farah, "An image encryption scheme based on a new hybrid chaotic map and optimized substitution box " *Nonlinear Dynamics*, vol. 99, no. 4, pp. 3041–3064, 2020.

[6] J. Ferdush, M. Begum and M. S. Uddin, "Chaotic lightweight cryptosystem for image encryption,"*Advances in Multimedia*, vol. 2021, Article ID 5527295, 2021.

[7] X. Y. Gao, J. Mou, S. Banerjee, et al., "An effective multiple-image encryption algorithm based on 3D cube and hyperchaotic map," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 4, pp. 1535–1551, 2022.

[8] X. Y. Gao, J. Mou, L. Xiong, et al., "A fast and efficient multiple images encryption based on single-channel encryption and chaotic system," *Nonlinear Dynamics*, vol. 108, no. 1, pp. 613–636, 2022.

[9] M. Gupta, K. K. Gupta, M. R. Khosravi, et al., "An intelligent session key-based hybrid lightweight image encryption algorithm using logistic-tent map and crossover operator for internet of multimedia things," *Wireless Personal Communications*, vol. 121, no. 3, pp. 1857–1878, 2021.

[10] M. Gupta, K. K. Gupta and P. K. Shukla, "Session key based novel lightweight image encryption algorithm using a hybrid of Chebyshev chaotic map and crossover," *Multimedia Tools and Applications*, vol. 80, no. 25, pp. 33843–33863, 2021.

[11] M. Gupta, K. K. Gupta and P. K. Shukla,"Session key based fast, secure and lightweight image encryption algorithm," *Multimedia Tools and Applications*, vol. 80, no. 7. pp. 10391–10416, 2021.

[12] M. K. Hasan, S. Islam, R. Sulaiman, et al., "Lightweight encryption technique to enhance medical image security on internet of medical things applications," *IEEE Access*, vol. 9, pp. 47731–47742, 2021.

[13] A. Hayat, A. M. Abbas, S. Naz, et al., "A truly dynamic substitution box generator for block ciphers based on elliptic curves over finite rings," *Arabian Journal for Science and Engineering*, vol. 46,no. 9, pp. 8887–8899, 2021.

[14] R. Hedayati and S. Mostafavi, "A lightweight image encryption algorithm for secure communications in multimedia Internet of Things," *Wireless Personal Communications*, vol. 123, no. 2, pp. 1121–1143, 2021.

[15] S. Ibrahim and A. M. Abbas, "Efficient key-dependent dynamic S-boxes based on permutated elliptic curves," *Information Sciences*, vol. 558, pp. 246–264, 2021.

[16] B. Idrees, S. Zafar, T. Rashid, et al., "Image encryption algorithm using S-box and dynamic Hénon bit level permutation," *Multimedia Tools and Applications*, vol. 79, no. 9, pp. 6135–6162, 2020.

[17] A. Kifouche, M. S. Azzaz, R. Hamouche, et al., "Design and implementation of a new lightweight chaos-based cryptosystem to secure IoT communications," *International Journal of Information Security*, vol. 21, no. 6, pp. 1247–1262, 2022.

[18] A. Kumar and N. Raghava, "An efficient image encryption scheme using elementary cellular automata with novel permutation box," *Multimedia Tools and Applications*, vol. 80, no. 14, pp. 21727–21750, 2021.

[19] C. M. Kumar, R. Vidhya and M. Brindha, "An efficient chaos based image encryption algorithm using enhanced thorp shuffle and chaotic convolution function," *Applied Intelligence* , vol. 52, no. 3, pp. 2556–2585, 2022.

[20] H. Liu, B. Zhao, J. W. Zou, et al., "A lightweight image encryption algorithm based on message passing and chaotic map," *Security and Communication Networks* , vol. 2020, Article ID 7151836, 2020.

[21] Z. H. Lv, L. Qiao and H. B. Song, "Analysis of the security of Internet of multimedia things," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 16, no. 3s, pp. 1–16, 2020.

[22] B. Mondal and J. P. Singh, "A lightweight image encryption scheme based on chaos and diffusion circuit," *Multimedia Tools and Applications*, vol. 81, no. 24, pp. 34547–34571, 2022.

[23] S. M. Wang, C. Y. Li, Q. Q. Peng, et al., "Chaotic color image encryption based on 4D chaotic maps and DNA sequence," *Optics & Laser Technology*, vol. 148, 107753, 2022.

[24] X. Y. Wang, S. Gao, L. Tian, et al., "Image encryption algorithm for synchronously updating Boolean networks based on matrix semi-tensor product theory," *Information sciences*, vol. 507, pp. 16–36, 2020.

[25] B. Xing, D. D. Wang, Y. Q. Yang, et al., "Accelerating DES and AES Algorithms for a Heterogeneous Many-core Processor," *International Journal of Parallel Programming*, vol. 49, pp. 463–486, 2021.

[26] S. C. Xu, X. Y. Wang and X. L. Ye, "A new fractional-order chaos system of Hopfield neural network and its application in image encryption," *Chaos, Solitons & Fractals*, vol. 157, 111889, 2022.

[27] G. D Ye, K. X. Jiao and X. L. Huang, "Quantum logistic image encryption algorithm based on SHA-3 and RSA," *Nonlinear Dynamics*, vol. 104, no. 3, pp. 2807–2827, 2021.

[28] A. Yousaf, A. Razaq and H. Baig, "A lightweight image encryption algorithm based on patterns in Rubik's revenge cube," *Multimedia Tools and Applications*, vol. 81, no. 20, p. 28987–28998, 2022.

[29] S. H. Zahid, M. Ahmad, A. Alkhayyat, et al., "Efficient dynamic S-box generation using linear trigonometric transformation for security applications," *IEEE Access*, vol. 9, pp. 98460–98475, 2021.

[30] J. M. Zheng and Q. X. Zeng, “An image encryption algorithm using a dynamic S-box and chaotic maps,” *Applied Intelligence*, vol. 52, no. 13, pp. 15703–15717, 2022.

Biography

Qiu-yu Zhang Researcher/Ph.D. supervisor, graduated from Gansu University of Technology in 1986, and then worked at school of computer and communication in Lanzhou University of Technology. He is vice dean of Gansu manufacturing information engineering research center, a CCF senior member, a member of IEEE and ACM. His research interests include network and information security, information hiding and steganalysis, multimedia communication technology.

Rui-hong Chen is currently a master's student of the School of Computer and Communication, Lanzhou University of Technology, China. She received her BS degree in computer science and technology from Kashi University, Xinjiang, China, in 2021. Her research interests include network and information security, multimedia data security and research on lightweight image encryption methods.

Ling-tao Meng is currently a master's student of the School of Computer and Communication, Lanzhou University of Technology, China. He received a BS degree in computer science and technology from the Lanzhou University of Technology, Gansu, China, in 2022. His research interests include network and information security, deep learning and image retrieval.

Yi-lin Liu is currently a master's student of the School of Computer and Communication, Lanzhou University of Technology, China. She received a BS degree in software engineering from Lanzhou University of Technology, in 2020. Her research interests include network and information security, multimedia data security and research on lightweight image encryption methods.