A Symmetric Plaintext-Related Color Image Encryption System Based on Bit Permutation

Shuting Cai 1,*, Linqing Huang 1, Xuesong Chen 2 and Xiaoming Xiong 1

1 School of Automation, Guangdong University of Technology, Guangzhou 510006, China; 2111604026@mail2.gdut.edu.cn (L.H.); sxmxiong@gdut.edu.cn (X.X.)
2 School of Applied Mathematics, Guangdong University of Technology, Guangzhou 510006, China; chenxs@gdut.edu.cn
* Correspondence: shutingcai@gdut.edu.cn; Tel.: +86-20-3932-2556

Received: 23 February 2018; Accepted: 11 April 2018; Published: 13 April 2018

Abstract: Recently, a variety of chaos-based image encryption algorithms adopting the traditional permutation-diffusion structure have been suggested. Most of these algorithms cannot resist the powerful chosen-plaintext attack and chosen-ciphertext attack efficiently for less sensitivity to plain-image. This paper presents a symmetric color image encryption system based on plaintext-related random access bit-permutation mechanism (PRRABPM). In the proposed scheme, a new random access bit-permutation mechanism is used to shuffle 3D bit matrix transformed from an original color image, making the RGB components of the color image interact with each other. Furthermore, the key streams used in random access bit-permutation mechanism operation are extremely dependent on plain image in an ingenious way. Therefore, the encryption system is sensitive to tiny differences in key and original images, which means that it can efficiently resist chosen-plaintext attack and chosen-ciphertext attack. In the diffusion stage, the previous encrypted pixel is used to encrypt the current pixel. The simulation results show that even though the permutation-diffusion operation in our encryption scheme is performed only one time, the proposed algorithm has favorable security performance. Considering real-time applications, the encryption speed can be further improved.

Keywords: bit-level permutation; image encryption; PRRABPM; plaintext related

1. Introduction

With the dramatic development of Internet technology, a great deal of sensitive information conveyed by digital images has been transmitted over public networks. The security problems of image transmission have become increasingly serious, especially for those related to confidential, medical, military, or commercial affairs. However, since digital images have some inherent characteristics (e.g., high redundancy, large data capacity, and strong correlation between adjacent pixels), the traditional block ciphers like Data Encryption Standard (DES), International Data Encryption Algorithm (IDEA), Advanced Encryption Standard (AES), RSA (Rivest–Shamir–Adleman), etc. do not have high performance. In recent years, chaotic maps have been used in image encryption, which have benefited from their excellent properties, such as strong ergodicity as well as sensitivity to initial conditions and control parameters. As early as 1989, Matthew suggested the use of logistic maps to generate pseudo-random numbers, which can be used to encrypt messages [1]. In 1998, a new symmetric block encryption scheme proposed by Fridrich [2] drew a great deal of attention. The architecture is similar to the one Shannon introduced in [3], which includes a pixel-level permutation-diffusion structure. In the permutation stage, the pixel position is scrambled to disturb the strong correlation between two adjacent pixels of the original image, but the pixel’s statistical...
property is not changed. Later, in the diffusion stage, the pixel values are modified to achieve a uniform distribution of pixel values [4–9]. For instance, Gao et al. used a total shuffling matrix to change the image pixel positions in the permutation stage, and then a hyper-chaotic system is used to modify the pixel values of the shuffled-image to obtain the cipher-image [4]. In [5], four values obtained from the logistic map are used to disorder four equal sub-images divided from the plain-image, and then a total shuffling matrix is used to shuffle the position of the pixels in the whole plain-image. Finally, the four sub-images are diffused simultaneously in parallel. More recently, chaotic systems have been used to encrypt images in specific fields. For example, Abundiz-Pérez et al. proposed a high-security and fast fingerprint image encryption scheme based on hyperchaotic Rössler maps [6]. In [8], a novel symmetric encryption algorithm based on confusion-diffusion architecture is provided and used to encrypt clinical information. All simulation results of the encryption system show its effectiveness, security, and robustness. Compared with gray-level images, color images can provide more information, so color image encryption attracts increasing attention. In recent years, plenty of chaos-based color image encryption algorithms have been proposed [10–17]. In order to make the RGB components of a color image affect each other and obtain high security, Wang et al. [10] used a chaotic system to encrypt these three components at the same time. In [12], Wang et al. transformed the R, G, and B components of a color plain-image into a matrix. When the matrix is passed through a permutation operation using zigzag path scrambling and a substitution process, the ciphered color image is obtained. Later, in 2017, Huang et al. [15] used Logistic map to diffuse the color image, then the RGB components are scrambled by Logistic mapping. Secondly, double random-phase encoding is used to encrypt the three scrambled sub-images into one encrypted image.

A bit-level permutation (BLP)-based cryptosystem has been proposed as a new image encryption algorithm [18–26]. BLP considers images as 3D bit matrices (width, height, and bit-length). So, the basic operation unit in the permutation stage is performed on bits rather than pixels. As the bits in different bit-planes of an image contribute different effects to visualization, Xiang et al. [18] proposed an image encryption scheme in which only the higher four bits of each pixel are encrypted and the lower four bits are unchanged. Compared with pixel-level permutation, bit-level permutation not only changes the position of the pixel, but also modifies its value. Although several rounds of 2D scrambling on each bit-plane of a plain-image are performed in some bit-level-based image encryption algorithms, the statistical property of each scrambled bit-plane are not changed. However, by combining these scrambled bit-planes to produce encrypted pictures, the statistical property of pixels in the encrypted image will be changed. For example, Zhu et al. permuted the higher four bit-planes independently and permuted the four lower bit-planes together with the Arnold cat map in [19]. Due to the problem that the permutation using 2D chaotic maps has a repeated pattern and there are strong correlations among the adjacent bit-planes (especially between higher bit-planes like the seventh and the eighth bit-planes) [21], the BLP algorithm should allow one bit in any plane to be moved to any other position in any plane. Recently, various schemes with improved properties have been proposed [22–26]. In [23], a symmetric chaos-based image cipher with a spatial bit-level permutation strategy is proposed. Compared with the recently proposed bit-level permutation methods, the confusion and diffusion effect of this new method is superior, as the bits are shuffled among different bit-planes rather than within the same bit-plane. Zhang et al. [26] proposed a new 3D bit matrix permutation mechanism which can access the bits of the plain-image randomly rather than in an orderly fashion. Furthermore, for color image encryption, bit-level permutation-based encryption algorithms have the advantages that they can achieve the interaction between RGB components in the scrambling phase, which can improve the security of encryption.

However, for most chaotic-based image encryption schemes, the relationship between permutation stage, diffusion stage, and the plaintext image is independent. Such algorithms have the following security flaws: (1) the architecture is insensitive to the original image; (2) the statistical property of the original image can be observed once the diffusion key or diffusion sequence is cracked; (3) the algorithm cannot resist chosen-plaintext and chosen-ciphertext attack. As shown in Table 1, most
of the permutation-diffusion structure-based cryptosystems [27–33] are attacked by chosen-plaintext attack and chosen-ciphertext attack.

Table 1. Some approaches to the cryptanalysis of permutation-diffusion structure-based image ciphers.

| Schemes                  | Cryptanalyzed by          | Attacks Employed               |
|--------------------------|---------------------------|--------------------------------|
| Gao et al. (2008) [4]    | Rhouma et al. (2014) [27] | chosen plaintext and ciphertext |
| Mirzaei et al. (2012) [5]| Wang et al. (2013) [28]   | chosen-plaintext               |
| Parvin et al. (2016) [7] | Norouzi et al. (2016) [29]| chosen-plaintext               |
| Wang et al. (2012) [10]  | Li et al. (2012) [30]     | chosen-plaintext               |
| Pak et al. (2017) [17]   | Wang et al. (2018) [31]   | chosen-plaintext               |
| Zhu et al. (2011) [19]   | Zhang et al. (2014) [32]  | chosen plaintext and ciphertext |
| Zhang et al. (2016) [26] | Wu et al. (2018) [33]     | chosen-plaintext               |

More recently, in order to resist the powerful chosen-plaintext and chosen-ciphertext attacks, a plaintext related image encryption scheme was proposed [12,23,34–41]. For some algorithms, the previous encrypted pixel is used to encrypt the current pixel, and after several rounds of processing in the diffusion stage of some algorithms, so the information of one pixel in the plain-image can be spread into the entire cipher image [23,35,36]. In some other image encryption systems [12,23,35–39], the key streams for encryption are related to the plain images. For instance, in [38], the initial state conditions of chaotic maps are extremely dependent on plain image, so the generated key streams are highly sensitive to the original pictures to resist known/chosen plaintext attacks. In [39], Liu et al. presented a fast image encryption algorithm. In the scheme, the iteration values of 2D-SIMM are influenced by the encrypted pixel value, and the step size of cyclic shift and the secret key for substitution will be different with different images. Therefore, the designed algorithm can resist known-plaintext and chosen-plaintext attacks. A chaos-based color image encryption algorithm was proposed in [41], in which the color image is converted into three bit-level images and combined to one bit-level image. Then, only permutation operation is performed to encrypt the integrated bit-level image to reduce the execution time. Some of the plaintext-related algorithms mentioned above present low space keys, high encryption time, or insufficient security to resist powerful known/chosen plaintext attack. For instance, the encryption algorithm presented in [34] was cryptanalyzed and broken with chosen-plaintext attack in [42].

Based on the analysis above, this paper presents a new symmetric color image encryption system based on plaintext-related random access bit-permutation mechanism (PRRABPM). Our encryption system has the following features:

(1) For color image encryption, bit-level permutation-based encryption algorithms have the advantages that they can achieve the interaction between RGB components in the scrambling phase, which can improve the security of encryption. So, this paper proposes a new random access bit-permutation mechanism in the permutation stage, which can obtain a good permutation effect and mask the statistical properties of the original image even though the diffusion key or diffusion sequence is cracked.

(2) In order to obtain high plain sensitivity and key sensitivity, the key streams used in random access bit-permutation mechanism operation are extremely dependent on plain image in an ingenious way. Therefore, the encryption system is sensitive to tiny differences in key and original images, which means that it can efficiently resist chosen-plaintext and chosen-ciphertext attacks.

(3) Not only color images but also gray images of any size can be encrypted by our encryption scheme.

(4) For the excellent performance of PRRABPM used in the permutation stage, the permutation-diffusion operation in our encryption scheme is performed only once.

The structure of this paper is as follows. Section 2 briefly reviews the chaotic maps used in this dissertation: tent map, Chebyshev map, and piecewise linear map. Section 3 proposes a
plaintext-related random access bit permutation mechanism (PRRABPM). In Section 4, we evaluate the performance of the new algorithm and show the results of simulation and analysis. The last section gives a conclusion.

2. The Involved Chaotic Systems

One-dimensional chaotic system which has the advantages of simple structure and easy realization is an ideal choice for fast encryption of large-capacity data. In this section, three 1D chaotic maps for our new chaotic encryption scheme are briefly discussed: tent map, Chebyshev map, and piecewise linear map.

2.1. Tent Map

A chaotic tent map (CTM) is a piecewise linear map which can be defined as:

\[
x_{n+1} = F_1(x_n, u) = \begin{cases} 
ux_n/2, & x_n < 0.5 \\
(1 - x_n)/2, & x_n \geq 0.5
\end{cases},
\]

where \( u \in (0, 4) \) and \( x_n \in (0, 1) \) is the output chaotic sequence.

The chaotic sequence generated by the chaotic map is used for confusing and diffusing the pixels or bits of the original image. To a large extent, the uniform level of the output chaotic sequence determines the security of the encryption system. Bifurcation analysis and Lyapunov exponent analysis are often used to measure the chaotic property of the chaotic system. As shown in Figure 1a, the tent map has a chaotic behavior when parameter \( u \in (2, 4) \).

2.2. Chebyshev Map

The Chebyshev map has similar chaotic behavior to the tent map. The expression of Chebyshev maps is shown as:

\[
x_{n+1} = F_2(x_n, a) = \cos(a \times \arccos x_n),
\]

where \( x_n \in [-1, 1] \) is the output chaotic sequence and the \( a \in \mathbb{N} \) parameter and \( x_0 \) is the initial value of the sequence which can be viewed as the secret key in our proposed algorithm. When \( a \geq 2 \), the bifurcation behavior of the Chebyshev system enters chaotic state and the Lyapunov exponent of Chebyshev maps is positive as shown in Figure 1b.

2.3. Piecewise Linear Map

The piecewise linear chaotic map (PWLCM) is a famous 1D chaotic map composed of multiple linear segments. The PWLCM is defined by the following equation:

\[
x_{n+1} = F_3(x_n, p) = \begin{cases} 
x_n/p, & 0 < x_n < p \\
(x_n - p)/(0.5 - p), & p < x_n < 0.5 \\
F(1 - x_n, p), & 0.5 < x_n < 1
\end{cases},
\]

where \( x_n \in (0, 1) \) is the output chaotic sequence and \( p \) is the control parameter satisfying \( p \in (0, 0.5) \).

The parameter \( p \) can serve as a key, as the PWLCM system is chaotic and has few periodic windows in its bifurcation diagram in the whole range of the parameter [43]. For a uniform invariant distribution and excellent ergodicity, the PWLCM chaotic map is employed in the proposed algorithm with a given initial value \( x_0 \) and control parameter \( p \). The bifurcation diagram and Lyapunov Exponent diagram of PWLCM are shown in Figure 1c.
3. New Image Encryption Algorithm

In this section, we propose a new image encryption algorithm. A block diagram of the proposed image encryption system is shown in Figure 2. Color images with RGB components can be viewed as a 3D matrix with size $M \times N \times 3$. Each component in RGB components can be represented by an 8-bit binary. The value of the pixel at coordinate $(x, y, z)$ is denoted as

$$P(x, y, z) = \{ R_1 R_2 \cdots R_8 G_1 G_2 \cdots G_8 B_1 B_2 \cdots B_8 \}_{(x,y,z)}.$$  \hspace{1cm} (4)
As shown in Figure 3, color images can be transformed into a 3D bit matrix (width, height, and bit-length). The binary bit-planes $R_p^i$, $G_p^i$, and $B_p^i (i = 1, 2, \ldots, 8)$ were transformed from the RGB components of the original color image. Since the bits in the higher bit-planes of the component contribute more effect to visualization, the lowest bit-plane of the RGB component of the 3D matrix $R_p^1$, $G_p^1$, $B_p^1$ is picked up to permute independently. The remaining bit-planes of each RGB component are used to form a new 3D matrix with size $M \times N \times 21$, denoted as $P_{21\text{bit}}$. The new formed matrix is permuted with PRRABPM.

![Figure 2. The proposed cryptosystem.](image)

**Figure 2.** The proposed cryptosystem.

**Figure 3.** 3D bit matrix of color images.

### 3.1. Permutation Stage of the Encryption System Using PRRABPM

Step 1: Decompose the $M \times N$ original color image into 24 bit-planes with size $M \times N$. As illustrated in Figure 3, the 24 bit-planes are transformed into a 3D bit matrix, denoted as $P_{24\text{bit}}$. The lowest bit-plane ($R_p^1$, $G_p^1$, $B_p^1$) of RGB components are picked up from the 3D matrix $P_{24\text{bit}}$. The remaining bit-planes are used to form $P_{21\text{bit}}$, as illustrated in Figure 4a.

Step 2: The sequences $X_1, Y_1,$ and $Z_1$ used in PRRABPM are obtained in steps 2 and 3 by sorting and searching operations. Iterate the tent map in Equation (1), the Chebyshev map in Equation (2), and PWLCM in Equation (3) $(3M/2 + N_0)$, $(3N/2 + N_0)$, and $(45 + N_0)$ times, respectively. The first $N_0$ elements are discarded to avoid the harmful effects. Then, three new sequences $X_1, Y_1, Z_1$ are obtained with sizes $3M/2$, $3N/2$, and 45, given as

$$
\begin{align*}
X_1 &= \{x_{1,1} + x_{1,2} + x_{1,3} + \cdots + x_{1,3M/2}\}, \\
Y_1 &= \{y_{1,1} + y_{1,2} + y_{1,3} + \cdots + y_{1,3N/2}\}, \\
Z_1 &= \{z_{1,1} + z_{1,2} + z_{1,3} + \cdots + z_{1,45}\}.
\end{align*}
$$
The parameter of the three chaotic maps are defined as $u = 3.999998$, $a = 4$, and $p = 0.256$, respectively. Note that these three parameters are not used as security keys.

Step 3: Take the first $M$, $N$ and 21 elements of the sequences $X_1$, $Y_1$, $Z_1$ to form three new sequences $X_2$, $Y_2$, $Z_2$, given by

\[
\begin{align*}
X_2 &= \{x_{1,1} + x_{1,2} + x_{1,3} + \cdots + x_{1,M}\}, \\
Y_2 &= \{y_{1,1} + y_{1,2} + y_{1,3} + \cdots + y_{1,N}\}, \\
Z_2 &= \{z_{1,1} + z_{1,2} + z_{1,3} + \cdots + z_{1,21}\}.
\end{align*}
\]  

Then, the three sequences are sorted by ascending order to obtain the sorted sequences $X_2s$, $Y_2s$, $Z_2s$. By using sequence $X_2s$ and the corresponding original sequence $X_2$, the sequence $X1$ mapping to the width of the matrix $P_{2bit}$ can be obtained. The specific approach is that if the $i$-th element value in $X_2$ is equal to the $j$-th element value in $X_2s$, the $i$-th element value in $X1$ is $j$. Similarly, by using sequences $Y_2s$, $Y_2$ and $Z_2s$, $Z_2$, the other two sequences $Y1$, $Z1$ mapping to the height and bit-length of the matrix can be obtained, respectively, given by

\[
\begin{align*}
X1 &= \{x_1 + x_2 + x_3 + \cdots + x_M\}, \\
Y1 &= \{y_1 + y_2 + y_3 + \cdots + y_N\}, \\
Z1 &= \{z_1 + z_2 + z_3 + \cdots + z_{21}\}.
\end{align*}
\]

Step 4: Obtain another three sequences $X2$, $Y2$, and $Z2$ with the sizes $M$, $N$, and 21 used in PRRABPM in step 4 by sorting and searching operations. The generation of the sequences $X2$ and $Y2$ is extremely dependent on plain image in an ingenious way. Three sequences are mapped to the width, height, and bit-length of the permuted bit matrix $P_{2bit}$, which is illustrated in Figure 4b. The specific approach using the plaintext-related algorithm can be described as follows:

1. Calculating the sum of the elements in $R_1^p$, $G_1^p$, $B_1^p$, and $P_{24bit}$, respectively, one can get

\[
\begin{align*}
\text{sum}_1^R &= \sum_{i=1}^{M} \sum_{j=1}^{N} R_1^p(i, j), \\
\text{sum}_1^G &= \sum_{i=1}^{M} \sum_{j=1}^{N} G_1^p(i, j), \\
\text{sum}_1^B &= \sum_{i=1}^{M} \sum_{j=1}^{N} B_1^p(i, j), \\
\text{Sum} &= \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{k=1}^{24} P_{24bit}(i, j, k).
\end{align*}
\]

2. Calculating the value of the parameters of $\text{num}1$ and $\text{num}2$, respectively, one can get

\[
f_1 = \frac{10^5 \times (\text{sum}_1^R + \text{sum}_1^G + \text{sum}_1^B)}{\text{Sum}},
\]

\[
\text{num}1 = \text{floor}((f_1 - \text{floor}(f_1)) \times 10^5, \text{floor}(0.5 \times M))) + 1,
\]

\[
\text{num}2 = \text{floor}((f_1 - \text{floor}(f_1)) \times 10^5, \text{floor}(0.5 \times N))) + 1.
\]

If all the values of $\text{sum}_1^R$, $\text{sum}_1^G$, and $\text{sum}_1^B$ are zero, the values of $\text{num}1$ and $\text{num}2$ are set to $\frac{M}{2}$ and $\frac{N}{2}$, respectively.
(3) Obtain the sequence $X_3, Y_3, Z_3$ with sizes $M, N,$ and 21, given by:

$$
\begin{align*}
X_3 &= \{x_{1, num1} + x_{1, num1+1} + x_{1, num1+2} + \cdots + x_{1, num1+M-1}\}, \\
Y_3 &= \{y_{1, num1} + y_{1, num1+1} + y_{1, num1+2} + \cdots + y_{1, num1+N-1}\}, \\
Z_3 &= \{z_{1, 22} + z_{1, 23} + z_{1, 24} + \cdots + z_{1, 42}\},
\end{align*}
$$

and then, by way of Step 3, the sequences $X_2, Y_2, Z_2$ can be obtained using the sequences $X_3, Y_3, Z_3$. One can get

$$
\begin{align*}
X_2 &= \{x'_1 + x'_2 + x'_3 + \cdots + x'_M\}, \\
Y_2 &= \{y'_1 + y'_2 + y'_3 + \cdots + y'_N\}, \\
Z_2 &= \{z'_1 + z'_2 + z'_3 + \cdots + z'_21\}.
\end{align*}
$$

Step 5: Permute the three 2D bit matrices $R^{p}_{i}, G^{p}_{i},$ and $B^{p}_{i}$ independently. The permutation equations can be described as

$$
\begin{align*}
R^{p}_{i}(X1(i), Y1(j)) &= R_{i}(i, j), \\
G^{p}_{i}(X1(i), Y1(j)) &= G_{i}(i, j), \\
B^{p}_{i}(X1(i), Y1(j)) &= B_{i}(i, j),
\end{align*}
$$

where $i = 1, 2, \cdots, M; j = 1, 2, \cdots, N$ and $R^{p}_{i}, G^{p}_{i}, B^{p}_{i}$ are the permuted bit matrices.

Step 6: Permute the 3D bit matrix $P_{21bit}$ using PRRABPM, given by

$$
P_{21bit,p}(X2(i), Y2(j)Z2(k)) = P_{21bit}(X1(i), Y1(j)Z1(k)),
$$

where $i = 1, 2, \cdots, M; j = 1, 2, \cdots, N; k = 1, 2, \cdots, 21$.

According to Equation (18) and Figure 4, the sequences $X1, Y1, Z1$ are used to access the bit element in $P_{21bit}$ randomly and the sequences $X2, Y2, Z2$ are used to permute the bit positions in $P_{21bit,p}$.

![Figure 4](image-url)

**Figure 4.** Permutation with the plaintext-related random access bit-permutation mechanism (PRRABPM).

(a) The remaining bit-planes are used to form $P_{21bit}$; (b) Three sequences are mapped to the width, height, and bit-length of the permuted bit matrix $P_{21bit,p}$.

Step 7: In order to obtain good permutation performance and make the RGB components of the color image interact with each other more sufficiently, the permuted 2D bit matrices $R^{p}_{1}, G^{p}_{1},$ and $B^{p}_{1}$ are added to the permuted 3D bit matrix $P_{21bit,p}$ to form the permuted 3D bit matrix $P_{24bit,p}$ in the following way. Firstly, $K1, K2, K3$ are calculated using the value of $z_{1,43}, z_{1,44}, z_{1,45}$ in $Z_2$. One can get
\[
\begin{align*}
K_1 &= \text{floor}(z_{1,43} \times 10^9 \mod 7) + 2, \\
K_2 &= \text{floor}(z_{1,44} \times 10^9 \mod 7) + 9, \\
K_3 &= \text{floor}(z_{1,45} \times 10^9 \mod 7) + 16.
\end{align*}
\] (19)

The bit matrix \(B_i^p\) is added to the bit-plane between the \((K_1 - 1)\)-th and the \(K_1\)-th bit-plane of \(P_{21\text{bit},p}\). Similarly, the bit matrix \(R_i^p\) is added to the bit-plane between the \((K_2 - 1)\)-th and the \(K_2\)-th bit-plane, and the bit matrix \(G_i^p\) is added between the \((K_3 - 1)\)-th and the \(K_3\)-th bit-planes.

3.2. Diffusion Stage of the Encryption System

Note that the diffusion operation is performed at the pixel-level.

Step 1: Firstly, the matrix \(P_{21\text{bit},p}\) is converted into the color image \(P_p\) with size \(M \times N \times 3\), and then the image \(P_p\) can be divided into RGB components. Secondly, three gray images are transformed into 1D pixel arrays \((P_{R,p}, P_{G,p}, P_{B,p})\), respectively. One can get

\[
\begin{align*}
P_{R,p} &= \{P_{R,p1}, P_{R,p2}, P_{R,p3}, \cdots P_{R,p(M \times N)}\}, \\
P_{G,p} &= \{P_{G,p1}, P_{G,p2}, P_{G,p3}, \cdots P_{G,p(M \times N)}\}, \\
P_{B,p} &= \{P_{B,p1}, P_{B,p2}, P_{B,p3}, \cdots P_{B,p(M \times N)}\}. \\
\end{align*}
\] (20)

Step 2: Obtain the diffusion matrix \(D = \{d_1, d_2, d_3, \cdots d_{(M \times N)}\}\), given by

\[
D(k) = \text{mod}(X1(i) \times 10^{13} + Y1(j) \times 10^{13}, 256),
\] (21)

where \(i = 1, 2, \cdots, M; j = 1, 2, \cdots, N; k = i \times j\).

Step 3: Obtain the encrypted image pixel arrays \(C_R, C_G, C_B\) using the 1D pixel arrays \(P_{R,p}, P_{G,p}, P_{B,p}\) and the diffusion matrix \(D\), given by

\[
\begin{align*}
C_R(i) &= \text{mod}(P_{R,p}(i) \odot D(i) + D(i), 256) \odot C_R(i - 1), \\
C_G(i) &= \text{mod}(P_{G,p}(i) \odot D(i) + D(i), 256) \odot C_G(i - 1), \\
C_B(i) &= \text{mod}(P_{B,p}(i) \odot D(i) + D(i), 256) \odot C_B(i - 1),
\end{align*}
\] (22)

where \(i = 1, 2, \cdots, M \times N\), and symbol “\(\odot\)” represents bitwise exclusive or operator.

Step 4: Treat the encrypted image pixel arrays \(C_R, C_G, C_B\) as RGB components of a color image so that an encrypted color image with size \(M \times N \times 3\) can be obtained.

3.3. Decryption Process

The decryption procedure is the inverse process of encryption. The flowchart of the decryption process is shown in Figure 5. The diffusion equation used in decryption is given as

\[
\begin{align*}
P_{R,p}(i) &= \text{mod}(C_R(i) \odot C_R(i - 1) + 256 - D(i), 256) \odot D(i), \\
P_{G,p}(i) &= \text{mod}(C_G(i) \odot C_G(i - 1) + 256 - D(i), 256) \odot D(i), \\
P_{B,p}(i) &= \text{mod}(C_B(i) \odot C_B(i - 1) + 256 - D(i), 256) \odot D(i),
\end{align*}
\] (23)

where \(P_{R,p}, P_{G,p}, P_{B,p}\) are 1D pixel arrays.

\[
A(i, j) = A_p(X1(i), Y1(j)),
\] (24)

where \(A\) is the 2D bit matrix which is permuted independently in the encryption process and \(A_p\) is the corresponding permuted 2D bit matrix.

The permutation equation used in decryption is given as

\[
P_{21\text{bit}}(X1(i), Y1(j), Z1(k)) = P_{21\text{bit},p}(X2(i), Y2(j), Z2(k))
\] (25)
where $P_{21\text{bit}}$ is the 3D bit matrix which is permuted using PRRABPM in the encryption process and $P_{21\text{bit}}_p$ is the corresponding permuted 3D bit matrix.

**Figure 5.** Flowchart of the decryption process.

### 3.4. Simulation Results

In this section, we evaluate the performance of the proposed scheme. The initial values of the tent map, the Chebyshev map, and the piecewise linear map are chosen as $key_{x0} = 0.22521231547896$; $key_{z0} = 0.58749654123587$; and $key_{z0} = 0.98564123475621$, respectively, and $N_0 = 2000$. A baboon is used as the testing plain-image. The plain-image, the results of encryption-decryption images, and their corresponding distribution histograms are shown in Figure 6.

**Figure 6.** The histograms of plain-image, permuted image, and final encrypted images.
4. Security Analysis

4.1. Security Key Space

For a security encryption algorithm, its key space should be larger than $2^{100}$ [44]. There are four secret keys in the proposed encryption algorithm, including the initial values $(key_{x0}, key_{y0}, key_{z0})$ of the three chaotic maps and the iteration times $N_0$. For these four secret keys, $key_{x0}$ belongs to $(0, 1)$, $key_{y0}$ belongs to $(-1, 1)$, $key_{z0}$ belongs to $(0, 1)$, and $N_0$ belongs to $(1000, 2500]$. As the computational precision of double-precision numbers is taken as $10^{16}$, the key space is $key_{total} = 10^{16} \times 10^{16} \times 10^{16} \times 1500 \approx 2^{1170}$. So, the key space of the proposed cryptosystem is large enough to resist brute force attack. In Table 2, the comparison between our method and similar image encryption algorithms is given, showing that the key space size of the proposed algorithm is larger than most of the similar algorithms.

Table 2. Keyspace comparisons.

| Schemes            | Proposed Scheme | Ref. [17] | Ref. [23] | Ref. [34] | Ref. [35] | Ref. [37] | Ref. [38] |
|--------------------|-----------------|-----------|-----------|-----------|-----------|-----------|-----------|
| Key space size     | $2^{170}$       | $2^{138}$ | $2^{241}$ | $2^{104}$ | $2^{128}$ | $2^{128}$ | $2^{572}$ |

4.2. Statistical Analysis

4.2.1. Histogram Analysis

The histogram of the encrypted image is often used to measure the security of the encryption system. For a secure encryption system, the histogram of the encrypted image should be flat, which can resist statistical attacks. The histograms of the plain-images and the corresponding cipher-images are shown in Figure 6a,c. As shown in Figure 6c, the encrypted image is completely scrambled and its histogram has a good uniform distribution, so it can resist statistical attacks. Furthermore, the histogram of the permuted image shown in Figure 6b is different from the histogram of the original image to some extent, so even though the diffusion key or diffusion sequence is cracked, the statistical information of the original image can be masked.

4.2.2. Correlation Analysis

The correlation coefficient of the pixels of the plain-image is always high because the adjacent pixels in the plain-image have a high correlation which can be used by attackers. So, the correlation coefficients of the pixels should be significantly reduced after the plain-image is encrypted. According to Equation (26), we calculate the correlation coefficients of the four directions, including the vertical, horizontal, diagonal, and anti-diagonal directions.

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

where $\text{cov}(x, y) = \frac{1}{N} \sum_{i=0}^{N} (x_i - E(x))(y_i - E(y))$, $D(x) = \frac{1}{N} \sum_{i=0}^{N} (x_i - E(x))^2$, $E(x) = \frac{1}{N} \sum_{i=0}^{N} x_i$. $x, y$ are two adjacent pixel values from four directions as mentioned above, and $N$ is the number of image pixels.

The correlation coefficients of plain-images and the corresponding cipher-images are provided in Table 3, including the vertical (V), horizontal (H), diagonal (D), and anti-diagonal (A) directions. As shown in Table 3, the correlation coefficients of the plain-images are close to 1, but the correlation coefficients of the cipher-images are close to 0. This indicates that the proposed algorithm can resist a statistical attack. Detailed results compared with some related references are given in Table 4.
Table 3. Correlation coefficients of some original images and the corresponding cipher-images in R, G, and B channels.

| Image | Plain-Image | Cipher-Image |
|-------|-------------|--------------|
|       | V    | H    | D    | A    | V    | H    | D    | A    |
| Lena  | R    | 0.9753 | 0.9853 | 0.9734 | 0.9648 | 0.0028 | 0.0046 | 0.0013 | −0.0001 |
|       | G    | 0.9666 | 0.9802 | 0.9630 | 0.9536 | 0.0004 | −0.0009 | 0.0007 | 0.0008 |
|       | B    | 0.9334 | 0.9558 | 0.9264 | 0.9198 | −0.0029 | −0.0007 | −0.0050 | 0.0013 |
| Baboon| R    | 0.9235 | 0.8740 | 0.8649 | 0.8670 | 0.0015 | 0.0002 | −0.0014 | 0.0006 |
|       | G    | 0.8668 | 0.7759 | 0.7432 | 0.7494 | 0.0033 | 0.0018 | 0.0003 | −0.0003 |
|       | B    | 0.9067 | 0.8844 | 0.8544 | 0.8540 | 0.0006 | 0.0004 | 0.0008 | 0.0012 |
| Fruits| R    | 0.9936 | 0.9928 | 0.9897 | 0.9868 | −0.0009 | 0.0003 | −0.0002 | 0.0006 |
|       | G    | 0.9855 | 0.9848 | 0.9783 | 0.9694 | −0.0019 | −0.0004 | 0.0003 | 0.0010 |
|       | B    | 0.9265 | 0.9192 | 0.8809 | 0.8531 | −0.0024 | −0.0008 | −0.0043 | 0.0015 |
| Flowers| R    | 0.9718 | 0.9719 | 0.9504 | 0.9551 | −0.0043 | −0.0028 | −0.0049 | 0.0045 |
|       | G    | 0.9510 | 0.9497 | 0.9123 | 0.9218 | −0.0043 | −0.0054 | 0.0017 | 0.0054 |
|       | B    | 0.9527 | 0.9527 | 0.9178 | 0.9256 | −0.0035 | 0.0005 | −0.0062 | −0.0001 |

Table 4. Correlation coefficients of encrypted Lena image in R channel with different algorithms.

| Direction | Original Image | Proposed Scheme | Ref. [15] | Ref. [37] | Ref. [17] | Ref. [38] | Ref. [40] |
|-----------|----------------|-----------------|-----------|-----------|-----------|-----------|-----------|
| Horizontal| 0.9853         | 0.0046          | 0.0027    | 0.0012    | −0.0026   | −0.0030   | 0.0005    |
| Vertical  | 0.9753         | −0.0028         | −0.0013   | 0.0035    | −0.0038   | 0.0025    | −0.0070   |
| Diagonal  | 0.9734         | 0.0014          | 0.0039    | 0.0056    | 0.0017    | −0.0001   | 0.0006    |

In order to evaluate the correlation property of images, we randomly select 2000 pixels from the plain-image or their corresponding cipher-image. The correlation diagram among adjacent pixels at vertical, horizontal, diagonal, and anti-diagonal directions of the R channel are shown in Figure 7. As shown in Figure 7, the values of the adjacent pixels of the cipher-image are completely different.

(a) The correlation of the original image in four directions.

(b) The correlation of the corresponding encrypted image in four directions.

Figure 7. Correlation analysis of the original image and the corresponding encrypted image in R channel (The original image is Lena).
4.2.3. Key Sensitivity and Plaintext Sensitivity Analysis

A differential attack is usually used to break a cryptosystem. For a secure encryption system to effectively resist such an attack, it should be sensitive to any tiny modification in the keys or the original image. The NPCR (number of pixels change rate) and UACI (unified average changing intensity) are usually used to evaluate the sensitivity of the key and plain-image. NPCR and UACI are expressed in the following equation:

\[
\begin{align*}
\text{NPCR} &= \frac{\sum_{i=0}^{H} \sum_{j=0}^{W} D(i,j) \times 100\%}{\text{H} \times \text{W}}, \\
\text{UACI} &= \frac{1}{\text{H} \times \text{W}} \sum_{i=0}^{H} \sum_{j=0}^{W} \left| \frac{c_1(i,j) - c_2(i,j)}{255} \right| \times 100\%,
\end{align*}
\]

(27)

where \(c_1, c_2\) are encrypted images, and \(D(i,j) = \begin{cases} 0, & \text{if } c_1(i,j) = c_2(i,j), \\
1, & \text{if } c_1(i,j) \neq c_2(i,j). \end{cases}\)

In this simulation, four plain-images were used to evaluate the key sensitivity. The proposed algorithm has four secret keys \((\text{key}_{\text{x}0}, \text{key}_{\text{y}0}, \text{key}_{\text{z}0}, N_0)\). For example, the sensitivity of \(\text{key}_{\text{x}0}\) is evaluated here. First, we selected 200 key groups \(k_{\text{xy}0}(i) = (\text{key}_{\text{x}0}(i), \text{key}_{\text{y}0}(i), \text{key}_{\text{z}0}(i), N_0(i))\) from the security key space randomly and then we used every key group to encrypt the plain-images. Then, the corresponding cipher-images \(C_1(i) (i = 1, 2, 3 \cdots 200)\) could be obtained. Secondly, a slight change \(10^{-15}\) was added into the secret key \(\text{key}_{\text{xy}0}(i)\) of the key group. The values of the remaining three keys \((\text{key}_{\text{y}0}(i), \text{key}_{\text{z}0}(i), N_0(i))\) were unchanged. Then, the key group containing the modified key was used to encrypt the plain-images again to obtain corresponding cipher-images \(C_2(i) (i = 1, 2, 3 \cdots 200)\). According to Equation (27), 200 pairs of NPCR and UACI could be calculated. The average values of NPCR and UACI are shown in Table 5. It should be noted that the key sensitivity of \(\text{key}_{\text{xy}0}, \text{key}_{\text{z}0}, N_0\) was evaluated in the same way.

Table 5. The mean NPCR (number of pixels change rate) and UACI (unified average changing intensity) of some encrypted images.

| Image  | NPCR (99.6094) | UACI (33.4635) |
|--------|----------------|----------------|
|        | R   | G   | B   | R   | G   | B   |
| Lena   | \text{key}_{\text{x}0} | 99.5952 | 99.5957 | 99.5940 | 33.4657 | 33.4633 | 33.4666 |
|        | \text{key}_{\text{y}0} | 99.6092 | 99.6079 | 99.6103 | 33.4623 | 33.4599 | 33.4582 |
|        | \text{key}_{\text{z}0} | 99.5984 | 99.5883 | 99.5958 | 33.4637 | 33.4635 | 33.4665 |
|        | \text{N}_0 | 99.6088 | 99.6110 | 99.6098 | 33.4695 | 33.4635 | 33.4701 |
| Baboon | \text{key}_{\text{x}0} | 99.6077 | 99.6075 | 99.6083 | 33.4685 | 33.4635 | 33.4611 |
|        | \text{key}_{\text{y}0} | 99.6091 | 99.6087 | 99.6110 | 33.4631 | 33.4656 | 33.4721 |
|        | \text{key}_{\text{z}0} | 99.5977 | 99.5880 | 99.5965 | 33.4602 | 33.4696 | 33.4669 |
|        | \text{N}_0 | 99.6092 | 99.6112 | 99.6083 | 33.4678 | 33.4680 | 33.4637 |
| Fruits | \text{key}_{\text{x}0} | 99.6054 | 99.6065 | 99.6060 | 33.4639 | 33.4632 | 33.4634 |
|        | \text{key}_{\text{y}0} | 99.6109 | 99.6086 | 99.6108 | 33.4665 | 33.4655 | 33.4641 |
|        | \text{key}_{\text{z}0} | 99.5964 | 99.5884 | 99.5957 | 33.4629 | 33.4624 | 33.4685 |
|        | \text{N}_0 | 99.6101 | 99.6099 | 99.6093 | 33.4676 | 33.4637 | 33.4661 |
| Flowers| \text{key}_{\text{x}0} | 99.6026 | 99.6004 | 99.6018 | 33.4635 | 33.4535 | 33.4662 |
|        | \text{key}_{\text{y}0} | 99.6076 | 99.6101 | 99.6071 | 33.4628 | 33.4641 | 33.4627 |
|        | \text{key}_{\text{z}0} | 99.5977 | 99.5885 | 99.5937 | 33.4648 | 33.4696 | 33.4725 |
|        | \text{N}_0 | 99.6089 | 99.6084 | 99.6106 | 33.4687 | 33.4664 | 33.4633 |

As shown in Table 5, the NPCR and UACI values were very close to the ideal values (NPCR: 99.6094 and UACI: 33.4635), indicating that the encryption system is sensitive to any tiny modifications in keys.
Another key sensitivity test is shown in Figure 8. Firstly, a key set was chosen from the security key space denoted as $k_{ey}(1) = (0.2252123154789601, 0.5874965412358700, 0.9856412347562100, 2000)$. To evaluate the sensitivity of $key_{x0}$, a slight change $10^{-16}$ is added into the secret key $key_{x0}(1)$ while the others were unchanged. Then the key denoted as $k_{ey}(2) = (0.2252123154789601, 0.5874965412358700, 0.9856412347562100, 2000)$ could be obtained. After that, we used $k_{ey}(1)$ and $k_{ey}(2)$ to encrypt the same original image as shown in Figure 8a to obtain two corresponding cipher-images denoted as $E1$ and $E2$, shown in Figure 8b,c. The image of pixel-to-pixel difference $|E1 - E2|$ is shown in Figure 8d, and its histogram is shown in Figure 8h, which can prove that a tiny change $10^{-16}$ in the security key will result in a significant change in cipher-image. Finally, we used $k_{ey}(1)$ to decrypt the cipher images $E1$ and $E2$ individually, and the decryption results are shown in Figure 8i,j. As can be seen, only the correct key can completely reconstruct the original image. Thus, the proposed algorithm has high key sensitivity.

![Figure 8](image)

**Figure 8.** The key sensitivity test: (a,e): the original image and corresponding histogram; (b,f): the encrypted image $E1$ with the security key set $k_{ey}(1)$ and corresponding histogram; (c,g): the encrypted image $E2$ with the security key set $k_{ey}(2)$ and corresponding histogram; (d,h): the pixel-by-pixel difference $|E1 - E2|$ and corresponding histogram; (i): the decrypted image from $E1$ using the correct security key set $k_{ey}(1)$; (j): the decrypted image from $E2$ using an incorrect security key set $k_{ey}(1)$.

In order to evaluate the sensitivity to small changes in the plain-image, we selected 200 key groups $k_{ey}(i) = (key_{x0}(i), key_{y0}(i), key_{z0}(i), N_0(i)) (i = 1, 2, 3 \ldots 200)$ from the security key space and selected 200 pixels denoted as $P_{x0el}(x, y, z) (i = 1, 2, 3 \ldots 200)$ from the original color images at random location $(x, y, z)$. For instance, the cipher image $C_1$ can be obtained by using key $k_{ey}(1)$ to encrypt the plain-image. Then, we modified the value of pixel $P_{x0el}(x, y, z)$ in the plain-image slightly, as shown in Equation (28).

$$P_{x0el}(x_i, y_j, z_k) = \text{mod} (P_{x0el}(x_i, y_j, z_k) + 1, 256).$$ (28)
The plain-image containing the modified pixel $P_{xyz}^1(x, y, z)$ was encrypted with the same key $k_{xy}(1)$, so that the corresponding cipher image $C_2$ could be obtained. According to Equation (27), NPCR and UACI were calculated using $C_1$ and $C_2$. Finally, by using different keys in key groups and slightly changed plain-images, a total of 200 pairs of NPCR and UACI were calculated. The average of the NPCR and UACI is shown in Table 6. Through Table 6, we can see that the NPCR and UACI values were very close to the ideal values, indicating that the encryption system is sensitive to any little change in plain-image. As shown in Tables 7 and 8, our NPCR and UACI mean values passed the randomness test, compared with the expected values [45].

Table 6. The mean NPCR and UACI of the some encrypted images, evaluating the plain-image sensitivity.

| Image     | NPCR (99.6094) | UACI (33.4635) |
|-----------|----------------|----------------|
| Lena      | 99.6086 99.6083 99.6104 | 33.4709 33.4683 33.4682 |
| Baboon    | 99.6088 99.6099 99.6088 | 33.4548 33.4618 33.4684 |
| Fruits    | 99.6091 99.6083 99.6091 | 33.4654 33.4647 33.4577 |
| Flowers   | 99.6094 99.6096 99.6083 | 33.4681 33.4591 33.4663 |
| Girl      | 99.6090 99.6101 99.6095 | 33.4670 33.4651 33.4641 |
| Flower    | 99.6095 99.6084 99.6102 | 33.4629 33.4614 33.4615 |
| Yacht     | 99.6086 99.6095 99.6087 | 33.4629 33.4654 33.4626 |
| Lena in Ref. [15] | 99.9985 99.9985 99.9985 | – – – |
| Lena in Ref. [16] | 99.6097 99.5994 99.5975 | 33.4476 33.4655 33.4769 |
| Lena in Ref. [37] | 99.4800 99.5158 99.4788 | 33.6322 33.7336 33.6005 |
| Lena in Ref. [41] | 99.6429 99.6140 99.6277 | 33.3935 33.5637 33.4814 |

Table 7. NPCR randomness test.

| Tested Image Size 512 by 512 | Theoretical NPCR Critical Value [45] | NPCR Test Results |
|-----------------------------|--------------------------------------|------------------|
| Our mean NPCR Value         | 0.05-level 0.01-level 0.001-level    | Pass Pass Pass    |
| 99.6091                     |                                      |                  |

Table 8. UACI randomness test.

| Tested Image Size 512 by 512 | Theoretical UACI Critical Value [45] | UACI Test Results |
|-----------------------------|--------------------------------------|------------------|
| Our mean UACI Value         | 0.05-level 0.01-level 0.001-level    | Pass Pass Pass    |
| 33.4691                     |                                      |                  |

4.2.4. Information Entropy Analysis

Information entropy is an important performance index which can be used to measure the randomness and unpredictability of an information source. So, it is usually used to measure the strength of a cryptosystem, given by

$$H(m) = \sum_{i=0}^{2^N-1} p(m_i) \log \frac{1}{p(m_i)},$$  \hspace{1cm} (29)
where \( m \) denotes an information source, and \( 2^N \) denotes the number of all possible pixel values. If \( m \) has \( 2^N \) possible values, the corresponding theoretical value should be \( H(m) = N \). So, for a cipher-image with 256 gray levels, the ideal value for the information entropy is 8. According to Equation (29), the information entropy of seven plain-images and their corresponding cipher-images—which were encrypted by our algorithm and other recent algorithms in the literature—were calculated as shown in Table 9. From this table, we can see that the information entropies of the cipher-images (RGB components) were all close to the ideal value. In other words, the cipher-images had a good property of randomness and the unpredictability.

| Image      | Plain-Image | Cipher-Image |
|------------|-------------|--------------|
|            | R           | G           | B           | R           | G           | B           |
| Lena       | 5.0465      | 5.4576      | 4.8001      | 7.9992      | 7.9993      | 7.9994      |
| Baboon     | 6.4998      | 6.4445      | 6.2709      | 7.9991      | 7.9992      | 7.9992      |
| Fruits     | 7.5172      | 7.3230      | 6.7785      | 7.9992      | 7.9992      | 7.9993      |
| Flowers    | 7.3824      | 7.2345      | 7.3641      | 7.9990      | 7.9990      | 7.9991      |
| Girl       | 7.4346      | 7.2354      | 7.0578      | 7.9995      | 7.9995      | 7.9996      |
| Flower     | 7.4428      | 7.4062      | 7.3371      | 7.9992      | 7.9993      | 7.9992      |
| Yacht      | 7.6071      | 7.4062      | 7.3371      | 7.9994      | 7.9991      | 7.9993      |
| Lena in Ref. [14] | –       | –           | –           | 7.9994      | 7.9994      | 7.9994      |
| Lena in Ref. [16] | –       | –           | –           | 7.9914      | 7.9915      | 7.9916      |
| Lena in Ref. [37] | –       | –           | –           | 7.9974      | 7.9975      | 7.9969      |
| Lena in Ref. [38] | –       | –           | –           | 7.9972      | 7.9972      | 7.9976      |
| Lena in Ref. [39] | –       | –           | –           | 7.9975      | 7.9972      | 7.9973      |
| Lena in Ref. [41] | –       | –           | –           | 7.9942      | 7.9943      | 7.9942      |

### 4.2.5. Encrypted Time Analysis

In practice, quick running speed is also significant for a good cryptosystem. The experimental environment was MATLAB R2014b with Intel Core i7-7500U CPU@ 3.5 GHz and 4.0 GB RAM on Windows 10 OS. Because the proposed scheme is bit-level permutation-based and related to plain images, encrypted time analysis was performed in comparison with similar algorithms for references [25,37,40]. Murillo-Escobar et al. presented a plaintext-related color image encryption algorithm based on total plain image characteristics and chaos in [37], in which the chaotic sequence used in permutation and diffusion is related to the total plain image characteristics. Then, the Z value needs to be inserted into the encrypted image, as it cannot be calculated from the encrypted image during the decryption process. In [40], a novel image encryption algorithm is proposed, in which the plain image is permuted by using the 2D rectangular transform. Then, in the diffusion stage, the previous encrypted pixel is used to encrypt the current pixel, while the first encrypted pixel in the cipher is related to all the pixels in the plain image. In [25], Xu et al. presented a novel bit-level image encryption algorithm that is based on piecewise linear chaotic maps (PWLCM). In the diffusion phase, the cycle shift operation is controlled by the sum of the binary sequence transformed from the plain image. In the confusion phase, the initial value of the chaotic map is determined by the permuted binary sequence.

The execution times for the proposed and comparable schemes including the time consumption in all of the encryption operations and key-stream generations are listed in Table 10. Encryption throughput (ET) in megabytes per second (MBps) and the number of cycles needed to encrypt one byte are also used to evaluate cryptosystem performance, and are given by Equations (30) and (31):

\[
ET = \frac{\text{Image Size (Byte)}}{\text{Encryption Time (second)}},
\]  

(30)
\[
\text{Number of cycles per Byte} = \frac{\text{CPU Speed (Hertz)}}{\text{ET (Byte)}}. \quad (31)
\]

Table 10. Encryption time (seconds).

| Scheme          | Color \((512 \times 512)\) | Color \((256 \times 256)\) | Gray \((512 \times 512)\) | Gray \((256 \times 256)\) | Platform     |
|-----------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|--------------|
| Proposed        | 4.6058                      | 1.1347                      | 1.8112                      | 0.4389                      | Matlab       |
| Ref. [37] (2015)| 0.3722                      | 0.1225                      | –                           | –                           | Matlab       |
| Ref. [40] (2017)| 14.8119                     | 3.6175                      | –                           | –                           | Matlab       |
| Ref. [25] (2016)| –                           | –                           | 12.6917                     | 3.1342                      | Matlab       |
| Ref. [46] (2013)| –                           | –                           | 0.030                       | 0.0075                      | VisualC++    |
| Ref. [47] (2011)| –                           | –                           | 0.033                       | 0.0078                      | C            |
| Ref. [16] (2016)| 0.009                       | 0.002                       | –                           | –                           | C            |
| Ref. [48] (2013)| –                           | –                           | 0.92                        | 0.16                        | Matlab       |

Table 11 presents a comparison of ET and number of cycles needed to encrypt performance of the proposed cryptosystem with some recent cryptosystems. As shown in Tables 10 and 11, the execution speed of the proposed scheme was slower than Murillo-Escobar’s algorithm, but more efficient than algorithms in [25,40]. There are three reasons for the relatively slower encryption speed compared to Murillo-Escobar’s algorithm. (1) Since the original color image has to be decomposed into 24 bit-planes in the permutation stage and then the permuted bit-planes are transformed into a color image, it will take more time than pixel-level permutation-based algorithms like [37]; (2) The image data to be processed in the permutation stage of bit-level permutation-based algorithms is eight times compared with pixel-level permutation-based algorithms; (3) The use of the sorting and searching operations in key-stream generation in our algorithms are particularly time-consuming. Considering its high level of security, the running speed is acceptable.

Table 11. Encryption throughput (ET) and number of cycles for one encrypted byte.

| Algorithm          | ET in MBps | Number of Cycles Per Byte | Platform  |
|--------------------|------------|----------------------------|-----------|
| Proposed           | 0.165      | 20229.45                   | Matlab    |
| Ref. [37] (Murillo-Escobar et al., 2015) | 1.531      | 2180.18                    | Matlab    |
| Ref. [40] (Wu et al., 2017) | 0.052      | 64189.61                   | Matlab    |
| Ref. [25] (Xu et al., 2016) | 0.020      | 166893.006                 | Matlab    |
| Ref. [46] (Zhang et al., 2013) | 8.33       | 366.35                     | VisualC++ |
| Ref. [47] (Wang et al., 2011) | 8.01       | 369.08                     | C         |
| Ref. [16] (M. Farajallah et al., 2016) | 93.82      | 31.51                      | C         |
| Ref. [48] (Pareekallah et al., 2013) | 0.27       | 10596.38                   | Matlab    |

5. Conclusions

In contrast with the similar studies, a plaintext-related random access bit-permutation mechanism (PRRABPM) is presented in this paper. This method is used in the permutation stage to shuffle the RGB components of a color image at the same time, making these three components interact with each other. Furthermore, the key streams used in random access bit-permutation mechanism operation is extremely dependent on plain image in an ingenious way, which makes the encryption system sensitive to key and original images. Thus, the proposed encryption system can efficiently resist the chosen-plaintext and chosen-ciphertext attacks. Experimental results analysis including key space, histogram, correlation, sensitivity, information entropy, and speed are also given, showing that the proposed algorithm has a good security performance.

The proposed method may be used for image security communication applications. According to simulation results, the proposed algorithm is not suitable for real-time applications. Now, our main concern is the strength of the encryption algorithm and its ability to work with the limitations of
security communication systems, which require further study. In our future work, we will consider this part in detail and improve the encryption speed. It may also be considered to integrate the image encryption with an image compression algorithm so as to enhance security for image transmission over communication systems.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (No.61201392), the Natural Science Foundation of Guangdong Province, China (No. 2015A030313497), the Science and Technology Planning Project of Guangdong Province, China (No.2017B09090004).

Author Contributions: Shuting Cai and Lingqing Huang conceived of and designed the experiments; Lingqing Huang performed the experiments; Lingqing Huang and Xuesong Chen analyzed the data; Xiaoming Xiong contributed reagents/materials/analysis tools; Shuting Cai and Lingqing Huang wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Matthews, R. On the derivation of a chaotic encryption algorithm. Cryptologia 1989, 8, 29–41. [CrossRef]
2. Fridrich, J. Symmetric ciphers based on two-dimensional chaotic maps. Int. J. Bifurc. Chaos 1998, 8, 1259–1284. [CrossRef]
3. Shannon, C.E. Communication theory of secrecy system. Bell Syst. Tech. J. 1949, 28, 656–715. [CrossRef]
4. Gao, T.; Chen, Z. A new image encryption algorithm based on hyper-chaos. Phys. Lett. A 2008, 372, 394–400. [CrossRef]
5. Mirzaei, O.; Yaghoobi, M.; Irani, H. A new image encryption method: parallel sub-image encryption with hyper chaos. Nonlinear Dyn. 2012, 67, 557–566. [CrossRef]
6. Murillo-Escobar, M.A.; Arellano-Delgado, A. A Fingerprint Image Encryption Scheme Based on Hyperchaotic Rössler Map. Math. Probl. Eng. 2016, 2016, 2670494.
7. Parvin, Z.; Seyedarabi, H.; Shamsi, M. A new secure and sensitive image encryption scheme based on new substitution with chaotic function. Multimed. Tools Appl. 2016, 75, 10631–10648. [CrossRef]
8. Murillo-Escobar, M.A.; Cardoza-Avendaño, L.; López-Gutiérrez, R.M.; Cruz-Hernández, C. A Double Chaotic Layer Encryption Algorithm for Clinical Signals in Telemedicine. J. Med. Syst. 2017, 41, 1–17. [CrossRef]
9. Zhang, X.; Wang, X. Multiple-image encryption algorithm based on mixed image element and permutation. Opt. Lasers Eng. 2017, 92, 6–16. [CrossRef]
10. Wang, X.; Teng, L.; Qin, X. A novel colour image encryption algorithm based on chaos. Signal Process. 2012, 92, 1101–1108. [CrossRef]
11. Liu, H.; Wang, X. Color image encryption based on one-time keys and robust chaotic maps. Comput. Math. Appl. 2010, 59, 3320–3327. [CrossRef]
12. Wang, X.Y.; Zhang, Y.Q.; Bao, X.M. A Colour Image Encryption Scheme Using Permutation-Substitution Based on Chaos. Entropy 2015, 17, 3877–3897. [CrossRef]
13. Huang, X.; Sun, T.; Liang, J. A Color Image Encryption Algorithm Based on a Fractional-Order Hyperchaotic System. Entropy 2015, 17, 28–38. [CrossRef]
14. Stoyanov, B.; Kordov, K. Image Encryption Using Chebyshev Map and Rotation Equation. Entropy 2015, 17, 2117–2139. [CrossRef]
15. Huang, H.; Yang, S. Color image encryption based on logistic mapping and double random-phase encoding. IET Image Process. 2017, 11, 211–216. [CrossRef]
16. Farajallah, M.; Assad, S.E.; Deforges, O. Fast and Secure Chaos-Based Cryptosystem for Images. Int. J. Bifurc. Chaos 2016, 26, 1650021-1–1650021-21. [CrossRef]
17. Pak, C.; Huang, L. A new color image encryption using combination of the 1d chaotic map. Signal Process. 2017, 138, 129–137. [CrossRef]
18. Tao, X.; Wong, K.; Liao, X. Selective image encryption using a spatiotemporal chaotic system. Chaos 2007, 17, 023115.
19. Zhu, Z.L.; Zhang, W.; Wong, K.W.; Yu, H. A chaos-based symmetric image encryption scheme using a bit-level permutation. Inf. Sci. 2011, 181, 1171–1186. [CrossRef]
20. Zhang, W.; Wong, K.W.; Yu, H.; Zhu, Z.L. An image encryption scheme using lightweight bit-level confusion and cascade cross circular diffusion. Opt. Commun. 2012, 285, 2343–2354. [CrossRef]
21. Zhang, W.; Wong, K.W.; Yu, H.; Zhu, Z.L. A symmetric color image encryption algorithm using the intrinsic features of bit distributions. *Commun. Nonlinear Sci. Numer. Simul.* 2013, 18, 584–600. [CrossRef]

22. Fu, C.; Huang, J.B.; Wang, N.N.; Hou, Q.B.; Lei, W.M. A symmetric chaos-based image cipher with an improved bit-level permutation strategy. *Entropy* 2014, 16, 770–788. [CrossRef]

23. Zhu, H.; Zhang, X.; Yu, H.; Zhao, Y.L.; Zhu, Z.L. Image encryption based on three-dimensional bit matrix permutation. *Signal Process.* 2014, 118, 36–50. [CrossRef]

24. Rhouma, R.; Belghith, S. Cryptanalysis of a new image encryption algorithm based on hyper-chaos. *Signal Process.* 2008, 372, 5973–5978. [CrossRef]

25. Wang, X.; Liu, L.; Hu, W. A novel bit-level image encryption algorithm based on chaotic maps. *Opt. Lasers Eng.* 2016, 78, 17–25. [CrossRef]

26. Liu, W.; Sun, K.; Zhu, C. A fast image encryption algorithm based on chaotic map. *Opt. Lasers Eng.* 2016, 84, 26–36. [CrossRef]

27. Mollaeefar, M.; Sharif, A.; Nazari, M. A novel encryption scheme for colored image based on high level chaotic maps. *Signal Process.* 2015, 76, 1–23. [CrossRef]
47. Wang, Y.; Wong, K.W.; Liao, X.; Chen, G. A new chaos-based fast image encryption algorithm. *Appl. Soft Comput.* **2011**, *11*, 514–522. [CrossRef]

48. Pareek, N.K.; Patidar, V.; Sud, K.K. Diffusion–substitution based gray image encryption scheme. *Digit. Signal Process.* **2013**, *23*, 894–901. [CrossRef]