Signal Processing: *Image Communication* 29 (2014) 902–913

Contents lists available at ScienceDirect

Signal Processing: *Image Communication*

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

Chaotic image encryption based on circular substitution box and key stream buffer

Xuanping Zhang [a], Zhongmeng Zhao [a,*], Jiayin Wang [b]

[a] *Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China*
[b] *The Genome Institute, Washington University in St. Louis, St. Louis, MO 63108, USA*

ARTICLE INFO

Article history:
Received 20 December 2013
Received in revised form
4 June 2014
Accepted 26 June 2014
Available online 3 July 2014

Keywords:
Image encryption
Chaos
S-box
Substitution
Diffusion

ABSTRACT

A new image encryption algorithm based on spatiotemporal chaotic system is proposed, in which the circular S-box and the key stream buffer are introduced to increase the security. This algorithm is comprised of a substitution process and a diffusion process. In the substitution process, the S-box is considered as a circular sequence with a head pointer, and each image pixel is replaced with an element of S-box according to both the pixel value and the head pointer, while the head pointer varies with the previous substituted pixel. In the diffusion process, the key stream buffer is used to cache the random numbers generated by the chaotic system, and each image pixel is then enciphered by incorporating the previous cipher pixel and a random number dependently chosen from the key stream buffer. A series of experiments and security analysis results demonstrate that this new encryption algorithm is highly secure and more efficient for most of the real image encryption practices.

1. Introduction

Digital images have some well-known intrinsic properties such as bulk data capacity, high redundancy, strong correlation among adjacent pixels [1]. Due to these features, traditional text encryption schemes are no longer suitable for image encryptions [2]. Along with the rapid growth of requests for image transmission through open networks, the security of digital image has become an imperative issue [3], and attracted much attention of researchers. Various encryption schemes [4–7] have been developed based on different techniques which are summarized in a comprehensive review in [8]. Among the existing state-of-the-art approaches, chaos-based methods are extensively investigated and a large number of chaos-based encryption algorithms have been proposed [9–13]. This is because chaotic systems have good features of aperiodicity, pseudorandomness, and high sensitivity to initial conditions.

In chaos-based encryption schemes, chaotic maps are generally used as one time pad for encrypting messages. Since image encryption schemes based on low dimensional chaotic map have low computational complexity, they can be analyzed with low computational cost using iteration and correlation functions [14]. Recently, many researchers adopted the high dimensional chaotic maps to improve chaos-based cryptosystems with enlarged key space and long periodicity of chaotic system [15–18]. Nevertheless, some cryptosystems based on high dimensional chaotic maps or spatiotemporal chaotic maps are still not acceptably secure [19–22], where one main reason is that the key stream generated is independent from the plain image and the cipher image [23]. To overcome this weakness, a key stream buffer is introduced in [24] to

* Corresponding author. Tel.: +86 2982668645; fax: +86 2982668971.
E-mail address: zmzhao@mail.xjtu.edu.cn (Z. Zhao).

http://dx.doi.org/10.1016/j.image.2014.06.012
0923-5965/

cache the random numbers generated by the chaotic system, which makes the generated key stream related to the images.

A chaos-based image encryption scheme basically comprises iterations of the confusion and the diffusion [25]. In each iteration, Substitution box (S-box), Permutation box (P-box) or other nonlinear operations are utilized to provide confusion and diffusion. S-box is a type of basic nonlinear components for symmetric key algorithms. Benefiting from its properties, e.g. nonlinearity, differential uniformity and strict avalanche criterion, S-box has been widely used in new encryption strategies. Many researchers have shown attention to utilizing chaotic nonlinear properties to design S-boxes. Lots of S-box construction algorithms [26–29] have been proposed using different chaotic systems. Furthermore, S-box applications in image ciphers become more popular where S-box is progressively considered as a main function for performing substitution. However, it is found that some S-box-only ciphers are vulnerable to chosen plaintext attacks. Zhang and Xiao [30] carefully studied the security issues for the general S-box-only image ciphers and presented a successful cryptanalysis. In order to obtain the secure cipher images, some S-box-based image encryption algorithms either use the dynamic S-boxes [31,32] or incorporate the S-box with other encryption methods [33,34]. For example, in [31], a block cipher with dynamic S-boxes is studied, in which a tent map is chosen to generate S-box that is required in the confusion phase, and then a left-cyclic-shift operation is used for diffusion step. Wang and Wang [32] also proposed a chaotic image encryption algorithm based on dynamic S-boxes, where the image pixels are divided into several groups and each group uses a new S-box generated separately, according to the plain image. In [33,34], image encryption algorithms are proposed by combining S-box transformation with permutation-diffusion scheme, where S-boxes are used for substituting pixels to provide extra confusion.

In this paper we propose a novel secure image encryption scheme based on spatiotemporal chaotic system. The chaotic system constituted by the logistic map and the piecewise linear chaotic map (PWLCM) is adopted to produce random numbers. The proposed scheme consists of a substitution process and a diffusion process. In the substitution process, the S-box is considered as a circular sequence with a head pointer to the start position. For each image pixel, it is replaced with an element of S-box, according to both the pixel value and the head pointer to obtain a cipher pixel. Then the head pointer is reset following the cipher pixel. By this way, each image pixel can be substituted using a different S-box. During the diffusion process, the key stream buffer is used to cache the random numbers generated by the chaotic system. For enciphering an image pixel, a random number is chosen dependently to the previous cipher pixel. Thus, the key stream used in diffusion process is highly dependent on the image. Experimental results on various types of security analysis indicate that the proposed scheme is robust against various common attacks and performs higher encryption speed.

The rest of this paper is organized as follows: Section 2 describes our new image encryption scheme in detail. Section 3 shows results of experiments which demonstrate the performance by various security analysis. We compare our scheme with some existing chaos-based encryption schemes in Section 4, and the conclusion of this paper is given in Section 5.

2. The proposed cryptosystem

The architecture of the proposed scheme is shown in Fig. 1. The pipeline of our scheme is as follows: initially, the external secret key and the size of plain image are utilized to produce the initial states and parameters of the chaotic system. The pseudo-random number generator iterates the chaotic system with the initial states and generates the random numbers for image encryption. Then, the encryption process which consists of substitution and diffusion enciphers each pixel of the image using these random numbers.

2.1. Pseudo-random number generator

The pseudo-random number in encryption process is produced by a spatiotemporal chaotic system. This system is modeled by the Coupled Map Lattice (CML). A CML is an array of states whose values are continuous (usually within the unit interval) or discrete space and time. The CML model adopted in this paper is a two-dimensional dynamical map, which can be described as following [35]:

$$\begin{cases} x_{i+1} = (1-\beta) f_1(x_i) + \beta f_2(y_i) \\ y_{i+1} = (1-\beta) f_1(y_i) + \beta f_2(x_i) \end{cases} \tag{1}$$

where the parameter β controls the strength of the coupling, while the functions f_1 and f_2 are the chaotic maps. Let f_1 be the logistic map and f_2 be the piece-wise linear chaotic map (PWLCM), which are represented as follows, separately:

$$f_1(x) = \alpha x(1-x) \tag{2}$$

$$f_2(x) = \begin{cases} x/\gamma, & 0 \le x < \gamma \\ (x-\gamma)/(0.5-\gamma), & \gamma \le x < 0.5 \\ f_2(1-x), & 0.5 \le x < 1 \end{cases} \tag{3}$$

where $x \in [0,1]$ is the state variable, both $\alpha \in (0,4]$ and $\gamma \in (0,0.5)$ are control parameters.

The initial conditions of the above chaotic system include initial state denoted by (x_0, y_0) and three parameter

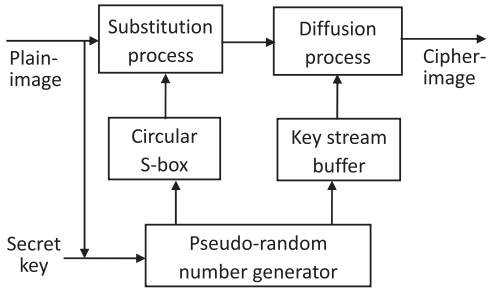

Fig. 1. Architecture of the proposed cryptosystem.

α, β and γ. Note that, there are two two-tuples of initial states, denoted by (sx_0, sy_0) and (dx_0, dy_0) corresponding to the substitution process and the diffusion process, respectively. The initial conditions are computed based on a 280-bit-long external secret key and the size of plain image.

First, the external secret key is divided into 40-bit-long blocks k_i $(i=1,2,\ldots,7)$. Each k_i is considered as an integer number. Using Eq. (4), we can calculate an integer number k which depends on each bit of external key

$$k = \bigoplus_{i=1}^{7} cycL(k_i, 5i) \tag{4}$$

where $\oplus$ denotes bitwise XOR operation and $cycL(x,y)$ denotes a left cyclic shift of x by y bits.

Then, each k_i $(i=1,2,\ldots,7)$ is modified by

$$k_i = k_i \oplus k \tag{5}$$

Since each k_i is 40-bit-long, its value is in the range from 0 to 2^{40}. So we can convert k_i into a real number $R_i \in (0,1)$ using the following equation:

$$R_i = (k_i + n)/(2^{40} + n) \tag{6}$$

where n $(n>0)$ is the size of the plain image. Now, the initial states and parameters of the chaotic system can be generated as follows:

$$sx_0 = R_1, \quad sy_0 = R_2, \quad dx_0 = R_3, \quad dy_0 = R_4,$$
$$\alpha = 3.99 + 0.01R_5, \quad \beta = R_6, \quad \gamma = 0.5R_7$$

From the above process, we can find that the generated initial states and parameters satisfy the constraints of the CML described by Eqs. (1)–(3), and are relevant to the image size and each bit of external key, so the key sensitivity is enhanced.

Given two initial states x_0 and y_0 to the CML, a series of new states x and y can be generated by iterating the CML. The values of the CML states are floating-point numbers, but the pseudo-numbers are usually required in the form of an integer in encryption. Thus, a conversion process from floating-points to integers is necessary. Although couples of complex methods can convert a floating-point number into an integer, e.g. in [36,37], we prefer the simple ones. Complicating the approach at this step is not smart; actually, the simple method we use is effective enough but is more efficient. Thus, first iterate the CML once to obtain the new state values x and y, then generate 4 numbers d_1,d_2,d_3,d_4 using the following formula:

$$\begin{cases} d_1 = floor(2^{32} \cdot x) \bmod 256 \\ d_2 = floor(2^{32} \cdot y) \bmod 256 \\ d_3 = floor(2^{24} \cdot x) \bmod 256 \\ d_4 = floor(2^{24} \cdot y) \bmod 256 \end{cases} \tag{7}$$

where function $floor(x)$ returns the nearest integer less than or equal to x, mod is the modular operator. These 4 numbers are supplied to the encryption process one after another when needed. After these numbers are used, the CML iterates again to generate another 4 new random numbers.

2.2. Substitution process

Substitution is a nonlinear transformation which performs confusion on the input image pixels. A nonlinear transformation is essential for any modern encryption algorithms and is proved to be a strong cryptographic primitive against linear or differential cryptanalysis. Substitution can be implemented as the form of a lookup table, which is also called S-box in many literatures. In our scheme's substitution process, the S-box is first constructed using the generated random numbers, and then the value of each plain image pixel is replaced with another by the S-box transformation.

2.2.1. Construction of S-box

Mathematically, an $M \times N$ S-box is a nonlinear mapping from V_M to V_N, where V_M and V_N represent the vector spaces of the elements in the M and N tuples from $GF(2)$, respectively. Several methods are designed for constructing cryptographically strong S-boxes. In our proposed image encryption scheme, we use 8×8 S-box and present a simple but fast method to construct the S-box. The outline of our method is described as follows:

(1) Set initial states of the chaotic system to (sx_0, sy_0). Iterate the chaotic system for CL times to get rid of the transient effect, where CL is a constant.
(2) Assume that $S = \{S[0], S[1], \ldots, S[255]\}$ is the S-box that needs to be generated, $Q = \{Q[0], Q[1], \ldots, Q[255]\}$ is a sequence of integers and each $Q[k]$ is initialized to k.
(3) Set $i=0$.
(4) Generate a random number d and calculate an index j of the sequence Q.

$$j = d \bmod (256 - i)$$

(5) Let $S[i] = Q[j]$, $Q[j] = Q[255-i]$, and $i = i+1$.
(6) Repeat steps (4) and (5) until $i > 255$.

After this procedure, we obtain an S-box in the form of a sequence $S = \{S[0], S[1], \ldots, S[255]\}$, where each element $S[k] \in [0, 255]$ is unique to others in S, so S is bijective. Fig. 2 shows an instance of an S-box generated by this method.

To evaluate the performance of the generated S-box, the nonlinearity, strict avalanche criterion (SAC), output bits independence criterion (BIC), differential approximation probability (DP) and linear approximation probability (LP) are calculated and listed in Table 1, where column 2 shows the performance of the generated S-box shown in Fig. 2, and column 3 shows the average performance of 500 S-boxes randomly generated by our method. From Table 1, we can see that our method has nearly as good performance as the schemes proposed in [27,28]. Therefore, this method can be used to generate ideal S-box for encryption.

2.2.2. Substitution based on the circular S-box

The plain image P with size n can be considered as a sequence of pixels $\{p_1,p_2,\ldots,p_n\}$ ordered from the left-most pixel to the right-most pixel per line, and from the top line

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	d9	54	6d	c5	80	0f	e9	35	3d	e4	73	6e	f0	ce	2b	16
1	78	f5	bb	b8	18	04	61	81	ec	84	fa	62	e6	21	a7	5b
2	68	a5	f9	6c	64	4e	23	95	ef	d6	c3	01	a4	8b	91	5e
3	eb	48	67	06	d7	0d	45	97	28	a1	f1	7b	bc	72	19	7f
4	a8	4b	e5	a3	b4	2c	ca	2e	c7	15	13	96	cb	cc	e2	fd
5	38	ae	37	dc	20	00	55	93	30	09	c4	f7	df	1a	07	17
6	b7	33	41	b1	cf	fe	83	e0	af	7d	ad	e1	88	1b	8d	d3
7	b3	58	5d	82	14	12	90	76	8f	63	2d	25	0c	39	85	1c
8	d5	1f	89	98	99	b6	d1	69	aa	92	a0	52	44	31	7c	22
9	c0	57	9d	74	de	70	3f	10	9b	05	a9	3e	9f	77	9c	a2
a	f2	b5	b9	c1	86	db	5c	47	c8	2a	c9	b2	94	36	87	fb
b	71	d0	da	5f	f4	cd	75	6f	53	65	66	e7	34	d2	fc	be
c	4f	bf	79	dd	f6	7e	d8	e3	8e	ba	c6	f3	4a	5a	46	49
d	9e	ea	bd	d4	6b	56	4d	9a	3b	24	e8	a6	b0	6a	11	50
e	08	43	59	03	42	8a	29	4c	27	ab	7a	2f	8c	1d	f8	51
f	3a	26	c2	ee	60	0b	ed	ac	0a	32	40	0e	3c	1e	ff	02

Fig. 2. An S-box instance (in hexadecimal format).

Table 1
Performance of the generated S-box.

Performance tests	S-box of Fig. 2	Average of 500 S-boxes	Scheme in Ref. [27]	Scheme in Ref. [28]
Nonlinearity	108.00	113.10	104.88	108.00
SAC	0.5007	0.4965	0.4844	0.4922
BIC	112.00	109.36	103.82	103.36
DP	0.0468	0.0398	0.0391	0.0391
LP	0.1390	0.1439	0.1289	0.1406

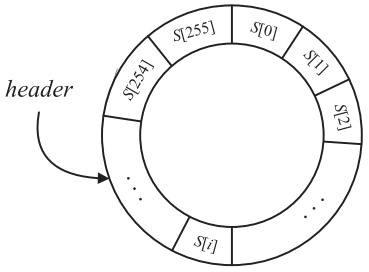

Fig. 3. The circular S-box.

to the bottom line. After an S-box is constructed, it can be used to substitute for the image pixels. In order to break down the correlations between adjacent pixels, ideally, each pixel should be substituted using a different S-box. However, this is extremely time-consuming and thus is unrealistic to construct different S-boxes for every pixel. In this paper, the S-box is considered to be a circular sequence shown in Fig. 3, in which the first element in the sequence is assumed to follow the last element. A head pointer *header* is set to the start position of the circular sequence, which is initialized to a constant or a random generated number.

To encrypt the plain image, we replace each pixel p_i with an element in the S-box S according to both the values of *header* and p_i, and then obtain a cipher pixel. After a pixel is enciphered, the head pointer *header* is set to a new value by considering both the cipher pixel and a random number. Suppose that p'_i is the cipher pixel corresponding to the original pixel p_i, the substitution for each p_i is

$$\begin{cases} p'_i = S[(header + p_i) \bmod 256] \\ header = p'_i \oplus m \end{cases} \tag{8}$$

where m is a random number for masking the pixel.

In the decryption process, an inverse S-box S' is derived from S satisfying $S'[S[k]] = k$. Then, each p_i can be recovered by the following formula:

$$\begin{cases} p_i = (S'[p'_i] + 256 - header) \bmod 256 \\ header = p'_i \oplus m \end{cases} . \tag{9}$$

By introducing the circular S-box, the substitution for a pixel depends on not only the pixel value itself, but the previous cipher pixel as well. The aforementioned method can achieve an ideal avalanche effect and makes the substituted image have a high randomness. The performance of this method will be carefully discussed and demonstrated in Section 3 through different sets of experiments and security analysis.

2.3. Diffusion process

The function of the diffusion process is to modify the pixel value sequentially so that a small change in one pixel can spread out to as many pixels as possible, hopefully can affect the whole image. In order to enable the key stream used in the diffusion process to have high dependence on the image, in our scheme we use the key stream buffer, which is proposed in [24], to cache the random numbers generated by a pseudo-random number generator.

2.3.1. Key stream buffer

The key stream buffer is a pool of random numbers. It provides a selection mechanism for the diffusion process which chooses the random number based on the pixels. The diffusion process takes advantage of two operations to manipulate the key stream buffer. One operation is $init(x_0, y_0)$ which initializes the key stream buffer, while the other one is $Get(i)$ that selects the i-th random number from the buffer.

The operation $init(x_0,y_0)$ first builds up a storage space (the buffer) with the size of BL to cache the generated random numbers, where BL is a constant. Next, this operation sets the initial states of the chaotic system to (x_0,y_0) and iterates the system for constant CL times to get rid of the transient effect. Then it performs pseudo-random number generation to produce BL random numbers and stores them in the buffer.

The operation $Get(i)$ takes the i-th random number out of the buffer and returns it to diffusion process, then the operator replaces the i-th random number in the buffer with a new generated one.

In our encryption scheme, the selection of a random number from the buffer is dependent on the previous cipher-pixel which has the value from 0 to 255. In order to easily and directly access the random numbers according to the image pixels, we set the buffer size BL to 256.

2.3.2. diffusion based on key stream buffer

The diffusion process is performed on the image that has shuffled by substitution process. It changes each pixel value from the first pixel to the last one. At the start point of the diffusion process, the key stream buffer is initialized by the operation $init(dx_0,dy_0)$. During enciphering a pixel, a random number is chosen from the key stream buffer according to the previous pixel, and this chosen random number is used to encipher the current pixel. Suppose that $P'=\{p'_1,p'_2,\ldots,p'_n\}$ is the image that needs to be diffused and $C=\{c_1,c_2,\ldots,c_n\}$ is the diffused image corresponding to P'. The diffusion operation can be represented as follows:

$$c_i=[(p'_i \oplus c_{i-1})+Get(c_{i-1})] \bmod 256, \quad \text{for } i=1,2,\ldots,n \tag{10}$$

where $Get(k)$ denotes the function of selecting the k-th random number from key stream buffer and c_0 is a constant or a random number. Similarly, we can deduce the inverse diffusion operation as

$$p'_i=[(c_i-Get(c_{i-1})+256) \bmod 256] \oplus c_{i-1}. \tag{11}$$

Based on the key stream buffer, the random number used for enciphering a pixel depends on the previous cipher-pixel. Because the elements in the buffer change according to the previous cipher pixels, the sequence of random numbers used in diffusion process is obviously different from the one generated by the random number generator, and thus varies for different images. Therefore, in our method the key stream highly depends on the image, and thus can achieve a potential avalanche effect.

2.4. Encryption and decryption algorithms

Our encryption approach consists of the substitution process and the diffusion process. Initially, the initial states and parameters of the chaotic system are generated by the external secret key and plain image size. In the substitution process, to achieve higher security, the image pixels are randomly divided into a set of groups G_i $(i=1,2,\ldots,t)$ with the length of l_i such that $\sum_{i=1}^{t} l_i=n$, $l_i \in [L,2L)$ $(1\le i<t)$ and $l_t<2L$, where L is a pre-set minimum group length. For each group, a brand-new S-box is constructed and performed.

In order to determine the value of L, we choose several images for test, where each image has the identical pixels. For different value of L, these test images are substituted using Eq. (8), then the average information entropy, local Shannon entropy [38] and correlation between adjacent pixels are carried out to evaluate the performance of substitution. The test results are shown in Table 2. We can find that the smaller the value of L is, the better performance of substitution will be achieved, and when $L\le 4096$, the substitution can obtain an acceptable security. However, the substitution process employing smaller groups requires more S-boxes to be generated. Taking into account the substitution speed, we set L to 4096. Thus, our encryption algorithm can be summarized as follows:

(1) Generate the initial states and parameters for the chaotic system, which includes sx_0, sy_0, dx_0, dy_0, α, β and γ.

Table 2
Performance of substitution with different group sizes.

Minimum group size	Entropy	Local entropy	Correlation
256	7.98504	7.87066	0.00855
512	7.98449	7.84500	0.01391
1024	7.97761	7.72838	0.01989
2048	7.96381	7.61403	0.02429
4096	7.95497	7.58338	0.02846
8192	7.86332	7.14062	0.03850
16,384	7.76149	6.89267	0.04222
32,768	7.65251	6.74489	0.05365

(2) Let $i=n$ and the chaotic system states (x, y) be (sx_0, sy_0).
(3) Construct an S-box S as described in Section 2.2, and set *header* and m to the random numbers produced by pseudo-random number generator.
(4) Generate two new random numbers r_1 and r_2, and calculate the length l of the group as

$$l=\begin{cases} L+(r_1+256r_2) \bmod L & \text{if } i\ge 2L \\ i & \text{if } i<2L \end{cases}.$$

(5) Successively substitute the pixels $p_i,p_{i-1},\ldots,p_{i-l+1}$ using Eq. (8) to obtain the substituted pixels p'_i, $p'_{i-1},\ldots,p'_{i-l+1}$.
(6) Let $i=i-l$, $z=p'_i \oplus p'_{i-1} \oplus \cdots \oplus p'_{i-l+1}$. Update the states (x, y) of the chaotic system by Eq. (12) to make the chaotic state related to the cipher pixels.

$$\begin{cases} x=(1-\varepsilon)x+\varepsilon z \\ y=(1-\varepsilon)y+\varepsilon z \end{cases} \tag{12}$$

where $\varepsilon \in (0,1)$ is the factor to perturb the chaotic system. Because chaotic system has the property of high sensitivity to initial conditions, any change in the chaotic state will alter the trajectory of the chaotic system. So we arbitrarily set $\varepsilon=0.3$ in this paper.
(7) Repeat steps (3)–(6) until $i\le 0$.
(8) Initialize the key stream buffer through the operation $init(dx_0, dy_0)$, and perform diffusion as represented by Eq. (10) on the substituted image P', thus obtain the cipher image C.

In the above algorithm, step (1) generates the initial conditions of the chaotic system, steps (2)–(7) perform substitution on the plain image P to compute a substituted image P', and finally step (8) performs diffusion operations on the substituted image P' to obtain the cipher image C. The decryption algorithm is similar to the encryption algorithm; however, the only difference is that the inverse diffusion described by Eq. (11) should be performed first, then the inverse substitution represented by Eq. (9) can follow.

3. Experiments and security analysis

An effective encryption algorithm should be robust against all kinds of known attacks. In order to measure the performance of the proposed scheme, we choose six

Fig. 4. Test images: (a) Cameraman (256 × 256), (b) Peppers (4096 × 4096), (c) Lena (512 × 512), (d) White (640 × 480), (e) Flower (1024 × 768) and (f) Black (1600 × 900).

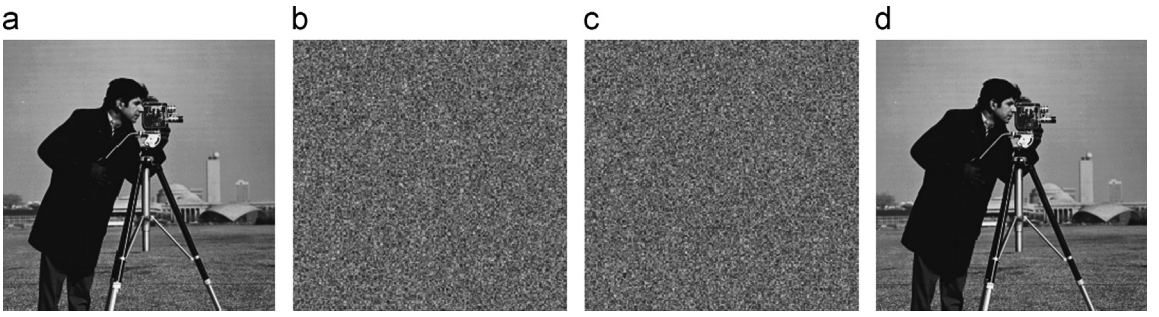

Fig. 5. The plain and cipher images. (a) Plain, (b) substituted, (c) encrypted and (d) decrypted.

sample images with 256 gray level, shown in Fig. 4, where Fig. 4(d) is the white image with the same pixel value 255 and Fig. 4(f) is the black image with the same pixel value 0. These images are then encrypted by the proposed approach. Various analysis, including statistical analysis, differential attack analysis and key security analysis are carried out to demonstrate the performance and the robustness of our image encryption scheme.

3.1. Encryption effect

In the experiment, we apply our scheme to the sample images. The external secret key is "abcdefghijklmnopqrstuvwxyz123456789". Fig. 5 shows the encryption and decryption results of the image *Cameraman* with the size of 256 × 256, where Fig. 5(b) is the enciphered image only by substitution. From the figure, it is clear that the image encrypted by our proposed approach is similar to a random noisy image.

To evaluate the randomness of the encrypted images, we encrypt the image *Peppers* with the size of 4096 × 4096 and apply NIST 800-22 [39] randomness test to the encrypted images. The test results are given in Table 3. From Table 3, we find that the encrypted image by substitution process only can pass all the tests in NIST. Comparing pass rates between substituted image and encrypted image, we can see that the pass rates of encrypted image are further improved by the diffusion process.

Table 3
Randomness tests of substituted image and encrypted image.

Tests	Substituted image		Encrypted image	
	Pass rate	Result	Pass rate	Result
Frequency	0.9850	Success	1.0000	Success
Block frequency	1.0000	Success	1.0000	Success
Cumulative sums	0.9800	Success	1.0000	Success
Runs	0.9900	Success	1.0000	Success
Longest run	0.9899	Success	0.9899	Success
Rank	1.0000	Success	1.0000	Success
FFT	0.9899	Success	0.9900	Success
Non-overlapping	0.9887	Success	0.9893	Success
Overlapping	0.9900	Success	0.9899	Success
Universal	0.9825	Success	1.0000	Success
Approximate entropy	0.9999	Success	0.9898	Success
Random excursions	0.9883	Success	0.9945	Success
Random E-variant	0.9939	Success	0.9931	Success
Serial	0.9798	Success	0.9949	Success
Linear complexity	0.9900	Success	0.9899	Success

3.2. Statistical analysis

It is known that an ideal encryption algorithm should be robust against any statistical attack. To prove the robustness of our new scheme, we apply several statistical analysis on the encryption results. The results are collected by calculating the histogram, the information entropy and the correlations of two adjacent pixels in the cipher image. According to the results, we will show that this novel scheme has superior confusion and diffusion properties which strongly resists the statistical attacks.

3.2.1. Histogram of cipher image

In order to resist the statistical attacks, the histogram of the encrypted images should approximate to uniform distribution. Fig. 6 shows the histograms of the plain and the cipher images of *Lena* and *White*. We can see that the histograms of the cipher images are almost uniformly distributed. We use χ^2 test to evaluate the uniformity of the pixel-value distribution. The χ^2 value of a image with

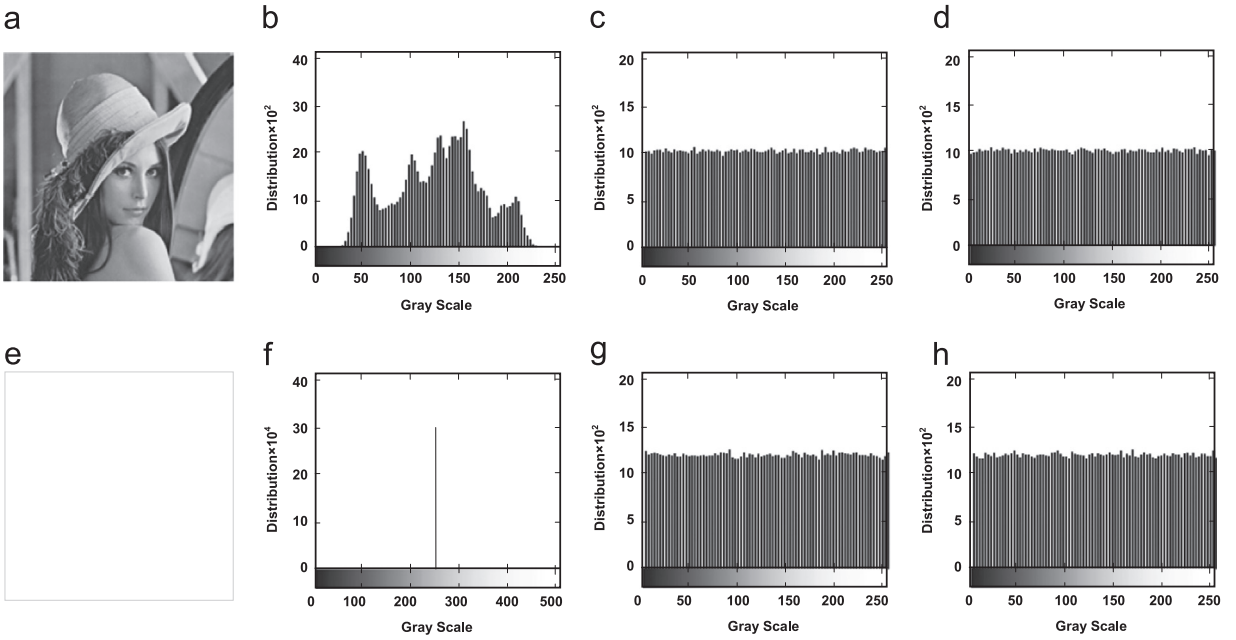

Fig. 6. Histograms: (a) image *Lena*, (b) histogram of plain *Lena*, (c) histogram of substituted *Lena*, (d) histogram of encrypted *Lena*, (e) image *White*, (f) histogram of plain *White*, (g) histogram of substituted *White* and (h) histogram of encrypted *White*.

256 gray levels is computed as

$$\chi^2 = \sum_{i=0}^{255} \frac{(n_i - n/256)^2}{n/256} \tag{13}$$

where n_i is the occurrence frequency of gray level i, $n/256$ is the expected occurrence frequency of each gray level. The χ^2 statistics for different images are listed in Table 4. Using the significant level of 0.05, the corresponding $\chi^2(0.05, 255)$ is 293.25. All the χ^2 statistics of cipher images are less than $\chi^2(0.05, 255)$, which indicates that the histogram distributions of cipher images are significantly uniform.

3.2.2. Information entropy analysis

The information entropy is defined to express the degree of uncertainties or randomness in a given system. The entropy $H(m)$ of an m is calculated as

$$H(m) = \sum_{i=0}^{2^N-1} p(m_i) \log_2 \frac{1}{p(m_i)}, \tag{14}$$

where 2^N is the total number of symbols, $m_i \in m$, and $p(m_i)$ represents the probability of the symbol m_i. For a random image with 256 gray levels, the entropy should ideally be $H(m)=8$. Therefore, any effective encryption algorithm should produce an encrypted image with the entropy close to 8. Table 5 contains the entropies of each pair of the plain image and its cipher image. From Table 5, the entropies of all cipher images are very close to the theoretical optimal value, which demonstrates that the cipher images are almost close to a random source.

The local randomness of the encrypted image has also been tested by using local Shannon entropy which was proposed in [38]. The (k,T_B)-local Shannon entropy is defined as

$$\overline{H}_{k,T_B}(m) = \sum_{i=1}^{k} \frac{H(S_i)}{k} \tag{15}$$

where $S_1, S_2, \ldots, S_k$ are non-overlapping and randomly selected image blocks with T_B pixels. In the experiment, we choose $k=30$ and $T_B=1936$, which are suggested in [38]. Table 6 lists the test results and indicates that the local Shannon entropy of each encrypted image is greater than 7.90. So the image encrypted using our method has the good local randomness.

Table 4
χ^2 statistics of the plain and the cipher images.

Image	Plain	Substituted	Encrypted
Cameraman	110,973.3	259.789	223.633
Peppers	138,836.2	215.240	215.240
Lena	158,063.6	194.365	236.923
White	66,846,720	17,079.1	247.197
Flower	840,309.4	232.175	259.408
Black	33,423,360	69,374.4	229.238

Table 5
Entropies of the plain and the cipher images.

Image	Plain	Substituted	Encrypted
Cameraman	7.0097	7.9971	7.9975
Peppers	7.5715	7.9992	7.9994
Lena	7.4455	7.9995	7.9993
White	0.0000	7.9565	7.9993
Flower	7.1522	7.9998	7.9998
Black	0.0000	7.9229	7.9987

Table 6
Local entropies of the plain and the cipher images.

Image	Plain	Substituted	Encrypted
Cameraman	6.7729	7.8907	7.9103
Peppers	5.6494	7.9025	7.9152
Lena	6.8142	7.8865	7.9137
White	0.0000	7.8278	7.9065
Flower	4.2069	7.8964	7.9147
Black	0.0000	7.6289	7.9051

3.2.3. Correlation analysis

We randomly select 1000 pairs of adjacent pixels from the plain image and its cipher image in vertical, horizontal and diagonal directions, respectively, and then plot out the correlation between them. Fig. 7 shows the correlations between pairs of the adjacent pixels in plain images *Flower*, *Black* and their encrypted images as well. Fig. 7(b) and (d) indicates that strong correlations between adjacent pixels in the plain images are drastically reduced in the encrypted images.

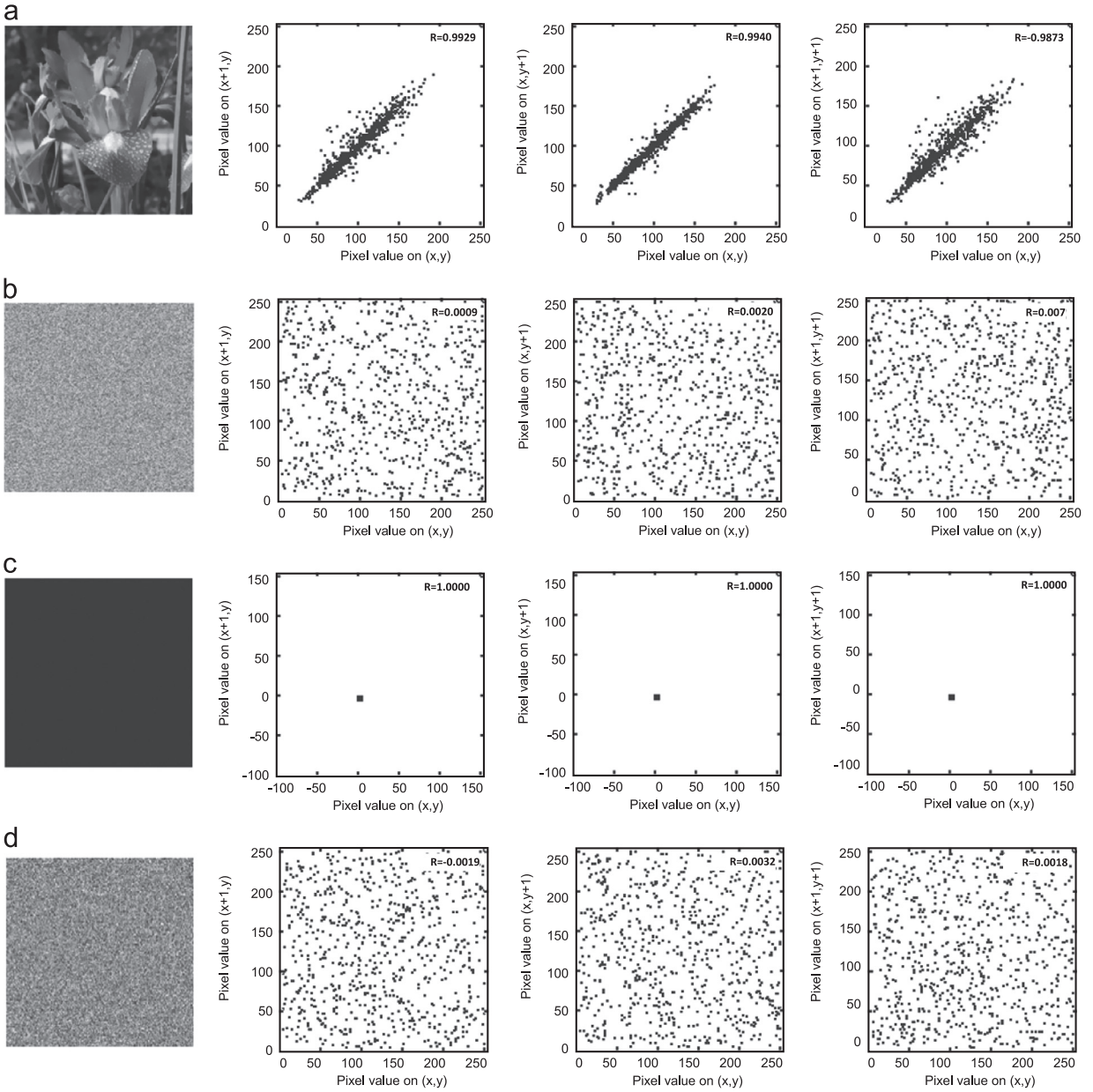

Fig. 7. Images and their horizontal, vertical and diagonal correlations. (a) and (b) The plain and the encrypted images of *Flower*. (c) and (d) The plain and the encrypted images of *Black*.

Table 7
Correlations of the plain and the encrypted images (HC – horizontal correlation, VC – vertical correlation, DC – diagonal correlation).

Image	Pain image			Substituted image			Encrypted image		
	HC	VC	DC	HC	VC	DC	HC	VC	DC
Cameraman	0.93348	0.95922	0.91299	0.00356	−0.00411	0.00061	0.00292	−0.00120	−0.00045
Peppers	0.97917	0.98264	0.96892	−0.00165	−0.00449	0.00156	0.00140	−0.00153	−0.00067
Lena	0.97187	0.98498	0.96387	0.00152	−0.00417	−0.00186	0.00171	−0.00217	−0.00087
White	1.00000	1.00000	1.00000	−0.01988	0.02580	0.00102	−0.00011	0.00336	0.00055
Flower	0.99292	0.99397	0.98728	0.00153	−0.00009	−0.00051	0.00088	0.00203	0.00067
Black	1.00000	1.00000	1.00000	−0.01266	−0.01476	0.02196	−0.00189	0.00317	0.00175

The correlation property can also be quantified by the means of correlation coefficients:

$$r = \frac{cov(x,y)}{\sqrt{D(x)(D(y)}} \tag{16}$$

where

$$cov(x,y) = \frac{1}{M}\sum_{i=1}^{M}(x_i - \overline{x})(y_i - \overline{y}),$$

$$D(x) = \frac{1}{M}\sum_{i=1}^{M}(x_i - \overline{x})^2,$$

where x_i and y_i are two adjacent pixels, M is the total number of the adjacent pixel pairs (x_i, y_i), $\overline{x}$ and $\overline{y}$ denote the mean values of x and y, separately. Table 7 collects the correlation coefficients of the plain and the encrypted images. From Table 7 it is obvious that all of the correlation coefficients of encrypted images are close to zero, which means our approach can effectively remove the correlations among adjacent pixels in plain image.

3.3. Differential attack

In order to resist differential attack, a secure cryptosystem should have high plaintext sensitivity, that is, the output ciphertext of the cryptosystem should change dramatically in response to any small change in plaintext. In the proposed scheme, when a plain image pixel changes, both the substitution process and diffusion process will propagate this change to all its subsequent pixels. Because the substitution process and diffusion process are performed in opposite directions separately, any change of a plain image pixel can be spread over the entire encrypted image. To demonstrate the high plaintext sensitivity of our scheme, we set 0 to the pixel located in position (133, 461) in the image *Lena* to get a modified image, and then encrypt the original image and modified image using our method. The pixel-to-pixel differences between two encrypted images can be obtained as shown in Fig. 8. It can be seen that any slight change in the plain image will cause the significant changes in the encrypted image.

The plaintext sensitivity can be quantitatively evaluated using the number of pixels change rate (NPCR) and unified average changing intensity (UACI). Given two images $x = \{x_1, x_2, \ldots, x_n\}$ and $y = \{y_1, y_2, \ldots, y_n\}$, the NPCR and the UACI are defined as

$$NPCR = \frac{1}{n}\sum_{i=1}^{n} D(x_i, y_i) \times 100\% \tag{17}$$

$$UACI = \frac{1}{n}\sum_{i=1}^{n} \frac{|x_i - y_i|}{255} \times 100\% \tag{18}$$

where $D(x_i, y_i) = 0$ if $x_i = y_i$; otherwise, $D(x_i, y_i) = 1$.

In the experiment, the original plain image and its modified image generated by randomly changing only one pixel in original plain image are encrypted to obtain two cipher images. These two cipher images are then used to calculate the NPCR and UACI. For each sample image in Fig. 4, this test is performed 1000 times, and the average, maximum, minimum values of NPCR and UACI are given in Table 8. As can be seen from Table 8, the NPCR and UACI of encrypted images produced by our algorithm are close to 99.61 and 33.46, which are the average NPCR and UACI of random images. This means that the proposed scheme can effectively resist the differential attack.

3.4. Key security analysis

A good image encryption algorithm should be sensitive to the secret keys, at the same time the key space should be large enough to make any brute-force attacks infeasible. To measure the key sensitivity of our approach, we randomly choose two secret keys with the only one-bit difference to encrypt the plain image, and then calculate the NPCR and the UACI of the encrypted images. To obtain powerful result, for each sample image, we repeat this test 1000 times, and collect the average, the maximum and minimum values of NPCR and UACI, shown in Table 9. As a result, NPCRs and UACIs of the encrypted images are close to the mean values of those random images. Hence, the proposed algorithm is highly sensitive to the changes of a secret key.

In addition, the key space of the proposed scheme is sufficient to resist all kinds of brute-force attacks. An external 280-bit secret key is utilized to generate the initial conditions of the chaotic system, thus the key space is $2^{280} \approx 1.943 \times 10^{84}$, which satisfies the general requirement of resisting brute-force attacks.

4. Comparison with existing schemes

In this section, we compare our scheme with several existing chaos-based schemes. We choose those newly

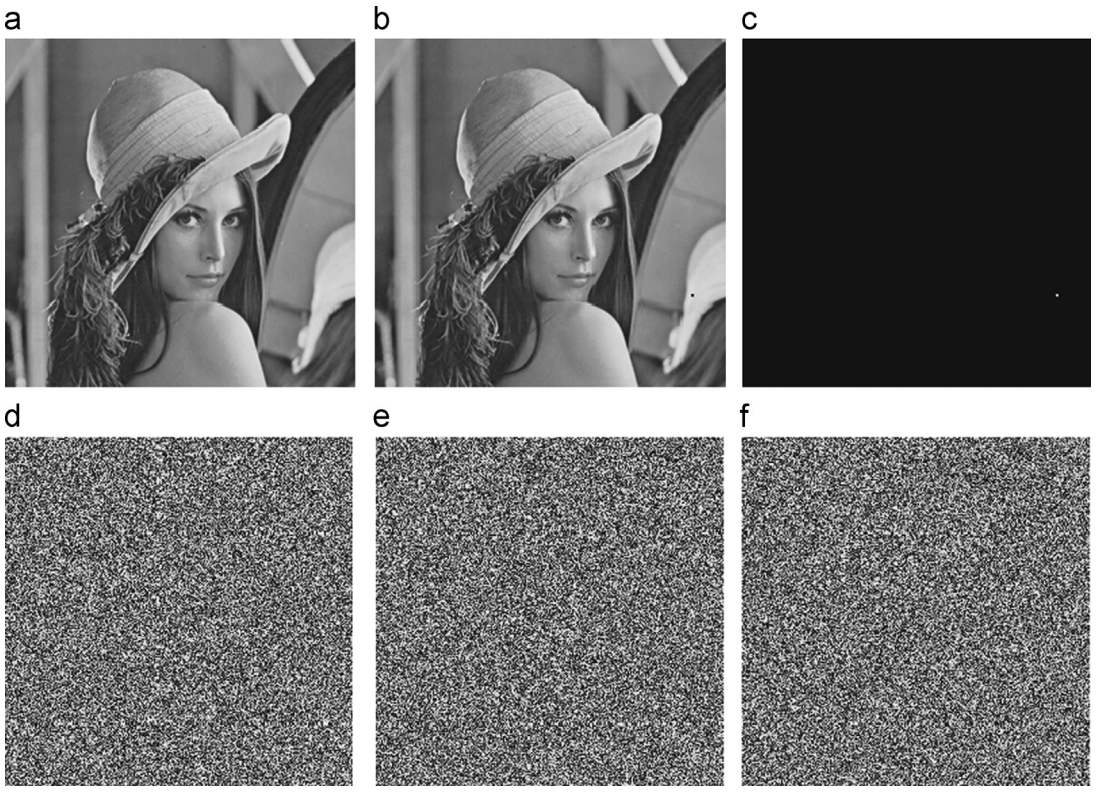

Fig. 8. The impact of a single pixel change in original image. (a) Original image, (b) modified image, (c) image differences between the original and modified images, (d) encrypted image of (a), (e) encrypted image of (b) and (f) image differences between (d) and (e).

Table 8
Plaintext sensitivity of the proposed scheme.

Image	Average		Maximum		Minimum	
	NPCR	UACI	NPCR	UACI	NPCR	UACI
Cameraman	99.6059	33.4594	99.6643	33.7007	99.5499	33.3542
Peppers	99.6080	33.4713	99.6399	33.6005	99.5804	33.3750
Lena	99.6083	33.4554	99.6456	33.5787	99.5808	33.3518
White	99.6090	33.4551	99.6403	33.5606	99.5804	33.3666
Flower	99.6095	33.4601	99.6230	33.5045	99.5942	33.3754
Black	99.6067	33.4805	99.6437	33.6397	99.5430	33.3843

proposed image encryption schemes, which were proposed in [13,24,32–34]. In the comparison, we focus on the statistical analysis, plaintext sensitivity and encryption speed, which are also compared in the previous literatures. The schemes proposed in [32–34] are based on S-box, while the schemes in [13,24] claim to have faster speeds.

We encrypt the image *Lena* using different algorithms, and calculate the histograms, the information entropies and the correlation coefficients of two adjacent pixels in the encrypted images. The comparison results are summarized in Table 10. We can see that our scheme has the competitive performance comparing with the other five existing schemes.

In order to compare the plaintext sensitivities for the different encryption methods, the image *Lena* and its modified image generated by randomly changing a single pixel in original plain image are encrypted for several rounds using each algorithm. After each round of encryption, both NPCR and UACI of encrypted images are calculated and the averages of NPCR and UACI are listed in Table 11. The test results support that the approach we proposed has at least the same excellent performance as the other algorithms in [13,24,32], while the algorithms in [33,34] require additional multiple rounds of encryption to achieve a desired effect.

The speed of the proposed scheme is also compared on the same platform, which is the Microsoft VC++ programming on a personal computer with 3.10 GHz Intel(R) Core(TM) i5-2400 CPU and 4 GB memory running on Microsoft Windows 7. We run our scheme and the other five algorithms for 100 times on 256 gray-scale images with different sizes. The average encryption time for

Table 9
Key sensitivity of the proposed scheme.

Image	Average		Maximum		Minimum	
	NPCR	UACI	NPCR	UACI	NPCR	UACI
Cameraman	99.6131	33.4581	99.6689	33.6550	99.5529	33.3913
Peppers	99.6080	33.4713	99.6399	33.6005	99.5804	33.3750
Lena	99.6128	33.4594	99.6464	33.5442	99.5796	33.3759
White	99.6086	33.4572	99.6441	33.5259	99.5800	33.3591
Flower	99.6101	33.4591	99.6248	33.4980	99.5977	33.3895
Black	99.6087	33.4732	99.6460	33.6134	99.5689	33.3461

Table 10
Comparison on statistical analysis.

Encryption scheme	Histogram (χ^2)	Entropy	Local entropy	Correlation		
				Horizontal	Vertical	Diagonal
Algorithm in [13]	392.63	7.9989	7.9089	0.0001	0.0031	−0.0043
Algorithm in [24]	288.47	7.9992	7.9070	0.0001	−0.0012	0.0012
Algorithm in [32]	391.10	7.9892	7.8706	0.0025	−0.0061	0.0002
Algorithm in [33]	233.72	7.9994	7.9025	0.0010	−0.0008	−0.0005
Algorithm in [34]	236.01	7.9993	7.9088	0.0003	−0.0047	0.0005
Proposed algorithm	236.92	7.9993	7.9137	0.0017	−0.0022	-0.0009

Table 11
Comparison on plaintext sensitivity.

Algorithm	*round*=1		*round*=2		*round*=3	
	NPCR	UACI	NPCR	UACI	NPCR	UACI
Ref. [13]	99.5771	33.4936	99.5872	33.4634	99.5763	33.4742
Ref. [24]	99.6052	33.4111	99.6143	33.4654	99.6276	33.4599
Ref. [32]	99.5761	33.7153	99.6019	33.4523	99.6218	33.4687
Ref. [33]	27.2237	9.1527	99.6099	33.4660	99.6119	33.4863
Ref. [34]	28.2829	9.4960	99.6020	33.4661	99.6096	33.4721
Ours	99.6083	33.4554	99.6103	33.4612	99.6128	33.4626

Table 12
Encryption time among different algorithms (in ms).

Size	Ref. [13]	Ref. [24]	Ref. [32]	Ref. [33]	Ref. [34]	Ours
256×256	3.3960	2.1371	1.5446	12.408	12.582	1.5039
512×512	7.8713	8.5686	5.0891	52.857	53.816	4.9020
1024×1024	25.9505	35.1765	22.7129	229.531	232.408	19.5686
2048×2048	99.6238	136.725	117.842	1015.59	1025.78	77.6863
3072×3072	222.723	357.274	290.228	2475.39	2494.08	173.765
4096×4096	394.792	669.882	1104.82	4515.73	4556.80	308.314

images with different sizes are listed in Table 12. Obviously, our scheme has higher encryption speed.

5. Conclusion

In this paper, we propose a novel chaotic image encryption approach with the substitution–diffusion structure. The circular S-box and the key stream buffer are introduced to improve the security. In the substitution process, the circular S-box is used to obtain the highly random confused image and to achieve the ideal avalanche effect. In the diffusion process, the key steam buffer makes the key stream different from the random sequence generated by chaotic system, and greatly dependent on the image. The experimental results and various analysis on security demonstrate that the proposed approach can achieve higher security level to resist various common attacks, such as statistical attack, differential attack and

brute-force attack. In the meantime, our scheme performs faster than the existing approaches and more practical for real image encryption.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant no.: 61100239), the Ph.D. Programs Foundation of Ministry of Education of China (Grant no.: 20100201110063), the Shaanxi Provincial Natural Science Foundation of China (Grant nos.: 2014JM8322, 2014JM8350).

References

[1] Y. Wang, K. Wong, X. Liao, G. Chen, A new chaos-based fast image encryption algorithm, Appl. Soft Comput. 11 (1) (2011) 514–522.
[2] M. Ghebleh, A. Kanso, H. Noura, An image encryption scheme based on irregularly decimated chaotic maps, Signal Process.: Image Commun. 29 (5) (2014) 618–627.
[3] Y. Zhou, L. Bao, C. Chen, A new 1D chaotic system for image encryption, Signal Process. 97 (2014) 172–182.
[4] N.K. Pareek, V. Patidar, K.K. Sud, Diffusion substitution based gray image encryption scheme, Digital Signal Process. 23 (3) (2013) 894–901.
[5] R. Enayatifar, A.H. Abdullah, I.F. Isnin, Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence, Opt. Lasers Eng. 56 (2014) 83–93.
[6] 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.
[7] L. Sui, K. Duan, J. Liang, X. Hei, Asymmetric double-image encryption based on cascaded discrete fractional random transform and logistic maps, Opt. Express 22 (9) (2014) 10605–10621.
[8] S. Zhou, Q. Zhang, X. Wei, C. Zhou, A summarization on image encryption, IETE Tech. Rev. 27 (6) (2010) 503–510.
[9] Y. Liu, X. Tong, S. Hu, A family of new complex number chaotic maps based image encryption algorithm, Signal Process.: Image Commun. 28 (10) (2013) 1548–1559.
[10] X. Zhang, L. Shao, Z. Zhao, Z. Liang, An image encryption scheme based on constructing large permutation with chaotic sequence, Comput. Electr. Eng. 40 (3) (2014) 931–941.
[11] G. Ye, Image scrambling encryption algorithm of pixel bit based on chaos map, Pattern Recognit. Lett. 31 (5) (2010) 347–354.
[12] M. François, T. Grosges, D. Barchiesi, R. Erra, A new image encryption scheme based on a chaotic function, Signal Process.: Image Commun. 27 (3) (2012) 249–259.
[13] J.A.E. Fouda, J.Y. Effa, S.L. Sabat, M. Ali, A fast chaotic block cipher for image encryption, Commun. Nonlinear Sci. Numer. Simul. 19 (3) (2014) 578–588.
[14] Y. Zhou, L. Bao, C. Chen, Image encryption using a new parametric switching chaotic system, Signal Process. 93 (11) (2013) 3039–3052.
[15] C. Song, Y. Qiao, X. Zhang, An image encryption scheme based on new spatiotemporal chaos, Optik – Int. J. Light Electron Opt. 124 (18) (2013) 3329–3334.
[16] S.M. Seyedzadeh, S. Mirzakuchaki, A fast color image encryption algorithm based on coupled two-dimensional piecewise chaotic map, Signal Process. 92 (5) (2012) 1202–1215.
[17] 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.
[18] Z. Wang, X. Huang, Y. Li, X. Song, A new image encryption algorithm based on the fractional-order hyperchaotic Lorenz system, Chin. Phys. B 22 (1) (2013) 010504.
[19] 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.
[20] C. Li, S. Li, K.-T. Lo, Breaking a modified substitution–diffusion image cipher based on chaotic standard and logistic maps, Commun. Nonlinear Sci. Numer. Simul. 16 (2) (2011) 837–843.
[21] C. Li, S. Li, G. Chen, W.A. Halang, Cryptanalysis of an image encryption scheme based on a compound chaotic sequence, Image Vis. Comput. 27 (8) (2009) 1035–1039.
[22] R. Rhouma, S. Belghith, Cryptanalysis of a new image encryption algorithm based on hyper-chaos, Phys. Lett. A 372 (2008) 5973–5978.
[23] X. Wang, L. Liu, Cryptanalysis and improvement of a digital image encryption method with chaotic map lattices, Chin. Phys. B 22 (5) (2013) 050503.
[24] X. Zhang, Z. Zhao, Chaos-based image encryption with total shuffling and bidirectional diffusion, Nonlinear Dyn. 75 (1–2) (2014) 319–330.
[25] K.W. Wong, S.H.K. Bernie, C.H. Yuen, An efficient diffusion approach for chaos-based image encryption, Chaos, Solitons Fractals 41 (5) (2009) 265–2663.
[26] G. Chen, Y. Chen, X. Liao, An extended method for obtaining S-boxes based on three-dimensional chaotic baker maps, Chaos, Solitons Fractals 31 (3) (2007) 571–579.
[27] F. Özkaynak, A.B. Özer, A method for designing strong S-boxes based on chaotic Lorenz system, Phys. Lett. A 374 (36) (2010) 3733–3738.
[28] Y. Wang, K.W. Wong, C. Li, Y. Li, A novel method to design S-box based on chaotic map and genetic algorithm, Phys. Lett. A 376 (6) (2012) 827–833.
[29] M. Khan, T. Shah, M.A. Gondal, An efficient technique for the construction of substitution box with chaotic partial differential equation, Nonlinear Dyn. 73 (3) (2013) 1795–1801.
[30] Y. Zhang, D. Xiao, Cryptanalysis of S-box-only chaotic image ciphers against chosen plaintext attack, Nonlinear Dyn. 72 (4) (2013) 751–756.
[31] Y. Wang, K. Wong, X. Liao, T. Xiang, A block cipher with dynamic S-boxes based on tent map, Commun. Nonlinear Sci. Numer. Simul. 14 (7) (2009) 3089–3099.
[32] X. Wang, Q. Wang, A novel image encryption algorithm based on dynamic S-boxes constructed by chaos, Nonlinear Dyn. 75 (3) (2014) 567–576.
[33] I. Hussain, T. Shah, M.A. Gondal, Image encryption algorithm based on total shuffling scheme and chaotic S-box transformation, J. Vib. Control, in press, http://dx.doi.org/10.1177/1077546313482960.
[34] I. Hussain, T. Shah, M.A. Gondal, Application of S-box and chaotic map for image encryption, Math. Comput. Model. 57 (9–10) (2013) 2576–2579.
[35] S. Ahadpour, Y. Sadra, A chaos-based image encryption scheme using chaotic coupled map lattices, Int. J. Comput. Appl. 49 (2) (2012) 15–18.
[36] R. Yin, J. Yuan, Q. Yang, X. Shan, X. Wang, Discretization of coupled map lattices for a stream cipher, Tsinghua Sci. Technol. 16 (3) (2011) 241–246.
[37] I.S. Sam, P. Devaraj, R.S. Bhuvaneswaran, A novel image cipher based on mixed transformed logistic maps, Multimed. Tools Appl. 56 (2) (2012) 315–330.
[38] Y. Wu, Y. Zhou, G. Saveriades, S. Agaian, J.P. Noonan, P. Natarajan, Local Shannon entropy measure with statistical tests for image randomness, Inf. Sci. 222 (2013) 323–342.
[39] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, et al., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST Special Publication 800-22, May 15, 2001.