entropy

MDPI

Article

Encryption of Color Images with an Evolutionary Framework Controlled by Chaotic Systems

Xinpeng Man and Yinglei Song *

School of Automation, Jiangsu University of Science and Technology, Zhenjiang 212003, China; 202030060@stu.just.edu.cn

* Correspondence: syinglei2013@163.com; Tel.: +86-0511-84401157

Abstract: In the past decade, a large amount of important digital data has been created and stored in the form of color images; the protection of such data from undesirable accesses has become an important problem in information security. In this paper, a new approach based on an evolutionary framework is proposed for the secure encryption of color images. The image contents in a color image are first fully scrambled with a sequence of bit-level operations determined by a number of integer keys. A scrambled image is then encrypted with keys generated from an evolutionary process controlled by a set of chaotic systems. Analysis and experiments show that the proposed approach can generate encrypted color images with high security. In addition, the performance of the proposed approach is compared with that of a few state-of-the-art approaches for color image encryption. The results of the comparison suggest that the proposed approach outperforms the other approaches in the overall security of encrypted images. The proposed approach is thus potentially useful for applications that require color image encryption.

Keywords: image encryption; color images; evolutionary process; chaotic systems; security

check for updates

Citation: Man, X.; Song, Y. Encryption of Color Images with an Evolutionary Framework Controlled by Chaotic Systems. *Entropy* **2023**, *25*, 631. https://doi.org/10.3390/e25040631

Academic Editor: Congxu Zhu

Received: 13 March 2023
Revised: 2 April 2023
Accepted: 2 April 2023
Published: 7 April 2023

1. Introduction

With the rapid development of information technologies, a tremendous amount of image data has been created for the analysis, storage and transmission of important information. In practice, undesirable accesses to certain image data often need to be prevented and encryption is a computational technique extensively utilized to enhance the security of such images [1]. Since images encrypted by traditional encryption methods generally cannot reach the security levels required by many applications [2], researchers have proposed numerous methods that can encrypt images with improved security [2–7]. Most of the existing approaches use techniques from one of three major classes of methods for image encryption. These classes consist of approaches that apply the permutation of pixel positions [7,8], techniques that transform gray values of pixels [5,9,10] and methods that encrypt images with chaotic systems [6,11].

An important method for image encryption permutes the positions of pixels to generate a cipher image. For example, a skew tent map system is combined with a permutation–diffusion architecture in [7] for image encryption. In [8], cipher images are generated by changing both gray values and locations of pixels. The approach proposed in [12] utilizes ergodic matrices to change the locations of pixels for image encryption. The approach proposed in [13] uses an elliptic curve random generator and an AES to improve the security of cipher images. Peano–Hilbert curves are used in [14] to relocate pixels such that spatial correlations can be eliminated in cipher images. In [15], edge maps generated based on source images are used along with a number of different permutation techniques for image encryption.

Gray value transformation is another technique that has been extensively used for image encryption. The image contents in each pixel of an image are processed by a

transformation and new pixel values are generated for the pixel. The approach proposed in [9] encrypts an image multiple times with the fractional Fourier Transform (FRFT). In [5], images are iteratively encrypted by an approach that uses gyrator transform and random phase encoding. The approach proposed in [16] encrypts an image with a combination of Arnold transformation and gyrator transformation. In [8], the hue(H), saturation(S) and intensity(I) components of a color image are processed with discrete fractional random transform (DFRNT) and Arnold transform for encryption. In [17], image encryption is performed based on a new category of Discrete Fractional Fourier Transforms (DFrFs) with eigensystems generated by a new random-matrix scheme. The approach proposed in [18] uses DNA encoding to diversify the Elliptic Curve Cryptography (ECC) to obtain cipher images with improved security. A novel modular approach is proposed in [19] to construct a nonlinear S-box for image encryption. Self-adaptive permutation–diffusion and deoxyribonucleic acid (DNA) random encoding are combined in [20] for the adaptive encryption of images. The work in [21] encrypts images with an approach that uses both compressive sensing and information hiding. In [22], an approach that integrates Arnold map, DNA sequence operations with a Mandelbrot set are proposed to securely encrypt color images. Recently, DNA coding and compressive sensing have been combined to obtain cipher images with improved security [23]. The work in [24] proposes a novel layer-based image steganography method that can hide a color image into color images.

A large number of approaches have been proposed to perform image encryption with chaotic systems [3,6]. Such approaches generally utilize outputs of chaotic systems for the generation of cipher images. Since a tiny amount of change in initial values can alter the behavior of a chaotic system significantly, the exact initial values of these systems must be obtained to decipher a cipher image. The security of a cipher image can thus be significantly enhanced when chaotic systems are used for encryption. Chaotic systems are used in [4] to generate chaotic sequences that can change the locations of pixels for encryption. In [10], three sets of pseudo random sequences are generated with a simple perception and a high-dimensional chaotic system to encrypt images. In [25], images are encrypted with a pseudo-random key stream sequence generated by a piecewise linear chaotic map. In [26], an image is encrypted with two sets of one-dimensional logistic systems designed for pixel relocation and gray value transformation, respectively. The work in [27] designs a neotype chaotic product trigonometric map (PTM) system for image encryption. The approach proposed in [28] combines chaotic systems with particle swarm optimization algorithm to improve the security of color image encryption. In [29], a dual permutation and dual substitution framework that utilizes cellular automata, chaos theory and image mixing is proposed for color image encryption. The work in [30] proposes an Enhanced Logistic Map (ELM) that can be combined with chaotic maps and simple encryption techniques to improve the security of cipher images.

Recently, numerous methods that combine DNA operations with chaotic systems have been developed for image encryption [22,31–33]. The approach proposed in [31] encrypts images with a combination of DNA sequence operations and chaotic systems. The work in [34] performs image fusion encryption with an approach based on hyper-chaotic systems and DNA sequence operations. A hybrid model is proposed in [35] to combine DNA masking, Lorenz system and a Secure Hash Algorithm SHA-2 for image encryption. In [36], highly secure cipher images are obtained by combining DNA sequence operations with various types of chaotic systems. The work in [33] proposes a method that performs image encryption with a 5D hyper chaotic system and DNA technology.

Research results have shown that using chaotic systems together with other encryption approaches can also improve the security of cipher images. In [37], global chaotic pixel diffusion is used together with fractal sorting matrices (FSM) for image encryption. The approach proposed in [38] applies the Knuth–Durstenfeld algorithm with a hidden attractor chaos system to improve the security of cipher images. In [39], color images are encrypted with a two-dimensional logistic tent modular map (2D-LTMM). In [40], chaotic systems are integrated with a permutation–substitution (SP) network to achieve improved security

for image encryption. In [41], images are encrypted by chaotic data generated with Mixed Linear–Nonlinear Coupled Map Lattices (MLNCML).

Existing methods for image encryption have significantly improved the overall security of cipher images. However, most of the state-of-the-art approaches for image encryption only change the locations of pixels to reduce the local correlations among pixels. The image contents associated with a pixel are completely or partially retained after it is relocated to a different position in the image. The original image contents are thus not completely eliminated by the scrambling process. Moreover, the transformations utilized to change the image contents in each pixel are generally based on numeric functions only. The security of cipher images can possibly be further improved if more sophisticated transformations are available for encoding the image contents associated with each pixel in an image. A new approach that can perform the scrambling process more effectively and apply a more sophisticated transformation to change the image contents associated with each pixel is thus highly desirable to achieve further improved security for image encryption.

In this paper, a new method is proposed to encrypt color images with an evolutionary framework controlled by a set of chaotic systems. In the first phase of the encryption, pixels in a color image are grouped based on a virtual image constructed from the original image and arrays formed by the binary bits in the pixel groups are shuffled with a set of integer keys. The R, G and B components of pixels are then determined from the shuffled arrays as the result of scrambling. In the second phase of encryption, an evolutionary system comprised of integers obtained from a set of 3D chaotic systems is used to change the image contents in each pixel of the scrambled image. Integers in the evolutionary system are processed and changed by two operators, including cross-over and mutation. The operations of each operator are controlled by a set of one-dimensional logistic systems.

The proposed method has two major contributions for the encryption of color images. Firstly, a bit-level scrambling process is developed to change the location of each bit in the image contents of a color image. Correlations that usually exist between pixels that are spatially close in natural images can thus be significantly reduced. Secondly, due to the fact that it is difficult to simulate the outputs of an evolutionary process controlled by chaotic systems with numeric transformations, the underlying mechanisms of the proposed approach can be well hidden from adversary sides. The proposed method can thus significantly enhance the robustness and security of cipher images.

The results of an analysis on the key space of the proposed approach suggest that it is robust against exhaustive attacks. Experimental results on benchmark color images and a set of images selected from the BSD500 dataset [42] show that the proposed approach can generate cipher images with high security. In addition, the performance of the proposed approach is compared with that of other state-of-the-art encryption approaches on a variety of security measures for cipher images. The results of comparison show that the proposed approach is able to provide cipher images with security higher than those generated by other tested approaches.

2. Materials and Methods

The encryption of a color image with the proposed approach is performed in two phases. In the first phase, the binary bits of the R, G and B components in the pixels of the plain image are scrambled based on a set of integer keys and a virtual image constructed from the plain image. In the second phase, the R, G and B components in each pixel of the scrambled image are encoded by an evolutionary system constructed from the outputs of a set of 3D chaotic systems. The evolutionary process of the system is controlled by a set of one-dimensional logistic systems.

Figure 1 illustrates the scrambling process in the proposed approach. In the first step, a virtual image is constructed from the plain image by a mapping that can place each pixel in the plain image to its mapped location in the virtual image. Pixels in the plain image are organized into groups based on the rows and columns of the virtual image. In the second step, the pixels in the same row are included in a group and the binary bits of the R, G and

B components of the pixels in the same group are combined into an array of bits. The bits in each array are shuffled with an integer key associated with the corresponding row. The R, G and B components of pixels are replaced by bits from the corresponding locations in the shuffled arrays. In the third step, the pixels in the same column are included in a group and the same bit-shuffling operation is performed for pixels in each group. The resulting image is a scrambled image of the plain image.

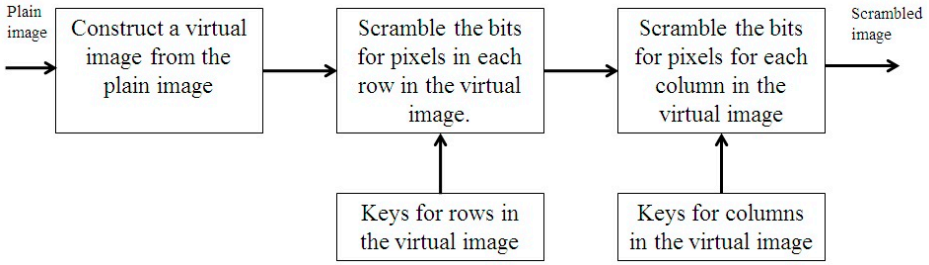


Figure 1. Steps in the scrambling process of the proposed approach.

Figure 2 shows the steps followed by the proposed approach to encode the R, G and B components of pixels in a scrambled image. For each pixel, an evolutionary system is constructed from the outputs of a set of 3D chaotic systems. The integers in the evolutionary system are then varied by a number of cross-over and mutation operations controlled by a set of one-dimensional logistic systems. Finally, an integer is selected for each component of the pixel from the integers in the evolutionary system and the component is encoded by an integer key computed from the selected integer.

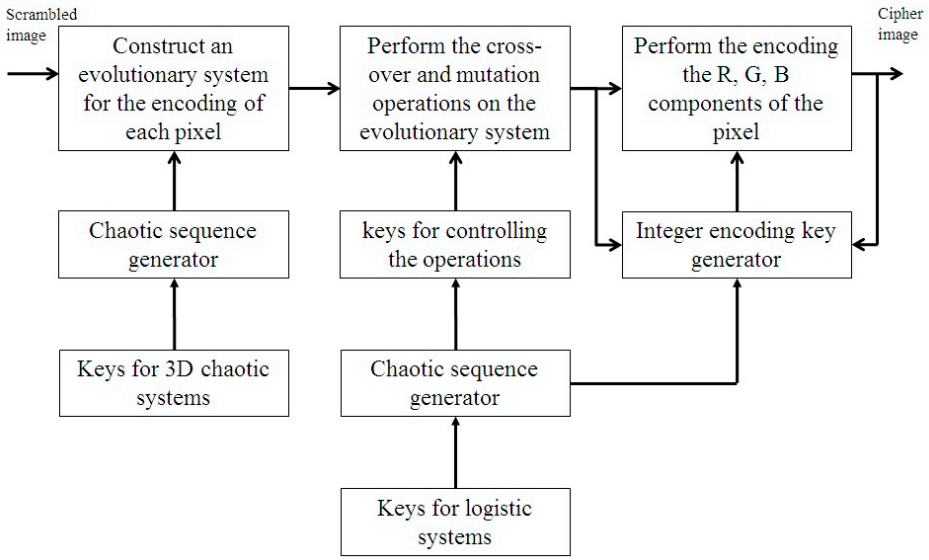


Figure 2. Steps in the encoding process of the proposed approach.

The decryption of a cipher image can be performed in two steps. In the first step, the evolutionary system used for the encoding of each pixel is constructed and the corresponding cross-over and mutation operations are applied to the integers in the system. The integers selected for encoding the R, G and B components are obtained from the system and the components of the pixel in the scrambled image are computed from the selected integers. In the second step, the virtual image used for the scrambling process in encryption is constructed and pixels are grouped by columns in the virtual image. A reversed bit-shuffling operation is performed to reset the R, G and B components of pixels for each group. In the third step, pixels are grouped by rows in the virtual image and a reversed bit-shuffling operation is performed to reset the R, G and B components of pixels for each group. The resulting image is the plain image decrypted from the given cipher image.

2.1. The Bit-Level Scrambling Process

Let $I(m,n,3)$ be a plain color image that contains m rows and n columns; the R, G and B components of the pixel in the jth row and kth column are denoted by $I(j,k,1)$, $I(j,k,2)$ and $I(j,k,3)$, respectively, where $1 \leq j \leq m$ and $1 \leq k \leq n$ hold for integers j and k. A bit-level scrambling of I considers the binary bits in the R, G and B components of all pixels in I and changes the location of each bit in the image, the resulting image $S(m,n,3)$ is a scrambled image of I.

2.1.1. Virtual Image

As the first step of the scrambling process, a virtual image $V(p,q,3)$ is generated from I, where q is a given integer that satisfies $2 \leq q \leq \lfloor mn/2 \rfloor$ and $p = \lceil mn/q \rceil$ holds for p. V is obtained from I by a mapping Ψ that associates each pixel in I with a pixel in V. Specifically, for each pair of integers (j,k) where $1 \leq j \leq m$ and $1 \leq k \leq n$ hold, Ψ maps the pixel in the jth row and kth column in I to a pixel in row $R(j,k)$ and column $C(j,k)$ in V, where $R(j,k)$ and $C(j,k)$ are determined based on Equations (1) and (2).

$$R(j,k) = \begin{cases} 1, & if\ j = 1\ and\ k = 1 \\ \left\lceil \frac{(j-1)n+k-1}{q} \right\rceil, & otherwise \end{cases} \tag{1}$$

$$C(j,k) = ((j-1)n + k - 1)\ mod\ q + 1 \tag{2}$$

In addition, the following equations hold for each pair of integers (j,k) where j and k satisfy $1 \leq j \leq m$ and $1 \leq k \leq n$.

$$V(R(j,k), C(j,k), 1) = I(j,k,1) \tag{3}$$

$$V(R(j,k), C(j,k), 2) = I(j,k,2) \tag{4}$$

$$V(R(j,k), C(j,k), 3) = I(j,k,3) \tag{5}$$

2.1.2. Scrambling Based on Rows in the Virtual Image

It is clear from Equations (1) and (2) that row p in V may contain less than q pixels mapped from I. For each integer h that satisfies $1 \leq h \leq p$, let r_h be the number of mapped pixels in row h of V; row h in V is assigned a positive integer k_h for scrambling. k_h is required to be coprime with $24q$ for $1 \leq h < p$ and k_p must be coprime with $24r_p$.

For each integer h that satisfies $1 \leq h \leq p$, the 8-bit binary representations of the R, G and B components of the pixels in row h of V are sequentially combined into an array of $24r_h$ binary bits. Specifically, let B_h be the array of binary bits constructed for row h and l be an integer that satisfies $1 \leq l \leq r_h$; the binary bits from positions $24(l-1)+1$ through to $24(l-1)+8$ in B_h are the 8-bit binary representation of $V(h,l,1)$. Similarly, the binary bits from positions $24(l-1)+9$ through to $24(l-1)+16$ in B_h are the 8-bit binary representation of $V(h,l,2)$ and the binary bits from positions $24(l-1)+17$ through to $24l$ in B_h are the 8-bit binary representation of $V(h,l,3)$. The order of bits in B_h is changed for each integer h that satisfies $1 \leq h \leq p$ to complete the row-based scrambling operation.

Let s be an integer that satisfies $1 \leq s \leq 24r_h$; $B_h(s)$ denotes the sth bit in B_h. $B_h(s)$ is relocated to position t in B_h, where t is determined from k_h and r_h based on Equation (6).

$$t = ((s-1)k_h + 2k_h - 1) mod\ 24r_h + 1 \tag{6}$$

Since k_h is coprime with $24r_h$, no two bits in B_h are relocated to the same position in B_h. Otherwise, there exists two different integers s_1 and s_2 that satisfy $1 \leq s_1 \leq 24r_h$ and $1 \leq s_2 \leq 24r_h$, and are relocated to the same position based on Equation (6). This implies that Equation (7) holds for s_1 and s_2.

$$(s_1 - s_2)k_h = 24r_h u \tag{7}$$

where u is an integer. Since k_h is coprime with $24r_h$, Equation (7) implies that $24r_h$ is a factor of $s_1 - s_2$. However, due to the fact that $0 < |s_1 - s_2| < 24r_h$ holds, $24r_h$ cannot be a factor of $s_1 - s_2$, which is a contradiction. This implies that no such pair of integers exists and a relocation of all bits in B_h can be performed based on Equation (6). Figure 3a shows the pseudocode for scrambling the binary bits of pixels in row h of the virtual image.

Inputs: V, h, r_h
for $l = 1$ to r_h
  for j = 1 to 8
    $B_h(24(l-1)+j)$= the jth bit in $V(h,l,1)$
  end
  for j = 1 to 8
    $B_h(24l-16+j)$= the jth bit in $V(h,l,2)$
  end
  for j = 1 to 8
    $B_h(24l-8+j)$= the jth bit in $V(h,l,3)$
  end
end
for $s = 1$ to $24r_h$
  $t = \big((s-1)k_h + 2k_h - 1\big) mod\ 24r_h + 1$
  $N_h(t) = B_h(s)$
end
for $s = 1$ to $24r_h$
  $B_h(s) = N_h(s)$
end
Output: B_h

(**a**)

Inputs: V, d, c_d
for $v = 1$ to c_d
  for j = 1 to 8
    $D_d(24(v-1)+j)$= the jth bit in $V(v,d,1)$
  end
  for j = 1 to 8
    $D_d(24v-16+j)$= the jth bit in $V(v,d,2)$
  end
  for j = 1 to 8
    $D_d(24v-8+j)$= the jth bit in $V(v,d,3)$
  end
end
for $g = 1$ to $24c_d$
  $y = \big((g-1)w_d + 2w_d - 1\big) mod\ 24c_d + 1$
  $N_d(y) = D_d(g)$
end
for $g = 1$ to $24c_d$
  $D_d(g) = N_d(g)$
end
Output: D_d

(**b**)

Figure 3. Pseudocodes for scrambling. (**a**) Scrambling the binary bits of pixels in row h in the virtual image; (**b**) scrambling the binary bits of pixels in column d in the virtual image.

After all bits in B_h have been relocated, the R, G and B components of pixels in row h of V are reset based on B_h. Specifically, for each integer l that satisfies $1 \leq l \leq r_h$, the 8-bit binary representation of $V(h,l,1)$ is reset to be the binary bits from positions $24(l-1)+1$ through to $24(l-1)+8$ in B_h. Similarly, the 8-bit binary representation of $V(h,l,2)$ is reset to be the binary bits from positions $24(l-1)+9$ through to $24(l-1)+16$ in B_h, and the 8-bit binary representation of $V(h,l,3)$ is reset to be the binary bits from positions $24(l-1)+17$ through to $24l$ in B_h.

2.1.3. Scrambling Based on Columns in the Virtual Image

Similarly, Equations (4) and (5) suggest that a column in V may contain p or $p-1$ pixels mapped from I. For each integer d that satisfies $1 \leq d \leq q$, c_d denotes the number of mapped pixels in column d of V. A positive integer w_d is assigned to column d in V and w_d must satisfy the requirement that it is coprime with $24c_d$.

For each integer d such that $1 \leq d \leq q$ holds, an array of $24c_d$ binary bits is constructed from a sequential combination of the 8-bit binary representations for the R, G and B components of the pixels in column d of V. Specifically, D_d denotes the array of binary bits obtained from pixels in column d; for integer v that satisfies $1 \leq v \leq c_d$, the 8-bit binary representation of $V(v,d,1)$ is placed in positions $24(v-1)+1$ through to $24(v-1)+8$ of D_d. Similarly, the 8-bit binary representation of $V(v,d,2)$ is placed in positions $24(v-1)+9$ through to $24(v-1)+16$ of D_d and the 8-bit binary representation of $V(v,d,3)$ is placed in positions $24(v-1)+17$ through to $24v$ of D_d. A complete column-based scrambling operation changes the location of each bit in D_d for each integer d that satisfies $1 \leq d \leq q$.

Let g be an integer that satisfies $1 \leq g \leq 24c_d$ and $D_d(g)$ be the gth bit in D_d. The location of $D_d(g)$ is changed to position y in D_d, where y is determined from w_d and c_d based on Equation (8).

$$y = ((g-1)w_d + 2w_d - 1) mod\ 24c_d + 1 \tag{8}$$

Based on an argument similar to the one presented in Section 2.1.2, no two bits in D_d are relocated to the same position in D_d due to the fact that w_d is coprime with $24c_d$. Equation (8)

thus generates a relocation of all bits in D_d. Figure 3b shows the pseudocode for scrambling the binary bits of pixels in column d of the virtual image.

After the locations of all bits in D_d have been changed, D_d is used to reset the R, G and B components of pixels in column d of V. For each integer v that satisfies $1 \leq v \leq c_d$, the binary bits from positions $24(v-1)+1$ through to $24(v-1)+8$ in D_d are assigned to $V(v,d,1)$, the binary bits from positions $24(v-1)+9$ through to $24(v-1)+16$ in D_d are assigned to $V(v,d,2)$ and the binary bits from positions $24(l-1)+17$ through to $24l$ in D_d are assigned to $V(v,d,3)$. A scrambled image $S(m,n,3)$ is obtained for $I(m,n,3)$ after the column-based scrambling operation is completed. In practice, the operations in the scrambling process can be performed multiple times to further improve the security of encryption.

2.1.4. Recover a Plain Image from Its Scrambled Image

Let q be the number of columns in the virtual image used to scramble the plain image in encryption. The virtual image $V(p,q,3)$ used for the scrambling of the plain image is constructed from S, where $p = \lceil mn/q \rceil$. For each pair of integers (j,k) where $1 \leq j \leq m$ and $1 \leq k \leq n$ hold, the pixel that corresponds to (j,k) in V is obtained from S based on Equations (9)–(11).

$$V(R(j,k),C(j,k),1) = S(j,k,1) \tag{9}$$

$$V(R(j,k),C(j,k),2) = S(j,k,2) \tag{10}$$

$$V(R(j,k),C(j,k),3) = S(j,k,3) \tag{11}$$

where $R(j,k)$ and $C(j,k)$ are obtained based on Equations (1) and (2).

For each integer d that satisfies $1 \leq d \leq q$, let c_d be the number of mapped pixels in column d of V and w_d be the integer key associated with column d in V for scrambling. A sequential combination of the 8-bit binary representations for the R, G and B components of the pixels in column d of S are performed to construct an array D_d of $24c_d$ binary bits. Specifically, for each integer v that satisfies $1 \leq v \leq c_d$, the binary bits in positions $24(v-1)+1$ through to $24(v-1)+8$ of D_d are the 8-bit binary representation of $S(v,d,1)$, the binary bits in positions $24(v-1)+9$ through to $24(v-1)+16$ of D_d are the 8-bit binary representation of $S(v,d,2)$, and the binary bits in positions $24(v-1)+17$ through to $24v$ of D_d are the 8-bit binary representation of $S(v,d,3)$.

Figure 4a shows the pseudocode for recovering the binary bits in column d of the virtual image. For each integer g such that $1 \leq g \leq 24c_d$ holds, an integer y is determined from w_d and c_d using Equation (8). The location of $D_d(y)$ is changed to position g in D_d. After the locations of all bits in D_d are changed, D_d is used to reset the R, G and B components of pixels in column d of V. For each integer v that satisfies $1 \leq v \leq c_d$, $V(v,d,1)$ is set to be the value of the binary bits from positions $24(v-1)+1$ through to $24(v-1)+8$ in D_d, $V(v,d,2)$ is set to be the value of the binary bits from positions $24(v-1)+9$ through to $24(v-1)+16$ in D_d and $V(v,d,3)$ is set to be the value of the binary bits from positions $24(v-1)+17$ through to $24v$ in D_d.

For each integer h that satisfies $1 \leq h \leq p$, let r_h be the number of mapped pixels in row h of V and k_h be the integer key associated with row h for scrambling. The 8-bit binary representations of the R, G and B components of the pixels in row h of V are sequentially combined into an array B_h of $24r_h$ binary bits. Specifically, for each integer l that satisfies $1 \leq l \leq r_h$, the binary bits from positions $24(l-1)+1$ through to $24(l-1)+8$ in B_h are set to be the 8-bit binary representation of $V(h,l,1)$, the binary bits from positions $24(l-1)+9$ through to $24(l-1)+17$ in B_h are set to be the 8-bit binary representation of $V(h,l,2)$ and the binary bits from positions $24(l-1)+17$ through to $24l$ in B_h are the 8-bit binary representation of $V(h,l,3)$.

Inputs: d, c_d, D_d
for $g = 1$ to $24c_d$
$y = \big((g-1)w_d + 2w_d - 1\big) mod\ 24c_d + 1$
$N_d(g) = D_d(y)$
end
for $g = 1$ to $24c_d$
$D_d(g) = N_d(g)$
end
for $v = 1$ to c_d
for j = 1 to 8
the jth bit in $V(v, d, 1) = D_d(24(v-1) + j)$
end
for j = 1 to 8
the jth bit in $V(v, d, 2) = D_d(24v - 16 + j)$
end
for j = 1 to 8
the jth bit in $V(v, d, 3) = D_d(24v - 8 + j)$
end
end
Outputs: the R, G, B components of pixels in column d of V.

(**a**)

Inputs: h, r_h, B_h
for $s = 1$ to $24r_h$
$t = \big((s-1)k_h + 2k_h - 1\big) mod\ 24r_h + 1$
$N_h(s) = B_h(t)$
end
for $s = 1$ to $24r_h$
$B_h(s) = N_h(s)$
end
for $l = 1$ to r_h
for j = 1 to 8
the jth bit in $V(h, l, 1)$=$B_h(24(l-1) + j)$
end
for j = 1 to 8
the jth bit in $V(h, l, 2) = B_h(24l - 16 + j)$
end
for j = 1 to 8
the jth bit in $V(h, l, 3) = B_h(24l - 8 + j)$
end
end
Outputs: the R, G, B components of pixels in row h of V.

(**b**)

Figure 4. Pseudocodes for the recovery of a plain image from its scrambled image. (**a**) Recovering the binary bits of pixels in column d in the virtual image; (**b**) recovering the binary bits of pixels in row h in the virtual image.

Figure 4b shows the pseudocode for recovering the binary bits in row h of the virtual image. For each integer that satisfies $1 \le s \le 24r_h$, an integer t is determined from k_h and r_h based on Equation (6), and $B_h(t)$ is relocated to position s in B_h. After all bits in B_h have been relocated, the R, G and B components of pixels in row h of V are reset based on B_h. For each integer l that satisfies $1 \le l \le r_h$, the value of the binary bits from positions $24(l-1)+1$ through to $24(l-1)+8$ in B_h is assigned to $V(h, l, 1)$, the value of the binary bits from positions $24(l-1)+9$ through to $24(l-1)+16$ in B_h is assigned to $V(h, l, 2)$ and the value of the binary bits from positions $24(l-1)+17$ through to $24l$ in B_h is assigned to $V(h, l, 3)$. The plain image can be obtained after the pixels in V have been reset.

2.2. Encoding a Scrambled Image

2.2.1. A 3D Chaotic System

Recently, a new 3D chaotic system is proposed in [43]. The system is formulated in spherical coordinates, and it is shown in [43] that four hidden attractors and three unstable equilibrium points exist for the system. The hidden attractors include two limit cycles and two strange attractors. One of the strange attractors is inside a sphere of radius 7.0 and the other one is outside the sphere. A description of the system in spherical coordinates is shown in Equation (12).

$$\begin{cases} \dot{\rho} = \rho\varphi - 7\varphi \\ \dot{\theta} = -\beta\theta^2 + \varphi^2 \\ \dot{\varphi} = -\rho^2 + \alpha\theta^2 + 14\rho + \varphi - 49 \end{cases} \tag{12}$$

where ρ,φ and θ are the radial distance, polar angle and azimuthal angle, respectively. The simulation results show that the system demonstrates a chaotic behavior when parameters α and β are set to be 3.0 and 1.0, respectively. A more detailed analysis of the dynamical properties of the system can be found in [43].

2.2.2. The Evolutionary System for Encryption

The encryption of a scrambled image is based on an evolutionary system constructed based on a set of 3D chaotic systems described by Equation (9) with different initial values. Let b be a positive integer and $I_1 = \{\rho_{1,0}, \rho_{2,0}, \ldots, \rho_{b,0}\}$, $I_2 = \{\theta_{1,0}, \theta_{2,0}, \ldots, \theta_{b,0}\}$ and $I_3 = \{\varphi_{1,0}, \varphi_{2,0}, \ldots \varphi_{b,0}\}$ be three number sets that contain the initial values for b different 3D chaotic systems $K_1, K_2, \ldots, K_b$. Specifically, for integer i such that $1 \le i \le b$ holds, $\rho_{i,0}$, $\theta_{i,0}$ and $\varphi_{i,0}$ are the initial values of ρ, θ and φ for system K_i. For the pixel

in row j and column k in $S(m,n,3)$, let $L = (j-1)n + k - 1$; three sets of integers $E_{1,0} = \{X_{1,L}, X_{2,L}, \ldots, X_{b,L}\}$, $E_{2,0} = \{Y_{1,L}, Y_{2,L}, \ldots, Y_{b,L}\}$ and $E_{3,0} = \{Z_{1,L}, Z_{2,L}, \ldots, Z_{b,L}\}$ can be obtained with Equation (13).

$$\begin{cases} X_{i,L} = \lfloor M \times |x_{i,L}| \rfloor \ mod\ 256 \\ Y_{i,L} = \lfloor M \times |y_{i,L}| \rfloor \ mod\ 256 \\ Z_{i,L} = \lfloor M \times |z_{i,L}| \rfloor \ mod\ 256 \end{cases} \tag{13}$$

where i is an integer that satisfies $1 \leq i \leq b$ and M is a large positive integer;$x_{i,L}$, $y_{i,L}$ and $z_{i,L}$ are computed based on Equation (14).

$$\begin{cases} x_{i,L} = \rho_{i,L} cos\theta_{i,L} sin\varphi_{i,L} \\ y_{i,L} = \rho_{i,L} sin\theta_{i,L} sin\varphi_{i,L} \\ z_{i,L} = \rho_{i,L} cos\varphi_{i,L} \end{cases} \tag{14}$$

where $\rho_{i,L}$, $\theta_{i,L}$ and $\varphi_{i,L}$ are the outputs of K_i at time $L\Delta t$, given $\rho_{i,0}$, $\theta_{i,0}$ and $\varphi_{i,0}$ as its initial values for ρ, θ and φ, respectively.Δt is a positive constant.

The integer sets $E_{1,0}$, $E_{2,0}$ and $E_{3,0}$ together form the initial configuration of the evolutionary system for encoding the pixel in row j and column k in $S(m,n,3)$. The integers in $E_{1,0}$, $E_{2,0}$ and $E_{3,0}$ are changed by a series of cross-over and mutation operations controlled by a set of one-dimensional logistic systems; the resulting sets are denoted by $E_{1,f}$, $E_{2,f}$ and $E_{3,f}$. The integer keys for the encoding of $S(j,k,1)$, $S(j,k,2)$ and $S(j,k,3)$ are computed based on integers selected from $E_{1,f}$, $E_{2,f}$ and $E_{3,f}$, respectively.

2.2.3. The Cross-Over Operation

A cross-over operation for the evolutionary system is controlled by a set of one-dimensional logistic systems. A logistic system is defined by an initial value l_0 and the recursion relation shown in Equation (15).

$$l_{i+1} = 4l_i(1 - l_i) \tag{15}$$

where i is a positive integer. A well-known fact is that the logistic system defined in Equation (15) is a one-dimensional chaotic system when l_0 satisfies $0 < l_0 < 1$ [44]. The system defined in Equation (15) generates a sequence of numbers $l_1, l_2, \ldots, l_N, \ldots$ for a given initial value l_0. Due to the chaotic property of the system, a tiny amount of change in l_0 would lead to a significantly different l_N if N is a large enough integer.

Three one-dimensional logistic systems with different initial values are needed to complete a cross-over operation. The initial values of the three one-dimensional logistic systems are denoted by $l_{1,0}$, $l_{2,0}$ and $l_{3,0}$. For the pixel in row j and column k in $S(m,n,3)$, three sets of integers $V_{1,0} = \{u_{1,L}, u_{2,L}, \ldots, u_{b,L}\}$, $V_{2,0} = \{v_{1,L}, v_{2,L}, \ldots, v_{b,L}\}$ and $V_{3,0} = \{z_{1,L}, z_{2,L}, \ldots, z_{b,L}\}$ can be obtained based on Equation (16).

$$\begin{cases} u_{i,L} = \lfloor M_0 \times l_{1,Lb+i-1} \rfloor \ mod\ 7 + 1 \\ v_{i,L} = \lfloor M_0 \times l_{2,Lb+i-1} \rfloor \ mod\ 7 + 1 \\ z_{i,L} = \lfloor M_0 \times l_{3,Lb+i-1} \rfloor \ mod\ 7 + 1 \end{cases} \tag{16}$$

where M_0 is a large positive integer and $l_{1,Lb+i-1}$, $l_{2,Lb+i-1}$ and $l_{3,Lb+i-1}$ are the $Lb + i - 1$th elements in the chaotic number sequences generated from Equation (15) with $l_{1,0}$, $l_{2,0}$ and $l_{3,0}$ as the initial values, respectively.

The cross-over operation is performed based on $V_{1,0}$, $V_{2,0}$ and $V_{3,0}$. In addition, three positive integer keys g_1, g_2 and g_3 are needed to determine the pairs of integers where the cross-over operations need to be performed in $V_{1,0}$, $V_{2,0}$ and $V_{3,0}$, respectively. g_1,g_2 and g_3 are all coprime with b.

To perform the cross-over operation on $E_{1,0}$, for each integer e that satisfies $1 \leq e \leq b$, $X_{e,L}$ is paired with $X_{a_1,L}$ for cross-over, where a_1 is determined by Equation (17).

$$a_1 = ((e-1)g1 + 2g_1 - 1) \ mod \ b + 1 \tag{17}$$

Based on $u_{e,L}$, two new integers $X'_{e,L}$ and $X'_{a,L}$ are generated by Equations (18) and (19).

$$X'_{e,L} = \lfloor X_{e,L}/2^{u_{e,L}} \rfloor + X_{a_1,L} \ mod \ 2^{u_{e,L}} \tag{18}$$

$$X'_{a,L} = \lfloor X_{a_1,L}/2^{u_{e,L}} \rfloor + X_{e,L} \ mod \ 2^{u_{e,L}} \tag{19}$$

As the result of the cross-over operation, $X_{e,L}$ is replaced by $X'_{e,L}$ and $X_{a_1,L}$ is replaced by $X'_{a_1,L}$.

The cross-over operation on $E_{2,0}$ is performed with a similar method. For each integer e that satisfies $1 \leq e \leq b$, a pair between $Y_{e,L}$ and $Y_{a_2,L}$ is formed for cross-over, and a_2 is obtained from Equation (20).

$$a_2 = ((e-1)g_2 + 2g_2 - 1) \ mod \ b + 1 \tag{20}$$

Two new integers $Y'_{e,L}$ and $Y'_{a_2,L}$ are obtained from $v_{e,L}$ by Equations (21) and (22).

$$Y'_{e,L} = \lfloor Y_{e,L}/2^{v_{e,L}} \rfloor + Y_{a_2,L} \ mod \ 2^{v_{e,L}} \tag{21}$$

$$Y'_{a_2,L} = \lfloor Y_{a_2,L}/2^{v_{e,L}} \rfloor + Y_{e,L} \ mod \ 2^{v_{e,L}} \tag{22}$$

The cross-over operation replaces $Y_{e,L}$ by $Y'_{e,L}$ and $Y_{a_2,L}$ is replaced by $Y'_{a_2,L}$.

Similarly, the cross-over operation on $E_{3,0}$ is performed on integer pairs formed based on g_3. For each integer e such that $1 \leq e \leq b$ holds, an integer pair is generated between $Z_{e,L}$ and $Z_{a_3,L}$ for cross-over, and a_3 is computed based on Equation (23).

$$a_3 = ((e-1)g_3 + 2g_3 - 1) \ mod \ b + 1 \tag{23}$$

Two new integers $Z'_{e,L}$ and $Z'_{a_3,L}$ are determined from $z_{e,L}$ by Equations (24) and (25).

$$Z'_{e,L} = \lfloor Z_{e,L}/2^{z_{e,L}} \rfloor + Z_{a_3,L} \ mod \ 2^{z_{e,L}} \tag{24}$$

$$Z'_{a_3,L} = \lfloor Z_{a_3,L}/2^{z_{e,L}} \rfloor + Z_{e,L} \ mod \ 2^{z_{e,L}} \tag{25}$$

The cross-over operation substitutes $Z_{e,L}$ and $Z_{a_3,L}$ with $Z'_{e,L}$ and $Z'_{a_3,L}$, respectively. Figure 5a shows the pseudocode for a cross-over operation.

```
for e = 1 to b
    a_1 = ((e − 1)g_1 + 2g_1 − 1) mod b + 1
    X'_{e,L} = ⌊X_{e,L}/2^{u_{e,L}}⌋ + X_{a_1,L} mod 2^{u_{e,L}}
    X'_{a,L} = ⌊X_{a_1,L}/2^{u_{e,L}}⌋ + X_{e,L} mod 2^{u_{e,L}}
    X_{e,L} = X'_{e,L}
    X_{a_1,L} = X'_{a_1,L}
    a_2 = ((e − 1)g_2 + 2g_2 − 1) mod b + 1
    Y'_{e,L} = ⌊Y_{e,L}/2^{v_{e,L}}⌋ + Y_{a_2,L} mod 2^{v_{e,L}}
    Y'_{a_2,L} = ⌊Y_{a_2,L}/2^{v_{e,L}}⌋ + Y_{e,L} mod 2^{v_{e,L}}
    Y_{e,L} = Y'_{e,L}
    Y_{a_2,L} = Y'_{a_2,L}
    a_3 = ((e − 1)g_3 + 2g_3 − 1) mod b + 1
    Z'_{e,L} = ⌊Z_{e,L}/2^{z_{e,L}}⌋ + Z_{a_3,L} mod 2^{z_{e,L}}
    Z'_{a_3,L} = ⌊Z_{a_3,L}/2^{z_{e,L}}⌋ + Z_{e,L} mod 2^{z_{e,L}}
    Z_{e,L} = Z'_{e,L}
    Z_{a_3,L} = Z'_{a_3,L}
end
```

(**a**)

```
for i = 1 to b
    m_{1,Lb+i−1} = 4m_{1,Lb+i−2}(1 − m_{1,Lb+i−2})
    m_{2,Lb+i−1} = 4m_{2,Lb+i−2}(1 − m_{2,Lb+i−2})
    m_{3,Lb+i−1} = 4m_{3,Lb+i−2}(1 − m_{3,Lb+i−2})
end
for i = 1 to b
    h_{i,L} = ⌊N_0 × m_{1,Lb+i−1}⌋ mod 7 + 1
    s_{i,L} = ⌊N_0 × m_{2,Lb+i−1}⌋ mod 7 + 1
    t_{i,L} = ⌊N_0 × m_{3,Lb+i−1}⌋ mod 7 + 1
end
for e = 1 to b
    X̃_{e,L} = ⌊X'_{e,L}/2^{h_{e,L}}⌋ + (X'_{e,L} mod 2^{h_{e,L}})2^{8−h_{e,L}}
    Ỹ_{e,L} = ⌊Y'_{e,L}/2^{s_{e,L}}⌋ + (Y'_{e,L} mod 2^{s_{e,L}})2^{8−s_{e,L}}
    Z̃_{e,L} = ⌊Z'_{e,L}/2^{t_{e,L}}⌋ + (Z'_{e,L} mod 2^{t_{e,L}})2^{8−t_{e,L}}
    X'_{e,L} = X̃_{e,L}
    Y'_{e,L} = Ỹ_{e,L}
    Z'_{e,L} = Z̃_{e,L}
end
```

(**b**)

Figure 5. Pseudocodes for the cross-over and mutation operations. (**a**) Cross-over operation; (**b**) mutation operation.

2.2.4. The Mutation Operation

Let $E_{1,c} = \left\{X'_{1,L}, X'_{2,L}, \ldots, X'_{b,L}\right\}$, $E_{2,c} = \left\{Y'_{1,L}, Y'_{2,L}, \ldots, Y'_{b,L}\right\}$ and $E_{3,c} = \left\{Z'_{1,L}, Z'_{2,L}, \ldots, Z'_{b,L}\right\}$ be the integer sets generated from $E_{1,0}$, $E_{2,0}$ and $E_{3,0}$ by a cross-over operation. A mutation operation performs a rotational right shift on each integer in $E_{1,c}$, $E_{2,c}$ and $E_{3,c}$. The number of bit positions by which the shift is performed for an integer is determined by a set of three one-dimensional logistic systems. Let $m_{1,0}$, $m_{2,0}$ and $m_{3,0}$ be the initial values of the three logistic systems; three sets of integers $M_{1,0} = \left\{h_{1,L}, h_{2,L}, \ldots, h_{b,L}\right\}$, $M_{2,0} = \left\{s_{1,L}, s_{2,L}, \ldots, s_{b,L}\right\}$ and $M_{3,0} = \left\{t_{1,L}, t_{2,L}, \ldots, t_{b,L}\right\}$ are generated based on Equation (26) for the pixel in row j and column k in $S(m, n, 3)$.

$$\begin{cases} h_{i,L} = \lfloor N_0 \times m_{1,Lb+i-1} \rfloor \bmod 7 + 1 \\ s_{i,L} = \lfloor N_0 \times m_{2,Lb+i-1} \rfloor \bmod 7 + 1 \\ t_{i,L} = \lfloor N_0 \times m_{3,Lb+i-1} \rfloor \bmod 7 + 1 \end{cases} \tag{26}$$

where N_0 is a large positive integer and $m_{1,Lb+i-1}$, $m_{2,Lb+i-1}$ and $m_{3,Lb+i-1}$ are the $Lb + i - 1$th elements in the chaotic number sequences obtained from Equation (15) using $m_{1,0}$, $m_{2,0}$ and $m_{3,0}$ as the initial values, respectively.

Integers in $M_{1,0}$, $M_{2,0}$ and $M_{3,0}$ provide the number of bit positions by which each integer in $E_{1,c}$, $E_{2,c}$ and $E_{3,c}$ needs to be rotationally shifted to complete the mutation operation. For each integer e such that $1 \le e \le b$ holds, $X'_{e,L}$, $Y'_{e,L}$ and $Z'_{e,L}$ are rotationally shifted to the right by $h_{e,L}$, $s_{e,L}$ and $t_{e,L}$ bit positions, respectively. In other words, $X'_{e,L}$, $Y'_{e,L}$ and $Z'_{e,L}$ are replaced by $\widetilde{X}_{e,L}$, $\widetilde{Y}_{e,L}$ and $\widetilde{Z}_{e,L}$ generated by Equations (27), (28) and (29), respectively. The resulting integer sets in the evolutionary system are denoted by $E_{1,f}$, $E_{2,f}$ and $E_{3,f}$. Figure 5b shows the pseudocode for a mutation operation.

$$\widetilde{X}_{e,L} = \lfloor X'_{e,L} / 2^{h_{e,L}} \rfloor + (X'_{e,L} \bmod 2^{h_{e,L}}) 2^{8-h_{e,L}} \tag{27}$$

$$\widetilde{Y}_{e,L} = \lfloor Y'_{e,L} / 2^{s_{e,L}} \rfloor + (Y'_{e,L} \bmod 2^{s_{e,L}}) 2^{8-s_{e,L}} \tag{28}$$

$$\widetilde{Z}_{e,L} = \lfloor Z'_{e,L} / 2^{t_{e,L}} \rfloor + (Z'_{e,L} \bmod 2^{t_{e,L}}) 2^{8-t_{e,L}} \tag{29}$$

2.2.5. Encoding of Pixels

An integer is selected from each of the three integer sets $E_{1,f} = \left\{\widetilde{X}_{1,L}, \widetilde{X}_{2,L}, \ldots, \widetilde{X}_{b,L}\right\}$, $E_{2,f} = \left\{\widetilde{Y}_{1,L}, \widetilde{Y}_{2,L}, \ldots, \widetilde{Y}_{b,L}\right\}$ and $E_{3,f} = \left\{\widetilde{Z}_{1,L}, \widetilde{Z}_{2,L}, \ldots, \widetilde{Z}_{b,L}\right\}$ to generate integer keys that can encode the pixel in row j and column k in $S(m, n, 3)$. The selection is controlled by a one-dimensional logistic system with an initial value of s_0; three integers $c_{1,L}$, $c_{2,L}$ and $c_{3,L}$ are obtained based on Equation (30) for the pixel in row j and column k in $S(m, n, 3)$.

$$\begin{cases} c_{1,L} = \lfloor N_1 \times s_{3L} \rfloor \bmod b + 1 \\ c_{2,L} = \lfloor N_1 \times s_{3L+1} \rfloor \bmod b + 1 \\ c_{3,L} = \lfloor N_1 \times s_{3L+2} \rfloor \bmod b + 1 \end{cases} \tag{30}$$

where N_1 is a large positive integer and s_{3L}, s_{3L+1} and s_{3L+2} are the $3l$th, $3l + 1$th and $3l + 2$th elements, respectively, in the chaotic number sequences obtained from Equation (15) using an initial value of s_0.

Let $C(m, n, 3)$ be the cipher image obtained from $S(m, n, 3)$; the integer keys used to encode $S(j, k, 1)$, $S(j, k, 2)$ and $S(j, k, 3)$ are denoted by $I_k(j, k, 1)$, $I_k(j, k, 2)$ and $I_k(j, k, 3)$, respectively. $I_k(j, k, 1)$, $I_k(j, k, 2)$ and $I_k(j, k, 3)$ are obtained based on Equation (31).

$$\begin{cases} I_k(j,k,1) = (H_{j,k} \widetilde{X}_{c_{1,L},L}) \bmod 256 \\ I_k(j,k,2) = (H_{j,k} \widetilde{Y}_{c_{2,L},L}) \bmod 256 \\ I_k(j,k,3) = (H_{j,k} \widetilde{X}_{c_{3,L},L}) \bmod 256 \end{cases} \tag{31}$$

where $H_{j,k}$ is 1 when $j = 1$ and $k = 1$; otherwise,$H_{j,k} = \sum_{v=1}^{3} C(j_p, k_p, v)$, where j_p and k_p can be computed from j and k with Equations (32) and (33).

$$j_p = \begin{cases} 1, & if\ j = 1\ and\ k = 2 \\ \left\lceil \frac{(j-1)n+k-2}{n} \right\rceil, & otherwise \end{cases} \tag{32}$$

$$k_p = ((j-1)n + k - 2)\ mod\ q + 1 \tag{33}$$

Finally, $C(i, j, 1)$, $C(i, j, 2)$ and $C(i, j, 3)$ are determined from Equation (34).

$$\begin{cases} C(j,k,1) = S(j,k,1) \oplus I_k(j,k,1) \\ C(j,k,2) = S(j,k,2) \oplus I_k(j,k,2) \\ C(j,k,2) = S(j,k,3) \oplus I_k(j,k,3) \end{cases} \tag{34}$$

Equations (31)–(33) show that the encryption keys $I_k(j,k,1)$, $I_k(j,k,2)$ and $I_k(j,k,3)$ are dependent on the encrypted image contents associated with the pixel in row j_p and column k_p. This fact suggests that the proposed approach is plain-image-sensitive. In practice, the security of encryption can be further enhanced by applying the encoding process multiple times.

2.2.6. Decoding a Cipher Image

The scrambled image of a cipher image can be obtained based on the initial values of the 3D chaotic systems used to construct the evolutionary system for encoding and the initial values of the one-dimensional logistic systems that control the evolutionary system during the encoding.

Let $C(m, h, 3)$ be a cipher image.$I_1 = \{\rho_{1,0}, \rho_{2,0}, \ldots, \rho_{b,0}\}$, $I_2 = \{\theta_{1,0}, \theta_{2,0}, \ldots, \theta_{b,0}\}$ and $I_3 = \{\varphi_{1,0}, \varphi_{2,0}, \ldots \varphi_{b,0}\}$ are the sets of initial values for the b3D chaotic systems;$l_{1,0}$, $l_{2,0}$ and $l_{3,0}$ are the initial values of the logistic systems for cross-over operations;$m_{1,0}$, $m_{2,0}$ and $m_{3,0}$ are the initial values of the logistic systems for mutation operations; and s_0 is the initial value of the logistic system that selects integers from the evolutionary system for computing the encoding keys.

To decode the pixel in row j and column k in $C(m, h, 3)$, Equations (13) and (14) are used to obtain three integer sets $E_{1,0} = \{X_{1,L}, X_{2,L}, \ldots, X_{b,L}\}$, $E_{2,0} = \{Y_{1,L}, Y_{2,L}, \ldots, Y_{b,L}\}$ and $E_{3,0} = \{Z_{1,L}, Z_{2,L}, \ldots, Z_{b,L}\}$ that together constitute the evolutionary system. A cross-over operation is then applied to $E_{1,0}$, $E_{2,0}$ and $E_{3,0}$ as described in Section 2.2.3 to obtain three integer sets $E_{1,c} = \left\{X'_{1,L}, X'_{2,L}, \ldots, X'_{b,L}\right\}$, $E_{2,c} = \left\{Y'_{1,L}, Y'_{2,L}, \ldots, Y'_{b,L}\right\}$ and $E_{3,c} = \left\{Z'_{1,L}, Z'_{2,L}, \ldots, Z'_{b,L}\right\}$. The operations described in Section 2.2.4 are applied to $E_{1,c}$, $E_{2,c}$ and $E_{3,c}$ to complete the mutation operation and three integer sets $E_{1,f} = \left\{\widetilde{X}_{1,L}, \widetilde{X}_{2,L}, \ldots, \widetilde{X}_{b,L}\right\}$, $E_{2,f} = \left\{\widetilde{Y}_{1,L}, \widetilde{Y}_{2,L}, \ldots, \widetilde{Y}_{b,L}\right\}$ and $E_{3,f} = \left\{\widetilde{Z}_{1,L}, \widetilde{Z}_{2,L}, \ldots, \widetilde{Z}_{b,L}\right\}$ are obtained as the result. Based on $E_{1,f}$, $E_{2,f}$ and $E_{3,f}$, Equations (30)–(33) are utilized to generate $I_k(j,k,1)$, $I_k(j,k,2)$ and $I_k(j,k,3)$, which are the encoding keys for the pixel. $S(i, j, 1)$, $S(i, j, 2)$ and $S(i, j, 3)$ are obtained based on Equation (35).

$$\begin{cases} S(j,k,1) = C(j,k,1) \oplus I_k(j,k,1) \\ S(j,k,2) = C(j,k,2) \oplus I_k(j,k,2) \\ S(j,k,2) = C(j,k,3) \oplus I_k(j,k,3) \end{cases} \tag{35}$$

2.3. Computational Complexity

For a plain color image that contains m rows and n columns, a virtual image for scrambling can be constructed in $O(mn)$ time. The bit-level scrambling based on rows and columns of a virtual image can be accomplished in $O(mn)$ time. The scrambling process of the proposed approach thus needs $O(mn)$ time.

The encoding process encodes each pixel in a scrambled image with an evolutionary system that contains $3b$ integers. A cross-over operation can be performed in $O(b)$ time and the computation time needed for a mutation operation is $O(b)$. The encoding of a scrambled image thus requires $O(bmn)$ time. Therefore, the computation time needed for the encryption of an image is $O(bmn)$.

In practice, it is often desirable to perform the scrambling and encoding processes multiple times. For an encryption process that scrambles a plain image for n_s times and encodes the scrambled image for n_e times, a total amount of $O((n_s + n_e b)mn)$ computation time is needed to complete the encryption.

3. Results

A computer program has been created to implement the proposed approach in MATLAB and its encryption security has been analyzed based on the cipher images generated for seven benchmark images and 100 color images selected from the BSD500 dataset [42]. In addition, a comparison of the proposed approach with several state-of-the-art encryption methods is performed based on a number of security measures. Other approaches tested for comparison are the methods proposed in [7,20,28,38,43–49]. In the testing, an evolutionary system that contains five 3D chaotic systems is constructed to encode pixels in a scrambled image. A value of 10^{12} is chosen for integers M, M_0, N_0 and N_1.

3.1. Brutal Force Attacks

The robustness of an encryption method against brutal force attacks can be evaluated based on the size of its key space. A larger key space usually implies stronger robustness against brutal force attacks. In the scrambling process of the proposed approach, each row or column in a virtual image is associated with an integer key for bit-level scrambling. Let Z_1 be the size of the integer set where a key for a row or column can be selected, for a virtual image with p rows and q columns, Z_1^{p+q} different combinations exist for the integer keys utilized in scrambling process. In the encoding process, Z_2^{3d+7} different combinations exist for the initial values of d 3D chaotic systems and seven one-dimensional logistic systems; Z_2 is the size of the set of real numbers where the initial value for a chaotic system can be selected.

The key space of the proposed approach is thus $Z_1^{p+q} Z_2^{3d+7}$. For a plain color image that contains at least 10^4 pixels, the value of $p + q$ is at least $2\sqrt{pq} \geq 200$. When five 3D chaotic systems are used for the encryption of such an image, the key space size is at least $Z_1^{200} Z_2^{22}$. In practice, both $Z_1 > 2^{10}$ and $Z_2 > 2^{60}$ hold, and the key space size for a cipher image of the image is thus at least 2^{3320}.

Table 1 shows the key space sizes of a few encryption methods, including the proposed approach and several other state-of-the-art approaches. It is evident from Table 1 that the proposed approach has a key space size larger than those of the other encryption methods and its robustness against brutal force attacks is thus higher than that of the other methods.

Table 1. The key space sizes of the proposed approach and several other encryption methods.

Methods	Size of Key Space
The proposed	2^{3320}
Ref. [28]	2^{2092}
Ref. [20]	2^{199}
Ref. [38]	2^{170}
Ref. [50]	2^{138}
Ref. [44]	2^{241}
Ref. [7]	2^{104}
Ref. [45]	2^{128}
Ref. [46]	2^{128}
Ref. [47]	2^{128}

3.2. Statistical Attacks

The proposed approach is applied to seven popular benchmark images for encryption and an analysis is performed on the obtained cipher images to evaluate the overall strength of the approach against potential statistical attacks. Five of the seven benchmark images, including Lena, Airplane, Fruits, Peppers and Baboon, have a size of 512 × 512 while the other two benchmark images, including Monarch and Girl, have a size of 768 × 512. The seven testing benchmark images are shown in Figure 6a–g.

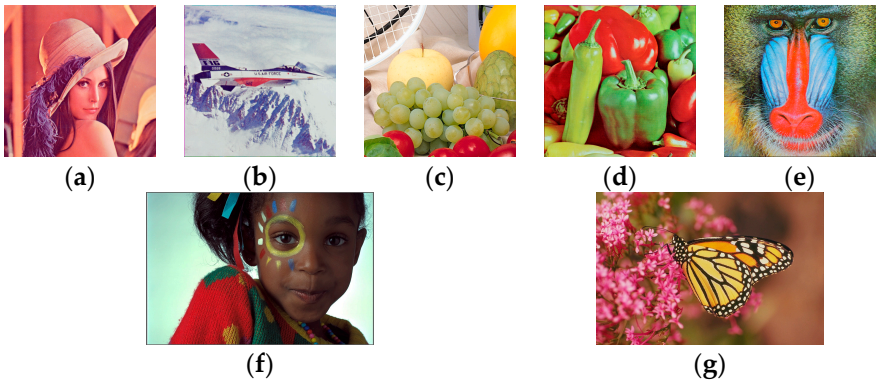


Figure 6. Benchmark images used for testing. (**a**) Lena; (**b**) Airplane; (**c**) Fruits; (**d**) Peppers; (**e**) Baboon; (**f**) Girl; (**g**) Monarch.

3.2.1. Analysis of Histograms

The cipher image obtained by the proposed approach for each benchmark image is shown in Figure 7a–g. It is clear from Figure 7 that the image contents in a plain image are completely removed from its cipher image and all cipher images appear to be highly random.

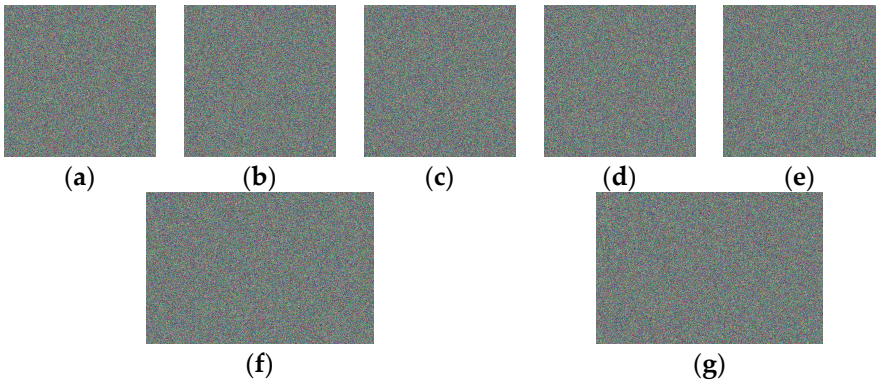
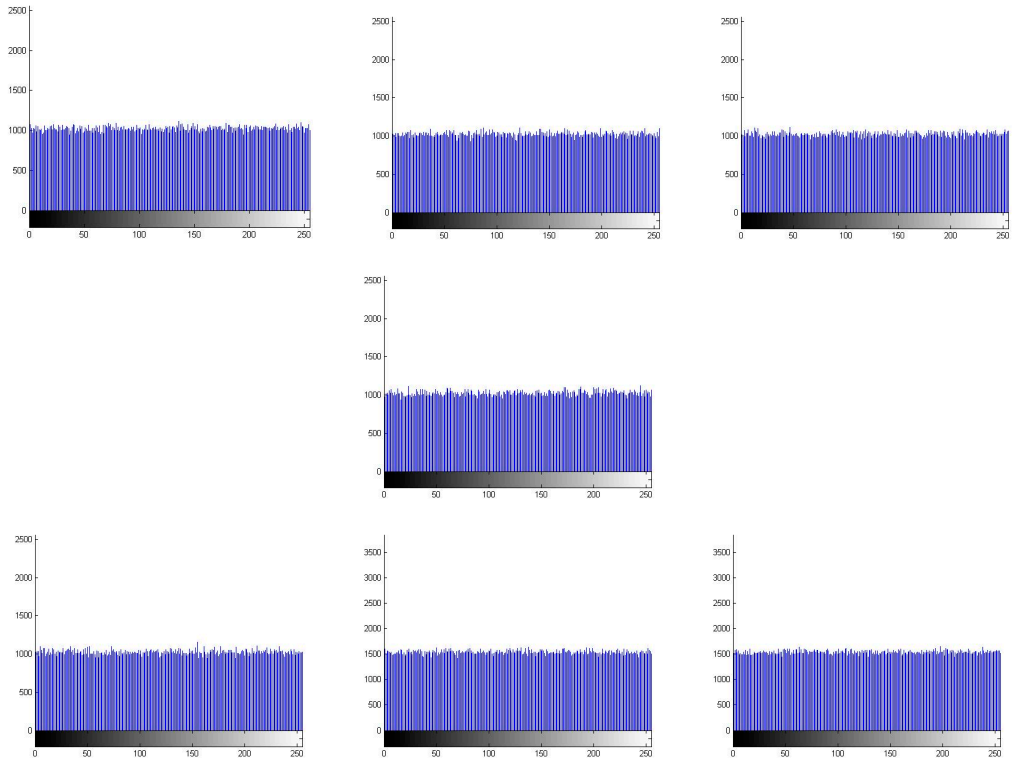
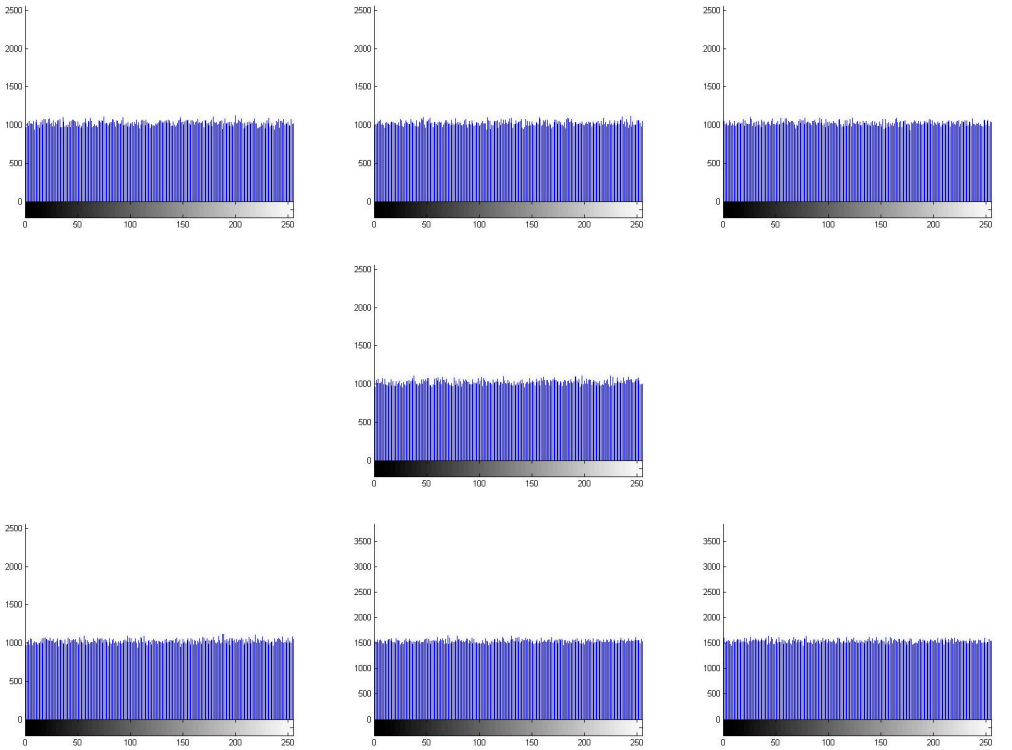


Figure 7. Cipher images obtained for benchmark images. (**a**) For Lena; (**b**) for Airplane; (**c**) for Fruits; (**d**) for Peppers; (**e**) for Baboon; (**f**) for Girl; (**g**) for Monarch.

The histograms of the R, G and B components in the cipher images of the benchmark images are shown in Figure 8a–c. It can be seen from Figure 8 that the R, G and B components in cipher images all follow near-uniform distributions and the histograms of a cipher image do not contain information related to the image contents in its plain image.

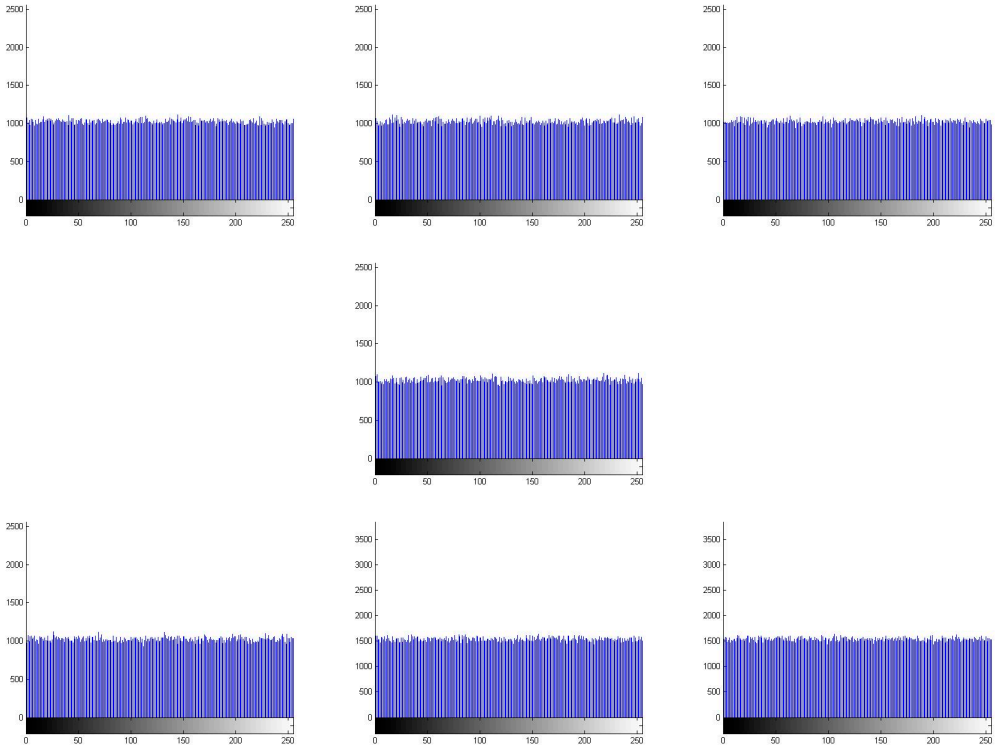


(**a**)



(**b**)

Figure 8. *Cont*.



(**c**)

Figure 8. Histograms of the R, G and B components in the cipher images. From **upper left** to **lower right**: Lena, Airplane, Fruits, Peppers, Baboon, Girl and Monarch; (**a**–**c**) show the histograms of the R, G and B components, respectively.

An important measure often used to evaluate the uniformity of a histogram is the variance of histograms [33]. Histogram P contains a counting value for each integer between 0 and 255. The variance of histogram $Var(P)$ of P is calculated by Equation (36).

$$Var(P) = \frac{1}{2n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (p_i - p_j)^2 \tag{36}$$

where $n = 256$ is the number of counting values in P. p_i and p_j are the counting values associated with integers i and j in P. Equation (36) clearly shows that histograms with lower values of variances of histograms generally are closer to a uniform distribution.

The variances of histograms for the cipher images are calculated and compared with those obtained with several other encryption methods, including methods proposed in [22,33,48,49]. The results of the comparison are shown in Table 2. Table 2 clearly shows that the proposed approach achieves the lowest value for variances of histograms on Airplane, Fruits and Monarch and it ranks the second position on Lena and Baboon. The results in Table 2 suggest that the overall performance of the proposed approach on the variances of histograms is better than that of the other methods.

Table 2. The variance of histograms for cipher images obtained with the proposed approach and several other encryption methods.

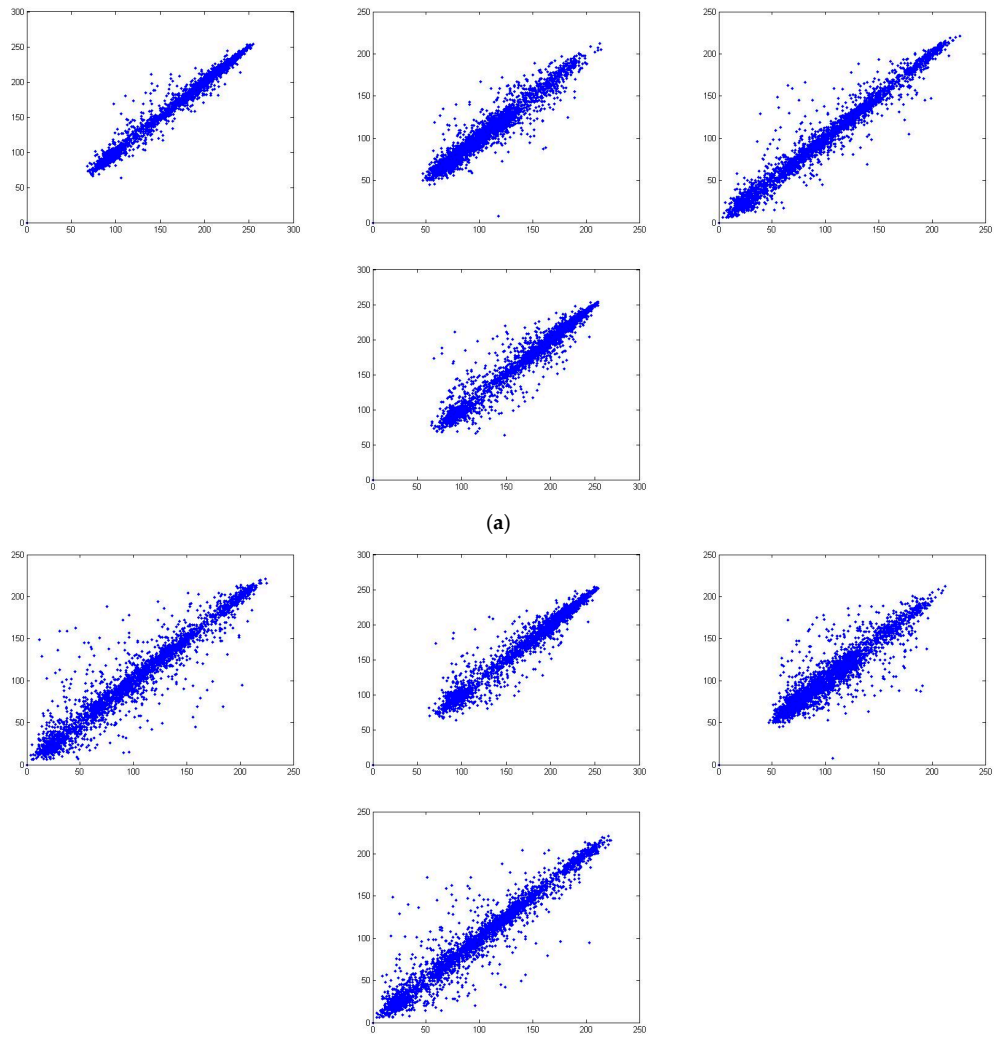
Images	The Proposed	Ref. [22]	Ref. [33]	Ref. [48]	Ref. [49]
Lena	1040.39	1047.40	1054.78	1043.21	1027.59
Airplane	1082.83	1132.27	1268.02	1141.38	1103.16
Fruits	951.94	1026.51	1034.23	1013.32	1082.34
Peppers	1146.23	1046.25	1037.78	1105.72	946.67

Table 2. *Cont.*

Images	The Proposed	Ref. [22]	Ref. [33]	Ref. [48]	Ref. [49]
Baboon	1024.29	1098.64	901.22	1061.04	1058.13
Girl	1536.50	1513.42	1543.23	1620.12	1530.21
Monarch	1409.30	1564.39	1607.51	1653.27	1563.72

3.2.2. Analysis of Correlations

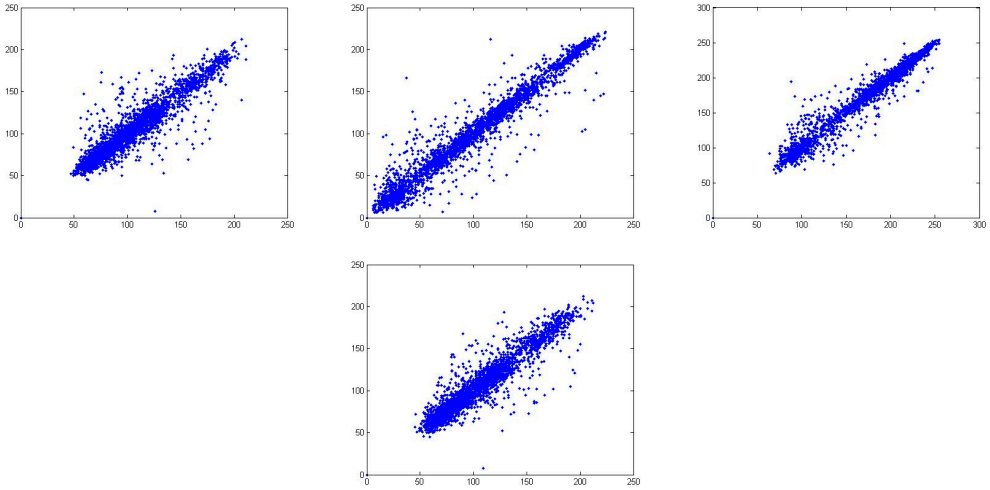
A well-known fact is that strong correlations usually exist between pixels that are spatially close in a plain color image. Such correlations are often closely associated with the contents contained in a plain image and thus need to be reduced to values close to zero when encryption is complete. In general, the correlations among pixels that are adjacent in horizontal, vertical, diagonal and anti-diagonal directions in a cipher image are used as important measures on its strength over potential statistical attacks. The correlations between adjacent pixels in the above four directions in Lena and its cipher image are plotted based on 3000 pixels randomly chosen from the images. Figure 9a–c show the plots obtained for the R, G and B components of pixels in Lena. Figure 10a–c show the plots obtained on its cipher image.



(**a**)

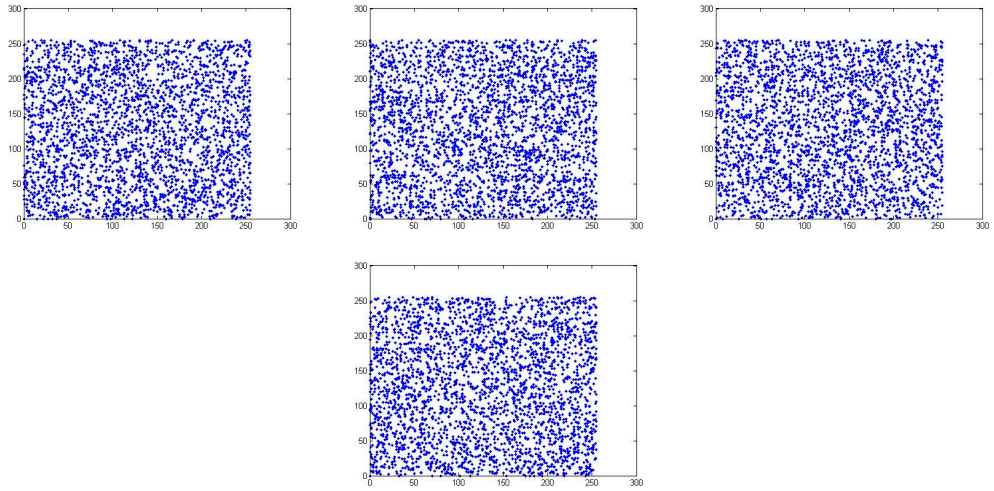
(**b**)

Figure 9. *Cont.*

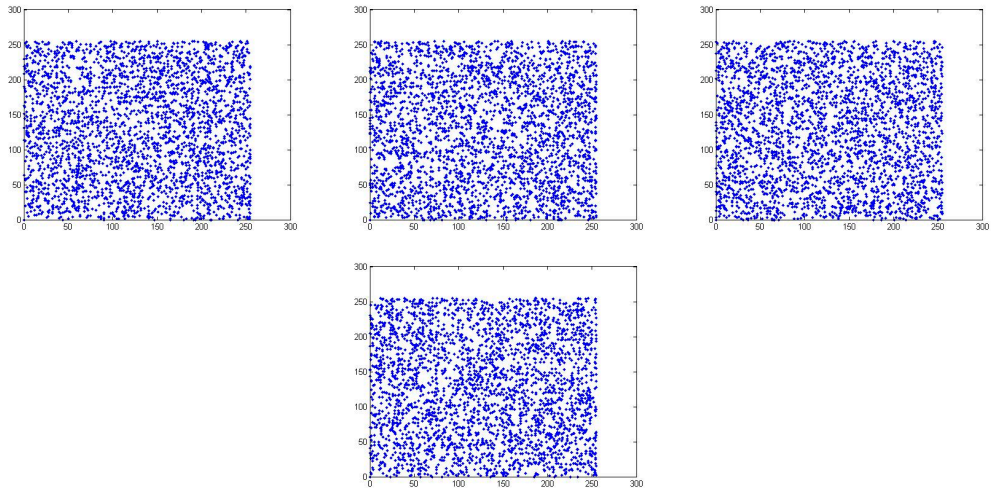


(**c**)

Figure 9. Correlations in the R, G and B components in Lena. From **left** to **right**: horizontal, vertical, diagonal and anti-diagonal; (**a**–**c**) show the correlations in the R, G and B components, respectively.

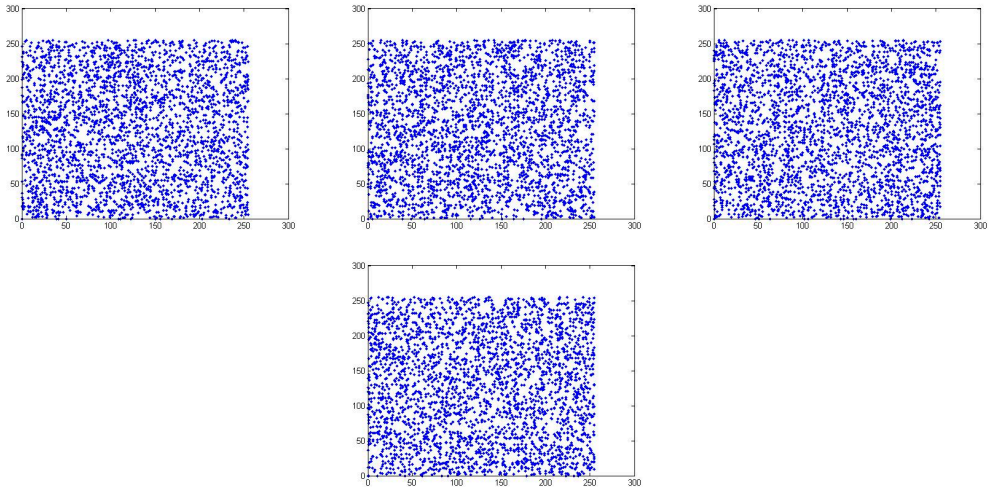


(**a**)



(**b**)

Figure 10. *Cont*.



(**c**)

Figure 10. Correlations in different directions for all components in the cipher image of Lena. From **left** to **right**: horizontal, vertical, diagonal and anti-diagonal; plots of correlations in R, G and B components are shown in (**a**–**c**), respectively.

The correlations for pixels adjacent in the four directions have been obtained for all benchmark images and their cipher images; these are shown in Table 3. Table 3 suggests that the correlations between pixels adjacent in a cipher image reach values close to zero. This fact ensures that cipher images generated with the proposed approach cannot be deciphered by statistical attacks. The correlations for the R components of pixels adjacent in cipher images obtained with several different methods on Lena are compared in Table 4. The results in Table 4 suggest that the proposed approach can achieve a performance comparable with other state-of-the-art encryption methods on eliminating local correlations in cipher images.

Table 3. The correlations for the R, Gand B components of pixels adjacent in four directions in the testing benchmark images and their cipher images obtained with the proposed approach.

Images	Directions	Plain Image			Cipher Image		
		R	G	B	R	G	B
Lena	V	0.9753	0.9666	0.9334	0.0019	−0.0013	−0.0006
	H	0.9853	0.9802	0.9558	0.0028	−0.0001	0.0022
	D	0.9734	0.9630	0.9264	−0.0011	0.0024	−0.0010
	A	0.9648	0.9536	0.9198	0.0002	0.0010	−0.0030
Airplane	V	0.9726	0.9578	0.9640	−0.0006	0.0017	−0.0001
	H	0.9568	0.9678	0.9353	0.0016	0.0011	−0.0013
	D	0.9343	0.9326	0.9146	−0.0002	0.0013	0.0034
	A	0.9350	0.9300	0.9075	−0.0013	0.0030	0.0020
Fruits	V	0.9936	0.9855	0.9265	−0.0007	−0.0025	−0.0030
	H	0.9928	0.9848	0.9192	−0.0022	0.0006	−0.0004
	D	0.9897	0.9783	0.8809	−0.0027	0.0013	−0.0020
	A	0.9868	0.9694	0.8531	0.0024	−0.0005	−0.0005
Peppers	V	0.9635	0.9811	0.9665	−0.0003	0.0023	0.0006
	H	0.9663	0.9817	0.9664	−0.0004	−0.0009	0.0021
	D	0.9563	0.9686	0.9477	−0.0009	0.0016	0.0022
	A	0.9585	0.9708	0.9477	0.0020	−0.0009	0.0034
Baboon	V	0.9235	0.8668	0.9067	0.0021	0.0006	0.0012
	H	0.8740	0.7759	0.8844	0.0011	−0.0002	−0.0029
	D	0.8649	0.7432	0.8544	0.0024	−0.0034	−0.0023
	A	0.8670	0.7494	0.8540	−0.0016	0.0037	0.0004

Table 3. *Cont.*

Images	Directions	Plain Image			Cipher Image		
		R	**G**	**B**	**R**	**G**	**B**
Girl	V	0.9811	0.9887	0.9866	−0.0019	−0.0012	−0.0004
	H	0.9901	0.9937	0.9927	0.0013	0.0014	−0.0002
	D	0.9750	0.9848	0.9823	0.0017	−0.0037	0.0024
	A	0.9772	0.9858	0.9832	−0.0019	0.0021	0.0035
Monarch	V	0.9648	0.9523	0.9566	1.7×10^{-5}	-4.5×10^{-5}	−0.0018
	H	0.9597	0.9453	0.9506	0.0020	−0.0018	0.0009
	D	0.9450	0.9252	0.9309	0.0016	0.0031	0.0014
	A	0.9245	0.8984	0.9118	0.0003	-8.7×10^{-6}	−0.0036

Table 4. The correlations for pixels adjacent in horizontal (H), vertical (V) and diagonal (D) directions in the cipher images of Lena obtained with the proposed approach and several other encryption methods. The best values are shown in bold.

Approaches	H	V	D
The proposed	0.0028	0.0019	−0.0011
Ref. [28]	0.0023	−0.0020	0.0013
Ref. [38]	0.0046	−0.0028	0.0014
Ref. [51]	0.0027	**−0.0013**	0.0039
Ref. [33]	**0.0012**	0.0035	0.0056
Ref. [20]	−0.0026	−0.0038	0.0017
Ref. [52]	−0.0030	0.0025	**−0.0001**
Ref. [22]	0.0021	0.0018	−0.0026

3.3. Differential Attacks

The strength of an encryption approach over differential attacks is generally evaluated by its sensitivities to tiny changes in encryption keys and plain image. The Number of Pixels Change Rate (NPCR) and unified average changing intensity (UACI) are two measures often used to evaluate such sensitivities [22,35,40,52].

Given a plain image I with m rows and n columns, a set of keys for encryption and the resulting cipher image C_{e_1},C_{e_2} is the cipher image generated after one of the keys in the key set, or one of the R, G and B components in one pixel in I is changed by a tiny amount. The NPCR and UACI for component t are calculated by Equations (37) and (38), respectively.

$$N(t) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} X(i,j,t)}{mn} \tag{37}$$

$$U(t) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} |C_{e_1}(i,j,t) - C_{e_2}(i,j,t)|}{255mn} \tag{38}$$

where $X(i,j,t)$ is an integer that depends on a comparison between $C_{e_1}(i,j,t)$ and $C_{e_2}(i,j,t)$, its value is 1 if $C_{e_1}(i,j,t)$ and $C_{e_2}(i,j,t)$ are different and is 0 otherwise. In an ideal case, NPCR has a value of 99.6094 and UACI has a value of 33.4635.

To evaluate the key sensitivity of the proposed approach, a perturbation of 10^{-16} is applied to the initial values of the chaotic systems used for the encoding process; the values of NPCR and UACI for the R, Gand B components are calculated with Equations (36) and (37) for each testing benchmark image. Table 5 shows the values of key sensitivity NPCR and UACI obtained for each testing benchmark image. It can be seen from Table 5 that, for all components, the values of key sensitivity NPCR and UACI obtained on all testing images are close to their ideal values.

Table 5. The values of NPCR and UACI in percentage for the key sensitivities of the proposed approach on the R, G and B components of cipher images. The best values are shown in bold.

Images	NPCR (%)			UACI (%)		
	R	G	B	R	G	B
Lena	**99.61**	99.60	99.63	33.45	33.42	33.51
Airplane	**99.61**	99.62	**99.61**	33.38	**33.47**	**33.46**
Fruits	99.62	99.62	99.60	33.38	33.49	33.45
Peppers	**99.61**	99.62	**99.61**	33.48	33.50	33.40
Baboon	**99.61**	**99.61**	99.63	33.55	33.38	33.49
Girl	**99.61**	**99.61**	**99.61**	**33.47**	33.45	33.42
Monarch	99.60	**99.61**	**99.61**	33.50	33.49	**33.46**

The results in Table 5 suggest that the mean key sensitivity NPCR of the proposed approach is larger than 99.6100 and its mean key sensitivity UACI is close to 33.4571. In [52], values of 99.5893, 99.5810 and 99.5717 are established for NPCR randomness tests at levels 0.05, 0.01 and 0.001, respectively; the proposed approach thus passes the NPCR randomness tests for all three levels. Similarly, in [52], lower bound values 33.3730, 33.3445 and 33.3115 are established along with upper bound values 33.5541, 33.5826 and 33.6156 for UACI randomness tests at levels 0.05, 0.01 and 0.001, respectively. Since the mean UACI of the proposed approach is close to 33.4571, it also passes the UACI randomness tests for all three levels.

Table 6 shows the values of key sensitivity NPCR and UACI obtained with the proposed approach and several other encryption methods on Lena. It is evident from Table 6 that the mean value of key sensitivity NPCR of the proposed approach is 99.6133 on Lena and its mean value of key sensitivity UACI is 33.4600. Since the ideal values for NPCR and UACI are 99.6064 and 33.4635, respectively, the proposed approach has the best overall performance on key sensitivities.

Table 6. The values of NPCR and UACI in percentage for the key sensitivities of the proposed approach and several other encryption methods on the R, G and B components of the cipher images of Lena. The best values are shown in bold.

Methods	NPCR (%)			UACI (%)		
	R	G	B	R	G	B
The proposed	**99.61**	99.60	99.63	33.45	33.42	33.51
Ref. [29]	99.62	99.62	99.62	33.48	33.45	33.50
Ref. [40]	**99.61**	**99.61**	**99.61**	**33.47**	**33.47**	**33.47**
Ref. [22]	99.57	99.58	99.57	33.35	33.37	33.38
Ref. [35]	99.60	99.59	99.61	33.45	33.45	33.45

To evaluate the plain image sensitivity of the proposed approach, a pixel is randomly selected from a plain image and one of its R, G and B components is changed by 1. The cipher image of the resulting image is then compared with that of the original image; the NPCR and UACI for each component can then be calculated based on Equations (36) and (37).

The NPCR and UACI values for plain image sensitivity of the proposed approach on the testing images are shown in Table 7. The proposed approach can achieve a mean NPCR of 99.6147 and a mean UACI of 33.4457. Due to the fact that values of 99.5893, 99.5810 and 99.5717 are associated with levels 0.05, 0.01 and 0.001, respectively, for NPCR randomness tests [52], the proposed approach is able to pass the NPCR randomness tests for all three levels. In addition, since the UACI randomness tests at levels 0.05, 0.01 and 0.001 adopt values 33.3730, 33.3445 and 33.3115 for lower bounds and values 33.5541, 33.5826 and 33.6156 for upper bounds, respectively [52], the proposed approach passes the random tests at all levels for UACI. Table 8 compares the plain image sensitivity values obtained with the proposed approach and those of several other encryption methods on Lena. Table 8 clearly

suggests that the proposed approach achieves a mean value of NPCR 99.6200 on Lena for plain image sensitivity and its mean value of plain image sensitivity UACI is 33.4567. The overall performance of the proposed approach is thus the best of all tested methods on plain image sensitivities.

Table 7. The values of NPCR and UACI in percentage for the plain image sensitivities of the proposed approach on the R, G and B components of cipher images. The best values are shown in bold.

Images	NPCR (%)			UACI (%)		
	R	G	B	R	G	B
Lena	99.62	99.62	99.62	33.47	33.43	**33.47**
Airplane	99.63	99.62	99.60	**33.46**	33.44	33.45
Fruits	**99.61**	**99.61**	**99.61**	33.45	33.44	33.48
Peppers	99.62	99.63	**99.61**	**33.46**	33.42	33.41
Baboon	**99.61**	**99.61**	99.63	33.39	33.43	**33.47**
Girl	99.60	99.62	**99.61**	33.42	33.39	33.51
Monarch	99.62	99.60	**99.61**	33.47	**33.45**	33.45

Table 8. The values of NPCR and UACI in percentage for the plain image sensitivities of the proposed approach and several other encryption methods on the R, G and B components of the cipher images of Lena. The best values are shown in bold.

Methods	NPCR (%)			UACI (%)		
	R	G	B	R	G	B
The proposed	99.62	99.62	99.62	**33.47**	33.43	**33.47**
Ref. [29]	**99.61**	99.63	**99.61**	33.45	**33.46**	33.45
Ref. [40]	99.60	99.58	99.59	33.44	33.43	33.43
Ref. [22]	99.58	99.57	99.58	33.34	33.34	33.34
Ref. [35]	99.59	**99.60**	99.59	33.33	33.33	33.33

3.4. Analysis of Information Entropy

Information entropy is a measure often utilized to represent the randomness of a cipher image. Specifically, let $C(m, n, 3)$ be a cipher image. Equation (39) is used to calculate the information entropy associated with the tth component of $C(m, n, 3)$.

$$E(C, t) = -\sum_{i=0}^{255} d(i, C, t) \log_2 d(i, C, t) \tag{39}$$

where $d(i, C, t)$ is the probability that the tth component of a pixel is i. Since a uniform distribution has an entropy of 8.0, the information entropy for each component in an ideally encrypted image is 8.0.

The information entropies of the R, G and B components in cipher images generated with the proposed approach are shown in Table 9. It can be seen from Table 9 that the information entropies for cipher images obtained by the proposed approach are all nearly ideal.

The information entropies of the cipher images generated with the proposed approach and several other encryption methods on Lena are shown in Table 10. Table 10 clearly suggests that the proposed approach achieves the highest entropies for components R and B and ranks the second position on component G. Its overall performance on Lena is thus the same as that of the approach proposed in [37] and better than that of the other tested methods.

Table 9. The information entropy for the R, G and B components of pixels in the cipher images of testing images obtained with the proposed approach. The best values are shown in bold.

Images	R	G	B
Lena	7.9993	7.9992	7.9994
Airplane	7.9993	7.9992	7.9992
Fruits	7.9993	7.9994	7.9994
Peppers	7.9992	7.9992	7.9992
Baboon	7.9992	7.9994	7.9993
Girl	7.9995	**7.9996**	7.9995
Monarch	**7.9996**	**7.9996**	**7.9996**

Table 10. The information entropy for the R, G and B components of pixels in the cipher images of Lena obtained with the proposed approach and several other encryption methods. The best values are shown in bold.

Methods	R	G	B
The proposed	**7.9993**	7.9992	**7.9994**
Ref. [28]	7.9992	7.9992	7.9992
Ref. [37]	7.9992	**7.9993**	**7.9994**
Ref. [22]	7.9992	7.9992	7.9992
Ref. [33]	7.9976	7.9976	7.9976

3.5. Analysis of Peak Noise Signal Ratio

The Peak Signal Noise Ratio (PSNR) provides a measure for the difference between a cipher image and its plain image. A higher PSNR value thus often suggests higher security for a cipher image [22]. The PSNR for a cipher image $C(m,n,3)$ is calculated based on its plain color image $I(m,n,3)$ with Equations (40) and (41).

$$MSE(I,C) = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{t=1}^{3}|I(i,j,t) - C(i,j,t)|^2 \tag{40}$$

$$PSNR(I,C) = 20\log_{10}\frac{255\sqrt{3mn}}{\sqrt{MSE(I,C)}} \tag{41}$$

The PSNR values of the cipher images generated by the proposed approach and several other encryption approaches on testing images are shown in Table 11. It is clear from Table 11 that the overall performance of the proposed approach on PSNR is comparable to that of the method in [28] and is higher than that of the other tested methods.

Table 11. The values of PSNR for the proposed approach and several other encryption methods on all testing images. The best values are shown in bold.

Images	The Proposed	Ref. [28]	Ref. [22]	Ref. [33]	Ref. [48]	Ref. [49]
Lena	**83.022**	83.021	81.131	80.927	81.027	81.225
Airplane	**83.022**	**83.022**	82.462	82.234	82.478	82.531
Fruits	83.022	**83.023**	81.023	80.835	81.147	80.632
Peppers	83.022	83.021	82.971	83.073	82.735	**83.105**
Baboon	83.022	**83.023**	82.873	83.145	82.652	82.534
Girl	**84.783**	84.782	84.653	84.641	84.437	84.685
Monarch	84.783	**84.784**	84.732	84.697	84.625	84.746

3.6. Experimental Results on General Color Images

In addition to benchmark color images, the proposed approach is applied to encrypt 100 color images selected from the BSD500 dataset [42]. All tested images are of size 481×321. Its overall performance on the encryption of these images is compared with that of the methods proposed in [22,28,33,38]. Table 12 shows the means and standard

deviations of the cipher images generated by the proposed approach and several other encryption methods. It is clear from Table 12 that the proposed approach outperforms the methods proposed in [22,33,38] on the variance of histograms. However, Table 12 suggests that the method in [28] is slightly better than the proposed approach on variances of histograms. Since the method in [28] optimizes the uniformity of the histograms of all components in cipher images with particle swarm optimization [53], it is not surprising that it can obtain cipher images with higher uniformity for histograms.

Table 12. The means and standard deviations of the variances of histograms for cipher images generated by the proposed approach and several other encryption methods. The best values are shown in bold.

Components		The Proposed	Ref. [28]	Ref. [38]	Ref. [22]	Ref. [33]
R	Mean	594.79	**583.19**	648.91	637.28	625.53
	STD	53.73	44.21	47.52	46.76	45.94
G	Mean	601.27	**586.32**	644.93	646.37	631.13
	STD	49.06	50.37	55.28	52.94	54.06
B	Mean	617.94	**607.25**	666.21	663.71	656.22
	STD	45.06	89.61	98.91	96.92	95.66

Table 13 compares the entropies of the cipher images obtained with the proposed approach and the other tested methods. Table 13 suggests that the proposed approach achieves the highest mean entropies for cipher images. Information on the PSNRs of the cipher images obtained with all tested methods is shown in Table 14. It can be seen from Table 14 that the proposed approach slightly outperforms all other methods in PSNR.

Table 13. The means and standard deviations of the information entropies for the R, G and B components of the cipher images obtained with the proposed approach and other tested encryption methods. The best values are shown in bold.

Methods	R		G		B	
	Mean	STD	Mean	STD	Mean	STD
The proposed	**7.9988**	0.0001	**7.9988**	0.0001	**7.9988**	0.0001
Ref. [28]	7.9961	0.0005	7.9962	0.0006	7.9962	0.0005
Ref. [38]	7.9952	0.0004	7.9953	0.0003	7.9954	0.0004
Ref. [22]	7.9947	0.0005	7.9982	0.0004	7.9952	0.0004
Ref. [33]	7.9935	0.0003	7.9936	0.0002	7.9935	0.0003

Table 14. The means and standard deviations of the PSNRs of the cipher images obtained with the proposed approach and other tested encryption methods. The best value is shown in bold.

Methods	PSNR	
	Mean	STD
The proposed	**80.7231**	0.0000
Ref. [28]	80.7229	0.0001
Ref. [38]	80.6343	0.0002
Ref. [22]	80.4352	0.0001
Ref. [33]	80.2396	0.0002

A comparison of the key sensitivities of the proposed approach with those of other tested methods is shown in Table 15. It is clear from Table 15 that the proposed approach can achieve mean values of NPCR and UACI closest to their ideal values. Table 16 compares the plain image sensitivities of all tested methods. The mean values of NPCR and UACI in Table 16 suggest that the overall plain image sensitivities of the proposed approach are slightly higher than those of the other tested methods.

Table 15. The means and standard deviations of key sensitivity NPCRs and UACIs in percentage for all tested methods on the R, G and B components of cipher images. The best values are shown in bold.

Methods		NPCR (%)			UACI (%)		
		R	G	B	R	G	B
The proposed	Mean	99.6025	**99.6110**	**99.6130**	**33.4700**	33.4835	**33.4555**
	STD	0.0152	0.0155	0.0130	0.0439	0.0589	0.0867
Ref. [28]	Mean	99.6233	99.6231	99.6232	33.4512	**33.4615**	33.4513
	STD	0.0221	0.0232	0.0124	0.1053	0.0733	0.1042
Ref. [38]	Mean	99.6002	99.6003	99.6006	33.4315	33.4326	33.4378
	STD	0.0204	0.0213	0.0193	0.1032	0.0985	0.0927
Ref. [22]	Mean	99.6203	99.6201	99.6202	33.4432	33.4428	33.4527
	STD	0.0225	0.0213	0.0204	0.0923	0.1121	0.1027
Ref. [33]	Mean	**99.6152**	99.6151	99.6153	33.4302	33.4301	33.4415
	STD	0.0227	0.0208	0.0183	0.0925	0.0834	0.1023

Table 16. The means and standard deviations of plain image sensitivity NPCRs and UACIs in percentage for all tested methods on the R, G and B components of cipher images. The best values are shown in bold.

Methods		NPCR (%)			UACI (%)		
		R	G	B	R	G	B
The proposed	Mean	**99.6033**	**99.6121**	**99.6107**	**33.4642**	33.4527	**33.4658**
	STD	0.0101	0.0098	0.0125	0.1017	0.1032	0.1073
Ref. [28]	Mean	99.6062	99.6053	99.6044	33.4573	**33.4582**	33.4535
	STD	0.0032	0.0011	0.0023	0.1124	0.1025	0.1327
Ref. [38]	Mean	99.5842	99.5837	99.5842	33.4435	33.4483	33.4216
	STD	0.0026	0.0034	0.0015	0.1127	0.1103	0.1114
Ref. [22]	Mean	99.5873	99.5852	99.5973	33.4592	33.4519	33.4612
	STD	0.0015	0.0026	0.0031	0.0833	0.1015	0.0954
Ref. [33]	Mean	99.5851	99.5842	99.5862	33.4532	33.4518	33.4526
	STD	0.0023	0.0021	0.0017	0.0903	0.0874	0.1217

The computational efficiency of each tested method is evaluated based on the computation time needed by the method to encrypt a color image. Table 17 shows information on the amount of computation time each method requires to generate a cipher image. Table 17 clearly shows that the computational efficiency of the proposed approach is comparable with that of the other methods.

Table 17. The means and standard deviations of the computation time each tested method needs to generate a cipher image. The best value is shown in bold.

Methods	Computation Time (in Seconds)	
	Mean	STD
The proposed	51.18	0.52
Ref. [28]	202.51	2.67
Ref. [38]	70.57	0.64
Ref. [22]	41.73	0.21
Ref. [33]	**35.68**	0.35

4. Discussion

Although the proposed approach has been tested on a few benchmark images and general color images, additional experimental results are needed to evaluate its overall performance on general color images. Moreover, the virtual image in the scrambling process is constructed based on a pixel-level mapping, which can partially reorder the pixels in the image before the bit-level scrambling operations are performed. However, virtual images constructed based on bit-level mappings can separate bits in the same pixel well apart before the scrambling starts; scrambled images generated based on such virtual images thus may contain less information on the contents of their plain images. In addition, more sophisticated operations can possibly be developed for the evolutionary system to further improve the security of cipher images.

It is clear from Section 2 that the proposed approach may require a large number of keys to generate a cipher image. The encryption of images with large sizes thus could be computationally inefficient. In addition, the randomness of the chaotic systems used in the proposed approach may need to be further improved to enhance the security of cipher images.

A multi-dimension discrete chaotic map with excellent ergodicity and randomness is proposed in [54] for image encryption. The proposed approach can probably be combined with the chaotic map proposed in [54] for further improvement in security. Moreover, continuous chaotic systems proposed in [55] can be used with parameter perturbation to enhance randomness.

The proposed approach utilizes a set of randomly selected fixed keys for encryption and the round keys are obtained with modular operations. The method proposed in [56] can possibly be utilized to eliminate the potential weaknesses in the key expansion method used in the proposed approach.

5. Conclusions

In this paper, a new approach is proposed for the encryption of color images. The approach performs encryption in two phases. In the first phase, binary bits in the R, G and B components of pixels in a plain color image are scrambled based on the rows and columns of a virtual image constructed from the plain image. In the second phase, an evolutionary system controlled by a set of chaotic systems is utilized to generate the integer keys needed to encode the R, G and B components of pixels in a scrambled image. An analysis on the size of key space and testing results suggest that cipher images generated by the proposed approach are secure against various types of attacks. In addition, comparisons of the proposed approach with several other state-of-the-art approaches on a variety of security measures show that its overall performance is better than that of the other tested encryption methods.

Author Contributions: Conceptualization, X.M. and Y.S.; methodology, Y.S.; software, X.M.; validation, X.M. and Y.S.; formal analysis, X.M.; investigation, Y.S.; resources, Y.S.; data curation, X.M.; writing—original draft preparation, X.M.; writing—review and editing, Y.S.; visualization, X.M.; supervision, Y.S.; project administration, Y.S.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The source code of the computer program developed in this study is available from the corresponding author upon request.

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

References

1. Kanso, A.; Ghebleh, M. An algorithm for encryption of secret images into meaningful images. *Opt. Lasers Eng.* **2017**, *90*, 196–208. [CrossRef]
2. Alvarez, G.; Li, S.J. Some basic cryptographic requirements for chaos-based cryptosystem. *Int. J. Bifurcat. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
3. Chen, G.; Mao, Y.; Chui, C.K. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos Solitons Fractals* **2004**, *21*, 749–761. [CrossRef]
4. Huang, C.K.; Nien, H.H. Multi chaotic systems based pixel shuffle for image encryption. *Opt. Commun.* **2009**, *282*, 2123–2127. [CrossRef]
5. Liu, Z.; Xu, L.; Lin, C.; Dai, J.; Liu, S. Image encryption scheme by using iterative random phase encoding in gyrator transform domains. *Opt. Lasers Eng.* **2011**, *49*, 542–546. [CrossRef]
6. Wang, Y.; Wong, K.W.; Liao, X.; Chen, G. A new chaos-based fast image encryption algorithm. *Appl. Soft Comput. J.* **2011**, *11*, 514–522. [CrossRef]
7. Zhang, G.; Liu, Q. A novel image encryption method based on total shuffling scheme. *Opt. Commun.* **2011**, *284*, 2775–2780. [CrossRef]
8. 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]
9. Guo, Q.; Liu, Z.; Liu, S. Color image encryption by using Arnold and discrete fractional random transforms in IHS space. *Opt. Lasers Eng.* **2010**, *48*, 1174–1181. [CrossRef]
10. Tao, R.; Meng, X.Y.; Wang, Y. Image encryption with multiorders of fractional fourier transforms. *IEEE Trans. Inf. Forensics Secur.* **2010**, *5*, 734–738. [CrossRef]
11. Wang, X.Y.; Yang, L.; Liu, R.; Kadir, A. A chaotic image encryption algorithm based on perceptron model. *Nonlinear Dyn.* **2010**, *62*, 615–621. [CrossRef]
12. Zhao, X.Y.; Chen, G. Ergodic matrix in image encryption. In Proceedings of the 2nd International Conference on Image and Graphics, Boston, MA, USA, 31 July 2002; Volume 4875, pp. 394–401.
13. Toughi, S.; Fathi, M.H.; Sekhavat, Y.A. An image encryption scheme based on elliptic curve pseudo random and advanced encryption system. *Signal Process.* **2017**, *141*, 217–227. [CrossRef]
14. Zunino, R. Fractal circuit layout for spatial decorrelation of images. *Electron. Lett.* **1998**, *34*, 1929–1930. [CrossRef]
15. Cao, W.; Zhou, Y.; Chen, C.L.P.; Xia, L. Medical image encryption using edge maps. *Signal Process.* **2017**, *132*, 96–109. [CrossRef]
16. Liu, Z.; Chen, H.; Liu, T.; Li, P.; Xu, L.; Dai, J.; Liu, S. Image encryption by using gyrator transform and Arnold transform. *J. Electron. Imaging* **1993**, *2*, 345–351. [CrossRef]
17. Mirzaei, M.Y.; Irani, H. A new image encryption method: Parallel sub-image encryption with hyper chaos. *Nonlinear Dyn.* **2012**, *67*, 557–566. [CrossRef]
18. Kumar, M.; Iqbal, A.; Kumar, P. A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie-Hellman cryptography. *Signal Process.* **2016**, *125*, 187–202. [CrossRef]
19. Zahid, A.H.; Al-Solami, E.; Ahmad, M. A Novel Modular Approach Based Substitution-Box Design for Image Encryption. *IEEE Access* **2020**, *8*, 150326–150340. [CrossRef]
20. Chen, J.; Zhu, Z.; Zhang, L.; Zhang, Y.; Yang, B. Exploiting self-adaptive permutation-diffusion and DNA random encoding for secure and efficient image encryption. *Signal Process.* **2018**, *142*, 340–353. [CrossRef]
21. Ye, G.; Pan, C.; Dong, Y.; Shi, Y.; Huang, X. Image encryption and hiding algorithm based on compressive sensing and random numbers insertion. *Signal Process.* **2020**, *172*, 107563. [CrossRef]
22. Jithin, K.C.; Sankar, S. Colour image encryption algorithm combining Arnold map, DNA sequence operation, and a Mandelbrot set. *J. Inf. Secur. Appl.* **2020**, *50*, 102428. [CrossRef]
23. Bao, W.; Zhu, C. A secure and robust image encryption algorithm based on compressive sensing and DNA coding. *Multim. Tools Appl.* **2022**, *81*, 15977–15996. [CrossRef]
24. Durdu, A. Nested Two-Layer RGB Based Reversible Image Steganography Method. *Inf. Technol. Control.* **2021**, *50*, 264–283. [CrossRef]
25. Liu, H.; Wang, X. Color image encryption based on one-time keys and robust chaotic maps. *Comput. Math. Appl.* **2010**, *59*, 3320–3327. [CrossRef]
26. Song, Y.; Song, J.; Qu, J. A secure image encryption algorithm based on multiple one-dimensional chaotic systems. In Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications, Chengdu, China, 14–17 October 2016; pp. 584–588.
27. Lu, Q.; Yu, L.; Zhu, C. Symmetric Image Encryption Algorithm Based on a New Product Trigonometric Chaotic Map. *Symmetry* **2022**, *14*, 373. [CrossRef]
28. Adeel, M.; Song, Y. Secure Encryption of Color Images with Chaotic Systems and Particle Swarm Optimization. *Iran J. Sci. Technol. Trans. Electr. Eng.* **2022**, *46*, 847–872. [CrossRef]
29. Sundara Krishnan, K.; Rajia, S.P.; Jaison, B. A Symmetric Key Multiple Color Image Cipher Based on Cellular Automata, Chaos Theory and Image Mixing. *Inf. Technol. Control* **2021**, *50*, 55–75. [CrossRef]

30. Ramasamy, P.; Ranganathan, V.; Kadry, S.; Damaševičius, R.; Blažauskas, T. An Image Encryption Scheme Based on Block Scrambling, Modified Zigzag Transformation and Key Generation Using Enhanced Logistic—Tent Map. *Entropy* **2019**, *21*, 656. [CrossRef]
31. Wang, X.Y.; Zhang, Y.Q.; Bao, X.M. A novel chaotic image encrypytion scheme using DNA sequence operations. *Opt. Lasers Eng.* **2015**, *73*, 53–61. [CrossRef]
32. Hu, T.; Liu, Y.; Gong, L.H.; Guo, S.F.; Yuan, H.M. Chaotic image cryptosystem using DNA deletion and DNA insertion. *Signal Process.* **2017**, *134*, 234–243. [CrossRef]
33. Liu, L.; Wang, D.; Lei, Y. An image encryption scheme based on hyper chaotic system and DNA with fixed secret keys. *IEEE Access* **2020**, *8*, 46400–46416. [CrossRef]
34. Zhang, Q.; Guo, L.; Wei, X. A novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Optik* **2013**, *124*, 3596–3600. [CrossRef]
35. Guesmi, R.; Farah, M.A.B.; Kachouri, A.; Samet, M. A novel chaos-based image encryption using DNA sequence operation and secure hash algorithm SHA-2. *Nonlinear Dyn.* **2016**, *83*, 1123–1136. [CrossRef]
36. Li, X.; Wang, L.; Yan, Y.; Liu, P. An improvement color image encryption algorithm based on DNA operations and real and complex chaotic systems. *Optik* **2016**, *127*, 2558–2565. [CrossRef]
37. Xian, Y.; Wang, X. Fractal sorting matrix and its application on chaotic image encryption. *Inf. Sci.* **2021**, *547*, 1154–1169. [CrossRef]
38. Cai, S.; Huang, L.; Chen, X.; Xiong, X. A symmetric plaintext-related color image encryption system based on bit permutation. *Entropy* **2018**, *20*, 282. [CrossRef]
39. Hua, Z.; Zhu, Z.; Yi, S.; Zhang, Z.; Huang, H. Cross-plane colour image encryption using a two-dimensional logistic tent modular map. *Inf. Sci.* **2021**, *546*, 1063–1083. [CrossRef]
40. Belazi, A.; EI-Latif, A.A.A.; Belghith, S. A novel image encryption scheme based on substitution-permutation network and chaos. *Signal Process.* **2016**, *128*, 155–170. [CrossRef]
41. Zhang, Y.Q.; Wang, X.Y.; Liu, J.; Chi, Z.L. An image encryption scheme based on the MLNCML system using DNA sequences. *Opt. Lasers Eng.* **2016**, *82*, 95–103. [CrossRef]
42. Arbelaez, P.; Maire, M.; Fowlkes, C.; Malik, J. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 898–916. [CrossRef]
43. Pak, C.; Huang, L. A new color image encryption using combination of the 1d chaotic map. *Signal Process.* **2017**, *138*, 129–137. [CrossRef]
44. 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]
45. Ganesan, K.; Murali, K. Image encryption using eight dimensional chaotic cat map. *Eur. Phys. J. Spec. Top.* **2014**, *223*, 1611–1622. [CrossRef]
46. Murillo-Escobar, M.A.; Cruz-Hernández, C.; Abundiz-Pérez, F.; López-Gutiérrez, R.M.; Campo, O.R.A.D. A RGB image encryption algorithm based on total plain image characteristics and chaos. *Signal Process.* **2015**, *109*, 119–131. [CrossRef]
47. 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]
48. Li, C.; Luo, G.; Qin, K.; Li, C. An image encryption scheme based on chaotic tent map. *Nonlinear Dyn.* **2017**, *87*, 127–133. [CrossRef]
49. Wu, J.; Liao, X.; Yang, B. Image encryption using 2D Hénon-sine map and DNA approach. *Signal Process.* **2018**, *153*, 11–23. [CrossRef]
50. Veeman, D.; Mehrabbeik, M.; Natiq, H.; Rajagopal, K.; Jafari, S.; Hussain, I. A New Chaotic System with Coexisting Attractors. *Int. J. Bifurc. Chaos* **2022**, *32*, 2230007. [CrossRef]
51. Edward, O. *Chaos in Dynamical Systems*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2003.
52. Yue, W.; Noonan, J.P.; Agaian, S. NPCR and UACI randomness tests for image encryption. *Cyber J.* **2011**, *2011*, 31–38.
53. Eberhart, R.C.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the 6th International Symposium on Micro Machine and Human Science, Nagoya, Japan, 13–16 March 1995; pp. 39–43.
54. Liu, H.; Kadir, A.; Xu, C. Color image encryption with cipher feedback and coupling chaotic map. *Int. J. Bifurc. Chaos* **2020**, *30*, 2050173. [CrossRef]
55. Liu, H.; Kadir, A.; Liu, J. Color pathological image encryption algorithm using arithmetic over galois field and coupled hyper chaotic system. *Opt. Lasers Eng.* **2019**, *122*, 123–133. [CrossRef]
56. Liu, H.; Wang, X.; Zhao, M.; Niu, Y. Constructing strong s-box by 2d chaotic map with application to irreversible parallel key expansion. *Int. J. Bifurc. Chaos* **2022**, *32*, 2250163. [CrossRef]