Article

Journal of Vibration and Control
1–10

DOI: 10.1177/1077546314534717
jvc.sagepub.com

A feedback chaotic image encryption scheme based on both bit-level and pixel-level

Guodong Ye[1,2] and Xiaoling Huang[1]

Abstract

Recently, there are more and more image encryption algorithms being proposed based on chaotic systems. The classical architecture employs permutation plus diffusion. However, the traditional permutation operation can only shuffle the positions of the image pixels, but not change the pixels values and not implement the bit shifting in different bit planes. We can also see that most of them are key-dependent structures. So, they cannot resist efficiently the known-plaintext or chosen-plaintext attacks. To overcome these existing low-security problems, we suggest a new feedback image encryption scheme, in which a mathematical model is established for updating the initial conditions of the chaotic system in the permutation stage. Then, the secure SHA-3 algorithm is employed further to generate the keystream combined with new initial keys of the chaotic system in a diffusion stage. Both keystreams generated in the permutation and diffusion processes are dependent on the plain-image. Theoretical analysis and numerical simulations demonstrate the high security of the proposed encryption scheme, especially the ability to resist the known-plaintext and chosen-plaintext attacks.

Keywords

Image encryption, bit-level, pixel-level, SHA-3, cryptography

1. Introduction

In the modern diversified information era, most of us are dealing with large amounts of digital data at work or at home every day. Image information is one of the most important media, as it can give us a direct way to study clearly and understand the context. On the one hand, image transmission has been received increasing attention due to the convenience of modern networks. On the other hand, the security has been also noticed for our images. However, image encryption is different from text encryption because image context has bulk data capacity, high redundancy and strong correlation among adjacent pixels (Zhang and Xiao, 2014). Traditional methods such as the Advanced Encryption Standard (AES) or Data Encryption Standard (DES) are not suitable for image encryption. We need to search for appropriate approaches to meet the challenge of image security.

In recent years, chaos-based methods have increasingly attracted the attention of researchers (Alvarez and Li, 2006; Azzaz et al., 2013; Chen et al., 2004; El-Latifa et al., 2013; Lima et al., 2013; Ye, 2013; Zhang et al., 2013a, 2014; Zhang and Wei, 2013). The main reasons for this are the perfect characteristics in chaotic systems such as ergodicity, sensitivity to initial conditions, and pseudo-randomness. Currently, the structure of the image encryption is permutation plus diffusion. (Eslami and Bakhshandeh, 2013) aimed to promote the plaintext sensitivity in the approach (Zhang and Liu, 2011). The diffusion stage is enhanced significantly with an overall security of the algorithm after being improved (Zhang and Liu, 2011). However, we find the key-dependence only in the permutation,

[1]College of Science, Guangdong Ocean University, Zhanjiang, Guangdong, China
[2]Department of Electronic Engineering, City University of Hong Kong, Kowloon Tong, Hong Kong

Received: 25 November 2013; accepted: 9 April 2015

Corresponding author:
Xiaoling Huang, College of Science, Guangdong Ocean University, Zhanjiang 524088, Guangdong, China.
Email: xyxhuang@hotmail.com

which is fragile to the chosen-plaintext attack. Babaei (2013) suggested a logistic map as an input of the one-time pad (OTP) algorithm and proved the efficiency of the algorithm for image encryption. Unfortunately, the keystream is generated dependent only on the initial keys, then it is put into the OTP system with some XOR functions. A color image encryption scheme (Dong, 2014) using one-time keys based on coupled chaotic systems was presented. This scheme has a new technology that the keystream has been produced by both key and plaintext, so it is very sensitive to keys and plain-image. However, the hash value is computed from plain-image. The new hash value must be transmitted every time, which cannot satisfy the Shannon cipher rule.

Activated by the shortcoming of the above algorithms and other analyses of encryption methods (Sam et al., 2012; Zhang et al., 2013b), we propose a feedback chaotic image algorithm, in which the keystream is generated dependent on the plain-image in both stages of permutation and diffusion. Permutation is carried out in the bit plane which can act as another diffusion function at the same time. To be different from other bit permutation, we design a novel bit shifting at different bit planes in the permutation. Then, SHA-3 is adopted to the permuted image in the diffusion process. A pixel changed (one-bit changed in bit plane) will produce decisively a different cipher-image in our method. Fortunately, double-diffusion operations are done in our encryption structure.

The rest of this paper is organized as follows. In the next section, the concept of the 3D cat map is described and analyzed in detail. Section 3 shows us the proposed image cryptosystem with two stages of permutation and diffusion. Updating systems for initial conditions are established in this section. Then, experimental results are displayed in Section 4. Performance analyses are reported in Section 5 with security detection. Some conclusions are given in the final Section 6.

2. 3D cat map

The classical Arnold cat map (Arnold and Avez, 1968; Ye, 2011) is a two-dimensional map with two initial conditions x_0 and y_0 seen in the following equation (it is widely used in image encryption algorithms for two positive Lyapunov characteristic exponents and a simple structure):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \bmod 1 \tag{1}$$

where the notation y mod 1 denotes the fractional part of the real number y by adding or subtracting an appropriate integer, i.e. modulus after division. The map is chaotic because the Lyapunov characteristic exponents are $\lambda_1 \approx 2.6180$ and $\lambda_2 \approx 0.3820$ respectively. Then, this 2D Arnold cat map can be generalized (Chen et al., 2004; Ye, 2011) easily by introducing two control parameters (a and b) as follows

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & a \\ b & 1+ab \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \bmod 1 \tag{2}$$

where, for any $a > 0$, $b > 0$, we can also have a chaotic system of (2) due to the largest Lyapunov characteristic exponent $\lambda_1 = 1 + \frac{ab+\sqrt{a^2b^2+4ab}}{2} > 1$. The system will become stronger in the chaotic sense (Ye, 2011) if larger $a > 1$ and $b > 1$ are set.

In order to enlarge the key space of initial conditions, a natural method is to design a higher-dimensional chaotic system. Alvarez and Li (2006) pointed out that the key space of an encryption algorithm should reach at leat 10^{30} to make the brute-force attack infeasible. The 3D Arnold cat map as an extension of the 2D Arnold cat map was studied by Chen et al. (2004). The definition of a 3D Arnold cat map is described as

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ z_{i+1} \end{pmatrix} = A \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \bmod 1 \tag{3}$$

where matrix $A = [a_{ij}]_{3\times 3}$ with all elements are computed as follows and with six parameters a_x, a_y, a_z, b_x, b_y, and b_z:

$$\begin{cases} a_{11} = 1 + a_x a_z b_y \\ a_{12} = a_z \\ a_{13} = a_y + a_x a_z + a_x a_y a_z b_y \\ a_{21} = b_z + a_x b_y + a_x a_z b_y b_z \\ a_{22} = a_z b_z + 1 \\ a_{23} = a_y a_z + a_x a_y a_z b_y b_z + a_x a_z b_z + a_x a_y b_y + a_x \\ a_{31} = a_x b_x b_y + b_y \\ a_{32} = b_x \\ a_{33} = a_x a_y b_x b_y + a_x b_x + a_y b_y + 1 \end{cases} \tag{4}$$

To simplify the case, we just set all six parameters equal to one (Chen et al., 2004), i.e. the 3D Arnold map is given by

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \\ z_{i+1} \end{pmatrix} = \begin{pmatrix} 2 & 1 & 3 \\ 3 & 2 & 5 \\ 2 & 1 & 4 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \bmod 1 \tag{5}$$

Here, the three Lyapunov characteristic exponents of the system (5), i.e. $\lambda_1 \approx 7.1842$, $\lambda_2 \approx 0.2430$, $\lambda_3 \approx 0.5728$, can be easily calculated by eigenvalues. So, the 3D Arnold map is in a chaotic state. Figure 1 shows the chaotic behavior of the system (5) with initial conditions $x_0 = 0.1145$, $y_0 = 0.3022$, and $z_0 = 0.5771$ (the dotted line with a ring denotes 10^{-14} changed in x_0). In this paper, we will use this system for the proposed image encryption algorithm.

3. The proposed cryptosystem

3.1. Bit-level permutation

We know that strong correlation often exists in natural images, so the permutation operation is considered commonly to shuffle the positions of pixels to reduce the correlation coefficients. However, on the one hand, traditional methods consider only the pixel-level permutation, for example, in Eslami and Bakhshandeh (2013), the chaotic skew tent map is used in the permutation stage with key x_0 and control parameter p. Then it obtains the state values $\{x_1, x_2, \ldots, x_{M \times N}\}$ for a plain-image with size $M \times N$ after MN iterations. Sort these values and form a new set $\{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_{M\times N}\}$, then a position transform $T = \{t_1, t_2, \ldots, t_{M \times N}\}$ will be obtained if we find the position of every $\overline{x}_i$ in the old set. As a result, the permuted image can be obtained when T is used as a P-box for shuffling. Using the permutation method given by Eslami and Bakhshandeh (2013), Figure 2(b) shows the corresponding permuted image of the plain-image cameraman in Figure 2(a). Unfortunately, the histograms of Figures 2(a) and (b) are the same as shown in Figure 2(c). As the permutation operation does not change the pixel values, the histogram of the permuted image cannot be fairly uniform but is the same as the plain-image, thus it cannot resist the statistical analysis. We can also find the same weakness in other algorithms such as those of Babaei (2013), Chen et al. (2004), and Dong (2014).

On the other hand, a key-dependent structure is detected in most of the encryption algorithms. For example, in the work of Ye (2011), a key sequence of 128 bits is randomly selected with 8 groups being divided, which are used as the parameters of the 3D

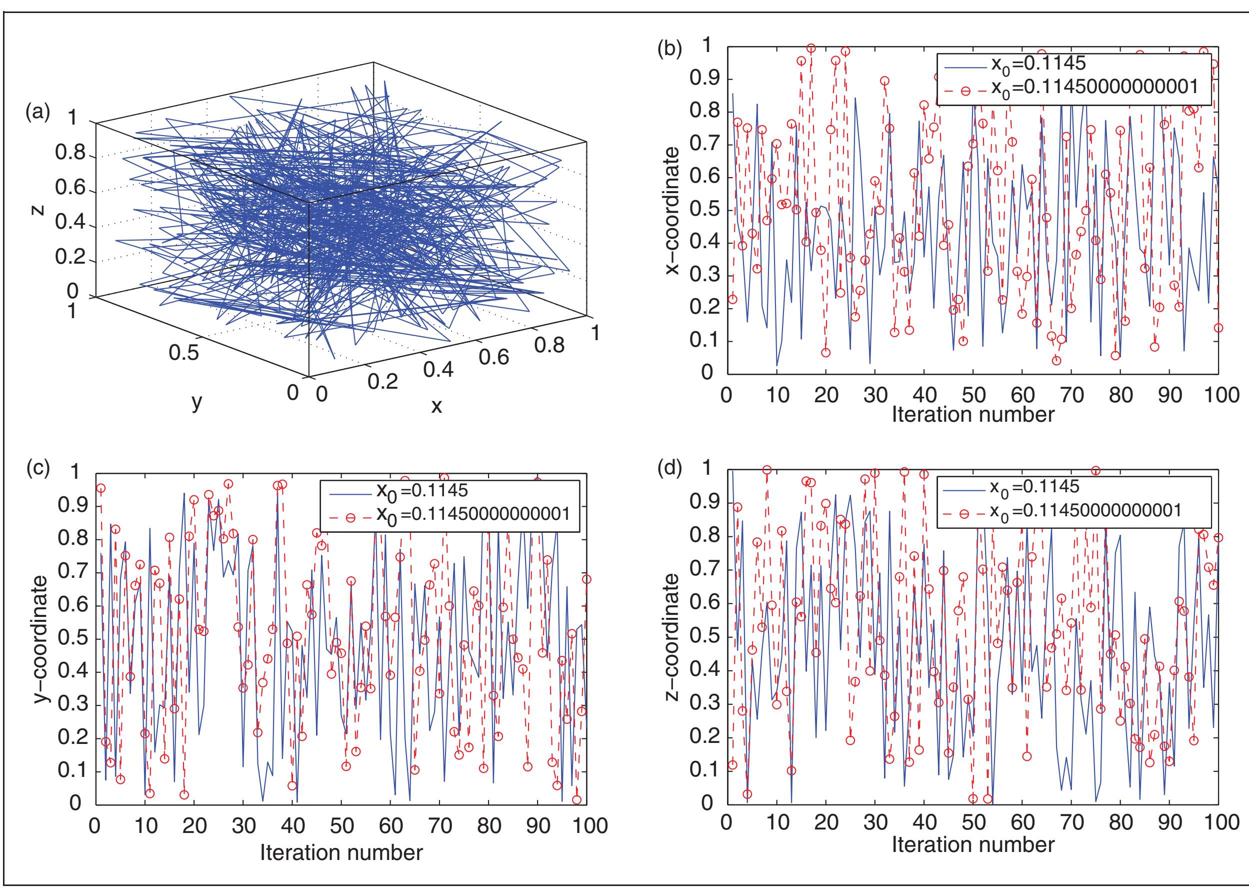

Figure 1. Chaotic behavior of the 3D Arnold map.

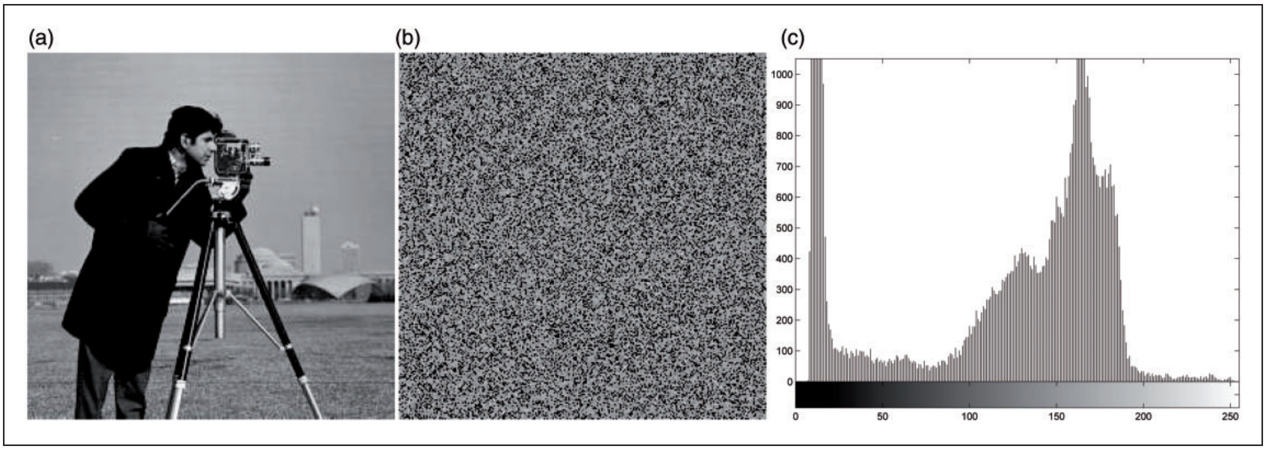

Figure 2. Cameraman image: (a) plain-image, (b) permuted image, and (c) histogram.

cat map and logistic map. Then it performs the permutation for the plain-image not considering whether the keystream is dependent on the plain-image or not. A new image encryption scheme based on intertwining chaotic maps (Sam et al., 2012) has been proposed. Three-dimensional intertwining logistic maps are adopted in the algorithm. As to the key generation, $x_{1,1}$, $y_{1,1}$, and $z_{1,1}$ are randomly chosen float numbers for initial conditions. After being iterated in the chaotic system, chaotic sequences $x_{i,j}$, $y_{i,j}$, and $z_{i,j}$ are obtained. They are then modified to become the corresponding integer values $X_{i,j}$, $Y_{i,j}$, and $Z_{i,j}$. Then the pixels are permuted by position without changing their values. Here, we can also see that there is no relation between plain-image and keystream. Many other key-dependent only on permutation operations are presented by Gao and Chen (2008), Pareek et al. (2013), and Zhang et al. (2013b).

An interesting phenomena of pixel-level permutation, observed from the bit-level point, we find that every bit in a pixel is only shuffled or permuted in the same bit-plane. We know that at the binary level every pixel value is composed of 8 bits, for example, pixel b is expressed as $b = b_7b_6b_5b_4b_3b_2b_1b_0$, here, $b_i = 0$ or 1, $i = 0, 1, \ldots, 7$. Suppose that we have an image matrix A_1 with size 3×3, after a permutation operation we get a new image matrix A_2 as follows

$$A_1 = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \longrightarrow A_2 = \begin{pmatrix} h & g & i \\ e & d & f \\ b & a & c \end{pmatrix} \tag{6}$$

If we decompose the gray value into 8-bit binary, we can easily see that the jth bit in A_1 is also the jth bit in A_2, $j = 0, 1, \ldots, 7$. The 7th bit plane of A_1 is shown in the left compared with the 7th bit plane of A_2 in the right of

$$\begin{pmatrix} a_6 & b_6 & c_6 \\ d_6 & e_6 & f_6 \\ g_6 & h_6 & i_6 \end{pmatrix} \longrightarrow \begin{pmatrix} h_6 & g_6 & i_6 \\ e_6 & d_6 & f_6 \\ b_6 & a_6 & c_6 \end{pmatrix} \tag{7}$$

Table 1. Percentage of pixel information for different bits.

Bit position	*PI*	Bit position	*PI*
1st	0.3922%	5th	6.2745%
2nd	0.7843%	6th	12.5490%
3rd	1.5686%	7th	25.0980%
4th	3.1373%	8th	50.1961%

Considering the bit's position in a pixel, different bits contain different amounts of information. Number 1 in a higher-order position is more significant than in a lower-order position. For example, 1 in 7th position represents 64 while it is only 2 in the 2nd position. The percentage of information (PI) of every bit brought for pixel p can be calculated using

$$PI(p_i) = \frac{2^i}{\Sigma_{i=0}^{7} 2^i} \tag{8}$$

Table 1 shows the PI results for different bits.

From these results, we can see the weight coefficient is different from different positions. Activated by the PI, we design a new position permutation in bit-level. Assume that the plain-image to be processed is P with size $M \times N$. First, we transform all decimal number pixels into binary pixels and get matrix $\overline{P}$

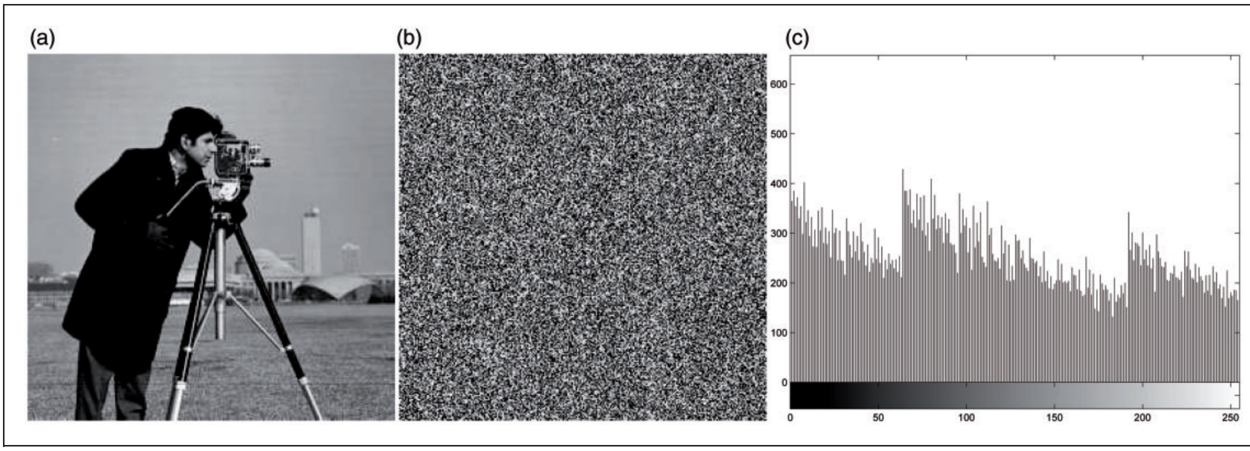

Figure 3. Permutation operation: (a) plain-image, (b) permutated image, and (c) histogram of (b).

with size $M \times 8N$. Then, a control parameter is generated as

$$\delta = \frac{\Sigma(\overline{P})}{8MN} < 1 \tag{9}$$

which is dependent on the plain-image. Given initial conditions x_0, y_0 and z_0 for system (3), we design a mathematical model of updating system for these keys as (before being iterated)

$$\begin{cases} x_0 = x_0 + \dfrac{\delta \times 10^6 - \langle \delta \times 10^6 \rangle}{\langle x_0 \times 10^{14} \rangle \bmod 256 + 1} \bmod 1 \\ y_0 = y_0 + \dfrac{\delta \times 10^6 - \langle \delta \times 10^6 \rangle}{\langle y_0 \times 10^{14} \rangle \bmod 256 + 1} \bmod 1 \\ z_0 = z_0 + \dfrac{\delta \times 10^6 - \langle \delta \times 10^6 \rangle}{\langle z_0 \times 10^{14} \rangle \bmod 256 + 1} \bmod 1 \end{cases} \tag{10}$$

where $\langle x \rangle$ represent the floor of x to the near integer. Then we can get a chaotic sequence $s = \{s_1, s_2, \ldots, s_{M+8N}\}$ after throwing away some r (for example $r = 100$) former iteration values x_i, y_i, and z_i. Collect the former M values into set H and put the left $8N$ values into set L. Here H is used for the row permutation, L is for column permutation. They should be reprocessed before being applied to permutation operation as

$$\begin{cases} H = \langle H \times 10^{14} \rangle \bmod 8N + sign \\ L = \langle L \times 10^{14} \rangle \bmod M \end{cases} \tag{11}$$

where

$$sign = \begin{cases} 1, & H_i = 0 \\ -1, & H_i \neq 0 \text{ and } \bmod (H_i, 8) = 0 \\ 0, & \text{otherwise} \end{cases}.$$

Under this environment, we can ensure that bits permutation are carried out in different bit-planes. It is a novelty in this permutation stage. Here, we continue to consider the circular shift permutation operation for $\overline{P}$, which is different from our former works and other circular operations (Ye, 2013; Ye and Wong, 2012) as we have avoided the numbers of multiple 8 (8 is the gray level). The ith row circles to the right with H_i units while the jth column circles to the bottom with L_j units, $i = 1, 2, \ldots, M$, $j = 1, 2, \ldots, 8N$. Once both row and column circular shifts are done, the permutation operation is finished. As an instance for matrix A_1 again, we transform it into binary first. Then, we can get the 7th bit plane

$$\begin{pmatrix} e_6 & g_4 & a_3 \\ i_4 & c_3 & d_4 \\ b_3 & f_6 & h_6 \end{pmatrix} \tag{12}$$

using chaotic sequence {13,10,5} for row circular and {1,2,0,1,2,0,2,2,1,0,0,2,1,1,1,0,0,0,0,1,2,1,2,1} for column circular. Finally, the exchange of different bit planes and different bit orders are done compared with matrices in (7).

Figure 3 shows the permutation results with histograms. We note here, compared with Figure 2, the proposed permutation can not only shift the position of pixels but also change the pixel values. Further, bits are shuffled among 8 bit planes randomly. The proposed method can make the known-plaintext and chosen-plaintext attacks infeasible because the key-streams are generated dependent on the plain-image.

3.2. Pixel-level diffusion

Assume that the above bit-level permuted matrix is transformed into decimal matrix denoted as Q. To fulfill the classical Shannon requirements of confusion and diffusion, a tiny change even one bit difference in the plain-image should lead to an absolutely different cipher-image. As SHA-3 is fast and very sensitive to input, it is employed in diffusion stage for the former permuted image Q to make an important effect on image pixels. The newest SHA-3 is a cryptographic hash function announced as the winner of the NIST hash function competition, which is known as Keccak (Bertoni et al., 2014) belonging to a family of sponge functions. Because of the avalanche effect, any tiny revision to the input will cause a significant change (Dong, 2014) of hash value as the output. Thus, SHA-3 is suitable to be combined in the sensitivity of initial conditions.

In the structure of SHA-3, the output of the hash value is a fixed length to any length message input. In this paper, we let the SHA-3 algorithm generate a 256-bit hash value for a plain-image message (of course, the SHA-3 function can also generate other length hash values with the sizes of 224, 384 and 512 bits (Bertoni et al., 2014)). However, we know that it will take us more time to perform the diffusion using pixel-by-pixel iteration (Eslami and Bakhshandeh, 2013; Mazloom and Eftekhari-Moghadam, 2009; Tong, 2012). So, we consider the block method for diffusion operations at the pixel-level. First, the permuted image Q is divided into four equal blocks (other division is allowed in practice), i.e. $Q=[Q_1, Q_2; Q_3, Q_4]$. The last block Q_4 is selected out as the message input and the output hash value is computed according to the SHA-3 algorithm. Then, we get a 256-bit hash value h which is transformed into 32 gray values forming the set hv. The former three maximum numbers hv_1, hv_2, are hv_3 are selected out. Third, the given initial keys $\overline{x}_0$, $\overline{y}_0$, and $\overline{z}_0$ are updated by

$$\begin{pmatrix} \overline{x}_0 \\ \overline{y}_0 \\ \overline{z}_0 \end{pmatrix} = \left[\begin{pmatrix} \overline{x}_0 \\ \overline{y}_0 \\ \overline{z}_0 \end{pmatrix} + \begin{pmatrix} hv_1 \\ hv_2 \\ hv_3 \end{pmatrix} \times 10^{-6} \right] \bmod 1 \qquad (13)$$

After iterating the system (3) for some rounds, a chaotic matrix K can be generated with the same size of sub-block. However, the elements of K should be transformed into integers 0–255, so we perform a preprocess for it, i.e. $K=\langle K\times 10^{14}\rangle \bmod 256$. The diffusion operation is carried out then by blocking, i.e. $C_i=Q_i+\mathrm{hv}_iK+C_{i-1} \bmod 256$, here $hv_4=0$, C_0 is a constant matrix with the same Q_i. Therefore, the cipher-image C is obtained with four sub-blocks $C=[C_1, C_2; C_3, C_4]$.

In our proposed encryption method, the keys are performed like one-time pad because the keys are dependent on both the initial keys of the chaotic system and the plain-image. They can automatically update themselves for different images and in different rounds of iteration. A small change in the initial keys or one bit change in the plain-image will affect the input of the chaotic map or the SHA-3 hash value. As a result, both changes can effectively influence the whole output of the cipher-image. Figure 4 shows the diffusion operation when updating the system for initial keys by SHA-3.

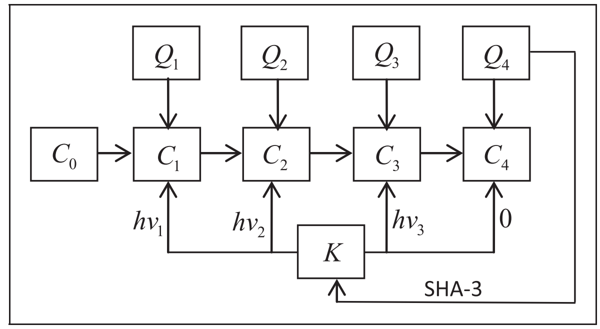

Figure 4. Block diagram of the diffusion operation.

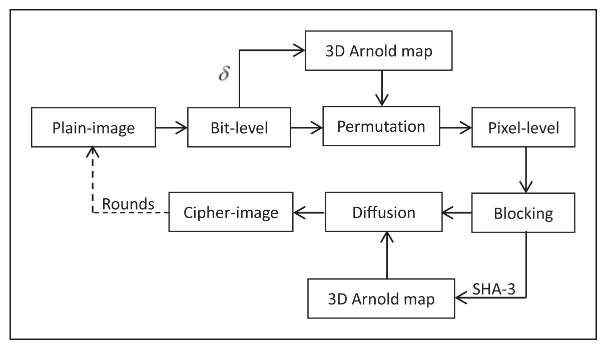

Figure 5. Block diagram of the proposed algorithm.

4. Numerical experiments

With initial conditions $x_0=0.2013$, $y_0=0.3018$, $z_0=0.8511$, $\overline{x}_0=0.5081$, $\overline{y}_0=0.4214$, and $\overline{z}_0=0.7001$ in the proposed algorithm, we implement the whole cryptosystem. All experimental tests are performed on a Windows 7 PC with an Intel® Core™ i3-2350M, 2.30 GHz CPU. Here, Matlab R2011b is used for encoding and decoding. Further, the details of the whole encryption process are listed in following seven steps. Figure 5 gives the block diagram. In the test, a Lenna image of size 256×256 as the plain-image is shown in Figure 6(a). Different from other algorithms, we just need one round of iterations which can reduce the mathematical complexity. Figure 6(b) shows the cipher-image by using our algorithm. The running time of the whole algorithm is 0.4836 seconds.

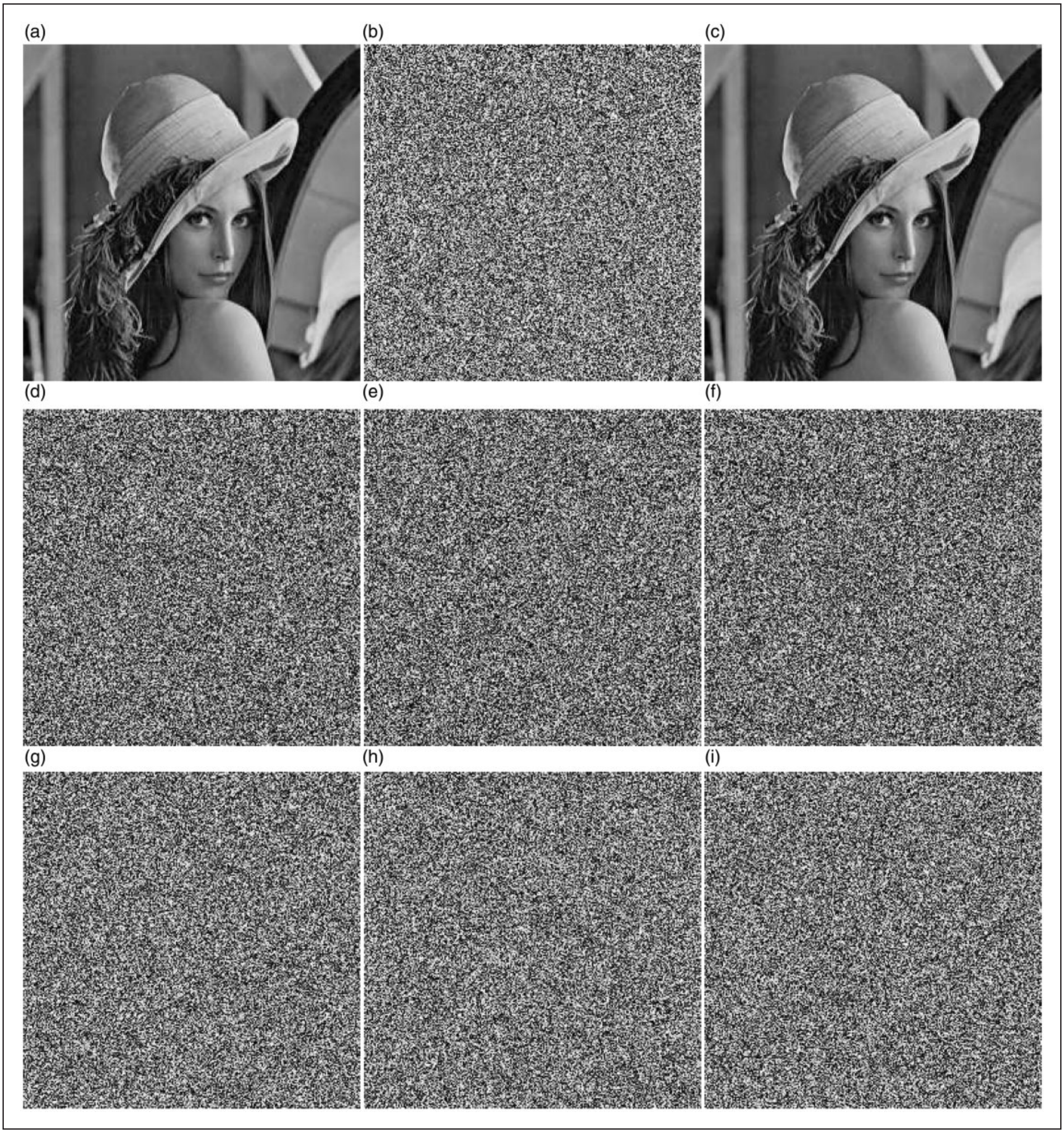

Figure 6. Test: (a) plain-image, (b) cipher-image, (c) correct decrypted image, (d) wrong decryption with 10^{-14} change in x_0, (e) wrong decryption with 10^{-14} change in y_0, (f) wrong decryption with 10^{-14} change in z_0, (g) wrong decryption with 10^{-14} change in $\overline{x}_0$, (h) wrong decryption with 10^{-14} change in $\overline{y}_0$, and (i) wrong decryption with 10^{-14} change in $\overline{z}_0$.

Step 1. Read the plain-image with matrix P and transform it into a binary format matrix $\overline{P}$.
Step 2. Calculate the sum of $\overline{P}$, and obtain δ from Equation (9).
Step 3. Update x_0, y_0, and z_0 for system (3) using (10).
Step 4. Perform the permutation function for $\overline{P}$ after obtaining the chaotic sequence H and L from (11).
Step 5. Transform the permuted image into a decimal number matrix and obtain Q. Then, divide Q into four blocks $Q = [Q_1, Q_2; Q_3, Q_4]$.
Step 6. Update $\overline{x}_0$, $\overline{y}_0$, and $\overline{z}_0$ by equation (13) for system (3) again by hash value hv of Q_4 using SHA-3 algorithm.
Step 7. Perform the diffusion operation for Q_1, Q_2, Q_3,Q_4 together with the former three maximum

Table 2. Comparisons of key space.

Methdos	Eslami and Bakhshandeh (2013)	Dong (2014)	Pareek et al. (2013)
Key space	2^{102}	10^{79}	10^{38}

Methdos	Ye (2013)	Zhang and Xiao (2014)	Ours
Key space	10^{42}	10^{70}	10^{84}

numbers of hv and K, then we can obtain the cipher-image C.

5. Performance analyses

5.1. Key space analysis

Brute-force attack aims to analyze the algorithms just with small key space; however, we have six initial keys in the 3D Arnold maps, i.e. x_0, y_0, z_0, $\overline{x}_0$, $\overline{y}_0$, $\overline{z}_0$. Thus, the key space can reach 10^{84} if we let the precision of floating number be 10^{-14}. Table 2 shows us a comparison of key space used in the different algorithms. From this, we can see that brute-force attack is infeasible in the proposed algorithm.

5.2. Sensitivity analysis

Any change in keys or pixel in the plain-image can cause an completely different cipher-image, which is a common rule for a good encryption algorithm. In our cryptosystem, all of the keys are very sensitive to the system. Figure 6(c) shows the correct decrypted image using the correct keys. However, we can not recover the plain-image if there is a tiny changed in the keys. For example, Figures 6(d), (e), (f), (g), (h), and (i) are the incorrectly decrypted images corresponding to 10^{-14} shifted in x_0, y_0, z_0, $\overline{x}_0$, $\overline{y}_0$, and $\overline{z}_0$, respectively. Therefore, sensitivity for initial keys is ensured in our encryption algorithm.

5.3. Chosen-plaintext and known-plaintext attacks

Chosen-plaintext and known-plaintext attacks are commonly seen as important methods for cipher text analysis (Arroyo et al., 2013; Li and Lo, 2011). The main reason is that these methods can be easily applied to the algorithms belonging to key-dependence only. In our algorithm, keystreams are generated dependent on the sum of the plain-image in the permutation stage, while the hash value is computed by SHA-3 for the last block in the diffusion stage. Both keystreams are related to the plain-image. So, a feedback from the plain-image and permuted-image is employed to make the keystream different in different rounds. Overviewing all of these, we conclude that the proposed algorithm can resist effectively the known-plaintext attack and chosen-plaintext attack.

Table 3. UACI and NPCR for different images.

Directions	Lenna	Baboon	Boat	Barb
UACI	33.5348	33.4649	33.3780	33.4791
NPCR	99.5804	99.5914	99.6166	99.6265

Table 4. Comparisons of UACI and NPCR with Ye (2011) using a Lenna image.

Positions	(164,179)	(242,25)	(163,255)	(122,24)
UACI of Ye (2011)	33.6844	33.2842	26.9481	36.9054
NPCR of Ye (2011)	98.4406	98.4787	98.8052	99.3698
UACI of our method	33.5331	33.5333	33.3357	33.5332
NPCR of our method	99.5803	99.5804	99.5499	99.5804

Table 5. Correlation coefficient analysis.

Directions	Vertically	Horizontally	Diagonally
Plain-image	0.9752	0.9749	0.9059
Cipher-image	−0.0018	−0.0221	0.0028

5.4. Differential attacks analysis

The following formulas are usually taken to test the influence of a pixel changed (one bit changed in the bit-plane) in the plain-image to the cipher-image:

$$NPCR = \frac{\sum_{ij} D(i,j)}{M \times N} \times 100\% \tag{14}$$

$$UACI = \frac{1}{M \times N}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\% \tag{15}$$

where $D(i, j) = 0$ if $C_1(i, j) = C_2(i, j)$; otherwise, $D(i,j) = 1$. These equations refer to the number of pixels change rate (NPCR) and the unified averaged changing intensity (UACI) respectively (Hussain et al., 2014). Table 3 shows us the results for different images. The values are near the ideal values (Sam et al.,

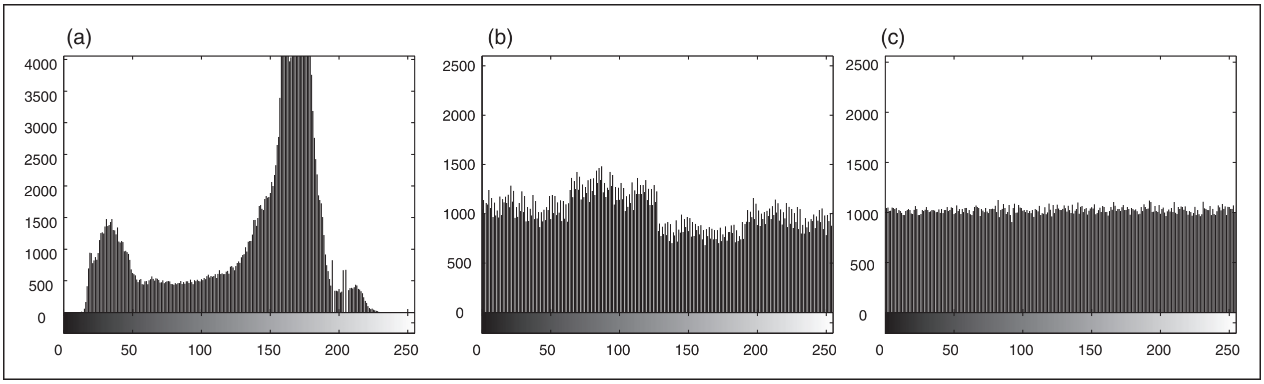

Figure 7. Histogram of a boat: (a) plain-image, (b) permuted image, and (c) cipher-image.

2012; Zhang and Xiao, 2014) as we can see. Further, a comparison with the method of Ye (2011) is given in Table 4. In addition, a good encryption algorithm should have the ability to reducing the strong correlation existing in the plain-image. Randomly, we choose some pairs of two adjacent pixels in the vertical direction, horizontal direction and diagonal direction. The following equations are used to calculate the correlation coefficient with results listed in Table 5:

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

where $\mathrm{cov}(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))$, $D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2$, $E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i$, and x_i and y_i represent the gray values of two adjacent pixels. Here, the Lenna image is taken to perform the test again.

5.5. Histogram analysis

An image can show us its properties by the distribution of pixel values, i.e. a histogram. If there is a uniform distribution of pixels, then it would make illegal statistics analysis infeasible. By using our algorithm, Figure 7 shows the histograms of the plain-image, permuted image and cipher-image, respectively. Thus, the statistics analysis attack cannot be achieved in our proposed algorithm due to the uniform distribution of pixels.

6. Conclusions

In this paper, a novel feedback chaotic image encryption algorithm has been proposed, which uses both bit-level permutation and pixel-level block diffusion. The novelty is that permutation is implemented in an external bit-plane and internal bit-plane at the same time (see function (11)). The keystream is generated dependent on the binary plain-image. Further, in the diffusion stage, a block method is adopted with the keystream produced by SHA-3 of the last permuted block. Both stages use the feedback technology to resist the chosen-plaintext and known-plaintext attacks. Different from traditional methods, we perform the permutation plus diffusion process with only one round. Fortunately, double diffusion effects are carried out due to the bit permutation in different bit planes. Finally, the experimental results and security analyses have proved the satisfactory security performance for the proposed algorithm.

Funding

The work described in this paper was fully supported by the National Natural Science Foundation of China (grant number 11301091), the Project of Enhancing School With Innovation of Guangdong Ocean University of China (grant number Q14217) and the Science and Technology Planning Project of Zhanjiang City of China (grant numbers 2015B01051 and 2015B01098).

References

Alvarez G and Li SJ (2006) Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos* 16: 2151–2199.

Arnold VI and Avez A (1968) *Ergodic problems of classical mechanics*. New York: Benjamin.

Arroyo D, Diaz J and Rodriguez FB (2013) Cryptanalysis of a one round chaos-based substitution permutation network. *Signal Processing* 93: 1358–1364.

Azzaz MS, Tanougast C, Sadoudi S and Dandache A (2013) Robust chaotic key stream generator for real-time images encryption. *Journal of Real-Time Image Processing* 8: 297–306.

Babaei M (2013) A novel text and image encryption method based on chaos theory and DNA computing. *Natural Computing* 12: 101–107.

Bertoni G, Daemen J, Peeters M and Van Assche G (2014) The Keccak sponge function family. Available at: http://keccak.noekeon.org.

Chen GR, Mao YB and Chui CK (2004) A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons and Fractals* 21: 749–761.

Dong CE (2014) Color image encryption using one-time keys and coupled chaotic systems. *Signal Processing: Image Communication* 29: 628–640.

El-Latifa AAA, Li L, Wang N, Han Q and Niu XM (2013) A new approach to chaotic image encryption based on quantum chaotic system, exploiting color spaces. *Signal Processing* 93: 2986–3000.

Eslami Z and Bakhshandeh A (2013) An improvement over an image encryption method based on total shuffling. *Optics Communications* 286: 51–55.

Gao TG and Chen ZQ (2008) A new image encryption algorithm based on hyper-chaos. *Physics Letters A* 372: 394–400.

Hussain I, Shah T and Gondal MA (2014) Image encryption algorithm based on total shuffling scheme and chaotic S-box transformation. *Journal of Vibration and Control* 20: 2133–2136.

Li CQ and Lo KT (2011) Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. *Signal Processing* 91: 949–954.

Lima JB, Lima EAO and Madeiro F (2013) Image encryption based on the finite field cosine transform. *Signal Processing-Image Communication* 28: 1537–1547.

Mazloom S and Eftekhari-Moghadam AM (2009) Color image encryption based on doupled nonlinear chaotic map. *Chaos, Solitons and Fractals* 42: 1745–1754.

Pareek NK, Patidar V and Sud KK (2013) Diffusion-substitution based gray image encryption scheme. *Digital Signal Processing* 23: 894–901.

Sam IS, Devaraj P and Bhuvaneswaran RS (2012) An intertwining chaotic maps based image encryption scheme. *Nonlinear Dynamics* 69: 1995–2007.

Tong XJ (2012) The novel bilateral-Diffusion image encryption algorithm with dynamical compound chaos. *The Journal of Systems and Software* 85: 850–858.

Ye GD (2013) Chaotic image encryption algorithm using multi-generalized logistic maps. *Journal of Computational and Theoretical Nanoscience* 10: 2789–2795.

Ye GD and Wong KW (2012) An efficient chaotic image encryption algorithm based on a generalized Arnold map. *Nonlinear Dynamics* 69: 2079–2087.

Ye RS (2011) A novel chaos-based image encryption scheme with an efficient permutation–diffusion mechanism. *Optics Communications* 284: 5290–5298.

Zhang GJ and Liu Q (2011) A novel image encryption method based on total shuffling scheme. *Optics Communications* 284: 2775–2780.

Zhang Q, Guo L and Wei XP (2013a) A novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Optik* 124: 3596–3600.

Zhang Q, Liu LL and Wei XP (2014) Improved algorithm for image encryption based on DNA encoding and multi-chaotic maps. *AEUE–International Journal of Electronics and Communications* 68: 186–192.

Zhang Q and Wei XP (2013) RGB color image encryption method based on Lorenz chaotic system and DNA computation. *IETE Technical Review* 30(2013): 404–409.

Zhang W, Wong KW, Yu H and Zhu ZL (2013b) A symmetric color image encryption algorithm using the intrinsic features of bit distributions. *Communications in Nonlinear Science and Numerical Simulation* 18: 584–600.

Zhang YS and Xiao D (2014) An image encryption scheme based on rotation matrix bit-level permutation and block diffusion. *Communications in Nonlinear Science and Numerical Simulation* 19: 74–82.