Information Sciences 396 (2017) 97–113

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Design of image cipher using block-based scrambling and image filtering

Zhongyun Hua [a], Yicong Zhou [b,*]

[a] School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, 518055, China
[b] Department of Computer and Information Science, University of Macau, Macau, 999078, China

ARTICLE INFO

Article history:
Received 12 May 2016
Revised 19 December 2016
Accepted 15 February 2017
Available online 17 February 2017

Keywords:
Block-based scrambling
Cryptosystem
Image encryption
Image filtering

ABSTRACT

The operation of image filtering is widely used to deblur digital images. However, using inappropriate masks, it can also blur images. Motivated by this concept, this paper introduces an image cipher using block-based scrambling and image filtering (IC-BSIF). To the best of our knowledge, this is the first time that image filtering has been used for image encryption. IC-BSIF uses the well-known substitution-permutation network and strictly follows the concepts of confusion and diffusion. The block-based scrambling is able to separate neighboring pixels to different rows and columns, and thus can efficiently weaken the strong correlations between adjacent pixels. Using randomly generated masks by the secret key, the image filtering can spread little change of plain-images to the entire pixels of cipher-images. Simulation results show that IC-BSIF can encrypt different kinds of images into noise-like ones, and security evaluations demonstrate that it can achieve better performance than several state-of-the-art encryption schemes.

1. Introduction

The two-dimensional (2D) digital images play more and more important roles in the modern digital technology. A 2D digital image is a type of 2D data, which carries data with a visualized and meaningful way. Then, if secret images are stolen, used or viewed by unauthorized users, disastrous security issues may happen. For example, if the spy steals the images of one new martial weapon, the hostile country may obtain the detail settings and parameters of the weapon by analyzing these images. Therefore, it is quite important to protect digital images and image cipher is an efficient solution of image security issue by encrypting a digital image into a random-like cipher-image [3,31,35]. Only with the correct secret key, one can recover the original image from the cipher-image.

Up to now, researchers have developed many image ciphers using different kinds of techniques [8,15–18,23,29]. Among these techniques, chaos theory is the most widely used one [1,21,34,36,37]. This is due to that chaotic maps have the properties of initial state sensitivity, unpredictability and ergodicity, and these properties can be found similar counterparts in image cipher [7,9]. Some examples of chaos-based image ciphers are as follows. In [33], a new image encryption scheme was designed using a new one-dimensional (1D) chaotic map, which is developed by combining two existing chaotic maps. In [11], an image cipher was developed using a new 2D chaotic map, named 2D-SLMM. In [10], another image cipher was proposed using a new 2D chaotic map, called 2D-LASM. For these chaos-based image ciphers, their security levels are highly

* Corresponding author.
E-mail addresses: huazyum@gmail.com (Z. Hua), yicongzhou@umac.mo (Y. Zhou).

http://dx.doi.org/10.1016/j.ins.2017.02.036
0020-0255/

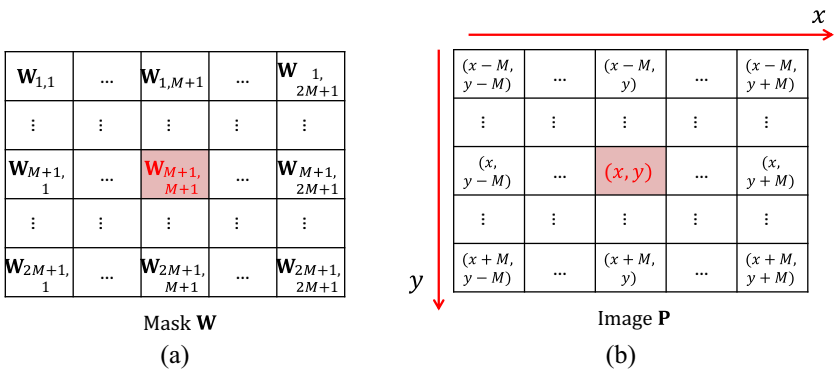

Fig. 1. Demonstration of image filtering. (a) Structure of a 2D mask; (b) a current pixel and its adjacent ones in an image.

dependent on the performance of their used chaotic maps. However, the chaotic behaviors of chaotic maps may degrade to periodic behaviors when they are implemented in the finite precision platforms. This greatly reduces the security levels of the corresponding encryption schemes [22]. Many researchers have proved that some chaos-based ciphers owning low security levels can be successfully attacked [13,14,20,28].

Besides chaos theory, many other techniques were also applied to designing image ciphers [2,6,30]. Some examples are as follows. In [27], the authors developed an image encryption scheme, which uses the Latin sequences to do pixel permutation and substitution. In [5], the authors proposed a novel image encryption scheme using Gray code. The Gray code is used to do pixel permutation and a plain pixel-related image diffusion structure is used to achieve the diffusion property.

It is well-known that image filtering is widely used in many digital image processing technologies, such as image deblurring, image smoothing and edge detection. This paper first presents the concept of using image filtering to encrypt a digital image, and then designs a new image cipher using block-based scrambling and image filtering (IC-BSIF). IC-BSIF adopts the well-known substitution-permutation network and strictly follows the confusion and diffusion concepts. The block-based scrambling first divides image into blocks, and then randomly shuffles pixels of each block into different rows and columns. It can simultaneously shuffle the row and column positions of a pixel and thus can achieve a high efficiency to reduce the strong correlations between adjacent pixels. Using randomly generated masks, the image filtering operation can spread little change in plain-image to the entire cipher-image. Simulation results and security analysis show that IC-BSIF can encrypt different kinds of digital images into random-like ones with high security levels.

The rest of this paper is organized as follows. Section 2 presents the concepts of image filtering. Section 3 introduces IC-BSIF and Section 4 displays its simulation results. Section 5 evaluates the security performance of IC-BSIF and the last section concludes this paper.

2. Concept of image encryption using image filtering

This section introduces the detail operation of image filtering, and presents the concepts of using image filtering to do image encryption.

2.1. Image filtering

Image filtering is to do convolution operation to a 2D image using a 2D matrix, called mask. More specifically, for each pixel of an image, image filtering takes the sum of products between the image pixels and the weights of the 2D mask. The current pixel is usually multiplied to the central of the mask and the adjacent pixels of the current one corresponds to other elements of the 2D mask. Fig. 1 shows the structure of a 2D mask and a current pixel with its adjacent ones in an image. Suppose the 2D mask is of size $(2M+1)\times(2M+1)$, the mathematical operation of image filtering can be defined as

$$Out_{x,y} = \sum_{i=-M}^{M}\sum_{j=-M}^{M} \mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}. \tag{1}$$

Image filtering can be used to smooth or sharp images, or to detect image edges. The used 2D masks in these operations have some common properties: (1) the sum of all the weights usually equals to 1 to make the operation result has the same brightness as the original image; (2) to make the mask symmetric, the size of the 2D mask is usually set as uneven, e.g. 5×5; (3) the current pixel corresponds to the central of the mask. Using an appropriate mask, image filtering can effectively remove the image noise. However, it can also blur an image with an inappropriate mask. By this concept, we can encrypt a digital image into a noise-like one using image filtering.

2.2. Image filtering for encryption

The traditional usage of image filtering don't need to recover the current pixel. However, the encryption process must be reversible in an encryption algorithm. First, we introduce Proposition 1 to identify that the image filtering operation satisfying some conditions can be reversible.

Proposition 1. *For any given mask* $\mathbf{W}$ *of size* $(2M+1)\times(2M+1)$, *image* $\mathbf{P}$ *of size* $X\times Y$ *and* $\mathbf{P}'$ *grayscale level F, the operation*

$$\mathbf{I}_{x,y}=\left(\sum_{i=-M}^{M}\sum_{j=-M}^{M}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)\bmod F \tag{2}$$

can be reversible and its inverse operation is

$$\mathbf{P}_{x,y}=\left(\mathbf{I}_{x,y}-\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)\bmod F$$

if $\mathbf{P}\in\mathbb{N}$, $\mathbf{W}\in\mathbb{N}$ *and* $\mathbf{W}_{M+1,M+1}=1$.

Proof. Because $\mathbf{W}_{M+1,M+1}=1$, Eq. (2) can be rewritten as

$$\begin{aligned}\mathbf{I}_{x,y}&=\left(\left(\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)+\mathbf{W}_{M+1,M+1}\mathbf{P}_{x,y}\right)\bmod F\\&=\left(\left(\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)+\mathbf{P}_{x,y}\right)\bmod F\\&=\left(\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)+\mathbf{P}_{x,y}-kF,\end{aligned}$$

where k is an integer. Substituting the above equation, we can obtain

$$\mathbf{P}_{x,y}=\mathbf{I}_{x,y}+kF-\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}. \tag{3}$$

Because $\mathbf{P}\in\mathbb{N}$ and F is $\mathbf{P}$'s grayscale level, $0\le\mathbf{P}_{x,y}<F$. Then Eq. (3) can be rewritten as

$$\begin{aligned}\mathbf{P}_{x,y}&=\left(\mathbf{I}_{x,y}+kF-\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)\bmod F\\&=\left(\mathbf{I}_{x,y}-\sum_{i,j\in\{-M,\ldots,M\}\cap(i,j)\neq(0,0)}\mathbf{W}_{i+M+1,j+M+1}\mathbf{P}_{x+i,y+j}\right)\bmod F,\end{aligned}$$

which completes the proof of Proposition 1. □

Using Proposition 1, we can design image encryption scheme using image filtering. To enhance the diffusion efficiency of the encryption algorithm, we set the lower-right weight of the mask corresponds to the current pixel. Then, the detail settings of the 2D mask used in image encryption are shown as

$$\mathbf{W}=\begin{pmatrix}\mathbf{W}_{1,1}&\cdots&\mathbf{W}_{1,M+1}&\cdots&\mathbf{W}_{1,2M+1}\\\vdots&\vdots&\vdots&\vdots&\vdots\\\mathbf{W}_{M+1,1}&\cdots&\mathbf{W}_{M+1,M+1}&\cdots&\mathbf{W}_{M+1,2M+1}\\\vdots&\vdots&\vdots&\vdots&\vdots\\\mathbf{W}_{2M+1,1}&\cdots&\mathbf{W}_{2M+1,M+1}&\cdots&\mathbf{1}\end{pmatrix}. \tag{4}$$

where $\mathbf{W}_{2M+1,2M+1}=1$ corresponding to the current pixel and $\mathbf{W}\in\mathbb{N}$. Thus, the upper and left adjacent pixels are used to process the current pixel $\mathbf{P}_{x,y}$. When processing the pixel $\mathbf{P}_{x,y}$, its upper and left adjacent pixels have been processed; when doing the inverse filtering to the pixel $\mathbf{I}_{x,y}$, its upper and left adjacent pixels haven't been recovered yet, because the processing order in the inverse filtering is opposite to that in the forward operation. This guarantees that the used adjacent pixel values in the filtering operation and its corresponding inverse operation are the same. Fig. 2 depicts an example of filtering operation and its corresponding inverse operation. It straightforwardly demonstrates that the used adjacent pixels

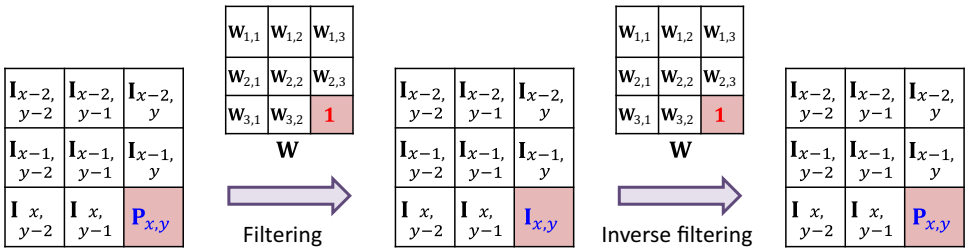

Fig. 2. An example of filtering operation and the corresponding inverse operation to an image pixel.

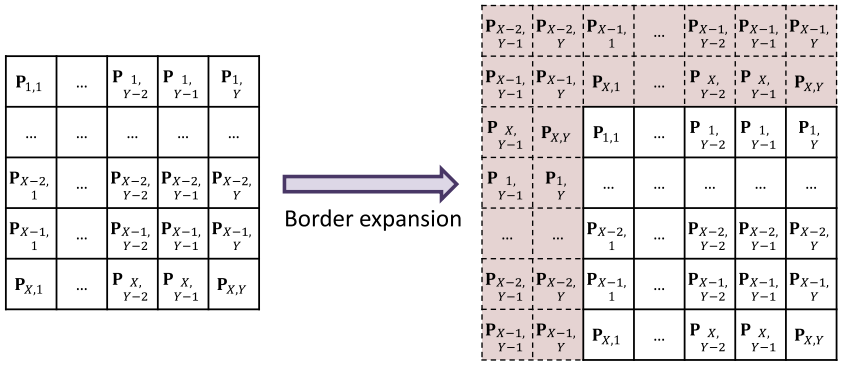

Fig. 3. Demonstration of handing border pixels.

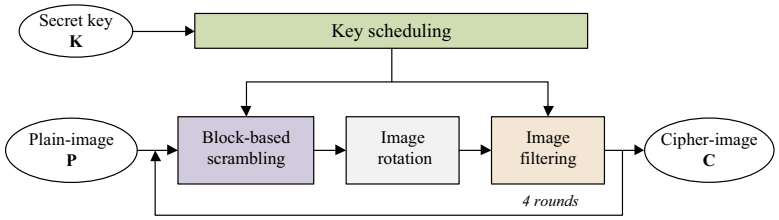

Fig. 4. Encryption process of IC-BSIF.

in the forward and backward operations are the same. Combining Proposition 1 and the mask presented in Eq. (4), we can design the image filtering operation for image encryption as

$$\mathbf{I}_{x,y} = \left(\left(\sum_{i,j \in \{-2M,\ldots,0\} \cap (i,j) \neq (0,0)} \mathbf{W}_{i+2M+1,\, j+2M+1} \mathbf{I}_{x+i,y+j} \right) + \mathbf{P}_{x,y} \right) \mod F, \tag{5}$$

where $\mathbf{P}$ is the input image, F is the grayscale level of $\mathbf{P}$. The inverse operation of image filtering can be calculated as

$$\mathbf{P}_{x,y} = \left(\mathbf{I}_{x,y} - \sum_{i,j \in \{-2M,\ldots,0\} \cap (i,j) \neq (0,0)} \mathbf{W}_{i+2M+1,\, j+2M+1} \mathbf{I}_{x+i,y+j} \right) \mod F. \tag{6}$$

The mask size in our proposed encryption algorithm is set as 3×3. Then, the eight upper and left adjacent pixels should be used to calculate a current pixel. Because the border pixels in the two leftmost columns and two uppermost rows don't have (or have insufficient) left and upper adjacent pixels, we use the rightmost and lowermost border pixels to handle them. Fig. 3 demonstrates the strategy of handing border pixels. It is noticed that the border pixels in the two leftmost columns and two uppermost rows can be correctly recovered. This is due to that when doing inverse filtering to them, those used rightmost and lowermost border pixels have been already recovered.

3. Image cipher

This section presents IC-BSIF and Fig. 4 shows its structure. The secret key is to generate pseudo-random numbers for the block-based scrambling and image filtering in each encryption round. The block-based scrambling fast shuffles neighboring

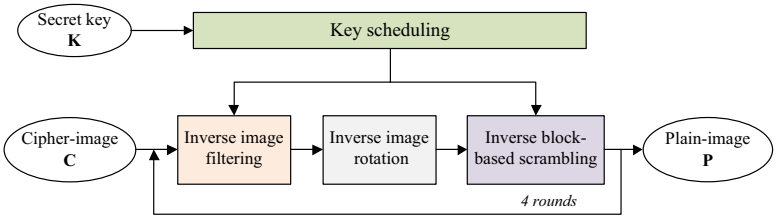

Fig. 5. Decryption process of IC-BSIF.

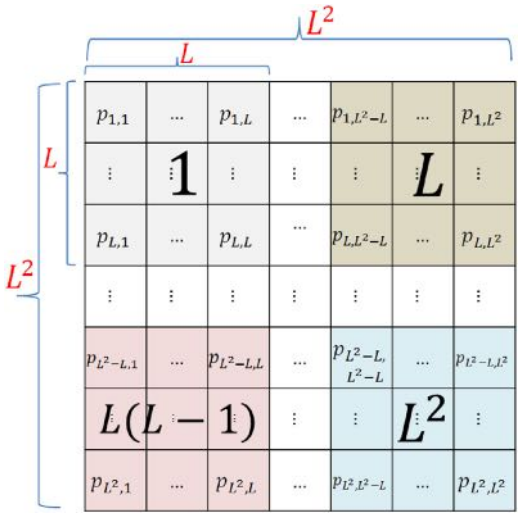

Fig. 6. Demonstration of diving an image into blocks.

pixels. The image rotation rotates image by 90 degrees clockwise. The image filtering is to randomly change pixel values. The proposed IC-BSIF is a private key encryption scheme. Using the identical secret key, the receiver can losslessly recover the original image by doing inverse operations of each step in the encryption process. The structure of decryption process is depicted in Fig. 5. The encryption process is represented as $\mathbf{C} = \mathtt{Enc}(\mathbf{P}, \mathbf{K})$ while the decryption process is denoted by $\mathbf{D} = \mathtt{Dec}(\mathbf{C}, \mathbf{K})$, where $\mathbf{K}$ is the secret key.

The secret key is of length 256 bits and composed of 8 parts, $\mathbf{K} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3, \mathbf{b}_4, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4\}$, in which $\mathbf{b}_1 \sim \mathbf{b}_4$ are the primitive sub-keys and $\mathbf{s}_1 \sim \mathbf{s}_4$ are the interference parameters to enlarge the key space. The sub-key in each encryption round is generated by doing bitwise XOR to $\mathbf{b}_i$ using $\mathbf{s}_i$,

$$\mathbf{k}_i = \mathbf{b}_i \oplus \mathbf{s}_i, \tag{7}$$

where $i \in \{1, 2, 3, 4\}$. The sub-key $\mathbf{k}_i$ is used as seeds to generate pseudo-random numbers in the block-based scrambling and image filtering.

3.1. Block-based scrambling

Neighboring pixels of natural images may have strong correlations, an image encryption scheme should be able to reduce these correlations. The block-based scrambling was designed to weakness these strong correlations by randomly separating neighboring pixels to different rows and columns. For an image of size $M \times N$, the block size L can be calculated by

$$L = \min\{\lfloor\sqrt{M}\rfloor, \lfloor\sqrt{N}\rfloor\}. \tag{8}$$

The block-based scrambling is performed within range $L^2 \times L^2$. First, the image of size $L^2 \times L^2$ is divided into L^2 blocks and each block is of size $L \times L$ (Fig. 6 depicts the operation). Then, a scrambling box $\mathbf{O}$ of size $L^2 \times L^2$ is generated using the sub-key. Finally, the pixels in a block can be permutated to different rows and columns based on $\mathbf{O}$. The detail procedure is described as follows:

- *Step 1*: Calculate the block size L using Eq. (8). Divide the image of size $L^2 \times L^2$ into L^2 blocks and each block is of size $L \times L$, which can be seen in Fig. 6.
- *Step 2*: Generate two vectors $\mathbf{A}$ and $\mathbf{B}$ using the sub-key. Both are of length L^2.
- *Step 3*: Sort $\mathbf{A}$ and $\mathbf{B}$ to obtain two index vectors $\mathbf{I}$ and $\mathbf{J}$, respectively.

- *Step 4*: Initialize the scrambling box $\mathbf{O}$ of size $L^2 \times L^2$ and assign each column of $\mathbf{O}$ as $\mathbf{I}$.
- *Step 5:* Shift each column of $\mathbf{O}$ using each element of $\mathbf{J}$.
- *Step 6:* Set row index $i = 1$.
- *Step 7:* For the ith block of the original image, permutate its pixels to the positions $\{(1, \mathbf{O}_{i,1}), (2, \mathbf{O}_{i,2}), \cdots, (L^2, \mathbf{O}_{L^2})\}$ in the scrambled result $\mathbf{R}$.
- *Step 8:* Repeat *Step 6* to *Step 7* $L^2 - 1$ times for $i = 2 \sim L^2$.

Algorithm 1 shows the pseudocode of the block-based scrambling while Algorithm 2 shows that of the inverse block-based scrambling. The inverse block-based scrambling is to do the inverse operations of each step in the block-based scrambling using the same scrambling box $\mathbf{O}$.

Algorithm 1 Block-based scrambling.

Input: Image $\mathbf{P}$ of size $M \times N$ and sub-key $\mathbf{k}$.
Output: Scrambled result $\mathbf{R}$.

Calculate L using Eq. (8).
$\mathbf{V} = \mathtt{PRNG}(\mathbf{k})$, where $\mathbf{V} \in \mathbb{Z}^{2L^2 \times 1}$;{$\mathtt{PRNG}(\cdot)$ can be any pseudo-random number generater.}
$\mathbf{A} = \mathbf{V}_{1:L^2}$; $\mathbf{B} = \mathbf{V}_{L^2+1:2L^2}$;
$[\mathbf{A}', \mathbf{I}] = \mathtt{Sort}(\mathbf{A})$, where $\mathbf{A}' = \mathbf{A}_{\mathbf{I}}$; $[\mathbf{B}', \mathbf{J}] = \mathtt{Sort}(\mathbf{B})$, where $\mathbf{B}' = \mathbf{B}_{\mathbf{J}}$;
Set $\mathbf{R} = \mathbf{P}$, $\mathbf{O} \in \mathbb{N}^{L^2 \times L^2}$.
for $j = 1$ to L^2 **do**
  for $i = 1$ to L^2 **do**
    $m = ((i - \mathbf{J}(j) - 1) \mod L^2) + 1$;
    $\mathbf{O}_{i,j} = \mathbf{I}_m$;
  end for
end for
for $i = 1$ to L^2 **do**
  for $j = 1$ to L^2 **do**
    $m_1 = \lfloor (i-1)/L \rfloor \times L + 1$; $n_1 = ((i-1) \mod L) \times L + 1$;
    $m_2 = \lfloor (j-1)/L \rfloor$; $n_2 = (j-1) \mod L$;
    $x = m_1 + m_2$; $y = n_1 + n_2$;
    $\mathbf{R}(j, \mathbf{O}(i, j)) = \mathbf{P}(x, y)$;
  end for
end for

Algorithm 2 Inverse block-based scrambling.

Input: Scrambled image $\mathbf{R}$ of size $M \times N$ and sub-key $\mathbf{k}$.
Output: Original Image $\mathbf{P}$.

Calculate L using Eq. (8).
$\mathbf{V} = \mathtt{PRNG}(\mathbf{k})$, where $\mathbf{V} \in \mathbb{Z}^{2L^2 \times 1}$;{$\mathtt{PRNG}(\cdot)$ can be any pseudo-random number generater.}
$\mathbf{A} = \mathbf{V}_{1:L^2}$; $\mathbf{B} = \mathbf{V}_{L^2+1:2L^2}$;
$[\mathbf{A}', \mathbf{I}] = \mathtt{Sort}(\mathbf{A})$, where $\mathbf{A}' = \mathbf{A}_{\mathbf{I}}$; $[\mathbf{B}', \mathbf{J}] = \mathtt{Sort}(\mathbf{B})$, where $\mathbf{B}' = \mathbf{B}_{\mathbf{J}}$;
Set $\mathbf{P} = \mathbf{R}$, $\mathbf{O} \in \mathbb{N}^{L^2 \times L^2}$.
for $j = 1$ to L^2 **do**
  for $i = 1$ to L^2 **do**
    $m = ((i - \mathbf{J}(j) - 1) \mod L^2) + 1$;
    $\mathbf{O}_{i,j} = \mathbf{I}_m$;
  end for
end for
for $i = 1$ to L^2 **do**
  for $j = 1$ to L^2 **do**
    $m_1 = \lfloor (i-1)/L \rfloor \times L + 1$; $n_1 = ((i-1) \mod L) \times L + 1$;
    $m_2 = \lfloor (j-1)/L \rfloor$; $n_2 = (j-1) \mod L$;
    $x = m_1 + m_2$; $y = n_1 + n_2$;
    $\mathbf{P}(x, y) = \mathbf{R}(j, \mathbf{O}(i, j))$;
  end for
end for

Fig. 7 shows a numeral example of the block-based scrambling of size 4×4, namely $L = 2$ according to Eq. (8). Fig. 7(a) shows the generation process of $\mathbf{O}$ from a sub-key; Fig. 7(b) displays the pixels of the original image $\mathbf{P}$ and their distribu-

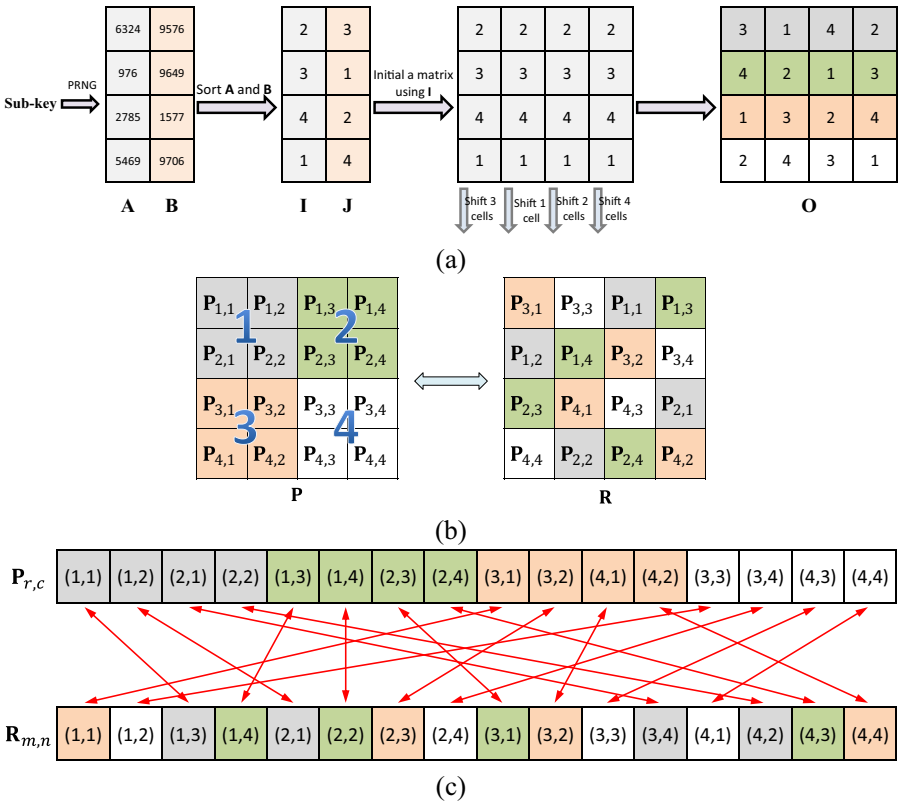

Fig. 7. Demonstration of the block-based scrambling with a block of size 4 × 4: (a) generation process of the scrambling box **O**; (b) pixels of the original image **P** and their distributions in the scrambled result **R**; (c) one-to-one pixel position mapping between **P** and **R**.

tions in the scrambled result **R**; and Fig. 7(c) demonstrates pixel position mapping between **P** and **R**. The detail scrambling process is described as follows: the 1st row of **O** is {3, 1, 4, 2}, then the corresponding 1st image block in gray, namely the pixels in positions {(1, 1), (1, 2), (2, 1), (2, 2)} of **P** map to the pixels in positions {(1, 3), (2, 1), (3, 4), (4, 2)} of **R**; The 2nd row of **O** is {4, 2, 1, 3}, then the corresponding 2nd image block in cyan, namely the pixels in positions {(1, 3), (1, 4), (2, 3), (2, 4)} of **P** map to the pixels in positions {(1, 4), (2, 2), (3, 1), (4, 3)} of **R**; The 3rd row of **O** is {1, 3, 2, 4}, then the corresponding 3rd image block in khaki, namely the pixels in positions {(3, 1), (3, 2), (4, 1), (4, 2)} of **P** map to the pixels in positions {(1, 1), (2, 3), (3, 2), (4, 4)} of **R**; The 4th row of **O** is {2, 4, 3, 1}, then the corresponding 4th image block in white, namely the pixels in positions {(3, 3), (3, 4), (4, 3), (4, 4)} of **P** map to the pixels in positions {(1, 2), (2, 4), (3, 3), (4, 1)} of **R**.

Fig. 8 shows an image example of the block-based scrambling of size 256 × 256. According to Eq. (8), the block size $L = \min\{\lfloor\sqrt{256}\rfloor, \lfloor\sqrt{256}\rfloor\} = 16$. Fig. 8(a) shows the straightforward result of the generated scrambling box **O**. First, divide the input image into $L^2 = 256$ image blocks and each block is of size 16 × 16. Then, permute the pixels in each block to different rows and columns according to **O**. With one-time block-based scrambling, the pixel positions are totally shuffled, which can be observed from the scrambling result in Fig. 8(c). Fig. 8(d) shows the inverse block-based scrambling result using the same scrambling box **O**. This means that the inverse block-based scrambling can completely recover the original image.

3.2. Image rotation

As the block-based scrambling is to randomly shuffle image pixel positions within range $L^2 \times L^2$ and the image block size L is calculated by Eq. (8). For an image of size $M \times N$, if it satisfies that $L = \sqrt{M} = \sqrt{N}$, all the pixels can be shuffled; otherwise, only the pixels within range $L^2 \times L^2$ can be shuffled and the rest pixels still locate at the same positions. To totally shuffle all the pixel positions, we rotate image by 90 degrees clockwise after the block-based scrambling. Then all the pixels can be shuffled after four encryption rounds. Note that the inverse image rotation is to rotate image 90 degrees anticlockwise in the decryption process.

3.3. Image filtering

The image filtering can randomly change the pixel values and spread little change of the plain-image to the entire pixels of the cipher-image to achieve the diffusion property. It first utilizes a matrix **Q** to normalize the pixels of the image, and

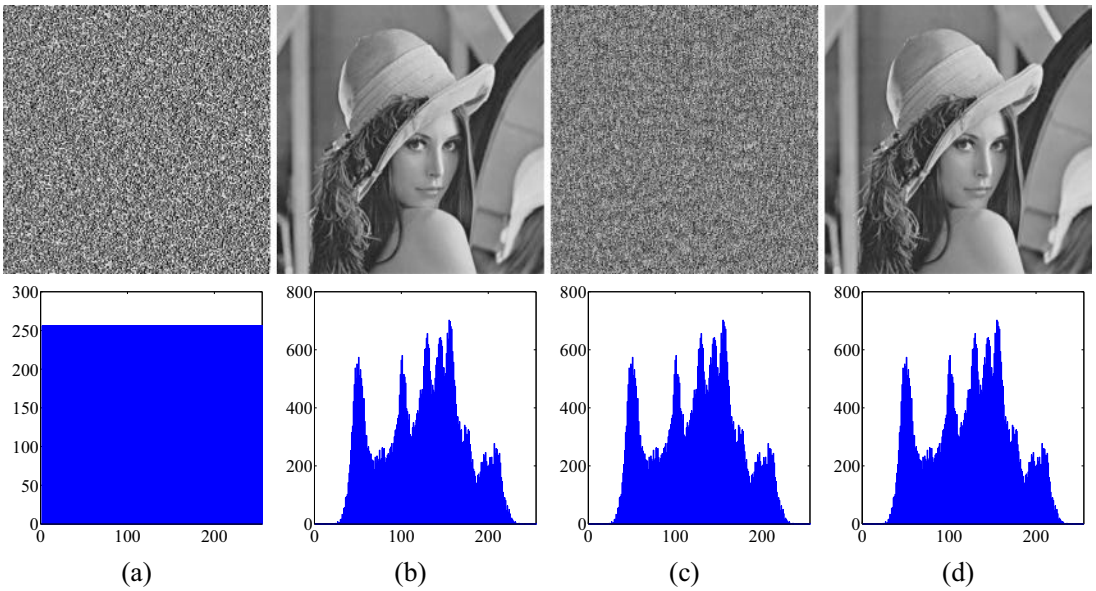

Fig. 8. Demonstration of the block-based scrambling results with their histograms: (a) scrambling box **O**; (c) input image; (c) operation result; (d) inverse operation result.

then uses a mask **W** satisfying the requirements of Eq. (4) to filter the normalized result. The matrix **Q** and mask **W** are randomly generated from the sub-key in each encryption round.

*3.3.1. Image normalization using matrix **Q***

The image normalization can reduce the patterns of natural image to balance the numbers of pixels in different intensity levels. The normalization operation to the block-based scrambling result **R** using the randomly generated matrix **Q** is defined as

$$\mathbf{R}_{i,j} = (\mathbf{R}_{i,j} + \mathbf{Q}_{i,j}) \mod F, \tag{9}$$

where $1 \le i \le M$, $1 \le j \le N$ and F is the grayscale level of the image. For example, $F = 256$ if an image pixel is represented by 8 bits. The inverse operation in the decryption process is defined as

$$\mathbf{R}_{i,j} = (\mathbf{R}_{i,j} - \mathbf{Q}_{i,j}) \mod F. \tag{10}$$

3.3.2. Image filtering

The proposed encryption algorithm uses the mask **W** of size 3×3 to filter the image. According the requirements of Eq. (4), we set $\mathbf{W}_{3,3} = 1$ and other weights as random integers generated using the sub-key. As the border pixels in the two leftmost columns and two uppermost rows don't have (or have insufficient) upper and left neighborhood pixels, the rightmost and lowermost border pixels are used to handle these pixels, which is shown as Fig. 3.

Suppose the normalization result **R** is of size $M \times N$, **C** is the image filtering result. First, initialize **C** using the pixel values of **R**. Then, update each pixel value of **C** using the following equation,

$$\begin{aligned}\mathbf{C}_{i,j} &= \sum_{m,n\in\{1,2,3\}} \mathbf{T}_{m,n} \times \mathbf{W}_{m,n} \mod F \\ &= (\mathbf{T}_{1,1}\mathbf{W}_{1,1} + \mathbf{T}_{1,2}\mathbf{W}_{1,2} + T_{1,3}\mathbf{W}_{1,3} + \mathbf{T}_{2,1}\mathbf{W}_{2,1} + \mathbf{T}_{2,2}\mathbf{W}_{2,2} \\ &\quad + \mathbf{T}_{2,3}\mathbf{W}_{2,3} + \mathbf{T}_{3,1}\mathbf{W}_{3,1} + \mathbf{T}_{3,2}\mathbf{W}_{3,2} + \mathbf{T}_{3,3}\mathbf{W}_{3,3}) \mod F,\end{aligned} \tag{11}$$

where $1 \le i \le M$ and $1 \le j \le N$, **T** is a 3×3 image block, whose elements are from **C**. Because the weight $\mathbf{W}_{3,3} = 1$ and it corresponds to the current pixel $\mathbf{R}_{i,j}$, the element of **T** with position (3, 3) is the current pixel, namely $\mathbf{T}_{3,3} = \mathbf{R}_{i,j}$. Thus, Eq. (11) can be rewrote as

$$\begin{aligned}\mathbf{C}_{i,j} &= (\mathbf{T}_{1,1}\mathbf{W}_{1,1} + \mathbf{T}_{1,2}\mathbf{W}_{1,2} + \mathbf{T}_{1,3}\mathbf{W}_{1,3} + \mathbf{T}_{2,1}\mathbf{W}_{2,1} + \mathbf{T}_{2,2}\mathbf{W}_{2,2} \\ &\quad + \mathbf{T}_{2,3}\mathbf{W}_{2,3} + \mathbf{T}_{3,1}\mathbf{W}_{3,1} + \mathbf{T}_{3,2}\mathbf{W}_{3,2} + \mathbf{R}_{i,j}) \mod F.\end{aligned} \tag{12}$$

In the decryption process, using the same operations to generate the mask **W** and to obtain the image block **T**, the inverse operation of Eq. (12) can be defined as

$$\mathbf{R}_{i,j} = \left(\mathbf{C}_{i,j} - \sum_{m,n\in\{1,2,3\}\cap(m,n)\neq(3,3)} \mathbf{T}_{m,n} \times \mathbf{W}_{m,n}\right) \mod F$$

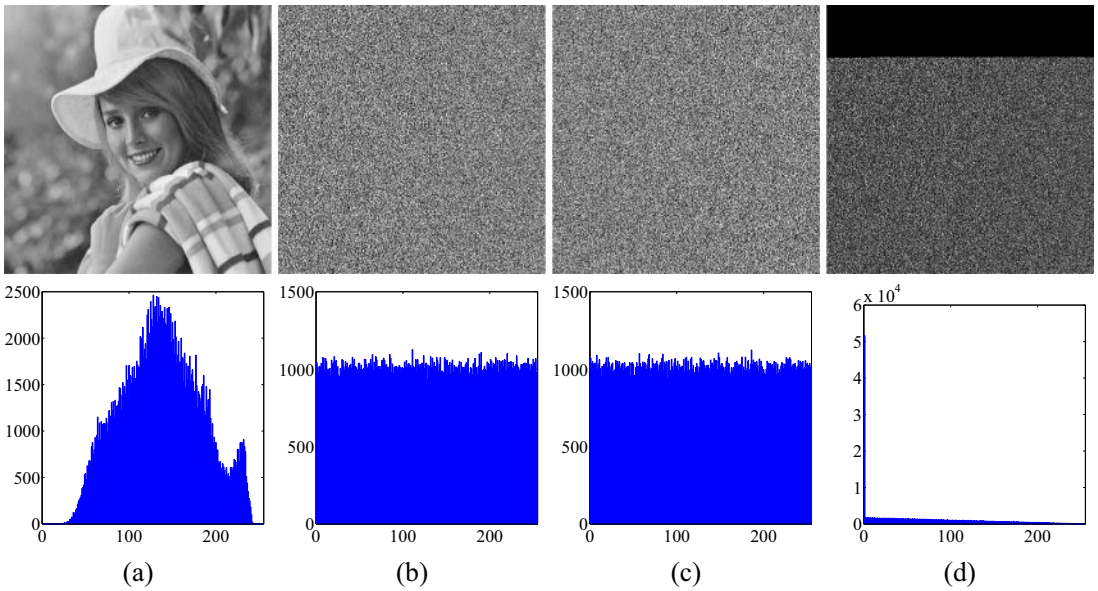

Fig. 9. Demonstration of the image filtering results with their histograms: (a) original image **P**; (b) one-time image filtering result of **P**; (c) one-time image filtering result of $\mathbf{P}_2$, where $\mathbf{P}_2$ has one bit difference in position (100, 84) with **P**; (d) the difference between (b) and (c).

$$= (\mathbf{C}_{i,j} - \mathbf{T}_{1,1}\mathbf{W}_{1,1} - \mathbf{T}_{1,2}\mathbf{W}_{1,2} - \mathbf{T}_{1,3}\mathbf{W}_{1,3} - \mathbf{T}_{2,1}\mathbf{W}_{2,1} - \mathbf{T}_{2,2}\mathbf{W}_{2,2} - \mathbf{T}_{2,3}\mathbf{W}_{2,3} - \mathbf{T}_{3,1}\mathbf{W}_{3,1} - \mathbf{T}_{3,2}\mathbf{W}_{3,2}) \mod F. \quad (13)$$

It is noticed that the processing order in the inverse image filtering is opposite to that in the forward operation.

Fig. 9 shows an image example of image filtering. Fig. 9(b) is the one-time image filtering result of Fig. 9(a). Fig. 9(c) is the one-time image filtering result of a new image, which is obtained by changing one bit of the pixel with position (100, 84) in Fig. 9(a). Fig. 9(d) shows the difference between Fig. 9(b) and (c). This demonstrates that the image filtering can spread the little change of a pixel to all the pixels behind it. After four encryption rounds, these little change can be spread to the entire image. Thus, the proposed IC-BSIF can achieve the diffusion property and has strong ability of resisting chosen-plaintext attack.

4. Simulation results

This section provides the simulation results of IC-BSIF in the MATLAB 2012b environment. Most of the used images in the experiments are from the CVG-UGR image database.

The proposed IC-BSIF can be directly applied to digital images with all kinds of data formats. In our experiments, we use IC-BSIF to encrypt a large number of images with different data formats and some representative simulation results are displayed in Figs. 10 and 11. Fig. 10 shows the simulation results of five grayscale images with different data formats and Fig. 11 demonstrates the simulation results of five color images. These plain-images have many patterns or textures that make them hard to be processed. However, the encryption process can transform them into random-like cipher-images. Even the plain-images have many patterns, the pixels of their corresponding cipher-images are uniformly distributed and attackers cannot get any useful information by observing their pixel distributions. The decryption process is able to accurately reconstruct the original images.

Because the inverse image filtering in the decryption process can spread little change of the cipher-images to almost all the pixels of the decrypted images, the decryption process cannot completely recover the original images if some pixel values of the corresponding cipher-images are modified. Thus, the proposed encryption algorithm cannot defend some kinds of attacks, such the lossy compression, rotation and cropping attacks.

5. Security analysis

This section analyzes the security of IC-BSIF from the following four aspects: secret key, randomness, differential attack and adjacent pixel correlation. The test images are selected from BOWS-2 image database and several typical encryption schemes are used as the reference schemes: HZPC [11], LLZ [16], WNA [25], WYJN [26], ZBC1 [32], ZBC2 [33], CMC [4], FLMLC [8] and WZNS [27].

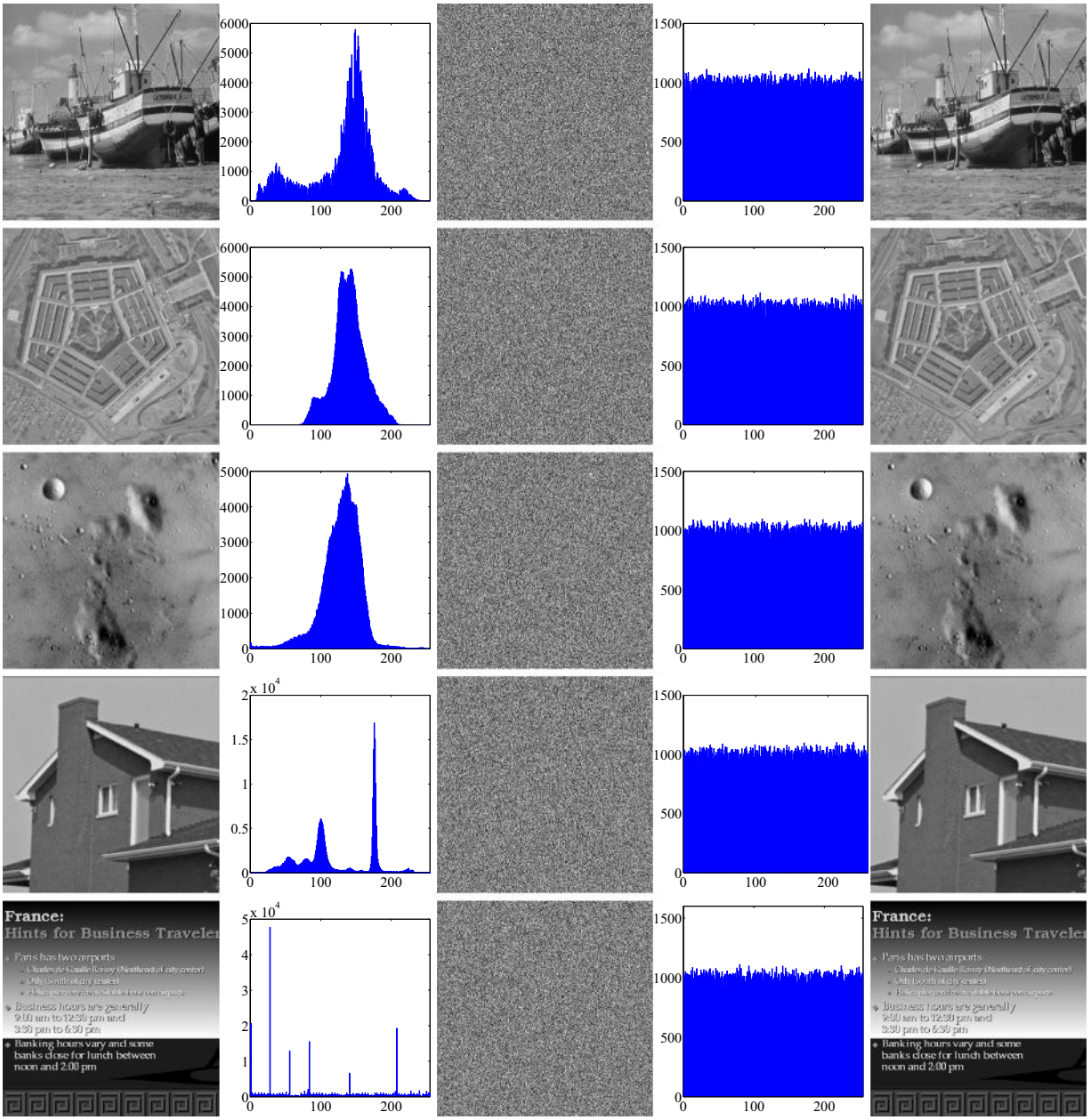

Fig. 10. Simulation results of five grayscale images: (a) plain-images, whose filenames from top to bottom are "boat.png", "pentagon.jpg", "moon.tif", "house.bmp" and "france.pgm"; (b) histograms of (a); (c) encryption results of (a); (d) histograms of (c); (e) decryption results of (c).

5.1. Secret key analysis

The key security is the most concerned issue for an encryption scheme. On one hand, the key space should be large enough to defend the brute-force attack. The key space of IC-BSIF is 2^{256}, which has a proper size [1]. On the other hand, the encryption scheme should be extremely sensitive with the key's change. This means that when encrypting (decrypting) an identical plain-image (cipher-image) with two slightly different secret keys, the obtained two cipher-images (decrypted images) should be totally different.

To test the key sensitivity of IC-BSIF in encryption and decryption processes, we first randomly generate a secret key $\mathbf{K}_1$, and then change one bit of $\mathbf{K}_1$ to obtain two different keys: $\mathbf{K}_2$ and $\mathbf{K}_3$. The three secret keys $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$ are listed as

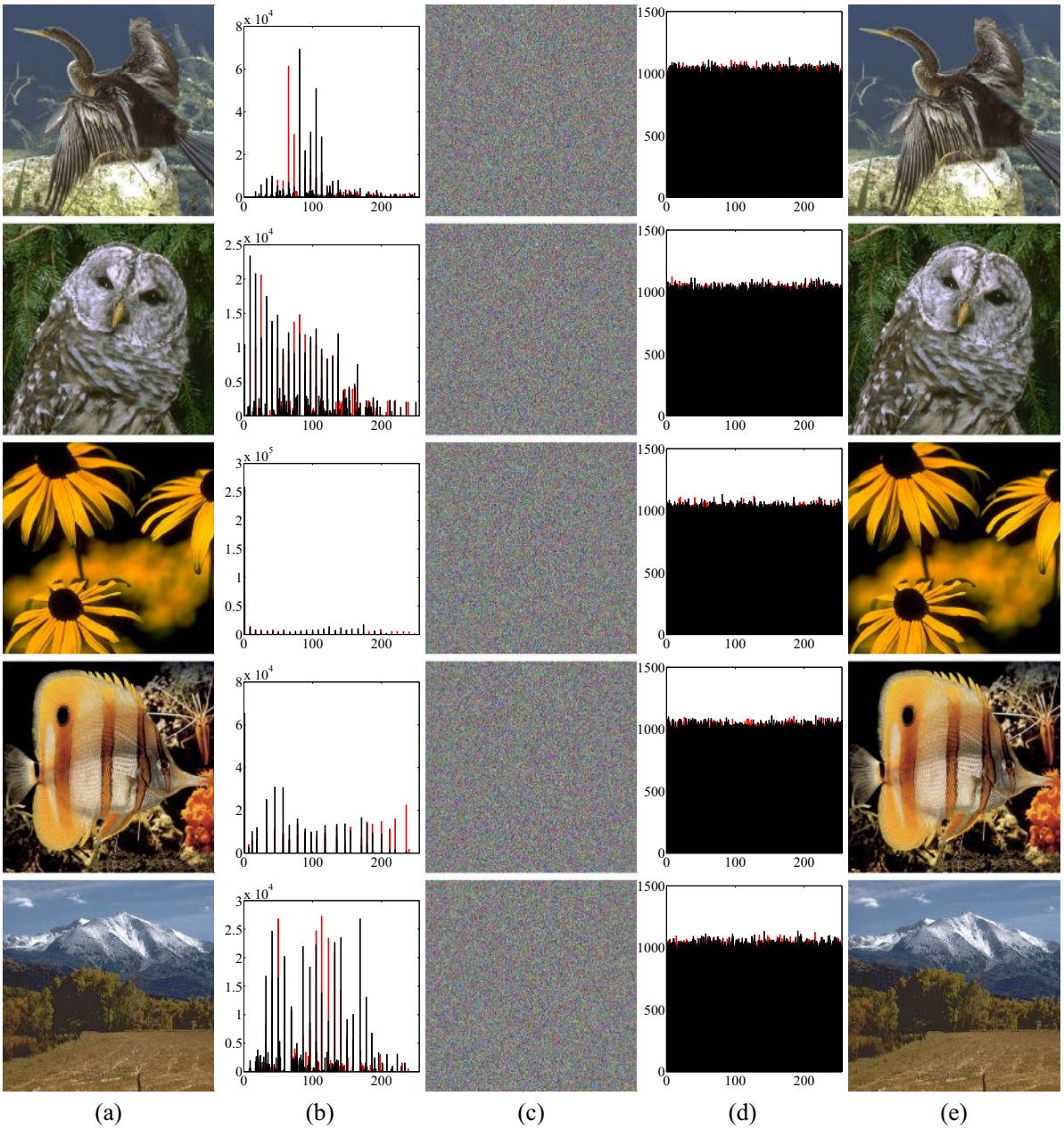

Fig. 11. Simulation results of color images from CVG-UGR image database: (a) plain-images, whose filenames from top to bottom are "anhinga.pgm", "bardowl.pgm", "blakeyed.pgm", "butfish1.pgm" and "colomtn.pgm"; (b) histograms of (a); (c) encryption results of (a); (d) histograms of (c); (e) decryption results of (c).

followings,

$$\mathbf{K}_1 = A49AD500090263914ACA157CE1AA2324A1344C090FAB82F067499CB982A503A9,$$
$$\mathbf{K}_2 = B49AD500090263914ACA157CE1AA2324A1344C090FAB82F067499CB982A503A9,$$
$$\mathbf{K}_3 = E49AD500090263914ACA157CE1AA2324A1344C090FAB82F067499CB982A503A9.$$

Fig. 12 shows the key sensitivity analysis results in encryption process. As can be observed from the figures that the three cipher-images encrypted using $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$ are totally different, and their differences are shown in Fig. 12(e), (f) and (g). This means that the proposed IC-BSIF is quite sensitive with its secret key in encryption process. Fig. 13 shows the results of decrypting an identical cipher-image using $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$, respectively. Only the correct key can accurately

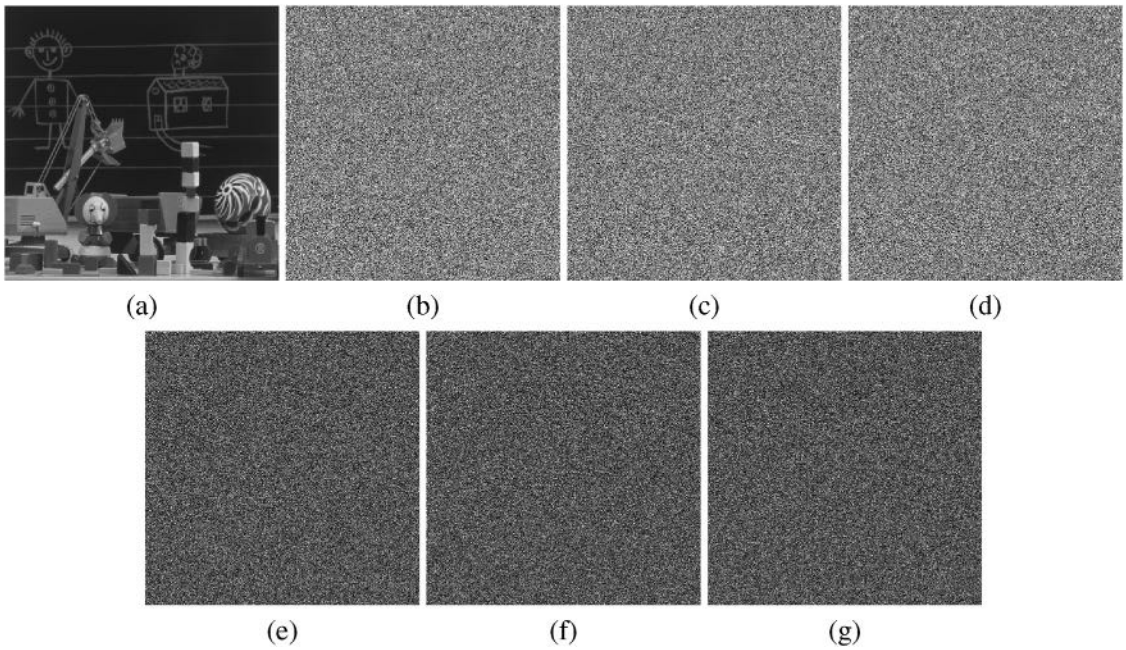

Fig. 12. Encryption key sensitivity analysis: (a) the plain-image $\mathbf{P}$; (b) $\mathbf{C}_1 = \mathtt{Enc}(\mathbf{P}, \mathbf{K}_1)$; (c) $\mathbf{C}_2 = \mathtt{Enc}(\mathbf{P}, \mathbf{K}_2)$; (d) $\mathbf{C}_3 = \mathtt{Enc}(\mathbf{P}, \mathbf{K}_3)$; (e) $|\mathbf{C}_1 - \mathbf{C}_2|$; (f) $|\mathbf{C}_1 - \mathbf{C}_3|$; (g) $|\mathbf{C}_2 - \mathbf{C}_3|$.

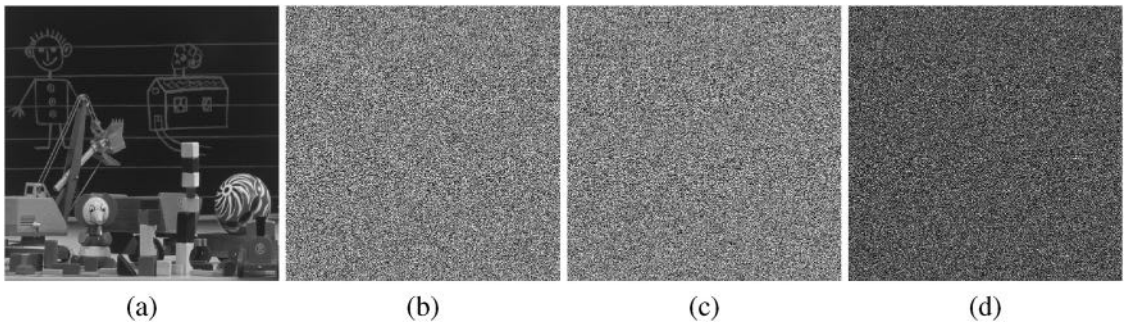

Fig. 13. Decryption key sensitivity analysis: (a) $\mathbf{D}_1 = \mathtt{Dec}(\mathbf{C}_1, \mathbf{K}_1)$; (b) $\mathbf{D}_2 = \mathtt{Dec}(\mathbf{C}_1, \mathbf{K}_2)$; (c) $\mathbf{D}_3 = \mathtt{Dec}(\mathbf{C}_1, \mathbf{K}_3)$; (d) $|\mathbf{D}_2 - \mathbf{D}_3|$. ($\mathbf{C}_1$ is the cipher-image in Fig. 12(b)).

recover the original image (see Fig. 13(a)). Secret keys with one bit difference result in totally different decrypted images (see Fig. 13(d)). This demonstrates that the secret keys of IC-BSIF are extremely sensitive in decryption process.

5.2. Randomness of cipher-image

The cipher-images of a cryptosystem are expected to randomly distribute to defend statistical attacks. Their randomness can be measured by the National Institute of Standards and Technology (NIST) SP800-22 Statistical Test Suite [12,19]. The NIST SP800-22 Statistical Test Suite has 15 different sub-tests that aim to find different types of non-randomness area in a long binary sequence. It uses a set of binary sequences as the input. According to the recommendation in [12], the used significance level α is set as 0.01 and the size of binary sequences s should be not smaller than the inverse of α, 120 in our experiment. Each sub-test generates a $P-value$ for all the binary sequences and obtains a $p-value$ for each binary sequence as well.

The empirical results of each sub-test can be interpreted in three ways: the $P-value$, pass rate and $p-value_T$ interpretations. The $P-value$ interpretation checks whether the $P-value$s fall into range $[\alpha, 1]$. As $\alpha = 0.01$, $P-value$ falling into range [0.01, 1] is considered to pass the corresponding sub-test. The pass rate interpretation is to count the pass proportion of all the test samples. The minimum pass rate T is defined as

$$T = \hat{p} - 3\sqrt{\frac{\hat{p}(1-\hat{p})}{s}},$$

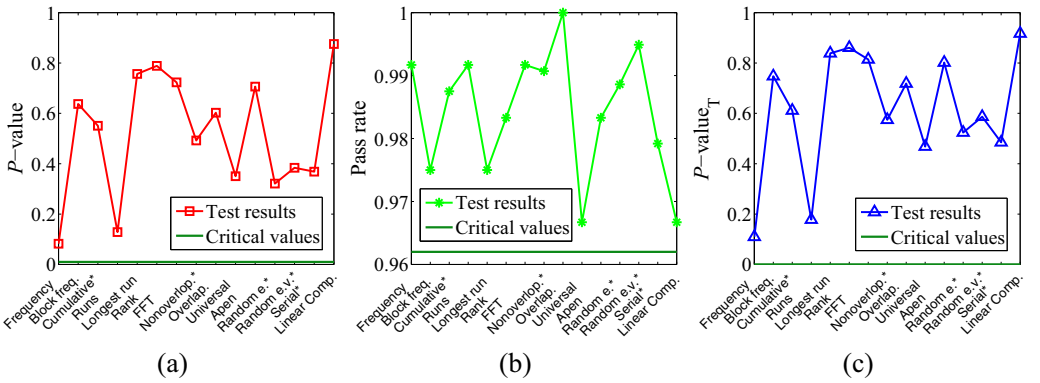

Fig. 14. NIST SP800-22 test results of cipher-images encrypted by IC-BSIF. (a) $P-value$ interpretation; (b) pass rate interpretation; (c)$p-value_T$ interpretation. Note that the symbol * means average value of multiple tests.

where $\hat{p}=1-\alpha$. As $\alpha=0.01$ and $s=120$, we can obtain that $T=0.9628$. The $p-value_T$ interpretation is to calculate the distribution of the $p-values$. For each sub-test, the generated $p-values$ are expected to uniformly distribute in the range (0, 1). The $p-value_T$ is calculated by $p\text{-}value_T=\texttt{igamc}(9/2,\chi^2/2)$, where $\texttt{igamc}(\cdot\ )$ is the incomplete gamma function, and χ^2 is defined as

$$\chi^2=\sum_{i=1}^{10}\frac{(F_i-s/10)^2}{s/10},$$

where uniformly dividing the range (0, 1) into 10 sub-intervals and F_i is the frequency of occurrence of $p-value$ in i-th sub-interval. The test binary sequences pass the sub-test if $p-value_T\geq 0.0001$.

We chose 120 images (filenames are from "1" to "120") from the BOWS-2 image database to do the experiment. These images are first encrypted by IC-BSIF and the obtained cipher-images are then decomposed into binary sequences. The binary numbers from one cipher-image are used as one binary sequence. As all the images are of size 512×512 and each pixel is represented by 8 bits, the length of a binary sequence is $512\times 512\times 8=2097152$. Thus, 120 binary sequences of length 2,097,152 are tested by NIST SP800-22 test suite and the interpretation results are shown in Fig. 14. The results show that the 120 cipher-images encrypted by IC-BSIF can pass all the $P-value$, pass rate and $p-value_T$ interpretations for all the 15 sub-tests. This means than IC-BSIF can encrypt images into cipher-images with high randomness.

5.3. Ability of resisting differential attack

The differential attack, as a kind of chosen-plaintext attack, investigates how the differences in plain-images can affect the corresponding cipher-images in an encryption scheme. It traces the differences and tries to find the connections between plain-images and cipher-images. The number of pixel change rate (NPCR) and uniform average change intensity (UACI) can be used to measure the ability of resisting differential attack. Suppose $\mathbf{P}_1$ and $\mathbf{P}_2$ are two plain-images with one bit difference, $\mathbf{C}_1$ and $\mathbf{C}_2$ are their corresponding cipher-images encrypted by an identical secret key, NPCR is defined as

$$\texttt{NPCR}(\mathbf{C}_1,\mathbf{C}_2)=\sum_{i,j}\frac{\mathbf{A}(i,j)}{G}\times 100\%,$$

and UACI is described as

$$\texttt{UACI}(\mathbf{C}_1,\mathbf{C}_2)=\sum_{i,j}\frac{|\mathbf{C}_1(i,j)-\mathbf{C}_2(i,j)|}{T\times G}\times 100\%,$$

where G denotes the total number of pixel, T demonstrates the largest allowed pixel value, and $\mathbf{A}$ represents the difference between $\mathbf{C}_1$ and $\mathbf{C}_2$, which is defined as

$$\mathbf{A}(i,j)=\begin{cases}0, & \text{if}\quad \mathbf{C}_1(i,j)=\mathbf{C}_2(i,j),\\ 1, & \text{if}\quad \mathbf{C}_1(i,j)\neq\mathbf{C}_2(i,j).\end{cases}$$

Recently, strict NPCR and UACI critical values were developed in [24]. For a given significance level α, the corresponding critical NPCR score $\mathcal{N}_\alpha^*$ can be calculated by

$$\mathcal{N}_\alpha^*=\frac{G-\Phi^{-1}(\alpha)\sqrt{G/T}}{G+1}.$$

Table 1
NPCR results of different image encryption schems with the significance level $\alpha = 0.05$.

Image Size	File name	Ciphertext images						
		HZPC	ZBC1	WNA	LLZ	WYJN	ZBC2	IC-BSIF
	cuadrado3	99.6582	99.5361	99.6094	49.8230	99.5422	99.7864	99.7253
	nature8	98.2300	99.6155	99.5850	99.5789	99.5178	99.8413	99.7253
128×128	p2	99.3286	99.6155	99.5605	99.5667	99.6155	99.1272	99.5667
	Rombos2	99.6094	99.4385	99.5911	99.5972	99.6521	99.4446	99.5667
≥ 99.5292	thuodd	99.6582	99.6094	99.6765	99.6399	99.6460	99.0723	99.5667
	tipo4_d	99.6521	99.6094	99.6460	99.6033	99.5544	99.4202	99.6765
	Triangulo_de_ang	99.6643	99.5789	99.6582	99.6399	99.5972	99.4568	99.6765
	4.1.06	99.6750	99.5911	99.6094	49.8230	99.5667	99.5422	99.6216
	5.2.10	99.4583	99.6445	99.6002	49.8001	99.5911	99.8047	99.6292
256×256	fig31_10	99.6536	99.5987	99.5575	99.6231	99.5895	99.4278	99.5956
	fiore	99.5697	99.5880	99.6338	49.7894	99.6063	99.5605	99.6353
≥ 99.5693	montage	99.5773	99.6246	99.5575	49.8230	99.5773	99.7971	99.5880
	pallon	99.6399	99.6017	99.6033	99.6292	99.6078	99.6262	99.6429
	papav	99.5941	99.5789	99.6124	49.8276	99.6414	99.4308	99.5895
	5.2.09	99.5960	99.6048	99.6197	99.6063	99.6056	99.7059	99.6048
	7.1.02	99.6105	99.5968	99.6010	99.6170	99.6166	99.7898	99.6178
512×512	aerial2	99.6281	99.6048	99.6185	99.6483	99.6105	99.3973	99.6044
	bike	99.6284	99.6075	99.6296	99.6101	99.5953	99.6105	99.6044
≥ 99.5893	blackb	99.6147	99.5861	99.6140	49.8268	99.6212	99.6086	99.6288
	cmpndd	99.6098	99.6033	99.5865	49.8028	99.6262	99.5712	99.5953
	france	99.6117	99.6071	99.5937	49.8039	99.5949	99.7154	99.6159
Pass rate		18/21	19/21	18/21	12/21	19/21	10/21	21/21

If the obtained NPCR score is not smaller than $\mathcal{N}^*_\alpha$, the encryption scheme is considered to have strong ability of resisting differential attack. The corresponding critical UACI scores $(\mathcal{U}^{*-}_\alpha, \mathcal{U}^{*+}_\alpha)$ can be calculated by

$$\begin{cases} \mathcal{U}^{*-}_\alpha = \mu_\mathcal{U} - \Phi^{-1}(\alpha/2)\sigma_\mathcal{U}, \\ \mathcal{U}^{*+}_\alpha = \mu_\mathcal{U} + \Phi^{-1}(\alpha/2)\sigma_\mathcal{U}, \end{cases}$$

where

$$\mu_\mathcal{U} = \frac{G+2}{3G+3},$$

$$\sigma_\mathcal{U} = \frac{(G+2)(G^2+2G+3)}{18(G+1)^2GT}.$$

An encryption scheme is considered to pass the UACI test if the obtained UACI score falls into range $(\mathcal{U}^{*-}_\alpha, \mathcal{U}^{*+}_\alpha)$.

Our experiment randomly selected 21 grayscale images from the CVG-UGR image database to do the test. Among these 21 images, the numbers of images of size 128×128, 256×256 and 512×512 are equal. According to the recommendation in [24], we set the significance level $\alpha = 0.05$. Then, for image of size 128×128, $\mathcal{N}^*_{0.05} = 99.5292\%$ and $(\mathcal{U}^{*-}_{0.05}, \mathcal{U}^{*+}_{0.05}) = (33.1012\%, 33.8259\%)$; for image of size 256×256, $\mathcal{N}^*_{0.05} = 99.5693\%$ and $(\mathcal{U}^{*-}_{0.05}, \mathcal{U}^{*+}_{0.05}) = (33.2824\%, 33.6447\%)$; for image of size 512×512, $\mathcal{N}^*_{0.05} = 99.5893\%$ and $(\mathcal{U}^{*-}_{0.05}, \mathcal{U}^{*+}_{0.05}) = (33.3730\%, 33.5541\%)$. Tables 1 and 2 demonstrate the NPCR and UACI scores of different image encryption schemes. From the two tables, we can observe that the proposed IC-BSIF can pass both NPCR and UACI tests for all the 21 test images. It has the highest pass rates compared with other encryption schemes. Thus, IC-BSIF can achieve strong ability of resisting differential attack.

5.4. Adjacent pixel correlation

A natural image may have strong correlations between adjacent pixels. An efficient image encryption scheme should be able to weaken these strong correlations. The adjacent pixel correlation can be quantitively calculated by

$$AC = \frac{E[(\mathbf{X} - \mu_\mathbf{X})(\mathbf{Y} - \mu_\mathbf{Y})]}{\sigma^2}, \tag{14}$$

where $\mathbf{X}$ is a pixel sequence of the image and $\mathbf{Y}$ is another pixel sequence, in which each pixel in $\mathbf{Y}$ is the adjacent pixel of $\mathbf{X}$ along the horizontal, vertical or diagonal direction, μ is mathematical expectation and σ is the standard derivation. It is obvious that the AC value is in the range $[-1, 1]$. If pixel sequences $\mathbf{X}$ and $\mathbf{Y}$ have a strong correlation, their AC value approaches to 1 or -1; If they have a weak correlation, their AC value closes to 0. Therefore, smaller absolute AC value means a weaker correlation of two pixel sequences.

Table 2
UACI results of different image encryption schemes with the significance level $\alpha = 0.05$.

Image Size	File name	Ciphertext images						
		HZPC	ZBC1	WNA	LLZ	WYJN	ZBC2	IC-BSIF
	cuadrado3	33.4438	33.7013	33.6535	17.1931	33.4076	33.4545	33.5900
	nature8	29.8763	33.2757	33.4870	33.4609	33.6193	32.8481	33.7523
128×128	p2	32.6957	33.8692	33.4449	33.4951	33.3605	33.3084	33.4534
	Rombos2	33.6854	33.5074	33.4706	33.4745	33.2754	32.7508	34.4626
33.1012/33.8259	thuodd	32.9288	33.8620	33.6911	34.1045	33.4375	33.1221	33.5137
	tipo4_d	33.5302	33.6632	33.2958	33.1969	33.2956	32.1341	33.4832
	Triangulo_de_ang	33.9034	33.7655	33.0439	33.3442	33.4668	32.7604	33.6306
	4.1.06	33.4282	33.2280	33.4413	17.0287	33.4610	33.7711	33.4743
	5.2.10	32.8332	33.4139	33.3319	16.7253	33.4378	33.5547	33.2897
256×256	fig31_10	33.5212	33.6864	33.2168	33.6405	33.5460	33.2172	33.4226
	fiore	33.1129	33.2512	33.5288	16.8842	33.6462	33.5994	33.4377
33.2824/33.6447	montage	32.9954	33.5491	33.5684	17.0063	33.6242	33.0124	33.5017
	pallon	33.5965	33.2897	33.3414	33.7884	33.4349	32.8465	33.5765
	papav	33.2034	33.5483	33.3393	16.7986	33.4452	33.5885	33.5257
	5.2.09	33.5230	33.5176	33.5115	33.4825	33.5012	33.4902	33.5107
	7.1.02	33.5149	33.4418	33.4726	33.5864	33.4369	33.4940	33.4397
512×512	aerial2	33.4768	33.4616	33.4617	33.6219	33.4678	32.9591	33.4328
	bike	33.5588	33.5920	33.4384	33.4566	33.4845	33.4077	33.5277
33.3730/33.5541	blackb	33.5144	33.4438	33.4138	16.8010	33.4731	33.4916	33.4686
	cmpndd	33.4595	33.5532	33.4341	16.7550	33.4657	33.3282	33.4992
	france	33.3785	33.5048	33.5254	16.7867	33.4372	33.6441	33.4768
Pass rate		12/21	15/21	19/21	8/21	20/21	10/21	21/21

Table 3
AC results of the "Lena" image and its cipher-images encrypted by different image encryption schemes.

Encryption schemes	Horizontal	Vertical	Diagonal
"Lena" image	0.9400	0.9709	0.9710
CMC	−0.00024	−0.24251	0.23644
LLZ	0.0127	−0.0190	−0.0012
FLMLC	0.0368	0.0392	0.0068
WYJN	−0.0002150	0.0014913	0.0040264
ZBC1	−0.0054	0.0045	0.0031
WZNS	0.0053365	−0.0027616	0.0016621
HZPC	−0.0006259	0.0013210	−0.0020946
WNA	−0.0007233	0.0012893	−0.0003565
IC-BSIF	**0.0033455**	**−0.0008454**	**−0.0002044**

Table 3 shows the AC values of the "Lena" image and its cipher-images encrypted by different image encryption schemes. For CMC, LLZ, FLMLC, WYJN, ZBC1 and WZNS, we directly reference their AC values reported in [27]; for other algorithms, we implemented them and calculated their AC values. The test results show that IC-BSIF has smaller absolute AC values than LLZ, FLMLC, ZBC1 and WZNS in all the three directions, and outperforms CMC, WYJN, HZPC and WNA in two directions.

Fig. 15 plots the adjacent pixels of the plain-images and their corresponding cipher-images encrypted by IC-BSIF. First, encrypt 10 images from BOWS-2 image database (filenames are from "1" to "10") using IC-BSIF. Then, randomly select 1000 pixels from every plain-image and its cipher-image and plot the selected 1000 pixels with their adjacent pixels along the horizontal, vertical and diagonal directions. For each figure in Fig. 15, the X-axis denotes the 10 images, and Y-Z plane plots theses pixel pairs. The adjacent pixel pairs in the plain-images mostly distribute on or nearby the diagonal lines of the Y-Z plane. This means that the adjacent pixels of these plain-images have strong correlations. The adjacent pixel pairs in the cipher-images randomly distribute in the whole data range, which means that a pixel has a weak correlation with its adjacent pixels in the cipher-images. Thus, IC-BSIF can efficiently weakness the strong correlations between adjacent pixels.

6. Conclusion

This paper first presented the concept of encrypting an image using image filtering, and then proposed an image cipher using block-based scrambling and image filtering, named IC-BSIF. The block-based scrambling can separate pixels in an image block into different rows and columns, and thus can achieve high efficiency to weaken the strong correlations between adjacent pixels. Using randomly generated masks, the image filtering can blur the image and achieve the diffusion property. Experimental results showed that IC-BSIF can encrypt different kinds of digital images into random-like ones. Security evaluations demonstrated that, compared with several typical encryption schemes, IC-BSIF can achieve better

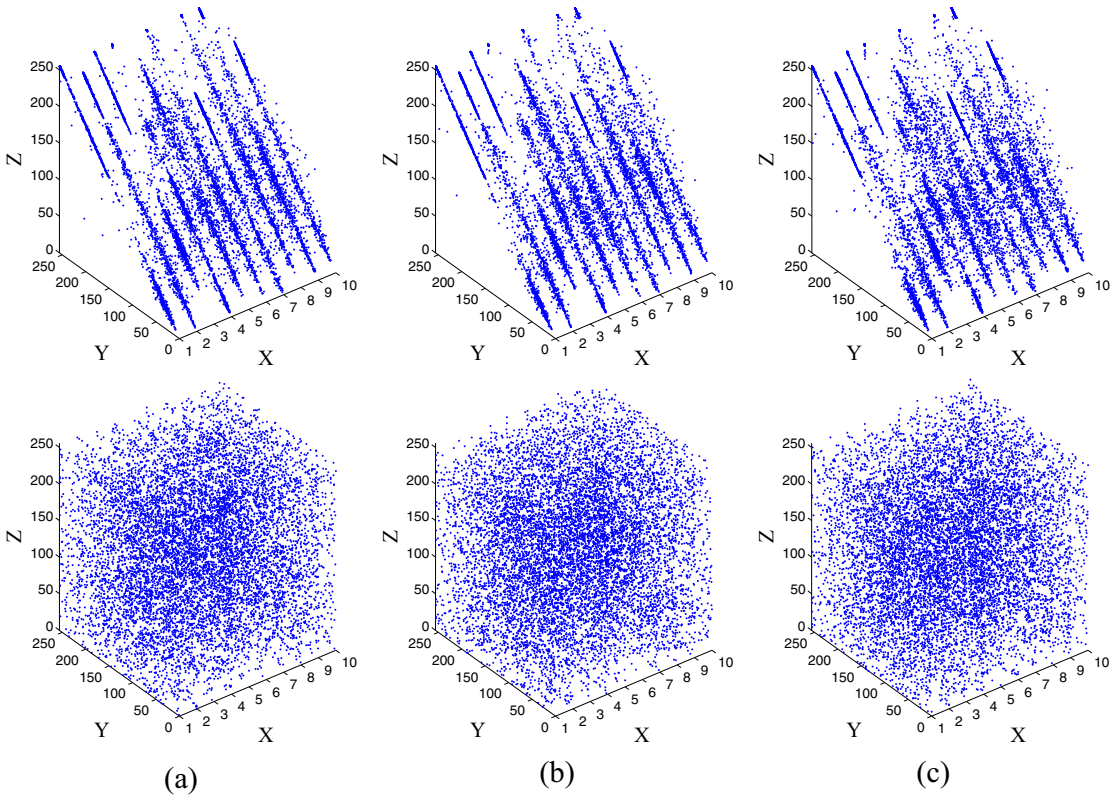

Fig. 15. The adjacent pixel pairs of 10 plain-images and their cipher-images encrypted by IC-BSIF. The top row plots the pixel pairs of plain-images while the bottom row plots that of cipher-images along the (a) horizontal, (b) vertical and (c) diagonal directions. In each figure, the X-axis denotes the index of the image and Y-Z plane plots the pixel pairs.

performance. Because this is the first time that image filtering has been used to do encryption, the developed encryption scheme still has some limitations, including incapable of resisting lossy compression, rotation and cropping attacks. The performance of the proposed encryption scheme can be significantly improved by solving these limitations.

Acknowledgement

This work was supported in part by the Macau Science and Technology Development Fund under Grant FDCT/016/2015/A1 and by the Research Committee at University of Macau under Grants MYRG2014-00003-FST and MYRG2016-00123-FST.

References

[1] G. Alvarez, S. Li, Some basic cryptographic requirements for chaos-based cryptosystems, Int. J. Bifurcation Chaos 16 (8) (2006) 2129–2151.
[2] L. Bao, Y. Zhou, Image encryption: generating visually meaningful encrypted images, Inf. Sci. 324 (2015) 197–207.
[3] X. Chai, Z. Gan, Y. Chen, Y. Zhang, A visually secure image encryption scheme based on compressive sensing, Signal Process. 134 (2017) 35–51.
[4] G. Chen, Y. Mao, C.K. Chui, A symmetric image encryption scheme based on 3D chaotic cat maps, Chaos, Solitons Fractals 21 (3) (2004) 749–761.
[5] J.-X. Chen, Z.-L. Zhu, C. Fu, H. Yu, L.-B. Zhang, An efficient image encryption scheme using gray code based permutation approach, Opt. Lasers Eng. 67 (2015) 191–204.
[6] J.-X. Chen, Z.-L. Zhu, C. Fu, H. Yu, Y. Zhang, Reusing the permutation matrix dynamically for efficient image cryptographic algorithm, Signal Process. 111 (2015) 294–307.
[7] J. Fridrich, Symmetric ciphers based on two-dimensional chaotic maps, Int. J. Bifurcation Chaos 8 (06) (1998) 1259–1284.
[8] C. Fu, B.-B. Lin, Y.-S. Miao, X. Liu, J.-J. Chen, A novel chaos-based bit-level permutation scheme for digital image encryption, Opt. Commun. 284 (23) (2011) 5415–5423.
[9] T. Habutsu, Y. Nishio, I. Sasase, S. Mori, A secret key cryptosystem by iterating a chaotic map, in: Advances in Cryptology-EUROCRYPT'91, Springer, 1991, pp. 127–140.
[10] Z. Hua, Y. Zhou, Image encryption using 2D logistic-adjusted-Sine map, Inf. Sci. 339 (2016) 237–253.
[11] Z. Hua, Y. Zhou, C.-M. Pun, C.L.P. Chen, 2D Sine logistic modulation map for image encryption, Inf. Sci. 297 (2015) 80–94.
[12] L.E.B. III, et al., SP 800-22 Rev. 1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, Technical Report, SP 800-22, National Institute of Standards & Technology, 2010.
[13] C. Li, Cracking a hierarchical chaotic image encryption algorithm based on permutation, Signal Process. 118 (2016) 203–210. http://dx.doi.org/10.1016/j.sigpro.2015.07.008.
[14] C. Li, Y. Liu, T. Xie, M.Z.Q. Chen, Breaking a novel image encryption scheme based on improved hyperchaotic sequences, Nonlin. Dyn. 73 (3) (2013) 2083–2089. http://dx.doi.org/10.1007/s11071-013-0924-6.

[15] C. Li, K.-T. Lo, Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks, Signal Process. 91 (4) (2011) 949–954. http://dx.doi.org/10.1016/j.sigpro.2010.09.014.
[16] X. Liao, S. Lai, Q. Zhou, A novel image encryption algorithm based on self-adaptive wave transmission, Signal Process. 90 (9) (2010) 2714–2722.
[17] Z. Lin, S. Yu, C. Li, J. Lü, Q. Wang, Design and smartphone-based implementation of a chaotic video communication scheme via wan remote transmission, Int. J. Bifurcation Chaos 26 (9) (2016). http://dx.doi.org/10.1142/S0218127416501583. Article number 1650158.
[18] C. Ling, X. Wu, S. Sun, A general efficient method for chaotic signal estimation, IEEE Trans. Signal Process. 47 (5) (1999) 1424–1428.
[19] F. Pareschi, R. Rovatti, G. Setti, On statistical tests for randomness included in the NIST SP800-22 test suite and based on the binomial distribution, IEEE Trans. Inf. Forensics Secur. 7 (2) (2012) 491–505.
[20] A. Shakiba, M.R. Hooshmandasl, M.A. Meybodi, Cryptanalysis of multiplicative coupled cryptosystems based on the chebyshev polynomials, Int. J. Bifurcation Chaos 26 (7) (2016), doi:10.1142/S0218127416501121. Article number 1650112.
[21] E. Solak, C. Cokal, O.T. Yildiz, T. Biyikoglu, Cryptanalysis of Fridrich's chaotic image encryption, Int. J. Bifurcation Chaos 20 (5) (2010) 1405–1413.
[22] Q. Wang, S. Yu, C. Li, J. Lü, X. Fang, C. Guyeux, J.M. Bahi, Theoretical design and FPGA-based implementation of higher-dimensional digital chaotic systems, IEEE Trans. Circuits Syst I-Regular Pap. 63 (3) (2016) 401–412. http://dx.doi.org/10.1109/TCSI.2016.2515398.
[23] X. Wang, L. Teng, X. Qin, A novel colour image encryption algorithm based on chaos, Signal Process. 92 (4) (2012) 1101–1108.
[24] Y. Wu, J.P. Noonan, S. Agaian, NPCR and UACI randomness tests for image encryption, Cyber J. (2011) 31–38.
[25] Y. Wu, J.P. Noonan, S. Agaian, A wheel-switch chaotic system for image encryption, in: Proceedings of 2011 International Conference on System Science and Engineering (ICSSE), 2011, pp. 23–27.
[26] Y. Wu, G. Yang, H. Jin, J.P. Noonan, Image encryption using the two-dimensional logistic chaotic map, J. Electron. Imaging 21 (1) (2012). art. no. 013014
[27] Y. Wu, Y. Zhou, J.P. Noonan, S. Agaian, Design of image cipher using latin squares, Inf. Sci. 264 (0) (2014) 317–339.
[28] L.Y. Zhang, C. Li, K.-W. Wong, S. Shu, G. Chen, Cryptanalyzing a chaos-based image encryption algorithm using alternate structure, J. Syst. Softw. 85 (9) (2012) 2077–2085.
[29] Y. Zhang, D. Xiao, Y. Shu, J. Li, A novel image encryption scheme based on a linear hyperbolic chaotic system of partial differential equations, Signal Process. Image Commun. 28 (3) (2013) 292–300.
[30] Y. Zhang, D. Xiao, W. Wen, K.-W. Wong, On the security of symmetric ciphers based on DNA coding, Inf. Sci. 289 (2014) 254–261.
[31] Y. Zhang, L.Y. Zhang, Exploiting random convolution and random subsampling for image encryption and compression, Electron. Lett. 51 (20) (2015) 1572–1574.
[32] Y. Zhou, L. Bao, C.L.P. Chen, Image encryption using a new parametric switching chaotic system, Signal Process. 93 (11) (2013) 3039–3052.
[33] Y. Zhou, L. Bao, C.L.P. Chen, A new 1D chaotic system for image encryption, Signal Process. 97 (2014) 172–182.
[34] Y. Zhou, Z. Hua, C.M. Pun, C.L.P. Chen, Cascade chaotic system with applications, IEEE Trans. Cybern. 45 (9) (2015) 2001–2012.
[35] H. Zhu, C. Zhao, X. Zhang, A novel image encryption–compression scheme using hyper-chaos and chinese remainder theorem, Signal Process. Image Commun. 28 (6) (2013) 670–680.
[36] H. Zhu, C. Zhao, X. Zhang, L. Yang, An image encryption scheme using generalized arnold map and affine cipher, Optik-Int. J. Light Electron Opt. 125 (22) (2014) 6672–6677.
[37] Z.-L. Zhu, W. Zhang, K.-W. Wong, H. Yu, A chaos-based symmetric image encryption scheme using a bit-level permutation, Inf. Sci. 181 (6) (2011) 1171–1186.