Information Sciences 480 (2019) 403–419

Contents lists available at ScienceDirect

Information Sciences

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

Cosine-transform-based chaotic system for image encryption

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

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

ARTICLE INFO

Article history:
Received 25 January 2018
Revised 3 December 2018
Accepted 21 December 2018
Available online 22 December 2018

Keywords:
Chaotic system
Chaos-based encryption
Cryptography
Image privacy
Image encryption
Security analysis

ABSTRACT

Chaos is known as a natural candidate for cryptography applications owing to its properties such as unpredictability and initial state sensitivity. However, certain chaos-based cryptosystems have been proven to exhibit various security defects because their used chaotic maps do not have complex dynamical behaviors. To address this problem, this paper introduces a cosine-transform-based chaotic system (CTBCS). Using two chaotic maps as seed maps, the CTBCS can produce chaotic maps with complex dynamical behaviors. For illustration, we produce three chaotic maps using the CTBCS and analyze their chaos complexity. Using one of the generated chaotic maps, we further propose an image encryption scheme. The encryption scheme uses high-efficiency scrambling to separate adjacent pixels and employs random order substitution to spread a small change in the plain-image to all pixels of the cipher-image. The performance evaluation demonstrates that the chaotic maps generated by the CTBCS exhibit substantially more complicated chaotic behaviors than the existing ones. The simulation results indicate the reliability of the proposed image encryption scheme. Moreover, the security analysis demonstrates that the proposed image encryption scheme provides a higher level of security than several advanced image encryption schemes.

© 2018 The Authors. Published by Elsevier Inc.
This is an open access article under the CC BY-NC-ND license.
(http://creativecommons.org/licenses/by-nc-nd/4.0/)

1. Introduction

At present, a large amount of digital information is generated and transmitted through various networks every moment [23]. The digital image is a widely used data format, as it carries information in a visualized manner. Among the images spreading across networks, some are secret images that owners do not want others to access without authorization. A typical example could be the military secret image. Thus, it is extremely important to protect the contents of these secret images. To protect digital images, researchers have developed different types of technologies, such as data hiding [29], watermarking [10], and encryption [9,27]. Among these technologies, image encryption offers the most straightforward method, by transforming meaningful images into unrecognized noise-like ones [13,24,25]. An image encryption algorithm generally follows the well-known confusion-diffusion concept [7,16,44]. The confusion property is achieved by randomly separating adjacent image pixels, while the diffusion property can be obtained by spreading a small change in the plain-image to all

* Corresponding author.
E-mail addresses: huazyum@gmail.com, huazhongyun@hit.edu.cn (Z. Hua), yicongzhou@um.edu.mo (Y. Zhou), huanghejiao@hit.edu.cn (H. Huang).

https://doi.org/10.1016/j.ins.2018.12.048
0020-0255/© 2018 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license.
(http://creativecommons.org/licenses/by-nc-nd/4.0/)

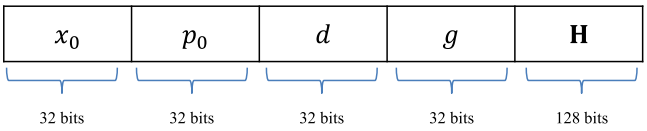

Fig. 2. Secret key structure of LSC-IES.

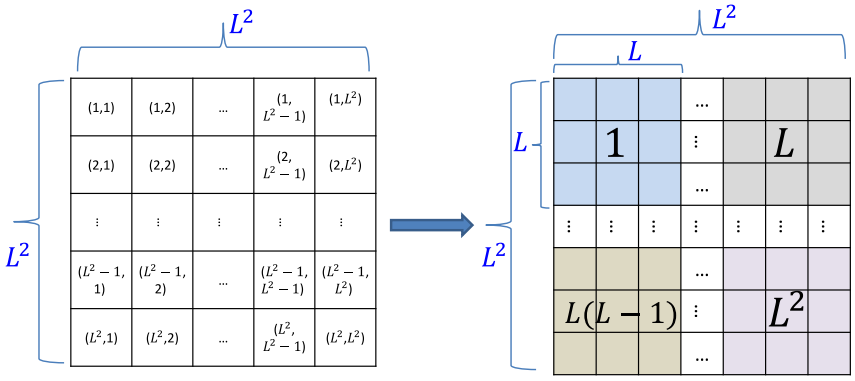

Fig. 3. Concept of high-efficiency scrambling.

as 256 bits, and therefore its key space is 2^{256}. Fig. 2 illustrates the structure of the secure key. It can be observed that it consists of five parts $\mathbf{K} = \{x_0, p_0, d, g, \mathbf{H}\}$. Among these, (x_0, p_0) are the original initial states, d is the perturbation parameter to disturb the original initial states, g is the coefficient of the initial states, and $\mathbf{H} = \{H_1, H_2, H_3, H_4\}$ contains four coefficients for the perturbation parameter. Each of x_0, p_0, d, g, H_1, H_2, H_3, H_4 has a length of 32 bits. The variables x_0, p_0, d are float numbers within [0,1), and each can be obtained from a 32-bit stream by

$$FN = \sum_{i=1}^{32} \mathrm{Bin}_i \times 2^{-i}.$$

The coefficients g, H_1, H_2, H_3, H_4 are integer numbers, each of which can be obtained by

$$IN = \sum_{i=1}^{32} \mathrm{Bin}_i \times 2^{i-1}.$$

Thereafter, the initial states for the four encryption rounds can be calculated as follows:

$$\begin{cases} x_0^{(i)} = (x_0 \times g + d \times H_i) \bmod 1; \\ p_0^{(i)} = (r_0 \times g + d \times H_i) \bmod 1, \end{cases} \tag{3}$$

where $i = \{1, 2, 3, 4\}$. Using the initial states $(x_0^{(i)}, p_0^{(i)})$, the LSC map can generate randomly distributed chaotic sequences for the high-efficiency scrambling and random order substitution.

3.2. High-efficiency scrambling

The high-efficiency scrambling is designed to decorrelate the high correlations of adjacent pixels. It is performed within a square matrix $L^2 \times L^2$, where L is the block size. Supposing that the plain-image to be encrypted has a size of $M \times N$, the block size L is obtained as

$$L = \min\left\{\lfloor\sqrt{M}\rfloor, \lfloor\sqrt{N}\rfloor\right\}, \tag{4}$$

where $\lfloor\alpha\rfloor$ is the floor operation to obtain the greatest integer that is not larger than α. Fig. 3 displays a straightforward diagram of the high-efficiency scrambling. It can be observed that the image to be encrypted is divided into L^2 blocks. The pixels in each row are first randomly permutated into different blocks according to the generated chaotic sequences, following which their positions in each block are also randomly determined using other chaotic sequences. The overall procedures of the high-efficiency scrambling are described as follows.

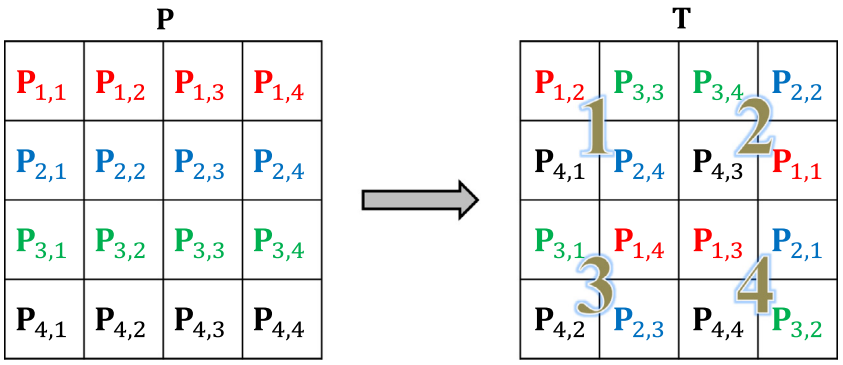

Fig. 4. Demonstration of high-efficiency scrambling procedures using **O** and **Q** in Eq. (5).

- *Step 1*: Truncate an image of size $L^2 \times L^2$ from the plain-image, where the block size L is calculated using Eq. (4).
- *Step 2*: Generate four chaotic sequences: **A**, **B**, **Y**, and **Z**, each with length L^2.
- *Step 3*: Sort **A**, **B**, **Y**, and **Z**, and obtain their corresponding index vectors **IA**, **IB**, **IY**, and **IZ**.
- *Step 4*: Initialize two matrices **O** and **Q** with a size $L^2 \times L^2$. Set each column of **O** as **IA**, and each column of **Q** as **IY**.
- *Step 5*: Shift each column of **O** according to the elements of **IB**, and shift each column of **Q** using the elements of **IZ**.
- *Step 6*: Set row index $i = 1$.
- *Step 7*: For the ith row of the image, move its jth pixel to the $\mathbf{Q}_{\mathbf{O}_{i,j},j}$th position of the $\mathbf{O}_{i,j}$th block.
- *Step 8*: Repeat *step 6* and *step 7* for $i = 2 \sim L^2$.

To provide an improved demonstration of the high-efficiency scrambling procedure, we provide a numeral example to illustrate each step, using a plain-image with a size of 4×4. According to the calculation in Eq. (4), the block size $L = 2$. Firstly, the two index matrices **O** and **Q** are generated from the chaotic sequences. Suppose that the obtained **O** and **Q** values are as follows:

$$\mathbf{O} = \begin{pmatrix} 2 & 1 & 4 & 3 \\ 4 & 2 & 3 & 1 \\ 3 & 4 & 1 & 2 \\ 1 & 3 & 2 & 4 \end{pmatrix}, \qquad \mathbf{Q} = \begin{pmatrix} 3 & 1 & 2 & 4 \\ 4 & 2 & 3 & 1 \\ 1 & 3 & 4 & 2 \\ 2 & 4 & 1 & 3. \end{pmatrix} \tag{5}$$

The index matrix **O** determines which block a pixel permutes to, and **Q** decides the position of a pixel within a block. Fig. 4 illustrates the mapping procedures between the plain-image **P** and scrambling result **T** using the two index matrices **O** and **Q**. The detailed mapping procedures are as follows.

- The 1*st* row of **O** is (2, 1, 4, 3) and its determined positions in **Q** are $(\mathbf{Q}_{2,1}, \mathbf{Q}_{1,2}, \mathbf{Q}_{4,3}, \mathbf{Q}_{3,4}) = (4, 1, 1, 2)$. Then, the four pixels in the 1st row of **P** are permuted to the 4th cell of the 2nd block (position (2, 4)), the 1st cell of the 1st block (position (1, 1)), the 1st cell of the 4th block (position (3, 3)), and the 2nd cell of the 3rd block (position (3, 2)), respectively. That is, $\mathbf{T}_{2,4} = \mathbf{P}_{1,1}$, $\mathbf{T}_{1,1} = \mathbf{P}_{1,2}$, $\mathbf{T}_{3,3} = \mathbf{P}_{1,3}$, and $\mathbf{T}_{3,2} = \mathbf{P}_{1,4}$.
- The 2nd row of **O** is (4, 2, 3, 1) and its determined positions in **Q** are $(\mathbf{Q}_{4,1}, \mathbf{Q}_{2,2}, \mathbf{Q}_{3,3}, \mathbf{Q}_{1,4}) = (2, 2, 4, 4)$. Then, the four pixels in the 2nd row of **P** are permuted to the 2nd cell of the 4th block (position (3, 4)), the 2nd cell of the 2nd block (position (1, 4)), the 4th cell of the 3rd block (position (4, 2)), and the 4th cell of the 1st block (position (2, 2)), respectively. That is, $\mathbf{T}_{3,4} = \mathbf{P}_{2,1}$, $\mathbf{T}_{1,4} = \mathbf{P}_{2,2}$, $\mathbf{T}_{4,2} = \mathbf{P}_{2,3}$, and $\mathbf{T}_{2,2} = \mathbf{P}_{2,4}$.
- The 3rd row of **O** is (3, 4, 1, 2) and its determined positions in **Q** are $(\mathbf{Q}_{3,1}, \mathbf{Q}_{4,2}, \mathbf{Q}_{1,3}, \mathbf{Q}_{2,4}) = (1, 4, 2, 1)$. Then, the four pixels in the 3rd row of **P** are permuted to the 1st cell of the 3rd block (position (3, 1)), the 4th cell of the 4th block (position (4, 4)), the 2nd cell of the 1st block (position (1, 2)), and the 1st cell of the 2nd block (position (1, 3)), respectively. That is, $\mathbf{T}_{3,1} = \mathbf{P}_{3,1}$, $\mathbf{T}_{4,4} = \mathbf{P}_{3,2}$, $\mathbf{T}_{1,2} = \mathbf{P}_{3,3}$, and $\mathbf{T}_{1,3} = \mathbf{P}_{3,4}$.
- The 4th row of **O** is (1, 3, 2, 4) and its determined positions in **Q** are $(\mathbf{Q}_{1,1}, \mathbf{Q}_{3,2}, \mathbf{Q}_{2,3}, \mathbf{Q}_{4,4}) = (3, 3, 3, 3)$. Then, the four pixels in the 4th row of **P** are permuted to the 3rd cell of the 1st block (position (2, 1)), the 3rd cell of the 3rd block (position (4, 1)), the 3rd cell of the 2nd block (position (2, 3)), and the 3rd cell of the 4th block (position (4, 3)), respectively. That is, $\mathbf{T}_{2,1} = \mathbf{P}_{4,1}$, $\mathbf{T}_{4,1} = \mathbf{P}_{4,2}$, $\mathbf{T}_{2,3} = \mathbf{P}_{4,3}$, and $\mathbf{T}_{4,3} = \mathbf{P}_{4,4}$.

Algorithm 1 presents the pseudo-code of the high-efficiency scrambling.

3.3. Image rotation

The image rotation operation involves rotating the image clockwise by 90° As high-efficiency scrambling is performed within the range of $L^2 \times L^2$, and the block size is obtained by Eq. (4), only pixels within the range of $L^2 \times L^2$ can be processed. For an image of size $M \times N$, all of its pixels can be scrambled only when it satisfies $L = \sqrt{M} = \sqrt{N}$. To ensure that all pixels can be shuffled, we rotate the image 90° clockwise following each high-efficiency scrambling.

Algorithm 1 High-efficiency scrambling.

Input: Image $\mathbf{P}$ of size $M \times N$ and initial state $(x_0^{(i)}, y_0^{(i)})$.
Output: Scrambled result $\mathbf{T}$.

Calculate L using Eq. (4);
Generate chaotic sequence $\mathbf{S}$ using LSC map with initial state (x_0^i, y_0^i), where $\mathbf{S} \in \mathbb{R}^{4L^2 \times 1}$;
$\mathbf{A} = \mathbf{S}_{1:L^2}$; $\mathbf{B} = \mathbf{S}_{L^2+1:2L^2}$; $\mathbf{Y} = \mathbf{S}_{2L^2+1:3L^2}$; $\mathbf{Z} = \mathbf{S}_{3L^2+1:4L^2}$;
$[\mathbf{A}', \mathbf{IA}] = \mathtt{Sort}(\mathbf{A})$, where $\mathbf{A}' = \mathbf{A}_{\mathbf{IA}}$;
$[\mathbf{B}', \mathbf{IB}] = \mathtt{Sort}(\mathbf{B})$, where $\mathbf{B}' = \mathbf{B}_{\mathbf{IB}}$;
$[\mathbf{Y}', \mathbf{IY}] = \mathtt{Sort}(\mathbf{Y})$, where $\mathbf{Y}' = \mathbf{Y}_{\mathbf{IY}}$;
$[\mathbf{Z}', \mathbf{IZ}] = \mathtt{Sort}(\mathbf{Z})$, where $\mathbf{Z}' = \mathbf{Z}_{\mathbf{IZ}}$;
Initialize $\mathbf{T} = \mathbf{P}$, $\mathbf{O} \in \mathbb{N}^{L^2 \times L^2}$, $\mathbf{Q} \in \mathbb{N}^{L^2 \times L^2}$.
for $j = 1$ to L^2 **do**
  for $i = 1$ to L^2 **do**
    $m = ((i + \mathbf{IB}(j) - 1) \bmod L^2) + 1$;
    $n = ((i + \mathbf{IZ}(j) - 1) \bmod L^2) + 1$;
    $\mathbf{O}_{i,j} = \mathbf{IA}_m$;
    $\mathbf{Q}_{i,j} = \mathbf{IY}_n$;
  end for
end for
for $i = 1$ to L^2 **do**
  for $j = 1$ to L^2 **do**
    $a = \mathbf{O}_{i,j}$; $b = \mathbf{Q}_{a,j}$;
    $m_1 = \lfloor (a-1)/L \rfloor$; $n_1 = (a-1) \bmod L$;
    $m_2 = \lfloor (b-1)/L \rfloor + 1$; $n_2 = ((b-1) \bmod L) + 1$;
    $x = m_1 \times L + m_2$; $y = n_1 \times L + n_2$;
    $\mathbf{T}(x, y) = \mathbf{P}(i, j)$;
  end for
end for

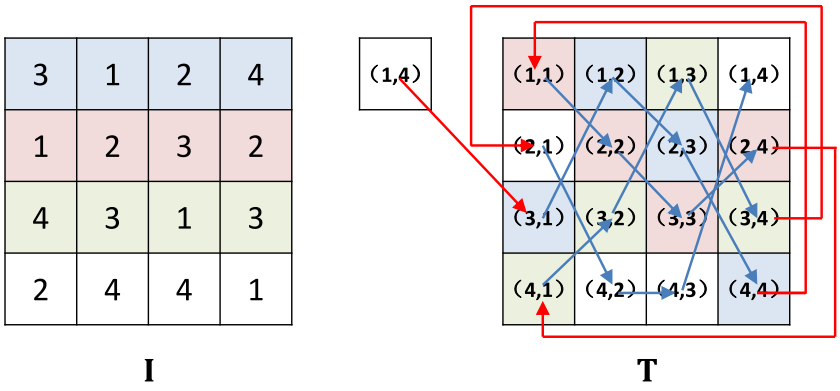

Fig. 5. An example of determining substitution order. Matrix **I** is an index matrix obtained from a chaotic matrix, and **T** is the image to be processed.

3.4. Random order substitution

The substitution changes the pixel values and diffuses a small difference in the plain-image to all pixels of the cipher-image. Most existing substitution algorithms diffuse the change in pixel values row by row and column by column, or by following certain fixed rules. These fixed rules provide valuable information for security attacks. To overcome this drawback and obtain a higher level of security, we design a random order substitution. The substitution order relies on an index matrix, which is generated using the chaotic sequence.

Fig. 5 presents a numeral example of the manner in which to determine the substitution order using an index matrix with an image size 4×4. The matrix **I** is an index matrix obtained by sorting each column of a chaotic matrix, and **T** is the image to be processed. It can be observed that the first row of **I** is (3, 1, 2, 4) and its determined substitution order is (3, 1) → (1, 2) → (2, 3) → (4, 4); the second row of **I** is (1, 2, 3, 2) and its determined substitution order is (1, 1) → (2, 2) → (3, 3) → (2, 4); the third row of **I** is (4, 3, 1, 3) and its determined substitution order is (4, 1) → (3, 2) → (1, 3) → (3, 4); and the fourth row of **I** is (2, 4, 4, 1) and its determined substitution order is (2, 1) → (4, 2) → (4, 3) → (1, 4). The first pixel in each iteration is changed by the final pixel in the previous iteration, and the first pixel in the first iteration is changed by the final pixel in the final iteration. Therefore, we can obtain a substitution circle, as indicated by **T** in Fig. 5.

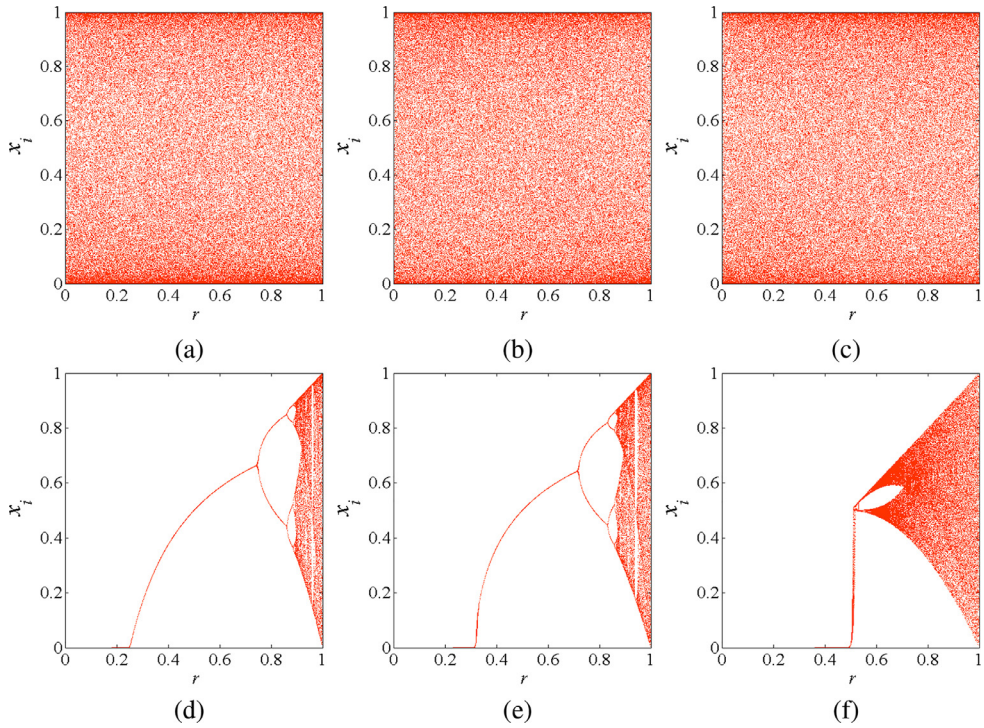

Fig. 6. Bifurcation diagrams of (a) LSC; (b) STC; (c) TLC; (d) Logistic; (e) Sine; and (f) Tent maps.

With the determined substitution order, we can randomly change the current pixel using the previous one and the chaotic sequence. Suppose that the high-efficiency scrambling result $\mathbf{T}$ and generated chaotic matrix $\mathbf{S}$ both have sizes of $M \times N$, and $\mathbf{I}$ is an index matrix obtained by sorting each column of $\mathbf{S}$. The random order substitution can be described as follows:

$$\mathbf{C}_{\mathbf{I}_{i,j},j} = \begin{cases} (\mathbf{T}_{\mathbf{I}_{i,j},j} + \mathbf{T}_{\mathbf{I}_{M,N},N} + \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 1, j = 1; \\ (\mathbf{T}_{\mathbf{I}_{i,j},j} + \mathbf{C}_{\mathbf{I}_{i-1,N},N} + \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 2 \sim N, j = 1; \\ (\mathbf{T}_{\mathbf{I}_{i,j},j} + \mathbf{C}_{\mathbf{I}_{i,j-1},j-1} + \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 1 \sim N, j = 2 \sim N, \end{cases} \tag{6}$$

where the function $\lfloor \alpha \rfloor$ is used to obtain the greatest integer that is not larger than α, and F is the number of intensity levels; for example, $F = 256$ if every pixel is presented using 8 bits. The substitution in the decryption process carries out the inverse operation using the same substitution order, which can be defined as:

$$\mathbf{T}_{\mathbf{I}_{i,j},j} = \begin{cases} (\mathbf{C}_{\mathbf{I}_{i,j},j} - \mathbf{C}_{\mathbf{I}_{i,j-1},j-1} - \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 1 \sim N, j = 2 \sim N; \\ (\mathbf{C}_{\mathbf{I}_{i,j},j} - \mathbf{C}_{\mathbf{I}_{i-1,N},N} - \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 2 \sim N, j = 1; \\ (\mathbf{C}_{\mathbf{I}_{i,j},j} - \mathbf{T}_{\mathbf{I}_{M,N},N} - \lfloor 2^{32} \times \mathbf{S}_{\mathbf{I}_{i,j},j} \rfloor) \bmod F & \text{for } i = 1, j = 1. \end{cases} \tag{7}$$

It should be noted that the substitution order in the decryption process is the exact opposite.

4. Performance evaluation of generated chaotic maps

The chaotic maps produced by the CTBCS exhibit complex chaotic behaviors. To demonstrate this property, we estimate the chaos performance of the three 1D chaotic maps produced by the CTBCS, and compare these with the chaotic maps produced by Zhou's method in [47] and their seed maps. The comparisons are performed in terms of bifurcation diagrams, Lyapunov exponents [47], and sample entropy [33].

4.1. Bifurcation diagram

The bifurcation diagram of a dynamical system plots its visited or approximately visited points in a phase plane. It provides a visualized method to exhibit the chaotic system behaviors. Fig. 6 presents the bifurcation diagrams of the LSC, STC, and TLC maps as well as their seed maps. The three seed maps, namely the Logistic, Sine, and Tent maps, have fixed or periodic points in most parameter settings. Moreover, their output states are only distributed in a small part of the phase plane. However, the three generated chaotic maps exhibit complicated behaviors in all parameter ranges, and their output states are randomly distributed in the entire phase plane, indicating that they have robust chaotic behaviors and their outputs are more random.

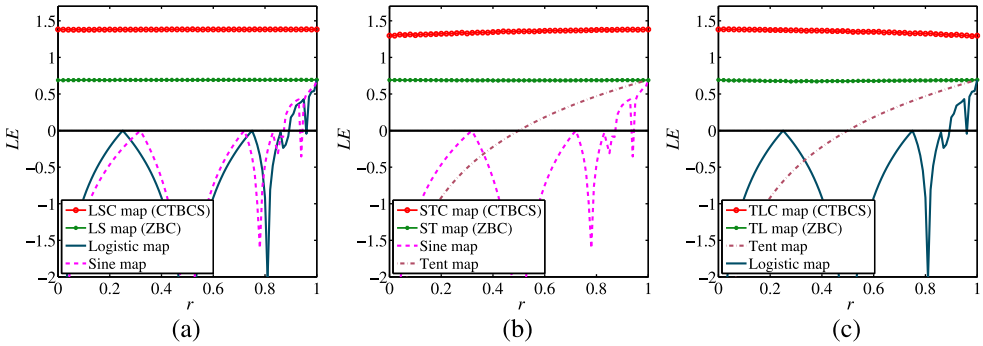

Fig. 7. Comparisons between various generated chaotic maps and their seed maps in terms of LE: (a) LSC map generated by CTBCS and LS map generated by ZBC [47]; (b) STC map generated by CTBCS and ST map generated by ZBC [47]; (c) TLC map generated by CTBCS and TL map generated by ZBC [47].

4.2. Lyapunov exponent

Because chaotic behavior is one type of observed behavior, researchers from different fields have different viewpoints regarding the existence of chaos. Among many technologies for evaluating chaos, the Lyapunov exponent (LE) is a commonly accepted metric that describes the separation degree of two trajectories of a dynamical system, starting from extremely close initial points. It can be described using Definition 1.

Definition 1. The LE of a first-order difference equation $x_{i+1} = F(x_i)$ is mathematically defined by

$$\lambda_{F(x)} = \lim_{n\to\infty} \left\{ \frac{1}{n} \ln \left| \frac{F^n(x_0+\epsilon) - F^n(x_0)}{\epsilon} \right| \right\}, \tag{8}$$

where ϵ is a very small positive value.

If $F(x)$ is differentiable, Eq. (8) can be rewritten as

$$\lambda_{F(x)} = \lim_{n\to\infty} \left\{ \frac{1}{n} \sum_{i=0}^{n-1} \ln \left| F'(x_i) \right| \right\}. \tag{9}$$

An LE greater than 0 means that two trajectories of a chaotic system will diverge exponentially in each iteration. Thus, a dynamical system is considered to exhibit chaotic behaviors if it can obtain a positive LE. A larger LE means that the trajectories diverge faster, indicating superior chaos performance. Fig. 7 plots the LEs of the chaotic maps produced by the CTBCS and ZBC [47] with their used seed maps. The LSC map (CTBCS), STC map (CTBCS), TLC map (CTBCS), LS map (ZBC), ST map (ZBC), and TL map (ZBC) can obtain positive LEs in every parameter setting, while the used seed maps have positive LEs in only a few parameters. This indicates that both the CTBCS and ZBC can generate new chaotic maps that have wider chaotic ranges than their seed maps. Furthermore, it can be observed that, utilizing the same seed maps, the CTBCS can produce chaotic maps with significantly larger LEs than ZBC. This demonstrates that the proposed CTBCS is able to produce chaotic maps with more complex chaotic behaviors.

4.3. Sample entropy

The sample entropy (SE) was developed to describe the complexity of a time series [33]. It provides a quantitative description of the similarity of outputs produced by dynamical systems, and is defined in Definition 2. The dimension m and distance r are usually set as 2 and $0.2 \times std$, respectively, where std indicates the standard deviation of the tested time series. A larger SE demonstrates a lower degree of regularity of the time series; that is, higher complexity of the corresponding dynamical system.

Definition 2. The SE of a time series $\{x_1, x_2, \ldots, x_N\}$ with a given dimension m is defined as

$$SE(m, r, N) = -\log \frac{A}{B}, \tag{10}$$

where A and B are the numbers of vectors having $d[X_{m+1}(i), X_{m+1}(j)] < r$ and $d[X_m(i), X_m(j)] < r$, respectively. The template vector $X_m(i) = \{x_i, x_{i+1}, \ldots, x_{i+m-1}\}$, $d[X_m(i), X_m(j)]$ is the Chebyshev distance [2] between $X_m(i)$ and $X_m(j)$, and r is the given distance.

Fig. 8 compares the SEs of the chaotic maps produced by the CTBCS and ZBC [47] with their seed maps. In each comparison, new chaotic maps are first produced by the CTBCS and ZBC using the same seed maps, following which the SEs of

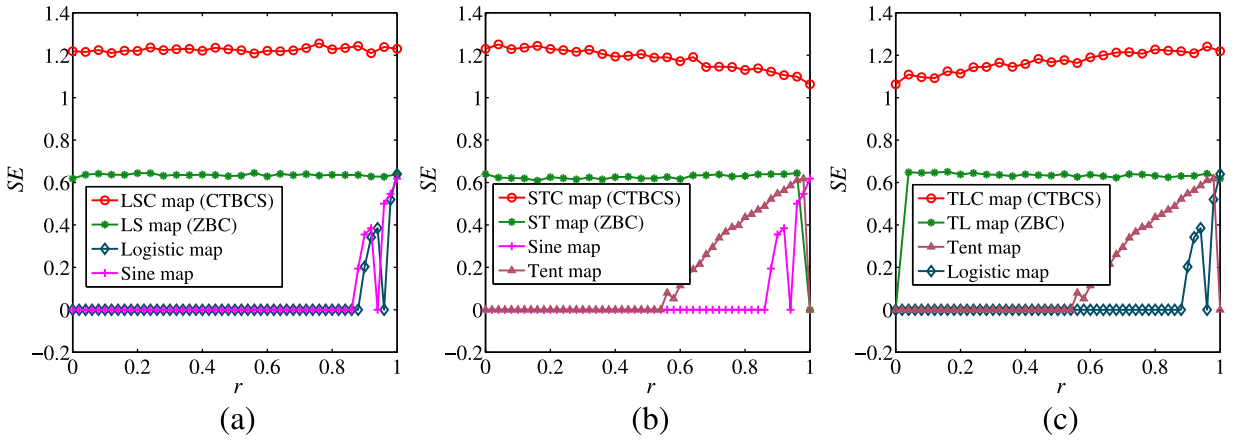

Fig. 8. Comparisons among various generated chaotic maps and their seed maps in terms of SE: (a) LSC map generated by CTBCS and LS map generated by ZBC [47]; (b) STC map generated by CTBCS and ST map generated by ZBC [47]; (c) TLC map generated by CTBCS and TL map generated by ZBC [47].

the generated chaotic maps and their seed maps are calculated. It can be observed that the chaotic maps produced by the CTBCS have significantly larger SEs compared to those produced by ZBC and their used seed maps. This indicates that the CTBCS can produce chaotic maps with more complex output sequences.

5. Simulation results and security analysis of LSC-IES

This section implements the proposed LSC-IES in the MATLAB software and analyzes its security. The test images are mainly obtained from the USC-SIPI "Miscellaneous"[1], the USC-SIPI "Aerials"[2] and the BOWS-2[3] image datasets. The USC-SIPI "Miscellaneous" consists of 44 images, including 16 color and 28 monochrome images. The USC-SIPI "Aerials" dataset consists of 38 images, including 37 color and 1 monochrome. The BOWS-2 contains 10,000 original images used in the 2nd BOWS Contest.

5.1. Simulation results

An encryption algorithm should be capable of encrypting different types of digital images into cipher-images with high security levels. Only by using the correct secret key can one completely recover the information of the original image. Without the correct secret key, one cannot obtain any information regarding the original image. Fig. 9 illustrates the encrypted results of the LSC-IES using different types of images. The grayscale image is selected from the USC-SIPI "Miscellaneous" dataset, and the color image is from the USC-SIPI "Aerials" dataset. When a color image is encrypted, its three color channels are encrypted independently. As can be observed, the five test images have various patterns, but LSC-IES can encrypt these to be cipher-images with random distributions. Attackers experience difficulties in obtaining the original information from their pixel distribution. As each step is completely reversible, the proposed LSC-IES can fully recover the original images from the corresponding cipher-images.

An image encryption algorithm should exhibit high encryption efficiency. Because the LSC map used has a rapid implementation speed, the high-efficiency scrambling and random order substitution have low computational complexity. Thus, the LSC-IES can achieve a fast encryption speed. Table 2 lists the required times of encrypting one image with different sizes using various image encryption algorithms. The experiments are performed on a computer under the following environments: Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz with 8 GB memory, Windows 10 operating system. It can be observed that, for different image sizes, the proposed LSC-IES exhibits faster encryption speeds than other encryption algorithms.

An image encryption algorithm is also required to produce decrypted images with high quality. In the LSC-LES encryption operation, a slight difference of a pixel in the plain-image can influence all pixels in the cipher-image, which can guarantee that the cipher-images exhibit high security levels. However, in the decryption operation, a slight difference of a pixel in the cipher-image can influence only a few pixels of the decrypted result. In this case, if a cipher-image encrypted by the LSC-IES loses a certain amount of data, the decryption process can also recover most visual contents of the original image. Fig. 10 illustrates the image quality of the decrypted results when the cipher-images of the LSC-IES suffer from noise or different data loss percentages. It can be observed from Fig. 10(a) that, when the cipher-image experiences no data loss, the decryption procedure can completely recover the original image. However, even when the cipher-images have noise or

[1] http://sipi.usc.edu/database/database.php?volume=misc.
[2] http://sipi.usc.edu/database/database.php?volume=aerials.
[3] http://bows2.ec-lille.fr/.

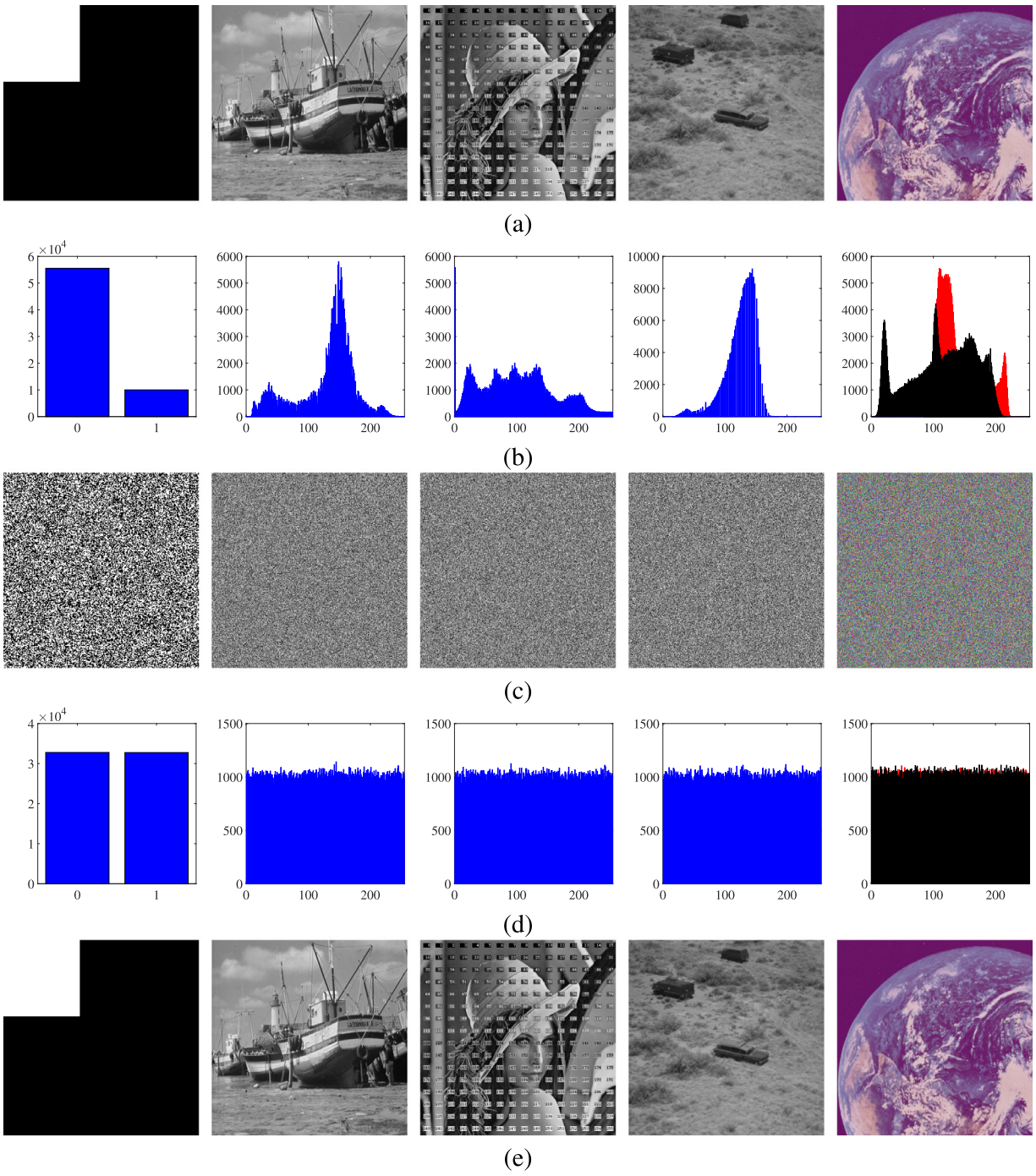

Fig. 9. Simulation results of five different digital images: (a) plain-images, including one binary image, three grayscale images, and one color image; (b) histograms of (a); (c) encryption results of (a); (d) histograms of (c); and (e) decryption results of (c).

lose some data, their decrypted results contain the most visual information of the original images, as can be observed from Figs. 10(b) to (e). Thus, the proposed LSC-IES can decrypt cipher-images with high quality.

5.2. Security analysis

To demonstrate the superiority of the LSC-IES, this section analyzes its security level in terms of the following four aspects: key security, randomness of cipher-images, ability to resist differential attacks, and adjacent pixel correlation. To

Table 2
Required time (seconds) for encrypting one image with different sizes using different image encryption algorithms.

Image size	128×128	256×256	512×512	1024×1024
Diaconu [9]	0.0579	0.2224	0.9731	3.8377
PXMW [32]	0.0902	0.3440	1.3357	5.3223
CCB [5]	0.2757	0.9810	3.8539	15.4565
HZ [18]	0.1531	0.6347	2.4913	9.9185
XLLH [40]	0.0247	0.1164	0.4924	20.144
ZBC1 [47]	0.0933	0.3843	1.4824	5.8175
LLZ [28]	0.0323	0.1440	0.5510	2.0864
LSC-IES	0.0244	0.0949	0.4010	1.9857

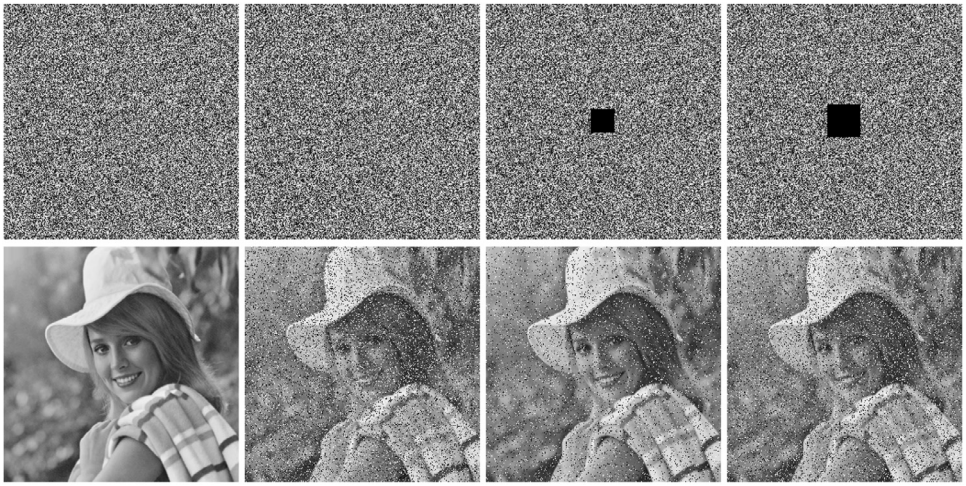

Fig. 10. Quality analysis of image decryption: (a) cipher-image and its decrypted image; (b) cipher-image with 2% 'salt & pepper' noise and its decrypted image; (c) cipher-image with 1% data loss and its decrypted image; (d) cipher-image with 2% data loss and its decrypted image.

illustrate the effectiveness of the LSC-IES further, we compare it with the following advanced image encryption methods: WWZ [36], ZBC1 [47], XLLH [40], LSZ [30], HZ [18], ZBC2 [46], WZNS [38], LLZ [28], and CKW [3]. For these competing image encryption schemes, several test results are reported in the reference papers and some of their source codes can be accessed. To conduct a fair comparison, we report the test results of the competing encryption algorithms as follows. Firstly, if the experimental results of the algorithm have been reported by authors, we reference the reported results directly. Next, if the experimental results have not been reported but the algorithm source codes can be accessed, we run the source codes to obtain the experimental results. Finally, if neither the experimental results nor source codes can be obtained, we implement the algorithm to obtain the experimental results.

5.2.1. Key security analysis

The key size of an encryption algorithm should be appropriate. The key space of the LSC-IES is 2^{256}, which satisfies the key size requirements and exhibits high performance in resisting various security attacks [1]. Moreover, an encryption algorithm should be extremely sensitive with its secret key. Otherwise, incorrect secret keys with slight differences may also correctly decrypt the information of the original image, which may make the actual key space smaller than the theoretical one [1].

The number of bit change rate (NBCR) defined in [4] can be used to test the key sensitivity of the LSC-IES. For two images $\mathbf{B}_1$ and $\mathbf{B}_2$, the NBCR is defined as

$$\mathtt{NBCR}(\mathbf{B}_1, \mathbf{B}_2) = \frac{\mathtt{Ham}(\mathbf{B}_1, \mathbf{B}_2)}{Len}, \tag{11}$$

where $\mathtt{Ham}(\mathbf{B}_1, \mathbf{B}_2)$ is the Hamming distance of $\mathbf{B}_1$ and $\mathbf{B}_2$, and *Len* is the bit length of $\mathbf{B}_1$ or $\mathbf{B}_2$. From Eq. (11), we can determine that, if the obtained NBCR approaches 50%, the two images $\mathbf{B}_1$ and $\mathbf{B}_2$ are totally different.

Our experiment is established as follows. For a given secret key $\mathbf{K}_1$, we change one of its 256 bits to obtain $\mathbf{K}_2$. To test the key sensitivity in the encryption process, we first encrypt the plain-image $\mathbf{P}$ using $\mathbf{K}_1$ and $\mathbf{K}_2$, respectively, and generate two cipher-images $\mathbf{C}_1$ and $\mathbf{C}_2$. Subsequently, the NBCR values of $\mathbf{C}_1$ and $\mathbf{C}_2$ are calculated. To test the key sensitivity in the decryption process, we first decrypt the cipher-image $\mathbf{C}_1$ using $\mathbf{K}_1$ and $\mathbf{K}_2$, respectively, and obtain two decrypted results

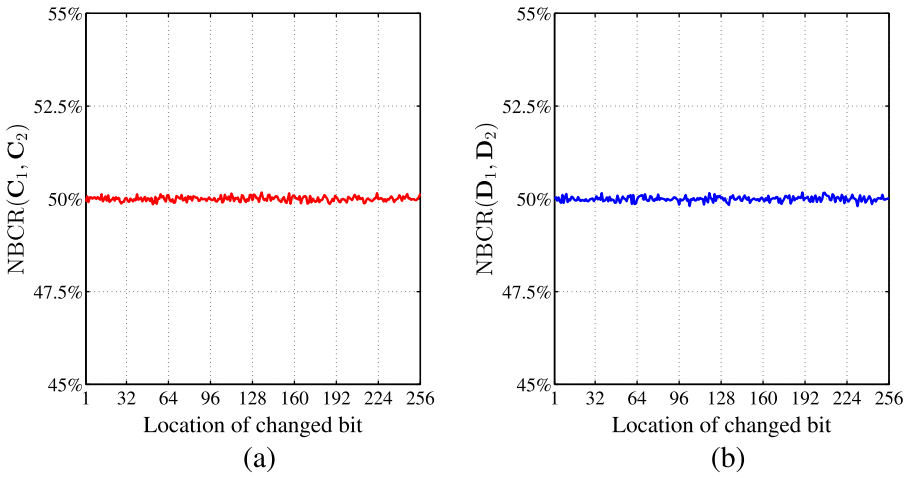

Fig. 11. Sensitivity analysis of secret key: (a) NBCR between $\mathbf{C}_1$ and $\mathbf{C}_2$, where $\mathbf{C}_1$ and $\mathbf{C}_2$ are two encrypted images obtained from two secret keys with one bit difference; and (b) NBCR between $\mathbf{D}_1$ and $\mathbf{D}_2$, where $\mathbf{D}_1$ and $\mathbf{D}_2$ are two decrypted results from two secret keys with one bit difference.

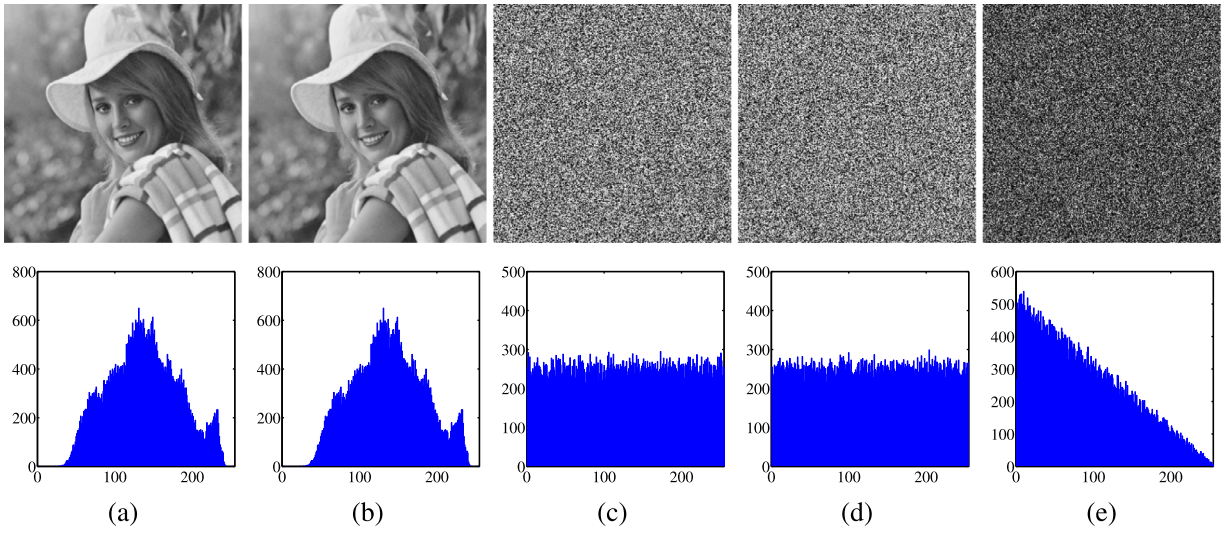

Fig. 12. Demonstration of diffusion property of LSC-IES: (a) plain-image $\mathbf{P}_1$; (b) plain-image $\mathbf{P}_2$, which has one bit difference with $\mathbf{P}_1$ in position (100,100); (c) cipher-image $\mathbf{C}_1 = Enc(\mathbf{P}_1, \mathbf{K})$; (d) cipher-image $\mathbf{C}_2 = Enc(\mathbf{P}_2, \mathbf{K})$; and (e) difference $|\mathbf{C}_1 - \mathbf{C}_2|$.

$\mathbf{D}_1$ and $\mathbf{D}_2$. Then, the NBCR values of $\mathbf{D}_1$ and $\mathbf{D}_2$ are calculated. Fig. 11 illustrates the key sensitivity analysis results of the LSC-IES. It is indicated that, when any one bit of the secret key is changed, the two obtained cipher-images and decrypted images are completely different. This means that the LSC-IES has an extremely sensitive secret key.

5.2.2. Ability to resist differential attacks

The differential attack is a commonly used and an efficient security attack. By studying the manner in which the differences in plain-images can influence cipher-images, the differential attack aims to build a relationship between plain-images and their corresponding cipher-images. An image encryption scheme exhibits high performance in resisting differential attacks if it possesses the diffusion property. The diffusion property indicates that a slight change in the plaintexts can spread over the entire data in the ciphertexts. Fig. 12 illustrates the diffusion property of the LSC-IES. When changing one bit of a plain-image, the two cipher-images encrypted by the LSC-IES are independent and different, as can be observed in Fig. 12(e).

The number of pixel change rate (NPCR) and uniform average change intensity (UACI) are two measures for testing whether an encryption algorithm can resist differential attacks [37]. The NPCR counts the number of different pixels, while the UACI calculates the average change in pixels between two images. Supposing that $\mathbf{C}_1$ and $\mathbf{C}_2$ are two cipher-images that their plain-images have only one bit difference, their NPCR and UACI values can be calculated asfollows:

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

and

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

where G is the number of pixels in an image, R indicates the largest allowed pixel value, and $\mathbf{A}$ records the difference between $\mathbf{C}_1$ and $\mathbf{C}_2$, which is defined as

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

In recent years, strictly critical NPCR and UACI values were proposed in [37]. Given a significance level α, the critical NPCR score $\mathcal{N}^*_\alpha$ is obtained as follows:

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

An image encryption algorithm offers a high ability to resist differential attacks if its obtained NPCR value is greater than $\mathcal{N}^*_\alpha$. The critical UACI scores $(\mathcal{U}^{*-}_\alpha, \mathcal{U}^{*+}_\alpha)$ with the given α can be obtained using

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

where

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

and

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

An encryption algorithm can pass the test if the calculated UACI value is within the range $(\mathcal{U}^{*-}_\alpha, \mathcal{U}^{*+}_\alpha)$.

Our experiment uses the 25 grayscale images from the USC-SIPI "Miscellaneous" dataset as the test images. Among these images, seven images have a size of 256×256, 15 images have a size of 512×512, and the remaining four images have a size of 1024×1024. The significance level α is set as 0.05 in our experiment, as recommended by Wu et al. [37]. Then, for an 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 an image of size

Table 3
NPCR results of various image encryption schemes. The image sizes images from "5.1.09" to "5.2.08", from "5.2.09" to "ruler.512", and from "5.3.01" to "testpat.1k" are 256×256, 512×512 and 1024×1024, respectively. The numbers in bold fonts are the test results that fail to pass the NPCR test.

File name	WWZ [36]	ZBC1 [47]	XLLH [40]	LSZ [30]	HZ [18]	ZBC2 [46]	WZNS [38]	CKW [3]	LSC-IES
5.1.09	99.6338	**99.5575**	99.6246	99.6109	99.6124	99.6170	99.6506	99.6140	99.6292
5.1.10	99.6017	**99.5544**	**0.0092**	99.5987	99.5972	99.6017	99.6063	99.5880	99.6292
5.1.11	99.5956	99.8123	99.6445	99.5697	99.5956	99.6460	99.6490	99.6033	99.7055
5.1.12	99.6292	99.6109	99.5972	99.6338	99.6017	99.6048	99.6170	**99.5651**	99.7055
5.1.13	99.6109	99.7421	99.6582	99.6201	99.6552	99.5758	**99.5605**	99.5789	99.6765
5.1.14	99.6185	99.6933	99.5987	99.6094	99.6002	**99.5621**	99.6216	99.6765	99.6765
5.2.08	99.6059	99.6101	99.6216	99.6056	99.6220	99.6292	99.5987	99.6037	99.6250
5.2.09	99.6029	99.7025	99.6048	99.6410	99.6208	99.6048	99.6220	99.6029	99.6292
5.2.10	99.6010	99.6120	**99.5861**	99.6067	99.5968	99.6155	99.6162	99.6124	99.6212
7.1.01	99.6040	**99.5190**	99.6162	99.6132	99.6181	99.6166	99.6166	99.6082	99.6208
7.1.02	99.5941	99.7200	99.6025	99.6124	99.6140	99.5968	99.6109	99.6174	99.6025
7.1.03	99.6132	**99.4072**	99.5998	99.5983	99.6166	99.6075	99.6216	99.6120	99.6078
7.1.04	99.6098	99.6037	99.6033	99.6052	99.6227	99.6277	99.6090	99.5911	99.6082
7.1.05	99.5995	**99.4572**	99.6307	**99.5842**	99.5960	99.6284	99.6063	99.6178	99.6014
7.1.06	99.6109	**99.5213**	99.6105	99.6086	99.6212	99.6025	99.6101	99.6174	99.6063
7.1.07	99.5934	**99.5007**	99.6029	99.6193	99.6113	99.6037	99.6220	99.5922	99.5964
7.1.08	99.6098	99.6902	99.6120	**99.5869**	99.5914	99.6304	99.6101	99.6056	99.5953
7.1.09	**99.5815**	99.7181	99.6048	99.6094	99.6067	99.6193	**99.5861**	99.6086	99.6094
7.1.10	**99.5850**	**99.5163**	99.6212	99.6063	99.6056	99.6380	99.6120	99.5941	99.6078
boat.512	99.6284	99.7128	99.6067	99.6132	99.6021	99.6109	99.6086	99.6101	99.6181
gray21.512	99.6014	99.6120	99.6094	99.6162	99.6239	99.6109	99.6040	99.6159	99.6029
ruler.512	99.6151	**99.3118**	99.6113	99.6189	99.5930	99.6082	99.6227	99.6212	99.6033
5.3.01	99.6132	99.6040	99.6116	99.6169	99.6100	99.6173	99.6099	99.6072	99.6061
5.3.02	99.6107	**99.4789**	99.6223	99.6010	99.6129	99.6099	99.6099	99.6116	99.6190
7.2.01	99.6010	99.7578	99.6042	99.6147	**99.5964**	**99.5981**	99.6130	99.6204	99.6077
Pass rate	23/25	15/25	23/25	23/25	24/25	23/25	23/25	24/25	25/25

Table 4
UACI results of various image encryption schemes. The image sizes images from "5.1.09" to "5.2.08", from "5.2.09" to "ruler.512", and from "5.3.01" to "testpat.1k" are 256×256, 512×512, and 1024×1024, respectively. The numbers in bold fonts are the test results that fail to pass the UACI test.

File name	WWZ [36]	ZBC1 [47]	XLLH [40]	LSZ [30]	HZ [18]	ZBC2 [46]	WZNS [38]	CKW [3]	LSC-IES
5.1.09	33.5119	**33.7574**	33.4425	33.5527	33.4652	**33.2580**	33.4387	33.4032	33.3651
5.1.10	33.4263	**33.1739**	**0.0011**	33.4381	33.5448	33.2845	33.4701	33.3557	33.5240
5.1.11	33.4192	33.3198	33.4840	33.4390	33.4162	33.5448	33.4150	33.4696	33.5106
5.1.12	**33.2672**	**33.6656**	33.3609	33.4373	**33.2152**	33.3578	33.5082	33.4634	33.4172
5.1.13	33.4252	**34.3306**	33.3039	33.3488	33.4699	**33.7371**	33.4939	33.3046	33.5065
5.1.14	33.2919	**33.1888**	33.5008	33.5133	33.5554	**33.2361**	**33.7240**	33.4796	33.4875
5.2.08	33.4509	**32.7443**	33.5233	33.4377	33.4575	**33.2432**	33.4694	33.4493	33.4973
5.2.09	33.4543	**34.0963**	33.4834	33.4939	33.4175	33.5176	33.4704	33.5077	33.4778
5.2.10	33.4365	33.4982	33.4532	33.3888	33.4315	**33.3565**	**33.5688**	33.4457	33.4327
7.1.01	33.4811	33.5512	**33.3369**	**33.5553**	33.5150	**33.1477**	33.4531	33.4890	33.4154
7.1.02	33.4762	**33.0872**	33.4121	33.4342	33.5221	33.4418	33.3931	33.4190	33.4698
7.1.03	33.5346	**33.7230**	33.4970	33.4585	33.4777	**33.2279**	33.4599	33.4689	33.4632
7.1.04	**33.3450**	**33.6036**	33.4412	33.4830	33.4721	**33.1993**	33.4471	33.4997	33.4996
7.1.05	33.5380	**33.1520**	33.4753	33.4393	33.4757	**33.2974**	33.3758	33.4313	33.4647
7.1.06	33.4766	**33.8290**	33.4571	**33.5634**	33.5035	**33.3352**	33.4942	33.4760	33.4416
7.1.07	33.4695	**33.5833**	33.3844	33.5241	33.4317	**33.2157**	33.4876	33.4470	33.3906
7.1.08	33.4258	33.4212	33.3863	33.4251	33.4274	**33.2077**	33.5078	33.5203	33.4029
7.1.09	33.4954	**32.8751**	33.3879	33.4606	33.4452	**33.2849**	33.4584	33.4704	33.4686
7.1.10	33.4389	**32.9976**	33.4615	33.4119	33.4434	**33.1952**	33.4332	33.4892	33.4434
boat.512	33.4693	**33.9503**	33.4589	33.4993	33.4059	**33.3673**	33.4197	33.5414	33.4472
gray21.512	33.4667	33.4646	33.3857	33.4634	33.4554	33.4830	33.4906	33.4331	33.4781
ruler.512	33.5154	33.4628	33.5253	33.5090	33.4795	**33.7365**	33.5193	33.4363	33.3883
5.3.01	33.4973	**32.9559**	33.5380	33.4698	33.4516	33.4406	33.4413	33.4886	33.4683
5.3.02	**33.5109**	**33.9252**	33.4525	33.4820	33.4579	**33.3072**	**33.5189**	33.4384	33.4428
7.2.01	33.4826	**33.3653**	33.4348	33.4878	33.4718	**33.5820**	33.4428	33.4192	33.4688
Pass rate	22/25	6/25	23/25	23/25	24/25	7/25	22/25	25/25	25/25

512×512, $\mathcal{N}^*_{0.05} = 99.5893\%$ and $(\mathcal{U}^{*-}_{0.05}, \mathcal{U}^{*+}_{0.05}) = (33.3730\%, 33.5541\%)$; for an image of size 1024×1024, $\mathcal{N}^*_{0.05} = 99.5994\%$ and $(\mathcal{U}^{*-}_{0.05}, \mathcal{U}^{*+}_{0.05}) = (33.4183\%, 33.5088\%)$. Tables 3 and 4 display the NPCR and UACI values of the different image encryption schemes, respectively. It can be observed that the LSC-IES can pass all NPCR and UACI tests, while other encryption schemes cannot pass certain tests. This indicates that the proposed LSC-IES exhibits high performance in terms of resisting differential attacks.

5.2.3. Randomness of cipher-images

The pixels of an ideal cipher-image are required to be uniformly distributed to resist statistic attacks. The National Institute of Standards and Technology (NIST) SP800-22 is an established criterion for measuring the randomness of data sequence [21]. It contains 15 sub-tests and uses a group of binary sequences as input. All sub-tests aim to identify the non-randomness areas of the binary sequences. Each sub-test produces a $P - value$ for the group of binary sequences, and generates a $p - value$ for every binary sequence. Our experiment sets the significance level α as 0.01, as recommended by III et al. [21]. The number of tested binary sequences should be larger than or equal to α^{-1}, and is 120 in our experiment.

The test results are interpreted from the following three perspectives: $P - value$, pass rate, and $p - value_T$. The $P - value$ interpretation checks the $P - value$s for all binary sequences and passes the test if the corresponding $P - value$ is within the range $[\alpha, 1]$. The pass rate interpretation counts the pass proportion of the 120 binary sequences. Using the calculation method provided in [21], the minimum pass rate $T = 0.9628$ when the number of binary sequences $s = 120$ and the significance level $\alpha = 0.01$. The $p - value_T$ interpretation collects the distribution of $P - value$s. It is calculated using the distribution of $P - value$ according to the calculation method in [21]. If the generated $p - value_T$ is larger than 0.0001, the test sequences are considered to pass the test.

Our experiment uses 120 digital images (filenames from "1.pgm" to "120.pgm") selected from the BOWS-2 database. First, these images are encrypted using the LSC-IES to obtain 120 cipher-images, and each cipher-image is then used as a binary sequence. As all test images have a size of 512×512 and every pixel is represented by 8 bits, a binary sequence has a bit length of $512 \times 512 \times 8 = 2,097,152$. Thus, our experiment uses 120 binary sequences as input, each of which has a length of 2,097,152 bits, and Table 5 lists the test results. It can be observed that the 120 images encrypted by the LSC-IES can pass all 15 sub-tests using the three interpretations. This indicates that the cipher-images of the LSC-IES exhibit high randomness.

5.2.4. Adjacent pixel correlation analysis

As data redundancy exists in natural images, the adjacent pixels of a natural image may exhibit high correlations. An image encryption algorithm needs to weaken these high correlations. The correlation of two pixel sequences can be calculated by

$$APC = \frac{E[(\mathbf{X} - \mu_{\mathbf{X}})(\mathbf{Y} - \mu_{\mathbf{Y}})]}{\sigma^2}, \tag{12}$$

Table 5
Randomness test results of 120 cipher-images encrypted by LSC-IES using NIST SP800-22 test suite.

Sub-tests		*P-value* ≥ 0.01		Pass rate ≥ 0.9628		$P-value_T$ ≥ 0.0001	
Frequency		0.4373	Pass	0.9917	Pass	0.5627	Pass
Block frequency ($m = 128$)		0.7565	Pass	0.9833	Pass	0.8383	Pass
Cumulative sums	Forward	0.6025	Pass	0.9917	Pass	0.7184	Pass
	Reverse	0.4686	Pass	0.9917	Pass	0.5946	Pass
Runs		0.4071	Pass	0.9667	Pass	0.5308	Pass
Long runs of ones		0.0781	Pass	1.0000	Pass	0.1033	Pass
Rank		0.0822	Pass	1.0000	Pass	0.1094	Pass
Spectral DFT		0.5681	Pass	0.9750	Pass	0.6884	Pass
Non-overlapping templates*		0.4787	Pass	0.9903	Pass	0.5546	Pass
Overlapping templates ($m = 9$)		0.2328	Pass	0.9917	Pass	0.3229	Pass
Universal		0.3925	Pass	0.9917	Pass	0.5150	Pass
Approximate entropy ($m = 10$)		0.1552	Pass	0.9750	Pass	0.2165	Pass
Random excursions*		0.2447	Pass	0.9936	Pass	0.7719	Pass
Random excursions variant*		0.2923	Pass	0.9929	Pass	0.7670	Pass
Serial ($m = 16, \nabla\Psi_m^2$)	$P-value1$	0.5681	Pass	0.9917	Pass	0.6884	Pass
	$P-value2$	0.2873	Pass	0.9833	Pass	0.3926	Pass
Linear complexity ($M = 500$)		0.4686	Pass	0.9917	Pass	0.5946	Pass

*Average value of multiple tests.

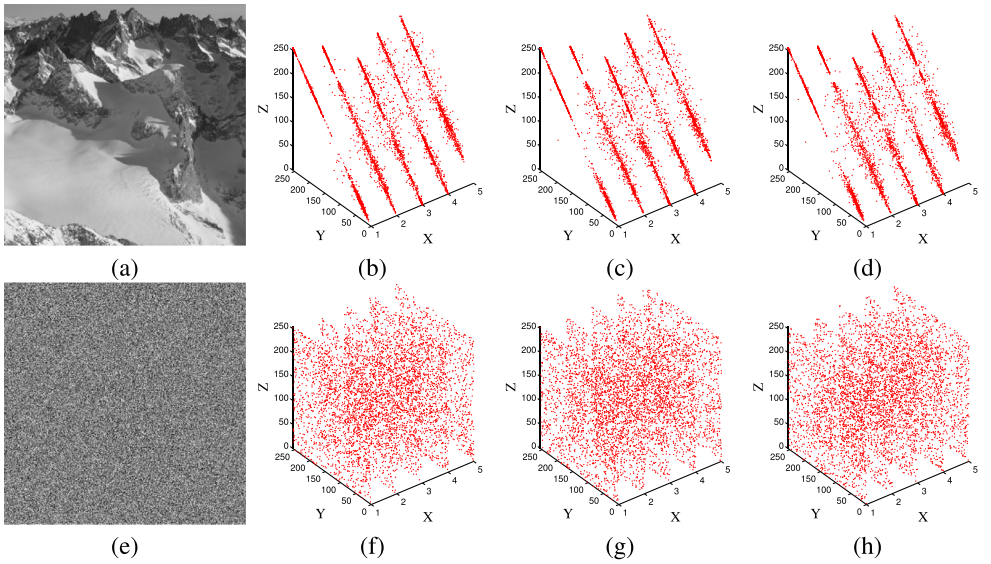

Fig. 13. Adjacent pixels of five original images and their encrypted images by LES-IES: (a) one of five plain-images; (b) horizontal adjacent pixel pairs of five plain-images; (c) vertical adjacent pixel pairs of five plain-images; (d) diagonal adjacent pixel pairs of five plain-images; (e) cipher-image of (a); (f) horizontal adjacent pixel pairs of five cipher-images; (g) vertical adjacent pixel pairs of five cipher-images; and (h) diagonal adjacent pixel pairs of five cipher-images. In each figure, the X-axis denotes the index of the five images, while the Y-Z plane plots the pixel pairs.

where **X** and **Y** denote the two pixel sequences, μ is the mathematical expectation, and σ is the mathematical standard derivation. When testing the adjacent pixel correlation of an image, **X** is a selected pixel sequence and **Y** is the adjacent pixel sequence, and each pixel of **Y** is the horizontal, vertical, or diagonal neighbor pixel of **X**. If **X** and **Y** exhibit a high correlation, the obtained APC approaches 1 or -1; otherwise, it approaches 0.

To demonstrate the ability of the LSC-IES in terms of decorrelating natural images visually, we first select five images from the BOWS-2 image dataset (filenames from “1” to “5”) and then encrypt these using the LSC-IES. Fig. 13 plots these adjacent pixel pairs of the plain-images and their cipher-images. A total of 1000 pixel pairs are randomly selected from every image. It can be observed that, in each figure, the X-axis indicates the five images, while the Y-Z plane plots the values of the adjacent pixels. The adjacent pixel pairs of the plain-images are mostly on or close the diagonal lines, indicating that these adjacent pixels exhibit strong correlations. However, the adjacent pixel pairs of all cipher-images are distributed quite randomly across the Y-Z phase plane, demonstrating that they exhibit weak correlations. This indicates that the proposed LSC-IES can efficiently decorrelate the high correlations of the plain-images.

Table 6 displays the APC values of the plain-image “Lena” and its cipher-images, encrypted by various image encryption methods. As the APC results of LLZ [28], ZBC2 [46], and WZNS [38] have been reported in [38], and those of [18] have

Table 6
Adjacent pixel correlations of plain-image "Lena" and its results using various image encryption methods.

Encryption schemes	Horizontal	Vertical	Diagonal
"Lena" image	0.9400	0.9709	0.9710
FLMLC [12]	0.0368	0.0392	0.0068
ZBC1 [47]	−0.0054	0.0045	0.0031
WZNS [38]	0.0053365	−0.0027616	0.0016621
WWZ [36]	−0.026878	0.024223	−0.000569
XLLH [40]	0.000369	0.001441	0.002217
LSZ [30]	−0.001458	−0.002029	0.001899
HZ [18]	0.0033455	−0.0008454	−0.0002044
ZBC2 [46]	0.002098	−0.000991	−0.001337
LLZ [28]	0.0127	−0.0190	−0.0012
CKW [3]	0.001381	0.0016061	0.002578
LSC-IES	−0.003280	−0.000777	−0.000181

been reported in its literature, we reference these results directly. We implemented the other encryption algorithms in the MATLAB software and calculated their APC results. From the table, it can be observed that the LSC-IES outperforms LLZ [28], FLMLC [12], ZBC1 [47], WZNS [38], WWZ [36], and HZ [18] in all the horizontal, vertical and diagonal directions, and has smaller absolute APC values than the remaining algorithms in two directions. This further demonstrates that the LSC-IES exhibits high efficiency in breaking the strong correlations of adjacent pixels in natural images.

6. Conclusion

This paper firstly proposed a cosine-transform-based chaotic system known as the CTBCS, which uses the cosine transform as a nonlinear transform to produce new chaotic maps with complex chaos performance. As examples, three new chaotic maps were produced to demonstrate the efficiency of the CTBCS. The performance evaluations demonstrated that the chaotic maps of the CTBCS exhibit significantly superior chaos performance over the chaotic maps produced by other methods and their seed maps. Using the LSC map generated by the CTBCS, we further proposed an image encryption scheme known as LSC-IES. The LSC-IES follows the well-known diffusion-confusion concept and we simulated it using different digital images. The security analysis indicated that the LSC-IES is quite sensitive to its secret keys, and has a higher security level than several competing image encryption algorithms. This work can promote the development of chaos theory and chaos-based encryption. We will investigate the further application of LSC-IES in video encryption.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China, under grants 2018YFB1003800, 2018YFB1003805 and 2016YFB0800804, the Shenzhen Science and Technology Program, under grants JCYJ20170307150704051 and JCYJ20170811160212033, the National Natural Science Foundation of China, under grant 61701137, the Macau Science and Technology Development Fund, under grant FDCT/189/2017/A3, and by the Research Committee at the University of Macau, under grants MYRG2016-00123-FST and MYRG2018-00136-FST.

References

[1] G. Alvarez, S. Li, Some basic cryptographic requirements for chaos-based cryptosystems, Int. J. Bifurcat. Chaos 16 (8) (2006) 2129–2151.
[2] C.D. Cantrell, Modern Mathematical Methods for Physicists and Engineers, Cambridge University Press, 2000.
[3] C. Cao, K. Sun, W. Liu, A novel bit-level image encryption algorithm based on 2D-licm hyperchaotic map, Signal Process. 143 (2018) 122–133.
[4] J.C.H. Castro, J.M. Sierra, A. Seznec, A. Izquierdo, A. Ribagorda, The strict avalanche criterion randomness test, Math. Comput. Simul. 68 (1) (2005) 1–7.
[5] X. Chai, Y. Chen, L. Broyde, A novel chaos-based image encryption algorithm using DNA sequence operations, Opt. Lasers Eng. 88 (2017) 197–213.
[6] X. Chai, Z. Gan, Y. Chen, Y. Zhang, A visually secure image encryption scheme based on compressive sensing, Signal Process. 134 (2017) 35–51.
[7] J. Chen, Z.-L. Zhu, L.-B. Zhang, Y. Zhang, B.-Q. Yang, Exploiting self-adaptive permutation-diffusion and dna random encoding for secure and efficient image encryption, Signal Process. 142 (2018) 340–353.
[8] Z. Chen, X. Yuan, Y. Yuan, H.H.C. Iu, T. Fernando, Parameter identification of chaotic and hyper-chaotic systems using synchronization-based parameter observer, IEEE Trans. Circuits Syst. I Regul. Pap. 63 (9) (2016) 1464–1475.
[9] A.-V. Diaconu, Circular inter–intra pixels bit-level permutation and chaos-based image encryption, Inf. Sci. (Ny) 355 (2016) 314–327.
[10] I.C. Dragoi, D. Coltuc, On local prediction based reversible watermarking, IEEE Trans. Image Process. 24 (4) (2015) 1244–1246.
[11] J. Fridrich, Image encryption based on chaotic maps, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation., 2, 1997, pp. 1105–1110.
[12] 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.
[13] L. Gong, X. Liu, F. Zheng, N. Zhou, Flexible multiple-image encryption algorithm based on log-polar transform and double random phase encoding technique, J. Mod. Opt. 60 (13) (2013) 1074–1082.
[14] T. Habutsu, Y. Nishio, I. Sasase, S. Mori, A secret key cryptosystem by iterating a chaotic map, in: Proceedings of the Advances in Cryptology-EUROCRYPT'91, 547, Springer, 1991, pp. 127–140.
[15] Z. Hua, F. Jin, B. Xu, H. Huang, 2D Logistic-Sine-coupling map for image encryption, Signal Process. 149 (2018) 148–161.
[16] Z. Hua, S. Yi, Y. Zhou, Medical image encryption using high-speed scrambling and pixel adaptive diffusion, Signal Process. 144 (2018) 134–144.

[17] Z. Hua, B. Zhou, Y. Zhou, Sine-transform-based chaotic system with FPGA implementation, IEEE Trans. Ind. Electron. 65 (3) (2018) 2557–2566.
[18] Z. Hua, Y. Zhou, Design of image cipher using block-based scrambling and image filtering, Inf. Sci. (Ny) 396 (2017) 97–113.
[19] X. Huang, G. Ye, An efficient self-adaptive model for chaotic image encryption algorithm, Commun. Nonlinear Sci. Numer. Simul. 19 (12) (2014) 4094–4104.
[20] X. Huang, G. Ye, An image encryption algorithm based on hyper-chaos and dna sequence, Multimed. Tools Appl. 72 (1) (2014) 57–70.
[21] 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, National Institute of Standards & Technology, 2010.
[22] V.G. Ivancevic, T.T. Ivancevic, Complex Nonlinearity: Chaos, Phase Transitions, Topology Change and Path Integrals, Springer Science & Business Media, Berlin, Heidelberg, Germany, 2008.
[23] S. John Walker, Big Data: A Revolution That Will Transform How We Live, Work, and Think, Taylor & Francis, 2014.
[24] C. Li, D. Lin, J. Lü, Cryptanalyzing an image-scrambling encryption algorithm of pixel bits, IEEE Multimed. 3 (2017) 64–71, doi:10.1109/MMUL.2017.3051512.
[25] C. Li, D. Lin, J. Lü, F. Hao, Cryptanalyzing an image encryption algorithm based on autoblocking and electrocardiography, IEEE Multimed. (2018), doi:10.1109/MMUL.2018.2873472.
[26] X. Li, Y. Wang, Q.-H. Wang, Y. Liu, X. Zhou, Modified integral imaging reconstruction and encryption using an improved SR reconstruction algorithm, Opt. Lasers Eng. 112 (2019) 162–169.
[27] X. Li, D. Xiao, Q.-H. Wang, Error-free holographic frames encryption with ca pixel-permutation encoding algorithm, Opt. Lasers Eng. 100 (2018) 200–207.
[28] X. Liao, S. Lai, Q. Zhou, A novel image encryption algorithm based on self-adaptive wave transmission, Signal Process. 90 (9) (2010) 2714–2722.
[29] Y.T. Lin, C.M. Wang, W.S. Chen, F.P. Lin, W. Lin, A novel data hiding algorithm for high dynamic range images, IEEE Trans. Multimed. 19 (1) (2017) 196–211.
[30] W. Liu, K. Sun, C. Zhu, A fast image encryption algorithm based on chaotic map, Opt. Lasers Eng. 84 (2016) 26–36.
[31] J. Lu, O. Dunkelman, N. Keller, J. Kim, New impossible differential attacks on AES, in: Proceedings of the Progress in Cryptology-INDOCRYPT, Springer, 2008, pp. 279–293.
[32] P. Ping, F. Xu, Y. Mao, Z. Wang, Designing permutation–substitution image encryption networks with Henon map, Neurocomputing 283 (2018) 53–63.
[33] J.S. Richman, J.R. Moorman, Physiological time-series analysis using approximate entropy and sample entropy, Am. J. Physiol. Heart Circ. Physiol. 278 (6) (2000) H2039–H2049.
[34] H.-J. Stöckmann, Quantum Chaos: An Introduction, 1, Cambridge University Press, Cambridge, CB2 2UR, UK, 2007.
[35] S.H. Strogatz, Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering, Westview press, Boulder, Colorado, 2014.
[36] X. Wang, Q. Wang, Y. Zhang, A fast image algorithm based on rows and columns switch, Nonlinear Dyn. 79 (2) (2015) 1141–1149.
[37] Y. Wu, J.P. Noonan, S. Agaian, NPCR And UACI randomness tests for image encryption, Cyber J. Multidisci. J. Sci. Technol. J. Select. Areas Telecommun. (JSAT) (2011) 31–38.
[38] Y. Wu, Y. Zhou, J.P. Noonan, S. Agaian, Design of image cipher using latin squares, Inf. Sci. (Ny) 264 (0) (2014) 317–339.
[39] E.Y. Xie, C. Li, S. Yu, J. Lü, On the cryptanalysis of Fridrich's chaotic image encryption scheme, Signal Process. 132 (2017) 150–154. https://doi.org/10.1016/j.sigpro.2016.10.002.
[40] L. Xu, Z. Li, J. Li, W. Hua, A novel bit-level image encryption algorithm based on chaotic maps, Opt. Lasers Eng. 78 (2016) 17–25.
[41] Y.-G. Yang, J. Tian, H. Lei, Y.-H. Zhou, W.-M. Shi, Novel quantum image encryption using one-dimensional quantum cellular automata, Inf. Sci. (Ny) 345 (2016) 257–270.
[42] E. Zeraoulia, Robust Chaos and its Applications, 79, World Scientific, 2012.
[43] Y.-Q. Zhang, X.-Y. Wang, A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice, Inf. Sci. (Ny) 273 (2014) 329–351.
[44] N. Zhou, H. Jiang, L. Gong, X. Xie, Double-image compression and encryption algorithm based on co-sparse representation and random pixel exchanging, Opt. Lasers Eng. 110 (2018) 72–79.
[45] N. Zhou, S. Pan, S. Cheng, Z. Zhou, Image compression–encryption scheme based on hyper-chaotic system and 2D compressive sensing, Opt. Laser Technol. 82 (2016) 121–133.
[46] Y. Zhou, L. Bao, C.L.P. Chen, Image encryption using a new parametric switching chaotic system, Signal Process. 93 (11) (2013) 3039–3052.
[47] Y. Zhou, L. Bao, C.L.P. Chen, A new 1D chaotic system for image encryption, Signal Process. 97 (2014) 172–182.
[48] Z.-L. Zhu, W. Zhang, K.-W. Wong, H. Yu, A chaos-based symmetric image encryption scheme using a bit-level permutation, Inf. Sci. (Ny) 181 (6) (2011) 1171–1186.