Nonlinear Dyn (2015) 81:151–168
DOI 10.1007/s11071-015-1979-3

ORIGINAL PAPER

A colour image encryption algorithm using 4-pixel Feistel structure and multiple chaotic systems

Wang Yao · Xiao Zhang · Zhiming Zheng · Wangjie Qiu

Received: 3 March 2014 / Accepted: 15 February 2015 / Published online: 4 March 2015

Abstract In recent years, chaos-based image encryption algorithms have attracted much attention. Particularly, with larger data capacity and higher correlation among pixels, encryption of colour images demands better statistic and diffusion properties of image algorithms than that of grey images. In this paper, a chaos-based algorithm aiming at colour image encryption is proposed. Two 3D chaotic systems are used as key generators for three colours of colour images' pixels. 4-Pixel Feistel structure and functions based on multiple chaotic maps are used to improve the statistic and diffusion properties of cipher image. Dependent encryption progress is used to resist certain cryptanalysis methods, such as known-/chosen plaintext attack and chosen cipher attack. According to large number of simulation experiments, with good speed performance being taken into account, our algorithm has better properties and higher security level than certain other chaos-based colour image encryption algorithms.

Keywords Colour image encryption · Chaos · 4-Pixel Feistel structure · Multiple chaotic systems

W. Yao · X. Zhang (✉) · Z. Zheng (✉)
Key Laboratory of Mathematics, Informatics and Behavioral Semantics, Ministry of Education, and School of Mathematics and Systems Science, Beihang University, Beijing 100191, China
e-mail: 09621@buaa.edu.cn

Z. Zheng
e-mail: zzheng@pku.edu.cn

W.Yao
e-mail: bznh3618@gmail.com

W. Qiu
Educational Equipment Research and Development Center, Ministry of Education, Beijing 100080, China

1 Introduction

In 1989, the first chaos-based encryption algorithm was proposed by the British mathematician Matthews [1], in which the logistic map was used to construct a key generator. Since then, cryptographers have proposed many kinds of chaotic image ciphers [2–8]. Particularly, with larger data capacity and higher correlation among pixels, encryption of colour images demands better statistic and diffusion properties of image algorithms than that of grey images. Thus, colour image encryption is really worth researching.

When designing a scheme, security is a basic property which should be considered. In recent years, cryptographers have proposed many new design methods to improve the security of colour image encryption.

In the earlier stage, a substitution–diffusion-based colour image cipher using chaotic standard and logistic maps was proposed by Patidar et al. [9]. But considering the interaction of different colours, multi-dimensional chaos systems were introduced into the design of colour image encryption scheme. There are many examples, such as the algorithm proposed by Wang et al. [10] which used chaotic system to encrypt the R, G, B com-

ponents of a colour image at the same time and makes these three components affect each other, the algorithm proposed by Kanso et al. [11] in which 3D chaotic cat map was used to scramble shuffled pixels through mixing and masking rules, and the 6-dimensional generalized chaos synchronization system proposed by Han and Min which was based on 3D-Lorenz map to enhance the security of image information [12].

Apart from using multi-dimensional systems, the combination of different chaotic systems is another way used widely by cryptographers to enhance the scheme security. Wu et al. [13] proposed a colour image cryptosystem based on synchronization of two different six-dimensional hyperchaotic systems, which has high security and can resist noise and crop attacks. Wu et al. [14] proposed an algorithm to secure three colour images simultaneously by combining scrambling with the reality-preserving fractional discrete cosine transform. Sankaran et al. [15] proposed a scheme which employs one of the three dynamic chaotic systems (Lorenz or Chen or LU chaotic system selected based on 16-byte key) to shuffle the position of the image pixels (pixel position permutation) and to confuse the relationship between the cipher image and the plain image (pixel value diffusion), thereby significantly increasing the resistance to attacks. Zhang et al. [16] proposed a new scheme based on coupled logistic map, self-adaptive permutation and other structures to strength the security performance.

But relatively, cryptographers focus less on the speed performance. Little of recent papers on colour image encryption made the speed analysis. The most impressing result is that in 2014, Mohamed proposed a new tweakable construction of block-enciphers using second-order reversible cellular automata, which focuses on the performance of RGB-coloured image encryption, which can encrypt a 512*512 image in 0.75s using a Delphi 6 programming environment and an i7-2600 3.40 GHz platform [17].

At the same time, the analysis of chaotic image cipher is also much developed. In [18], the encryption algorithm of Fridirich [19] was cryptanalyzed by a chosen ciphertext attack based on casuality paths. In [20], a novel image encryption scheme based on improved hyperchaotic sequences was broken by known plaintext attack. Considering larger data capacity and higher correlation among pixels of colour images, colour image encryption algorithms are faced with more danger. Soon after the proposal of colour image encryption algorithm PPS09 [9], Rhouma et al. [21] developed an equivalent description of the PPS09 cryptosystem which facilitated it in the cryptanalysis of the original cipher in terms of chosen plaintext and known plaintext attacks. In [22], a chaos-based colour image encryption algorithm [10] which was proposed by cascading two position permutation operations and one substitution operation was broken by chosen plaintext attack. Thus, algorithm for colour image encryption with better statistic and diffusion properties and higher level of security is needed.

In this paper, a chaotic encryption algorithm for colour images is proposed, which uses 4-Pixel Feistel structure, multiple chaotic maps and systems and dependent encryption process to realize good performance. The rest of paper is organized as follows. The Sect. 2 contains the description of construction and the process of the proposed algorithm. Simulation results, performance analysis and cryptanalysis are reported in Sect. 3. Conclusions are drawn in the last section.

2 Algorithm proposed

In this section, the proposed algorithm will be introduced in three levels. The basic level is the functions based on multiple chaotic maps which will be used to construct the round functions. Intermediate level is the block operation including two rounds of 4-Pixel Feistel structure and its round functions. The top level is the dependent encryption process. Otherwise, the key generators will also be presented.

2.1 Functions based on multiple chaotic maps

Functions based on multiple chaotic maps (namely f and f') are designed to realize the diffusion between 2 pixels, to improve the confusion effect of the cipher and to hide the plaintext for avoiding the plaintext participate in calculating directly.

f and f' both have four integer numbers between 0 and 255 as inputs, namely $a, b.1, b.2$ and $b.3$. And the output of $f$is an integer number between 0 and 255. Meanwhile, f and f' are same in the structure, both contains three parts: Parameter generator based on logistic map $LM0$, initial condition generator based on piecewise linear function $PL0$ and output generator based on piecewise linear function $PL1$. The difference of f and f' is the use of different parameters, initial

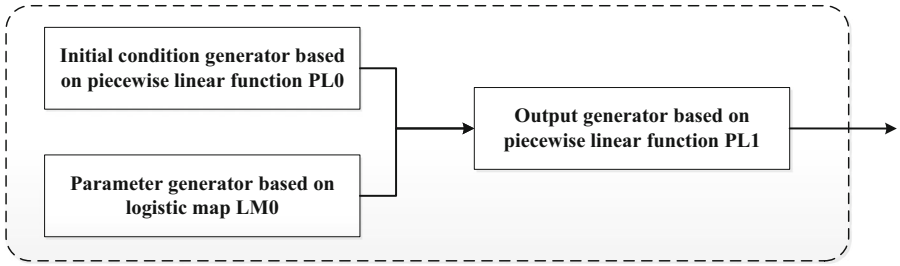


Fig. 1 Functions based on multiple chaotic maps f and f'

conditions and initialization iteration times to update $LM0$ and $PL0$ before encryption (Fig. 1).

2.1.1 Logistic map $LM0$

The output xlc of logistic map $LM0$ is used to generate parameter pin of piecewise linear function $PL1$. The logistic map is described as follows:

$$x(t+1) = px(t)(1-x(t)) \tag{1}$$

when $3.57 < p \leq 4$ and $0 < x(t) < 1$, the map is chaotic, and $x(t)$ is ergodic on $(0, 1)$.

The initialization of $LM0$ is determined by two parts: a fixed parameter $3.57 < p \leq 4$ and parts of keys of algorithm, including initial condition $xlc10$ ($xlc10$ for f and $xlc20$ for f') and initialization iteration times $pre1$ ($pre1$ for f and $pre2$ for f'). After initialization, the output xlc of $LM0$ updates when f (or f') is operated.

2.1.2 Piecewise linear function $PL0$

The output xpc of piecewise linear function $PL0$ is used to generate parameter xin of piecewise linear function $PL1$. The piecewise linear function is described as follows,

$$x(t+1) = \begin{cases} x(t)/p, & 0 \leq x(t) < p \\ (x(t)-p)/(0.5-p), & p \leq x(t) < 0.5 \\ (1-x(t)-p)/(0.5-p), & 0.5 \leq x(t) < 1-p \\ (1-x(t))/p & 1-p \leq x(t) \leq 1, \end{cases} \tag{2}$$

where $x \in [0, 1]$, $p \in (0, 0.5)$, the map is chaotic and $x(t)$ is ergodic on $(0, 1)$.

The initialization of $PL0$ is determined by two parts: a fixed parameter $p \in (0, 0.5)$ and parts of keys of algorithm, including initial condition $xpc10$ ($xpc10$ for f and $xpc20$ for f') and initialization iteration times $pre3$ ($pre3$ for f and $pre4$ for f'). After initialization, the output xpc of piecewise linear function $PL0$ updates when f (or f') is operated.

2.1.3 Piecewise linear function $PL1$

The output $xout$ of piecewise linear function $PL1$ is used to generate output of f (or f'). The piecewise linear function $PL1$ is described as follows,

$$xout = \begin{cases} xin/pin, & 0 \leq xin < pin \\ (xin-pin)/(0.5-pin), & pin \leq xin < 0.5 \\ (1-xin-pin)/(0.5-pin), & 0.5 \leq xin < 1-pin \\ (1-xin)/pin & 1-pin \leq xin \leq 1 \end{cases} \tag{3}$$

Parameter pin of $PL1$ is generated by formula (4), and initial condition xin of $PL1$ is generated by formula (5).

$$pin = fraction((b.1 + b.2 + b.3)/765 + xlc)/2 \tag{4}$$

$$xin = fraction(a/255 + xpc) \tag{5}$$

where $fraction()$ means taking the fraction part of the number.

With the ergodicity of $LM0$ and $PL0$ on $(0, 1)$, we can easily find that $xin \in (0, 1)$, $pin \in (0, 0.5)$. And thus, the piecewise linear function $PL1$ makes sense on mathematics.

The participation of pseudo-random numbers is to extend the data range of the parameters and initial situations. If there are not the pseudo-random numbers, the data range of the parameters will be just 766 numbers, namely $\{0, 1/1530, 2/1530, \ldots, 1/2\}$ and the data range of the initial situations will be 256 numbers, namely $\{0, 1/255, 2/255, \ldots, 1\}$. Because of the ergodicity of logistic map and the piecewise linear map, with the participation of the pseudo-random numbers, the data range of parameters will be the whole interval $(0, 1/2)$, and the data range of initial situations will be the whole interval $(0, 1)$.

2.1.4 Structure of function f and f'

The calculation process is as below.

- Before encrypting, $LM0$ and $PL0$ should finish initialization.

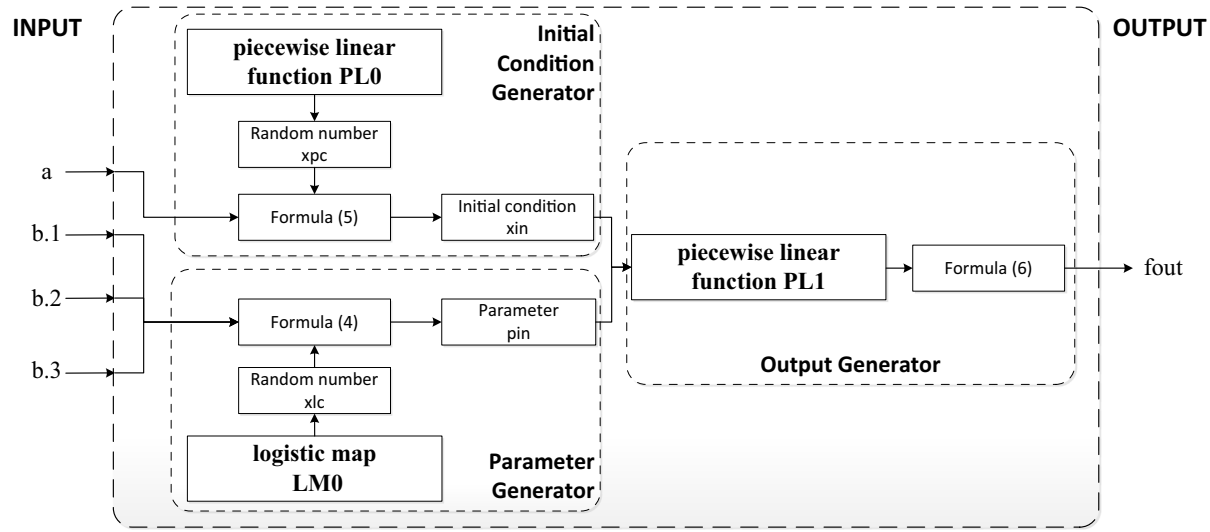


Fig. 2 Structure of function f and f'

- $LM0$ and $PL0$ generate two pseudo-random numbers xlc and xpc between 0 and 1. And then, generate parameter pin of $PL1$ by formula (4), and initial condition xin of $PL1$ by formula (5).
- $PL1$ generates a pseudo-random number $xout$ between 0 and 1. And then, generate an integer output between 0 and 255 by formula (6) (Fig. 2).

$$\begin{aligned} fout &= f(a, b.1, b.2, b.3) \\ &= \lfloor xout \times 10^{14} \rfloor \mod 256 \end{aligned} \tag{6}$$

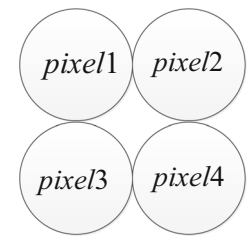
2.2 4-Pixel Feistel structure and round functions

4-Pixel Feistel structure and round functions are designed to improve the diffusion properties of algorithm. Respectively, 4-Pixel Feistel structure focuses more on diffusion among 4 pixels and round functions focus more on the diffusion among three colours of pixels.

2.2.1 Pixel Feistel structure

The smallest block that we use to operate the encryption and decryption is 4 pixels. The operation is inspired by the 4-branch Feistel structure of CLEFIA [23]. But differently, the number of rounds is 2 and is enough to realize the whole diffusion. More rounds can be accepted, and in theory, the security intensity will be promoted, but in cost of the speed (Figs. 3, 4)

Fig. 3 Block of operation



K is 3-dimensional vector (k_1, k_2, k_3) generated by Lorenz system, and K' is 3-dimensional vector (k'_1, k'_2, k'_3) generated by Chen's system.

There are two reasons why we design 4-Pixel Feistel structure. Firstly, as a character of Feistel structures, the encryption and the decryption have similar structures. It permits us to use relatively complex functions in the encryption and decryption process. Secondly, 4-Pixel Feistel structure can be calculated by parallel computing. It provides a potential of higher encryption speed.

2.2.2 Round functions

Round functions (namely pf and pf') are 3-dimensional functions, respectively, constructed by f and f' to improve the diffusion properties among three colours of pixels by the rotation of colours participating in operation. The mathematical description of pf and pf' is abstract and helpless. We can simply observe how pf

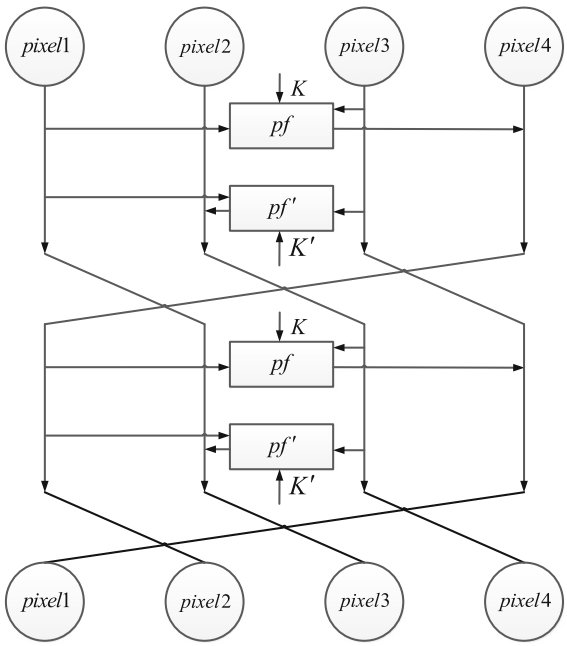


Fig. 4 4-Pixel Feistel structure of encryption

and pf' work by the description of one round operation of 4-Pixel Feistel structure.

We suppose that the RGB value of pixel p can, respectively, be presented as $p.r, p.g, p.b$. $(k_1(i), k_2(i), k_3(i))$ and $(k_1'(i), k_2'(i), k_3'(i))$ are round keys generated in the key expansion.

For each round, the encryption operation can be expressed as:

$$\begin{cases} pixel1.r = pixel4.g \oplus f(pixel1.b \oplus k_1(i), pixel3.r, pixel3.g, pixel3.b) \\ pixel1.g = pixel4.b \oplus f(pixel1.r \oplus k_2(i), pixel3.r, pixel3.g, pixel3.b) \\ pixel1.b = pixel4.r \oplus f(pixel1.g \oplus k_3(i), pixel3.r, pixel3.g, pixel3.b) \end{cases} \tag{7}$$

$$\begin{cases} pixel2.r = pixel1.r \\ pixel2.g = pixel1.g \\ pixel2.b = pixel1.b \end{cases} \tag{8}$$

$$\begin{cases} pixel3.r = pixel2.b \oplus f'(pixel3.g \oplus k_1'(i), pixel1.r, pixel1.g, pixel1.b) \\ pixel3.g = pixel2.r \oplus f'(pixel3.b \oplus k_2'(i), pixel1.r, pixel1.g, pixel1.b) \\ pixel3.b = pixel2.g \oplus f'(pixel3.r \oplus k_3'(i), pixel1.r, pixel1.g, pixel1.b) \end{cases} \tag{9}$$

$$\begin{cases} pixel4.r = pixel3.r \\ pixel4.g = pixel3.g \\ pixel4.b = pixel3.b \end{cases} \tag{10}$$

The diffusion among 3 colours is realized by the rotation of colours participating in operations. Thus, the change of any pixel's colour can affect the other pixels' colours.

2.3 3D chaotic key generators aiming at colour images

For colour images, each pixel has three colours as parameters. Using 3D chaotic system can generate three keys at one time. Using two different chaotic systems can effectively avoid the interaction of keys generated by two systems.

2.3.1 Lorenz system

The Lorenz system is described as follows:

$$\begin{cases} \dot{x}_1 = p(x_2 - x_1) \\ \dot{x}_2 = -x_1 x_3 + r x_1 - x_2 \\ \dot{x}_3 = x_1 x_2 - t x_3 \end{cases} \tag{11}$$

when $p = 10, r = 28, t = 8/3$, the system is chaotic.

For Lorenz system, with an initial condition $(x_1^*(0), x_2^*(0), x_3^*(0))$, use fourth-order Runge-Kutta to solve the Lorenz system with a step of $h = 0.001$. Firstly, iterate the Lorenz system M_0 times to generate the real initial condition $(x_1(0), x_2(0), x_3(0))$, and for the next ith solution $(x_1(i), x_2(i), x_3(i))$ of Lorenz system, the key $(k_1(i), k_2(i), k_3(i))$ is generated by

$$\begin{cases} k_1(i) = \lfloor (x_1(i) - \lfloor x_1(i) \rfloor) \times 10^{14} \rfloor \mod 256 \\ k_2(i) = \lfloor (x_2(i) - \lfloor x_2(i) \rfloor) \times 10^{14} \rfloor \mod 256 \\ k_3(i) = \lfloor (x_3(i) - \lfloor x_3(i) \rfloor) \times 10^{14} \rfloor \mod 256 \end{cases} \tag{12}$$

2.3.2 Chen's system

The Chen's system is described as follows:

$$\begin{cases} \dot{x}_4 = a(x_5 - x_4) \\ \dot{x}_5 = (c - a)x_4 - x_4x_6 + cx_5 \\ \dot{x}_6 = x_4x_5 - bx_6 \end{cases} \tag{13}$$

when $a = 35, b = 3, c = 28$, the system is chaotic.

Similarly with Lorenz system, for Chen's system, with an initial condition $(x_4^*(0), x_5^*(0), x_6^*(0))$, use fourth-order Runge-Kutta to solve the Chen's system with a step of $h = 0.001$. Firstly, iterate the Chen's system N_0 times to generate the real initial condition $(x_4(0), x_5(0), x_6(0))$, and for the next ith solution $(x_4(i), x_5(i), x_6(i))$ of Chen's system, the key $(k_1'(i), k_2'(i), k_3'(i))$ is generated by

$$\begin{cases} k_1'(i) = \lfloor (x_4(i) - \lfloor x_4(i) \rfloor) \times 10^{14} \rfloor \mod 256 \\ k_2'(i) = \lfloor (x_5(i) - \lfloor x_5(i) \rfloor) \times 10^{14} \rfloor \mod 256 \\ k_3'(i) = \lfloor (x_6(i) - \lfloor x_6(i) \rfloor) \times 10^{14} \rfloor \mod 256 \end{cases} \tag{14}$$

2.3.3 Key space

The proposed algorithm has a large key space composed of 10 double numbers and six integer numbers.

They are,

- The initial condition of Lorenz System $(x_1^*(0), x_2^*(0), x_3^*(0))$;
- The initial condition of Chen's System $(x_4^*(0), x_5^*(0), x_6^*(0))$;
- The initial conditions of two logistic maps $xcl10$ and $xcl20$;
- The initial conditions of two piecewise linear functions $xpie10$ and $xpie20$
- The initial iteration times of Lorenz System M_0
- The initial iteration times of Chen's System N_0
- The initial iteration times of two logistic maps $pre1$ and $pre2$
- The initial iteration times of two piecewise linear functions $pre3$ and $pre4$

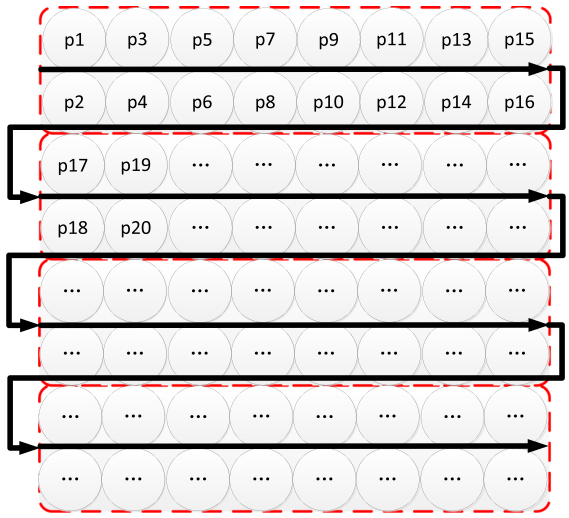


Fig. 5 Origin image

2.4 Dependent encryption process

Establishing dependent relationship among pixels is an effective mean to resist many cryptanalysis methods, such as known-/chosen plaintext attack and chosen cipher attack. In the proposed algorithm, the dependent relationship among pixels is established in the encryption process.

The block (i, j) presents a block as follows,

$$\begin{pmatrix} p(i,j) & p(i+2\times\lfloor (j+1)/\text{width} \rfloor, (j+1)(\text{mod width})) \\ p(i+1,j) & p(i+1+2\times\lfloor (j+1)/\text{width} \rfloor, (j+1)(\text{mod width})) \end{pmatrix}$$

$0 \le i \le Height - 2, 0 \le j \le width - 1, Height$ and $width$ represent the height and width of the image.

With the defined block of operation, the $height \times width$ image can be transferred into a new image with $height' = 2$ and $width' = height \times width/2$. The new image is quite easy to apply the proposed diffusion operation (Figs. 5, 6).

In the encryption process, after handling the first block, the colours of last 2 pixels depend on all the colours of 4 pixels. And then, when we handle the second block, the first 2 pixels are the last 2 pixels of the first block, and then the dependent relationship is established block by block (Fig. 7).

But we can also find that last block cannot affect previous block, thus, another time of encryption is needed.

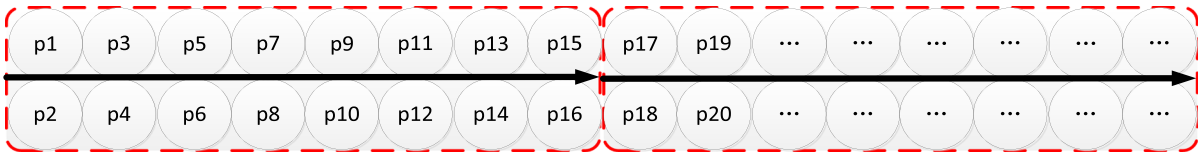


Fig. 6 New image

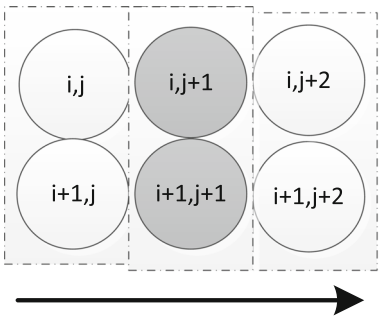


Fig. 7 Intersection of two blocks

The process of encryption can be described by pseudocode as follows,

Lorenz System Initialized
Chen's System Initialized
Logistic Maps Initialized
Piecewise Linear Functions Initialized
K and K′ generated
for $i = i + 2$ *from* 0 *to height* $- 2$
  for $j = j + 1$ *from* 0 *to width* $- 1$
    block operation_en (i, j);
  endfor
end for
for $i = i - 2$ *from height* $- 2$ *to* 0
  for $j = j - 1$ *from width* $- 1$ *to* 0
    block operation_en (i, j);
  endfor
end for

The *block operation_en* (i, j) represents the block operation of encryption for the block (i, j).

The process of encryption can be described by flowchart as follows (Fig. 8),

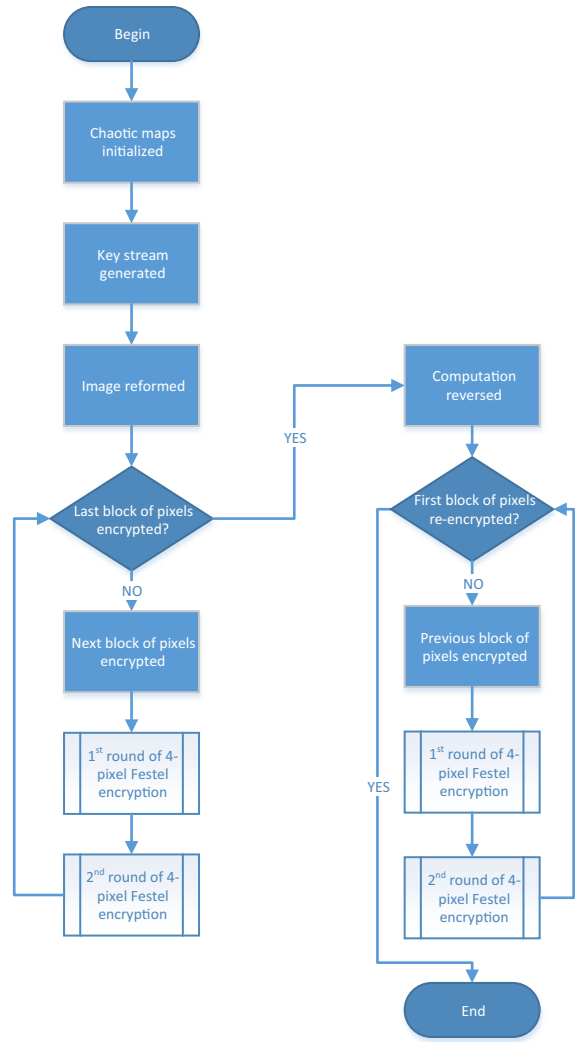


Fig. 8 Flowchart of the encryption process

The encryption result and corresponding decryption result of "Lena" (512*512) and all-zero image (512*512) show in figure below. (Using key $x_1^* = 3.05152212424679$, $x_2^* = 1.58254212245123$, $x_3^* = 15.6238853231785$, $x_4^* = 4.78999921123234$, $x_5^* = 1.98243221252248$, $x_6^* = 14.2532112455785$, $M_0 = 20$, $N_0 = 30$, $xcl10 = 0.589756425683531$, $xcl20 = 0.286245832183542$, $xpie10 = 0.96311588683553$, $xpie20 = 0.732108327953574$, $prel = 40$, $pre2 = 50$, $pre3 = 55$, $pre4 = 45$. If there is no other

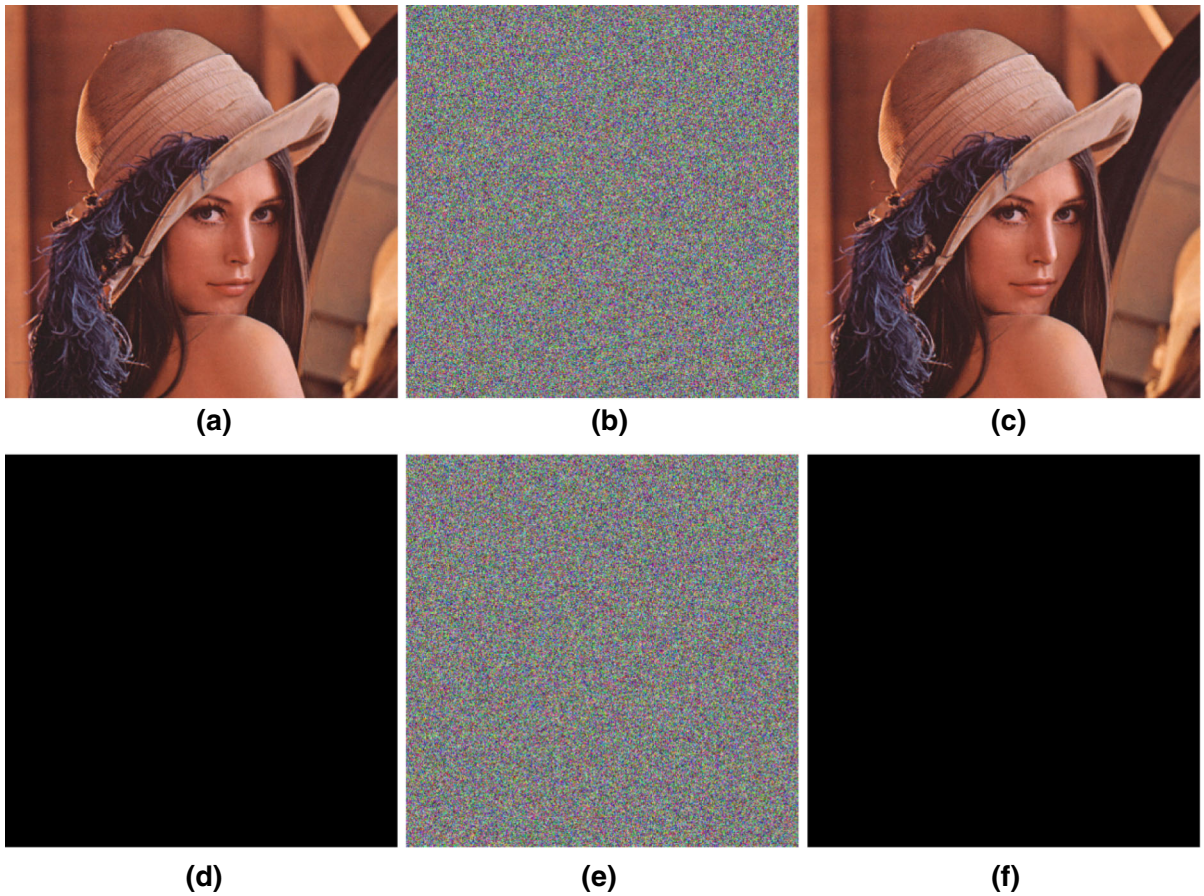


Fig. 9 Results of "Lena" and all-zero image. **a** Plain image "Lena". **b** Encrypted image. **c** Recovered image. **d** all-zero Image. **e** Encrypted image. **f** Recovered image

announce, all the encryption and decryption examples use the above key in this paper.) (Fig. 9)

3 Experimental results and cryptanalysis

In the following experimental environment, CPU: Intel Core i5-4210U CPU 1.70 GHz; Memory: 4.00 GB; Operating system: Windows 8.1; Coding tool: Visual studio 2012, the experiments include histogram analysis, correlation of two adjacent pixels, NPCR and UACI, sensitivity to cipher image, information entropy, key sensitivity, key space analysis and speed analysis.

3.1 Histogram analysis

Histograms show the distribution of pixel values in an image. The ideal histogram of a cipher image is uniform. In Figure below, we show the histograms of RGB values of the plain image "Lena" and those of the cipher image (Fig. 10).

In Figure below, we show the histograms of RGB values of the plain image, an all-zero image and those of the cipher image (Fig. 11).

We can find that the histograms of the cipher image are close to uniform. Thus, a frequency analysis cannot be used to break the algorithm.

3.2 Correlation of two adjacent pixels

The correlation of two adjacent pixels can be calculated according to the following formula:

$$r_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{15}$$

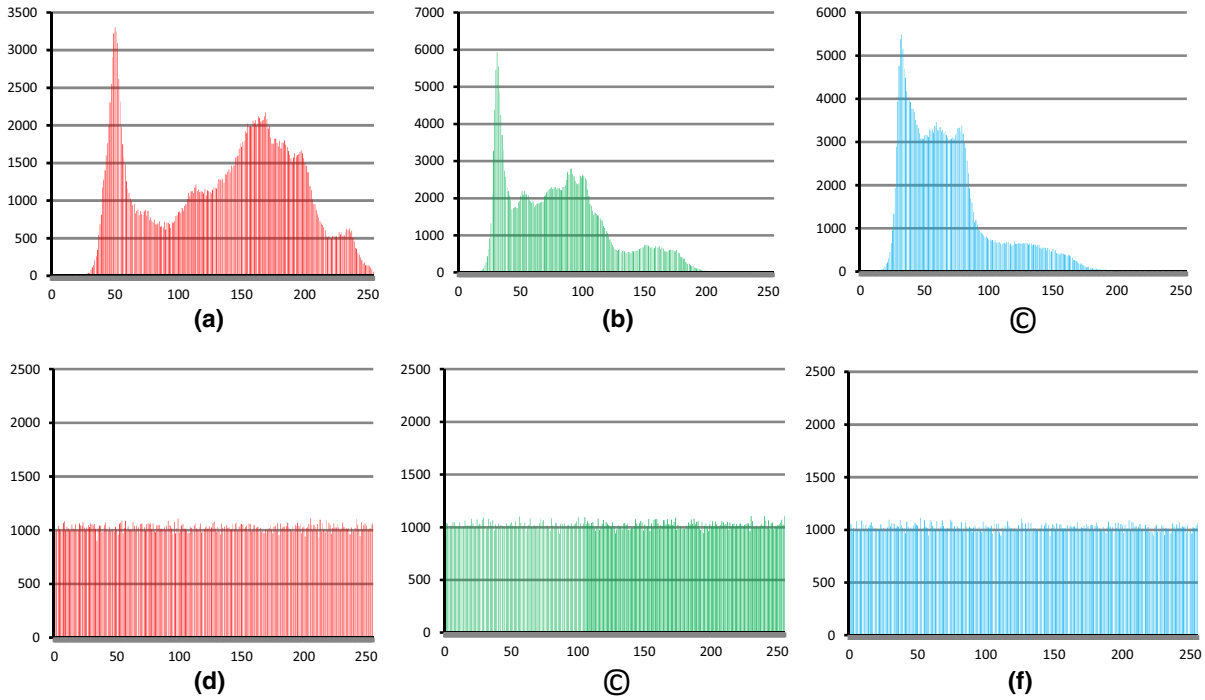


Fig. 10 **a** Histogram of red channel of "Lena". **b** Histogram of *green channel* of "Lena". **c** Histogram of *blue channel* of "Lena". **d** Histogram of *red channel* of cipher © Histogram of green channel of cipher. **f** Histogram of *blue channel* of cipher

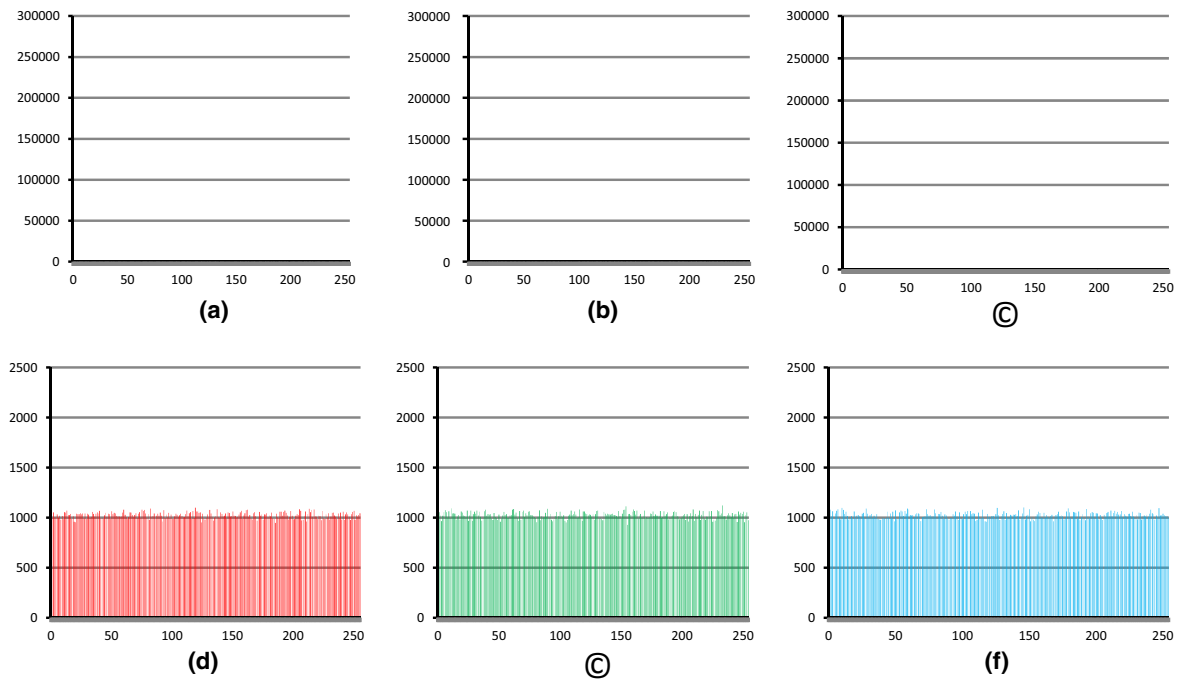


Fig. 11 **a** Histogram of *red channel* of all-zero image. **b** Histogram of *green channel* of all-zero image. **c** Histogram of *blue channel* of all-zero image. **d** Histogram of *red channel* of cipher © Histogram of *green channel* of cipher. **f** Histogram of *blue channel* of cipher

Table 1 Correlation between plain image "Lena" and its cipher image

	Plain Image "Lena"		
	Red	Green	Blue
PPS09			
Red	0.0026	−0.0105	0.0073
Green	0.0024	−0.0077	0.0064
Blue	0.0005	−0.0063	0.0047
mPPS09			
Red	0.0006	0.0019	0.0020
Green	0.0063	0.0056	0.0011
Blue	0.0057	0.0069	0.0012
Proposed algorithm			
Red	0.0013	−0.0022	0.0024
Green	0.0009	−0.0003	−0.0006
Blue	0.0010	0.0006	0.0009

Table 2 Horizontal correlation of "Lena" Cipher

	Red	Green	Blue
Red	−0.0010	−0.0016	−0.0013
Green	−0.0016	0.0004	−0.0042
Blue	−0.0036	0.0005	0.0016

Table 3 Vertical correlation of "Lena" Cipher

	Red	Green	Blue
Red	0.0010	0.0027	0.0001
Green	−0.0017	0.0038	−0.0006
Blue	0.0011	0.0049	0.0015

Table 4 Diagonal correlation of "Lena" Cipher

	Red	Green	Blue
Red	−0.0001	0.0012	−0.0005
Green	0.0004	−0.0008	0.0031
Blue	−0.0039	0.0031	0.0019

where

$$\mathrm{cov}(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y)),$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i,\ E(y) = \frac{1}{N}\sum_{i=1}^{N} y_i,$$

$$N = height \times width.$$

Table 5 Horizontal correlation of all-zero's Cipher

	Red	Green	Blue
Red	−0.0014	0.0004	0.0016
Green	0.0011	−0.0035	−0.0018
Blue	0.0002	0.0043	0.0033

Table 6 Vertical correlation of all-zero's Cipher

	Red	Green	Blue
Red	−0.0012	−0.0025	−0.0012
Green	−0.0033	−0.0004	0.0025
Blue	0.0007	0.0012	−0.0042

Table 7 Diagonal correlation of all-zero's Cipher

	Red	Green	Blue
Red	0.0003	−0.0006	0.0041
Green	0.0011	0.0018	0.0008
Blue	0.0003	0.0016	0.0050

Table 8 Results of NPCR and UACI of "Lena"

	NPCR (%)	UACI (%)
Red	99.6010	33.4745
Green	99.6120	33.4714
Blue	99.6109	33.4676

Table 9 Results of NPCR and UACI of all-zero image

	NPCR (%)	UACI (%)
Red	99.6288	33.3998
Green	99.5945	33.4828
Blue	99.6082	33.3749

First, we test the correlation between various colours of "Lena" and its cipher image, and compare the results with the algorithm PPS09 and mPPS09 [24] (Table 1).

We can find that the proposed algorithm have a much better statistic properties than PPS09 and mPPS09.

We will test the correlation of two adjacent in three directions, namely horizontal, vertical and diagonal. As the algorithm is for colour image, correlation of different channels should also been considered (Tables 2, 3, 4).

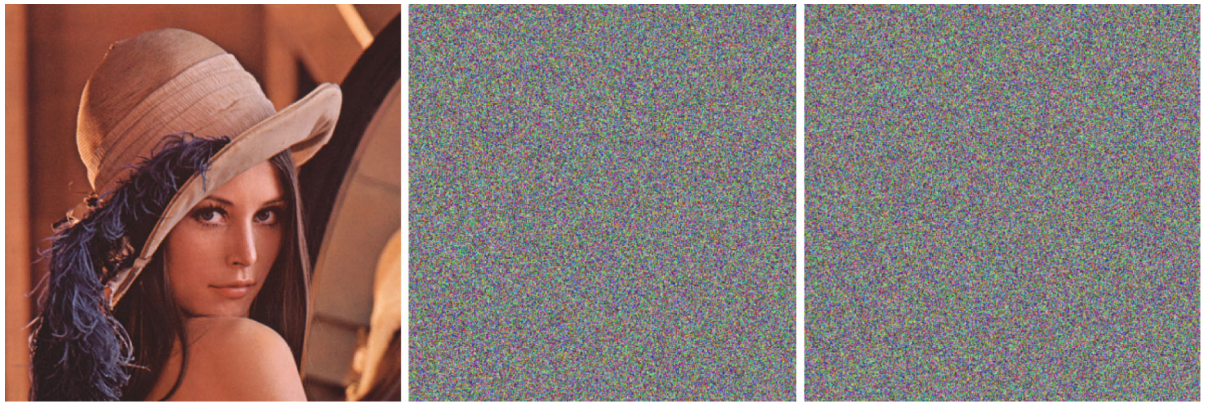


Fig. 12 **a** Origin Image "Lena". **b** Encrypted Image of "Lena". **c** Encrypted Image of "Lena" changing *red value* of one pixel

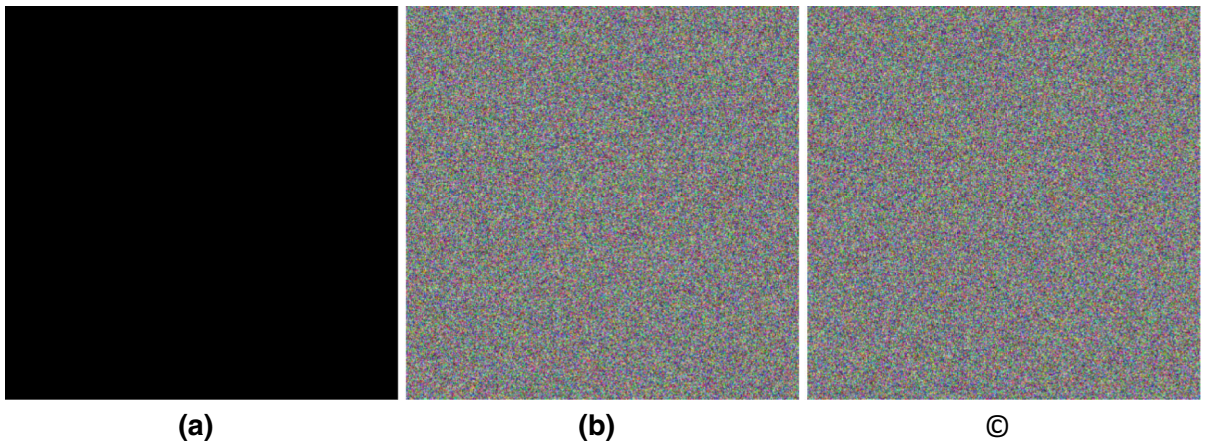


Fig. 13 **a** Origin Image all-zero image. **b** Encrypted Image of all-zero image. **c** Encrypted Image of all-zero image changing *red value* of one pixel

And then, as $D(x) = 0$ for all-zero image, we cannot test the correlation between various colours of all-zero image and its cipher image. But we can test the correlation of two adjacent of its cipher in three directions (Tables 5, 6, 7).

The results show that different colours in different directions have little correlation.

3.3 NPCR and UACI

Number of Pixels Change Rate (NPCR) stands for the number of pixels change rate while one pixel of plain image changed. The NPCR gets closer to 100 %, the more sensitive the cryptosystem to the changing of plain image, and the more effective for the cryptosystem to resist plaintext attack. UACI(Unified Average Changing Intensity) stands for the average intensity of differences between the plain image and ciphered image. The UACI gets closer to 33.333…%, the more effective for the cryptosystem to resist differential attack.

NPCR and UACI can be calculated as follows,

$$\mathrm{NPCR} = \frac{\sum_{ij} D(i,j)}{\mathrm{Width} \times \mathrm{Height}} \times 100\,\% \tag{16}$$

$$\mathrm{UACI} = \frac{1}{\mathrm{Width} \times \mathrm{Height}} \times \left[\sum_{ij} \frac{|c_1(i,j) - c_2(i,j)|}{255} \right] \times 100\,\% \tag{17}$$

Fig. 14 **a** Encrypted Image of "Lena". **b** Decrypted Image of (**a**). **c** Decrypted Image of (**a**) with a *red value* of one pixel of cipher changed

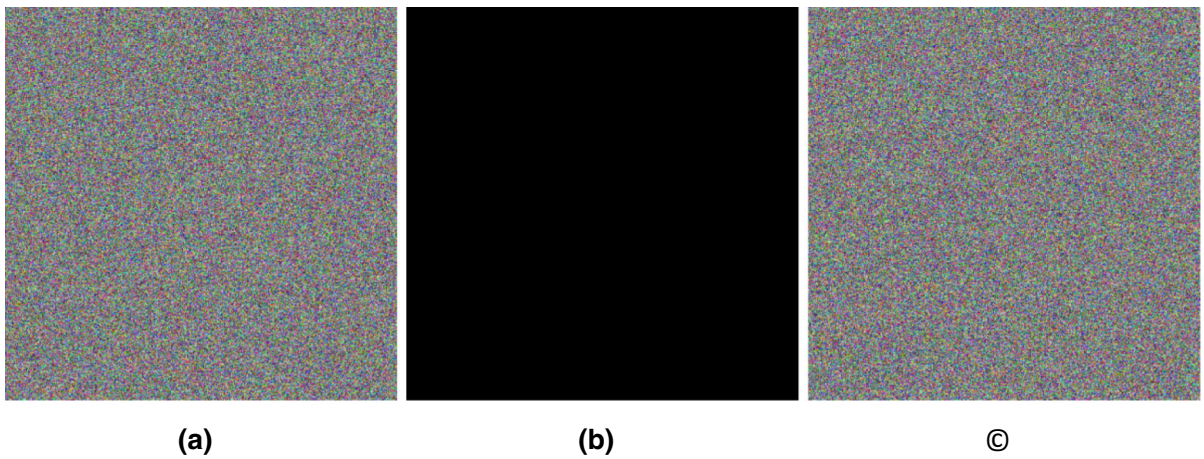


Fig. 15 **a** Encrypted Image of all-zero image. **b** Decrypted Image of (**a**) **c**. Decrypted Image of (**a**) with a *red value* of one pixel of cipher changed

Table 10 Results of Sensitivity to cipher image of "Lena"

	NPCR (%)	Correlation		
		Red	Green	Blue
Red	99.6143	0.0004	0.0003	0.0037
Green	99.6389	0.0004	0.0006	0.0044
Blue	99.5960	−0.0002	0.0008	0.0040

Table 11 Results of Sensitivity to cipher image of all-zero image

	NPCR (%)
Red	99.6113
Green	99.6120
Blue	99.6151

where $c_1(i, j)$ and $c_2(i, j)$ are, respectively, the cipher image before and after one pixel of the plain image is changed. And if $c_1(i, j) \neq c_2(i, j)$, $D(i, j) = 1$; otherwise, $D(i, j) = 0$.

The results of changing the red value of a pixel show in the Tables 8 and 9

The results show that bit change of the value of a colour channel of one pixel can cause change of whole cipher image. The good diffusion property is demon-

Table 12 Information Entropy of "Lena" cipher image

	Red channel	Green channel	Blue channel
Proposed scheme	7.999369	7.999299	7.999319
Wu's scheme [13]	7.9899	7.9894	7.9896
Zhang's scheme [16]	7.997341	7.997162	7.996889
Faraoun's scheme [17]	7.9899	7.9997	7.9979

Table 13 Information Entropy of all-zero's cipher image

	Red channel	Green channel	Blue channel
Proposed scheme	7.999374	7.999339	7.999231

strated. And the algorithm can resist plaintext attack and differential attack (Figs. 12, 13).

3.4 Sensitivity to cipher image

As is done in [11], when one pixel of cipher image is changed, the recovered plain image exhibits no correlation to the plain image, then the cipher can resist chosen cipher attack. Similarly, we calculate the NPCR and correlation for recovered image and plain image (Figs. 14, 15; Table 10).

Also as $D(x) = 0$ for all-zero image, we cannot test the correlation between various colours of the recovered image and plain image (Table 11).

The results show that the algorithm can resist chosen cipher attack.

3.5 Information entropy

The method to calculate information entropy of an image can be expressed as below,

$$H(m) = \sum_{i=0}^{2^N-1} p(m_i) \log_2 \frac{1}{p(m_i)} \tag{18}$$

where $p(m_i)$ represents the probability of symbol m_i, and $\log_2$ represents the base 2 logarithm so that the entropy is expressed in bits, N represents the number of bits we use to represent a pixel, and for one colour channel of a pixel, it is clear that $N = 8$. If an image is ideal random, then for each i, $p(m_i) = 1/256$, and we can easily find that $H(m) = 8$. And the results of cipher image of "Lena" and all-zero image are below (Tables 12, 13).

The results show that entropy of all the three colour channels are close to the ideal value 8. Thus, the algorithm is secure upon the entropy attack.

3.6 Key sensitivity

To prove the key sensitivity to the encryption process, for double numbers, we will change the last number of its fraction, and for the integer numbers, will plus 1. And then calculate the correlation between the ciphered

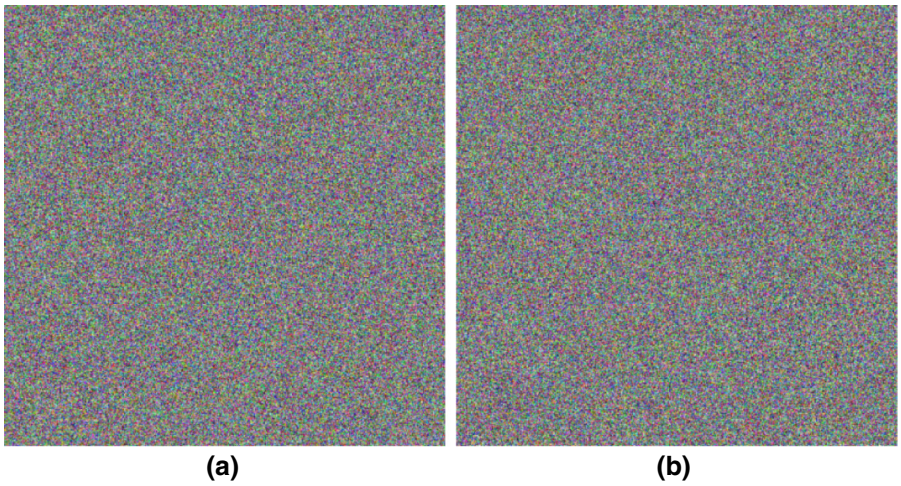


Fig. 16 **a** Encrypted Image with a 10^{-15}change of x_1^*. **b** Decrypted Image of (**a**) with origin key

Table 14 Correlation test for encryption key sensitivity

	Red	Green	Blue	Red	Green	Blue
	$x_1^* = 3.05152212424670$, the others rest invariant			$x_2^* = 1.58254212245124$, the others rest invariant		
Red	0.0007	−0.0025	−0.0002	−0.0011	−0.0029	0.0034
Green	0.0012	−0.0009	0.0011	0.0022	−0.0010	−0.0023
Blue	0.0028	0.0010	−0.0010	0.0024	0.0021	−0.0022
	$x_3^* = 15.6238853231786$, the others rest invariant			$x_4^* = 4.78999921123235$, the others rest invariant		
Red	−0.0013	0.0027	−0.0008	0.0040	−0.0014	0.0018
Green	0.0028	0.0024	0.0037	−0.0001	0.0023	−0.0030
Blue	0.0020	0.0016	−0.0034	−0.0001	0.0021	0.0017
	$x_5^* = 1.98243221252249$, the others rest invariant			$x_6^* = 14.2532112455786$, the others rest invariant		
Red	0.0017	0.0005	0.0008	−0.0013	0.0012	0.0028
Green	0.0029	−0.0019	0.0011	−0.0017	0.0010	−0.0009
Blue	−0.0011	−0.0010	0.0011	0.0009	−0.0010	0.0013
	$xcl10 = 0.589756425683532$, the others rest invariant			$xcl20 = 0.286245832183543$, the others rest invariant		
Red	−0.0029	−0.0015	−0.0011	0.0003	−0.0037	0.0006
Green	0.0008	−0.0026	−0.0018	0.0000	0.0001	−0.0011
Blue	0.0014	−0.0001	−0.0008	0.0030	0.0001	−0.0001
	$xpie10 = 0.963211588683554$, the others rest invariant			$xpie20 = 0.732108327953575$, the others rest invariant		
Red	0.0023	−0.0008	0.0003	0.0014	−0.0025	0.0011
Green	−0.0010	0.0007	−0.0015	0.0009	0.0011	−0.0016
Blue	0.0005	0.0004	−0.0004	−0.0009	0.0013	0.0018
	$M_0 = 21$, the others rest invariant			$N_0 = 31$, the others rest invariant		
Red	0.0000	0.0004	0.0011	0.0007	−0.0004	−0.0007
Green	0.0024	0.0006	0.0006	0.0018	0.0024	−0.0005
Blue	−0.0014	−0.0015	−0.0013	0.0011	−0.0050	0.0041
	$pre1 = 41$, the others rest invariant			$pre2 = 51$, the others rest invariant		
Red	−0.0007	0.0009	0.0004	0.0014	−0.0007	−0.0045
Green	−0.0015	0.0012	0.0010	−0.0010	−0.0003	0.0022
Blue	−0.0003	−0.0042	−0.0011	0.0020	0.0007	−0.0013
	$pre3 = 56$,the others rest invariant			$pre4 = 46$,the others rest invariant		
Red	0.0033	0.0036	−0.0002	0.0007	−0.0019	0.0027
Green	−0.0008	−0.0030	0.0004	−0.0017	0.0036	−0.0028
Blue	−0.0025	−0.0009	−0.0008	−0.0008	−0.0028	0.0037

image before changing and the ciphered images after changing (Fig. 16).

The origin encryption keys are

$$x_1^* = 3.05152212424679,$$
$$x_2^* = 1.58254212245123,$$
$$x_3^* = 15.6238853231785,$$
$$x_4^* = 4.78999921123234,$$
$$x_5^* = 1.98243221252248,$$
$$x_6^* = 14.2532112455785,$$
$$xcl10 = 0.589756425683531,$$
$$xcl20 = 0.286245832183542,$$
$$xpie10 = 0.963211588683553,$$
$$xpie20 = 0.732108327953574,$$
$$M_0 = 20, N_0 = 30, pre1 = 40,$$
$$pre2 = 50, \quad pre3 = 55, pre4 = 45$$

The results of correlation test for encryption key sensitivity show as below (Table 14),

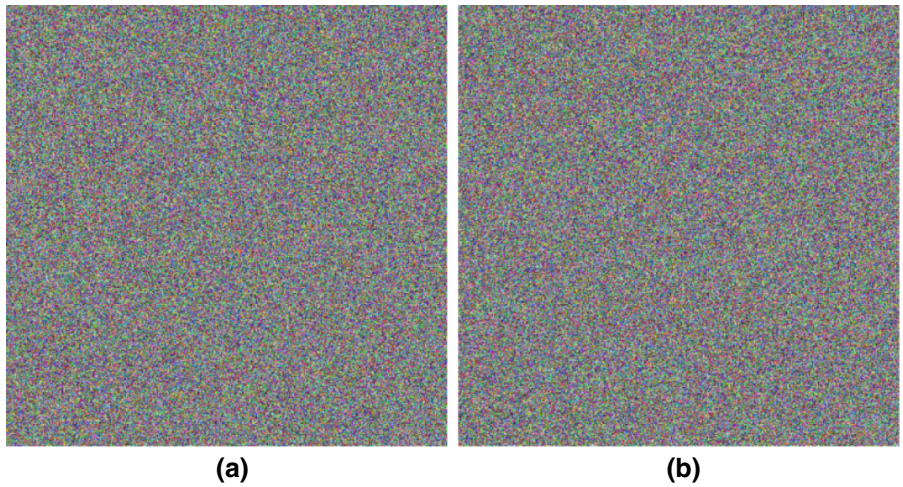


Fig. 17 **a** Encrypted Image with origin keys. **b** Decrypted Image of (**a**) with a 10^{-15} change of x_1^*

To prove the key sensitivity to the decryption process, for double numbers, we will change the last number of its fraction, and for the integer numbers, we will plus 1. And then use them as keys to decrypt the image encrypted by origin keys. After that, we calculate the correlation of the novel decrypted images and the origin decrypted image (Fig. 17).

The origin decryption keys are

$$\begin{aligned} x_1^* &= 3.05152212424679,\\ x_2^* &= 1.58254212245123,\\ x_3^* &= 15.6238853231785,\\ x_4^* &= 4.78999921123234,\\ x_5^* &= 1.98243221252248,\\ x_6^* &= 14.2532112455785,\\ xcl10 &= 0.589756425683531,\\ xcl20 &= 0.286245832183542,\\ xpie10 &= 0.963211588683553,\\ xpie20 &= 0.732108327953574,\\ M_0 &= 20,\ N_0 = 30,\ pre1 = 40,\\ pre2 &= 50, \quad pre3 = 55, \quad pre4 = 45 \end{aligned}$$

The results of correlation test for encryption key sensitivity show as below (Table 15),

So, it can be concluded that the new chaotic algorithm is sensitive to the key such that a small change of the key will generate a completely different decryption result and cannot get the correct plain image.

3.7 Key space analysis

As is declared in [12] and other papers [13,16,25], as the fact that operations on computers is with finite accuracy is taken into account, size of key space could be roughly estimated of all possible combinations of control parameters and initial values.

As having been said in Sect. 2.3, we have a large key space composed of 10 double numbers and six integer numbers. By the results of sensitivity, the precision of all 10 double numbers is 10^{-15}. Thus, one can obtain that the key space can be larger than $10^{15*10} > 2^{498}$ for the keys used in this scheme. Thus, the algorithm can resist the brute-force attack.

3.8 Speed analysis

The encryption speed is also an important factor. Because it is a responsibility of the practicability of the scheme. The proposed scheme also focus on the speed. The computation of 4 pixels at same time is for not only increasing the interaction of pixels, but also making the scheme more efficient. The use of 4-Pixel Feistel structure also provides the possibility of speeding-up using the parallel computing on multi-core computers.

We have used Visual studio 2012 with OpenCV3.0.0 to run the encryption algorithm in a computer with an Intel Core i5 1.70 GHz, 4.00 GB Memory, and Windows 8.1 operating system. The result is shown in the following table.

Table 15 Correlation test for decryption key sensitivity

	Red	Green	Blue	Red	Green	Blue
	$x_1^* = 3.05152212424670$, the others rest invariant			$x_2^* = 1.58254212245124$, the others rest invariant		
Red	0.0024	−0.0002	0.0027	−0.0012	0.0028	0.0020
Green	0.0015	0.0004	0.0036	−0.0007	0.0030	0.0029
Blue	0.0004	0.0001	0.0040	0.0001	0.0022	0.0035
	$x_3^* = 15.6238853231786$,the others rest invariant			$x_4^* = 4.78999921123235$, the others rest invariant		
Red	−0.0014	0.0017	0.0000	0.0008	0.0032	0.0027
Green	−0.0018	0.0010	−0.0014	0.0001	0.0042	0.0026
Blue	0.0021	0.0003	−0.0020	−0.0006	0.0045	0.0025
	$x_5^* = 1.98243221252249$,the others rest invariant			$x_6^* = 14.2532112455786$, the others rest invariant		
Red	−0.0022	0.0013	0.0013	0.0006	−0.0017	0.0007
Green	−0.0013	0.0000	0.0009	−0.0007	0.0013	0.0001
Blue	−0.0009	−0.0015	0.0009	−0.0021	−0.0009	0.0003
	$xcl10 = 0.589756425683532$, the others rest invariant			$xcl20 = 0.286245832183543$, the others rest invariant		
Red	0.0024	−0.0038	0.0001	0.0004	0.0006	−0.0024
Green	0.0026	−0.0035	0.0003	0.0004	0.0010	−0.0019
Blue	0.0021	−0.0031	0.0005	−0.0003	0.0008	−0.0016
	$xpie10 = 0.963211588683554$, the others rest invariant			$xpie20 = 0.732108327953575$, the others rest invariant		
Red	−0.0001	0.0005	0.0016	0.0004	−0.0030	−0.0026
Green	−0.0010	0.0011	0.0017	0.0004	−0.0024	−0.0031
Blue	−0.0011	0.0011	0.0022	0.0012	−0.0024	−0.0026
	$M_0 = 21$, the others rest invariant			$N_0 = 31$, the others rest invariant		
Red	0.0018	−0.0026	0.0011	0.0037	−0.0014	−0.0001
Green	0.0015	−0.0029	0.0011	0.0029	−0.0005	−0.0003
Blue	0.0007	−0.0032	0.0011	0.0026	0.0003	−0.0008
	$pre1 = 41$, the others rest invariant			$pre2 = 51$, the others rest invariant		
Red	0.0028	−0.0020	−0.0021	−0.0001	0.0026	−0.0001
Green	0.0024	−0.0014	−0.0015	−0.0001	0.0027	0.0001
Blue	0.0015	−0.0010	−0.0011	−0.0015	0.0032	−0.0001
	$pre3 = 56$, the others rest invariant			$pre4 = 46$, the others rest invariant		
Red	0.0021	0.0016	0.0000	0.0038	0.0014	0.0051
Green	0.0020	0.0025	−0.0009	0.0043	0.0015	0.0045
Blue	0.0017	0.0027	−0.0017	0.0045	0.0008	0.0035

But due to the difference of computer configurations and code optimization ways, running speed can hardly be compared exactly (Table 16).

Considering the CPU frequency, our scheme is roughly faster than other recent schemes and as the speed scale as Faraoun's scheme which is focusing on the encryption performance while our scheme is better in the aspect of information entropy (in Sect. 3.5).

4 Conclusions

With the rapid development in digital image processing and widespread dissemination of digital multimedia data, the security problem of image information has been more and more important. For this purpose, the algorithm in this paper is proposed. Three characters of the algorithm determine its good performance. Firstly, two 3D chaotic systems are used as key gener-

Table 16 Speed analysis result (in Second)

Image Size	Proposed scheme	Chen's scheme (2012) [25]	Ping's scheme (2014)[26]	Faraoun's scheme (2014)[17]
Software Platform	Visual studio 2012 with OpenCV3	-	Mathematica 8.0	Delphi 6
Hardware Platform	Intel Core i5-4210U 1.70 GHz	-	2.9 GHz Intel Pentium Dual Core CPU	i7-2600 3.40 GHz platform
128*128	0.07	1.03	-	-
256*256	0.37	2.51	1.18	-
512*512	1.38	4.83	-	0.75

ators for three colours of colour images' pixels. Secondly, 4-Pixel Feistel structure and functions based on multiple chaotic maps are used to improve the statistic and diffusion properties of cipher image. Thirdly, dependent encryption progress is used to resist certain cryptanalysis methods, such as known-/chosen plaintext attack and chosen cipher attack. The simulation experiments, including histogram analysis, correlation of two adjacent pixels, NPCR and UACI, sensitivity to cipher image, information entropy, key sensitivity, key space analysis and speed analysis, show that the proposed algorithm has good statistic and diffusion properties and can resist many kinds of attacks, while performing well in speed.

Acknowledgments This work is supported by Major Program of National Natural Science Foundation of China (11290141), NSFC (11201018, 61402030), Fundamental Research of Civil Aircraft no. MJ-F-2012-04.

References

1. Matthews, R.: On the derivation of a chaotic encryption algorithm. Cryptologia **4**, 29–42 (1989)
2. Mao, Y.B., Chen, G., Lian, S.G.: A novel fast image encryption scheme based on the 3D chaotic baker map. Int. J. Bifurcat. Chaos **14**, 3613–3624 (2004)
3. Gao, H.J., Zhang, Y.S., Liang, S.Y., Li, D.Q.: A new chaotic algorithm for image encryption. Chaos Solitons Fractals **29**, 393–399 (2006)
4. Xiang, T., Liao, X.F., Tang, G.P., Chen, Y., Wong, K.W.: A novel block cryptosystem based on iterating a chaotic map. Phys. Lett. A **349**, 109–115 (2006)
5. Zhou, Q., Wong, K.-W., Liao, X., Xiang, T., Hu, Y.: Parallel image encryption algorithm based on discretized chaotic map. Chaos Solitons Fractals **00**, 1081–1092 (2007)
6. Liu, H., Wang, X.: Colour image encryption based on one time keys and robust chaotic maps. Comput. Math. Appl. **59**(10), 3320–3327 (2010)
7. Wang, X., Wang, X., Zhao, J., Zhang, Z.: Chaotic encryption algorithm based on alternant of stream cipher and block cipher. Nonlinear Dyn. **63**, 587–597 (2011)
8. Mirzaei, O., Yaghoobi, M., Irani, H.: A new image encryption method: parallel sub-image encryption with hyper chaos. Nonlinear Dyn. **67**, 557–566 (2012)
9. Patidar, V., Pareek, N.K., Sud, K.K.: A new substitution-diffusion based image cipher using chaotic standard and logistic maps. Commun. Nonlinear Sci. Numer. Simulat. **14**, 3056–3075 (2009)
10. Wang, X., Teng, L., Qin, X.: A novel colour image encryption algorithm based on chaos. Signal Process. **92**(4), 1101–1108 (2012)
11. Kanso, A., Ghebleh, M.: A novel image encryption algorithm based on a 3D chaotic map. Commun. Nonlinear Sci. Numer. Simulat. **17**, 2943–2959 (2012)
12. Shuang-Shuang, H., Le-Quan, M.: A color image encryption scheme based on generalized synchronization theorem. TELKOMNIKA Indones. J. Elect. Eng. **12**(1), 685–692 (2014)
13. Xiangjun, W., Bai, C., Kan, H.: A new color image cryptosystem via hyperchaos synchronization. Commun. Nonlinear Sci. Numer. Simulat. **19**, 1884–1897 (2014)
14. Wu, J., Guo, F., Liang, Y., Zhou, N.: Triple color images encryption algorithm based on scrambling and the reality-preserving fractional discrete cosine transform. Optik **125**, 4474–4479 (2014)
15. Sankaran, K.S., Santhosh Krishna, B.V.: A new chaotic algorithm for image encryption and decryption of digital color images. Int. J. Inform. Educ. Technol. **1**(2), 137–141 (2011)
16. Zhang, Y., Xiao, D.: Self-adaptive permutation and combined global diffusion for chaotic color image encryption. Int. J. Elect. Commun. **68**, 361–368 (2014)
17. Faraoun, K.M.: Fast encryption of RGB color digital images using a tweakable cellular automaton based schema. Opt. Laser Technol. **64**, 145–155 (2014)
18. Solak, E., Cokal, C., Yildiz, O.T., Biyikoglu, T.: Cryptanalysis of Fridrich's chaotic image encryption. Int. J. bifurcat. Chaos **20**(5), 1405–1413 (2010)

19. Fridirich, J.: Symmetric ciphers based on two-dimensional chaotic maps. Int. J. bifurcat. Chaos **8**(6), 1259–1284 (1998)
20. Li, C., Liu, Y., Xie, T., Michael, Z.Q.C.: Breaking a novel image encryption scheme based on improved hyperchaotic sequences. Nonlinear Dyn. **73**, 2083–2089 (2013)
21. Rhouma, R., Solak, E., Belghith, S.: Cryptanalysis of a new substitution-diffusion based image cipher. Commun. Nonlinear Sci. Numer. Simul. **15**, 1887–1892 (2010)
22. Li, C., Zahng, L.Y., Ou, R., Kwok-Wo, S., Shu, S.: Breaking a novel colour image encryption algorithm based on chaos. Nonlinear Dyn. **70**, 2383–2388 (2012)
23. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-Bit Blockcipher CLEFIA. FSE 2007, LNCS 4593, 181–195 (2007)
24. Patidar, V., Pareek, N.K., Purohit, G., Sud, K.K.: Modified substitution-diffusion image cipher using chaotic standard and logistic maps. Commun. Nonlinear Sci. Numer. Simul. **15**, 2755–2765 (2010)
25. Chen, L.: A novel image encryption scheme based on hyperchaotic sequences. J. Comput. Inform. Syst. **8**(10), 4159–4167 (2012)
26. Ping, P., Feng, X., Wang, Z.J.: Image encryption based on non-affine and balanced cellular automata. Signal Process. **10**, 5419–5429 (2014)