A novel image block cryptosystem based on a spatiotemporal chaotic system and a chaotic neural network*

Wang Xing-Yuan(王兴元)† and Bao Xue-Mei(鲍雪梅)

Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, *China*

(Received 28 June 2012; revised manuscript received 15 November 2012)

In this paper, we propose a novel block cryptographic scheme based on a spatiotemporal chaotic system and a chaotic neural network (CNN). The employed CNN comprises a 4-neuron layer called a chaotic neuron layer (CNL), where the spatiotemporal chaotic system participates in generating its weight matrix and other parameters. The spatiotemporal chaotic system used in our scheme is the typical coupled map lattice (CML), which can be easily implemented in parallel by hardware. A 160-bit-long binary sequence is used to generate the initial conditions of the CML. The decryption process is symmetric relative to the encryption process. Theoretical analysis and experimental results prove that the block cryptosystem is secure and practical, and suitable for image encryption.

Keywords: image encryption, block cryptosystem, chaotic neural network, coupled map lattice

PACS: 05.45.Vx, 05.45.Ra, 87.85.dq, 05.45.−a **DOI:** 10.1088/1674-1056/22/5/050508

1. Introduction

With the rapid development of computers and the Internet, information security has become an increasingly serious issue. During the past few years, some traditional cryptosystems have been broken one after another, so designing new cryptosystems becomes important and necessary. In recent years, cryptosystems based on chaos theory have attracted a great deal of attention.[1–7] A chaotic system has several significant features, such as sensitive dependence on initial conditions, pseudo-randomness, and ergodicity.[8] These features characterize very good properties of diffusion and confusion, which are of great importance in cryptography. Up to now, many chaotic cryptosystems have been proposed, but most of them have proven to be insecure. The most serious problem in individual chaotic systems is that their chaotic dynamic properties degrade rapidly when they are realized with finite precisions in digital computers.[9]

CML-based spatiotemporal chaotic systems have been widely used in chaotic cryptography in recent years[10–13] for their excellent chaotic dynamic properties, which could maintain much longer periodicity in digitalization and gain good performance in cryptography.[14–16] Moreover, the CML could be implemented in parallel by hardware, and the CML has more initial parameters and its key space is larger. In our algorithm, the CML-based spatiotemporal chaotic system participates in generating the weight matrix and other parameters of the CNL.

Due to their good properties such as high nonlinearity, parameter sensitivity, and learning ability, neural networks have been widely used as an alternative to encryption.[17–25] As a combination of neural network and chaos, a CNN has the characteristics of both a neural network and chaos. Especially, it has more complex dynamic behaviours, so it is expected to have better performance as an encryption tool. Till now, various CNN-based encryption algorithms have been reported, which can be classified into two classes: stream cipher and block cipher. The first kind uses neural networks to produce pseudorandom sequences.[21,24] The security of these stream ciphers depends on the sequence randomness. However, it is still difficult to generate the real random sequence in practice, and its randomness should be analysed in detail. These factors restrict the applications of stream ciphers. The second kind makes use of the properties of a CNN to encrypt plaintext block by block,[17–19,25] which is used as the encryption scheme in this paper.

In this paper, we propose a novel block cipher based on CML and CNN derived from different fields. The cryptosystem is based on a CNN which is composed of only one 4-neuron CNL. A plaintext block containing four pixels of the plain image constitutes the input of the CNL, and the four encoded pixels contained in a ciphertext block constitute the output of the CNL. A 160-bit-long binary sequence is used to generate the initial conditions of the CML. The weight matrix and other parameters of the CNL are generated from this CML. Simulations show that the proposed image encryption algorithm has extraordinarily high security.

The remainder of the paper is organized as follows. In Section 2, the CML and the structure of the CNN used in the proposed algorithm are introduced. In Section 3, the encryption and decryption algorithms are described. In Section 4, the simulation results are presented. Security and performance analysis are given in Section 5. Finally, this paper is concluded

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61173183, 60973152, and 60573172), the Doctoral Program Foundation of Institution of Higher Education of China (Grant No. 20070141014), the Program for Excellent Talents in Universities of Liaoning Province, China (Grant No. LR2012003), the Natural Science Foundation of Liaoning Province, China (Grant No. 20082165), and the Fundamental Research Funds for the Central Universities of China (Grant No. DUT12JB06).

†Corresponding author. E-mail: wangxy@dlut.edu.cn

© 2013 Chinese Physical Society and IOP Publishing Ltd

in Section 6.

2. Preliminary materials

2.1. Description of CML

CML is a dynamic system with discrete time and discrete space. It consists of nonlinear maps called local maps on the lattice sites. Each local map is coupled with other local maps governed by certain coupling rules. Because of the intrinsic nonlinear dynamics of each local map and the diffusion due to the spatial coupling among the local maps, CML exhibits spatiotemporal chaos.[26] The CML introduced by Kaneko[27] is described as

$$x_{n+1}(j) = (1-\varepsilon)\tau(x_n(j)) + \frac{\varepsilon}{2}\{\tau[x_n(j+1)] + \tau[x_n(j-1)]\}, \quad (j = 1,2,\ldots,L), \tag{1}$$

where n is the time index, j is the lattice site index, $\varepsilon \in (0,\ 1)$ is a coupling constant, and L is the lattice length. The periodic boundary condition, i.e., $x_n(j) = x_n(L-j)$ for any valid j, is used in the CML. The $\tau(x)$ is a local map given by

$$\tau(x) = \mu x(1-x), \quad (x \in [0,1],\ \mu \in [0,4]), \tag{2}$$

which is chaotic when $\mu > 3.57$.

In our scheme, L and μ are set to be 16 and 4, respectively. The parameter ε is dependent on the plain image and serves as a secret key

$$\varepsilon = \sum_{x\in[1,N],y\in[1,N]} \frac{F(x,y)}{N\times N\times 255}, \tag{3}$$

where F is an $N \times N$ image, and $F(x,y)$ is the pixel value (8-bit) at the specified coordinate (x,y). For different plain images, our scheme has different values of secret key ε, so it could resist plaintext attack effectively.

The CML could be implemented in parallel by hardware. If available, the local maps on the lattice sites (namely, the L logistic maps) could be implemented in parallel using parallel computing techniques, such as CUDA.

2.2. Description of the structure of CNN

The proposed cryptosystem is based on a CNN containing a 4-neuron CNL. The CNL used in our scheme is 4-input 4-output and includes four neurons, as illustrated in Fig. 1. The input of the CNL is a plaintext block of the plain image, i.e., P_i (i is the block index and P_i contains four pixels). The linear combination of the four pixel values first passes through two stages, i.e., nonlinear normalization and XOR operation, and then enters into the linear activation function f. The output of the CNL comprises a ciphertext block of the ciphered image denoted by C_i. More details are presented below.

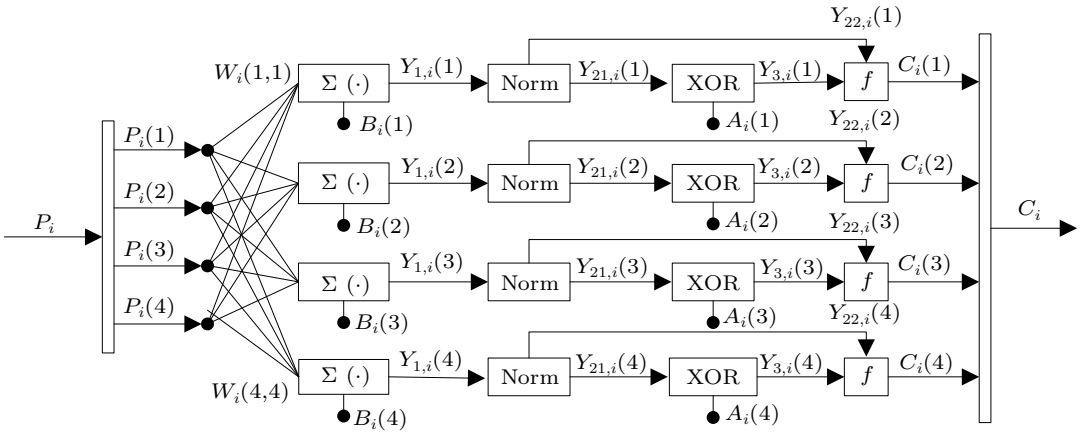


Fig. 1. Chaotic neuron layer.

3. Proposed block cryptographic algorithm

3.1. Encryption algorithm

The flowchart of the proposed encryption process is shown in Fig. 2. The chaotic parameter generator CML is used to generate the weight matrix and other necessary parameters of the CNL.

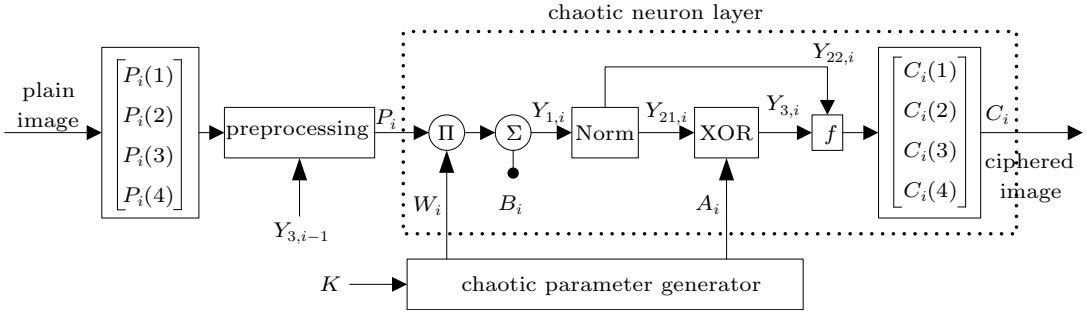


Fig. 2. Flowchart of encryption process.

In more detail, the encryption process may be summarized in the following steps.

Step 1 In our scheme, a 160-bit-long binary sequence K is used as a secret key. Splitting K into sixteen groups, we have

$$K=\underbrace{k_1k_2\cdots k_{10}}_{K_1}\underbrace{k_{11}k_{12}\cdots k_{20}}_{K_2}\cdots\underbrace{k_{10(j-1)+1}k_{10(j-1)+2}\cdots k_{10j}}_{K_j}\cdots\underbrace{k_{151}k_{152}\cdots k_{160}}_{K_{16}},\quad (j=1,2,\ldots,16). \tag{4}$$

They are further mapped into the sixteen initial conditions of Eq. (1) using the following rule:

$$x_0(j)=\frac{K_j}{2^{10}},\quad (j=1,2,\ldots,16). \tag{5}$$

Step 2 Dividing the plain image P into l blocks of 4 pixels yields

$$P=\underbrace{p_1p_2p_3p_4}_{P_1}\underbrace{p_5p_6p_7p_8}_{P_2}\cdots\underbrace{p_{4(i-1)+1}p_{4(i-1)+2}p_{4(i-1)+3}p_{4i}}_{P_i}\cdots\underbrace{p_{4(l-1)+1}p_{4(l-1)+2}p_{4(l-1)+3}p_{4l}}_{P_l}, \tag{6}$$

where $i=1,2,\ldots,l$ is the block index.

Step 3 Encrypt the i-th plaintext block P_i (the initial value of i is 1), and iterate the CML once to obtain $x_i(j)$ ($j=1,2,\ldots,16$). Then, construct matrices $\boldsymbol{P}_i$, $\boldsymbol{W}_i$, $\boldsymbol{A}_i$, and $\boldsymbol{B}_i$ in the following way:

$$\boldsymbol{P}_i=[p_{4(i-1)+1},\ p_{4(i-1)+2},\ p_{4(i-1)+3},\ p_{4i}]^{\mathrm{T}}, \tag{7}$$

$$\boldsymbol{W}_i=0.5\begin{bmatrix} x_i(1) & x_i(5) & x_i(9) & x_i(13)\\ x_i(2) & x_i(6) & x_i(10) & x_i(14)\\ x_i(3) & x_i(7) & x_i(11) & x_i(15)\\ x_i(4) & x_i(8) & x_i(12) & x_i(16)\end{bmatrix}+0.5\boldsymbol{I}, \tag{8}$$

$$\begin{cases} a_{i1}=\mathrm{mod}\left(\lfloor(x_i(1)-\lfloor x_i(1)\rfloor)\times 10^{14}\rfloor,256\right),\\ a_{i2}=\mathrm{mod}\left(\lfloor(x_i(6)-\lfloor x_i(6)\rfloor)\times 10^{14}\rfloor,256\right),\\ a_{i3}=\mathrm{mod}\left(\lfloor(x_i(11)-\lfloor x_i(11)\rfloor)\times 10^{14}\rfloor,256\right),\\ a_{i4}=\mathrm{mod}\left(\lfloor(x_i(16)-\lfloor x_i(16)\rfloor)\times 10^{14}\rfloor,256\right),\end{cases} \tag{9}$$

$$\boldsymbol{A}_i=[a_{i1},\ a_{i2},\ a_{i3},\ a_{i4}]^{\mathrm{T}}, \tag{10}$$

$$\boldsymbol{B}_i=[0,\ 0,\ 0,\ 0]^{\mathrm{T}}, \tag{11}$$

where $\boldsymbol{P}_i$ is the input of the CNL, $\boldsymbol{W}_i$ is the 4×4 weight matrix, $\boldsymbol{B}_i$ is the bias matrix and is always a 4×1 zero matrix for each plaintext block, $\boldsymbol{A}_i$ is a 4×1 integral matrix, mod (x, y) returns the remainder of x divided by y, $\lfloor x\rfloor$ rounds x to the nearest integer less than or equal to x, $\boldsymbol{A}^{\mathrm{T}}$ is the transposed matrix of $\boldsymbol{A}$, and $\boldsymbol{I}$ is a 4×4 identity matrix, aiming at avoiding the singularity problem in $\boldsymbol{W}_i$.

Step 4 Considering Fig. 2, in order to generate the ciphertext block $\boldsymbol{C}_i$, several operations are applied to the plaintext block P_i which is expressed as

$$\boldsymbol{Y}_{1,i}=\boldsymbol{W}_i\boldsymbol{P}_i+\boldsymbol{B}_i, \tag{12}$$

where $\boldsymbol{Y}_{1,i}$ is a 4×1 matrix.

The following normalization which maps the element values of $\boldsymbol{Y}_{1,i}$ into interval [0, 255] is used:

$$\boldsymbol{Y}_{2,i}=\alpha\boldsymbol{Y}_{1,i}+\beta, \tag{13}$$

$$\boldsymbol{Y}_{21,i}=\lfloor\boldsymbol{Y}_{2,i}\rfloor, \tag{14}$$

$$\boldsymbol{Y}_{22,i}=\boldsymbol{Y}_{2,i}-\lfloor\boldsymbol{Y}_{2,i}\rfloor, \tag{15}$$

where α and β serve as the secret keys, and $\boldsymbol{Y}_{2,i}$, $\boldsymbol{Y}_{21,i}$, and $\boldsymbol{Y}_{22,i}$ are all 4×1 matrices. The XOR operation is also used

$$\boldsymbol{Y}_{3,i}=\boldsymbol{Y}_{21,i}\oplus\boldsymbol{A}_i, \tag{16}$$

where the symbol $\oplus$ represents the exclusive OR operation bit-by-bit, and $\boldsymbol{Y}_{3,i}$ is a 4×1 matrix. Then, the following linear activation function is used:

$$\boldsymbol{C}_i=f(\boldsymbol{Y}_{3,i},\boldsymbol{Y}_{22,i})=\boldsymbol{Y}_{3,i}+\boldsymbol{Y}_{22,i}. \tag{17}$$

At this point, we obtain the ciphertext block with the block index i. If all the plaintext blocks have already been encrypted, encryption process is completed. Otherwise, we should set $i=i+1$ and proceed with Step 5 to encrypt the next plaintext block.

Step 5 Perform the following manipulation with $\boldsymbol{P}_i$ and $\boldsymbol{Y}_{3,i-1}$:

$$\boldsymbol{P}_i=\boldsymbol{P}_i\oplus\boldsymbol{Y}_{3,i-1}, \tag{18}$$

which is exactly the preprocessing operation. For the first plaintext block, we skip the preprocessing operation. Then, we go to Step 3.

3.2. Decryption algorithm

The decryption process is the reverse of the encryption process. The flowchart of the decryption process is shown in Fig. 3.

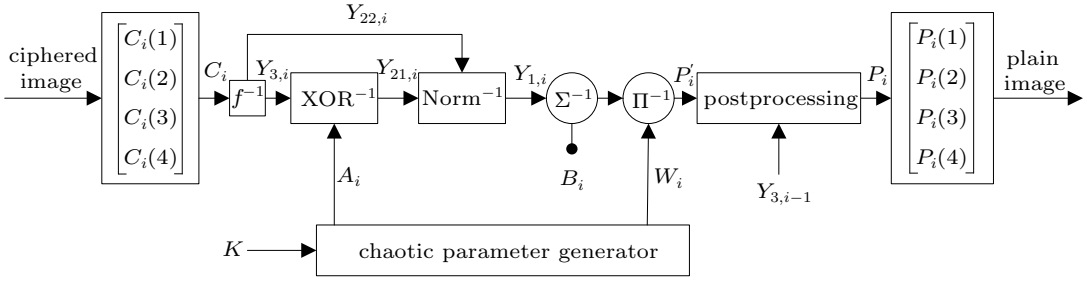


Fig. 3. Flowchart of the decryption process.

From this figure, the decryption process can be summarized as follows.

Step 1 Generate the sixteen initial conditions of the CML, i.e., $x_0(j)$ $(j=1,2,\ldots,16)$, from the 160-bit-long binary sequence K using Eqs. (4) and (5).

Step 2 Dividing the ciphered image C into l blocks of 4 pixels yields

$$C=\underbrace{c_1c_2c_3c_4}_{C_1}\underbrace{c_5c_6c_7c_8}_{C_2}\cdots\underbrace{c_{4(i-1)+1}c_{4(i-1)+2}c_{4(i-1)+3}c_{4i}}_{C_i}\cdots\underbrace{c_{4(l-1)+1}c_{4(l-1)+2}c_{4(l-1)+3}c_{4l}}_{C_l}, \tag{19}$$

where i is the block index and $i=1,2,\ldots,l$.

Step 3 Decrypt the i-th ciphertext block C_i (the initial value of i is 1) and iterate the CML once to obtain $x_i(j)(j=1,2,\ldots,16)$. Then, construct matrices $\boldsymbol{W}_i$, $\boldsymbol{A}_i$, and $\boldsymbol{B}_i$ according to Eqs. (8), (10), and (11), respectively.

Step 4 Decrypt the i-th ciphertext block as follows:

$$\boldsymbol{Y}_{3,i}=\lfloor C_i\rfloor, \tag{20}$$

$$\boldsymbol{Y}_{22,i}=C_i-\lfloor C_i\rfloor, \tag{21}$$

using XOR operation:

$$\boldsymbol{Y}_{21,i}=\boldsymbol{Y}_{3,i}\oplus A_i, \tag{22}$$

and inverse normalization:

$$\boldsymbol{Y}_{2,i}=\boldsymbol{Y}_{21,i}+\boldsymbol{Y}_{22,i}, \tag{23}$$

$$\boldsymbol{Y}_{1,i}=\alpha^{-1}(\boldsymbol{Y}_{2,i}-\beta), \tag{24}$$

$$\boldsymbol{P}'_i=\boldsymbol{W}_i^{-1}(\boldsymbol{Y}_{1,i}-B_i). \tag{25}$$

Step 5 Implement the postprocessing operation, as illustrated in Fig. 3. If $i=1$, the corresponding deciphered block can be obtained as

$$\boldsymbol{P}_i=\boldsymbol{P}'_i. \tag{26}$$

Otherwise,

$$\boldsymbol{P}_i=\boldsymbol{P}'_i\oplus\boldsymbol{Y}_{3,i-1}. \tag{27}$$

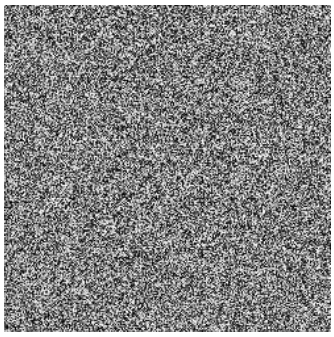
If all the ciphertext blocks have already been decrypted, the decryption process is completed. Otherwise, we set $i=i+1$ and return to Step 3.

4. Simulation results

We use MATLAB 7.6.0 to run the programs. Our simulation results are shown in Fig. 4. The 256×256 grey-scale images "Lena", "Baboon", and "Cameraman" (as shown in Figs. 4(a), 4(d), and 4(g), respectively) are used as the plain images. The secret keys are set as follows: $\varepsilon=0.3850$, $\alpha=0.4820$, and $\beta=-3.4676$. The ciphered images are respectively shown in Figs. 4(b), 4(e), and 4(h), which are no longer intelligible. Their recovered images are shown in Figs. 4(c), 4(f), and 4(i), when we decrypt the ciphered images with the same key.



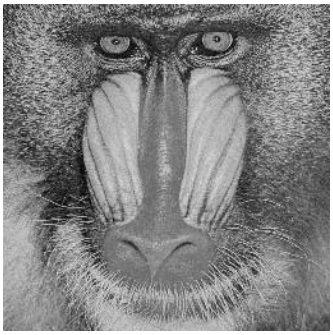
(a) Plain image of "Lena".



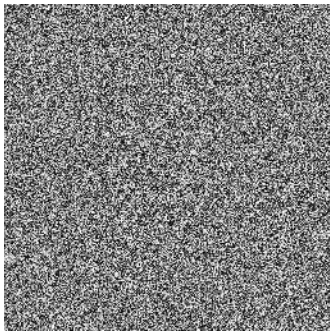
(b) Ciphered image of "Lena".



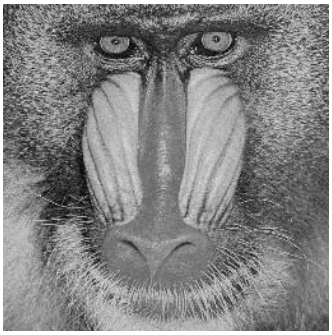
(c) Decryption of "Lena".



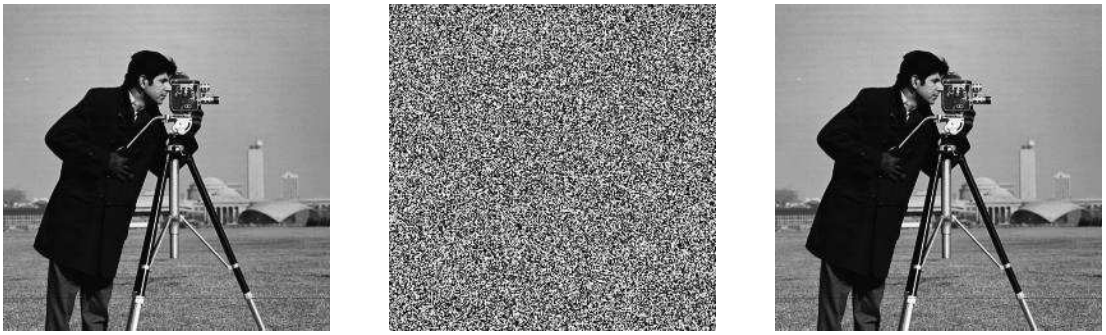
(d) Plain image of "Baboon".



(e) Ciphered image of "Baboon".



(f) Decryption of "Baboon".



(g) Plain image of "Cameraman". (h) Ciphered image of "Cameraman". (i) Decryption of "Cameraman".

Fig. 4. Encryption and decryption results of (a)–(c) "Lena", (d)–(f) "Baboon", and (g)–(i) "Cameraman".

Additionally, the ciphered image of "Lena" obtained by the proposed algorithm is compared with the results based on CNN-based algorithms proposed in Refs. [17] and [18] (referred to as Algorithm 17 and Algorithm 18), as illustrated in Fig. 5. Note that the ciphered images in this paper are all results after one-round encryption. Although the encryption results of Algorithms 17 and 18 also achieve good effects, these algorithms still have some security issues. We will discuss their security issues in the following.

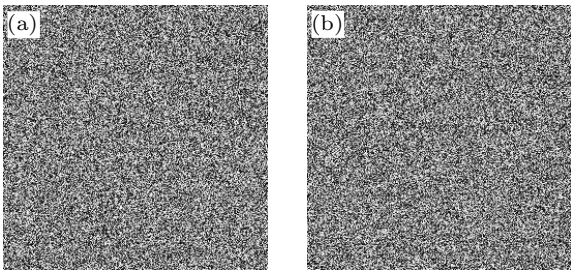


Fig. 5. Ciphered images obtained by other algorithms. (a) Ciphered image of "Lena" obtained by Algorithm 17; (b) ciphered image of "Lena" obtained by Algorithm 18.

5. Security and performance analysis

A good encryption scheme should be robust against all kinds of attacks, such as brute-force attack, statistical attack, differential attack, and plaintext attack. In addition, performance cannot be ignored when designing a cryptosystem. Security and performance analysis has been performed on the proposed algorithm, Algorithm 17, and Algorithm 18.

5.1. Key space

The size of the key space characterizes the capability of resisting brute-force attack. A short key means that the best encryption algorithm can be broken by exhaustive search (also known as brute-force attack) in a reasonable amount of time, while the reverse is not true. In our algorithm, a 160-bit-long binary sequence is used as secret keys. Moreover, the parameters $\alpha, \beta, \varepsilon$ are also used as the secret keys. The complexity of brute-force is great, so the key space is large enough for common applications to resist brute-force attack.

5.2. Distribution

The distribution of the ciphered image is a major concern. A histogram is often defined as a graph that shows the distribution of pixel values of an image. If it is not flat enough, a certain amount of information can be guessed by the statistical attack opponent. This makes cipher-only attack easier through analysing the statistic property of the ciphered image. Therefore, a flat distribution is desirable in cryptography.

Figure 6 illustrates the histograms of plain images "Lena", "Baboon", "Cameraman", and their ciphered images obtained by the proposed algorithm. It is clear from Fig. 6 that the proposed algorithm results in a very flat ciphered image distribution, and that statistical attack is not effective to our algorithm.

5.3. Randomness test

Cryptography should map a given plain image into a random ciphered image. Therefore, the randomness test of the key stream is extremely important. We adopt the SP 800-22 published by the NIST[28] to test and verify the statistical properties of the key stream generated by CML. The statistical test consists of 16 tests. Each of the tests is based on the value of P, which is the function of the data. If $P = 1$, then the test sequence is completely random; if $P = 0$, then the test sequence is completely non-random. For each test, a significance level, say α, is chosen. If $P \geq \alpha$, then the test sequence is random, or else the test sequence is non-random. Usually, the interval of the significance level is $[0.001, 0.01]$. In the following tests, α is set to be 0.01. The test results are shown in Table 1. We can see that all the values of P are greater than 0.01, which indicates that the key stream is a random sequence.

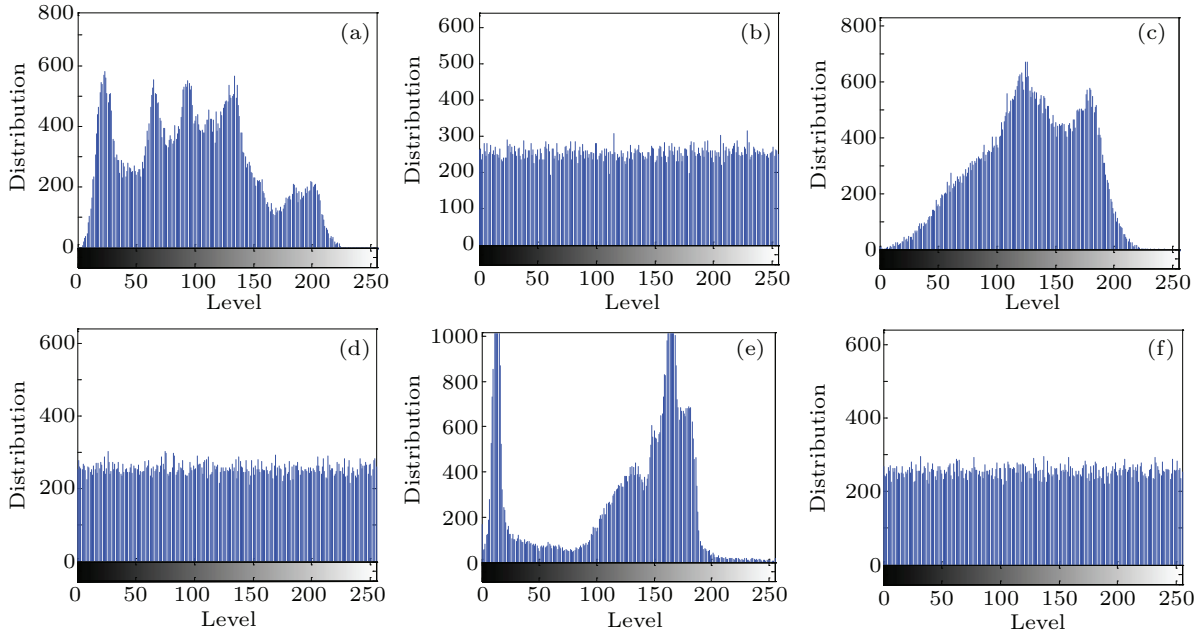


Fig. 6. Histograms of the plain images and ciphered images. (a) Distribution of plain image "Lena"; (b) distribution of ciphered image "Lena"; (c) distribution of plain image "Baboon"; (d) distribution of ciphered image "Baboon"; (e) distribution of plain image "Cameraman"; (f) distribution of ciphered image "Cameraman".

Table 1. Randomness test of the key stream.

Statistical tests	P
Frequency	0.839917
Block frequency	0.386801
Cusum-forward	0.495120
Cusum-reverse	0.349469
Runs	0.662866
Longest runs of ones	0.683711
Rank	0.071874
DFT	0.901879
Non-overlapping template matching	0.935345
Overlapping template matching	0.735470
Compression	0.332584
Universal	0.387246
Approximate entropy	0.665574
Random excursions	0.776808
Random excursions variant	0.399402
Linear complexity	0.709066
Serial	0.122958

5.4. Information entropy

Information entropy is the most important feature of randomness. Let m be the information source, and the formula for calculating information entropy is

$$H(m) = \sum_{i=0}^{2^n-1} p(m_i) \log_2 \frac{1}{p(m_i)}, \tag{28}$$

where $p(m_i)$ represents the probability of symbol m_i. Assume that there are 2^8 states of the information source and they appear with the same probability. According to Eq. (28), we can obtain the ideal $H(m) = 8$, which shows that the information is completely random. The information entropy of the ciphered image should be close to 8 after encryption. The closer it is to 8, the smaller the possibility for the scheme to divulge information is. Information entropies of the plain images "Lena", "Baboon", and "Cameraman" are 7.5683, 7.3903, and 7.1048, respectively. Information entropies of the ciphered images obtained by the proposed algorithm are compared with that obtained by Algorithms 17 and 18 in Table 2. From the table, test results based on the proposed algorithm are closer to the ideal value 8 than based on Algorithms 17 and 18. We can conclude that the ciphered images obtained by the proposed algorithm could hardly divulge information.

Table 2. Information entropies of the ciphered images obtained by the proposed algorithm, and Algorithms 17 and 18.

	Proposed algorithm	Algorithm 17	Algorithm 18
Lena	7.9968	7.9895	7.9875
Baboon	7.9973	7.9890	7.9883
Cameraman	7.9973	7.9892	7.9886

5.5. Correlation

Correlation indicates the strength and direction of a linear relationship between two random variables. In image processing, it is usually employed to investigate the relationship between two adjacent pixels. The correlation between adjacent pixels is usually high in a recognizable image. The less the correlation of two adjacent pixels, the safer the image is. In order to investigate the confusion effect of the ciphered image, the correlations between two vertically adjacent pixels, two horizontally adjacent pixels and two diagonally adjacent pixels are tested, respectively. The correlation between two adjacent pixels is calculated using the following formula:

$$r_{xy} = \frac{\operatorname{cov}(x,y)}{\sqrt{D(x)}\sqrt{D(y)}}, \tag{29}$$

where

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

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i,$$
$$D(x) = \frac{1}{N}\sum_{i=1}^{N} (x_i - E(x))^2. \quad (30)$$

As there are three kinds of adjacent directions as mentioned above, we randomly select 1000 pairs of adjacent pixels in each direction from the plain images "Lena", "Baboon", "Cameraman" and their ciphered images. Their correlation coefficients are then calculated. Test results are listed in Table 3.

Table 3 shows that there is strong correlation between adjacent pixels of each direction in plain images since the correlation coefficients are all close to 1. While in the ciphered images obtained by the proposed algorithm, correlation coefficients are all smaller than 0.001, indicating a negligible correlation between adjacent pixels.

Additionally, taking the image "Lena" for example, we present the correlation distribution of "Lena" and its ciphered image obtained by the proposed algorithm in each direction in Fig. 7. Figures 7(a)–7(c) show the correlation distributions of the plain image, while Figs. 7(d)–7(f) display the correlation distributions of the ciphered image. The strong correlation between adjacent pixels of the plain image is evident as all the dots are congregated along the diagonal in Figs. 7(a)–7(c). However, in Figs. 7(d)–7(f), the dots are scattered over the entire plane, which indicates that the correlation is greatly reduced in the ciphered image.

Table 3. Correlation coefficients of the plain images and their ciphered images obtained by the proposed algorithm, Algorithm 17, and Algorithm 18.

		Horizontal	Vertical	Diagonal
Plain images	Lena	0.9718	0.9526	0.9419
	Baboon	0.6421	0.7120	0.6448
	Cameraman	0.9549	0.9414	0.9212
Proposed algorithm	Lena	-5.6137×10^{-4}	1.5491×10^{-4}	9.2467×10^{-4}
	Baboon	4.0165×10^{-4}	7.5338×10^{-4}	2.2773×10^{-4}
	Cameraman	6.7809×10^{-4}	-7.6341×10^{-4}	3.1281×10^{-4}
Algorithm 17	Lena	0.0041	-0.0027	0.0425
	Baboon	−0.0194	−0.0020	−0.0140
	Cameraman	−0.0364	−0.0065	0.0234
Algorithm 18	Lena	0.0122	−0.0299	−0.0108
	Baboon	−0.0183	−0.0174	−0.0088
	Cameraman	0.0517	0.0440	0.0075

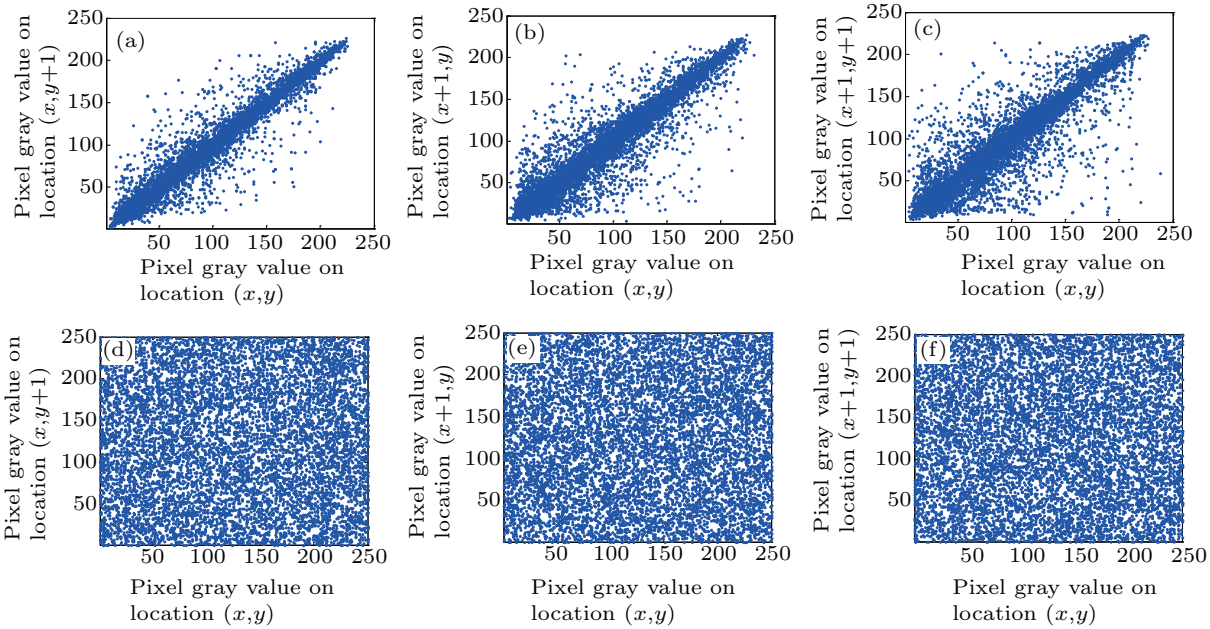


Fig. 7. (color online) Correlation distributions. (a)–(c) Correlations between two horizontally, vertically, and diagonally adjacent pixels of the plain image "Lena"; (d)–(f) correlations between two horizontally, vertically, and diagonally adjacent pixels of the ciphered image obtained by the proposed algorithm, respectively.

5.6. NPCR and UACI

Number of pixels change rate (NPCR) refers to the number of pixels change rate while one pixel of the plain image changes. The closer the NPCR gets to 100%, the more sensitive the cryptosystem is to the plain image, and the more effective it seems for the cryptosystem to resist plaintext at-

tack. Unified average changing intensity (UACI) refers to the average value of differences in intensity between the plain image and ciphered image. The bigger the UACI is, the better the cryptosystem resists the differential attack. Here are the formulas to calculate NPCR and UACI,

$$\mathrm{NPCR} = \frac{\sum_{ij} D(i,j)}{WH} \times 100\%, \tag{31}$$

$$\mathrm{UACI} = \frac{1}{WH}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\%, \tag{32}$$

where W and H represent the width and height of the image, respectively; C_1 and C_2 are respectively the ciphered images before and after one pixel of the plain image is changed. For the pixel at position (i,j), if $C_1(i,j) \neq C_2(i,j)$, let $D(i,j) = 1$; otherwise, let $D(i,j) = 0$. The NPCR and UACI of images "Lena", "Baboon", and "Cameraman" based on the proposed algorithm, Algorithm 17, and Algorithm 18 are listed in Table 4. The results show that the proposed algorithm could resist plaintext attack and differential attack effectively, while Algorithms 17 and 18 have security issues.

Table 4. NPCRs and UACIs of images "Lena", "Baboon", and "Cameraman" based on the proposed algorithm, Algorithm 17, and Algorithm 18.

	Algorithm	NPCR	UACI
Proposed algorithm	Lena	99.60%	33.40%
	Baboon	99.59%	33.63%
	Cameraman	99.61%	33.38%
Algorithm 17	Lena	9.7656×10^{-4}	4.6321×10^{-4}
	Baboon	9.7656×10^{-4}	4.7757×10^{-4}
	Cameraman	9.7656×10^{-4}	4.6865×10^{-4}
Algorithm 18	Lena	0	0
	Baboon	0	0
	Cameraman	0	0

5.7. Sensitivity

An excellent cryptosystem should be sensitive to the secret keys as well as the plaintext. Taking secret keys ε and K for example, sensitivity tests on the image "Lena" based on the proposed algorithm have been performed as shown in Fig. 8. Figure 8(a) shows the differences between two ciphered images when ε is changed from 0.3850 to 0.3850000000000001, while the other keys are the same. Similarly, the differences are shown in Fig. 8(b) when 1 bit of the sequence K is changed. Therefore, it can be concluded that the proposed scheme is sensitive to the secret keys. Figure 8(c) shows the differences between two ciphered images when 1 bit of the pixel data of the plain image is changed. It can be seen that the proposed scheme is sensitive to the plaintext. A tiny change in the plain image leads to dramatic change in the ciphered image. The high sensitivity to plaintext ensures that the cryptosystem could resist plaintext attack. Moreover, we implement the plaintext sensitivity tests by Algorithm 17 and Algorithm 18. The results are shown in Figs. 8(d) and 8(e). A tiny change in the plain image leads to little change in the ciphered image. It demonstrates again that Algorithms 17 and 18 have security issues.

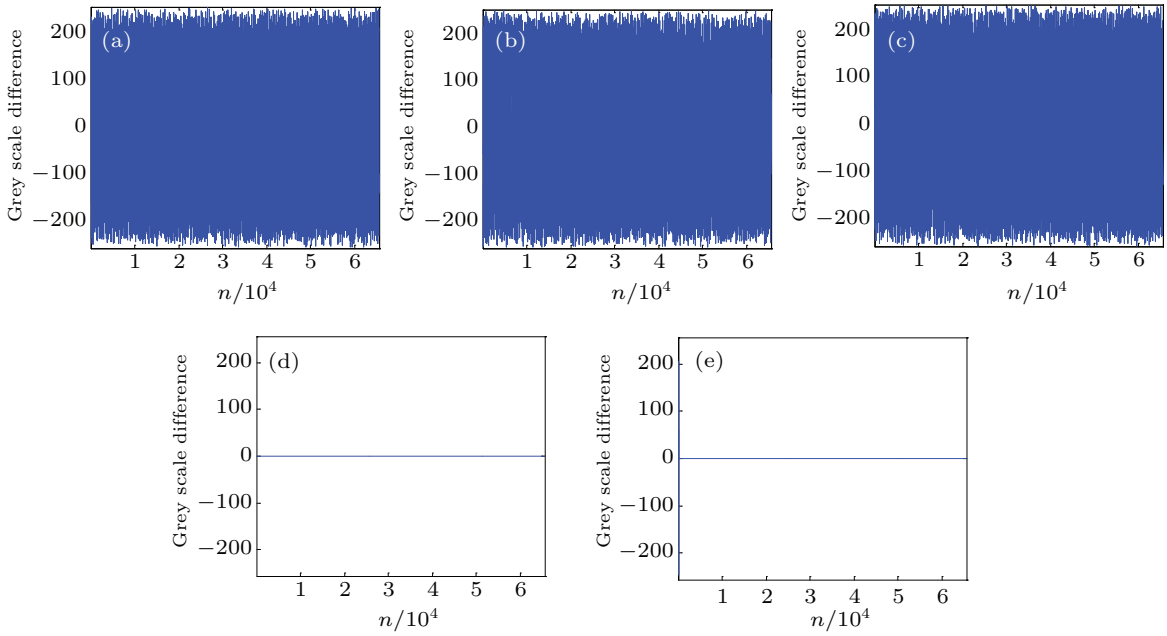


Fig. 8. (color online) Sensitivity tests on the image "Lena", showing (a) differences between two ciphered images when ε is changed based on the proposed algorithm, (b) differences between two ciphered images when 1 bit of the sequence K is changed based on the proposed algorithm, (c) differences between two ciphered images when 1 bit of the pixel data of the plain image is changed based on the proposed algorithm, (d) differences between two ciphered images when 1 bit of the pixel data of the plain image is changed based on Algorithm 17, and (e) differences between two ciphered images when 1 bit of the pixel data of the plain image is changed based on Algorithm 18.

5.8. Known-plaintext attack and chosen-plaintext attack

In the algorithm, we adopt several measures to ensure its ability to resist plaintext attack.

The parameter of the CML, say ε, is dependent on the plain image and serves as a secret key, as illustrated in Eq. (3). For different plain images, our scheme has different values of secret key ε, so it could resist plaintext attack effectively. It is a novelty of the proposed algorithm.

Moreover, the cryptosystem adopts the ciphertext block chaining mode. The encryption result of each byte is influenced by the capability of the previous bytes. The high sensitivity to plaintext ensures that the cryptosystem could resist plaintext attack.

The weight matrix $\boldsymbol{W}_i$, the integral matrix $\boldsymbol{A}_i$, and the parameters α and β in the normalization step are unknown to attackers. The proposed algorithm makes the relationship between plaintext and ciphertext nonlinear. Factors above prevent attackers from recovering these values. For example, in order to recover $\boldsymbol{W}_i$, the values of $\boldsymbol{A}_i$, α, and β should be broken first. However, these values could only be attacked through brute-force attack because of the nonlinear relationship. Brute-force attack is not effective because the large key space makes exhaustive search impossible in a reasonable amount of time. That is, the known-plaintext or chosen-plaintext attack is not effective for our algorithm.

5.9. Performance analysis

We use Matlab 7.6.0 to run the encryption and decryption programs on a personal computer. As shown in Table 5, the encryption and decryption time of the proposed algorithm are compared with those of Algorithms 17 and 18. The encryption speed and decryption speed based on the proposed algorithm are both very fast. The CML is suitable for implementation in parallel by hardware. In this way, the performance speed would be much faster and the cryptosystem would be even safer. Although the speeds of the other two algorithms are fast, they have security issues.

Table 5. Performances of the proposed algorithm, Algorithm 17, and Algorithm 18.

Algorithm		Encryption time/s	Decryption time/s
Proposed algorithm	Lena	0.224892	0.189967
	Baboon	0.207651	0.188195
	Cameraman	0.198710	0.190927
Algorithm 17	Lena	0.121671	0.099979
	Baboon	0.096275	0.096694
	Cameraman	0.101352	0.102832
Algorithm 18	Lena	1.025887	0.962677
	Baboon	0.962365	0.954266
	Cameraman	0.955995	1.001373

6. Conclusion

In this paper, we propose a block cryptographic scheme based on a CML and a CNN. The cryptosystem is based on a 4-input 4-output neural network that is composed of a 4-neuron CNL. A 160-bit-long binary sequence is used to generate the initial conditions of the CML. The weight matrix and other parameters of the CNL are generated by the CML. The proposed scheme is quite simple, so it is easy to implement. Experimental results and theoretical analysis show that the scheme is able to resist differential attack, brute-force attack, statistical attack, and plaintext attack, so it has extraordinarily high security. Meanwhile, the scheme offers satisfying speed. It is expected to attract more researchers in this field.

References

[1] Zhang Y Q and Wang X Y 2011 *Chin. Phys. Lett.* **28** 020505

[2] Du M K, He B and Wang Y 2011 *Chin. Phys. Lett.* **28** 010503

[3] Xu S J, Chen X B and Zhang R 2012 *Phys. Lett. A* **376** 1003

[4] Wang Z, Huang X and Li N 2012 *Chin. Phys. B* **21** 050506

[5] Jin J X and Qiu S S 2010 *Acta Phys. Sin.* **59** 0792 (in Chinese)

[6] Tong X J, Wang Z and Zuo K 2012 *Chin. Phys. B* **21** 020506

[7] Gao T G and Chen Z Q 2008 *Phys. Lett. A* **372** 394

[8] Liu B Z and Peng J H 2004 *Nonlinear Dynamics* (vol. 1) (Beijing: High Education Press) pp. 20–21 (in Chinese)

[9] Wheeler D D 1989 *Cryptologia.* **13** 243

[10] Wang X Y and Teng L 2012 *Nonlinear Dyn.* **67** 365

[11] Ge X, Liu F L, Lu B and Wang W 2011 *Phys. Lett. A* **375** 908

[12] Sun F Y and Wang L Z 2011 *Chin. Phys. B* **20** 040506

[13] Wang Y, Wong K W, Liao X F and Chen G R 2011 *Appl. Soft Comput.* **11** 514

[14] Xiang T, Wong K W and Liao X F 2007 *Chaos* **17** 023115

[15] Wang S H, Liu W R, Lu H P, Kuang J Y and Hu G 2004 *Int. J. Mod. Phys. B* **18** 2617

[16] Wang X G, Zhan M, Lai C H and Gang H 2004 *Chaos* **14** 128

[17] Lian S G 2009 *Neurocomputing* **72** 1296

[18] Bigdeli N, Farid Y and Afshar K 2012 *Eng. Appl. Artif. Intel.* **25** 753

[19] Lian S G, Chen G R, Cheung A and Wang Z Q 2004 *IEEE Symposium on Neural Networks* **3721** 627

[20] Chen G R, Mao Y B and Chui C K 2004 *Chaos Soliton. Fract.* **21** 749

[21] Karras D A and Zorkadis V 2003 *Neural Netw.* **16** 899

[22] Ren X X, Liao X F and Xiong Y H 2011 *J. Comput. Appl.* **31** 1528 (in Chinese)

[23] Li Y and Pan X J 2010 *Sci. Tech. Engng.* **10** 7310 (in Chinese)

[24] Chan C K and Cheng L M 2001 *IEEE Trans. Neural Netw.* **12** 340

[25] Yen J C and Guo J I 1999 *IEEE Workshop on Signal Processing Systems. SiPS 99. Design and Implementation*, August 1999 Nangtou Taiwan, China, p. 319

[26] Li P, Li Z, Halang W A and Chen G R 2007 *Chaos Soliton. Fract.* **32** 1867

[27] Kaneko K 1985 *Prog. Theor. Phys.* **74** 1033

[28] Rukhin A, Soto J and Nechvatal J 2001 *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications* (NIST Special Publication 800–22)