



# SPJEU: a self-sufficient plaintext-related JPEG image encryption scheme based on a unified key<sup>\*#</sup>

Ming LI<sup>1,2</sup>, Wenwen ZHOU<sup>1</sup>, Mengdie WANG<sup>3</sup>, Yushu ZHANG<sup>‡4</sup>, Yong XIANG<sup>5</sup>

<sup>1</sup>College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China

<sup>2</sup>Key Laboratory of Artificial Intelligence and Personalized Learning in Education of Henan Province, Xinxiang 453007, China

<sup>3</sup>School of Information Engineering, Anyang Vocational and Technical College, Anyang 455099, China

<sup>4</sup>College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

<sup>5</sup>School of Information Technology, Deakin University, Victoria 3125, Australia

E-mail: [liming@htu.edu.cn](mailto:liming@htu.edu.cn); [2205432344@qq.com](mailto:2205432344@qq.com); [207281083@qq.com](mailto:207281083@qq.com); [yushu@nuaa.edu.cn](mailto:yushu@nuaa.edu.cn); [yong.xiang@deakin.edu.au](mailto:yong.xiang@deakin.edu.au)

Received Aug. 18, 2024; Revision accepted Nov. 11, 2024; Crosschecked

**Abstract:** In recent research on image encryption, many schemes associate the key generation mechanism with the plaintext to resist chosen plaintext attacks. However, when the sender encrypts a large number of images, a large amount of additional data related to the plaintext needs to be transmitted, which leads to problems such as high transmission costs, high requirements for key storage space, and complex key management. Therefore, in this paper we propose a self-sufficient plaintext-related JPEG image encryption scheme based on a unified key (SPJEU). This scheme establishes the connection between the plaintext and the key by selecting the DC coefficients in the JPEG image through a unified key. Homomorphic encryption is applied to the selected DC coefficients, allowing plaintext information to be decrypted directly from the ciphertext domain using a specific calculation method. The remaining DC coefficients are encrypted through group diffusion, and AC coefficients are grouped and permuted based on the run length. Extensive experiments show that our scheme can resist chosen plaintext attacks, avoids transmitting plaintext-related data related to the plaintext in the communication channel, and simplifies key management. This scheme also ensures the security and format compatibility of the ciphertext image, and the file increment after encryption is very small.

**Key words:** Image encryption; self-sufficient; plaintext-related; JPEG; homomorphic encryption

<https://doi.org/10.1631/FITEE.2400721>

**CLC number:** TP

## 1 Introduction

Digital images are an important multimedia resource on the Internet and many are posted on network platforms every day. For convenience, users store or share a large amount of online image data to the cloud. To prevent privacy leakage, the most

common solution is image encryption (Li et al., 2022). However, traditional encryption methods such as AES and DES are not suitable for image encryption, because the data contained in the image have redundancy and correlation (Liu et al., 2016). Effective image encryption can be performed in either the spatial domain or frequency domain. Spatial domain encryption involves encrypting the pixels of the image directly. Frequency domain encryption means that the image is transformed from the spatial domain to the frequency domain using transforms such as Fourier, discrete cosine, or wavelet, and the coefficients in the frequency domain are encrypted (Refrégier and Javidi, 1995).

The main techniques used in both spatial image

<sup>‡</sup> Corresponding author

<sup>\*</sup> Project supported by the Key Program of the Higher Education Institutions of Henan Province (No. 23A520009)

ORCID: Ming LI, <http://orcid.org/0000-0003-3385-8364>; Yushu ZHANG, <http://orcid.org/0000-0001-8183-8435>

© Zhejiang University Press 2024

encryption and frequency image encryption are scrambling and diffusion (Yi et al., 2018; Mohammadi et al., 2020; Zheng et al., 2017; Xian et al., 2022). For example, Yi et al. (2018) and Mohammadi et al. (2022) used a stream cipher to encrypt images by block-based scrambling and pixel-XOR diffusion. Zheng et al. (2017) combined DNA encoding with an XOR operation. Xian et al. (2022) used the STFSM chaotic system to scramble and diffuse images. At present, the most used image format on the Internet is JPEG, so it is meaningful to encrypt JPEG images (Li and Yuan, 2017). Most JPEG encryption algorithms encrypt mainly DC coefficients and AC coefficients. Li and Yuan (2017) and Li and Zhang (2016) performed global scrambling of DC coefficients in the frequency domain of the images. Qin et al. (2023) proposed an adaptive DC coefficient prediction and partial RS pair permutation encryption algorithm, which does not increase the size of the ciphertext image. In the algorithm of Ting et al. (2019), the prediction error of the DC coefficient is encrypted by the group, and the scrambling of the RS pair is used for the AC coefficient. He et al. (2016) performed modulo operation encryption only on the DC coefficients and the first 14 AC coefficients sorted by “Zigzag”. Li and Lo (2019) changed the size of sub-blocks, performed DCT on sub-blocks with a size of  $16 \times 16$ , and then changed the size of the sub-blocks to  $8 \times 8$  in the quantization process, followed by global scrambling and DC coefficient XOR between sub-blocks. The scheme of Ong et al. (2015) performs regional scrambling on DC coefficients and extracts all RS pairs in sub-blocks for global scrambling. The method of Li and Lo (2018) and (2017) uses a new orthogonal transform to replace the discrete cosine transform, and achieves a good compression effect. Researchers also proposed encryption algorithms based on JPEG bitstream in the entropy coding stage. Qian et al. (2014) and Qian et al. (2018) encrypted the additional bits of DC and AC coefficients using a stream cipher. Qian et al. (2019) performed encryption operations on the quantization table and the DCA additional bits of the DC coefficients, and performed in-block scrambling on the RS pairs of the AC coefficients. He et al. (2019) and (2018) permuted the DCC with the same sign and scrambled the ACC with the same run length.

The encryption mechanisms of Yi et al. (2018),

Mohammadi et al. (2020), Li and Yuan (2017) and Li and Zhang (2016) are not associated with plaintext, and only the key and ciphertext need to be transmitted during transmission. However, such encryption mechanisms are fragile to chosen-plaintext attacks and known-plaintext attacks. Qu et al. (2022) pointed out that the encryption methods of Yi et al. (2018) and Mohammadi et al. (2020) could not resist a known plaintext attack and proposed an improved method to generate the key associated with the plaintext by using a hash algorithm. Li et al. (2019) obtained contours of plaintext images (Li and Yuan, 2007; Li and Zhang, 2016) by chosen plaintext attacks and non-zero count attacks.

Therefore, to improve security, researchers have proposed encryption mechanisms associated with plaintext (Zheng et al., 2017; Xian et al., 2022; Qin et al., 2022; He et al. 2019) that use hash algorithms to associate secret keys with the plaintext of the image. Zheng et al. (2017) and Qin et al. (2023) used the SHA-256 hash algorithm, Xian et al. (2022) used the SHA-512 algorithm, and He et al. (2018) used the SHA3-512 hash algorithm. But if someone encrypts and transmits a large number of ciphertext images, there will be many problems. For example, (1) since the key is associated with the plaintext, the additional data related to the plaintext need to be transmitted together with the key and the ciphertext image. When multiple ciphertext images are sent to the receiver, the amount of extra data transmitted increases significantly, thereby increasing the transmission cost. (2) Different ciphertext images must be decrypted using different decryption keys, which consist of the encryption keys and the data related to the plaintext. Managing so many different decryption keys is a huge challenge for the receiver. The receiver must know the one-to-one match between the decryption key and the ciphertext image. If there is a slight error in the correspondence, multiple ciphertext images may not be decrypted. In addition, current JPEG encryption schemes have their own drawbacks. For example, in the schemes of Li and Yuan (2007), Li and Zhang (2016) and He et al. (2016), security is improved, but the file change rate is increased. In the encryption process of Qin et al. (2023) and Ong et al. (2015), DC and AC coefficients in sub-blocks may overflow. Therefore, Qin et al. (2023) also embedded these overflow data into the ciphertext image to ensure the

compatibility of the encrypted image format. The schemes of Ting et al. (2019) and He et al. (2019 and 2018) require multiple rounds of encryption to avoid file format incompatibilities and ensure image security. If the sender needs to encrypt a large number of images, it will cost a huge amount of time. The exclusive-OR operation of the DC coefficient by Li and Lo (2019) causes the DC coefficient to overflow, leading to format incompatibility.

To address the shortcomings of the current image encryption mechanisms associated with plaintext, in this paper we propose a self-sufficient plaintext-related image encryption model based on a unified key (SPEU), in which the required information associated with the plaintext can be extracted from the encrypted image itself by the legal receiver with the secret keys. An illustration of the three types of image encryption schemes mentioned above, i.e. the traditional scheme, the plaintext-related scheme and SPEU, is provided in Supplementary Material 1.

Based on SPEU, we propose a self-sufficient plaintext-related JPEG image encryption scheme based on a unified key (SPJEU). In SPJEU, the DC coefficients related to the plaintext are directly selected in the plaintext image through a unified key to generate different key streams for different images, and different encryption effects are achieved for different images. The selected DC coefficients are encrypted by homomorphic technology so that the same information associated with the plaintext can be extracted in the ciphertext domain using a specific homomorphic calculation method. Therefore, in the decryption process, the receiver can extract the plaintext information and cooperate with the key to generate different decryption key streams for different images. The remaining DC coefficients are encrypted in groups to reduce damage to the correlation of the DC coefficients, so the change rate of the ciphertext file is small. The AC coefficients are first grouped according to run length, then permuted in groups, and finally, the sub-blocks except the DC coefficients are globally scrambled. The proposed encryption mechanism not only has the ability to resist the chosen plaintext attack but also does not generate additional information associated with the plaintext, which solves the fundamental problems in the existing plaintext-related image encryption mechanisms. The transmission cost is greatly reduced, the required key

storage space is small, and the unified key is easy to manage. The receiver can obtain the decryption key through the transmitted unified key and the ciphertext image to be decrypted. The relationship between the decryption key and the corresponding ciphertext image is very clear, which ensures the correct decryption of multiple images. Furthermore, compared with current JPEG encryption algorithms, the JPEG encryption scheme proposed here can ensure the security and format compatibility of the image without paying extra overhead and additional costs of multiple rounds of encryption, and the file growth rate is small.

The main contributions of this paper are as follows:

(1) We propose the SPEU model, which overcomes the shortcomings of existing plaintext-related encryption mechanisms. The SPEU model enables the legal receiver to extract the required plaintext-related information from the encrypted image itself using secret keys.

(2) Based on the SPEU model, we propose the SPJEU scheme for widely used JPEG images. This scheme selects DC coefficients related to the plaintext in the plaintext image through a unified key, generating different key streams for different images and achieving more secure encryption effects.

(3) We conducted a comprehensive evaluation, which shows that our scheme effectively resists chosen-plaintext attacks and avoids transmitting plaintext-related data, thereby reducing transmission costs, and minimizing key storage. Additionally, the scheme has passed various security analyses, ensuring the security of the ciphertext image.

## 2 The basic SPEU

The basic SPEU is shown in Fig. 1. To achieve self-sufficiency and plaintext-relatedness simultaneously, the plaintext-related information must be computed directly from the original image or the encrypted image. Therefore, the image should be divided into the plaintext-related part and the remaining part. The first part is used for computing the plaintext-related information and will be encrypted homomorphically so that the same information associated with the plaintext can be extracted from the encrypted image. The second part is encrypted by the

unified key with the plaintext-related information obtained from the image itself. Although the unified key is consistent, the plaintext-related information is different, and the pseudo-random sequences generated by the unified key and the plaintext-related information for encrypting different images are different, which can resist the chosen plaintext attack effectively. The plaintext-related part of the image to be encrypted is securely selected by the unified key, and can also be found from the encrypted image in the decryption process according to the secret key. When decryption is performed, the plaintext-related information is first extracted from the plaintext-related part of the encrypted image itself based on homomorphic computing. Then, the pseudo-random sequences used to decrypt the target image can be obtained by combining the unified key and the plaintext-related information, and the decryption will be successful. In the SPEU model, the plaintext-related information involved in the key generation is contained in the plaintext or ciphertext image. During encryption or decryption, any additional plaintext-related information does not need to be transmitted separately, since it can be extracted from the image through a unified key. Therefore, this model is "self-sufficient".

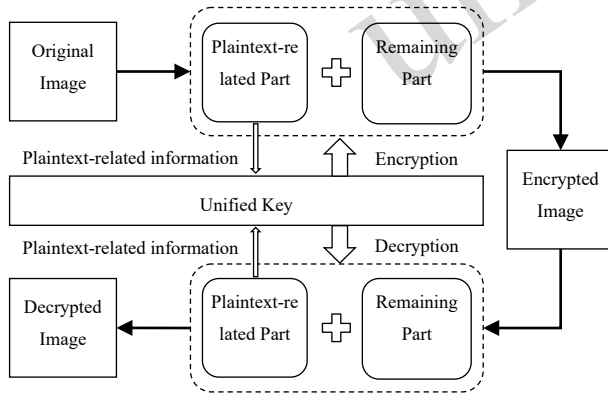


Fig. 1 Illustration of the basic SPEU.

### 3 Proposed SPJEU

JPEG encoding includes four main steps: block division, DCT transforming, quantization, and entropy encoding (Li and Lo, 2020). The encryption process mainly encrypts the quantized coefficients. Fig. 2 describes the entire encryption process of SPJEU. First, the image is subjected to block division,

DCT transformation, and quantization processing, and then the quantized DC coefficients and AC coefficients are encrypted. After encryption, entropy encoding is performed to finally obtain an encrypted JPEG image. Only the secret keys, including  $key_1$ ,  $key_2$ ,  $key_3$ ,  $key_4$ ,  $key_5$  and  $key_6$ , should be sent to the receiver through a secure channel, which is the same as in other symmetric encryption schemes. No additional plaintext-related data are required for transmission. Although the secret keys are unified, the generated keystreams are different, since the encryption is still plaintext-related. The main symbols in the proposed scheme are shown in Table 1.

The range of the quantized DC coefficients is  $[min, max]$ . The values of  $max$  and  $min$  are obtained by Eq. (1).

$$max = \frac{1023}{Qua}, min = \frac{-1024}{Qua}, \quad (1)$$

where  $Qua$  is the value of the DC coefficient in the quantization table, and the quantization tables under different quantization factors are different.

The key streams required for encryption of the quantized DC coefficients and AC coefficients are different ( $z_1$  and  $z_2$  are used for DC coefficients,  $z_3$  and  $z_4$  are used for AC coefficients), and the generation of the key streams is related to the plaintext (the specific process of keystream generation is shown in the Supplementary Material 2).

#### 3.1 Encryption of Quantized DC coefficients

The quantized DC coefficient  $D$  is divided into two parts,  $D=[B, Q]$ . The number of DC coefficients contained in  $B$  is 16, and the number contained in  $Q$  is  $S_1=S-16$ . Firstly, the DC coefficient  $B$  that constitutes the initial key is encrypted, and then the remaining DC coefficient  $Q$  is encrypted. The two encryption methods are different. After encryption, the two parts of the DC coefficients are recombined according to the original position (an example of the specific process is provided in the Supplementary Material 3).

##### 3.1.1 The encryption of the DC coefficients used for computing the initial key

Since the DC coefficients used for computing the initial key contain coefficients less than 0, to facilitate encryption operations, all coefficients in  $B$  need to be added to  $|min|$  to obtain  $B_1$ , and the range of  $B_1$  is  $[0, max+|min|]$ . In addition, to extract the same

plaintext-related information from plaintext and ciphertext, it is necessary to map operations on ciphertext to plaintext, ensuring that the encryption is not affected by information extraction. To solve this problem, it is necessary to consider both the encryption method and the information extraction method. This scheme adopts the modulo ( $L = \max + |\min| + 1$ )-based homomorphic technique to encrypt the DC coefficients that are used for compu-

ting the initial key. The process of the homomorphic encryption algorithm using the generated key stream is:  $C = E_n(B_1, H_2) = (B_1 + H_2) \bmod (L) = (b_i + h_i) \bmod (L) = c_i, \forall i = 1, 2, \dots, 16$ , where  $b_i, h_i$ , and  $c_i$  represent the DC coefficient of  $B_1$ , the random numbers in the key stream  $H_2$ , and the ciphertext DC coefficient, respectively. The decryption process is:  $D_n(C, H_2) = (c_i - h_i) \bmod (L) = b_i \bmod (L), \forall i = 1, 2, \dots, 16$ .

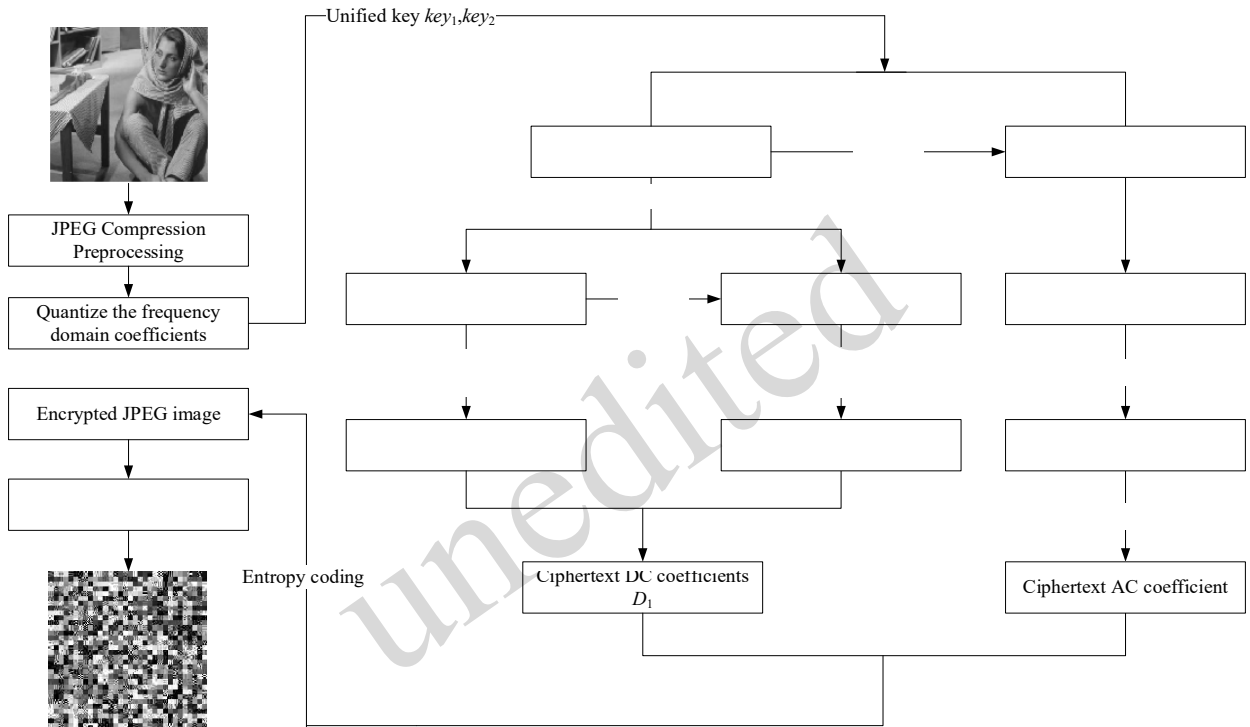


Fig. 2 A flow chart of SPJEU

Table 1 Definitions of several symbols used to describe our SPJEU

Symbols	Descriptions
$M, N$	The height and width of the original image
$key_1, key_2, key_3, key_4, key_5, key_6,$	Six unified keys
$S$	The number of $8 \times 8$ sub-blocks in the image
$k_1, k_2$	Pseudo-random sequences generated using by $key_1$ and $key_2$
$H_1, H_2$	Sequences obtained from $k_1$ and $k_2$ after processing
$W$	Sequence formed by the first 16 elements of $H_1$
$min, max$	The minimum and maximum values of the quantized DC coefficients
$B$	Sequence of 16 DC coefficients extracted from sub-blocks based on $W$
$L$	Parameter used for modulo operations, $L = \max +  \min  + 1$
$A$	The AC coefficient sequence
$Q$	The sequence of remaining DC coefficients after extracting $B$
$Qua$	The value of the DC coefficient in the quantization table
$D$	The sequence of all quantized DC coefficients
$A_i$	Set of AC coefficients with the same run length

Homomorphic encryption means that the results obtained by data operations in the ciphertext domain and in the plaintext domain are consistent, i.e. computing before decryption is equivalent to computing after decryption (Xie et al., 2019).

The homomorphic encryption algorithm based on modulo ( $L$ ) can just control the range of the encrypted DC coefficients in  $[0, \max + |\min|]$ . When the encrypted DC coefficient is minus  $|\min|$ , the coefficient range is  $[\min, \max]$ , so the problem of DC coefficient overflow does not occur (the proof of homomorphism is given in the Supplementary Material 4).

After homomorphic encryption, we scramble  $B_2$  using a pseudo-random sequence  $k$  derived from a pseudo-random generator, such as those of Haider et al. (2023) and Ramesh and Jain (2015), based on  $key_1 + key_2$  and an initial value  $Initx$  computed by  $(initial_1 + initial_2) \bmod L$  (see Eq. (S5) of the Supplementary Material 2). The scrambling method is similar to that of Qin et al. (2022).

### 3.1.2 Encryption of the remaining DC coefficients

To prevent the overflow of DC coefficients and the increase of the compression rate of ciphertext files, we encrypt the remaining DC coefficients  $Q$  in groups. Four consecutive DC coefficients are grouped into one group diffused using the same random number. The specific process is as follows:

Step 1: Perform encryption preprocessing on the remaining DC coefficients,  $Q_1 = Q + |\min|$ .

Step 2: Group the remaining DC coefficients, each set of 4 is a group, and the remainder of less than 4 DC coefficients are also a group, giving a total of  $(m+n)$  groups. The DC coefficient after grouping is  $Q_2 = [q_1, q_2, \dots, q_{m+n}]$ ,  $q_1 = [a_1, a_2, a_3, a_4]$ .  $a_i$  refers to the remaining DC coefficients, and  $fix$  refers to the quotient function.

$$m = \text{fix}\left(\frac{S-16}{4}\right), n = (S-16) \bmod 4 \quad (2)$$

Step 3: Perform (13) and (14) preprocessing on the pseudo-random sequences  $z_1$  and  $z_2$  to obtain key stream sequences  $Z_1$  and  $Z_2$ .

$$Z_1 = ((z_1 + z_2) \times 10^{13}) \bmod L \quad (3)$$

$$Z_2 = Z_1(1:(m+n)) \quad (4)$$

Step 4: Diffusion is performed in units of groups, and the process of (15) is performed on  $Q_2$  to obtain

the encrypted result  $Q_3$  after diffusion.

$$Q_3(i) = (Q_2(i) + Z_2(i)) \bmod L, i = 1, 2, \dots, (m+n) \quad (5)$$

Step 5:  $Q_4 = Q_3 - |\min|$ .  $Q_4$  is the final encrypted residual DC coefficient.

The encrypted DC coefficient  $B_2$  and the encrypted residual DC coefficient  $Q_4$  are recombined according to the original position. Finally, all encrypted DC coefficients  $D_1 = [B_2, Q_4]$ .

## 3.2 Encryption of AC coefficients

We adopt two encryption methods for the AC coefficients, namely scrambling the AC coefficients of the same run-length and global scrambling the sub-blocks except for the DC coefficients. The AC coefficient coding is determined by their run-length and value. By scrambling only the AC coefficients with the same run length and the sub-blocks except the DC coefficients, the amount of compression of the ciphertext image will not change (an example of the specific process is provided in the Supplementary Material 5).

### 3.2.1 Globally scrambling AC coefficients with the same run length

Step 1: First, the coefficients in all sub-blocks in the DCT image are sorted by Zigzag scan, and then the AC coefficients are grouped according to their run lengths, where  $A_i$  is the set of AC coefficients whose run length is  $i$ . The frequency of the run length of different images is different, and not all run lengths will appear, so  $i \in \{0, 1, 2, \dots, 63\}$ , as shown in the Supplementary Material 6.)

Step 2: Group the key stream  $z_3$  according to the number of AC coefficients in  $A_i$ , where  $Z_{3i}$  is the key stream required to scramble  $A_i$ .

Step 3: Scramble the AC coefficients of the same run length. Sort  $Z_{3i}$  in ascending order, and use the sorted index value to scramble the AC coefficient set  $A_i$ , whose run length is  $i$ . The scrambling method is similar to that of Qin et al. (2022).

Step 4: Reintegrate the scrambled AC coefficients according to the initial positions.

### 3.2.2 Globally scrambling sub-blocks except for DC coefficients

Step 1: Generate the keystream  $Z_4$  required for scrambling the subblocks,  $Z_4 = z_4(1:S)$ .

Step 2: Sort  $Z_4$  in ascending order and use the

sorted index value  $index_2$  to scramble the sub-blocks, except for the DC coefficient.

### 3.3 Decryption process

Decryption is the reverse process of encryption (the flowchart of decryption is provided in the Supplementary Material 7). The specific process is as follows:

Step 1: First, perform entropy decoding on the ciphertext JPEG image to obtain the ciphertext image in the frequency domain. The DC coefficient  $B_2$ , which is used to compute the initial key, is then extracted by  $H_1$  generated by  $key_1$ .

Step 2: According to homomorphism,  $Initx$  can be extracted from the ciphertext by performing  $\text{sum}(B_2(1:16)) \bmod L$ . The  $Initx$  can be extracted because the following equation is always established regardless of whether  $B_2$  is permuted or not:  $\text{sum}(B_2(1:16)) \bmod L = ((B_2(1:8)) \bmod L + (B_2(9:16)) \bmod L) \bmod L$ . According to section 3.1.1 and the Supplementary material 2, we have:  $\text{sum}(B_2(1:16)) \bmod L = (initial_1 + initial_2) \bmod L = Initx$ . Then, generate the pseudo-random sequence  $k$  using the pseudo-random generator based on  $key_1 + key_2$  and the initial value  $Initx$ , and perform inverse scrambling operations on  $B_2$  according to  $k$  to restore the scrambling.

Step 3: Perform homomorphic computation on the restored  $B_2$  to obtain the plaintext-related information  $initial_1$  and  $initial_2$ , and combine it with  $key_3$  and  $key_4$  to generate key streams  $Z_1$  and  $Z_2$ . Group the remaining encrypted DC coefficients  $Q_4$ , and then perform inverse diffusion according to the key stream to obtain the decrypted remaining DC coefficients  $Q$ .

Step 4: Then,  $B_2$  is decrypted using  $H_2$  generated by  $key_2$ . Recombine the decrypted DC coefficient  $B$  and the remaining DC coefficient  $Q$  to obtain all the DC coefficients  $D$ . Combining all DC coefficients  $D$  and keys  $key_5$  and  $key_6$  generates keystreams  $Z_3$  and  $Z_4$ . All sub-blocks except DC coefficients in the frequency domain image are inverse scrambled according to the key stream  $Z_4$ . Then, the AC coefficients of the same run length are inversely scrambled according to the key stream  $Z_3$ .

Step 5: Inverse quantization, inverse DCT transformation, and sub-block merging are performed on the decrypted DCT image to obtain the plaintext.

## 4 Experimental Results and Security Analysis

We took the grayscale images “Cameraman”, “Pepper”, “Barbara”, “Baboon” and the color image “Squirrel” with the size of  $256 \times 256$  as examples. Besides these five commonly used images, we tested the first 1,000  $512 \times 512$  images selected from the BOWS2 database. All algorithms were implemented by Matlab R2020b. The computer had an Intel(R) Core(TM) i7-8700 @ 3.20 GHz CPU, 8.00 GB memory, and Microsoft Windows 10. Images mainly with a quality factor of 85 were used to compare the performance of different schemes. We evaluated the proposed SPJEU in terms of image quality, JPEG format compatibility, file size change, and security analysis to verify the effectiveness, algorithm efficiency, additional data transmission key storage space, and superiority of the scheme. Note that our encryption scheme is also suitable for grayscale images and color images of different sizes since the color channels can be processed individually in the same manner as the grayscale images. Users can choose the number of DC coefficients as the initial key according to the size of the image and their own needs.

### 4.1 Visual Perception and Image Quality

In this section, we describe the evaluation of the visual perception and quality of the encrypted images. First, the five test images were encrypted and decrypted, with the results shown in Fig. 3. There was no observable connection between the encrypted image and the original image, indicating that our encryption scheme is visually secure. Additionally, we measured the quality of the encrypted images using mean square error (MSE), peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM) (Murillo-Escobar et al., 2019) to verify the effectiveness and security of the encryption scheme. The measured data are shown in Table 2. The quality factor was set to 85.

Table 2 Image quality

Image	SSIM	PSNR	MSE
Baboon	0.0008	9.05	8078
Cameraman	0.0009	7.99	10308
Pepper	0.0012	8.82	8536
Barbara	0.0003	8.35	9590
Squirrel	0.0013	9.02	9235

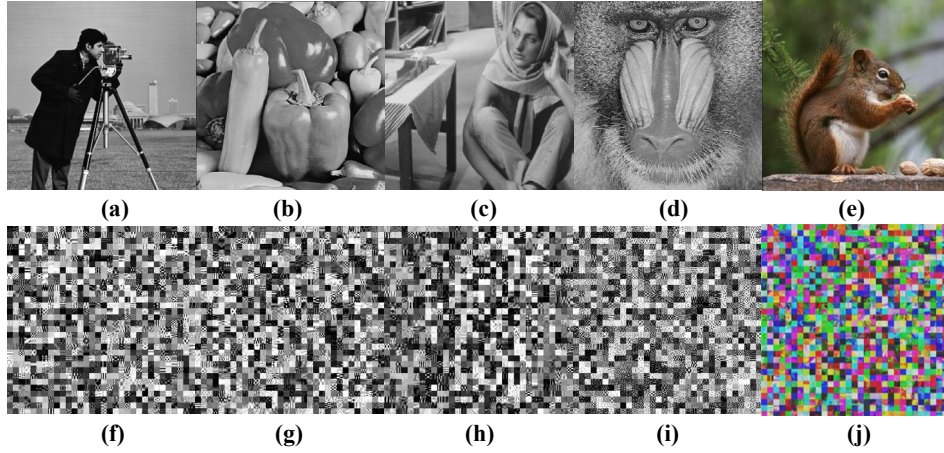


Fig. 3 Encryption results: (a)-(e) the original images, (f)-(g) the encrypted images.

## 4.2 JPEG format compatibility

Quantized DC coefficients of JPEG images have a fixed range, and these ranges vary under different quantization factors. Section 3.1 shows that the range of the quantized DC coefficients is  $[min, max]$ . If the encrypted DC coefficient exceeds this range, the DC coefficient will overflow, causing the JPEG format to be incompatible and impacting the visual effect of the image. For example, the scheme of Ong et al. (2013) adopts the method of RS pair scrambling, which can result in more than 63 AC coefficients in the sub-blocks of the JPEG image, making the image format incompatible with JPEG. In our scheme, the encryption method of the DC coefficient is based on modulo ( $L$ ). This method fixes the range of the DC coefficient at  $[min, max]$ , ensuring that the value of the encrypted DC coefficient will not overflow. The encryption of the AC coefficients adopts two scrambling methods: one for AC coefficients with the same run length and one for all the sub-blocks except the DC coefficient in the JPEG image. Neither method will cause the number of AC coefficients in a sub-block to exceed 63. Therefore, SPJEU has good compatibility with the JPEG format.

When designing a JPEG encryption scheme, the change in file size of the encrypted JPEG image should be considered. Table 3 shows the change in file size after encryption of different JPEG images under different quality factors (including 50, 60, 70, 80, 85, etc.). The JPEG encryption scheme we designed slightly increases the file size while ensuring security. However, under different quality factors, the rate of change in file size is controlled at 2%, which is neg-

ligible relative to the original data volume of the ciphertext.

Table 3 Change of file size under different quality factors

Image	Quality factors				
	50	60	70	80	85
Baboon	0.582%	0.610%	0.568%	1.486%	0.409%
Pepper	1.161%	1.045%	0.966%	0.826%	0.679%
Barbara	0.813%	0.665%	0.746%	0.603%	0.546%
Cam- eraman	1.101%	1.062%	1.239%	1.091%	0.897%
Squirrel	1.202%	1.068%	0.985%	0.793%	0.682%

We conducted comparative experiments with other algorithms in terms of the file size changing rate, which was computed by dividing the original file size by the difference between the encrypted file size and the original file size (Table 4). The experimental result represents the average value obtained from tests on the BOWS2 database. Relative to the schemes of Ong et al. (2015), Li and Lo (2018) and (2017), and Yuan et al. (2024), our file size changing rate was small. In comparison to Qin et al. (2022), Qian et al. (2014), He et al. (2018), and Hua et al. (2023), our file size changing rate is slightly larger. However, the method of Hua et al. (2023) has a changing rate close to 0 because the file size change stems only from byte padding. The schemes of Qian et al. (2014) and He et al. (2018) have lower changing rates because they are based on bitstream encryption, and there are not many elements in a JPEG bitstream that can be used for encryption, so the encryption effect is not good enough. The scheme of He et al. (2018) needs to iterate about 15 times to encrypt the DC algorithm to ensure image security, and the calculation cost is high.



The encryption algorithm of Qin et al. (2022), which has a negative file size change rate (the encrypted file is smaller than the original file), may lead to overflow of DC coefficients and AC coefficients, and it is necessary to embed the overflowed data into the ciphertext image to ensure the compatibility of the file format. Therefore, in view of its superiority in terms of self-sufficiency and plaintext-relation, the file size change rate of our scheme is still satisfactory. Compared with other encryption algorithms, we can ensure the security, efficiency and format compatibility of the encryption simultaneously without paying extra costs.

**Table 4 File change rate compared to other schemes**

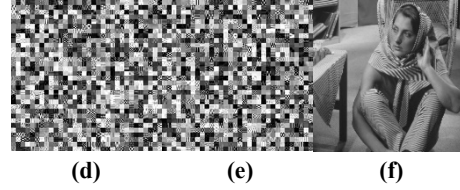
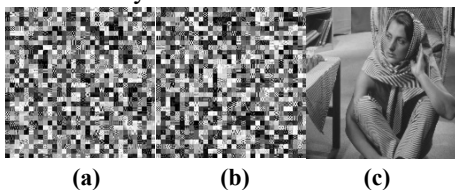
Schemes	Change rate of file size (%)
Ours	0.624
Li and Lo (2018)	3.752
He et al. (2018)	0.093
Li and Lo (2017)	7.208
Ong et al. (2015)+	1.632
Ong et al. (2015)-	1.381
Qian et al. (2014)	0.098
Qin et al. (2022)	-1.321
Hua et al. (2023)	0.076
Yuan et al. (2024)	1.149

### 4.3 Security Analysis

In addition to the security proof of the SPJEU model (see Supplementary Material 8), we evaluated the security of SPJEU from several aspects, including key sensitivity analysis, contour attack, histogram analysis, differential cryptanalysis, and Shannon entropy.

#### 4.3.1 Key sensitivity analysis

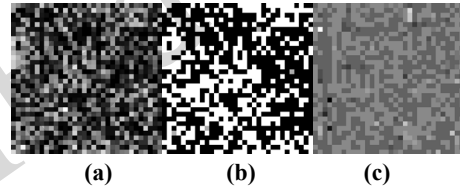
If the key changes slightly, the ciphertext image cannot be decrypted at all, indicating that the algorithm is very sensitive to the key. We used  $Key_1$  and  $Key_2$  to encrypt Barbara, where  $Key_2 = Key_1 + 10^{-14}$ . The results (Fig. 4) showed that our algorithm is very sensitive to the keys.



**Fig. 4 Evaluation of key sensitivity: (a) encrypted Barbara with  $Key_1$ , (b) decrypted (a) with  $Key_2$ , (c) decrypted (a) with  $Key_1$ , (d) encrypted Barbara with  $Key_2$ , (e) decrypted (d) with  $Key_1$ , (f) decrypted (d) with  $Key_2$ .**

#### 4.3.2 Outline attack

Fig. 5 shows the results obtained after NCC, EAC, and PLZ (Minemura et al., 2012) attacks performed on the ciphertext image of "Barbara". Clearly, the outline of the plaintext image cannot be identified, indicating that our proposed SPJEU can effectively resist these outline attacks and ensure the security of the encryption algorithm.



**Fig. 5 Results obtained by attacking ciphertext images using outline attacks. (a) NCC, (b) EAC, (c) PLZ.**

#### 4.3.3 Histogram analysis

Fig. 6 shows histograms of the original images of "Cameraman", "Pepper", "Barbara", "Baboon", and "Squirrel" with the size of  $256 \times 256$ , and their corresponding ciphertext images. Compared with the histogram of ordinary images, the ciphertext has a uniform pixel distribution, which means that our scheme is resistant to statistical attacks. The histogram bins at 0 and 255 in Fig. 6 are significantly higher, caused by the encryption of the AC coefficients and the DC coefficients, which makes it difficult to judge the range of the encrypted pixel values. We control values greater than 255 to 255 and values less than 0 to 0, but the pixel values in  $[0, 255]$  are still uniformly distributed, indicating that our proposed SPJEU can effectively resist statistical analysis.

#### 4.3.4 Differential cryptanalysis

Differential attack is an important analysis method to test the sensitivity of an algorithm to the plaintext, which can be measured by the pixel change

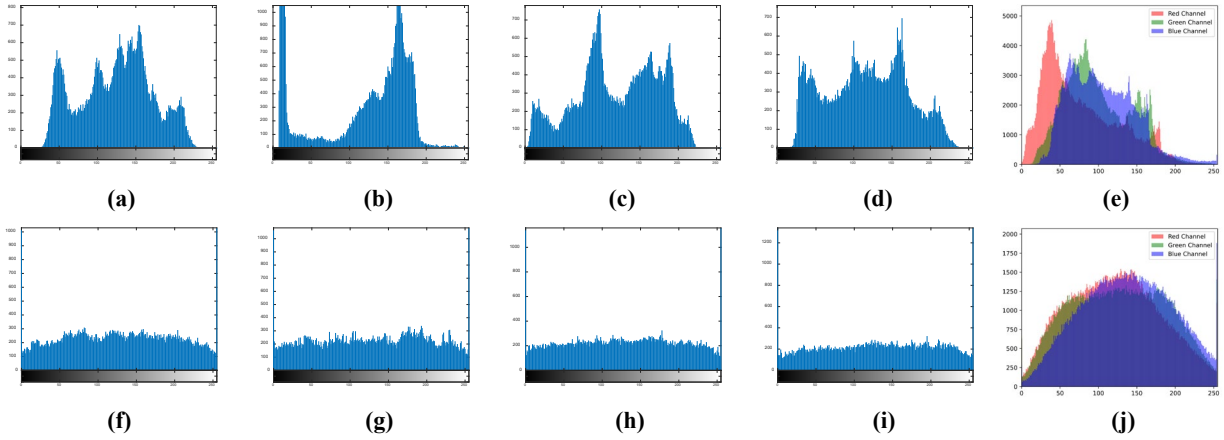


Fig. 6 Histogram analysis: (a)-(e) histograms of original images; (f)-(j) histograms of the corresponding encrypted images.

rate (NPCR) and the unified average change intensity (UACI) (Toughi et al., 2017). A higher NPCR/UACI represents a better resistance to differential attacks. Theoretically, the values of NPCR and UACI should be close to 99.61% and 33.46%, respectively (Zhang et al., 2020). Table 5 shows the NPCR values and UACI values of five images with a quality factor of 85. Our encryption scheme achieves NPCR values ranging from 99.31% to 99.57% and UACI values from 32.91% to 35.59% for the five test images with a quality factor of 85. These values are close to the theoretical expectations, indicating strong resistance to differential attacks.

Table 5 Values of NPCR and UACI for different images

Image	NPCR	UACI
Pepper	99.41%	34.27%
Cameraman	99.48%	33.62%
Barbara	99.31%	34.75%
Baboon	99.57%	32.91%
Squirrel	99.62%	32.90%

In addition, we conducted a comparative analysis of the NPCR and UACI values of our proposed scheme and several other algorithms. The data presented in Table 6 represent the average NPCR and UACI values calculated for each scheme based on the BOWS2 database. Table 6 shows that our scheme achieves an NPCR of 99.63% and a UACI of 35.68%, which is competitive compared with other schemes. In some cases, it is even superior to other schemes. For instance, while the scheme of Li and Lo (2018) has a lower NPCR of 96.12%, it exhibits a higher UACI of 38.34%. Conversely, that of He et al. (2018) achieves an NPCR of 99.37% but a lower UACI of

25.76%. The NPCR and UACI values of Li and Lo (2017) are not ideal because their DC coefficients are overflowed, causing pixels to exceed this range. The UACI value of Hua et al. (2023) is lower because their encryption is restricted. The indicators of the schemes of Yuan et al. (2024) and Qin et al. (2023) are more ideal and have higher security in terms of differential attack. Our scheme strikes a balance, providing robust differential attack resistance with NPCR and UACI values close to the theoretical optimum. The comparison indicates that our approach offers effective security measures while maintaining performance, as evidenced by consistently high NPCR and appropriate UACI values across different scenarios.

Table 6 Comparison of NPCR and UACI values

Scheme	NPCR	UACI
Ours	99.63%	35.68%
Li and Lo (2018)	96.12%	38.34%
He et al. (2018)	99.37%	25.76%
Li and Lo (2017)	99.38%	21.49%
Qin et al. (2022)	99.48%	35.26%
Hua et al. (2023)	99.36%	26.21%
Yuan et al. (2024)	99.68%	33.35%

#### 4.3.5 Information entropy

Information entropy can be used to measure the random distribution of gray values of ciphertext images. Table 7 lists the entropy values of the test images, where the quality factor is 85, and the calculated results are close to the ideal value of 8.

Additionally, we tested the average information entropy of encrypted images from the BOWS2 database using our encryption algorithm and other algo-

rithms, with the quality factor set to 85. Table 8 presents a comparison with the results of other algorithms. The results from Tables 7 and 8 demonstrate that our encryption algorithm achieves a high information entropy, close to the ideal value, indicating strong resistance to statistical attacks. Our scheme's average information entropy is 7.82, which is competitive with other schemes. These comparisons confirm that our algorithm provides robust encryption, ensuring high randomness and security in the ciphertext images.

**Table 7 Information entropy of different images**

Image	Plaintext entropy	Ciphertext entropy
Pepper	7.42	7.79
Cameraman	7.21	7.80
Barbara	7.56	7.78
Baboon	7.33	7.79
Squirrel	7.86	7.63

**Table 8 Comparison of average information entropy between different schemes**

Scheme	Average information entropy
Ours	7.82
Li and Lo (2018)	7.31
He et al. (2018)	7.79
Li and Lo (2017)	7.61
Qin et al. (2022)	7.85
Hua et al. (2023)	7.56
Yuan et al. (2024)	7.78

#### 4.4 Algorithm efficiency

Apart from security performance, encryption efficiency is important, especially in practical applications. The time complexity of the encryption algorithm is a crucial metric for evaluating its practicality. In our proposed encryption algorithm for an  $M \times N$  grayscale image, the time complexities for different operations are as follows. The time complexity is  $O(M \times N)$  for the block operation,  $O(M \times N \times \log(M \times N))$  for the DCT transformation, and  $O(M \times N)$  for the quantization operation. The time complexity is  $O(S)$  for generating the keystream to encrypt the quantized DC coefficients, generating random sequences  $k_1$  and  $k_2$  and processing them to get  $H_1$  and  $H_2$ ,  $O(I)$  for selecting the first 16 elements from  $H_1$  and extracting DC coefficients, and approximately  $O(I)$  for generating the initial key. For encrypting the quantized DC coefficients used for computing the initial key the time complexity is roughly  $O(I)$ , and for the grouped

encryption for the remaining DC coefficients, involving preprocessing, grouping, and diffusion, it is  $O(M \times N)$ . For the AC coefficients, the time complexity of scrambling the AC coefficients with the same run length is approximately  $O(M \times N)$ , and for the global scrambling of sub-blocks excluding the DC coefficients it is  $O(M \times N)$ . Overall, the total time complexity of the entire encryption process is dominated mainly by the image size  $M \times N$ , resulting in approximately  $O(M \times N \times \log(M \times N))$ . Table 9 compares the encryption runtime of different images with those of other encryption schemes. The encryption speed of our scheme compares well and is sufficient for practical applications.

#### 4.5 Additional data transmission and key storage space

In this section we compare the additional data transmission and key storage space of our scheme with those of other plaintext-related image encryption mechanisms. Additional data are plaintext-related information that needs to be transmitted in the public channel for image transmission. The key storage space is the space required to store decryption key data, including the secret key and additional data to be calculated using that key.

In most of the existing plaintext-related image encryption schemes, the additional data and the key storage space increased significantly with the rising number of images. This is because the plaintext-related information is separate from the image and should be transmitted alone in the communication channel. However, our proposed scheme fundamentally solves the problem of excessive plaintext-related data generated when encrypting massive images using existing plaintext-related image encryption systems. In our model, the plaintext-related information exists in the carrier image and can be extracted securely for decryption by legal users who have secret keys. Therefore, whether for a single image or a large number of images, our scheme maintains consistent and efficient performance in terms of additional data transmission and key storage space (Table 10). It is obvious that our scheme has superior performance when the number of images is large.

**Table 9 The encryption runtime of different schemes (s)**

Scheme	Pepper	Camerman	Baboon	Airplane	Squirrel	Average
Qin et al. (2022)	3.87	3.74	4.09	3.35	4.13	3.83
Ong et al. (2015)	3.67	3.85	3.98	3.24	3.86	3.72
Li and Lo (2018)	2.08	1.93	3.18	2.86	3.21	2.56
Li and Lo (2017)	2.98	2.86	3.24	2.74	3.32	3.03
He et al. (2018)	1.12	1.09	1.85	1.15	1.91	1.42
Hua et al. (2023)	0.76	0.68	1.10	0.84	1.21	0.92
Yuan et al. (2024)	1.73	1.68	1.95	1.72	1.88	1.79
Ours	1.89	1.96	1.87	1.61	1.90	1.85

**Table 10 Comparison of additional data and key storage space (in bits)**

Numbers of images	1	10	100	1000	10000	
Ours	Secret key	48	48	48	48	48
	Additional data	0	0	0	0	0
	Key storage space	48	48	48	48	48
He et al. (2018)	Secret key	32	32	32	32	32
	Additional data	512	5120	51200	512000	5120000
	Key storage space	544	5152	51232	512032	5120032
Li and Lo (2018)	Secret key	32	32	32	32	32
	Additional data	256	2560	25600	256000	2560000
	Key storage space	464	2592	25632	256032	2560032
Ong et al. (2015)	Secret key	40	40	40	40	40
	Additional data	512	5120	51200	512000	5120000
	Key storage space	552	5160	51240	512040	5120040
Qin et al. (2022)	Secret key	64	64	64	64	64
	Additional data	256	2560	25600	256000	2560000
	Key storage space	320	2624	25664	256064	2560064

**Table 11 Comprehensive performance comparison**

Metrics	Qin et al. (2022)	Li and Lo (2018)	Li and Lo (2017)	He et al. (2018)	Ong et al. (2015)	Hua et al. (2023)	Yuan et al. (2024)	Ours
File size preservation	***	*	*	***	**	***	**	**
Format Compatibility	**	***	**	***	-	***	***	***
Encryption runtime	*	**	**	***	*	***	***	**
Security	***	**	**	**	*	*	***	***
Related to plaintext	Yes	No	No	Yes	No	No	Yes	Yes
Extra transmission	Yes	No	No	Yes	No	No	Yes	No
Extra cost	Yes	No	No	Yes	No	No	Yes	No

#### 4.6 Comprehensive comparison of performance with other schemes

To compare the existing JPEG encryption schemes with ours comprehensively, we summarize the characteristics of each scheme, including file size preservation, format compatibility, encryption runtime, security, whether it is related to plaintext, and whether it requires additional transmission and extra cost. Table 11 shows that our algorithm is good enough based on its performance across the range of characteristics.

In Table 11, \* means average performance, \*\* means better performance, \*\*\* means excellent, - means no such performance. The extra transmission

refers to the schemes of Qin et al. (2023) and He et al. (2018) in which the hash value associated with the plaintext needs to be transmitted. The extra cost refers to the extra cost incurred to maintain the size, format compatibility, and security of the ciphertext file. The scheme of Qin et al. (2023) needs to embed the overflowed data into the ciphertext, and that of He et al. (2018) needs at least 15 iterations to guarantee visual security.

## 5 Conclusions

In this paper we propose SPJEU to solve the problems of additional transmission of information related to the plaintext and difficult management of a large number of different decryption keys in the tra-

ditional plaintext-related image mechanisms. Our proposed scheme first selects the quantized DC coefficients in the DCT image through a unified key and then generates a key stream associated with the plaintext according to the selected DC coefficients. Therefore, different images can be encrypted with different keystreams, which can resist chosen plaintext attacks effectively. Since only a unified key needs to be sent in the communication channel, the transmission cost is low and the potential threat of plaintext information leakage is eliminated. Since the relationship between the unified key and the image can be re-established easily for decryption, the receiver needs only to store the unified key, rather than different key information corresponding to different images, and key management is simple. In addition, experiments and analysis show that SPJEU ensures the security and format compatibility of the ciphertext image and has little effect on the size of the ciphertext file.

To sum up, the main contribution of this paper is the design of a JPEG image encryption algorithm associated with plaintext that fundamentally solves the problem of too much plaintext-related data generated by encrypting massive images using the existing plaintext-related image encryption systems. Since it is relatively simple to establish different decryption keys for different images using SPJEU, transmission costs and key storage space can be saved and decryption is easy. In the future, we aim to design an encryption algorithm for JPEG images that has less or no effect on the size of the ciphertext file.

#### Contributors (refer to <https://www.casrai.org/credit.html>)

Ming LI designed the research. Wenwen ZHOU and Mengdie WANG processed the data and drafted the paper. Yushu ZHANG and Yong XIANG helped organize the paper. Ming LI and Wenwen ZHOU revised and finalized the paper.

#### Conflict of interest

Ming LI, Wenwen ZHOU, Mengdie WANG, Yushu ZHANG and Yong XIANG declare that they have no conflict of interest.

#### Data availability

The online version of this article (<https://doi.org/10.1631/FITEE.2400721>) contains supplementary materials, which are available to authorized users.

#### References

- Haider T, Blanco SA, Hayat U, 2023. A novel pseudo-random number generator based on multi-objective optimization for image-cryptographic applications. <https://arxiv.org/abs/2307.03911>
- He JH, Huang SH, Tang SH, et al., 2018. JPEG image encryption with improved format compatibility and file size preservation. *IEEE Trans Multim*, 20(10):2645-2658. <https://doi.org/10.1109/TMM.2018.2817065>
- He JH, Chen JX, Luo WQ, et al., 2019. A novel high-capacity reversible data hiding scheme for encrypted JPEG bitstreams. *IEEE Trans Circuits Syst Video Technol*, 29(12):3501-3515. <https://doi.org/10.1109/TCSVT.2018.2882850>
- He K, Bidan C, Le Guelvouit G, et al., 2016. Robust and secure image encryption schemes during JPEG compression process. Proc IS&T Int Symposium on Electronic Imaging, p.1-7. <https://doi.org/10.2352/ISSN.2470-1173.2016.11.IMAW-M-461>
- Hua ZY, Wang ZY, Zheng YF, et al., 2023. Enabling large-capacity reversible data hiding over encrypted JPEG bitstreams. *IEEE Trans Circuits Syst Video Technol*, 33(3):1003-1018. <https://doi.org/10.1109/TCSVT.2022.3208030>
- Li M, Wang MD, Fan HJ, et al., 2022. A novel plaintext-related chaotic image encryption scheme with no additional plaintext information. *Chaos, Solitons Fractals*, 158:111989. <https://doi.org/10.1016/j.chaos.2022.111989>
- Li PY, Lo KT, 2017. Joint image compression and encryption based on order-8 alternating transforms. *J Vis Commun Image Represent*, 44:61-71. <https://doi.org/10.1016/j.jvcir.2017.01.021>
- Li PY, Lo KT, 2018. A content-adaptive joint image compression and encryption scheme. *IEEE Trans Multim*, 20(8):1960-1972. <https://doi.org/10.1109/TMM.2017.2786860>
- Li PY, Lo KT, 2019. Joint image encryption and compression schemes based on 16×16 DCT. *J Vis Commun Image Represent*, 58:12-24. <https://doi.org/10.1016/j.jvcir.2018.11.018>
- Li PY, Lo KT, 2020. Survey on JPEG compatible joint image compression and encryption algorithms. *IET Signal Process*, 14(8):475-488. <https://doi.org/10.1049/iet-spr.2019.0276>
- Li SS, Zhang YY, 2016. Quantized DCT coefficient category address encryption for JPEG image. *KSII Trans Internet Inf Syst*, 10(4):1790-1806. <https://doi.org/10.3837/tiis.2016.04.018>
- Li SS, Ma RL, Zhang HL, 2019. Enhancing security for JPEG image against mosaic attack using inter-block shuffle encryption. *IEEE Access*, 7:72696-72702. <https://doi.org/10.1109/ACCESS.2019.2918860>
- Li WH, Yuan Y, 2007. A leak and its remedy in JPEG image encryption. *Int J Comput Math*, 84(9):1367-1378. <https://doi.org/10.1080/00207160701294376>

- Liu H, Wan HB, Tse CK, et al., 2016. An encryption scheme based on synchronization of two-layered complex dynamical networks. *IEEE Trans Circuits Syst I: Regul Pap*, 63(11):2010-2021.  
<https://doi.org/10.1109/TCSI.2016.2598822>
- Mohammadi A, Nakhkash M, Akhaee MA, 2020. A high-capacity reversible data hiding in encrypted images employing local difference predictor. *IEEE Trans Circuits Syst Video Technol*, 30(8):2366-2376.  
<https://doi.org/10.1109/TCSVT.2020.2990952>
- Minemura K, Moayed Z, Wong K, et al., 2012. JPEG image scrambling without expansion in bitstream size. Proc 19<sup>th</sup> IEEE Int Conf on Image Processing, p. 261-264.  
<https://doi.org/10.1109/ICIP.2012.6466845>
- Murillo-Escobar MA, Meranza-Castillón MO, López-Gutiérrez RM, et al., 2019. Suggested integral analysis for chaos-based image cryptosystems. *Entropy*, 21(8):815.  
<https://doi.org/10.3390/e21080815>
- Ong SY, Minemura K, Wong KS, 2013. Progressive quality degradation in JPEG compressed image using DC block orientation with rewritable data embedding functionality. Proc IEEE Int Conf on Image Processing, p. 4574-4578.  
<https://doi.org/10.1109/ICIP.2013.6738942>
- Ong SY, Wong K, Qi XJ, et al., 2015. Beyond format-compliant encryption for JPEG image. *Signal Process: Image Commun*, 31:47-60.  
<https://doi.org/10.1016/j.image.2014.11.008>
- Qian ZX, Zhang XP, Wang SZ, 2014. Reversible data hiding in encrypted JPEG bitstream. *IEEE Trans Multimedia*, 16(5):1486-1491.  
<https://doi.org/10.1109/TMM.2014.2316154>
- Qian ZX, Zhou H, Zhang XP, et al., 2018. Separable reversible data hiding in encrypted JPEG bitstreams. *IEEE Trans Dependable Secure Comput*, 15(6):1055 – 1067.  
<https://doi.org/10.1109/TDSC.2016.2634161>
- Qian ZX, Xu HS, Luo XY, et al., 2019. New Framework of Reversible Data Hiding in Encrypted JPEG Bitstreams. *IEEE Trans Circuits Syst Video Technol*, 29(2):351-362.  
<https://doi.org/10.1109/TCSVT.2018.2797897>
- Qin C, Hu JC, Li FY, et al., 2023. JPEG image encryption with adaptive DC coefficient prediction and RS pair permutation. *IEEE Trans Multimedia*, 25:2528-2542.  
<https://doi.org/10.1109/TMM.2022.3148591>
- Qu LF, He HJ, Chen F, 2022. On the security of block permutation and Co-XOR in reversible data hiding. *IEEE Trans Circuits Syst Video Technol*, 32(3):920-932.  
<https://doi.org/10.1109/TCSVT.2021.3069811>
- Ramesh A, Jain A, 2015. Hybrid image encryption using Pseudo Random Number Generators, and transposition and substitution techniques. Proc Int Conf Trends in Automation, Communications and Computing Technology, p. 1-6.  
<https://doi.org/10.1109/ITACT.2015.7492652>
- Refregier P, Javidi B, 1995. Optical image encryption based on input plane and Fourier plane random encoding. *Opt Lett*, 20(7):767-769.  
<https://doi.org/10.1364/OL.20.000767>
- Ting J, Wong K, Ong S, 2019. Format-compliant perceptual encryption method for JPEG XT. Proc IEEE Int Conf on Image Processing, p. 4559-4563.  
<https://doi.org/10.1109/ICIP.2019.8803623>
- Toughi S, Fathi MH, Sekhavat YA, 2017. An image encryption scheme based on elliptic curve pseudo random and advanced encryption system. *Signal Process*, 141:217-227.  
<https://doi.org/10.1016/j.sigpro.2017.06.010>
- Xian YJ, Wang XY, Wang XY, et al., 2022. Spiral-transform-based fractal sorting matrix for chaotic image encryption. *IEEE Trans Circuits Syst I: Regul Pap*, 69(8):3320-3327.  
<https://doi.org/10.1109/TCSI.2022.3172116>
- Xie D, Chen FL, Luo YL, et al., 2019. One-to-many image encryption with privacy-preserving homomorphic outsourced decryption based on compressed sensing. *Digital Signal Process*, 95:102587.  
<https://doi.org/10.1016/j.dsp.2019.102587>
- Yi S, Zhou YC, Hua ZY, 2018. Reversible data hiding in encrypted images using adaptive block-level prediction-error expansion. *Signal Process: Image Commun*, 64:78-88.  
<https://doi.org/10.1016/j.image.2018.03.001>
- Yuan Y, He HJ, Chen F, et al., 2024. On the security of JPEG image encryption with RS pairs permutation. *J Inf Secur Appl*, 82:103722,  
<https://doi.org/10.1016/j.jisa.2024.103722>
- Zhang Y, Chen AG, Tang YJ, et al., 2020. Plaintext-related image encryption algorithm based on perceptron-like network. *Inf Sci*, 526:180-202.  
<https://doi.org/10.1016/j.ins.2020.03.054>
- Zheng WB, Wang FY, Wang KF, 2017. An ACP-based approach to color image encryption using DNA sequence operation and hyper-chaotic system. Proc IEEE Int Conf Systems, Man, and Cybernetics, p. 461-466.  
<https://doi.org/10.1109/SMC.2017.8122648>

### List of electronic supplementary materials

- 1 Three image encryption schemes
  - 2 Generation of key streams for DC and AC coefficients
  - 3 Example of DC coefficient encryption
  - 4 Proof of additively homomorphic
  - 5 Example of AC coefficient encryption
  - 6 Example of AC coefficient run-length distribution
  - 7 Flowchart of JPEG decryption
  - 8 Security proof of SPJEU model
- Fig. S1 Comparison of three encryption mechanisms  
 Fig. S2 Example of DC coefficient encryption  
 Fig. S3 Example of AC coefficient encryption  
 Fig. S4 AC coefficient run-length distribution of gray-scale image Barbara (256×256)  
 Fig. S5 Flowchart of JPEG decryption

Table S1 The number of times required for brute force cracking

unedited