Nonlinear Dyn (2015) 79:1511–1518
DOI 10.1007/s11071-014-1757-7

ORIGINAL PAPER

Breaking a novel image encryption scheme based on Brownian motion and PWLCM chaotic system

Congxu Zhu · Siyuan Xu · Yuping Hu · Kehui Sun

Received: 19 June 2014 / Accepted: 15 October 2014 / Published online: 31 October 2014

Abstract This paper analyzes the security of a novel image encryption scheme with a permutation–diffusion structure, which is based on Brownian motion and PWLCM chaotic system. By applying chosen plaintext, we demonstrate that a hacker can determine the permutation vector and the diffusion sequence used, respectively, in permutation and diffusion procedure, which can be exploited to reveal the plain image. The effectiveness of the proposed chosen plaintext attack is supported by concise theoretical analyses and is verified by experimental results.

Keywords Image encryption · Brownian motion · PWLCM · Chosen plaintext attack

C. Zhu (✉) · S. Xu
School of Information Science and Engineering, Central South University, Changsha 410083, China
e-mail: zhucx@csu.edu.cn

S. Xu
e-mail: xusiyuan10222@126.com

Y. Hu
School of Information, Guangdong University of Finance and Economics, Guangzhou 510320, China
e-mail: okhyp@gdufe.edu.cn

K. Sun
School of Physics and Electronics, Central South University, Changsha 410083, China
e-mail: kehui@csu.edu.cn

1 Introduction

With the rapid developments of information technology and popularization of digital products require that multimedia data are transmitted over all kinds of communication networks. Therefore, secure delivery of multimedia data becomes increasingly important. A digital image is one of the most popular multimedia formats, which is widely used in the political, economic, defense, education, etc. However, traditional text encryption schemes fail to be competent for images' encryption due to the big differences between textual and digital images. To meet a great demand for secure image transmission over networks, a variety of encryption schemes utilizing all kinds of nonlinear theories, such as chaos [1–21], reversible cellular automata [5] and other theories [10,21], have been proposed.

Since there exists the subtle similarity between chaos and cryptography, it makes chaos an ideal tool to design secure and efficient encryption schemes. For image encryption, chaos-based approaches have shown superior performance [1–5]. Chaotic systems are characterized by sensitive dependence on initial conditions and control parameters, pseudorandomness and ergodicity, which meet the classic Shannon requirements of confusion and diffusion. Therefore, chaotic systems have attracted much attention for cryptology [6–15]. Unfortunately, some chaotic encryption algorithms have been found to be insecure and/or incomplete from the viewpoint of modern cryptology [16–20]. In order to enhance the security level of chaos-

based image cryptosystems, it is of vital significance to evaluate the security of chaos-based image encryption algorithm.

In Ref. [21], a novel image encryption algorithm was introduced based on Brownian motion and piecewise linear chaotic map (PWLCM). The main idea of this algorithm is introducing Brownian motion to scramble the image, and the sum of image data is used in the image permutation process. However, this is not enough to make the cryptosystem secure. The equivalent secret keys can be revealed by using chosen plaintext attacks.

The rest of this paper is organized as follows. Section 2 describes briefly the image encryption algorithm under study. Detailed cryptanalysis on the algorithm is presented in Sect. 3. Section 4 provides the examples and experimental results. The last section concludes this paper.

2 Description of the original encryption scheme

Wang and Xu's encryption scheme is composed of two processes: pixel permutation and diffusion, which will be introduced in the following subsections, respectively.

2.1 Permutation

Wang and Xu take each pixel of the image as a dynamic Brownian particle, using the Monte Carlo method to simulate a Brownian motion, thus effectively scrambling the image. In the cryptosystem, two Logistic maps are used to generate permutation sequences:

$$u_{n+1} = 3.925 \times u_n \times (1 - u_n) \tag{1}$$

$$v_{n+1} = 3.925 \times v_n \times (1 - v_n) \tag{2}$$

where $u_n, v_n \in [0, 1]$ are the states of the Logistic systems. Because the control parameter is 3.925, the map is chaotic.

Suppose the plaintext image $\mathbf{P}$ to be encrypted is of size $M \times N$. Convert the 2D image $\mathbf{P}$ into a one-dimensional sequence $[p_1, p_2, \ldots, p_{M\times N}]$ (from left to right, up to down), where p_i denotes the pixel value of the plaintext image in the row floor (i/N) column mod (i, N). To calculate the sum (s) of all pixels in the image: $s = p_1 + p_2 + \cdots + p_{M\times N}$. Given u_{01} and v_{01} in advance and such that $u_{01} > 0.1$, $v_{01} > 0.1$, then s is used to generate the initial value $(u_0, v_{0)}$ of the Logistic maps of Eqs. (1) and (2) by using the following formulas:

$$u_0 = u_{01} - \left(s/10^{14} - \left[s/10^{14}\right]\right) \Big/ 10^2 \tag{3}$$

$$v_0 = v_{01} - \left(s/10^{14} - \left[s/10^{14}\right]\right) \Big/ 10^2 \tag{4}$$

where $[\cdot]$ means the integer part of a number.

Use the initial values u_0, v_0 and the Logistic maps to generate two sequences $(u_1, u_2, \ldots, u_{M\times N})$ and $(v_1, v_2, \ldots, v_{M\times N})$. For any pixel $i \in (1, 2, 3, \ldots, M \times N)$, use the following formulas to acquire the two angles under polar coordinates:

$$\alpha_i = u_i \times \pi \tag{5}$$

$$\beta_i = v_i \times 2\pi \tag{6}$$

Then, using formula (7) to obtain the x, y. components of the current moving distance.

$$\begin{cases} x_i = r \times \sin(\alpha_i) \times \cos(\beta_i) \\ y_i = r \times \sin(\alpha_i) \times \sin(\beta_i) \end{cases} \tag{7}$$

where r denotes the step length of movement, α_i and β_i denote the direction of movement of the ith particle(pixel). In Ref. [21], $r = 2$. So, one movement of all pixels are gotten, then iterate this movement for R rounds. Last, the positions of all pixels are recorded by an array l, and the position of the ith pixel is denoted by $l(i) = (l1(i), l2(i))$; $l1(i)$ contains the location of the x component, and $l2(i)$ contains the y component.

Then, start from the first pixel p_1, establish a one-to-one map between p_i and $p_{l(i)}$ $(i = 1, 2, 3, \ldots, M\times N)$ from 1 to $M \times N$, and swap their positions one by one. For example, choose a pixel p_3 (located at $x = 1$, $y = 3$), swap the values of the pixel p_3 with $p_{l(3)}$(located at $x = l1\,(3)$, $y = l2\ (3)$), and thus the values of the pixels p_3 and $p_{l(3)}$ are both changed.

It is quite obvious that the new position of anyone pixel after movement for R rounds is determined only by u_{10}, v_{10}, s and R. Namely, there exists a permutation vector $\mathbf{T} = \{t(i),\ i = 1, 2, \ldots, M \times N\}$, such that the ith pixel in the original image corresponds to the $t(i)$-th pixel in the permuted image. It is important to note that $t(i)$ is determined only by u_{10}, v_{10}, s and R in Ref. [21]. In fact, the permutation vector $\mathbf{T} = [t(1), t(2), \ldots, t(M \times N)]$ is equivalent to u_{10}, v_{10}, s and R.

2.2 Diffusion

In the diffusion process, the authors use the piecewise linear chaotic map, which is defined as:

$$x_i = F(x_{i-1}, \eta) = \begin{cases} x_{i-1}/\eta, & 0 < x_{i-1} < \eta \\ x_{i-1}/\eta, & \eta \le x_{i-1} < 0.5 \\ F(1 - x_{i-1}, \eta), & 0.5 \le x_{i-1} < 1 \end{cases} \tag{8}$$

Diffusion operations are described by Eq. (9):

$$\begin{cases} y_i = F(y_{i-1}, \eta) \\ d_i = (y_i \times 10^{14}) \bmod 256 \\ c_i = p'_i \oplus d_i \oplus c_{i-1} \end{cases} \tag{9}$$

where $i \in \{1, 2, 3, \ldots, M \times N\}$, c_0 is a given value provided as a key, F means the PWLCM map, $y_0 > 0.1$ is the PWLCM map's initial value (given in advance), η is the parameter, p'_i is the ith pixel of the permuted image with the scanning order from left to right and up to down, and c_i is the encrypted value of p'_i. In Ref. [21], $u_0 = 0.2$, $v_0 = 0.2$, $R = 100$, $y_0 = 0.2$, $\eta = 0.3$, $c_0 = 100$.

It is important to note that d_i is determined only by y_0 and η in Ref. [21]. In fact, sequence $\mathbf{D} = [d(1), d(2), \ldots, d(M \times N)]$ is the diffusion vector, which is equivalent to y_0 and η.

3 The cryptanalysis and attack

According to Kerchoff's principle, when analyzing an encryption algorithm, an assumption is that the cryptanalyst knows exactly the design and working of the cryptosystem. Namely, cryptanalyst knows everything about the cryptosystem except for the secret keys. There are numerous techniques to implement cryptanalysis; four classic attacks are presented below, from the hardest type to the easiest:

1. *Ciphertext only attack* The opponent possesses only the ciphertext.
2. *Known plaintext attack* The opponent possesses both a piece of ciphertext and the corresponding plaintext.
3. *Chosen plaintext attack* The opponent obtains temporary access to the encryption machinery, hence he or she can encrypt any plaintext and then obtain the corresponding ciphertext, but with an unknown key.
4. *Chosen ciphertext attack* The opponent obtains temporary access to the decryption machinery, hence he or she can decrypt any ciphertext and then obtain the corresponding plaintext, but without knowing the secret keys.

The encryption algorithm is considered to be insecure if it is not able to resist the attacks mentioned above. In Wang and Xu's encryption scheme, the secret keys are (u_0, v_0, y_0, η, R, c_0). However, we find that the permutation vector $\mathbf{T}$ is exactly equivalent to the secret keys u_{10}, v_{10}, s and R, and the diffusion vector $\mathbf{D}$ is exactly equivalent to the secret keys y_0 and η. Once we obtain the values of these two vectors, the cipher image can be recovered easily. Suppose the cipher image to be recovered in 1D form is $\mathbf{C} = [c(1), c(2), \ldots, c(M \times N)]$, the corresponding plaintext image in 1D form is $\mathbf{P} = [p(1), p(2), \ldots, p(M \times N)]$, and the corresponding permuted image in 1D form is $\mathbf{P}' = [p'(1), p'(2), \ldots, p'(M \times N)]$, where M is the number of rows, and N is the number of columns. We use $\mathbf{A} = [a(1), a(2), \ldots, a(M \times N)]$ to denote the chosen plaintext image, and $\mathbf{B} = [b(1), b(2), \ldots, b(M \times N)]$ to denote the ciphertext image corresponding to $\mathbf{A}$. The procedure of recovering $\mathbf{C}$ is divided into three stages, which are described, respectively, in Sects. 3.1 to 3.3.

3.1 Recover the diffusion vector D and the sum value s

In order to recover the diffusion vector $\mathbf{D}$ and the sum value s, we need only one pair of plaintext–ciphertext as long as all the pixels in the plaintext image have the same value. Our chosen plaintext attacks are described in detail below.

Step 1. Choose a plaintext image $\mathbf{A}$ = [0,0,…,0] and obtain its corresponding ciphertext image $\mathbf{B} = [b(1), b(2), \ldots, b(M \times N)]$ by using Wang and Xu's encryption machinery. Suppose the sequences $\mathbf{D}$ are denoted as $\mathbf{D} = [d(1), d(2), \ldots, d(M \times N)]$, which is unrelated to the plaintext image. Because all the pixels in the permuted image $\mathbf{A}'$ are still zero, namely $a'(i) = 0, i = 1, 2, \ldots, M \times N$. According to Eq. (9), we obtain the following results:

$$\begin{cases} d(i) = b(i) \oplus b(i-1), & i > 1 \\ d(1) = c_0 \oplus b(1) \end{cases} \tag{10}$$

Step 2. By using Eqs. (9) and (10), we can obtain $p'(i)$ corresponding to $\mathbf{C}$ as:

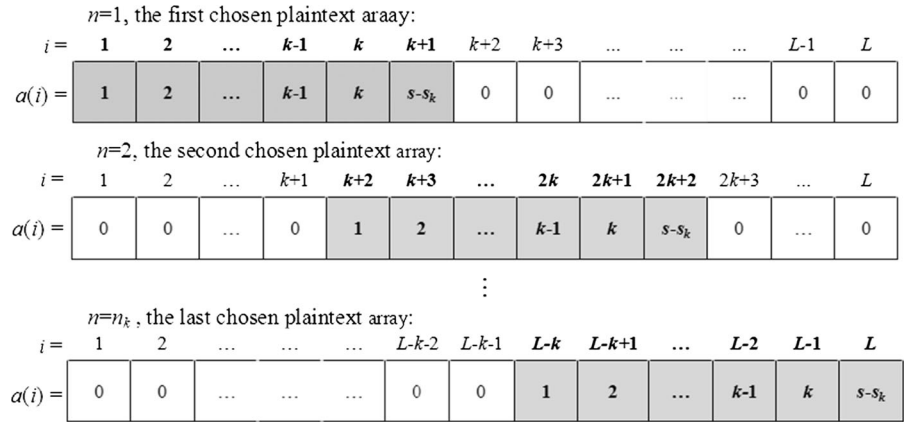

Fig. 1 The forms of the n_k chosen plaintext arrays

$$\begin{cases} p'(i) = c(i) \oplus c(i-1) \oplus d(i), & i > 1 \\ p'(1) = c(1) \oplus c_0 \oplus d(1) = c(1) \oplus b(1) \end{cases} \tag{11}$$

Step 3. Because the sum (s) of all pixels in the image **P** is equal to that of all pixels in the image **P**′, we can get the sum (s) of all pixels in the image **P** : $s = p'(1) + p'(2) + \cdots + p'(M \times N)$.

3.2 Recover the permutation vector T

We use L to denote the pixel number of image **C**, namely $L = M \times N$. To decode the permutation vector **T**, the chosen plaintext arrays must meet some conditions. There are two cases to be considered.

Case 1: If $s_L = 1 + 2 + \cdots + L \leq s$.

In this case, we chose the plaintext array **AA**, which has the following form: $\mathbf{AA} = \{1, 2, \ldots, L-2, L-1, L+s-s_L\}$, such that the sum of **AA** is s and the elements' values are different from each other. Once we get its corresponding ciphertext array $\mathbf{BB} = \{bb(1), bb(2), \ldots, bb(L)\}$ by using Wang and Xu's encryption machinery, we can obtain the transform vector **T** by using the following operations:

Step 1. Use the decoded **D** sequence in Eq. (10) to obtain the permuted array $\mathbf{AA}' = \{aa'(1), aa'(2), \ldots, aa'(L)\}$ from **BB**, and the fomula is as follows:

$$\begin{cases} aa'(i) = bb(i) \oplus bb(i-1) \oplus d(i), & i > 1 \\ aa'(1) = bb(1) \oplus d(1) \oplus c_0 = bb(1) \oplus b(1) \end{cases} \tag{12}$$

Step 2. Comparing $aa(i)$ and $aa'(j)$, $i = 1, 2, \ldots, L$; $j = 1, 2, \ldots, L$. If $aa(i) = aa'(j)$, one can obtain $t(i) : t(i) = j$. Repeat the comparing operation, until all $t(i)$ for $i \in \{1, 2, \ldots, L\}$ are obtained.

Case 2: If $s_L = 1 + 2 + \cdots + L > s$.

In this case, more than one chosen plaintext arrays are need.

Step 1. We will find a maximum integer k in the rang $[L-1, 2]$ such that $s - s_k > k$ and $\mathrm{mod}(L, k+1) = 0$, where $s_k = 1 + 2 + \cdots + k = k \times (k+1)/2$. Let $n_k = L/(k+1)$.

Step 2. We chose n_k pairs of plaintext–ciphertext arrays to decode $\mathbf{T} = [t(1), t(2), \ldots, t(L)]$. The forms of the n_k chosen plaintext arrays are shown in Fig. 1, $n = 1, 2, \ldots, n_k$.

From Fig. 1, one can see that every chosen plaintext array $[a(1), a(2), \ldots, a(L)]$ has $(k+1)$ elements whose values greater than zero and each element is different from others. The other $(L-k-1)$ elements are all zero. The sum of elements' values in each chosen plaintext array is s. For each chosen plaintext array, one can get its corresponding ciphertext array by using Wang and Xu's encryption machinery. Then, one can do the following operations:

Firstly, using the decoded **D** sequence in Eq. (10) to decode the permuted array corresponding to the chosen plaintext array, the formula is similar to Eq. (12).

Secondly, comparing each elements in the chosen plaintext array with elements in the permuted array, one can determine $(k+1)$ elements in **T** sequence. For example, if the chosen plaintext array is the first one, one can determine $t(1), t(2), \ldots, t(k+1)$; if the chosen plaintext array is the second one, one can determine $t(k+2), t(k+3), \ldots, t(2k+2)$; ...; if the chosen plaintext array is the last one, one can determine $t(L-k), t(L-k+1), \ldots, t(L)$.

After all n_k pairs of plaintext–ciphertext arrays being performed, one can obtain the whole permutation vector **T**.

3.3 Recover the plain image P

In Sect. 3.1, we obtained the whole random key sequence $\mathbf{D} = [d(1), d(2), \ldots, d(L)]$, which is only related to the secret keys y_0, η and independent of the image to be encrypted. In Sect. 3.2, we also obtained the whole permutation sequence $\mathbf{T} = [t(1), t(2), \ldots, t(L)]$, which is only related to the secret keys u_{10}, v_{10}, s and R. Therefore, we can break any other ciphertext image $\mathbf{C} = [c(1), c(2), \ldots, c(L)]$ as long as $\mathbf{C}$ has the same parameters $y_0, \eta, u_{10}, v_{10}, s$ and R. The decryption process to recover $\mathbf{P}$ from $\mathbf{C}$ is as follows:

Step 1 Let $i \leftarrow 1$.

Step 2 Do the reversed permutation operation to obtain $p(i)$ from $p'(j)$ by using the following formula:

$$p(i) = p'(t(i)) \tag{13}$$

Step 3 Let $i \leftarrow i + 1$.

Step 4 Repeat Step 2 to Step 3 until i reaches L. Then, all $p(i)$ of the plaintext are obtained.

4 Examples and experimental results

In this section, we demonstrate the process of attack with two simple examples and give some experimental results of recovered images.

4.1 Case 1

In this case, suppose the 3×3 cipher image to be recovered in 1D form is $\mathbf{C} = [207, 214, 12, 217, 79, 181, 212, 218, 136]$, $M = 3$, $N = 3$, and $L = M \times N = 9$. The aim is to recover the plaintext $\mathbf{P}$ corresponding to $\mathbf{C}$.

4.1.1 Recover the diffusion vector D and the sum value s

Step 1. We chose a plaintext array $\mathbf{A}$=[0, 0, 0, 0, 0, 0, 0, 0, 0], and if obtain its corresponding ciphertext array $\mathbf{B}$=[207, 178, 16, 216, 156, 84, 51, 50, 97} by using Wang and Xu's encryption machinery. According to Wang and Xu's scheme, all the pixels in the permuted array $\mathbf{A}'$ are still zero. According to Eq. (10), we obtain the following results:

$$[d(2), d(3), d(4), d(5), d(6), d(7), d(8), d(9)] = [125, 162, 200, 68, 200, 103, 1, 83], \tag{14a}$$

$$d(1) \oplus c_0 = 207; \tag{14b}$$

Step 2. By using Eq. (11), we can obtain $p'(i)$ corresponding to $\mathbf{C}$ as:

$$\begin{aligned}&\left[p'(1), p'(2), p'(3), p'(4), p'(5), p'(6), p'(7), p'(8), p'(9)\right] \\ &= [0, 100, 120, 29, 210, 50, 6, 15, 1]. \end{aligned} \tag{15}$$

Step 3. Because the values of elements are not changed in the permutation process, we can get the sum s of elements' values in the array $\mathbf{P}$: $s = p'(1) + p'(2) + \cdots + p'(L) = 531$.

4.1.2 Recover the permutation vector T

Because $s_L = 1 + 2 + \cdots + 9 = 9 \times (9 + 1)/2 = 45 \le s = 531$, the chosen plaintext array $\mathbf{AA}$ should have the following form: $\mathbf{AA}$=[1, 2, 3, 4, 5, 6, 7, 8, 495], such that the sum of $\mathbf{AA}$ is 531 and the elements' values are different from each other. Suppose we obtain its corresponding ciphertext array $\mathbf{BB}$=[204, 181, 17, 222, 159, 95, 57, 471, 390] by using Wang and Xu's encryption machinery. Then, by using the decoded $\mathbf{D}$ sequence in Eq. (14), we can obtain the permuted image $\mathbf{AA}' = \{aa'(1), aa'(2), \ldots, aa'(L)\}$ from $\mathbf{BB}$, and the results are as follows:

$$\mathbf{AA}' = [3, 4, 6, 7, 5, 8, 1, 495, 2] \tag{16}$$

Comparing $\mathbf{AA}$ and $\mathbf{AA}'$ for all elements: $a(1) = a'(7)$, $a(2) = a'(9)$, $a(3) = a'(1)$, $a(4) = a'(2)$, $a(5) = a'(5)$, $a(6) = a'(3)$, $a(7) = a'(4)$, $a(8) = a'(6)$, $a(9) = a'(8)$. Therefore, $\mathbf{T}$ is as follows:

$$\mathbf{T} = [7, 9, 1, 2, 5, 3, 4, 6, 8] \tag{17}$$

4.1.3 Recover the plain image P

According to Eqs. (15) and (17), do the reversed permutation operation to obtain $p(i)$ from $p'(t(i))$ by using sequence $\mathbf{T}$: $p(i) = p'(t(i))$. Therefore, $\mathbf{P}$ is recovered as follows:

$$\mathbf{P} = [6, 1, 0, 100, 210, 120, 29, 50, 15] \tag{18}$$

4.2 Case 2

In this case, suppose the 3×3 cipher image to be recovered in 1D form is $\mathbf{C} = [196, 179, 18, 219, 157, 93, 62, 58, 105]$; $M = 3$, $N = 3$, and $L = M \times N = 9$. The aim is to recover the plaintext $\mathbf{P}$ corresponding to $\mathbf{C}$.

4.2.1 Recover the diffusion vector D and the sum value s

Step 1. We chose a plaintext array **A**=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]. Suppose its corresponding ciphertext array obtained by using Wang and Xu's encryption machinery is **B**=[207, 178, 16, 216, 156, 84, 51, 50, 97]. According to Wang and Xu's scheme, all the pixels in the permuted array $\mathbf{A}'$ are still zero. According to Eq. (10), we obtain the following results:

$$[d(2), d(3), d(4), d(5), d(6), d(7), d(8), d(9)] = [125, 162, 200, 68, 200, 103, 1, 83], \quad (19a)$$

$$d(1) \oplus c_0 = 207; \quad (19b)$$

Step 2. By using Eqs. (9) and (19), we can obtain $p'(i)$ corresponding to **C** as:

$$[p'(1), p'(2), p'(3), p'(4), p'(5), p'(6), p'(7), p'(8), p'(9)] = [11, 10, 3, 1, 2, 8, 4, 5, 0]. \quad (20)$$

Step 3. Because the values of elements are not changed in the permutation process, we can get the sum s of elements' values in array $\mathbf{P}$: $s = p'(1) + p'(2) + \cdots + p'(9) = 44$.

4.2.2 Recover the permutation vector T

Because $L = M \times N = 3 \times 3 = 9$, $s_L = 1 + 2 + \cdots + 9 = 9 \times (9+1)/2 = 45 > s = 44$.

Step 1. We find the maximum integer $k = 2$ in the rang $[L-1, 2]$ such that $s_k = 1 + 2 = 3$, $s - s_k = 41 > k$, and $\mathrm{mod}(L, k+1) = 0$. $n_k = L/(k+1) = 3$. In this case, 3 chosen plaintext images are need.

Step 2. We chose the first plaintext array as $\mathbf{A} = [1, 2, s-s_2, 0, 0, 0, 0, 0, 0] = [1, 2, 41, 0, 0, 0, 0, 0, 0]$. Suppose its corresponding ciphertext array obtained by using Wang and Xu's encryption machinery is **B**=[207, 178, 17, 217, 180, 126, 25, 24, 75]. According to (19a) and (19b), by using Eq. (9), we can obtain $\mathbf{A}'$ corresponding to **B** as:

$$\mathbf{A}' = [a'(1), a'(2), a'(3), a'(4), a'(5), a'(6), a'(7), a'(8), a'(9)] = [0, 0, 1, 0, 41, 2, 0, 0, 0].$$

By comparing **A** and $\mathbf{A}'$, $a(1) = a'(3)$, $a(2) = a'(6)$, $a(3) = a'(5)$; therefore, we can determine $t(1) = 3$, $t(2) = 6$, $t(3) = 5$.

Step 3. We chose the second plaintext array as $\mathbf{A} = [0, 0, 0, 1, 2, s-s_2, 0, 0, 0] = [0, 0, 0, 1, 2, 41, 0, 0, 0]$. Suppose its corresponding ciphertext array obtained by using Wang and Xu's encryption machinery is **B** = [205, 177, 19, 219, 159, 87, 25, 24, 75]. According to (19a) and (19b), by using Eq. (9), we can obtain $\mathbf{A}'$ corresponding to **B** as:

$$\mathbf{A}' = [a'(1), a'(2), a'(3), a'(4), a'(5), a'(6), a'(7), a'(8), a'(9)] = [2, 1, 0, 0, 0, 0, 41, 0, 0].$$

By comparing **A** and $\mathbf{A}'$, $a(4) = a'(2)$, $a(5) = a'(1)$, $a(6) = a'(7)$; therefore, we can determine $t(4) = 2, t(5) = 1, t(6) = 7$.

Step 4. We chose the third plaintext array as $\mathbf{A} = [0, 0, 0, 0, 0, 0, 1, 2, s-s_2] = [0, 0, 0, 0, 0, 0, 0, 1, 2, 41]$, Suppose its corresponding ciphertext array obtained by using Wang and Xu's encryption machinery is **B** = [207, 178, 16, 241, 181, 125, 26, 25, 75]. According to (19a) and (19b), by using Eq. (9), we can obtain $\mathbf{A}'$ corresponding to **B** as:

$$\mathbf{A}' = [a'(1), a'(2), a'(3), a'(4), a'(5), a'(6), a'(7), a'(8), a'(9)] = [0, 0, 0, 41, 0, 0, 0, 2, 1].$$

By comparing **A** and $\mathbf{A}'$, $a(7) = a'(9)$, $a(8) = a'(8)$, $a(9) = a'(4)$; therefore, we can determine $t(7) = 9$, $t(8) = 8$, $t(9) = 4$.

To sum up, we obtain the whole **T** as follows:

$$\mathbf{T} = [3, 6, 5, 2, 1, 7, 9, 8, 4] \quad (21)$$

4.2.3 Recover the plain image P

According to Eqs. (20) and (21), do the reversed permutation operation to obtain **P** from $\mathbf{P}'$ by using the permutation sequence $\mathbf{T}$: $p(i) = p'(t(i))$. Therefore, **P** is recovered as follows:

$$\mathbf{P} = [3, 8, 2, 10, 11, 4, 0, 5, 1] \quad (22)$$

4.3 Results of some experiments

To verify the real performance of the above analysis, some experiments are carried out on three plain images of size 256 × 256: Lena, Cameraman, and a chosen plaintext matrix with all elements are zero **A** = [0, 0, …, 0]. The secret keys are $u_0 = 0.2$, $v_0 = 0.2$, $R = 100$, $y_0 = 0.2, \eta = 0.3$, $c_0 = 100$. The encryption results of the three images by using Wang and Xu's encryption machinery are shown in Fig. 2a–c, respectively. According to the chosen plaintext image **A** and its corresponding ciphertext image,

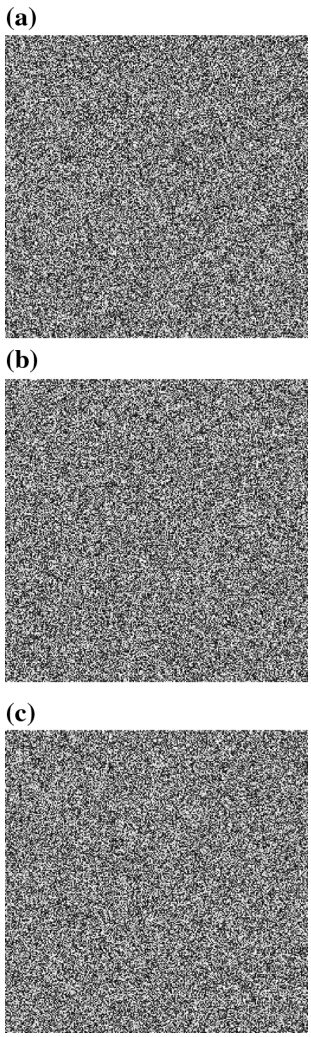

Fig. 2 Three encrypted images with the same keys. **a** The encrypted Lena, **b** the encrypted Cameraman, **c** the encrypted zeros matrix

Fig. 3 The recovered Key vector **D** and plain images. **a** Key vector, **D**, **b** recovered Lena, **c** recovered Cameraman

we obtain the random key code matrix $\mathbf{D} = [d(1) \oplus c_0,\ d(2), \ldots, d(256);\quad d(257),\ d(258), \ldots,\ d(512); \ldots; \ldots,\ d(256 \times 256)]$, which is shown in Fig. 3a. Further, the permutation vector **T** corresponding to plain images Lena and Cameraman are decoded, respectively. Then, the equivalent diffusion vector **D** and permutation vector **T** are used to decrypt the cipher images shown in Fig. 2a, b. The recovered results are shown in Fig. 3b, c, which coincide with the original plain images, respectively.

When the plaintext is of size $M \times N$, the attack complexity is $O(n \times M \times N)$, where n denotes the number of chosen plaintexts. In the worst case, the maximum of chosen plaintexts is $n_k = M \times N/(k+1)$, where k is an integer in the rang $[2, M \times N - 1]$. The time complexity of Wang's encryption algorithm is $O(M \times N)$. We tested the time complexity of our breaking method and Wang's encryption algorithm on a personal computer using MATLAB compiler. The average time to encrypt the image Lena in size 256×256 by using Wang's encryption algorithm is 3.53 s, and the average time to break the encrypted Lena by using our breaking algorithm is 86.44 s. The computer used for the test possesses two 2.60 GHz Intel Celeron processors with 2 GB RAM memory and 500 GB hard disk capacity, running on 32 bit Windows 7 and MATLAB R2013a.

5 Conclusion

In this paper, we have analyzed an image encryption scheme in detail, which is based on Brownian motion and PWLCM chaotic system. It is found that the equivalent keys can be recovered correctly with several chosen plain images, hence this encryption algorithm cannot resist chosen plaintext attacks. Both theoretical analysis and experimental results are presented to support the proposed attack. As a conclusion, the image encryption algorithm under study is not recommended in applications requiring a high level of security.

Our breaking method can be general to a certain extent for one type of encryption algorithm. If the permutation and diffusion keys of a encryption algorithm are independent of the plain images, its equivalent secret keys can be revealed by using our method.

Wang and Xu's algorithm has some excellent benefits, but two fatal flaws presented in Sect. 2 make it feasible to choose plaintext attack. One flaw is that both the permutation vector **T** and the diffusion vector **D** are unrelated to the plaintext image; another flaw is that the permutation vector **T** and the diffusion vector **D** were separated and independent. To avoid the chosen plaintext attack mentioned above, we should do one of the two things: (1) vary the permutation vector **T** or the diffusion vector **D** with different plaintext images and (2) make the permutation vector **T** and the diffusion vector **D** to establish relations.

Acknowledgments This work was supported by National Natural Science Foundation of China (Nos. 61161006, 61472451), Scientific Research Fund of Guangxi Provincial Education Department (No. 201202ZD080).

References

1. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. Int. J. Bifurc. Chaos **8**, 1259–1284 (1998)
2. Wang, X., Teng, L.: An image blocks encryption algorithm based on spatiotemporal chaos. Nonlinear Dyn. **67**, 365–371 (2011)
3. Ye, G.D., Wong, K.W.: An efficient chaotic image encryption algorithm based on a generalized Arnold map. Nonlinear Dyn. **69**, 2079–2087 (2012)
4. Tong, X.: Novel bilateral-diffusion image encryption algorithm with compound chaos and LFSR. J. Imaging Sci. J. **60**, 294–304 (2012)
5. Wang, X., Luan, D.: A novel image encryption algorithm using chaos and reversible cellular automata. Commun. Nonlinear Sci. Numer. Simul. **18**, 3075–3085 (2013)
6. Tong, X.: Design of an image encryption scheme based on a multiple chaotic map. Commun. Nonlinear Sci. Numer. Simul. **18**, 1725–1733 (2013)
7. Tong, X., Wang, Z., Zhang, M.: A new algorithm of the combination of image compression and encryption technology based on cross chaotic map. Nonlinear Dyn. **72**, 229–241 (2013)
8. Wang, X., Guo, K.: A new image alternate encryption algorithm based on chaotic map. Nonlinear Dyn **76**, 1943–1950 (2014)
9. Wang, X., Wang, Q.: A novel image encryption algorithm based on dynamic S-boxes constructed by chaos. Nonlinear Dyn. **75**, 567–576 (2014)
10. Ye, G.D.: A block image encryption algorithm based on wave transmission and chaotic systems. Nonlinear Dyn. **75**, 417–427 (2014)
11. Ye, G.D., Wong, K.-W.: An image encryption scheme based on time-delay and hyperchaotic system. Nonlinear Dyn. **71**, 259–267 (2013)
12. Fu, C., Huang, J., Wang, N.: A symmetric chaos-based image cipher with an improved bit-level permutation strategy. Entropy **16**, 770–788 (2014)
13. Fu, C., Chen, J., Zou, H.: A chaos-based digital image encryption scheme with an improved diffusion strategy. Opt. Express **20**, 2363–2378 (2012)
14. Zhu, C.: A novel image encryption scheme based on improved hyperchaotic sequences. Opt. Commun. **285**, 29–37 (2012)
15. Zhang, Y.S., Xiao, D., Shu, Y.L., Li, J.: A novel image encryption scheme based on a linear hyperbolic chaotic system of partial differential equations. Signal Process. Image Commun. **28**, 292–300 (2013)
16. Li, C.Q., Liu, Y., Xie, T.: Breaking a novel image encryption scheme based on improved hyperchaotic sequences. Nonlinear Dyn. **73**, 2083–2089 (2013)
17. Li, C.Q., Zhang, L.Y., Ou, R., Wong, K.W., Shu, S.: Breaking a novel colour image encryption algorithm based on chaos. Nonlinear Dyn. **70**, 2383–2388 (2012)
18. Li, C.Q., Lo, K.T.: Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. Signal Process. **91**, 949–954 (2011)
19. Zhu, C., Liao, C., Deng, X.: Breaking and improving an image encryption scheme based on total shuffling scheme. Nonlinear Dyn. **71**, 25–34 (2013)
20. Zhang, Y.S., Xiao, D., Wen, W., Li, M.: Breaking an image encryption algorithm based on hyper-chaotic system with only one round diffusion process. Nonlinear Dyn. **76**, 1645–1650 (2014)
21. Wang, X., Xu, D.: A novel image encryption scheme based on Brownian motion and PWLCM chaotic system. Nonlinear Dyn. **75**, 345–353 (2014)