

Received November 30, 2018, accepted December 19, 2018, date of publication January 1, 2019, date of current version January 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2890116

# Image Encryption Using Josephus Problem and Filtering Diffusion

ZHONGYUN HUA<sup>ID</sup>, (Member, IEEE), BINXUAN XU, FAN JIN,  
AND HEJIAO HUANG<sup>ID</sup>, (Member, IEEE)

School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China

Corresponding author: Hejiao Huang (huanghejiao@hit.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0800804 and Grant 2017YFB0803002, in part by the National Natural Science Foundation of China under Grant 61701137, and in part by the Shenzhen Science and Technology Plan under Grant JCYJ20170307150704051 and Grant JCYJ20170811160212033.

**ABSTRACT** Image encryption is an efficient visual technology to protect private images. This paper develops an image encryption algorithm utilizing the principles of the Josephus problem and the filtering technology. The encryption algorithm follows the classical diffusion and confusion structure. The principle of Josephus problem is used to shuffle the image pixels to different positions to achieve the confusion property. Using a randomly generated filter, the filtering technology can spread slight changes of the original image to all pixels of the cipher image to obtain diffusion property. The simulation results show that the developed image encryption algorithm is able to encrypt different kinds of images into cipher images with uniform distribution. The security analysis demonstrates that it has an extremely sensitive secret key, can resist various security attacks, and has a better performance than several advanced image encryption algorithms.

**INDEX TERMS** Cryptography, image encryption, image filtering, multimedia security, security analysis.

## I. INTRODUCTION

In recent decades, an increasing number of digital data are generated and transmitted in all kinds of networks. Among all these digital data, the digital image is one most commonly used data format, because a digital image carries information using a straightforward way so that it contains much potential information. For example, a man's photo may not only tell how he looks like, but also contain the information about his age, health conditions, his location, and so on. As a result, with the same data capability, a digital image may contain much more information than many other data formats such as text data. Thus, it is very important to protect the secret digital images [2]–[4]. Recently, researchers have developed many image security technologies such as information hiding [5], digital watermarking [6] and image encryption [7]–[10]. Among these technologies of protecting image, the image encryption is a visual and efficient technology that encrypts a meaningful image to be an unrecognized cipher-image [11]–[13]. Only using the correct key, one can recover the original image.

One practicable image encryption method is to treat a digital image as a binary data sequence, and then uses the

traditional data encryption technologies (e.g. Advanced Encryption Standard [14]) to encrypt the data sequence. However, each pixel in a digital image is usually presented using 8 or even more bits so that high information redundancy may exist between adjacent pixels. Encrypting an image as a data sequence doesn't consider the property of image pixel, and thus may result in low encryption efficiency. As a consequence, considering the image properties, many image encryption algorithms have been developed using various technologies such as chaos theory [15]–[18], quantum theory [19], [20], compressive sensing [21], [22] and DNA coding [23]–[25]. Because chaos has many inner properties such as initial condition sensitivity, unpredictability and ergodicity, and these properties are similar to the principles of image encryption [26], chaotic systems are widely used for image encryption. For example, Li *et al.* [27] proposed a novel encryption scheme based on chaotic tent map, which can generate chaotic key stream to encrypt images. Liu *et al.* [28] proposed a new 2D-SIMM map using close-loop modulation coupling model, and then further designed an image encryption algorithm using 2D-SIMM. In [29], a new two-point diffusion strategy based on Hénon map is proposed

and it can significantly accelerate the diffusion process. The authors in [30] developed a mixed chaotic system for image encryption. The chaotic system can generate new chaotic maps according to parametric switching.

However, when chaotic systems are implemented in finite precision platforms, the chaos degradation will happen and the chaotic behaviors may be degraded to be periodic behaviors [31]. If the chaos of a chaotic system is lost, its related image encryption algorithms will have weak security performance [32], [33]. Recently, many reports have shown that image encryption algorithms using chaos can be successfully broken [34]–[36]. Besides the chaos theory, many other technologies are also widely used in designing image encryption algorithm. Wu *et al.* [37] proposed a new Latin square image encryption algorithm that embeds random noise into the least significant bit-plane of images. Enayatifar *et al.* [38] proposed an image encryption algorithms that first converts the two-dimensional plain-image to one-dimensional pixel sequence, and then does permutation and diffusion operations to the one-dimensional pixel sequence in the same time. Wang *et al.* [39] developed a fast image algorithm and it can simultaneously encrypt the pixels from a row or a column. Besides, some bit-level image encryption algorithms have also been proposed [40]–[42]. These encryption algorithms encrypt images in the bit level instead of the pixel level.

In this paper, we propose an image encryption scheme utilizing Josephus problem and filtering diffusion (IES-JPDF). A Josephus scrambling is developed using the Josephus problem [1] and it can randomly permute image pixels to various columns and rows to obtain the confusion property. The filtering technology has wide applications in image processing such as image smoothing and image denoising. However, using an irregular filter, the image filtering can randomly change pixel values and spread little change of a pixel to all its adjacent pixels. The secret key of IES-JPDF is used to generate the variables of the Josephus scrambling and the filters of the filtering diffusion. Two rounds of the Josephus scrambling and filtering diffusion can ensure a high security level of the encrypted results. The simulation results show that IES-JPDF is able to encrypt a meaningful plain-image to be an unrecognized cipher-image with random distribution. The security analysis is performed in terms of the secret key analysis, ability of defending differential attack, adjacent pixel correlation, local Shannon entropy and deviation from uniform histogram. The analysis results demonstrate that IES-JPDF can obtain a high security level and can achieve better performance than several image encryption algorithms.

The remainder of this paper is organized as follows. Section II introduces the Josephus problem and image filtering, and proves their ability in image encryption. Section III presents the proposed IES-JPDF and Section IV simulates IES-JPDF in the MATLAB environment. Section V analyzes the security performance of IES-JPDF using different measures and Section VI concludes this work.

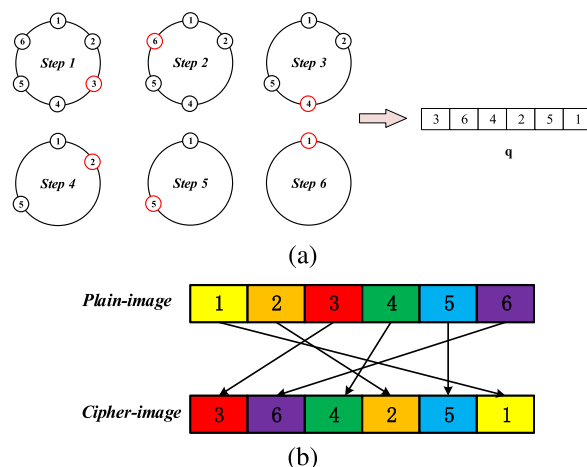


FIGURE 1. An example of Josephus sequence and its usage in image permutation. (a) Generation of Josephus sequence; (b) Image permutation using the Josephus sequence.

## II. PRELIMINARIES

This section introduces the Josephus problem and the image filtering, and illuminates their availability in image encryption.

### A. JOSEPHUS PROBLEM

The Josephus problem is an interesting ancient mathematics problem [1]. In the Roman-Jewish War, Josephus and his companions were trapped in a cave. To better survive, they decided to stand in a circle and kill every third man until the last one. To avoid being killed, Josephus correctly counted the position and became the last survivor. Generally, we can describe the Josephus problem like this. For a certain number of elements being placed in a circle, one starts from a specified element and remove it from the circle, and then circularly shift a fixed number of elements to figure out the current element. The next iteration starts from the current one. Repeat this action until the last element. Finally, according to the order of elements to be removed from the circle, we can obtain a sequence, namely Josephus sequence. Mathematically, the Josephus sequence can be described as

$$\mathbf{q} = JS(n, s, k), \tag{1}$$

where  $n$  is the total number of elements,  $s$  represents the starting position,  $k$  is the shifting number and  $\mathbf{q}$  is the produced Josephus sequence.

To better explain the Josephus sequence, we give a numerical example with  $n = 6$ ,  $s = 3$  and  $k = 3$  in Eq. (1). Fig. 1(a) shows the generation procedure of the Josephus sequence. As the starting position is 3, then we remove the element numbered 3 and place it in  $\mathbf{q}$ , namely  $\mathbf{q}_1 = 3$ . Next, circularly shift 3 elements and the current one is the element numbered 6, then we remove the element and place the number 6 in  $\mathbf{q}$ , namely  $\mathbf{q}_2 = 6$ . Repeating this operation until the last element and we obtain the final Josephus sequence  $\mathbf{q} = \{3, 6, 4, 2, 5, 1\}$ . The Josephus sequence has

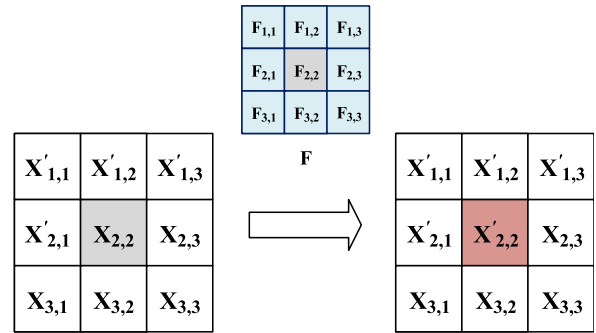
the properties of ergodicity and invertibility, and these properties are quite similar to the concept of the permutation in image encryption. Thus, the Josephus sequence can be used to design image permutation algorithm. In the permutation, the secret key is to generate the parameters in Eq. (1), and the generated Josephus sequence is used to scramble images pixels. Fig. 1(b) demonstrates an example of the image permutation using the Josephus sequence generated in Fig. 1(a), namely  $\mathbf{q} = \{3, 6, 4, 2, 5, 1\}$ . We first number the pixels of plain-image with  $1, 2, \dots, 6$ , and then rearrange the pixels to different positions according to  $\mathbf{q}$ .

Recently, researchers have developed some pixel permutation schemes using the concept of the Josephus problem [42]–[45]. These schemes perform the image pixel permutation in one-dimensional (1D) sequences, as demonstrated in Fig. 1(b). For example, Guo *et al.* [42] and Wang *et al.* [43] used 1D Josephus sequences to shuffle image pixels and can only shuffle the row or column positions of image pixels in one time shuffling. As a digital image is a two-dimensional (2D) data matrix, encrypting it as a 1D sequence doesn't consider the property of 2D matrix, which may result in low encryption efficiency. Besides, these previous works directly use the concept of Josephus traveling to shuffle image pixels and this may not obtain good shuffling effect [42], [43]. To address these weaknesses, we extend the Josephus permutation from 1D to 2D. Using a 2D Josephus sequence to shuffle the image pixels, the row and column positions of a pixel can be simultaneously shuffled. Thus, encrypting images as 2D matrices can achieve much higher efficiency than encrypting them as 1D sequences. The detailed 2D Josephus permutation will be presented in Section III-A.

**B. IMAGE FILTERING TECHNOLOGY**

The image filtering is a kind of image processing technology that applies convolution to an image block using a filter. A filter is a small 1D or 2D matrix that applies to each image pixel and its neighbor pixels. The center element of the filter is usually aligned with the current pixel and other elements correspond to the neighbor pixels. The filtering result is the additions of all the multiplications of the filter elements and image pixels. The image filtering technology has wide applications in many aspects such as edge detection, image restoration, and feature extraction. Fig. 2 demonstrates the image filtering operation. One can see that  $\mathbf{X}$  is an image block,  $X_{2,2}$  is the pixel to be processed, and  $\mathbf{F}$  is a filter. The center element of  $\mathbf{F}$  is multiplied with  $X_{2,2}$  and other elements are multiplied with the neighbor pixels. The sum of all the products is the filtering result. Specifically, assume that the filter  $\mathbf{F}$  has the size  $(2a + 1) \times (2b + 1)$ , the image filtering for each pixel of  $\mathbf{X}$  is calculated as

$$\mathbf{X}'(x, y) = \sum_{i=1}^{2a+1} \sum_{j=1}^{2b+1} \mathbf{F}(i, j)\mathbf{X}(x + i - a - 1, y + j - b - 1). \tag{2}$$



**FIGURE 2. Demonstration of convolution operation.**

Utilizing an appropriate filter, the image filtering can remove the noise of an image. However, using an improper or irregular filter, the image filtering can also blur an image. As a consequence, we can use the filtering technology to achieve the diffusion property in image encryption. The diffusion property indicates that tiny changes in original image can affect all pixels of the encrypted result.

All the traditional usage of image filtering don't need to restore the original pixel values. However, when filtering technology is used in image encryption, it must be reversible, as image encryption needs to recover the original image. This issue can be addressed by setting the center element of the filter as one and setting other elements as integers, namely  $\mathbf{F}(a+1, b+1) = 1$  and other elements in  $\mathbf{F}$  are integers. When doing image filtering to the image pixel  $\mathbf{I}(x, y)$ , a matrix  $\mathbf{T}$  with size  $(2a + 1) \times (2b + 1)$  is obtained as follows,

$$\mathbf{T} = \begin{pmatrix} \mathbf{I}'(x - a, y - b) \cdots \mathbf{I}'(x - a, y) \cdots \mathbf{I}'(x - a, y + b) \\ \vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ \mathbf{I}'(x, y - b) \quad \cdots \quad \mathbf{I}(x, y) \quad \cdots \quad \mathbf{I}(x, y + b) \\ \vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ \mathbf{I}(x + a, y - b) \cdots \mathbf{I}(x + a, y) \cdots \mathbf{I}(x + a, y + b) \end{pmatrix} \tag{3}$$

and then the image pixel  $\mathbf{I}(x, y)$  can be filtered using Eq. (4), as shown at the bottom of the next page, where  $F$  is the grayscale level of  $\mathbf{I}$ . As can be seen from the matrix  $\mathbf{T}$ , the used left and upper adjacent pixels have been processed, while the used right and lower adjacent pixels are the original pixels that haven't been processed. When doing the reverse operation to the current pixel, the elements of matrix  $\mathbf{T}$  are in the same states with the forward process. Thus, Eq. (4) is reversible and its reverse operation can be obtained as Eq. (5), as shown at the bottom of the next page. Using the Eqs. (4) and (5), we can design image encryption algorithm.

For many existing diffusion techniques, they usually use one or two previous pixels from the same row or column to change the current pixel. However, the filtering technology uses the surrounding pixels from different rows and columns to change the current pixel. Thus, the filtering technology can faster spread slight change of one pixel to all the pixels and can show better performance in defending the differential

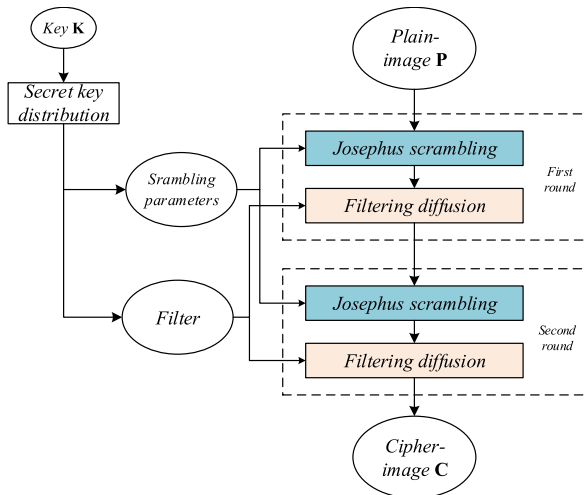


FIGURE 3. The structure of the proposed IES-JPFD.

attack, compare with many other diffusion techniques. This will be verified by the experiment results in Section V-B.

### III. NEW IMAGE ENCRYPTION ALGORITHM

According to the principles of the Josephus problem and the filtering diffusion, this section develops an image encryption scheme called IES-JPFD. The IES-JPFD follows the classical confusion and diffusion structure. This structure consists of two parts. One is used to achieve the diffusion property and the other is used to achieve the confusion property. As the Josephus scrambling can simultaneously shuffle the pixel row and column positions to obtain the confusion property, and the filtering technology can achieve good diffusion property, they are combined to design IES-JPFD. Fig. 3 shows the structure of IES-JPFD. As can be observed, the secret key is to obtain the scrambling parameters of the Josephus scrambling and obtain the coefficients of the filter for filtering diffusion. Two rounds of scrambling and filtering diffusion operations can ensure that a meaningful plain-image can be encrypted to be an unrecognized cipher-image with a high security level. The decryption process is the reverse operation of each step

TABLE 1. Descriptions of some important notations.

Notation	Description
$k$	the secret key with length of 256 bits
$k^{(1)}$	the sub-key of the first round of image encryption
$k^{(2)}$	the sub-key of the second round of image encryption
$g$	the sub-key of 24 bits used in Josephus scrambling
$d$	the sub-key of 96 bits used in filtering diffusion
$O$	the position index matrix for pixel permutation
$ri$	the Josephus sequence used to produce $O$
$ci$	the Josephus sequence used to produce $O$
$F$	the filter
$e$	the integer vector converted from $d$
$v$	the index vector generated by sorting $e$

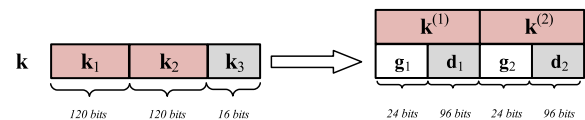


FIGURE 4. The structure of the secret key.

in IES-JPFD. To make the presentation of IES-JPFD more clear, we describe some important notations in Table 1.

The length of secret key  $k$  is set as 256 bits and it includes three parts  $k_1$ ,  $k_2$  and  $k_3$ , which can be seen in the left of Fig. 4. The  $k_1$  and  $k_2$  are 120 bits length and the perturbation parameter  $k_3$  is of size 16 bits. Using  $k_1$ ,  $k_2$  and  $k_3$ , we can obtain two sub-key  $k^{(1)}$  and  $k^{(2)}$  for two encryption rounds. Suppose  $t$  is the decimal integer converted from  $k_3$  and  $t' = t \bmod 120$ , then the sub-key  $k^{(1)}$  can be obtained as follows: 1) cyclically shift  $k_2$   $t'$  bits by right direction to obtain  $k'_2$ ; 2)  $k^{(1)} = k_1 \oplus k'_2 \oplus k_3$ , where  $\oplus$  represents the XOR operation. The sub-key  $k^{(2)}$  can be obtained as follows: 1) cyclically shift  $k_1$   $t'$  bits by right direction to obtain  $k'_1$ ; 2)  $k^{(2)} = k'_1 \oplus k_2 \oplus k_3$ . Each sub-key has two parts,  $g$  and  $d$ , which can be seen in the right of Fig. 4. The component  $g$  is to obtain the parameters for the Josephus scrambling and  $d$  is to produce the coefficients of filter for the filtering diffusion.

$$\begin{aligned}
 \mathbf{I}'(x, y) &= \left( \sum_{i=1}^{2a+1} \sum_{j=1}^{2b+1} \mathbf{F}(i, j) \mathbf{T}(i, j) \right) \bmod F \\
 &= \left( \mathbf{F}(a+1, b+1) \mathbf{T}(a+1, b+1) + \sum_{(i,j) \in [1, 2a+1] \times [1, 2b+1] \cap (i,j) \neq (a+1, b+1)} \mathbf{F}(i, j) \mathbf{T}(i, j) \right) \bmod F \\
 &= \left( \mathbf{I}(x, y) + \sum_{(i,j) \in [1, 2a+1] \times [1, 2b+1] \cap (i,j) \neq (a+1, b+1)} \mathbf{F}(i, j) \mathbf{T}(i, j) \right) \bmod F \tag{4}
 \end{aligned}$$

$$\mathbf{I}(x, y) = \left( \mathbf{I}'(x, y) - \sum_{(i,j) \in [1, 2a+1] \times [1, 2b+1] \cap (i,j) \neq (a+1, b+1)} \mathbf{F}(i, j) \mathbf{T}(i, j) \right) \bmod F \tag{5}$$

## A. JOSEPHUS SCRAMBLING

An image encryption algorithm should decorrelate the high correlations of image. As discussed in Section II-A that the Josephus sequence can randomly change the positions of a 1D array. However, a digital image is a 2D matrix. Considering the properties of 2D matrix, we design the 2D Josephus scrambling that can shuffle the row and column positions simultaneously. First, generate a series of parameters from the sub-key. Then, obtain several Josephus sequences using the parameters to perform row and column permutations.

### 1) PARAMETER GENERATION

Suppose the plain-image to be encrypted has the size of  $M \times N$ , four parameters are generated from the component  $\mathbf{g}$  of a sub-key,

$$\begin{aligned} MP &= (bi2de(\mathbf{g}(1 : 8)) \bmod M) + 1 \\ NP &= (bi2de(\mathbf{g}(9 : 16)) \bmod N) + 1 \\ MStep &= bi2de(\mathbf{g}(17 : 20)) + 1 \\ NStep &= bi2de(\mathbf{g}(21 : 24)) + 1 \end{aligned} \quad (6)$$

where  $bi2de(\cdot)$  is a function to convert a binary sequence to a decimal number,  $MP$  denotes the starting row position,  $MStep$  shows the row shifting step,  $NP$  indicates the initial column position, and  $NStep$  is the column shifting step.

### 2) SCRAMBLING USING PARAMETERS

Using the parameters generated from the sub-key, we can design the Josephus scrambling. For the image of size  $M \times N$ , the whole processing of the Josephus scrambling are as follows,

*Step 1:* Produce a Josephus sequence  $\mathbf{ri}$  of  $M$  length as follows: 1) initialize a vector  $\mathbf{a} = 1 : M$ ; 2) set  $\mathbf{ri}(1) = MP$  and remove  $\mathbf{ri}(1)$  from  $\mathbf{a}$ ; 3) start from the previously removed position, circularly shift  $MStep$  cells in  $\mathbf{a}$  and allocate the next value to  $\mathbf{ri}(2)$ ; 4)  $MStep = MStep + 1$ ; 5) repeat the operations 3) and 4) till all the  $M$  values in  $\mathbf{a}$  have been allocated to  $\mathbf{ri}$ ;

*Step 2:* Set row index  $i = 1$ ;

*Step 3:* Generate a Josephus sequence  $\mathbf{ci}$  of  $N$  length as follows: 1) initialize a vector  $\mathbf{b} = 1 : N$ ; 2) set  $\mathbf{ci}(1) = NP$  and remove  $\mathbf{ci}(1)$  from  $\mathbf{b}$ ; 3) start from the previously removed value, circularly shift  $NStep$  cells in  $\mathbf{b}$  and allocate the next value to  $\mathbf{ci}(2)$ ; 4)  $NStep = NStep + 1$ ; 5) repeat the operations 3) and 4) till all the  $N$  values in  $\mathbf{b}$  have been allocated to  $\mathbf{ci}$ ; 6) update  $NP$  using  $\mathbf{ci}(N)$ ;

*Step 4:* For these pixels in  $i$ -th row of the image, permute them into the positions  $\{(\mathbf{ri}(\mathbf{ci}(1)) + i, \mathbf{ci}(1)), (\mathbf{ri}(\mathbf{ci}(2)) + i, \mathbf{ci}(2)), \dots, (\mathbf{ri}(\mathbf{ci}(N)) + i, \mathbf{ci}(N))\}$ . Note that if  $\mathbf{ri}(\mathbf{ci}(k)) + i$  ( $k = 1 \sim N$ ) is larger than  $M$ , its modulus result to  $M$  is used to replace it;

*Step 5:* Repeat *Step 2* to *Step 4* for  $i = 2 \sim M$ .

It is noticed that to obtain more random sequences, we modify the generation procedures of Josephus sequence in

our scrambling. Specifically, we add one to the shifting steps  $MStep$  and  $NStep$  in each iteration, and update the  $NP$  using the last selected value when generating the next  $\mathbf{ci}$ .

To better explain the Josephus scrambling, Fig. 5 provides a numerical example using the image of size  $4 \times 4$  and the parameters  $MP = 3$ ,  $NP = 2$ ,  $MStep = 1$ , and  $NStep = 2$ . Fig. 5(a) demonstrates the generation of a position matrix  $\mathbf{O}$ . The Josephus sequence  $\mathbf{ri}$  is generated using  $MP$  and  $MStep$ , and four Josephus sequences  $\mathbf{cis}$  are generated using  $NP$  and  $NStep$ . In each iteration, we add the shifting steps  $MStep$  and  $NStep$  to one. When generating the next  $\mathbf{ci}$ , we update the  $NP$  using the last selected value. The position matrix  $\mathbf{O}$  of size  $4 \times 4$  is constructed using the  $\mathbf{ri}$  and four  $\mathbf{cis}$ . When obtaining the  $i$ -th row of  $\mathbf{O}$ , we use the  $\mathbf{ri}(\mathbf{ci}) + i$  as their row positions and directly use the corresponding  $\mathbf{ci}$  as their column positions. It is noticed that if  $\mathbf{ri}(\mathbf{ci}) + i$  is larger than 4, its modulus to 4 is used to replace it. The detailed generation of the position matrix  $\mathbf{O}$  is described as,

- As  $\mathbf{ri} = \{3, 4, 2, 1\}$  and the 1-st  $\mathbf{ci} = \{2, 4, 1, 3\}$ , the 1-st row of  $\mathbf{O}$  is  $\{(\mathbf{ri}(2) + 1, 2), ((\mathbf{ri}(4) + 1, 4)), ((\mathbf{ri}(1) + 1, 1)), ((\mathbf{ri}(3) + 1, 3))\} = \{(5, 2), (2, 4), (4, 1), (3, 3)\} \iff \{(1, 2), (2, 4), (4, 1), (3, 3)\}$ ;
- As  $\mathbf{ri} = \{3, 4, 2, 1\}$  and the 2-nd  $\mathbf{ci} = \{3, 2, 4, 1\}$ , the 2-nd row of  $\mathbf{O}$  is  $\{(\mathbf{ri}(3) + 2, 3), ((\mathbf{ri}(2) + 2, 2)), ((\mathbf{ri}(4) + 2, 4)), ((\mathbf{ri}(1) + 2, 1))\} = \{(4, 3), (6, 2), (3, 4), (5, 1)\} \iff \{(4, 3), (2, 2), (3, 4), (1, 1)\}$ ;
- As  $\mathbf{ri} = \{3, 4, 2, 1\}$  and the 3-rd  $\mathbf{ci} = \{1, 2, 3, 4\}$ , the 3-rd row of  $\mathbf{O}$  is  $\{(\mathbf{ri}(1) + 3, 1), ((\mathbf{ri}(2) + 3, 2)), ((\mathbf{ri}(3) + 3, 3)), ((\mathbf{ri}(4) + 3, 4))\} = \{(6, 1), (7, 2), (5, 3), (4, 4)\} \iff \{(2, 1), (3, 2), 1, 3), (4, 4)\}$ ;
- As  $\mathbf{ri} = \{3, 4, 2, 1\}$  and the 4-th  $\mathbf{ci} = \{4, 2, 3, 1\}$ , the 4-th row of  $\mathbf{O}$  is  $\{(\mathbf{ri}(4) + 4, 4), ((\mathbf{ri}(2) + 4, 2)), ((\mathbf{ri}(3) + 4, 3)), ((\mathbf{ri}(1) + 4, 1))\} = \{(5, 4), (8, 2), (6, 3), (7, 1)\} \iff \{(1, 4), (4, 2), (2, 3), (3, 1)\}$ .

Using the positions specified by  $\mathbf{O}$ , we can permute the pixels of  $\mathbf{P}$ , which is shown in Fig. 5(b).

Algorithm 1 demonstrates the pseudo-code of the Josephus scrambling.

## B. FILTERING DIFFUSION

An encryption scheme should achieve the diffusion property, which indicates that tiny changes in plaintext can affect the whole ciphertext. Our proposed IES-JFPD uses the filtering diffusion operation to achieve the diffusion property.

### 1) FILTER GENERATION

As discussed in Section II-B that when image filtering is used to do image encryption, the weight corresponding to the current pixel should equal to one and other weights should be integers. To obtain better diffusion effect, our operation sets the filter size as  $2 \times 2$ , and corresponds the lower-right weight to the current pixel, namely  $\mathbf{F}(2, 2) = 1$ . The other 3 weights are generated using the component  $\mathbf{d}$  of a sub-key as follows: 1) divide the 96-bit  $\mathbf{d}$  into three sequences with 32 bits and convert them into integers  $\mathbf{e} = (e_1, e_2, e_3)$ ;

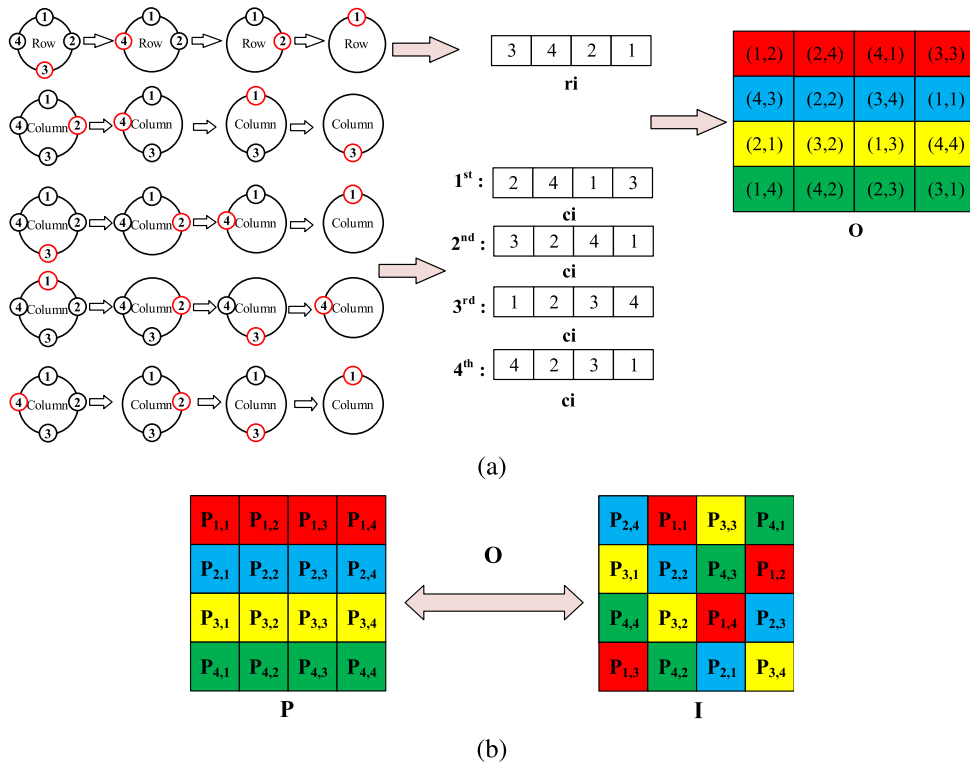


FIGURE 5. An example of the Josephus scrambling. (a) The procedure of generating position matrix  $O$ ; (b) pixel permutation using  $O$ .

**Algorithm 1** The Josephus Scrambling

**Input:** The plain-image  $P$  of size  $M \times N$ , and four parameters  $MP, NP, MStep$ , and  $NStep$  obtained by Eq. (6).

**Output:** The scrambling result  $I$ .

- 1: Initialize  $a = 1 : M$ ;
- 2: Set  $ri \in \mathbb{N}^{1 \times M}, MI = MP$ ;
- 3: **for**  $k = 1$  to  $M$  **do**
- 4:    $ri(k) = a(MI)$ ;
- 5:    $a(MI) = []$ ;
- 6:    $MI = (MI - 2 + MStep) \bmod (M - k) + 1$ ;
- 7:    $MStep = MStep + 1$ ;
- 8: **end for**
- 9: Set  $ci \in \mathbb{N}^{1 \times N}, NI = NP$ ;
- 10: **for**  $i = 1$  to  $M$  **do**
- 11:   Set  $b = 1 : N$ ;
- 12:   **for**  $j = 1$  to  $N$  **do**
- 13:      $ci(j) = b(NI)$ ;
- 14:      $row = (ri((ci(j)-1) \bmod M + 1) + i - 1) \bmod M + 1$ ;
- 15:      $column = ci(j)$ ;
- 16:      $b(NI) = []$ ;
- 17:      $NI = (NI - 2 + NStep) \bmod (N - j) + 1$ ;
- 18:      $NStep = NStep + 1$ ;
- 19:      $I(row, column) = P(i, j)$ ;
- 20:   **end for**
- 21:    $NI = ci(N)$ ;
- 22: **end for**

2) sort  $e$  and obtain an index vector  $v = (v_1, v_2, v_3)$ ; 3) disturb the three integers using vector  $v$  and obtain  $e' = (e_1 + v_1, e_2 + v_2, e_3 + v_3)$ . Then, the obtained filter can be written as

$$F = \begin{pmatrix} e_1 + v_1 & e_2 + v_2 \\ e_3 + v_3 & 1 \end{pmatrix}. \quad (7)$$

2) FILTERING DIFFUSION USING FILTER

The filtering diffusion is performed utilizing the filter shown in Eq. (7). First, initialize the filtering diffusion result  $C$  using the scrambling result. Then, update each pixel value of  $C$  using the filtering operation as

$$C'(x, y) = (y + \sum_{i,j \in \{1,2\}} F(i, j)C(x+i-2, y+j-2)) \bmod F, \quad (8)$$

where  $C'(x, y)$  represents the pixel that has been processed,  $F$  indicates the grayscale level of the cipher-image, e.g.  $F = 256$  if each pixel is presented using 8 bits. Notice that we add  $y$  in each operation, as this can avoid the failure of filtering diffusion in some special conditions, e.g. all pixel values of the image are zero.

Because the weight  $F(2, 2) = 1$ , Eq. (8) can be rewritten as Eq. (9), as shown at the bottom of the next page. When encrypting the current pixel, the three left and upper adjacent pixels have been encrypted. When decrypting the current pixel, the used three left and upper adjacent pixels haven't be

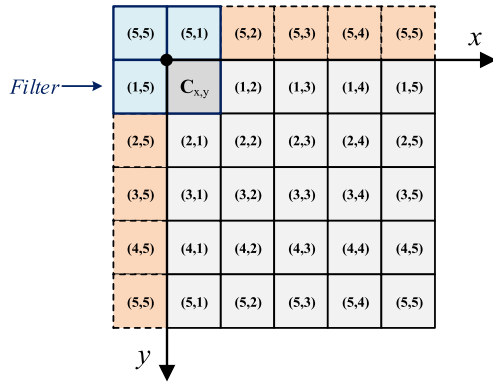


FIGURE 6. Demonstration of extending image.

decrypted, yet. Thus, Eq. (9) can be reversible and the original pixel  $C(x, y)$  can be recovered using the same filter  $F$  in the decryption process. The inverse operation of Eq. (9) is written as Eq. (10), as shown at the bottom of this page.

For the border pixels in the top row and left column, they have insufficient adjacent pixels. To deal with this situation, we use the border pixels in the opposite direction to extend the image when handling these pixels. Fig. 6 demonstrates this mechanism. As these extended pixels don't need to be stored, the size of encrypted image won't be enlarged.

Fig. 7 demonstrates the efficiency of the filtering diffusion using an image of size  $256 \times 256$ . Fig. 7(b) shows one-time filtering diffusion result to the image  $P_1$ . It indicates that the filtering diffusion can randomly change image pixels. Fig. 7(c) demonstrates the difference between two filtering diffusion results, where the two original images have one bit difference in position (50,60). As can be observed that their pixels behind the position (50,60) are totally different. This means that the filtering diffusion can spread little change of a pixel to all the pixels behind it. After two rounds of filtering diffusion, the tiny changes of a pixel could spread to the whole image, which can be seen from Fig. 7(d). Thus, the designed IES-JPIE can achieve high diffusion performance.

IV. SIMULATION RESULTS AND PROPERTY DISCUSSION  
A. SIMULATIONS RESULTS

A universal image encryption algorithm should have the ability to encrypt various kinds of images into random-like

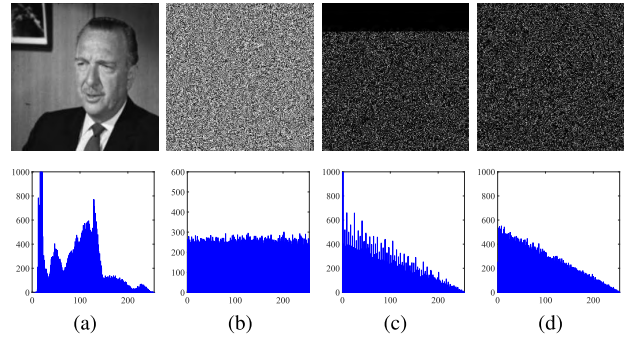


FIGURE 7. Demonstration of the filtering diffusion result. (a) Plain-image  $P_1$ ; (b) One-time filtering diffusion result of  $P_1$ ; (c) The difference between (b) and another one-time filtering result of  $P_2$ , where  $P_2$  has one bit difference in position(50,60) with  $P_1$ ; (d) The difference between  $P_1$  and  $P_2$  after two rounds of filtering diffusion.

cipher-images with high security levels. This section implements IES-JPDF in the MATLAB R2015b environment and analyzes its properties. The test images are from the BSDS<sup>1</sup> and CVonline<sup>2</sup> image databases.

Fig. 8 shows the simulation results of IES-JPDF for different images. All the test images are the pattern images that are hard to process, which can be seen from their histograms in Fig. 8(b). However, all the cipher-images are random-like and their pixels distribute very uniformly. Attackers can't get any valuable information by analyzing their distributions. With the correct key, one can totally recover the original images, which can be seen from Fig. 8(e).

B. PROPERTY DISCUSSION

The proposed IES-JPDF adopts the well-known confusion-diffusion architecture, the used Josephus scrambling can efficiently permute pixels to various columns and rows, and the filtering diffusion has high efficiency of spreading tiny changes of plain-image to the whole cipher-image. Thus, IES-JPDF can achieve the following advantages: 1) It has high performance of confusion and diffusion; 2) It can achieve strong ability of resisting some security risks, such as the exhaustive attack and differential attack. We will experimentally verify this in Section V.

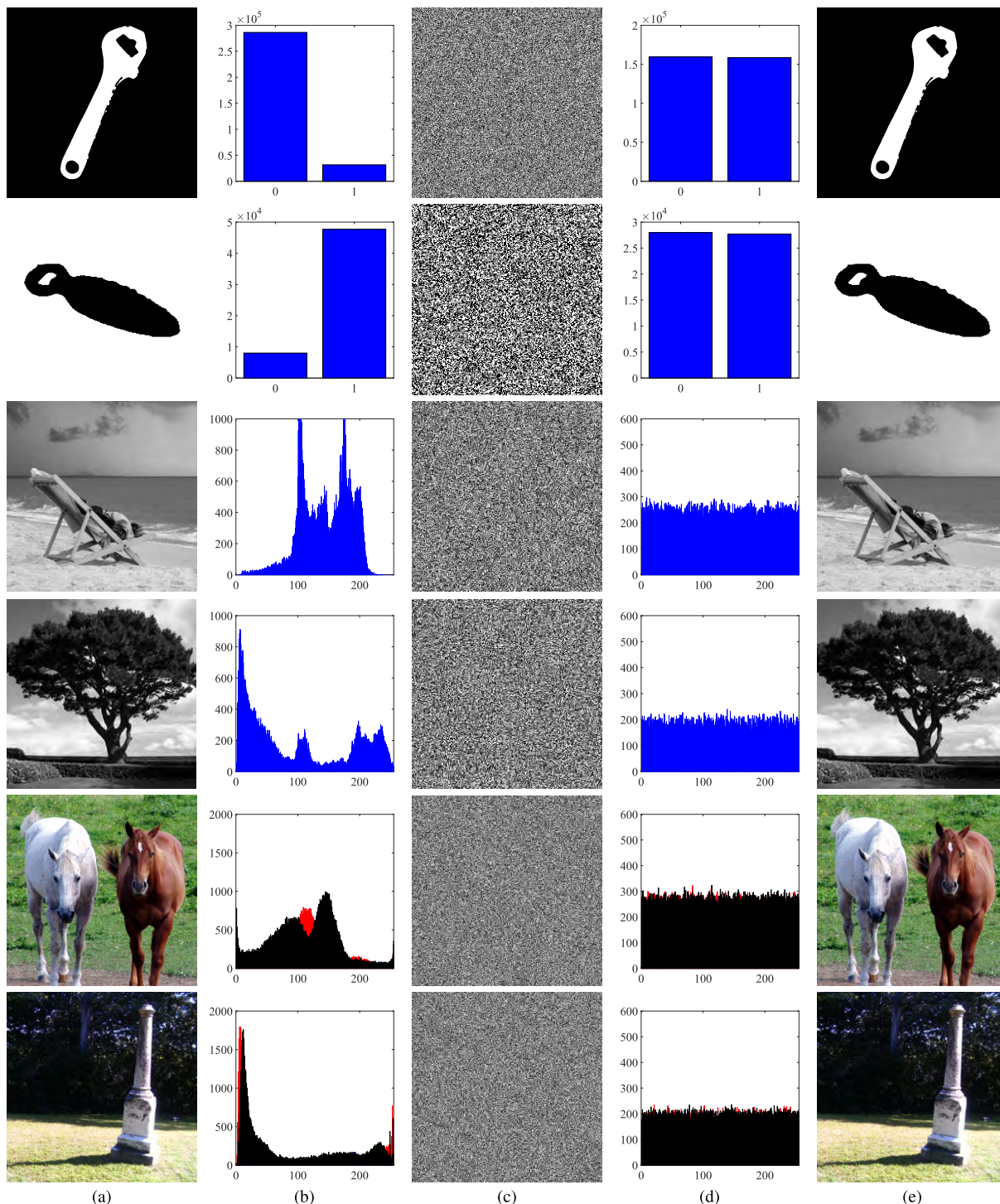
<sup>1</sup>[http://www.wisdom.weizmann.ac.il/~vision/Seg\\_Evaluation\\_DB/index.html](http://www.wisdom.weizmann.ac.il/~vision/Seg_Evaluation_DB/index.html)

<sup>2</sup><http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.html>

$$C'(x, y) = \left( y + C(x, y)F(2, 2) + \sum_{i,j \in \{1,2\} \cap (i,j) \neq (2,2)} F(i, j)C'(x + i - 2, y + j - 2) \right) \bmod F$$

$$= \left( y + C(x, y) + \sum_{i,j \in \{1,2\} \cap (i,j) \neq (2,2)} F(i, j)C'(x + i - 2, y + j - 2) \right) \bmod F \tag{9}$$

$$C(x, y) = \left( C'(x, y) - y - \sum_{i,j \in \{1,2\} \cap (i,j) \neq (2,2)} F(i, j)C'(x + i - 2, y + j - 2) \right) \bmod F \tag{10}$$



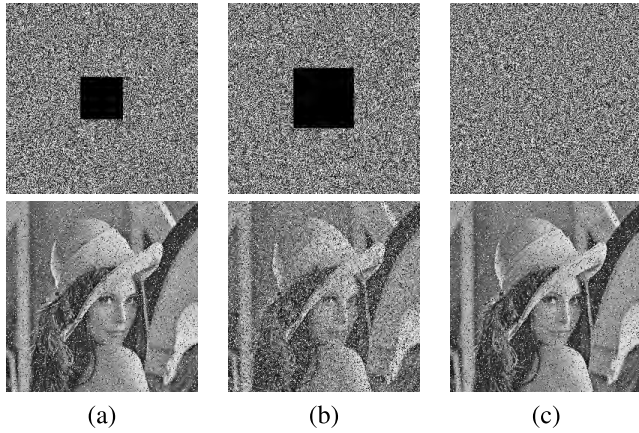
**FIGURE 8.** The experiment results of IES-JPDF. (a) The plain-images including binary, grayscale and color images; (b) The histograms of (a); (c) The cipher-images; (d) The histograms of (c); (e) The decrypted images.

The digital images may lose data or be blurred by noise when they are transmitted in networks. The proposed IES-JPDF has strong ability of defending data loss or noise. In the encryption process, the filtering diffusion can spread slight changes in the plain-image to the whole cipher-image. This makes IES-JPDF achieve good diffusion property. However,

in the decryption process, the inverse filtering diffusion can only spread the change of one pixel to several pixels. Thus, IES-JPDF has strong reliability of defending data loss and noise.

To quantitatively show the ability of IES-JPDF in defending data loss and noise, we use the peak signal-to-noise





**FIGURE 9.** The demonstration of IES-JPFD in defending data loss and noise. The first row shows the cipher-images with different kinds of data loss and noise, while the second row shows their decrypted images. (a) 5% data loss; (b) 10% data loss; (c) 1% salt and pepper noise.

**TABLE 2.** The MSEs and PSNRs between the original image and the decrypted images that their related cipher-images have different percentages of data loss or noise.

Data loss and noise attack	MSE	PSNR
5% data loss	21.172	34.873
10% data loss	40.398	32.067
1% salt and pepper noise	15.941	36.105

ratio (PSNR) and mean square error (MSE) to test the difference between the original images and the decrypted images. The PSNR is defined as

$$PSNR = 10 \times \log_{10} \left\{ \frac{(2^F - 1)^2}{MSE} \right\}, \quad (11)$$

where  $F$  represents the grayscale level of the image, and MSE is the difference between the decrypted image  $\mathbf{D}$  and original image  $\mathbf{P}$  and it is defined as

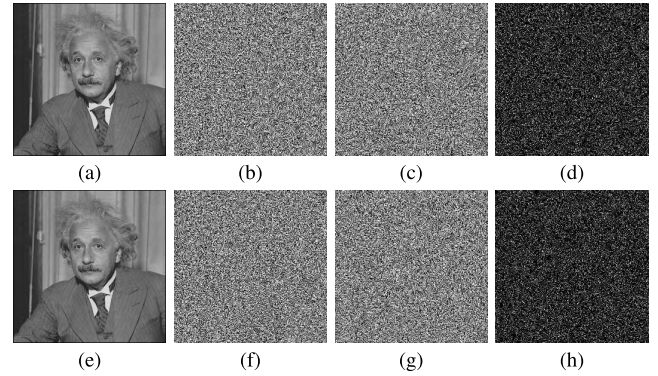
$$MSE = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (\mathbf{D}_{i,j} - \mathbf{P}_{i,j})^2, \quad (12)$$

where  $[M, N]$  is the image size. A higher PSNR and lower MSE indicate the less difference between the decrypted image and the original image.

Fig. 9 demonstrates the experiment results of IES-JPFD in defending data loss and noise and Table 2 shows the PSNR and MSE experiment results for different percentages of data loss and noise. One can see that when the cipher-images lose some data or are blurred by some noise, the decryption process can still recover the original images with high visual effect.

## V. SECURITY ANALYSIS

An efficient encryption algorithm should be able to defend the commonly used and unknown security risks. This section measures the security level of the IES-JPFD from different aspects and compares it with several typical image encryption



**FIGURE 10.** Key sensitivity analysis. (a) Plain-image  $\mathbf{P}$ ; (b) Cipher-image  $\mathbf{C}_1 = En(\mathbf{P}, \mathbf{K}_1)$ ; (c) Cipher-image  $\mathbf{C}_2 = En(\mathbf{P}, \mathbf{K}_2)$ ; (d) The difference between  $\mathbf{C}_1$  and  $\mathbf{C}_2$ ,  $|\mathbf{C}_1 - \mathbf{C}_2|$ ; (e) Decrypted image  $\mathbf{D}_1 = De(\mathbf{C}_1, \mathbf{K}_1)$ ; (f) Decrypted image  $\mathbf{D}_2 = De(\mathbf{C}_1, \mathbf{K}_2)$ ; (g) Decrypted image  $\mathbf{D}_3 = De(\mathbf{C}_1, \mathbf{K}_3)$ ; (h) The difference between  $\mathbf{D}_2$  and  $\mathbf{D}_3$ ,  $|\mathbf{D}_2 - \mathbf{D}_3|$ .

schemes: PSHM [29], LICM [40], PWLCM [41], CST [28], RCS [39], NSIE [37], PSCS [30], JHCM [42] and JTMC [43]. The test images are selected from the CVG-UGR<sup>3</sup> image database.

### A. SECRET KEY ANALYSIS

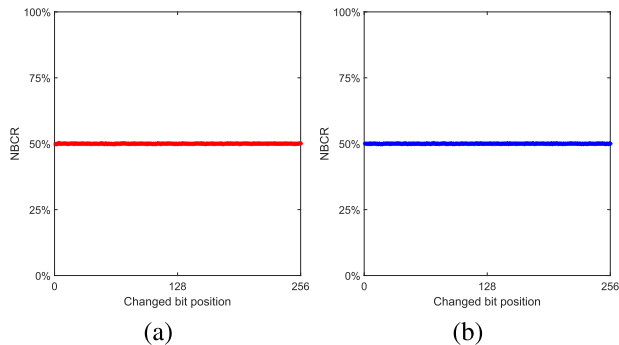
The secret key is the most important component of an encryption scheme. The key security includes two aspects: the key space and the key sensitivity. On one hand, the key space should be large enough to resist brute-force attack. The brute-force attack is to break a ciphertext by exhaustively searching all possible keys. According to the discussions in [46], the secret key has a sufficient security level against the brute-force attack if the key space is bigger than  $2^{100}$ . Considering the fast development of computer ability, the length of the secret key in IES-JPFD is 256 bit. Then the key space is  $2^{256}$ , which has high performance of resisting the brute-force attack.

On the other hand, the secret key must be extremely sensitive. Only using the correct key, one can correctly decrypt the original image. Fig. 10 demonstrates the key sensitivity experimental results in encryption and decryption processes. The encryption and decryption processes are denoted as  $\mathbf{C} = En(\mathbf{P}, \mathbf{K})$  and  $\mathbf{D} = De(\mathbf{C}, \mathbf{K})$ , respectively. First, randomly generate a secret key  $\mathbf{K}_1$ ,

$$\mathbf{K}_1 = 97157A6FC8E4BBE432C40D35F2716092 \\ EBA02E379817D636A144551DF49ADE37,$$

and then randomly change one bit of  $\mathbf{K}_1$  to obtain two other secret keys,  $\mathbf{K}_2$  and  $\mathbf{K}_3$ . When encrypting an identical image using  $\mathbf{K}_1$  and  $\mathbf{K}_2$ , the obtained two cipher-images are completely different (see Fig. 10(d)). The original image can be recovered using the correct key (see Fig. 10(e)). Using secret keys with one bit difference to recover an identical cipher-image, the decrypted results are noise images (see Figs. 10(f) and (g)), and also completely different

<sup>3</sup><http://decsai.ugr.es/cvg/dbimagenes>



**FIGURE 11.** The key sensitivity analysis results. (a) The NBCR between two cipher-images encrypted from an identical image with two secret keys owning one bit difference, (b) NBCR between two decrypted images from an identical cipher-image with two secret keys owning one bit difference.

(see Fig. 10(h)). As a result, the proposed IES-JPDF has quite sensitive secret key.

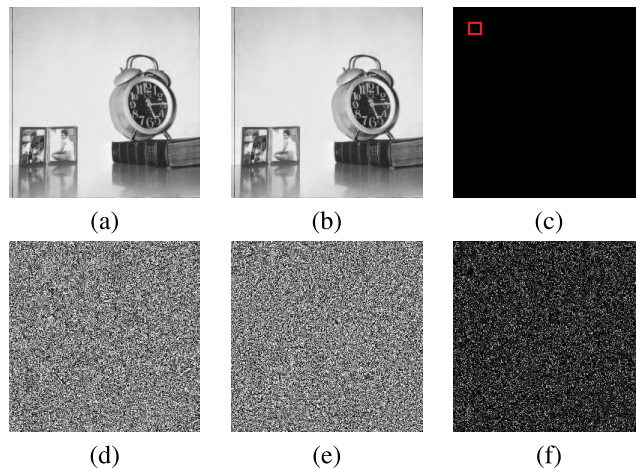
To test the sensitivity of the secret key in each bit, we test each of the 256 bits in  $K_1$  as follows: 1) change the bit of  $K_1$  to obtain another secret key  $K'$ ; 2) encrypt an identical plain-image using  $K_1$  and  $K'$  to obtain two cipher-images and calculate their difference; 3) decrypt an identical cipher-image using  $K_1$  and  $K'$  to obtain two decrypted results and calculate their difference; 4) repeat these operations until all the 256 bits in the secret key are tested. The difference of two images is calculated using the number of bit change rate (NBCR), which is defined as

$$NBCR = \frac{Hm[B_1, B_2]}{S} \times 100\%, \quad (13)$$

where  $Hm[\cdot]$  represents the function to calculate the Hamming distance between two bit streams, and  $B_1$  and  $B_2$  are two bit streams with length  $S$ . If the two bit streams  $B_1$  and  $B_2$  have weak correlation, their NBCR closes to 50%. Fig. 11 shows the key sensitivity analysis results. When changing any one bit of the secret key, the obtained two cipher-images in encryption process and two decrypted images in decryption process can achieve NBCR scores that approach to 50%. This means that the obtained two cipher-images and two decrypted images are completely different, indicating that each bit of the secret key in IES-JPDF is sensitive.

**B. ABILITY OF DEFENDING DIFFERENTIAL ATTACK**

The differential attack is an efficient and commonly used security attack. By tracing how the difference in plaintexts can affect the ciphertexts, the attackers try to establish the connections between plaintexts and ciphertexts, and use the established connections to reconstruct the original information without secret key. An encryption system with diffusion property has high performance of defending this attack. The diffusion property indicates that any slight difference in plaintext can cause change in the whole ciphertext. Fig. 12 demonstrates the diffusion property of the proposed IES-JPDF. One can see that when utilizing a same secret key to encrypt two images owning one bit difference, the generated encrypted



**FIGURE 12.** The demonstration of diffusion property. (a) Plain-image  $P_1$ ; (b) Plain-image  $P_2$ , which has one bit difference with  $P_1$  in position (50, 50); (c) Difference between  $P_1$  and  $P_2$ ,  $|P_1 - P_2|$ ; (d) Encrypted image  $C_1 = En(P_1, K_1)$ ; (e) Encrypted image  $C_2 = En(P_2, K_1)$ ; (f) Difference between  $C_1$  and  $C_2$ ,  $|C_1 - C_2|$ .

images are completely different, which can be observed from their difference in Fig. 12(f). This indicates that IES-JPDF has diffusion property.

To quantitatively test the ability of IES-JPDF in defending the differential attack, we calculate the number of pixel change rate (NPCR) and unified averaged changed intensity (UACI) of different encryption schemes. Suppose  $C_1$  and  $C_2$  are two encrypted images obtained from two plain-images with one-bit difference using a same secret key, the NPCR of  $C_1$  and  $C_2$  is defined as

$$NPCR(C_1, C_2) = \sum_{i=1}^M \sum_{j=1}^N \frac{W(i, j)}{G} \times 100\%, \quad (14)$$

where  $[M, N]$  indicates the image size,  $G$  is the number of pixels of an image, and  $W$  denotes the differences between  $C_1$  and  $C_2$ , and it is defined as

$$W(i, j) = \begin{cases} 0, & \text{if } C_1(i, j) = C_2(i, j); \\ 1, & \text{if } C_1(i, j) \neq C_2(i, j). \end{cases}$$

The UACI of  $C_1$  and  $C_2$  is defined as

$$UACI(C_1, C_2) = \sum_{i=1}^M \sum_{j=1}^N \frac{|C_1(i, j) - C_2(i, j)|}{T \times G} \times 100\%,$$

where  $T$  is the largest allowed pixel value in the image.

The NPCR and UACI critical values were developed in [47]. For NPCR test, an interval  $(U_{\alpha}^{*-}, U_{\alpha}^{*+})$  related to a significance level  $\alpha$  is calculated and the algorithm can pass the test if the obtained NPCR falls into the interval. For the UACI test, a threshold  $N_{\alpha}^*$  is calculated and the algorithm is considered to pass the test if the obtained UACI is bigger than  $N_{\alpha}^*$ . Wu et al. [47] have calculated out the critical  $U_{\alpha}^{*-}, U_{\alpha}^{*+}$  and  $N_{\alpha}^*$  when the significance level  $\alpha$  equals to 0.05, 0.01 and 0.001. As the significance level  $\alpha$  is

**TABLE 3.** The NPCR scores of different image encryption schemes using different sizes of images (significance level  $\alpha = 0.05$ ).

Image size	File name	NPCR (%)										
		IES-JFPD	PSHM [29]	LICM [40]	PWLCCM [41]	CST [28]	RCS [39]	NSIE [37]	PSCS [30]	JHCM [42]	JTMC [43]	
256 × 256	nat1	<b>99.6459</b>	<b>99.6276</b>	<b>99.5986</b>	99.5574	99.4644	99.3911	<b>99.6048</b>	<b>99.5880</b>	<b>99.6292</b>	<b>99.6231</b>	
	nat2	<b>99.6520</b>	<b>99.6048</b>	<b>99.6322</b>	<b>99.6139</b>	99.4506	99.5071	<b>99.6139</b>	<b>99.6200</b>	<b>99.6002</b>	<b>99.6017</b>	
	nat3	<b>99.6078</b>	<b>99.6460</b>	<b>99.6368</b>	<b>99.5971</b>	99.4583	99.5010	99.5498	<b>99.6444</b>	<b>99.6459</b>	99.2309	
	nat4	<b>99.6444</b>	<b>99.6017</b>	<b>99.5895</b>	<b>99.6124</b>	99.4751	99.4308	<b>99.8123</b>	<b>99.6139</b>	<b>99.6322</b>	<b>99.6231</b>	
	nat5	<b>99.6414</b>	<b>99.6200</b>	<b>99.6170</b>	99.5437	99.4262	99.2065	99.5544	<b>99.6032</b>	<b>99.6109</b>	<b>99.5803</b>	
	nat6	<b>99.6017</b>	<b>99.6353</b>	<b>99.6109</b>	<b>99.6139</b>	99.4613	99.4918	<b>99.7573</b>	<b>99.6353</b>	<b>99.6871</b>	<b>99.6002</b>	
	nat7	<b>99.5925</b>	<b>99.6170</b>	<b>99.6414</b>	<b>99.6109</b>	99.4415	99.3804	<b>99.6856</b>	<b>99.6048</b>	<b>99.5910</b>	<b>99.6353</b>	
512 × 512	61b	<b>99.5986</b>	<b>99.6238</b>	99.5838	<b>99.6135</b>	99.5758	99.0764	<b>99.8081</b>	<b>99.6311</b>	<b>99.6284</b>	<b>99.6021</b>	
	66b	<b>99.6128</b>	99.5876	<b>99.6238</b>	<b>99.5990</b>	99.5735	99.5483	<b>99.6822</b>	<b>99.5979</b>	<b>99.6009</b>	<b>99.6128</b>	
	67b	<b>99.6189</b>	<b>99.6223</b>	<b>99.6406</b>	<b>99.6231</b>	99.5437	99.5193	<b>99.8065</b>	<b>99.6181</b>	<b>99.5933</b>	<b>99.6009</b>	
	69b	<b>99.6120</b>	<b>99.6070</b>	<b>99.5899</b>	<b>99.6151</b>	99.5555	99.4751	<b>99.6013</b>	<b>99.5979</b>	<b>99.6135</b>	99.5883	
	70b	<b>99.6074</b>	<b>99.6055</b>	<b>99.5956</b>	<b>99.5967</b>	99.5677	99.4884	<b>99.7036</b>	<b>99.6208</b>	<b>99.6139</b>	<b>99.5971</b>	
	71b	<b>99.5914</b>	<b>99.6013</b>	<b>99.6166</b>	<b>99.6212</b>	99.5796	99.5788	<b>99.7116</b>	<b>99.6124</b>	<b>99.6196</b>	99.6780	
	73b	<b>99.6017</b>	<b>99.6036</b>	<b>99.6246</b>	<b>99.6131</b>	99.5868	99.5201	<b>99.7184</b>	<b>99.6185</b>	<b>99.6173</b>	<b>99.6143</b>	
75b	<b>99.5948</b>	<b>99.6170</b>	<b>99.6181</b>	<b>99.6109</b>	99.5632	<b>99.6063</b>	<b>99.8035</b>	<b>99.6135</b>	<b>99.5967</b>	99.5887		
1024 × 1024	3.2.25	<b>99.6041</b>	<b>99.6103</b>	<b>99.6191</b>	<b>99.6152</b>	<b>99.6027</b>	99.5611	<b>99.6530</b>	<b>99.5989</b>	<b>99.6079</b>	<b>99.6030</b>	
	5.3.01	<b>99.6033</b>	<b>99.6092</b>	<b>99.6109</b>	<b>99.6032</b>	99.5976	99.5368	<b>99.7048</b>	<b>99.6153</b>	<b>99.6138</b>	<b>99.6118</b>	
	5.3.02	<b>99.6073</b>	<b>99.6105</b>	<b>99.6139</b>	<b>99.6109</b>	99.5868	99.5720	99.4133	<b>99.6089</b>	<b>99.6170</b>	<b>99.6094</b>	
	≥ 99.5994	7.2.01	<b>99.6173</b>	<b>99.6227</b>	<b>99.6142</b>	<b>99.6083</b>	99.5959	99.5867	<b>99.7136</b>	<b>99.6105</b>	<b>99.6107</b>	<b>99.6176</b>
	testpat.1k	<b>99.5995</b>	<b>99.6151</b>	99.5953	<b>99.6044</b>	99.5868	99.5878	99.5144	<b>99.6091</b>	99.5788	<b>99.6050</b>	
Pass rate		20/20	19/20	18/20	18/20	1/20	1/20	16/20	20/20	19/20	16/20	

the test strength, a higher significance level indicates a stricter standard. To provide a convinced and strict comparison, our experiment uses the strictest NPCR and UACI critical values, namely  $\alpha = 0.05$ . Then according to the calculation in [47],  $\mathcal{N}_\alpha^* = 99.5693\%$  and  $(\mathcal{U}_\alpha^{*-}, \mathcal{U}_\alpha^{*+}) = (33.2824\%, 33.6447\%)$  for the size of  $256 \times 256$ .  $\mathcal{N}_\alpha^* = 99.5893\%$  and  $(\mathcal{U}_\alpha^{*-}, \mathcal{U}_\alpha^{*+}) = (33.3730\%, 33.5541\%)$  for the size of  $512 \times 512$ .  $\mathcal{N}_\alpha^* = 99.5994\%$  and  $(\mathcal{U}_\alpha^{*-}, \mathcal{U}_\alpha^{*+}) = (33.4183\%, 33.5088\%)$  for the size of  $1024 \times 1024$ .

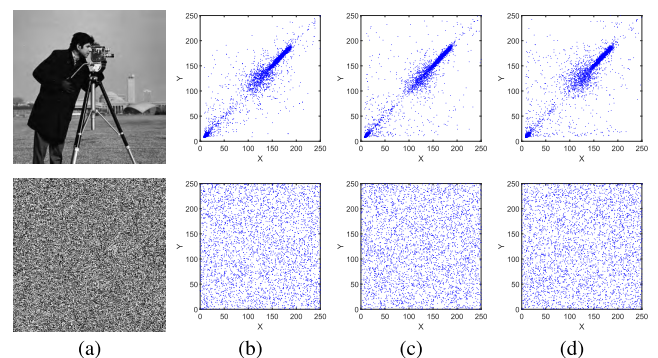
Table 3 and Table 4 show the obtained NPCR and UACI scores of different encryption algorithms for images with different sizes. One can observe that IES-JFPD and PSCS [30] can pass all the NPCR tests and IES-JFPD, PSHM [29], and LICM [40] can pass all the UACI tests. Only the proposed IES-JFPD can pass all the NPCR and UACI tests. This means that it has a strong ability of defending the differential attack.

**C. ADJACENT PIXEL CORRELATION**

A natural image may have strong correlations between its adjacent pixels, and these strong correlations can expose much information about the original image. Thus, an image encryption scheme should have the ability to break the correlations between adjacent pixels. The correlation coefficient (CC) provides a quantitative description about the correlations of adjacent pixel and it is defined as

$$CC(X, Y) = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}, \quad (15)$$

where  $X$  is a pixel sequence of the image and  $Y$  is another pixel sequence, in which each pixel is the adjacent pixel of  $X$  along one of the horizontal, vertical, and diagonal directions,  $E[\cdot]$  indicates the mathematical expectation,  $\mu$  is the mean value and  $\sigma$  is the standard derivation. A CC score closing



**FIGURE 13.** Adjacent pixel pairs of (a) the plain-image and cipher-image along the (b) horizontal, (c) vertical, and (d) diagonal directions.

to 0 means the weak correlation between pixel sequences  $X$  and  $Y$ , while a CC score closing to  $-1$  or  $1$  indicates the high correlation of  $X$  and  $Y$ .

Table 5 lists the CC scores of the ‘‘Cameraman’’ image and its cipher-images encrypted by several image encryption schemes. One can see that the cipher-image generated by IES-JFPD can achieve smallest absolute CC scores in both the horizontal and diagonal directions. This means that the proposed IES-JFPD can generate cipher-image with weak adjacent pixel correlations.

Fig. 13 plots the adjacent pixel pairs of the ‘‘Cameraman’’ image and its cipher-image encrypted by IES-JFPD. In each figure, 4000 pixel pairs are randomly selected with the horizontal, vertical, or diagonal directions. One can see that the pixel pairs of the plain-image mostly distribute on or nearby the diagonal line of the phase plane. This means that strong correlations exist between adjacent pixels in the original image. However, the pixel pairs of the cipher-image randomly distribute in the entire phase plane, which indicates that a

TABLE 4. The UACI scores of different image encryption schemes using different sizes of images (significance level  $\alpha = 0.05$ ).

Image size	File name	UACI (%)									
		IES-JPFD	PSHM [29]	LICM [40]	PWLICM [41]	CST [28]	RCS [39]	NSIE [37]	PSCS [30]	JHCM [42]	JTMC [43]
256 × 256	nat1	33.3797	33.6179	33.5226	33.3830	33.4018	33.9229	34.4944	33.5220	33.5053	33.6413
	nat2	33.4393	33.5607	33.5419	33.3936	33.3871	33.0890	33.3223	33.3932	33.4533	34.1214
	nat3	33.5112	33.4304	33.4905	33.3743	33.4571	33.4472	33.4875	33.3988	33.2852	36.3348
	nat4	33.4276	33.4335	33.4883	33.5272	33.6263	33.3890	32.8442	33.2704	33.3203	33.5050
	nat5	33.3430	33.5626	33.3369	33.2476	33.3033	33.8291	32.5726	33.3602	33.3742	33.5976
	nat6	33.3357	33.5775	33.5136	33.5775	33.4601	33.5481	34.4506	33.3169	33.6177	33.4639
	nat7	33.5037	33.4456	33.5821	33.2387	33.5489	33.3592	33.6113	33.2664	33.3554	33.4901
512 × 512	61b	33.4340	33.5293	33.4274	33.4165	33.4606	33.0445	32.6308	33.3893	33.4496	33.5432
	66b	33.3653	33.3767	33.5070	33.4830	33.3235	33.3987	33.3685	33.4459	33.4984	33.4550
	67b	33.4932	33.5052	33.4830	33.4524	33.4520	33.7916	33.7081	33.4274	33.4606	33.4578
	69b	33.4649	33.4984	33.4710	33.4183	33.4338	33.6068	33.7005	33.5155	33.5226	33.5548
	70b	33.4771	33.4326	33.3984	33.4187	33.4570	33.4365	33.9800	33.3967	33.4694	33.4175
	71b	33.5115	33.5132	33.4616	33.4363	33.4201	33.4176	33.0447	33.4112	33.4599	33.4696
	73b	33.4666	33.5736	33.4206	33.4611	33.4919	33.2537	33.0395	33.2934	33.4989	33.4951
75b	33.3928	33.4481	33.5279	33.5568	33.5123	33.4359	32.8966	33.5336	33.4083	33.4567	
1024 × 1024	3.2.25	33.4572	33.4688	33.5085	33.4836	33.4653	33.1989	33.4482	33.2480	33.4268	33.4811
	5.3.01	33.4765	33.4741	33.4651	33.3834	33.4468	33.1820	33.4149	33.4190	33.4761	33.4638
	5.3.02	33.4740	33.4341	33.4630	33.4180	33.4174	33.3483	33.0908	33.2921	33.4807	33.4777
	7.2.01	33.4909	33.4746	33.4755	33.4712	33.4725	33.5075	33.2945	33.5449	33.4540	33.4690
	testpat.1k	33.4367	33.4354	33.4240	33.4240	33.4601	33.4453	33.0628	33.4760	34.6524	33.5124
Pass rate		20/20	20/20	20/20	16/20	19/20	10/20	4/20	14/20	19/20	16/20

TABLE 5. The CC scores of the ‘‘Cameraman’’ image and its cipher-images encrypted by several image encryption schemes.

Image encryption algorithms	CC scores		
	Horizontal	Vertical	Diagonal
Plain-image	0.957001	0.931323	0.906305
IES-JPFD	<b>0.000378</b>	0.003031	<b>-0.002976</b>
PSHM [29]	0.008046	0.005859	0.020418
LICM [40]	-0.006003	-0.013350	-0.006758
PWLICM [41]	-0.021060	-0.017249	-0.010720
CST [28]	0.031091	0.030072	0.011433
RCS [39]	-0.028592	0.008530	-0.017066
NSIE [37]	-0.012077	<b>0.000483</b>	-0.015427
PSCS [30]	-0.019817	-0.016730	-0.016106
JHCM [42]	-0.007440	0.006913	-0.019123
JTMC [43]	0.007048	-0.001375	0.009184

pixel has weak correlation with its adjacent pixels in the cipher-image. Thus, the proposed IES-JPFD can efficiently decorrelate the strong correlations of the adjacent pixels in natural images.

D. LOCAL SHANNON ENTROPY

To resist various attacks, an expected cipher-image should have uniform-distributed pixels. Among many randomness test standards, the local Shannon entropy can quantitatively describe the randomness of an image from local view [48]. Mathematically, the local Shannon entropy can be written as

$$\overline{H_{k,T_B}} = \sum_{i=1}^k \frac{H(S_i)}{k}, \tag{16}$$

where  $k$  is the total number of used block,  $T_B$  indicates the number of pixel in each block,  $S_1 \sim S_k$  denote the  $k$  chosen

image blocks, and  $H(S_i)$  is the information entropy of  $S_i$ , which can be calculated as

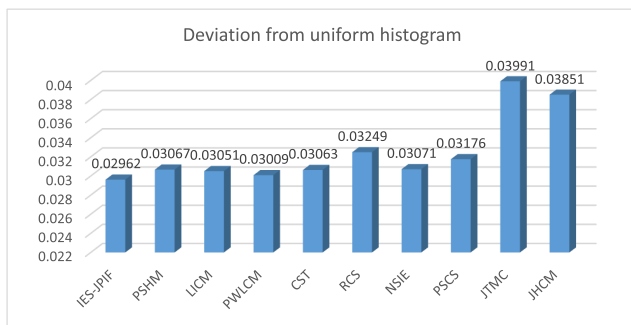
$$H(S_i) = - \sum_{j=1}^L Pr(j) \log_2 Pr(j), \tag{17}$$

where  $L$  is the grayscale level and  $Pr(j)$  denotes the probability of the  $j$ -th possible pixel.

Based on the setting in [48], we set the parameters  $k = 30$ ,  $T_B = 1936$  and the significance level  $\alpha = 0.05$ . Then an 8-bit image can pass the test if the obtained local Shannon entropy falls into the interval  $(h_{left}^*, h_{right}^*) = (7.9019013, 7.9030373)$ . Table 6 shows the local Shannon entropy results of the cipher-images encrypted using different encryption algorithms. One can observe that 11 cipher-images encrypted by IES-JPFD can pass the test

**TABLE 6.** The local Shannon entropy values of the cipher-images encrypted by several image encryption schemes ( $\alpha = 0.05, k = 30, T_B = 1936$ ).

File names	Local Shannon entropy values										
	IES-JPFD	PSHM [29]	LICM [40]	PWLICM [41]	CST [28]	RCS [39]	NSIE [37]	PSCS [30]	JHCM [42]	JTMC [43]	
nat1	7.9045137	<b>7.9025073</b>	7.9003463	7.9005481	7.9039709	7.8991822	7.9055808	<b>7.9019334</b>	7.9011959	7.9013392	
nat2	<b>7.9030149</b>	<b>7.9020267</b>	7.9011749	<b>7.9021302</b>	7.9007544	<b>7.9018166</b>	7.8988972	7.9046644	7.9041959	7.8519147	
nat3	7.9015089	<b>7.9026121</b>	7.9045564	<b>7.9026779</b>	<b>7.9016867</b>	<b>7.9029027</b>	7.9002384	7.9036739	7.9012995	7.8519146	
nat4	<b>7.9023721</b>	7.9010911	7.9014124	7.9014612	<b>7.9017630</b>	<b>7.9017223</b>	7.9054139	7.9036860	7.9040828	<b>7.9028722</b>	
nat5	<b>7.9024613</b>	7.9008055	<b>7.9020546</b>	7.9045483	7.9035597	7.9009318	<b>7.9028063</b>	7.8992271	7.9044027	7.9012584	
nat6	<b>7.9022007</b>	7.9032197	7.9014608	<b>7.9015224</b>	7.9013071	7.8991331	<b>7.9019211</b>	<b>7.9025826</b>	7.9044627	7.9007016	
nat7	7.9033595	7.9012648	7.8992418	7.9037053	7.9046493	7.9000667	7.9006150	<b>7.9019790</b>	7.9037452	7.9044096	
61b	<b>7.9030273</b>	7.9034822	7.9051410	7.9057800	7.9003507	7.9000156	7.9039646	7.9014409	7.9016076	<b>7.9025263</b>	
66b	<b>7.9025563</b>	<b>7.9023263</b>	7.9051059	7.8992979	7.9006968	7.8995499	7.9011833	<b>7.9032068</b>	7.9016324	<b>7.9021888</b>	
67b	<b>7.9026735</b>	<b>7.9023983</b>	7.9042299	7.9057464	<b>7.9014118</b>	7.9001948	7.9040076	7.9037526	<b>7.9025750</b>	<b>7.9020789</b>	
69b	<b>7.9023903</b>	<b>7.9004082</b>	7.9017849	<b>7.9032164</b>	7.9040141	7.8970821	7.9005158	7.9012652	7.9007100	<b>7.9021980</b>	
70b	7.9015474	7.8977740	7.9035768	<b>7.9016904</b>	<b>7.9024496</b>	7.8988024	<b>7.9034211</b>	7.8996038	<b>7.9025893</b>	7.9034663	
71b	7.9047979	7.9049649	<b>7.9027366</b>	7.9038349	<b>7.9031210</b>	7.8997981	7.9032420	7.9042243	<b>7.9019434</b>	7.8992216	
73b	7.9013718	<b>7.9022842</b>	<b>7.9022730</b>	<b>7.9016250</b>	7.9048264	7.8998642	<b>7.9022534</b>	<b>7.9017677</b>	7.9017094	<b>7.9020398</b>	
75b	<b>7.9027924</b>	7.9047071	7.9017525	7.9010514	<b>7.9018115</b>	7.9000315	7.9009619	7.9010060	7.9031991	7.9049766	
3.2.25	<b>7.9022964</b>	7.9032253	<b>7.9024714</b>	<b>7.9021433</b>	7.9006095	7.9000524	7.9005640	<b>7.9034174</b>	<b>7.9025129</b>	7.9036922	
5.3.01	7.9031988	7.9011992	7.9034736	7.9006874	<b>7.9033306</b>	7.8995328	<b>7.9023049</b>	7.9010272	7.9037935	7.9041326	
5.3.02	7.9039236	7.9041866	7.9037980	7.9048203	7.9006572	7.8954109	<b>7.9017226</b>	<b>7.9018753</b>	7.9009639	<b>7.9024531</b>	
7.2.01	7.9042409	7.9037162	<b>7.9025919</b>	7.8996649	7.8993959	7.9002487	<b>7.9034049</b>	<b>7.9031182</b>	7.9003341	7.9003529	
testpat.1k	<b>7.9025530</b>	7.9004960	<b>7.9021649</b>	7.9001138	<b>7.9029108</b>	7.9001344	7.9046440	7.9009145	7.8592195	7.8678365	
Pass rate	11/20	6/20	6/20	7/20	8/20	3/20	7/20	8/20	4/20	7/20	



**FIGURE 14.** The average DUH values of different image encryption algorithms.

and its pass rate is the largest. Thus, our proposed IES-JPFD can encrypt images with better randomness.

**E. DEVIATION FROM UNIFORM HISTOGRAM**

Different from the local Shannon entropy that can test the randomness of an image from the local view, the deviation from uniform histogram (DUH) can measure the randomness of an image in the global view [49]. For a cipher-image  $C$  of size  $M \times N$ , the DUH is defined as

$$D_H = \frac{\sum_{i=1}^F |H_{C_i} - H_{C'_i}|}{M \times N}, \tag{18}$$

where  $F$  represents the grayscale level,  $H_{C_i}$  is the histogram of the  $i$ -th possible value of  $C$ , the image  $C'$  is an ideal cipher-image with absolutely uniform distribution, and  $H_{C'_i}$  is the histogram of the  $i$ -th possible value in  $C'$  and it is defined as

$$H_{C'_i} = \begin{cases} \frac{M \times N}{F}, & 1 \leq i \leq F; \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

It is obvious that if the obtained  $D_H$  value is smaller, the actual cipher-image is more close to the ideal cipher-image. For different image encryption algorithms, Fig. 14 shows their average  $D_H$  values of 20 cipher-images. As can be seen that the proposed IES-JPFD has the smallest  $D_H$  value. This means that IES-JPFD can encrypt images into cipher-images that are most similar with the ideal cipher-image.

**VI. CONCLUSION**

This paper introduced an image encryption scheme utilizing the principles of the Josephus problem and the filtering technology, called IES-JPFD. The Josephus scrambling is derived from the Josephus problem and it can generate random sequences to fast separate the image pixels. The image filtering is commonly used to smooth an image. However, with an improper mask, it can also blur an image. Using this principle, the filtering diffusion can randomly change pixel values and spread tiny changes of original image to the whole encrypted result. The simulation results showed that IES-JPFD is able to encrypt different types of images into unrecognized cipher-images with random distribution. To prove the efficiency of IES-JPFD, we analyzed its security performance in terms of the secret key sensitivity, ability of defending differential attack, adjacent pixel correlation, local Shannon entropy and deviation from uniform histogram. The analysis results showed that IES-JPFD can achieve higher security performance than several classical image encryption algorithms.

**REFERENCES**

- [1] L. Halbeisen and N. Hungerbühler, "The Josephus problem," *J. Théorie Nombres Bordeaux*, vol. 9, no. 2, pp. 303–318, 1997.
- [2] C. Li, D. Lin, J. Lü, and F. Hao, "Cryptanalyzing an image encryption algorithm based on autoblocking and electrocardiography," *IEEE Multimedia*, to be published, doi: 10.1109/MMUL.2018.2873472.

- [3] C. Li, D. Lin, and J. Lü, "Cryptanalyzing an image-scrambling encryption algorithm of pixel bits," *IEEE Multimedia*, vol. 24, no. 3, pp. 64–71, Aug. 2017.
- [4] X. Chai, X. Fu, Z. Gan, Y. Lu, and Y. Chen, "A color image cryptosystem based on dynamic DNA encryption and chaos," *Signal Process.*, vol. 155, pp. 44–62, Feb. 2019.
- [5] S. Yi, Y. Zhou, and Z. Hua, "Reversible data hiding in encrypted images using adaptive block-level prediction-error expansion," *Signal Process., Image Commun.*, vol. 64, pp. 78–88, May 2018.
- [6] F. Y. Shih, *Digital Watermarking and Steganography: Fundamentals and Techniques*. Boca Raton, FL, USA: CRC Press, 2017.
- [7] Z. Hua and Y. Zhou, "Design of image cipher using block-based scrambling and image filtering," *Inf. Sci.*, vol. 396, pp. 97–113, Aug. 2017.
- [8] A. A. El-Latif, B. Abd-El-Atty, and M. Talha, "Robust encryption of quantum medical images," *IEEE Access*, vol. 6, pp. 1073–1081, 2018.
- [9] X. Huang and G. Ye, "An efficient self-adaptive model for chaotic image encryption algorithm," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 19, no. 12, pp. 4094–4104, 2014.
- [10] Z. Hua, S. Yi, and Y. Zhou, "Medical image encryption using high-speed scrambling and pixel adaptive diffusion," *Signal Process.*, vol. 144, pp. 134–144, Mar. 2018.
- [11] M. H. Annaby, M. A. Rushdi, and E. A. Nehary, "Image encryption via discrete fractional Fourier-type transforms generated by random matrices," *Signal Process., Image Commun.*, vol. 49, pp. 25–46, Nov. 2016.
- [12] Y.-Q. Zhang and X.-Y. Wang, "A new image encryption algorithm based on non-adjacent coupled map lattices," *Appl. Soft. Comput.*, vol. 26, pp. 10–20, Jan. 2015.
- [13] X. Li, Y. Wang, Q.-H. Wang, Y. Liu, and X. Zhou, "Modified integral imaging reconstruction and encryption using an improved sr reconstruction algorithm," *Opt. Lasers Eng.*, vol. 112, pp. 162–169, Sep. 2019.
- [14] *Advanced Encryption Standard (AES)*, FIPS PUB 197, Gaithersburg, MD, USA, 2001.
- [15] Z. Hua, B. Zhou, and Y. Zhou, "Sine chaotification model for enhancing chaos and its hardware implementation," *IEEE Trans. Ind. Electron.*, vol. 66, no. 2, pp. 1273–1284, Feb. 2019.
- [16] X. Huang and G. Ye, "An image encryption algorithm based on hyper-chaos and DNA sequence," *Multimedia Tools Appl.*, vol. 72, no. 1, pp. 57–70, 2014.
- [17] Y.-Q. Zhang and X.-Y. Wang, "A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice," *Inf. Sci.*, vol. 273, pp. 329–351, Jul. 2014.
- [18] Z. Hua, F. Jin, B. Xu, and H. Huang, "2D logistic-sine-coupling map for image encryption," *Signal Process.*, vol. 149, pp. 148–161, Aug. 2018.
- [19] D. Jiang, Y. Chen, X. Gu, L. Xie, and L. Chen, "Efficient and universal quantum key distribution based on chaos and middleware," *Int. J. Mod. Phys. B*, vol. 31, no. 2, p. 1650264, 2017.
- [20] N. Zhou, Y. Hu, L. Gong, and G. Li, "Quantum image encryption scheme with iterative generalized Arnold transforms and quantum image cycle shift operations," *Quantum Inf. Process.*, vol. 16, no. 6, p. 164, 2017.
- [21] N. Zhou, S. Pan, S. Cheng, and Z. Zhou, "Image compression–encryption scheme based on hyper-chaotic system and 2D compressive sensing," *Opt. Laser Technol.*, vol. 82, pp. 121–133, Aug. 2016.
- [22] X. Chai, Z. Gan, Y. Chen, and Y. Zhang, "A visually secure image encryption scheme based on compressive sensing," *Signal Process.*, vol. 134, pp. 35–51, May 2017.
- [23] R. Enayatifar, H. J. Sadaei, A. H. Abdullah, M. Lee, and I. F. Isnin, "A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata," *Opt. Lasers Eng.*, vol. 71, pp. 33–41, Aug. 2015.
- [24] X. Chai, Y. Chen, and L. Broyde, "A novel chaos-based image encryption algorithm using dna sequence operations," *Opt. Lasers Eng.*, vol. 88, pp. 197–213, Jan. 2017.
- [25] J. Chen, Z.-L. Zhu, L.-B. Zhang, Y. Zhang, and B.-Q. Yang, "Exploiting self-adaptive permutation–diffusion and DNA random encoding for secure and efficient image encryption," *Signal Process.*, vol. 142, pp. 340–353, Jan. 2018.
- [26] Z. Hua, B. Zhou, and Y. Zhou, "Sine-transform-based chaotic system with FPGA implementation," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2557–2566, Mar. 2018.
- [27] C. Li, G. Luo, K. Qin, and C. Li, "An image encryption scheme based on chaotic tent map," *Nonlinear Dyn.*, vol. 87, no. 1, pp. 127–133, 2017.
- [28] W. Liu, K. Sun, and C. Zhu, "A fast image encryption algorithm based on chaotic map," *Opt. Lasers Eng.*, vol. 84, pp. 26–36, Sep. 2016.
- [29] P. Ping, F. Xu, Y. Mao, and Z. Wang, "Designing permutation–substitution image encryption networks with Henon map," *Neurocomputing*, vol. 283, pp. 53–63, Mar. 2017.
- [30] Y. Zhou, L. Bao, and C. L. P. Chen, "Image encryption using a new parametric switching chaotic system," *Signal Process.*, vol. 93, no. 11, pp. 3039–3052, 2013.
- [31] C. Li, S. Li, M. Asim, J. Nunez, G. Alvarez, and G. Chen, "On the security defects of an image encryption scheme," *Image Vis. Comput.*, vol. 27, no. 9, pp. 1371–1381, 2009.
- [32] D. Arroyo, R. Rhouma, G. Alvarez, S. Li, and V. Fernandez, "On the security of a new image encryption scheme based on chaotic map lattices," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 18, no. 3, p. 033112, 2008.
- [33] X. Wu, B. Zhu, Y. Hu, and Y. Ran, "A novel color image encryption scheme using rectangular transform-enhanced chaotic tent maps," *IEEE Access*, vol. 5, pp. 6429–6436, 2017.
- [34] C. Li, Y. Liu, T. Xie, and M. Z. Q. Chen, "Breaking a novel image encryption scheme based on improved hyperchaotic sequences," *Nonlinear Dyn.*, vol. 73, no. 3, pp. 2083–2093, 2013.
- [35] E. Y. Xie, C. Li, S. Yu, and J. Lü, "On the cryptanalysis of Fridrich's chaotic image encryption scheme," *Signal Process.*, vol. 132, pp. 150–154, Mar. 2017.
- [36] C. Zhu and K. Sun, "Cryptanalyzing and improving a novel color image encryption algorithm using RT-enhanced chaotic tent maps," *IEEE Access*, vol. 6, pp. 18759–18770, 2018.
- [37] Y. Wu, Y. Zhou, J. P. Noonan, and S. Agaian, "Design of image cipher using latin squares," *Inf. Sci.*, vol. 264, pp. 317–339, Apr. 2014.
- [38] R. Enayatifar, A. H. Abdullah, I. F. Isnin, A. Altameem, and M. Lee, "Image encryption using a synchronous permutation–diffusion technique," *Opt. Lasers Eng.*, vol. 90, pp. 146–154, Mar. 2017.
- [39] X. Wang, Q. Wang, and Y. Zhang, "A fast image algorithm based on rows and columns switch," *Nonlinear Dyn.*, vol. 79, no. 2, pp. 1141–1149, 2015.
- [40] C. Cao, K. Sun, and W. Liu, "A novel bit-level image encryption algorithm based on 2D-LICM hyperchaotic map," *Signal Process.*, vol. 143, pp. 122–133, Feb. 2018.
- [41] L. Xu, Z. Li, J. Li, and W. Hua, "A novel bit-level image encryption algorithm based on chaotic maps," *Opt. Lasers Eng.*, vol. 78, pp. 17–25, Mar. 2016.
- [42] Y. Guo, L.-P. Shao, and L. Yang, "Bit-level image encryption algorithm based on Josephus and henon chaotic map," *Appl. Res. Comput.*, vol. 32, no. 4, pp. 1131–1137, 2015.
- [43] X. Wang, X. Zhu, and Y. Zhang, "An image encryption algorithm based on Josephus traversing and mixed chaotic map," *IEEE Access*, vol. 6, pp. 23733–23746, 2018.
- [44] G. Yang, H. Jin, and N. Bai, "Image encryption using the chaotic Josephus matrix," *Math. Problems Eng.*, vol. 2014, Mar. 2014, Art. no. 632060.
- [45] J. Wu and L. Tu, "An image encryption algorithm based on Josephus traversing and position disordering," in *Proc. Int. Conf. Cybern. Inform.* New York, NY, USA: Springer, 2014, pp. 1941–1946.
- [46] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *Int. J. Bifurcation Chaos*, vol. 16, no. 8, pp. 2129–2151, 2006.
- [47] Y. Wu, J. P. Noonan, and S. Agaian, "NPCR and UACI randomness tests for image encryption," *Cyber J., Multidisciplinary J. Sci. Technol., J. Sel. Areas Telecommun.*, vol. 1, no. 2, pp. 31–38, 2011.
- [48] Y. Wu, Y. Zhou, G. Saveriades, S. Agaian, J. P. Noonan, and P. Natarajan, "Local Shannon entropy measure with statistical tests for image randomness," *Inf. Sci.*, vol. 222, pp. 323–342, Feb. 2013.
- [49] I. F. Elashry, O. S. Faragallah, A. M. Abbas, S. El-Rabaie, and F. E. Abd-El-Samie, "A new method for encrypting images with few details using Rijndael and RC6 block ciphers in the electronic code book mode," *Inf. Secur. J., Global Perspective*, vol. 21, no. 4, pp. 193–205, 2012.



**ZHONGYUN HUA** (S'14–M'16) received the B.S. degree from Chongqing University, Chongqing, China, in 2011, and the M.S. and Ph.D. degrees from the University of Macau, Macau, China, in 2013 and 2016, respectively, all in software engineering.

He is currently an Associate Professor with the School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen, China. His research interests include chaotic system, chaos-based applications, and multimedia security.



**BINXUAN XU** received the B.S. degree in software engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2016. He is currently pursuing the master's degree with the School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen, China. His research interests include image security and reversible data hiding.



**HEJIAO HUANG** received the Ph.D. degree in computer science from the City University of Hong Kong, in 2004. She was an Invited Professor with INRIA, France. She is currently a Professor with the Harbin Institute of Technology (Shenzhen), Shenzhen, China. Her research interests include cloud computing, trustworthy computing, formal methods for system design, and wireless networks.

...



**FAN JIN** received the B.S. degree in software engineering from the Harbin Institute of Technology, Harbin, China, in 2016. He is currently pursuing the master's degree with the School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen, China. His research interests include chaotic systems and image encryption.