entropy

MDPI

Article

3C3R, an Image Encryption Algorithm Based on BBI, 2D-CA, and SM-DNA

Sajid Khan [1,*], Lansheng Han [2,*], Ghulam Mudassir [3], Bachira Guehguih [1] and Hidayat Ullah [4]

1 School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China; i201722242@hust.edu.cn
2 Faculty of Computer Science Department, Huazhong University of Science and Technology, Wuhan 430074, China
3 School of Information and Communication Technologies, University of L'Aquila, 67100 L'Aquila, Italy; ghulam.mudassir@graduate.univaq.it
4 School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China; hidayat@shu.edu.cn
* Correspondence: khansajid@hust.edu.cn (S.K.); hanlansheng@hust.edu.cn (L.H.); Tel.: +86-156-0862-0471 (S.K.); +86-189-7127-4240 (L.H.)

Received: 15 October 2019; Accepted: 30 October 2019; Published: 2 November 2019

Abstract: Color image encryption has enticed a lot of attention in recent years. Many authors proposed a chaotic system-based encryption algorithms for that purpose. However, due to the shortcomings of the low dimensional chaotic systems, similar rule structure for RGB channels, and the small keyspace, many of those were cryptanalyzed by chosen-plaintext or other well-known attacks. A Security vulnerability exists because of the same method being applied over the RGB channels. This paper aims to introduce a new three-channel three rules (3C3R) image encryption algorithm along with two novel mathematical models for DNA rule generator and bit inversion. A different rule structure was applied in the different RGB-channels. In the R-channel, a novel Block-based Bit Inversion (**BBI**) is introduced, in the G-channel Von-Neumann (VN) and Rotated Von-Neumann (RVN)- based 2D-cellular structure is applied. In the B-channel, a novel bidirectional State Machine-based DNA rule generator (**SM-DNA**) is introduced. Simulations and results show that the proposed **3C3R** encryption algorithm is robust against all well-known attacks particularly for the known-plaintext attacks, statistical attacks, brute-force attacks, differential attacks, and occlusion attacks, etc. Also, unlike earlier encryption algorithms, the **3C3R** has no security vulnerability.

Keywords: image encryption; DNA sequence; State Machine; bit inversion; entropy; cellular automata; security

1. Introduction

With the vast improvement of network and communication technology, communications were improved significantly. Communication for multimedia content, especially the image content over the Internet has become more and more chronic. Nevertheless, the security of the multimedia has a serious risk in the progression of communication because of the openness and distribution of the Internet. So the public has to take great care in terms of confidentiality and security of multimedia communication [1].

Among several protecting approaches, image encryption is one of the most efficient and frequent strategies for digital information protection. Now, greater and more new techniques are proposed for image encryption, which pursuits to reduce image content's redundancy by means of distinctive operations, such as DNA-based encryption operations [2] and chaos-based ciphers [3,4].

Many solo image encryption schemes were proposed by researchers to address the multiple-image encryption algorithms. The most widely used image encryption algorithms encompass chaotic-based image encryption [5–9], image encryption in the transform domain [10–13], DNA-based image encryption [14,15], and evolutionary-based image encryption [16].

The chaos system possesses a ramification of traits, which includes high sensitivity to initial conditions, determinacy, and ergodicity. Sequences produced by chaotic maps are habitually pseudo-random sequences, and their structures are very complicated and tough to be analyzed and foretold [17–19]. The typical ciphers primarily based on a chaotic map may be partitioned into two degrees: diffusion and permutation. In exercise, researchers frequently combine permutation and diffusion to get more computational security.

Because of the sensitivity and pseudo-randomness of the chaos, the chaotic sequences generated with the help of chaotic systems are intermittent and complicated. Researchers mostly encrypt the image in an early stage through the chaotic structures obtained by the low dimensional chaotic system, such as the one introduced in [20]. An image encryption technique with a new 1D chaotic system was proposed.

In [21] the author introduced 2D Logistic-Sine chaotic map-based image encryption algorithm. However, these sequences generated through low dimensional chaotic structures, and due to the inadequacy of the small keyspace cannot withstand against brute force attacks and thus possess the low security.

Conversely, Hyper-chaotic systems own larger keyspace and extra complicated dynamic features. Thus, the hyper-chaotic-based system can compensate for the shortcomings of a small keyspace of the low dimensional chaotic system and are more appropriate for the image encryption. In [22,23] the authors proposed a hyper-chaotic systems and fuzzy cellular automata-based novel image encryption algorithm that has higher security.

Sajid et al. [24] introduced a hybrid image encryption algorithm based on FSM and cellular automata accompanying DNA sequence. The algorithm has good results and also the concept of a local rule is appreciable with regards to algorithm efficiency. However, the shortcomings of the particular algorithm is its aimed for use only for grayscale images.

Huang et al. [25] proposed 7 dimensional CNN hyper-chaos-based application of image encryption scheme. Based on the results, the authors claimed that the hyper-chaotic systems are better than the low dimensional chaotic systems that own small keyspace. Astonishing information density, and the massive parallelism characteristics of DNA sequences, pushed the researchers to introduce collective hyper-chaos and DNA methods. Such collective technology schemes can be proved to be highly efficient and secure multimedia encryption schemes [26–28].

A new encryption algorithm for the color image was proposed in [29] that employs the hash-256 function to amend the control parameters and initial values of the chaotic system. The red, green and the blue channels of the image arranged into vector array form of one-dimensional. Then by rendering to the chaotic sequence generated by Piecewise Linear Chaotic Map 1-D vector array get sorted.

Nonetheless, in the utmost chaotic-based image encryption schemes that were stated above, the permutation phase and diffusion phase are autonomous of the plain image. Because the cryptosystem is impervious to the plain images and secret keys, such structures have the security flaws and cannot resist chosen/known-plaintext attacks or differential attacks. Table 1, listed some well known encryption algorithms that get successfully Cryptanalysis by the enlisted attack approach.

Therefore, after the analysis of above-mentioned papers, we proposed a novel three channel three rule (3C3R) color image encryption algorithm. We made three contributions to this paper with each having corresponding merits.

Firstly, for the red channel: a new block bit inversion (BBI) model was proposed. In every row simultaneous right and left two-bits selection is random and relies upon the integer value of a particular pixel and the block multiplier. The left and right direction bit selection efficiently lessen the pixel correlation among the plain and ciphered image.

Table 1. Algorithms that successfully get cryptanalyzed by well-know attack approaches.

Proposed by	Cryptanalysis by	Attack Approach
Zhang et al. (2013) [30]	Hoang et al. (2018) [31]	Chosen cipher-text
Zhou et al. (2015) [32]	Chen et al. (2017) [33]	Differential attack
Zhang et al. (2016) [34]	Wu et al. (2018) [35]	Chosen plaintext
Huang et al. (2012) [36]	Wang et al. (2014) [37]	Chosen plaintext
Chen et al. (2015) [38], Gao et al. (2008) [39]	Hu et al. (2017) [40], Rhouma et al. (2008) [41]	Chosen plaintext and cipher-text
Liu et al. (2016) [42], Tong et al. (2008) [43]	Zhang et al. (2017) [44], Li et al. (2009) [45]	Chosen plaintext
Zhu (2012) [46], Pak et al. (2017) [47]	Li et al. (2013) [48], Wang et al. (2018) [49]	Chosen plaintext

Secondly, for the green channel: cellular automata-based confined rules structure with Von-Neumann (VN) and Rotated Von-Neumann (RVN) structure was applied. Each sub-matrix gets different structural rule based on the particular bit values. Except for the fact that in the rule selection every sub-matrix has a direct relation with the previous one.

Thirdly, for the blue channel: a novel mathematical model of bidirectional State Machine (SM)-based DNA rule generator (SM-DNA) was proposed. The proposed model efficiently generates a random rule for each block-matrix. In each block-matrix i^{th} and j^{th} bit used as an input for the rule selection of next block-matrix. That means the bit arrangement of every predecessor block-matrix is responsible for the rule selection of the next block-matrix. The remainder of the paper organized as comply with; in Section 2: Literature survey/Preliminary work, Section 3: Proposed Image encryption scheme. Section 4: Experimental parameters and discussion while Section 5: Security analysis with the following Section 6: Conclusion and future work.

2. Literature Survey

A binary sequence $bin_{(n-1)}, bin_{(n-2)}, ..., bin_{(1)}, bin_{(0)}$ can be used to denote a non-negative decimal number (DN) by the following equation.

$$DN = \sum_{i=0}^{n-1} bin_{(i)} 2^i = bin_{(0)} 2^0 + bin_{(1)} 2^1 + ... + bin_{(n-1)} 2^{n-1} \quad (1)$$

As in any image, the pixel values of each channel are the non-negative decimal numbers between 0 and 255, thus every pixel can be denoted by the binary sequence of 1-byte or 8-bits. Similarly, the whole image can also be decomposed into binary 8-bit-planes [50]. In such case, the i^{th} bit-plane will comprise of all the i^{th} bits of the binary demonstration of every pixel. Among these $2^3 = 8$ bit-planes, most left bit-plane contains the highest significant visual information of the plain image, while the right most bit-plane contains the least visual information.

2.1. Scrambling Method

Scrambling is an easy and effective technique to lessen the correlation between the neighboring pixels. This paper introduced a one-to-one mapping for scrambling. Due to simultaneous changes of pixels in columns and rows, the proficiency of the algorithm enhanced enormously concerning time. The proposed algorithm has a one-to-one ratio between the pixels of scrambled image and the plain image, as displayed in Figure 1.

The pseudo-code of the scrambling method for Encryption process is described in Algorithm 1, while in the decryption process the same procedure is applied but using reverse shifting, as described in Algorithm 2.

Algorithm 1 Scrambling

Encryption Process:
Input: Generate $S(i,j)$ matrix equal to the size of Plain image matrix $P(M,N)$.
Output: $T(m,n)$ size Scrambled matrix.

Suppose for **M**= 4 and **N**= 4;

$S \in \mathbb{Z}^{(M\times N)}$; $P \in \mathbb{Z}^{(M\times N)}$; $T \in \mathbb{Z}^{(M\times N)}$;

A = Array sequence $[3,2,1,4]$;

B = Array sequence $[4,2,1,3]$;

$I = A.index(i)$ for i in $Sorted[A]$;

$J = B.index(j)$ for j in $Sorted[B]$;

$$P_{pixels} = \begin{bmatrix} 101 & 103 & 182 & 14 \\ 83 & 201 & 90 & 20 \\ 92 & 49 & 100 & 34 \\ 223 & 92 & 140 & 10 \end{bmatrix}$$

$img = np.zeros((4,4))$;
$img[0,:] = [101,103,182,14]$;
$img[1,:] = [83,201,90,20]$;
$img[2,:] = [92,49,100,34]$;
$img[3,:] = [223,92,140,10]$;
S = $np.zeros((M,N))$;
for i **in range** $(0,M)$; **do**
  for j **in range** $(0,N)$; **do**
    m = $[(j - I(i) - 1) mod N] + 1$
    $S[i,m-1] = J[j]$
    $S = S + 1$
    S
  end for
end for
T = $np.zeros((4,4))$
for j **in range** $(0,N)$; **do**
  $data$ = [ ];
  for i **in range** $(0,M)$; **do**
    data.append.img $[I, S(I,j) - 1]$
    a = shift $[data, (N/2) - S(0,j)]$
  end for
end for
for k **in range** 1 to N; **do**
  $T[k, S[k,j]] = a[k]$;
end for

$$\mathrm{S} = \begin{bmatrix} 1 & 3 & 2 & 4 \\ 4 & 1 & 3 & 2 \\ 2 & 4 & 1 & 3 \\ 3 & 2 & 4 & 1 \end{bmatrix} ; \overrightarrow{Scr.(T)} = \begin{bmatrix} 140 & 103 & 83 & 100 \\ 34 & 223 & 90 & 101 \\ 92 & 20 & 14 & 92 \\ 201 & 182 & 49 & 10 \end{bmatrix}$$

Algorithm 2 Scrambling

Decryption Process:
Input: Random matrix $S(i,j)$ and Cipher image matrix $C(M,N)$.
Output: Descrambled matrix **P**.
for j **in range** $(0,N)$; **do**
  $data$ = [ ];
  for i **in range** $(0,M)$; **do**
    data.append.img $[I, S(I,j)-1]$
    a = shift $[data, (N/2)-S(0,j)]$
  end for
end for
for k **in range** 1 to N; **do**
  $P[k, S[k,j]] \mapsto a[k]$;
end for

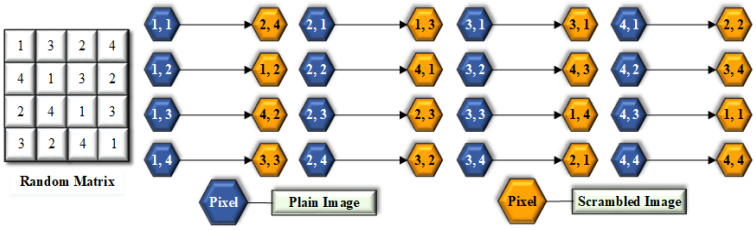


Figure 1. Scrambling one-to-one mapping.

2.2. Novel Block Bit Inversion (BBI)

In this paper, we propose a novel blocked-based bit inversion technique to alter the pixel values. The Figure 2 is the graphical illustration of the proposed **BBI** structure for the image sub-matrix of the size $M \times N$. Firstly, it divides the particular block matrix into two equal column blocks; labelled as Left bit Columns Blocks (LCB) and Right bit Columns Blocks (RCB). The purpose is to change the bits in every row and columns but in the opposite manner. It changes the particular bit **1** or **0** into opposite i.e., **0** if it is **1** or else **1** if it is **0**. For each pixel, the change in value occurs (either addition or subtraction) depending upon either its get converted into **1** from **0** or get converted to **0** from **1**. As each bit has a unique value, so because of bit inversion following change occurs in pixels based on the particular bit location.

$$MSB \leftrightarrow 8^{th}bit \mapsto 2^{8-1} = 2^7 = \pm 128;\ 7^{th}bit \mapsto 2^{8-2} = 2^6 = \pm 64$$

$$6^{th}bit \mapsto 2^{8-3} = 2^5 = \pm 32;\ 5^{th}bit \mapsto 2^{8-4} = 2^4 = \pm 16$$

$$4^{th}bit \mapsto 2^{8-5} = 2^3 = \pm 8;\ 3^{rd}bit \mapsto 2^{8-6} = 2^2 = \pm 4$$

$$2^{nd}bit \mapsto 2^{8-7} = 2^1 = \pm 2;\ LSB \leftrightarrow 1^{st}bit \mapsto 2^{8-8} = 2^0 = \pm 1$$

For example, we can see in Figure 2 that for each binary blocks the bit selection is in opposite direction means $LCB_{bit} \rightleftharpoons RCB_{bit}$ for left and right columns blocks. For the left columns block the selection of bit is from left to right $LCB_{bit} \Longrightarrow RCB_{bit}$ and for the right columns block the selection of bit is from right to left $LCB_{bit} \Longleftarrow RCB_{bit}$. How its locate; is illustrated in Figure 3. Like in main diagram for the 1^{st} row, 3^{rd} bit is selected for the inversion in left columns block which is actually 6^{th} while counting from right to left. In same passion, for the right columns block, 3^{rd} bit is selected that is actually 6^{th} when counting from left to right. So selection is the same for 3^{rd} bit but direction is opposite as $Left \rightleftharpoons Right$. Thus, to select the particular bit in the novel BBI model, the formula efficiently locate the right and left block bits by proceeding only by the left to the right direction. The starting bit selection for each 8×8 blocks is based on the following formula.

$$LCB_{bit}^{n} = [(n^2 \times PmodB + 1) + R_i]modB + 1 \quad (2)$$

Here $i = 1, 2, 3, \ldots, 8$, where P is the pixel value of particular pixel, n is the block number multiplier to change the 1^{st} bit for each block, B is the byte (8 bits), mod is the mod function that returns the remainder between **0** to **7** because of B is one byte (8-bits), and R_i is the particular row number range from **1** to **8**.

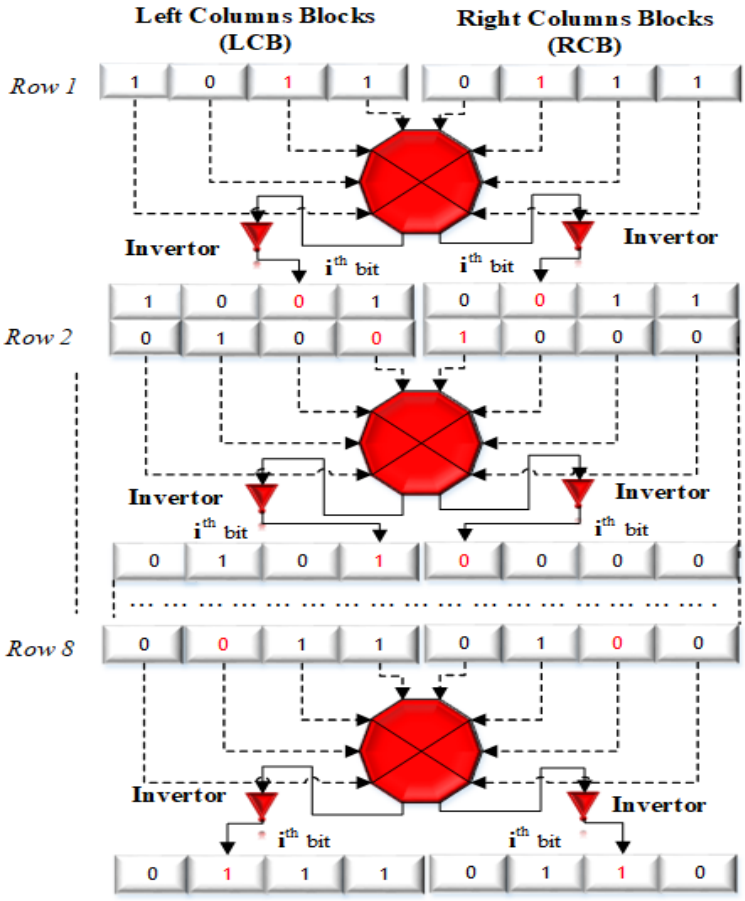


Figure 2. Block bit Inversion.

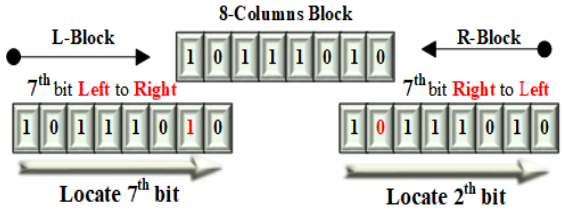


Figure 3. Bit selection.

For better understanding, Let suppose the decimal value of **P** for the 1^{st} sub-matrix came **183**, so the value of left columns block (LCB_{bit}) for the $\mathbf{1^{st}}$ row $i = 1$ or $R_i = 1$ can be gotten as follows.

$$LCB_{bit}^{n} = [(n^2 \times PmodB + 1) + R_i]modB + 1$$

$$LCB_{bit}^{1} = [(1 \times 183mod8 + 1) + 1]mod8 + 1$$

$$\Rightarrow [7 + 1 + 1]mod8 + 1 = 1 + 1 = 2^{nd}bit$$

Likewise, for the 2^{nd} row of this 1^{st} sub-block matrix, $i = 2$ or $R_i = 2$ the inverting bit will be;

$$LCB^1_{bit} = [(1 \times 183 mod 8 + 1) + 2] mod 8 + 1$$

$$\Rightarrow [7 + 1 + 2] mod 8 + 1 = 2 + 1 = 3^{rd} bit$$

Similarly, for the 3^{rd}, 4^{th}, 5^{th} to up to 8^{th} rows for every $LCB^{n^{th}}_{bit}$, desired inverting bit can be gotten through the formula. Alternatively, the formula of the bit selection for the right blocks is as follows.

$$RCB^n_{bit} = [B - LCB^n_{bit}] mod B + 1 \tag{3}$$

As an example, in the above case after putting values.

$$RCB^1_{bit} = [8 - 2] mod B + 1 \Rightarrow 6 mod 8 + 1 = 7^{th} bit$$

Similarly, for the 2^{nd} row, the bit will be following.

$$RCB^1_{bit} = [8 - 3] mod B + 1 \Rightarrow 5 mod 8 + 1 = 6^{th} bit$$

First of all, an initial configuration matrix of 8×8 is generated by the hash value of the key. Based on the above-mentioned formula of RCB_{bit} and LCB_{bit}, bit inversion is done over the initial configuration matrix (ICM^t_i) and later this updated initial configuration matrix (ICM^{t+1}_i) is XORed with the first plain matrix. The general equation of XORing the updated (ICM^{t+1}_i) with the plain image matrices is as follows.

$$SM^{t+1}_{n(i,j)} = \begin{cases} SM^t_1 \oplus ICM^{t+1}_0 & n = 1 \\ SM^t_n \oplus ICM^{t+1}_{n-1} & n > 1 \end{cases} \tag{4}$$

where $n = 1, 2, 3, \ldots, N$; where N is the entire number of sub-matrices of the red channel. While SM^{t+1}_i is the particular updated matrix, SM^t_i is the plain image sub-matrix of the red channel, and ICM^{t+1}_i is the updated initial configuration matrix. Similarly, for the 2^{nd} sub-matrix bit inversion is performed over the updated initial configuration matrix of 1^{st} block stage and then XORed with the second sub-matrix of the plain image as follows.

$$SM^{t+1}_2 = SM^t_2 \oplus ICM^{t+1}_1 \tag{5}$$

For the decryption phase, the reverse process to get back the plain image matrix is as follows.

$$SM^t_{n(i,j)} = \begin{cases} SM^{t+1}_1 \oplus ICM^{t+1}_0 & n = 1 \\ SM^{t+1}_n \oplus ICM^{t+1}_{n-1} & n > 1 \end{cases} \tag{6}$$

where $n = 1, 2, 3, \ldots, N$ and N is the total number of sub-matrices.

2.3. Two-Dimensional Cellular Automata

In this paper, a 2D-CA with VN and RVN structure was implemented as shown in Figure 4. Also, in cellular matrix form, every cell can have one of the two possible states that is either **1** or **0**. In cellular representation every cell has its neighboring cells. The cell with VN neighbors or a radius equivalent to **1** can be represented in equation form as follow:

$$C^{t+1}_{vn} = \delta(C^t_{i,j+1} : C^t_{i-1,j} : C^t_{i,j} : C^t_{i+1,j} : C^t_{i,j-1}) \tag{7}$$

In this Equation (7), δ is the Boolean function or a state transition function that takes to the new state, where $\delta : C \times \sum \mapsto \rho(C)$. Like VN, the cell with Rotated VN neighbors can be represented as follow.

$$C_{rvn}^{t+1} = \delta(C_{i-1,j+1}^{t} : C_{i+1,j+1}^{t} : C_{i,j}^{t} : C_{i-1,j-1}^{t} : C_{i+1,j-1}^{t}) \tag{8}$$

So, to get the combined structure of VN and RVN, We merged these two equations as follows.

$$C_{M(i,j)}^{t+1} = \delta(C_{i-1,j+1}^{t} : C_{i+1,j+1}^{t} : C_{i+1,j}^{t} : C_{i,j+1}^{t} : C_{i,j}^{t} : C_{i-1,j+1} : C_{i-1,j-1} : C_{i+1,j-1} : C_{i+1,j+1}) \tag{9}$$

We represent these **9** state variable with $C_{NW} : C_N : C_{NE} : C_W : C_0 : C_E : C_{SW} : C_S : C_{SE}$, as shown in Table 2. For the green channel as the bit depth or intensity is $\mathbf{2^8}$ so, we mapped the above mentioned **8** state variables with these eight bits as $C_{NW}; C_N; C_{NE}; C_W; C_E C_{SW}; C_S; C_{SE}$ = [1; 1; 1; 1; 1; 1; 1; 1], while excluding the central state variable $C_{0(0,0)}^{t}$ represented as C_0.

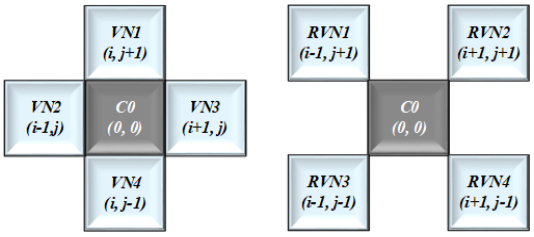


Figure 4. Von-Neumann (VN) + Rotated VN.

Finally, the confined rule structure is based on the following equation.

$$\begin{aligned} C_{M(i,j)}^{t+1} = {} & (C_{NW} \times C_{i-1,j+1}^{t}) \oplus (C_N \times C_{i,j+1}^{t}) \oplus (C_{NE} \times C_{i+1,j+1}^{t}) \oplus (C_E \times C_{i-1,j}^{t}) \oplus (C_3 \times C_{i+1,j}^{t}) \oplus \\ & (C_{SW} \times C_{i-1,j-1}) \oplus (C_S \times C_{i,j-1}) \oplus (C_{SE} \times C_{i+1,j-1}) \end{aligned} \tag{10}$$

In this updated form of Equation (10); C_{NW}, C_N, C_{NE}, C_W, C_E, C_{SW}, C_S, and C_{SE} are the state variables that can have the values either **1** or **0**. Therefore, the particular state variable will contribute in the XOR operation or state update process only if its respective bit will be equivalent to **1**, otherwise it will not take part in the state update process.

Table 2. VN + RVN Mapped Cell.

C_{NW}	C_N	C_{NE}
C_W	C_0	C_E
C_{SW}	C_S	C_{SE}

Thus, the structural combination of above direction variables based on the binary value of decimal number is employed to elect which and whom cells/cell will take part in the state update process. Hence different binary representation means different confined rule equation for the matrix update process of each succeeding matrix. The general formula of the confined rules is given below.

$$CA_{CR} = \begin{cases} [2(ICM_{\alpha}^{t} + ICM_{\beta}^{t})] mod F + 1 & T = 1 \\ [(T-1)_{\alpha} + (T-1)_{\beta}] mod F + 1 & T > 1 \end{cases} \tag{11}$$

In this formula $\alpha = count\ 0^s \times 48$, $\beta = count\ 1^s \times 49$, ICM_i^t is the initial configuration matrix for the green channel that we got from the reserved hash (SHA-512) values for the green channel. F is the highest value of pixel that is 255. Whereas, T is the total number of blocks. For clear demonstration, let suppose we have the initial configuration matrix C^0 with a size 4×8 as given in Table 3.

Table 3. Initial Configuration.

1	1	1	0	0	1	0	1
1	1	0	1	1	0	1	1
0	0	1	0	0	1	1	0
1	1	0	1	1	0	1	1

From the ASCII table as the value of **1** is **49** and the value of **0** is **48**, so the decimal value for the confined rule structure of the $\mathbf{1}^{st}$ sub-matrix can be found as follows.

$$CA_{CR} = [(12 \times 48) + (20 \times 49)] mod 255 + 1$$
$$\Rightarrow CA_{CR} = 26 + 1 = 27$$

Now converting this **27** into the binary form $\mathbf{[00011011]_2}$. As in this binary representation only **4** bits are **1**, So only these **4** cells will participate in the XOR operation of 1^{st} sub-matrix. To find those specific **four** cells we now put this binary value in Equation (10).

$$C^{t+1}_{M(i,j)} = (0 \times C^t_{i-1,j+1}) \oplus (0 \times C^t_{i,j+1}) \oplus (0 \times C^t_{i+1,j+1}) \oplus (1 \times C^t_{i-1,j}) \oplus (1 \times C^t_{i+1,j}) \oplus (0 \times C_{i-1,j-1}) \oplus (1 \times C_{i,j-1}) \oplus (1 \times C_{i+1,j-1}) \quad (12)$$

So Equation (12) will be simplified as follows.

$$C^{t+1}_{M(i,j)} = C^t_{i-1,j} \oplus C^t_{i+1,j} \oplus C_{i,j-1} \oplus C_{i+1,j-1} \quad (13)$$

This updated Equation (13) has four state variables that are $C_W = 1$, $C_E = 1$, $C_S = 1$ and $C_{SE} = 1$. So, only these four cells will contribute in the state update process of the 1^{st} sub-matrix. From Table 2 we can see $C^t_{i-1,j}$ is actually C_W, $C^t_{i+1,j}$ is actually C_E, $C^t_{i,j-1}$ is actually C_S, while $C^t_{i+1,j-1}$ is actually C_{SE}. Hence, by means of these confined rules structure, the processing power, computation time, efficiency and the security can be improved immensely.

The initial configuration matrix gets updated based on these confined rules mapping and then XORed with the particular plain image sub-matrix to acquire the updated matrix.

$$C^{t+1}_{k(i,j)} = C^{t+1}_{M(i,j)} \oplus C^t_{k(i,j)} \quad (14)$$

Here $C^{t+1}_{k(i,j)}$ is the updated matrix, $C^{t+1}_{M(i,j)}$ is the updated initial configuration based on confined rules, and $C^t_{k(i,j)}$ is the plain image matrix. In decryption process the reverse equation to get back the original plain sub-matrix will be as follows.

$$C^t_{k(i,j)} = C^{t+1}_{M(i,j)} \oplus C^{t+1}_{k(i,j)} \quad (15)$$

2.4. DNA Sequence

DNA molecule contain the genetic information that is used for the purpose of reproduction, functioning and growth of all living organism. Any DNA sequence comprised of four nucleic acid bases: Adenine (**A**), Cytosine (**C**), Guanine (**G**), and Thymine (**T**). These bases follow the principle of Watson–Crick, that is A and T are complementary, so are C and G same as in the binary system, **1** and **0** are complementary.

Likewise, the case of two-bits binary **11** and **00** are also complementary. Usually in DNA sequence every base is represented by the two-bits as 11, 10, 01 and 00 to represent these four bases T, G, C, and A respectively. There are basically **24** different kinds of DNA coding schemes. Nonetheless, out of those, only eight kinds fulfill the Watson–Crick complementary principle [51], which are given in

Table 4. Please note that, like DNA encoding, the DNA decoding rules are just the inverse process of the DNA encoding rule.

Table 4. DNA Rules.

Rule 1	Rule 2	Rule 3	Rule 4
00 ↦ **A**	00 ↦ **A**	01 ↦ **A**	01 ↦ **A**
11 ↦ **T**	11 ↦ **T**	10 ↦ **T**	10 ↦ **T**
01 ↦ **C**	10 ↦ **C**	00 ↦ **C**	11 ↦ **C**
10 ↦ **G**	01↦ **G**	11↦ **G**	00 ↦ **G**
Rule 5	**Rule 6**	**Rule 7**	**Rule 8**
10 ↦ **A**	10 ↦ **A**	11 ↦ **A**	11 ↦ **A**
01 ↦ **T**	01 ↦ **T**	00 ↦ **T**	00 ↦ **T**
00 ↦ **C**	11 ↦ **C**	01 ↦ **C**	10 ↦ **C**
11 ↦ **G**	00 ↦ **G**	10 ↦ **G**	01 ↦ **G**

In this paper, we propose a new bidirectional mathematical model of DNA rule generator based on two bits of input as shown in Figure 5. The working principle of this model is based on Automaton state machine, so it is called the **SM-DNA** rule generator. SM-DNA generates random DNA rules faster than the chaotic serious-based rules selection method.

The pixel value of image matrix was converted into binary, then divided into sub-matrices form. We used i^{th} and j^{th} bits as an input. SM-DNA select random DNA rule for each block matrix. In our case, the first and last bit of every sub-block matrix was used as an input. The proposed SM-DNA model has eight states and each state represents one rule. Each state has $\mathbf{2^2}$ possible output states that take to next or previous state, so it means 8^4= **4096** different rules combination, while rule transition from one to another depends solely on the following **4** combinations, either **11**, **10**, **01** or **00**. For 00 it moves anti-clockwise direction and for 01, 10 and 11 it moves clockwise direction.

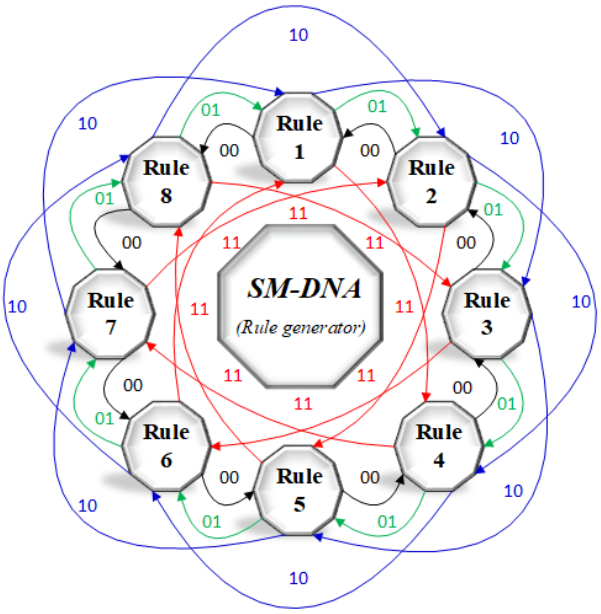


Figure 5. SM-DNA (Random rule generator).

For ease of implementation binary matrix of the image was converted into 4×8 sub-matrices. SM-DNA allocated random DNA conversion rule to each sub-matrix very efficiently and quickly. For more clear understanding, working principle illustration is given in Figure 6 with a total of N binary sub-block matrices termed as $B_1, B_2, B_3, \ldots, B_{(n-1)}, B_n$. So the general formula of SM-DNA described as follows.

$$SM:DNA=\begin{cases}[(NP_1+NP_2)]modB+1 & N=1\\ [(N-1)_{k^{th}}+(N-1)_{n^{th}}]modB+1 & N\neq 1\end{cases} \tag{16}$$

where $\mathbf{k^{th}}$ and $\mathbf{n^{th}}$ are the two particular bits of own choice to serve as an input for the rule selection, that is $first^{bit}$ and $last^{bit}$ in our case. Suppose BM_{1st} is our first block matrix and also suppose the $\mathbf{P_1}$ and $\mathbf{P_2}$ value came out **109** and **207**, respectively. So DNA starting rule for the 1st sub-block matrix can be gotten as follows.

$$SM:DNA_1=(1\times 109+1\times 207)mod8+1=5$$

So, **Rule 5** will be the starting rule that will be used for binary to DNA conversion of 1^{st} sub-block matrix.

$$BM_{1st}::\begin{bmatrix}10 & 11 & 01 & 01\\ 00 & 11 & 00 & 01\\ 00 & 10 & 10 & 01\\ 10 & 11 & 11 & 11\end{bmatrix}\xrightarrow{Rule5}::\begin{bmatrix}A & G & T & T\\ C & G & C & T\\ C & A & A & T\\ A & G & G & G\end{bmatrix}$$

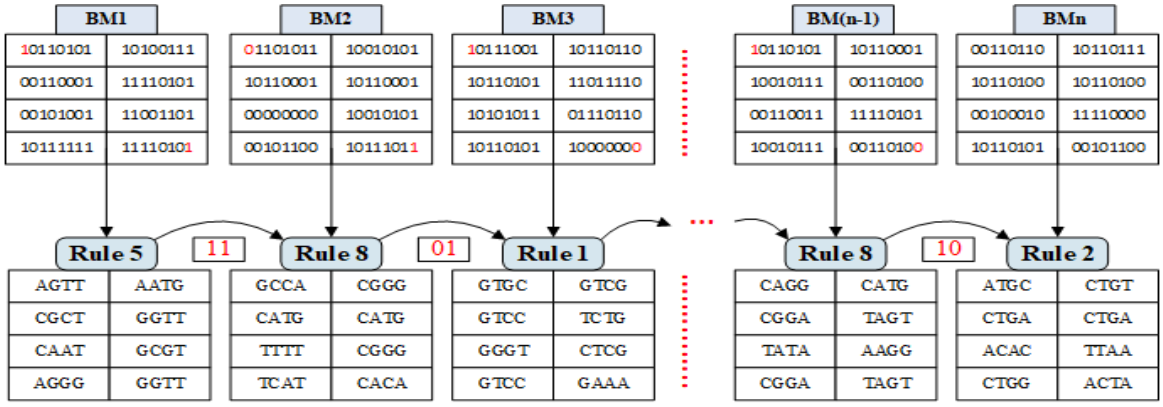


Figure 6. Binary to DNA Conversion.

For the 2^{nd} sub-block matrix $i^{th}=first^{bit}$ and $j^{th}=last^{bit}$ bit of 1^{st} block matrix will be used that are **1,1**. So, the **Rule 8** is the conversion rule for 2^{nd} sub-block matrix.

$$BM_{2nd}::\begin{bmatrix}01 & 10 & 10 & 11\\ 10 & 11 & 00 & 01\\ 00 & 00 & 00 & 00\\ 00 & 10 & 11 & 00\end{bmatrix}\xrightarrow{Rule8}::\begin{bmatrix}G & C & C & A\\ C & A & T & G\\ T & T & T & T\\ T & C & A & T\end{bmatrix}$$

As input bits of the 2^{nd} sub-block matrix are **0,0**, So it went to state **7**. Thus, **Rule 7** will be the conversion rule for the 3^{rd} sub-block matrix.

$$BM_{3rd}::\begin{bmatrix}10 & 11 & 10 & 01\\ 10 & 11 & 00 & 01\\ 00 & 10 & 01 & 11\\ 10 & 00 & 10 & 00\end{bmatrix}\xrightarrow{Rule7}::\begin{bmatrix}G & A & G & C\\ G & A & T & C\\ T & G & C & A\\ G & T & G & T\end{bmatrix}$$

Similarly, all the sub-block matrices were converted into DNA sequences based on rules generated by SM-DNA as shown in Figure 6. Remember that for the N^{th} sub-block matrix the $(N-1)^{th}$ sub-block matrix will decide the rule. From security prospective as rule selection based on two bits inputs so it is very difficult to predict that out of 32-bits, which two-bits are working as an input. Thus its difficult to guess the rule selection without getting adequate information about the working principle of SM-DNA.

3. Proposed Encryption Method

The general block diagram of the proposed image encryption method is shown in Figure 7. The proposed encryption method deal with the binary bit-planes of each channel. The detailed encryption steps are given below.

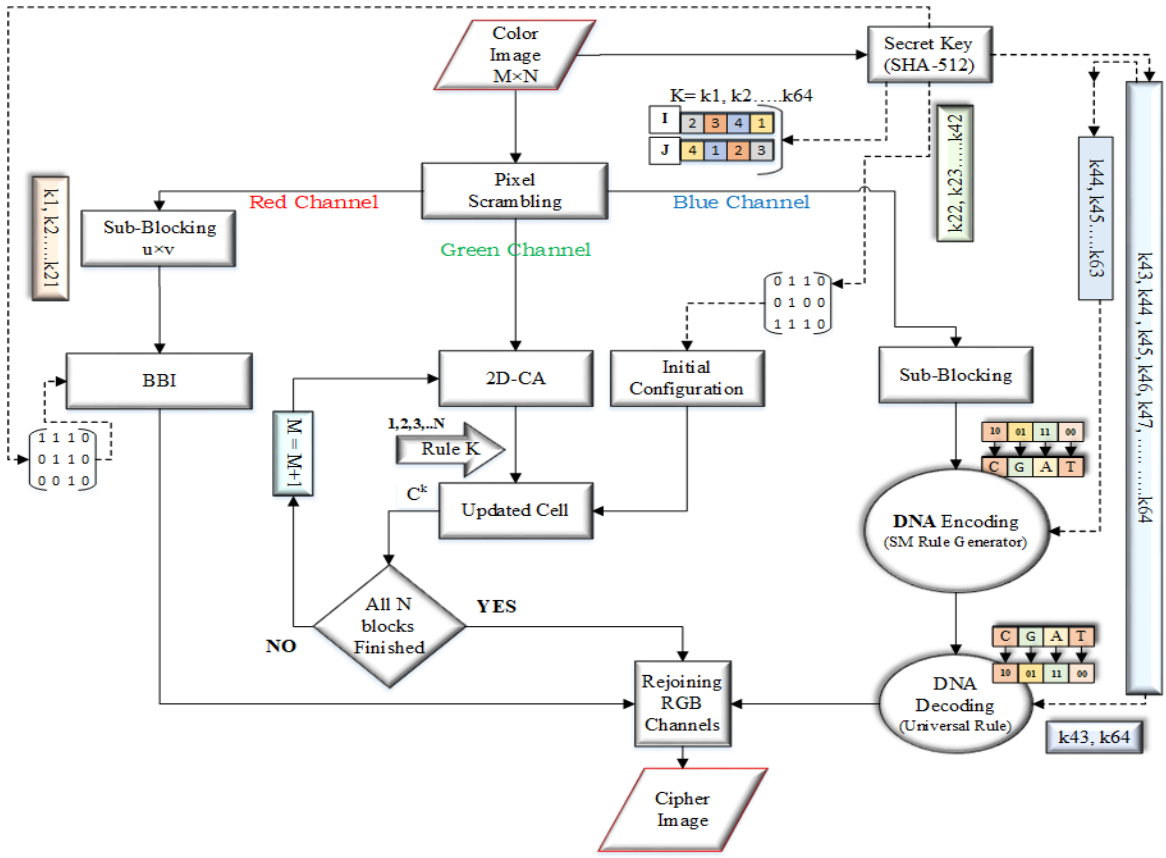


Figure 7. General Block Diagram

Step 1: Get the double hash-value of the image by using SHA-512. Hexadecimal forms of the key string will be gotten. Split the key string into three parts i.e., $K_1 = 21$, $K_2 = 21$ and $K_3 = 22$ hexadecimal pairs to use in the formulas and initial configuration matrices of red, green and blue channels as follows.

$$K_{red} = k_1, k_2, k_3, k_4, \ldots, k_{20}, k_{21}$$

$$K_{green} = k_{22}, k_{23}, k_{24}, k_{25}, \ldots, k_{41}, k_{42}$$

$$K_{blue} = k_{43}, k_{44}, k_{45}, k_{46}, \ldots, k_{63}, k_{64}$$

Step 2: Take the color image $(M, N, 3)$, where M and N denote the rows and columns of the image, respectively. Scramble as described in the scrambling portion, while the hexadecimal form of a key was used as a seed to generate array sequence by $LSS - PRNG$. For the two rounds of sequences, the seed values are set through the following equations.

$$x_0 = \frac{(k_1 k_2 \oplus k_3 k_4) + \ldots + (k_{13} k_{14} \oplus k_{15} k_{16})}{256} \tag{17}$$

$$y_0 = \frac{(k_{17} k_{18} \oplus k_{19} k_{20}) + \ldots + (k_{29} k_{30} \oplus k_{31} k_{32})}{256} \tag{18}$$

$$z_0 = \frac{(k_{33}k_{34} \oplus k_{35}k_{36}) + \ldots + (k_{45}k_{46} \oplus k_{47}k_{48})}{256} \tag{19}$$

The integer values for two round from the n^{bits} of the stream gotten by the following formula.

$$integer = \sum_{i=1}^{n} b_i \times 2^{i-1} \tag{20}$$

For the two rounds of array sequence the initial state can be set by the following way.

$$\begin{aligned} X_0^i &= [integer \times y_0 \times (x_0 + z_0)] mod 1 \\ r^i &= [integer \times x_0 \times (y_0 + z_0)] mod 4 \end{aligned} \tag{21}$$

where, $i = (1, 2)$ termed as rounds and it will generate two initial values X_0^1,r^1 and X_0^2,r^2 respectively.

Step 3: Split the scrambled color image $T^{Scr.}_{(M,N,3)}$ into respective three channels R, G and B channels and we can get the three components, $T_R^{Scr.}$, $T_G^{Scr.}$ and $T_B^{Scr.}$ as given below.

$$T_R^{Scr.} = T_{r1}, T_{r2}, T_{r3}, \ldots, T_{rMN} \tag{22}$$

$$T_G^{Scr.} = T_{g1}, T_{g2}, T_{g3}, \ldots, T_{gMN} \tag{23}$$

$$T_B^{Scr.} = T_{b1}, T_{b2}, T_{b3}, \ldots, T_{bMN} \tag{24}$$

Here T_{ri}, T_{gi} and T_{bi} are the i^{th} pixel values of the red, green and blue channels respectively, whereas $T_{ri}; T_{gi}; T_{bi} \in [0, 255]$. Transformed the pixel values of each channel into the binary windows as given below.

$$T_{R(M,N)} = R_{w1}, R_{w2}, R_{w3}, \ldots, R_{w8} \tag{25}$$

$$T_{G(M,N)} = G_{w1}, G_{w2}, G_{w3}, \ldots, G_{w8} \tag{26}$$

$$T_{B(M,N)} = B_{w1}, B_{w2}, B_{w3}, \ldots, B_{w8} \tag{27}$$

Now split the window matrix into $m \times n$ size sub-blocks, where $u \times m = M$ and $v \times n = N$.

Step 4: Take the 1^{st} block of $T_{R(M,N)}$ with a size 8×8 and split it into two equal columns block e.g., 8×4 and 8×4; termed as LCB and RCB. By using formula get the starting bit for the 1^{st} block-matrix and apply block bit inversion as described in Section 2.2.

Step 5: Take the green channel $T_{G(M,N)}$ and apply the Cellular Automata as described in Section 2.3.

Step 6: Take the blue channel $T_{B(M,N)}$ of size $H \times W$ divide it into the $u \times v$ size sub-blocks matrices as given below. While $u \times H = M$ and $v \times W = N$, and apply DNA conversion as described in Section 2.4.

$$T^{bi}_{B(M,N)} = BM_1^{bi}, BM_2^{bi}, BM_3^{bi}, \ldots, BM^{bi}_{(n-1)^{th}}, BM^{bi}_{n^{th}} \tag{28}$$

With SM-DNA convert every binary block matrix into DNA matrix based on allocated rule of SM-DNA as follows.

$$BM_1^{bi} \rightleftharpoons BM_1^{DNA}, BM_2^{bi} \rightleftharpoons BM_2^{DNA}, BM_3^{bi} \rightleftharpoons BM_3^{DNA}$$
$$, \ldots, BM^{bi}_{(n-1)} \rightleftharpoons BM^{DNA}_{(n-1)}, BM^{bi}_{n^{th}} \rightleftharpoons BM^{DNA}_{n^{th}}$$

Join all these DNA sub-block matrices into a single matrix as described below.

$$BM_1^{DNA}, BM_2^{DNA}, BM_3^{DNA}, \ldots, BM^{DNA}_{n^{th}} = T^{DNA}_{B(M,N)} \tag{29}$$

Get the universal $DNA \rightarrow binary$ rule through following formula.

$$DNA_{UR} = [P_1 + P_2] mod 8 + 1 \tag{30}$$

Convert the whole DNA sequence matrix back into binary form through this universal rule.

Step 7: Add the random matrix of the scrambling part to the pixel values of each channel as described below.

$$Cipher_{RGB} \leftrightarrow \begin{cases} C^{red}_{(i,j)} = [E^{red}_{(i,j)} + R_{(i,j)}] mod 256 \\ C^{green}_{(i,j)} = [E^{green}_{(i,j)} + R_{(i,j)}] mod 256 \\ C^{blue}_{(i,j)} = [E^{blue}_{(i,j)} + R_{(i,j)}] mod 256 \end{cases} \tag{31}$$

where $C_(i,j)$ denotes cipher value of particular pixel, $E_(i,j)$ denotes encrypted value of the pixel, $R_(i,i)$ denotes the Random matrix, $i = 1, 2, ...M$, and $j = 1, 2, ..., N$.

Step 8: Rejoin all the channels to get the cipher image.

$$Cipher_{(M,N,3)} = C^{red}_{(i,j)}; C^{green}_{(i,j)}; C^{blue}_{(i,j)} \tag{32}$$

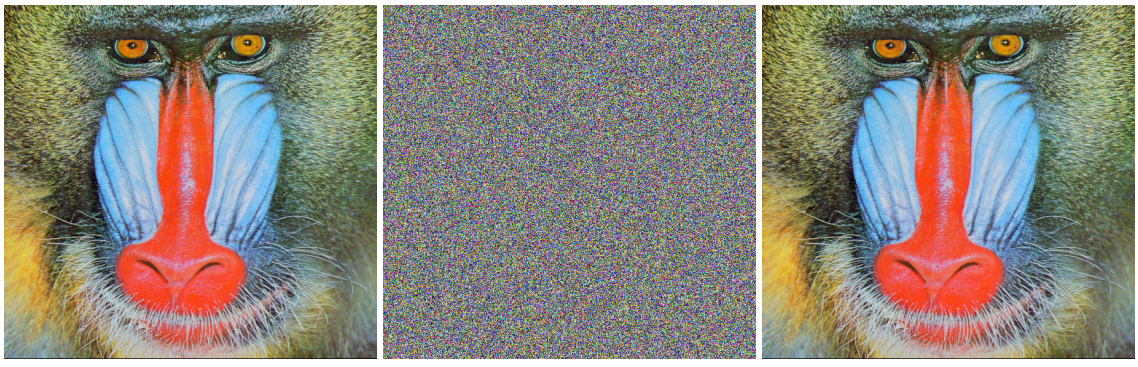
4. Experimental Results and Discussion

This section included the performance and simulation results along with the comparison of results with earlier proposed image encryption schemes. The experimental results were manipulated in **Python 3.6.5** (Jupyter Notebook Environment) installed over a personal Laptop, CPU Intel **Core I5** with **4GB** memory and operating system **Window 10**.

The test image baboon, its cipher image and the decrypyted image are shown in Figure 8. The experimental parameters are given in Table 5. Whereas all the test images that we used for experiments are shown at the end of paper in Figure 9.

Table 5. Experiment Parameters.

Terms	System Parameters/Values
512-bit Hexadecimal key	40CD744F6682BD0ACF73579A5DC353DB 3A295D3A2D8703566C8ACF9BE8AA688E 87621E8F5F3D073763C46E93FF7B1A2B 0476C3BB8408F2A2E8AFAB48087BB9C4
Seed	$int(K[0:2], 16) = 207$
P1	$int(K[45:48], 16) = 3086$
P2	$int(K[60:63], 16) = 1030$
$P1_{universal}$	$int(K[43], 16) = 12$
$P2_{universal}$	$int(K[64], 16) = 2$



(**a**) Plain Image (**b**) Ciphered Image (**c**) Decrypted image

Figure 8. Encryption and decryption results. (**a**) The plain image of baboon; (**b**) The cipher image of baboon; (**c**) The decrypted image of baboon.

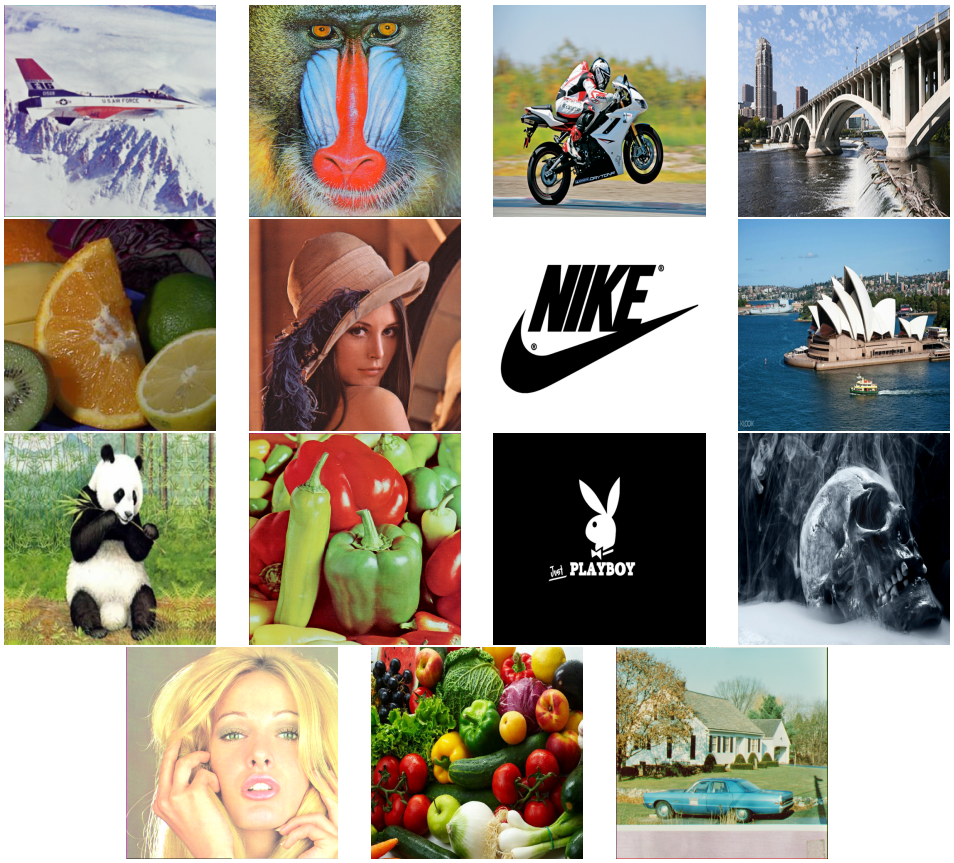


Figure 9. All the test Images.

5. Security Analysis and Test

5.1. Security Keyspace

The keyspace represents the entire number of likely combinations of the security key. The most common attack is the Brute-force attack in which an attacker endeavors to predict the accurate security key by overly searching the keyspace of the encryption algorithm. Thus, in order to withstand against the Brute-force attack, an adequately huge keyspace is one of the main factor that can guarantee more security [52].

For resisting the best attack, secure hash algorithm (SHA-512) uses a keyspace of $\mathbf{2^{256}}$. The comparison of keyspace and approach along with testing parameters of the proposed algorithm is given in Table 6.

Table 6. Keyspace and Comparison with earlier.

Algorithm	Rule/Map	Keyspace	Operation	Image Type	Testing Parameters
Enayatifar[2017] [53]	LM	120 bit	DNA XOR	Gray Scale	NPCR, UACI, Entropy, CC, Key-space, Histogram, Time parameter
Kumar[2016] [54]	ILM	128 bit	CA	Color	NPCR, UACI, Entropy, CC, Key-space, Histogram, Noise test, Crop test
Guesmi[2016] [55]	Lorenz system	SHA-256	DNA XOR	Color	NPCR, UACI, Entropy, CC, Key-space, Histogram
S.Suri[2018] [56]	ILM	SHA-256	DNA XOR DNA Addition	Color Binary	NPCR, UACI, Entropy, CC, Key-space, Histogram, Contrast
Our Proposed 3C3R	LSS PRNG	SHA-512 $2^{>310}$	DNA, BBI, 2D-CA	Color (24bit)	NPCR, UACI, Entropy, GVD CC, Key-sensitivity, Histogram, PSNR, Occlusion, Chosen/known plain text, Variance.

Moreover, the SM-DNA possible rules combination $\mathbf{8^4}$ can also be taken as keyspace. Except for the fact that the hash keys for generating initial configuration matrices for each channel and in their particular rule generating formulas are $\mathbf{2^{22}}$, $\mathbf{2^{22}}$, $\mathbf{2^{21}}$, and $\mathbf{2^5}$ respectively. So, the overall keyspace of the proposed algorithm is $2^{256} \times 2^{21} \times 2^{22} \times 2^{22} \times 2^5 \times 8^4$ which is very large as compared to $\mathbf{2^{128}}$. To calculate the computational load, let's the fastest computer computes 2^{80} computations in **1** second and $2^{80} \times 365(\text{days}) \times 24(\text{h}) \times 60(\text{min}) \times 60(\text{s})$ [57]. So, to compute $8^4 \times 2^{>310}$ computations, a total of following years required.

$$\frac{8^4 \times 2^{>310}}{2^{80} \times 365 \times 24 \times 60 \times 60} \Rightarrow \approx 2.68 \times 10^{65} years$$

This huge computation load is sufficient enough to break the crypto-system. The computational load also proved that the **3C3R** can effectively withstand against the brute-force attacks.

5.2. Histogram Analysis

The histogram of an image demonstrate the distribution of the pixel values. An intruders usually recover the meaningful information from the fluctuating histogram of the encrypted image. So, in order to prevent an intruder from recovering such information, it is important that the histogram of the cipher-image should have no statistical resemblance to the plain image and also should have uniform distribution. The histograms of the ciphered images of all the test images are shown in Figure 10. While Figure 11 shows the histogram of RGB channels of the plain image and its corresponding cipher image. The histogram of the each RGB channel of the cipher image is almost uniform. Moreover, by computing the variance of histogram, we evaluated the uniformity of our ciphered images. The lesser the variance means the higher is the uniformity of the encrypted images [58–61].

$$var(I) = \frac{1}{k^2} \sum_{i=1}^{k} \sum_{j=1}^{k} \frac{1}{2} (I_i - I_j)^2 \tag{33}$$

Table 7 listed the histogram variances of the plain images (R,G,B) and encrypted images along with a comparison with the other methods. From the table, we can see that in most of the test images the histogram variance of the proposed algorithm is less as than from Ref. [62] and Ref. [49]. This proved that the **3C3R** has better security in comparison to those algorithms.

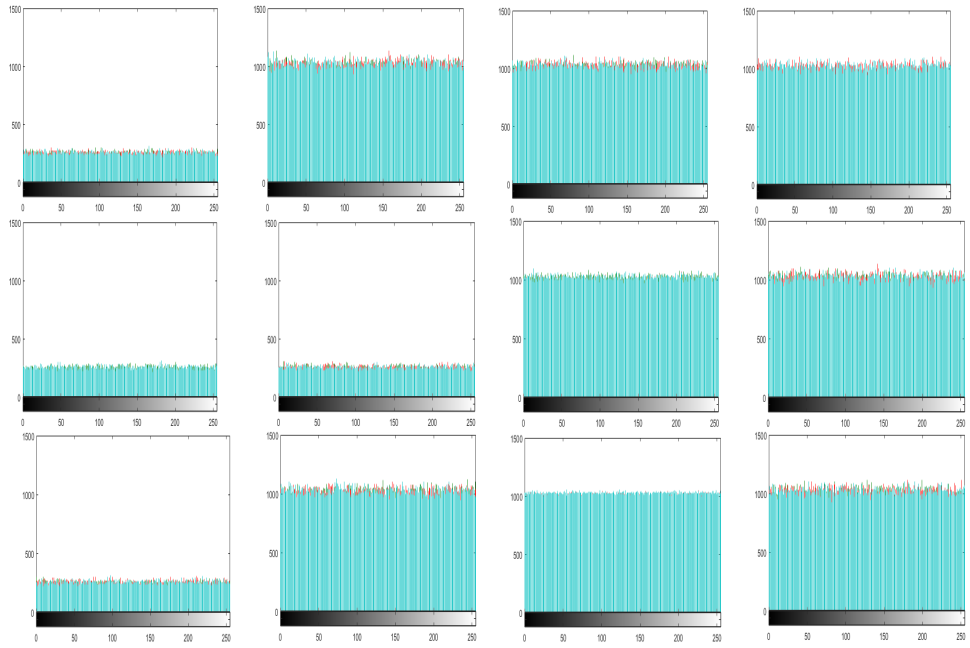


Figure 10. *Cont.*

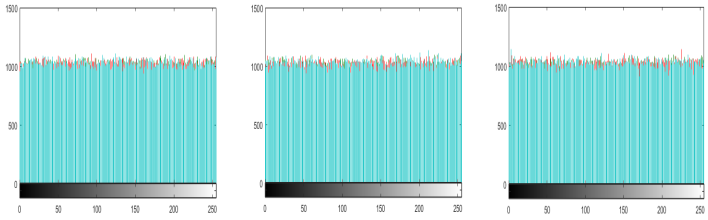
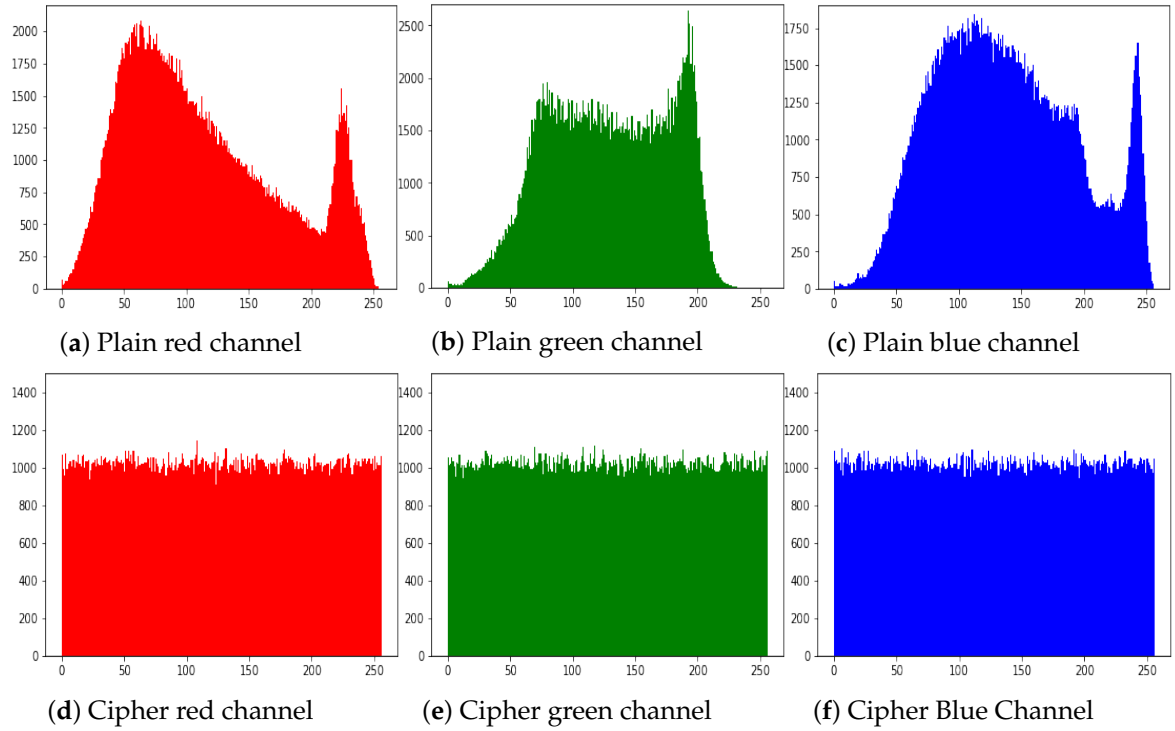


Figure 10. Histogram of ciphered images of the above all the test Images in Sequence.



(**a**) Plain red channel (**b**) Plain green channel (**c**) Plain blue channel

(**d**) Cipher red channel (**e**) Cipher green channel (**f**) Cipher Blue Channel

Figure 11. Histograms of RGB channels of Baboon. (**a**) Plain red channel; (**b**) Plain green channel; (**c**) Plain blue channel; (**d**) Ciphered red channel; (**e**) Ciphered green channel; (**f**) Ciphered blue channel.

Table 7. Histogram Variance Comparison.

Images	Plain			3C3R Cipher			Ref. [62] Cipher			Ref. [49] Cipher		
	Red	Green	Blue	Red	Green	Blue	Red	Green	Blue	Red	Green	Blue
Lena	123,072.5	87,100.835	33,522.734	293.99	**264.977**	**254.722**	**247.78**	279.62	265.71	527.32	504.75	501.68
Couple	289,630.656	337,863.062	210,359.81	**242.434**	**216.323**	**246.357**	284.35	247.37	260.76	-	-	-
Female	113,045.289	64,436.410	66,971.062	298.134	**256.690**	243.771	**280.64**	280.46	**230.42**	-	-	-
Tree	129,825.531	57,011.605	81,373.710	**239.590**	**233.216**	238.281	282.81	254.87	**225.79**	-	-	-
Bean	168,076.796	501,640.093	789,945.75	**231.815**	**257.557**	**228.984**	232.98	279.61	245.61	-	-	-
House	992,034.12	1,330,180.12	768,126.75	**998.60**	**1107.34**	1046.96	1070.2	1231.2	**941.65**	-	-	-
All White	2.67×10^8	2.67×10^8	2.67×10^8	**220.130**	**203.561**	**216.067**	291.021	223.145	264.58	-	-	-

5.3. Pixel Correlation Analysis

Pixel correlation analysis is another test used to find the relationship of neighboring pixels in the plain image and the ciphered image. A good encryption algorithm aims to minimize the relationship among the neighboring pixels with regards to prevent the leakage of actual information.

The correlation coefficient $C_{r(x,y)}$ between the two neighboring pixels can be calculated by the following formulas.

$$E(w) = \frac{1}{P}\sum_{i=1}^{P} w_i \tag{34}$$

$$D(w) = \frac{1}{P}\sum_{i=1}^{P}(w_i - E(w))^2 \tag{35}$$

$$C_{ovariance}(w,z) = \frac{1}{P}\sum_{i=1}^{P}(w_i - E(w))(z_i - E(z)) \tag{36}$$

$$C_{r(w,z)} = \frac{C_{ovariance}(w,z)}{\sqrt{D(w) \times D(z)}} \tag{37}$$

In above equations (w, z) is the gray values of neighboring pixels, $C_{ovariance}(w,z)$ is the covariance, P is the total number of pixels selected from the image, while $E(w)$ is the mean and $D(w)$ is the variance. Figure 12 displays the pixels of the plain image and ciphered image of the proposed algorithm in horizontal (H), vertical (V) and diagonal (D) distribution. In Tables 8 and 9 the pixel correlation comparison was done with some previous algorithms. Table 8 shows a comparison of **8K** pairs of neighboring pixels that are randomly selected from the plain image and the ciphered image in the H, V, and D directions to perform pixel correlation analysis. Whereas, Table 9 shows a correlation comparison of **1K** random pixels of the Lena image. Whereas, Table 10 listed the pixel correlation values of the different test images.

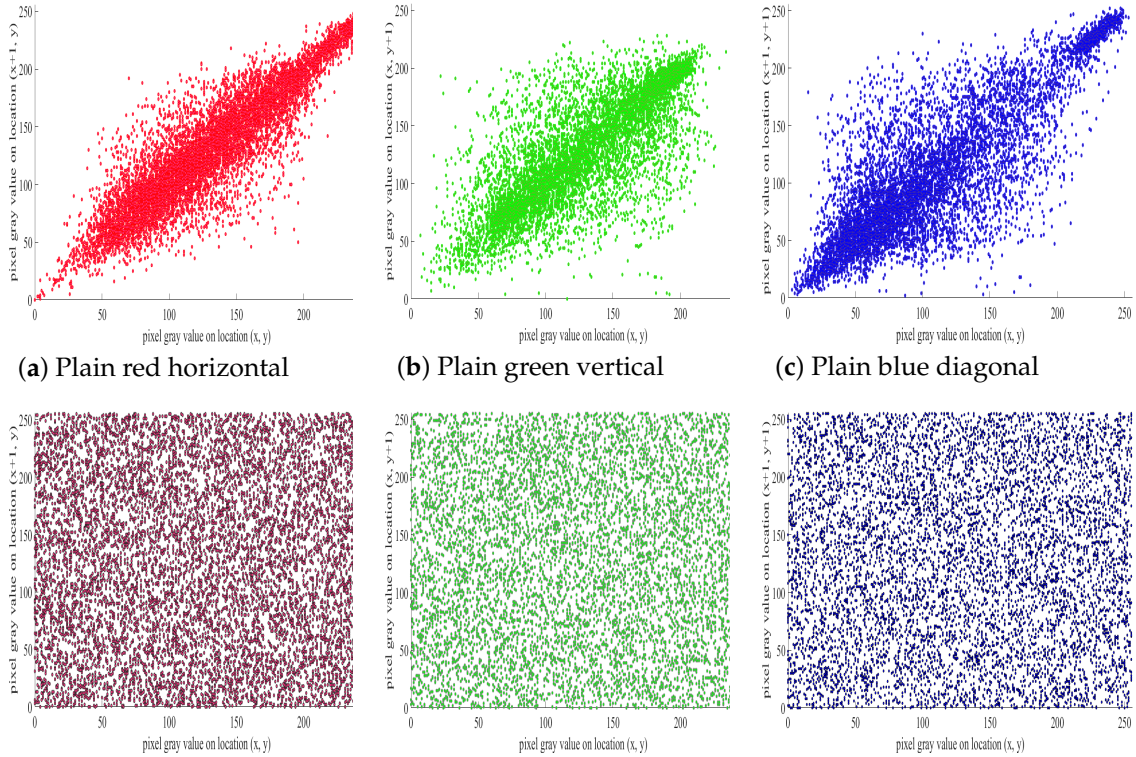
We casually selected the pixel pairs in the horizontal, vertical and diagonal axes, respectively. From Table 9 it is obvious that in term of overall correlation, proposed **3C3R** and Ref. [63] are performing well, while in term of diagonal values our algorithm, Refs. [63–65] are giving satisfactory values. While the overall and in terms of **UACI** and **NPCR**, the proposed **3C3R** outperformed.

Table 8. Pixel correlation of 8k random pixel of Lena.

Algorithms		Cipher Image		
		Horizontal	**Vertical**	**Diagonal**
3C3R	**Plain**	0.95589	0.96567	0.93313
	Cipher	0.00750	**−0.00184**	**0.00012**
Ref. [15]		−0.0082	−0.0128	−0.0012
Ref. [66]		0.0020	−0.0009	**0.0017**
Ref. [28]		0.0265	0.0792	0.0625
Ref. [67]		0.0055	0.0041	0.002
Ref. [68]		**0.0005**	0.003	0.0021
Ref. [69]		0.0044	0.0034	0.0020
Ref. [70]		**0.0012**	0.0026	0.0021
Ref. [71]		0.0024	0.0012	0.0016
Ref. [72]		0.0072	0.0058	0.0031
Ref. [55]		0.0022	**0.0001**	**−0.0017**
Ref. [73]		0.0214	0.0465	−0.0090
Ref. [58]		−0.0077	**0.0002**	−0.0055

Table 9. Correlation Comparison of 1000 random Pixels of Lena.

Algorithm	Horizontal	Vertical	Diagonal	UACI	NPCR
Ref. [64]	0.003	−0.0040	**0.0013**	33.45	99.60
Ref. [63]	**0.0018**	0.0011	**−0.0013**	33.43	99.61
Ref. [74]	−0.0023	0.0019	−0.0034	33.51	99.62
Ref. [75]	**0.0020**	−0.0007	−0.0014	27.97	98.36
Ref. [65]	−0.0098	−0.0050	**−0.0013**	32.48	93.21
Ref. [76]	−0.0237	−0.0178	−0.0284	33.58	99.62
Ref. [77]	0.0080	0.0098	−0.0058	33.43	99.60
3C3R	−0.0027	**−0.00054**	**−0.0013**	**34.45**	**99.998**



(**a**) Plain red horizontal (**b**) Plain green vertical (**c**) Plain blue diagonal

(**d**) Cipher red horizontal (**e**) Cipher green vertical (**f**) Cipher blue diagonal

Figure 12. Correlation of RGB Channels of baboon (**a**) Plain red in horizontal; (**b**) Plain green in vertical; (**c**) Plain blue in diagonal; (**d**) Ciphered red in horizontal; (**e**) Ciphered green in vertical; (**f**) Ciphered blue in diagonal.

Similarly, for **8K** random pixels the Refs. [55,58,66,68,70] are giving good results but the proposed **3C3R** is also performing well in regards to vertical and diagonal direction.

Table 10. Pixel Correlation.

Images	Channels	Plain Horizontal	Plain Vertical	Plain Diagonal	Cipher Horizontal	Cipher Vertical	Cipher Diagonal	Entropy (R,G,B)
Lena.jpg	Red	0.95589	0.96567	0.93313	0.00750	−0.00184	0.00012	7.9972
	Green	0.93722	0.95832	0.92499	0.036575	0.002284	−0.003513	7.9974
	Blue	0.91142	0.93501	0.88513	0.0014717	−0.008797	0.0096045	7.9967
Baboon.png	Red	0.92283	0.86082	0.85468	−0.003068	0.004990	−0.002213	7.999
	Green	0.86721	0.76839	0.7416	0.0076227	−0.002984	−0.007508	7.999
	Blue	0.91092	0.88181	0.83619	0.012334	0.0007122	0.006166	7.9994
Fruits.jpg	Red	0.9865	0.98558	0.97342	−0.006591	−0.021533	−0.008414	7.996
	Green	0.98127	0.97943	0.96401	−0.003553	0.015646	−0.003781	7.9968
	Blue	0.95148	0.94673	0.91089	0.0085275	0.0084825	−0.002547	7.9972
Pepper.bmp	Red	0.96088	0.96686	0.95436	−0.001994	−0.007431	−0.009151	7.999
	Green	0.98276	0.98156	0.96989	−0.004029	−0.001068	−0.001439	7.9992
	Blue	0.967	0.96797	0.94546	0.0011452	0.00081563	−0.005897	7.9998
Skull.png	Red	0.98542	0.99226	0.97901	0.003727	0.007569	−0.001029	7.999
	Green	0.98546	0.99283	0.9822	−0.009794	0.017652	−0.001104	7.9993
	Blue	0.98659	0.99249	0.98009	−0.009525	−0.010168	−0.008972	7.999
Nike.png	Red	0.98823	0.99089	0.972	−0.0876	0.000835	−0.007701	7.999
	Green	0.98618	0.99008	0.9706	−0.095098	0.0008801	0.0003289	7.9997
	Blue	0.98723	0.9906	0.97178	−0.008857	0.003444	0.0054982	7.999

Table 10. *Cont.*

Images	Channels	Plain Horizontal	Plain Vertical	Plain Diagonal	Cipher Horizontal	Cipher Vertical	Cipher Diagonal	Entropy (R,G,B)
Playboy.png	Red	0.97007	0.98775	0.95954	−0.08556	−0.006046	−0.000411	7.999
	Green	0.97039	0.98257	0.95089	−0.10513	−0.001043	−0.000546	7.999
	Blue	0.96919	0.98127	0.95565	−0.002814	−0.005621	0.0066023	7.999
Airplane.bmp	Red	0.94741	0.93617	0.88936	−0.029442	−0.001137	−0.008049	7.9975
	Green	0.94242	0.94759	0.90503	−0.034386	−0.000201	−0.014404	7.9971
	Blue	0.9586	0.92611	0.90723	0.013706	−0.009122	−0.001553	7.9973
Bike.png	Red	0.95786	0.95752	0.92943	−0.003292	0.0047614	0.0073559	7.9992
	Green	0.96244	0.96579	0.935	0.013224	0.007460	−0.004485	7.999
	Blue	0.97765	0.97543	0.96262	−0.003638	−0.003146	0.0006025	7.9992
Opera.png	Red	0.97407	0.97118	0.9524	−0.000680	0.004448	−0.01049	7.999
	Green	0.96834	0.96442	0.94837	0.0009368	0.0007775	−0.004144	7.999
	Blue	0.97498	0.97311	0.95446	−0.001574	0.0000351	−0.01126	7.9993
Bridge.png	Red	0.95037	0.97699	0.92247	−0.004985	0.0036574	−0.002305	7.9993
	Green	0.95693	0.97946	0.92737	−0.000379	0.008972	0.003696	7.999
	Blue	0.96218	0.98441	0.94551	0.0069794	0.017072	−0.001523	7.9992
Vegetables.jpg	Red	0.97696	0.97952	0.96247	0.005099	−0.005643	−0.001603	7.999
	Green	0.97391	0.9767	0.95684	0.002007	0.009278	0.003720	7.999
	Blue	0.96823	0.96769	0.94104	−0.007227	−0.004639	−0.003687	7.999

5.4. Key Sensitivity Analysis

Because of the enhanced computational power, the current era's encryption algorithms should not have key length less than 100 bits or (2^{100}). Such key length can withstand against the exhaustive key search attack (brute force attack). The keyspace of the proposed encryption method is $2^{(>312)}$, which has high ability to resist the brute-force attack. Except for the fact that the key should also be extremely sensitive to the bit change. If the secret key will not be subtle enough, then a slight change in the actual secret keys can also properly recover the original image. Also, the secret key may perverted and as a result the actual keyspace may less than the theoretical one [60,61,78].

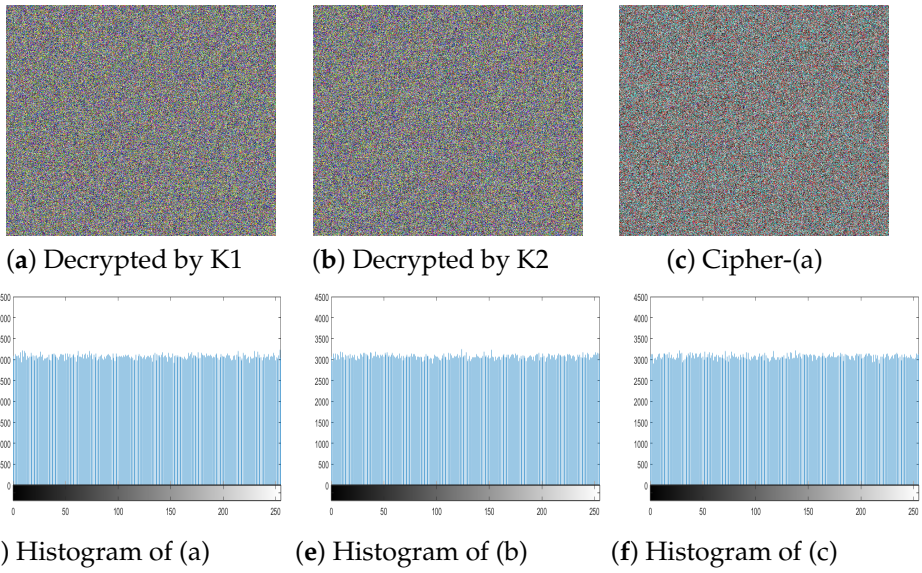
So, to check the sensitivity of the proposed algorithm towards the key, experiment has performed by changing one bit in the key with respect to the actual key as shown below.

Key$_o$ = "40 CD 74 4F 66 82 BD 0A CF 73 57 9A 5D C3 53 DB 3A 29 5D 3A 2D 87 03 56 6C 8A CF 9B E8 AA 68 8E 87 62 1E 8F 5F 3D 07 37 63 C4 6E 93 FF 7B 1A 2B 04 76 C3 BB 84 08 F2 A2 E8 AF AB 48 08 7B B9 C4".

Key$_1$ = "40 CD 74 4F 66 82 BD 0A CF 73 57 9A 5D C3 53 DB 3A 29 5D 3A 2D 87 03 56 6C 8A CF 9B E8 AA 68 8E 87 62 1E 8F 5F 3D 07 37 63 C4 6E 93 FF 7B 1A 2B 04 76 C3 BB 84 08 F2 A2 E8 AF AB 48 08 7B B9 C5".

Key$_2$ = "50 CD 74 4F 66 82 BD 0A CF 73 57 9A 5D C3 53 DB 3A 29 5D 3A 2D 87 03 56 6C 8A CF 9B E8 AA 68 8E 87 62 1E 8F 5F 3D 07 37 63 C4 6E 93 FF 7B 1A 2B 04 76 C3 BB 84 08 F2 A2 E8 AF AB 48 08 7B B9 C4".

Key$_o$ is the actual key while **Key$_1$**, **Key$_2$** are the one-bit changed keys from LSB (Right) and MSB (left) bit respectively. Figure 13a,b show the decrypted image by **Key$_1$** and **Key$_2$** respectively. Both **Key$_1$** and **Key$_2$** are one bit different from the actual key but the decrypted image is still like noise giving no useful information. The Figure 13c is the subtracted image of the actual cipher and the decrypted image of **Key$_1$**. The histogram of particular three resultant images is given in Figure 13d–f respectively. The subsequent uniform histograms proved that the proposed **3C3R** is very sensitive to key change even one-bit change leads towards totally different cipher image.



(**a**) Decrypted by K1 (**b**) Decrypted by K2 (**c**) Cipher-(a)

(**d**) Histogram of (a) (**e**) Histogram of (b) (**f**) Histogram of (c)

Figure 13. Key sensitivity test. (**a**) Ciphered image decrypted by K1; (**b**) Ciphered image decrypted by K2; (**c**) Subtracted image of cipher-(a); (**d**) Histogram of (a); Histogram of (b); Histogram of (c).

5.5. Differential Attack

Normally, hackers; to extract useful information create little modification in the plain image and then by using the encryption methodology they encrypt the identical images afore and afterwards these slight changes. Through this method, they attempt to find out the association among the plain images and the cipher images. Thus, we employed the number of pixel change rate **(NPCR)** and unified average changing intensity **(UACI)** to measure the robustness of the proposed algorithm against such attacks. The **NPCR** and **UACI** can be calculated by the following way.

$$NPCR_{U_1,U_2} = [\frac{\sum_{r,c} I(r,c)}{H \times M}] \times 100 \tag{38}$$

$$UACI_{U_1,U_2} = \frac{1}{H \times M}[\sum_{r,c} \frac{|U_1(r,c) - U_2(r,c)|}{2^8 - 1}] \times 100 \tag{39}$$

Here $\boldsymbol{U_1}$, $\boldsymbol{U_2}$ are two different ciphered images before and after one pixel of the plain image is changed, while $H \times M$ is the height and width of the test image. Whereas, ***I(r, c)*** can be defined as

$$I(r,c) = \begin{cases} 0 & U_1(r,c) \neq U_2(r,c) \\ 1 & otherwise \end{cases} \tag{40}$$

In the above equation, $\boldsymbol{I}$ depict the difference between $\boldsymbol{U_1}$ and $\boldsymbol{U_2}$.

Table 11 listed the NPCR and the UACI values of different test images along the comparison with some earlier algorithms. From the table values, we can see the test results of the **Lena** and **pepper** images of our **3C3R** giving **NPCR**$\geq$ **99.97** and **UACI**$\geq$ **33.46**. NPCR is high from Refs. [28,58,59,79,80], while it is comparable to [81]. Similarly, UACI is also comparable with the Refs. [59,79,81]. Thus the proposed **3C3R** has satisfactory security values.

5.6. Known and Chosen Plain Text Analysis

Most commonly, four kind of cryptanalsis attacks can be performed to crack the image encryption algorithms that are chosen-cipher-text attack, chosen-plaintext attack, cipher-text only attack, and known-plaintext attack [82]. Some famous image encryption algorithms given in Table 1 have already been broken with these attacks.

Table 11. NPCR and UACI comparison.

Algorithm	Lena		Pepper	
	$NPCR_{R,G,B}$	$UACI_{R,G,B}$	$NPCR_{R,G,B}$	$UACI_{R,G,B}$
3C3R	**99.978**	33.46	**99.998**	**34.54**
Ref. [28]	99.66	33.44	99.63	33.47
Ref. [58]	99.599	33.465	-	-
Ref. [59]	99.62	**33.65**	-	-
Ref. [79]	99.62	**33.77**	99.64	33.53
Ref. [81]	**99.71**	33.45	**99.74**	**33.53**
Ref. [80]	99.60	33.48	-	-
Ref. [83]	99.6037	33.44	-	-
Ref. [84]	99.61	33.463	99.608	33.49

The cryptanalysis model, where the attackers choose plaintext to obtain the corresponding cipher-text is called chosen-plain text attack. By examining the plaintext and the corresponding cipher-text, they try to presume some hidden useful information. Finally, by using that information they try to recover the original images [85,86].

The chosen-plaintext attack is the most powerful, and if the encryption may resist this attack, it has adequate security level to withstand the other three attacks. The proposed algorithm **3C3R** has satisfactory security level against known and chosen plaintext attacks.

We can see the cumulative $entropy_{(R,G,B)}$ value is **8.00**. In Table 15 comparison of the entropy values was also given, the value is higher from Refs. [82,87–89]. The proposed algorithms give the ideal all channels entropy values for all white, full black image and is given in Table 16. The proof of ideal entropy value is visible in the histogram of the ciphered image generated by our **3C3R** of all white and full black, and was given in Figure 14. The histogram of all white, full black and Playboy image is almost entirely flat. Furthermore, we created two special color images SP_{image1} and SP_{image2} respectively of size $P \times Q \times 3$. SP_{image1} is the color image with all pixels' values **0** except one pixel located at (252, 252) in R channel is 1. Similarly SP_{image2} is the color image with all pixels' values **1** except one pixel located at (252, 252) in R channel is 0. We made plaintext sensitivity analysis and the results are given in Table 12. The average values of NPCR and UACI are closed to the theoretical value. Thus on the bases of this test we can say that **3C3R** is robust against such attacks and can keep the image more secure.

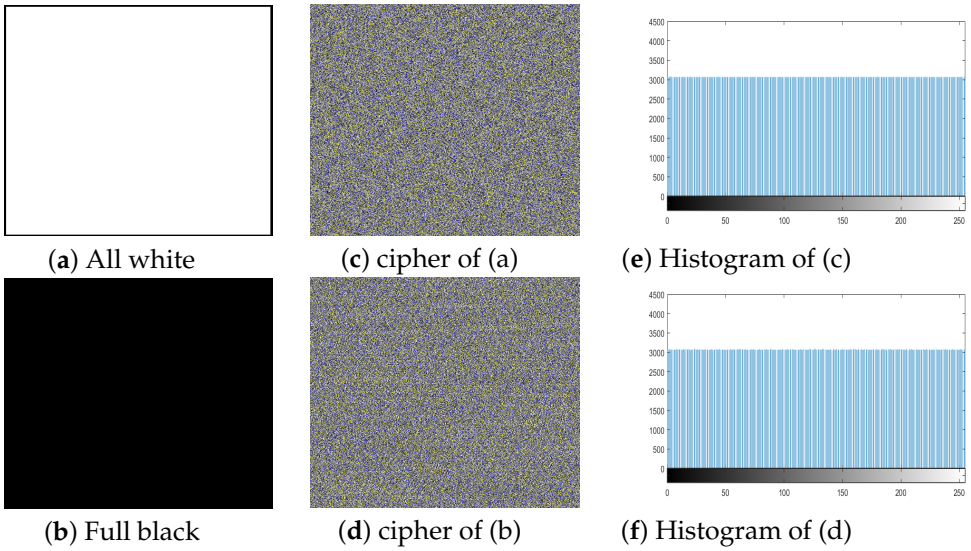


Figure 14. Encryption result of all white and full black images. (**a**) All white image; (**b**) Full black image; (**c**) Cipher image of (a); (**d**) Cipher image of (b); (**e**) Histogram of (c); (**f**) Histogram of (d).

Table 12. NPCR and UACI values for special plaintexts.

Images	$NPCR_{R,G,B(99.6174)}$			$UACI_{R,G,B(33.4738)}$		
	Red	**Green**	**Blue**	**Red**	**Green**	**Blue**
Full Black	99.7021	99.6903	99.6905	33.4641	33.4412	33.4710
All White	99.7025	99.6912	99.6908	33.4715	33.4698	33.4708
SP_{image1}	99.5975	99.4875	99.4764	33.4355	33.5970	33.4466
SP_{image2}	99.6105	99.5091	99.5622	33.4344	33.5201	33.4649

5.7. Robustness against Occlusion Attack

In image processing, PSNR and MSE are the most widely used parameters to test the encryption quality. The PSNR and MSE values can be calculated as follows.

$$PSNR = 10 \times log_{10}(\frac{255 \times 255 \times 3}{MSE_R + MSE_G + MSE_B})(dB) \tag{41}$$

$$MSE = \frac{1}{MN}\sum_{r}^{M}\sum_{c}^{N}\|I_p(r,c) - I_c(r,c)\|^2 \tag{42}$$

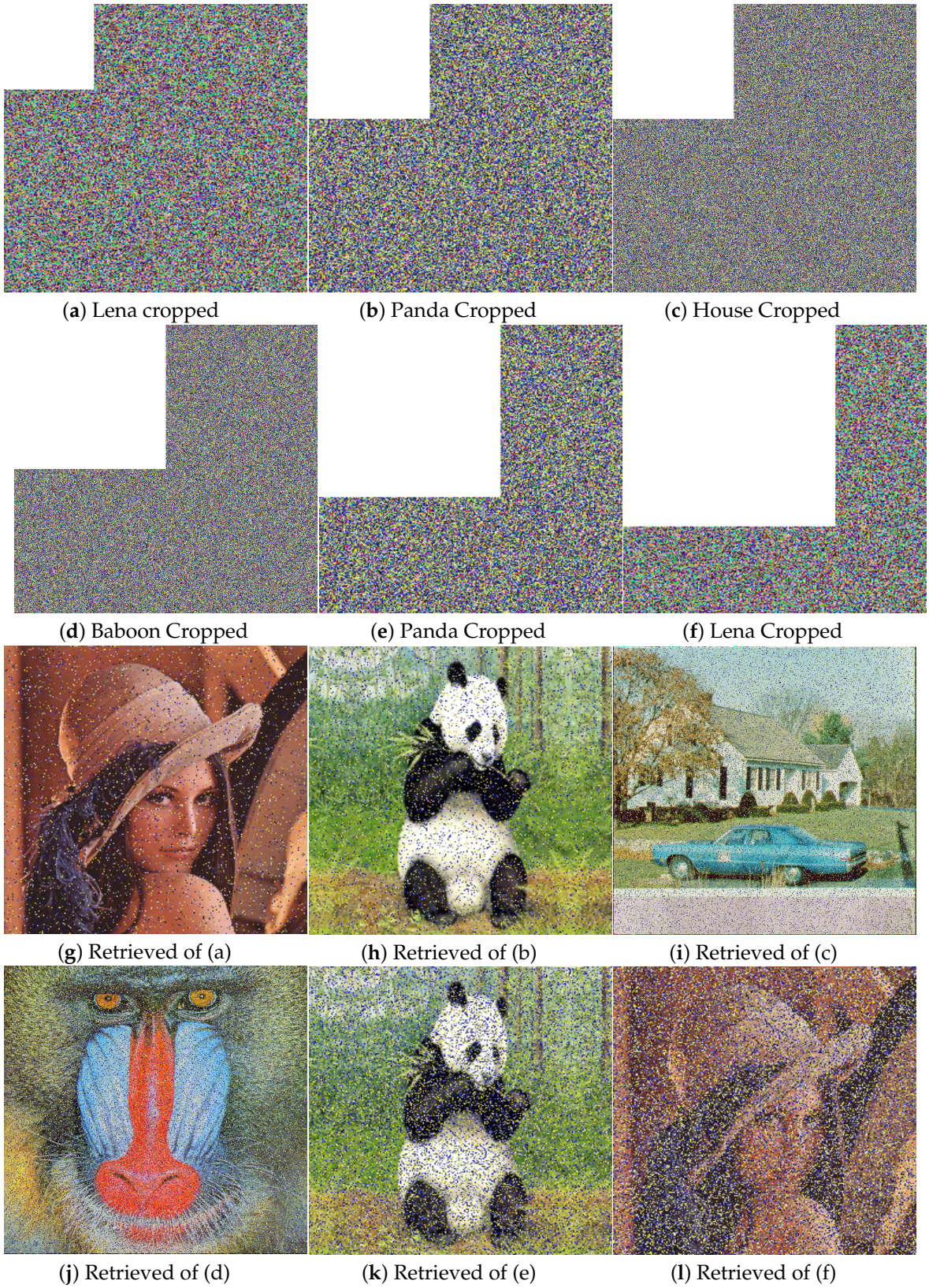
where $MSE_{R,G,B}$ is the mean square error of red, blue and green channel, between the cipher image $I_c(r,c)$ and original image $I_p(r,c)$, while M and N is the height and width of the image respectively.

Another way is quality checking at the receiver side, such as after passing through a noisy medium cipher, an image may get blurred or lose some data. Therefore, a trustworthy encryption scheme should be able to recuperate the original image without losing too much substantial information. Figure 15a–f shows the different test images with different cropped portion. While Figure 15g–l are the retrieved images, we can see the retrieved images are easily recognizable and carry good information even after clipping 1/2. While the less clipping gave a much better result. Table 13 listed the PSNR and MSE comparison with [29] of test image Lena with different proportion of cropping. The values show that our algorithm performs better when increased the clipping portion while values are comparable for the less clipping.

Table 13. PSNR and MSE comparison under different cropping size.

Cropped Size	Proposed 3C3R		Ref. [29]	
	PSNR	**MSE**	**PSNR**	**MSE**
1/2	**12.88**	**3121.1**	11.58	4578.34
1/4	**14.722**	**2192.2**	14.59	2289.90
1/8	16.75	1375.9	**17.57**	**1155.32**
1/16	19.25	772.65	**20.57**	**579.98**

Measuring the difference between the cipher image and original image is another way to evaluate the quality of the color images. So, for this purpose the PSNR can be viewed as a security evaluation parameter. The encryption effect is consider better if the value of PSNR is lower. Table 14 listed the PSNR value between $plain \mapsto decrypted$ and $plain \mapsto ciphered$ images. The comparison was also made with some well-known algorithms and values are listed in the particular table. We can see from the table that the PSNR value for the different test images is $PSNR \leq$**8.10** that is lower than Refs. [15,23,46,90,91] except the test image baboon in which Ref. [15] gave the lowest PSNR value between plain image and ciphered image. Thus, based on the PSNR value test, we can say that our **3C3R** performs very well and can guaranty more security in comparison to other algorithms.



(**a**) Lena cropped (**b**) Panda Cropped (**c**) House Cropped

(**d**) Baboon Cropped (**e**) Panda Cropped (**f**) Lena Cropped

(**g**) Retrieved of (a) (**h**) Retrieved of (b) (**i**) Retrieved of (c)

(**j**) Retrieved of (d) (**k**) Retrieved of (e) (**l**) Retrieved of (f)

Figure 15. Occlusion attack test. (**a**) Cropped image of Lena; (**b**) Cropped image of Panda; (**c**) Cropped image of House; (**d**) Cropped image of baboon; (**e**) Cropped image of Panda; **(f)** Cropped image of Lena; (**g**) Retrieved image of (a); (**h**) Retrieved image of (b); (**i**) Retrieved image of (c); (**j**) Retrieved image of (d); (**k**) Retrieved image of (e); (**l**) Retrieved image of (f).

Table 14. PSNR between Plain (O) and cipher (C) image & Plain and Decrypted(D) image.

Algorithm	PSNR	Lena	Baboon	Couple	Panda	Vegetables
3C3R	O to D	∞	∞	∞	∞	∞
	O to C	**8.1020**	8.011	**6.2414**	**7.7028**	**6.8459**
Ref. [15]	O to C	8.1300	**7.8569**	7.4892	7.7410	7.4395
Ref. [46]	O to D	96.295	-	-	-	-
	O to C	9.0348	-	-	-	-
Ref. [23]	O to C	8.6878	-	-	-	-
Ref. [92]	O to C	9.0486	-	-	-	-
Ref. [90]	O to C	8.3655	8.8532	-	-	-
Ref. [91]	O to C	8.2522	8.8223	-	-	-

5.8. Local and Shannon Information Entropy

The information entropy (IE) defines the degree of disorder or chaos in an encryption system through the gray value probability. IE for the image can be defined as follow:

Let an information source be a τ, then IE can be computed as follows.

$$H(\tau) = \sum_{i=0}^{2^n-1} \rho(\tau_i) log_{10} \frac{1}{\rho(\tau_i)} \tag{43}$$

Here $\rho(\tau_i)$ depicts the probability of the symbol τ_i. The ideal IE value for the image with gray intensity level of 2^8 is **8** [93]. So, it means the closer the IE value, the more is the randomness of an image, and as a result less information will be revealed by the particular encryption scheme. Tables 15 and 16 enlisted the entropy value of some famous test images and their comparison with some earlier encryption algorithms. The table values are the evidence of **IE** $\geq$ **7.996**, that is close enough to the ideal value **8.0**. While for Playboy, full white and Full black **3C3R** achieved the ideal value **IE = 8.00**. IE values of all the test images are higher than Refs. [82,87–89].

In [94] a new image uncertainty test introduced by means of Shannon entropy over the native image-blocks. The **(k, T_P)** Shannon entropy measure concerning local image blocks can be calculated by the following method:

Step 1: Select the non-overlapping image blocks randomly i.e., $B_1, B_2, B_3, \ldots, B_k$ with T_P Pixels within the ciphered or test image I with intensity scales L.

Step 2: Compute Shannon entropy for all $i \in (1, 2, 3, \ldots, k)$ by using Equation (44).

Step 3: Calculate the Shannon entropy sample mean over these **k** image blocks $B_1, B_2, B_3, \ldots, B_k$ by the following equation.

$$\bar{H}_{(k,T_P)}(B) = \sum_{i=1}^{k} \frac{H(B_i)}{k} \tag{44}$$

The local Shannon entropy value was calculated for the ciphered images. Firstly, non-overlapping image blocks with $k = 32$ and $\mathbf{T_P}$ **= 1936** pixels are randomly selected from the ciphered images. As the experiential value of local Shannon entropy must fall within the confidence interval i.e., **[7.9019, 7.9030]**, concerning the α-level sureness equal to the 0.05. Table 16 listed the Shannon entropy and global entropy values and also the comparison of Shannon entropy with Ref. [95]. The Shannon entropy value of the ciphered image of the proposed **3C3R** fully falls within the desired range. Thus, based on the ideal **IE** values, we can say the ciphered image generated by the proposed **3C3R** carries more haphazardness and as a result, assures more security.

Table 15. Information Entropy Comparison.

Algorithm	Test Image	Plain R	Plain G	Plain B	Ciphered R	Ciphered G	Ciphered B
Our 3C3R	Lena	7.568	7.058	6.779	**7.997**	**7.997**	**7.996**
	Pepper	7.338	7.496	7.058	**7.999**	**7.999**	**7.999**
	Baboon	7.706	7.474	7.752	**7.999**	**7.999**	**7.999**
	Panda	7.708	7.552	7.726	**7.996**	**7.997**	**7.997**
	Vegetable	7.905	7.674	6.345	**7.999**	**7.999**	**7.999**
Ref. [82]	Lena	7.293	7.581	7.085	7.989	7.989	7.989
	Pepper	7.331	7.524	7.079	7.989	7.988	7.989
	Baboon	7.700	7.512	7.765	7.989	7.989	7.988
	Panda	7.711	7.627	7.793	7.988	7.989	7.989
	Vegetable	7.797	7.821	7.359	7.989	7.989	7.989
Ref. [87]	Lena	-	-	-	7.987	7.987	7.986
Ref. [88]	Lena	-	-	-	7.927	7.974	7.970
Ref. [89]	Lena	-	-	-	7.973	7.975	7.971
Ref. [96]	Lena	-	-	-	7.987	7.988	7.987

Table 16. Global Entropy and Shannon Entropy.

Images	Our 3C3R Shannon_C	$\text{Global}_{R,G,B}$ Plain	$\text{Global}_{R,G,B}$ Cipher	Ref. [95] Shannon_C
Vegetables	7.9028	7.496	7.999	-
Bridge	7.9023	7.876	7.999	-
Opera	7.9018	7.798	7.999	-
Bike	7.9023	7.441	7.999	-
Airplane	7.9027	6.665	7.9991	-
House	7.9027	7.4858	7.9998	7.9021
Playboy	7.9030	0.5257	8.00	-
Nike	7.9029	1.1969	7.9998	-
Skull	7.9019	7.483	7.999	-
Pepper	7.9029	7.669	7.999	7.9024
Fruit	7.9019	7.532	7.998	-
Baboon	7.9025	7.7624	7.9998	7.9023
Lena	7.9028	7.4517	7.9991	7.9024
All White	7.9020	0	8.00	-
Full Black	7.9027	0	8.00	-

5.9. Gray Value Degree (GVD) Analysis

The gray difference degree or GVD is another statistical test of haphazardness that can be found by comparing the plain image and the ciphered image. The ideal value is **1**, so closer the value the better is the security. GVD can be computed by the following Equation;

$$G_D(r,c) = \sum [G(r,c) - G(r',c')] \tag{45}$$

whereas $G(r,c)$ symbolizes the gray score at position (r,c) and (r',c') is as given below.

$$(r',c') = \begin{cases} (r-1,c) \\ (r+1,c) \\ (r,c-1) \\ (r,c+1) \end{cases} \tag{46}$$

The average neighborhood gray difference of the whole image can be computed as follows.

$$AV_{erage}[G_D(r,c)] = \frac{\sum_{r=2}^{M-1}\sum_{c=2}^{N-1} G_D(r,c))}{(M-2)(N-2)} \tag{47}$$

$$AV_{erage}GVD = \frac{AV'[G_D(r,c)] - AV[G_D(r,c)]}{AV'[G_D(r,c)] + AV[G_D(r,c)]} \tag{48}$$

In these equations, $AN^{'}$ and AN, denotes the average neighborhood gray value; but the former represents after encrypting and the later represents before encryption. The final GVD value termed as gray value degree and it will be **1** if the two images are completely different or else it will be **0**, if the two images are same. The GVD score of the plain and encrypted images of **USC-SIPI** database are shown in Table 17. The GVD score for most of the test images is GVD $\geq$ **0.907** for each R, G and B channel which shows that the plain and encrypted images are entirely different. The listed results also show that **3C3R** ensures more security for images as compared to Refs. [29,69,97] except for the image **4.1.04** in which Ref. [29] has the higher value as compared to **3C3R**.

Table 17. GVD Comparison.

USC-SIPI	Our 3C3R GVD			Ref. [29] GVD			Ref. [69] GVD			Ref. [97] GVD		
	Red	Green	Blue	Red	Green	Blue	Red	Green	Blue	Red	Green	Blue
4.1.01	**0.981**	**0.984**	0.923	0.977	0.979	**0.975**	-	-	-	-	-	-
4.1.02	**0.984**	**0.989**	**0.987**	0.978	0.979	0.979	-	-	-	-	-	
4.1.03	0.886	0.774	0.814	**0.978**	**0.976**	**0.977**	-	-	-	-	-	-
4.1.04	**0.988**	**0.978**	0.966	0.979	0.975	**0.980**	-	-	-	-	-	-
4.1.05	**0.983**	0.921	**0.975**	0.982	**0.966**	0.969	-	-	-	-	-	-
4.1.06	**0.986**	**0.998**	**0.976**	0.943	0.912	0.934	-	-	-	-	-	-
4.1.08	0.700	**0.979**	0.921	**0.985**	0.973	**0.983**	-	-	-	-	-	-
4.2.01	0.958	**0.998**	**0.986**	**0.989**	0.968	0.977	-	-	-	-	-	-
4.2.03	**0.986**	0.988	**0.992**	0.936	0.906	0.903	-	-	-	0.9801	**0.989**	0.9865
4.2.07	**0.995**	**0.953**	0.847	0.976	0.948	**0.974**	-	-	-	-	-	-
Lena	0.960	**0.9856**	**0.9874**	-	-	-	**0.9805**	0.9812	**0.9876**	0.9701	0.9700	0.9690

5.10. Performance Comparison

Performance and encryption time are an important characteristic of any image encryption algorithm. Like in the Chaos-based encryption schemes, number of permutation and diffusion rounds have a direct impact over the encryption time. Similarly for color image rule processing scheme over the RGB channels has a direct relation with the encryption time. Because of the parallel processing characteristics of cellular automata and DNA sequence the proposed **3C3R** takes less encryption time as compared to chaos-based schemes. Table 18 enlisted the time comparison of our **3C3R** with the Refs. [79,81,98,99] that possess a satisfactory security level. The least encryption and decryption time was taken by the proposed **3C3R** shows that our algorithm gives satisfactory results with minimum times. Except that Table 19 listed the comparison with the recently introduced algorithms. The table values in bold fonts clearly concludes that our **3C3R** overall performing much better than Refs. [56,62,76,91]. Hence, our **3C3R** assures better image security.

Table 18. Performance comparison.

Image Size	Proposed 3C3R	Ref. [98]	Ref. [81]	Ref. [99]	Ref. [57]
256 × 256	**3.321** s	4.7795 s	-	3.617 s	-
512 × 512	**6.713** s	8.670 s	8.308 s	14.811 s	16.170 s

Table 19. General Random terms Comparison With Some Most Recent Algorithms.

Algorithms	**Images**		**Entropy Comparison**		
			Red	**Green**	**Blue**
S.S Moafimadani[2019] [100]	Full white		7.9994	7.9994	7.9993
	Full black		7.9993	7.9994	7.9993
Z. Liu[2019] [51]	Full white		7.9914	7.9942	7.9856
	Full black		7.9965	7.9948	7.9955
3C3R	Full white		**8.00**	**8.00**	**8.00**
	Full black		**8.00**	**8.00**	**8.00**
M. Wang[2019] [101]	Lena		7.9970	7.9973	**7.9973**
3C3R	Lena		**7.9972**	**7.9974**	7.9967
			Histogram Variance		
X. Chai[2019] [62]	House		Red	Green	Blue
			1070.2	1231.2	**941.65**
3C3R	House		**998.60**	**1107.34**	1046.96
			Correlation Comparison		
W. Zhang[2019] [102]	Pepper		Horizontal	Vertical	Diagonal
		Red	0.003853	**0.001284**	−**0.001832**
		Green	−**0.000912**	0.001460	0.002366
		Blue	−0.001647	0.006770	−**0.000366**
3C3R	Pepper	Red	−**0.001994**	−0.007431	−0.009151
		Green	−0.004029	−**0.001068**	−**0.001439**
		Blue	**0.0011452**	**0.00081563**	−0.005897
			UACI Comparison		
S.Suri[2019] [56]	Lena		32.1752		
	Baboon		30.3547		
3C3R	Lena		**33.45**		
	Baboon		**33.43**		
			NPCR, UACI Comparison		
K.A.K Patro[2019] [103]	Lena	NPCR	99.6314		
		UACI	**33.551**		
3C3R	Lena	NPCR	**99.978**		
		UACI	33.45		
			Corr. Comparison Of 1000 pixels		
P. Ramasamy[2019] [76]	Lena		Horizontal	Vertical	Diagonal
			−0.0237	−0.0178	−0.0284
3C3R	Lena		−**0.0027**	−**0.00054**	−**0.0013**
			PSNR Comparison		
X. Liu[2019] [91]	Lena	O to C	8.2522		
	Baboon	O to C	8.8223		
3C3R	Lena	O to C	**8.1020**		
	Baboon	O to C	**8.011**		

6. Conclusions and Future Work

To conquer the issue of low sensitivity to the secret key or low security against known plain or cipher-text attacks, this paper introduced a **3C3R** robust image encryption algorithm with adequate security level against well known attacks. Fully uniform histogram and ideal cumulative entropy **8.00** for some images are the proof of robustness and better security of an image by **3C3R**. Unlike most encryption methodologies in which the same encryption method or rules structure followed for all channels, this paper introduced a novel encryption method comprising different encryption strategy for each channels. Block bit inversion (binary) for the red channel, **VN** and **RVN** (cellular) structure-based pixel alteration for the green channel, and state machine-based **SM-DNA** rule allocation for the blue channel. Experimental results proved that our **3C3R** algorithm is highly subtle to the secret key along with better security. The proposed **3C3R** can keep different types of images safe and secure from attackers. **3C3R** outperforms state of the art algorithms in terms of encryption performance and image security. The experimental results also proved that the **3C3R** is robust against well-known attacks such as brute-force attacks, occlusion attacks, chosen/known plaintext attacks, and differential attacks. Therefore, we can say that **3C3R** is the algorithm of the current era requirement and hence it has potential applications in multimedia communication.

Author Contributions: S.K. conceived, designed as well as performed all the experiments including writing the manuscript; L.H. supervised the research and delivered funding acquisition; G.M. designed the methodology; B.G. worked over editing and visualization; H.U. performed formal analysis.

Funding: This research was funded by the National Natural Science Foundation of China grant number "N0:61272033, 61572222".

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the data collection, data analyses, or interpretation of results; as well as in the writing of the manuscript, and in the decision of publishing the results.

References

1. Wong, K.; Kwok, B.; Law, W. A fast image encryption scheme based on chaotic standard map. *Phys. Lett. A* **2008**, *372*, 2645–2652. [CrossRef]
2. Gehani, A.; LaBean, T.H.; Reif, J.H. DNA-based cryptography. In *Aspects of Molecular Computing*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 54, pp. 233–249.
3. Lian, S.G. *Multimedia Content Encryption: Techniques and Applications*; Auerbach Publication Taylor & Francis Group: Boca Raton, FL, USA, 2008; ISBN 1420065270.
4. Wang, Y.; Wong, K.; Liao, X.; Xiang, T.; Chen, G. A chaos-based image encryption algorithm with variable control parameters. *Chaos Solitons Fract.* **2009**, *41*, 1773–1783. [CrossRef]
5. Guan, Z.H.; Huang, F.; Guan, W. Chaos-based image encryption algorithm. *Phys. Lett. A* **2005**, *346*, 153–157. [CrossRef]
6. Yavuz, E.; Yazici, R.; Kasapbasi, M.C.; Yamac, C. A chaos-based image encryption algorithm with simple logical functions. *Comput. Electr. Eng.* **2016**, *54*, 471–483. [CrossRef]
7. Dou, Y.; Liu, X.; Fan, H.; Li, M. Cryptanalysis of a DNA and chaos based image encryption algorithm. *Optik* **2017**, *145*, 456–464. [CrossRef]
8. Chen, J.; Zhang, Y.; Qi, L.; Fu, C.; Xu, L. Exploiting chaos-based compressed sensing and cryptographic algorithm for image encryption and compression. *Opt. Laser Technol.* **2018**, *99*, 238–248. [CrossRef]
9. Jeng, F.G.; Huang, W.L.; Chen, T.H. Cryptanalysis and improvement of two hyper chaos based image encryption schemes. *Signal Process.* **2015**, *34*, 45–51. [CrossRef]
10. Qin, Y.; Wang, Z.; Wang, H.; Gong, Q. Binary image encryption in a joint transform correlator scheme by aid of run length encoding and QR code. *Opt. Laser Technol.* **2018**, *103*, 93–98. [CrossRef]
11. Kumar, R.; Bhaduri, B. Optical image encryption using Kronecker product and hybrid phase masks. *Opt. Laser Technol.* **2017**, *95*, 51–55. [CrossRef]
12. Chen, H.; Tanougast, C.; Liu, Z.; Blondel, W.; Hao, B. Optical hyperspectral image encryption based on improved Chirikov mapping and gyrator transform. *Opt. Lasers Eng.* **2018**, *107*, 62–70. [CrossRef]

13. Li, X.; Meng, X.; Yang, X.; Wang, Y.; Yin, Y.; Peng, X.; He, W.; Dong, G.; Chen, H. Multiple-image encryption via lifting wavelet transform and XOR operation based on compressive ghost imaging scheme. *Opt. Lasers Eng.* **2018**, *102*, 106–111. [CrossRef]
14. Chai, X.; Gan, Z.; Yang, K.; Chen, Y.; Liu, X. An image encryption algorithm based on the memristive hyper-chaotic system, cellular automata and DNA sequence operations. *Signal Process.* **2017**, *52*, 6–19.
15. Wu, X.; Wang, K.; Wang, X.; Kan, H.; Kurths, J. Color image DNA encryption using NCA map based CML and one time keys. *Signal Process.* **2018**, *148*, 272–287. [CrossRef]
16. Talarposhti, K.M.; Jamei, M.K. A secure image encryption method based on dynamic harmony search (DHS) combined with chaotic map. *Opt. Lasers Eng.* **2016**, *81*, 21–34. [CrossRef]
17. Lian, S. A block cipher based on chaotic neural networks. *Neurocomputing* **2009**, *72*, 1296–1301. [CrossRef]
18. Zhang, X.; Chen, W. A new chaotic algorithm for image encryption. In Proceedings of the 2008 International Conference on Audio, Language and Image Processing, Shanghai, China, 7–9 July 2008; pp. 889–892. [CrossRef]
19. Xue, X.L.; Zhang, Q. An image fusion encryption algorithm based on DNA sequence and multi-chaotic maps. *J. Comput. Theor. Nanosci.* **2010**, *7*, 397–403. [CrossRef]
20. Zhou, Y.; Bao, L.; Chen, C.L.P. A new 1D chaotic system for image encryption. *Signal Process.* **2014**, *97*, 172–182. [CrossRef]
21. Hua, Z.; Zhou, Y.; Pun, C.M. Image encryption using 2D Logistic-Sine chaotic map. In Proceedings of the 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), San Diego, CA, USA, 5–8 October 2014; pp. 3229–3234.
22. Zamani, S.; Javanmard, M.; Jafarzadeh, N. A novel image encryption scheme based on hyper chaotic systems and fuzzy cellular automata. In Proceedings of the 2014 22nd Iranian Conference on Electrical Engineering (ICEE), Tehran, Iran, 20–22 May 2014; pp. 1136–1141.
23. Norouzi, B.; Mirzakuchaki, S. A fast color image encryption algorithm based on hyper-chaotic systems. *Nonlinear Dyn.* **2014**, *78*, 995–1015. [CrossRef]
24. Khan, S.; Han, L.; Lu, H.; Butt, K.K.; Bachira, G.; Khan, N. A New Hybrid Image Encryption Algorithm Based on 2D-CA, FSM-DNA Rule Generator, and FSBI. *IEEE Access* **2019**, *7*, 81333–81350. [CrossRef]
25. Huang, Q.; Li, G. Research on the Application of Image Encryption Technology Based on 7 Dimensional CNN Hyper Chaos. In Proceedings of the 2016 International Conference on Smart City and Systems Engineering (ICSCSE), Hunan, China, 25–26 November 2016; pp. 531–534.
26. Liu, H.J.; Wang, X.Y.; Kadir, A. Image encryption using DNA complementary rule and chaotic maps. *Appl. Soft Comput.* **2012**, *12*, 1457–1466. [CrossRef]
27. Liu, W.H.; Sun, K.H.; He, Y.; Yu, M.Y. Color image encryption using three-dimensional sine ICMIC modulation map and DNA sequence operations. *Int. J. Bifurc. Chaos* **2017**, *27*, 1750171. [CrossRef]
28. Niyat, A.Y.; Moattar, M.H.; Torshiz, M.N. Color image encryption based on hybrid hyper chaotic system and cellular automata. *Opt. Lasers Eng.* **2017**, *90*, 225–237. [CrossRef]
29. Rehman, A.; Liao, X.; Ashraf, R.; Ullah, S.; Wang, H. A color image encryption technique using exclusive-OR with DNA complementary rules based on chaos theory and SHA-2. *Optik* **2018**, *159*, 348–367. [CrossRef]
30. 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]
31. Hoang, T.M.; Thanh, H.X. Cryptanalysis and security improvement for a symmetric color image encryption algorithm. *Optik* **2018**, *155*, 366–383. [CrossRef]
32. Zhou, G.; Zhang, D.; Liu, Y.; Yuan, Y.; Liu, Q. A novel image encryption algorithm based on chaos and Line map. *Neurocomputing* **2015**, *169*, 150–157. [CrossRef]
33. Chen, L.; Ma, B.; Zhao, X.; Wang, S. Differential cryptanalysis of a novel image encryption algorithm based on chaos and Line map. *Nonlinear Dyn.* **2016**, *84*, 1–11. [CrossRef]
34. Zhang, W.; Yu, H.; Zhao, Y.L.; Zhu, Z.L. Image encryption based on three-dimensional bit matrix permutation. *Signal Process.* **2016**, *118*, 36–50. [CrossRef]
35. Wu, J.; Liao, X.; Yang, B. Cryptanalysis and Enhancements of Image Encryption Based on Three-dimensional Bit Matrix Permutation. *Signal Process.* **2018**, *142*, 292–300. [CrossRef]
36. Huang, X. Image encryption algorithm using chaotic Chebyshev generator. *Nonlinear Dyn.* **2012**, *67*, 2411–2417. [CrossRef]

37. Wang, X.; Luan, D.; Bao, X. Cryptanalysis of an image encryption algorithm using Chebyshev generator. *Digit. Signal Process.* **2014**, *25*, 244–247. [CrossRef]
38. Chen, J.; Zhu, Z.; Fu, C.; Zhang, L.; Zhang, Y. An efficient image encryption scheme using lookup table-based confusion and diffusion. *Signal Process.* **2015**, *81*, 1151–1166. [CrossRef]
39. Gao, T.G.; Chen, Z.Q. A new image encryption algorithm based on hyper chaos. *Phys. Lett. A* **2008**, *372*, 394–400. [CrossRef]
40. Hu, G.; Xiao, D.; Wang, Y.; Li, X. Cryptanalysis of a chaotic image cipher using Latin square-based confusion and diffusion. *Nonlinear Dyn.* **2017**, *88*, 1305–1316. [CrossRef]
41. Rhouma, R.; Belghith, S. Cryptanalysis of a new image encryption algorithm based on hyperchaos. *Phys. Lett. A* **2008**, *372*, 5973–5978. [CrossRef]
42. Liu, Y.; Tong, X.; Ma, J. Image encryption algorithm based on hyper-chaotic system and dynamic S-box. *Multimed. Tools Appl.* **2016**, *75*, 7739–7759. [CrossRef]
43. Tong, X.; Cui, M. Image encryption with compound chaotic sequence cipher shifting dynamically. *Image Vis. Comput.* **2008**, *26*, 843–850. [CrossRef]
44. Zhang, X.; Nie, W.; Ma, Y.; Tian, Q. Cryptanalysis and improvement of an image encryption algorithm based on hyper-chaotic system and dynamic S-box. *Multimed. Tools Appl.* **2017**, *76*, 1–19. [CrossRef]
45. CLi, Q.; Li, S.J.; Chen, G.R.; Halang, W.A. Cryptanalysis of an image encryption scheme based on a compound chaotic sequence. *Image Vis. Comput.* **2009**, *27*, 1035–1039.
46. Zhu, C.X. A novel image encryption scheme based on improved hyper chaotic sequences. *Opt. Commun.* **2012**, *285*, 29–37. [CrossRef]
47. Pak, C.; Huang, L. A new color image encryption using combination of the 1d chaotic map. *Signal Process.* **2017**, *138*, 129–137. [CrossRef]
48. Li, C.Q.; Liu, Y.S.; Xie, T.; Chen, M.Z.Q. Breaking a novel image encryption scheme based on improved hyper chaotic sequences. *Nonlinear Dyn.* **2013**, *73*, 2083–2089. [CrossRef]
49. Wang, H.; Xiao, D.; Chen, X.; Huang, H. Cryptanalysis and Enhancements of Image Encryption Using Combination of the 1D Chaotic Map. *Signal Process.* **2018**, *144*, 444–452. [CrossRef]
50. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*, 3rd ed.; Pearson Prentice Hall: Upper Saddle River, NJ, USA, 2008.
51. Liu, Z.; Wu, C.; Wang, J.; Hu, Y. A Color Image Encryption Using Dynamic DNA and 4-D Memristive Hyper-Chaos. *IEEE Access* **2019**, *7*, 78367–78378. [CrossRef]
52. Zhou, Y.; Cao, W.; Chen, C. Image encryption using binary bit-plane. *Signal Process.* **2014**, *100*, 197–207. [CrossRef]
53. Enayatifar, R.; Abdullah, A.H.; Isnin, I.F.; Altameem, A.; Lee, M. Image encryption using a synchronous permutation-diffusion technique. *Opt. Lasers Eng.* **2017**, *90*, 146–154. [CrossRef]
54. Kumar, M.; Kumar, S.; Budhiraja, R.; Das, M.K.; Singh, S. Intertwining logistic map and Cellular Automata based color image encryption model. In Proceedings of the 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT), New Delhi, India, 11–13 March 2016; pp. 618–623. [CrossRef]
55. 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]
56. Suri, S.; Vijay, R. A synchronous intertwining logistic map-DNA approach for color image encryption. *J. Ambient Intell. Humaniz. Comput.* **2019**, *10*, 2277–2290. [CrossRef]
57. Brindha, M.; Gounden, N.A. A chaos based image encryption and lossless compression algorithm using hash table and chinese remainder theorem. *Appl. Soft Comput.* **2016**, *40*, 379–390. [CrossRef]
58. Hu, T.; Liu, Y.; Gong, L.; Guo, S.; Yuan, H. Chaotic image crypto-system using DNA deletion and DNA insertion. *Signal Process.* **2017**, *134*, 234–243. [CrossRef]
59. Ye, G.; Huang, X. An efficient symmetric image encryption algorithm based on an intertwining logistic map. *Neuro-Computing* **2017**, *251*, 45–53. [CrossRef]
60. Li, C. Cracking a hierarchical chaotic image encryption algorithm based on permutation. *Signal Process.* **2015**, *118*, 203–210. [CrossRef]
61. Castro, J.C.H.; Sierra, J.M.; Seznec, A.; Izquierdo, A.; Ribagorda, A. The strict avalanche criterion randomness test. *Math. Comput. Simul.* **2005**, *68*, 1–7. [CrossRef]

62. Chai, X.; Fu, X.; Gan, Z.; Lu, Y.; Chen, Y. A color image crypto-system based on dynamic DNA encryption and chaos. *Signal Process.* **2019**, *155*, 44–62. [CrossRef]
63. Zhang, Y.; Xiao, D. An image encryption scheme based on rotation matrix bit-level permutation and block diffusion. *Commun. Nonlinear Sci. Numer. Simul.* **2014**, *19*, 74–82. [CrossRef]
64. Huang, L.; Cai, S.; Xiao, M.; Xiong, X. A Simple Chaotic Map-Based Image Encryption System Using Both plaintext Related Permutation and Diffusion. *Entropy* **2018**, *20*, 535. [CrossRef]
65. Wang, X.; Zhang, H.L. A color image encryption with heterogeneous bit-permutation and correlated chaos. *Opt. Commun.* **2015**, *342*, 51–60. [CrossRef]
66. Zhu, Z.; Zhang, W.; Wong, K.; Yu, H. A chaos-based symmetric image encryption scheme using a bit-level permutation. *Inf Sci.* **2011**, *181*, 1171–1186. [CrossRef]
67. Song, C.; Qiao, Y.; Zhang, X. An image encryption scheme based on new spatio-temporal chaos. *Optik* **2013**, *124*, 3329–3334. [CrossRef]
68. Zhang, Y.; Wang, X. A symmetric image encryption algorithm based on mixed linear-nonlinear coupled map lattice. *Inf. Sci.* **2014**, *273*, 329–351. [CrossRef]
69. Wei, X.; Guo, L.; Zhang, Q.; Zhang, J.; Lian, S. A novel color image encryption algorithm based on DNA sequence operation and hyper-chaotic system. *J. Syst. Softw.* **2012**, *85*, 290–299. [CrossRef]
70. 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]
71. Zhang, Q.; Wei, X. A novel couple images encryption algorithm based on DNA sub-sequence operation and chaotic system. *Optik* **2013**, *124*, 6276–6281. [CrossRef]
72. Enayatifar, R.; Abdullah, A.H.; Isnin, I.F. Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence. *Opt. Lasers Eng.* **2014**, *56*, 83–93. [CrossRef]
73. Zhen, P.; Zhao, G.; Min, L.; Jin, X. Chaos-based image encryption scheme combining DNA coding and entropy. *Multimed. Tools Appl.* **2016**, *75*, 6303–6319. [CrossRef]
74. Xu, L.; Li, Z.; Li, J.; Hua, W. A novel bit-level image encryption algorithm based on chaotic maps. *Opt. Lasers Eng.* **2016**, *78*, 17–25. [CrossRef]
75. Wang, X.Y.; Zhang, Y.Q.; Bao, X.M. A novel chaotic image encryption scheme using DNA sequence operations. *Opt. Lasers Eng.* **2015**, *73*, 53–61. [CrossRef]
76. Ramasamy, P.; Ranganathan, V.; Kadry, S.; Damasevicius, R.; Blazauskas, 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]
77. Wu, X.; Kurths, J.; Kan, H. A robust and lossless DNA encryption scheme for color images. *Multimed. Tools Appl.* **2017**, *77*, 12349–12376. [CrossRef]
78. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based crypto-systems. *Int. J. Bifurc. Chaos* **2006**, *16*, 2129–2151. [CrossRef]
79. Tong, X.J.; Zhang, M.; Wang, Z.; Ma, J. A joint color image encryption and compression scheme based on hyper-chaotic system. *Nonlinear Dyn.* **2016**, *84*, 2333–2356. [CrossRef]
80. 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]
81. Wu, X.; Zhu, B.; Hu, Y.; Ran, Y. A novel color image encryption scheme using rectangular transform-enhanced chaotic tent maps. *IEEE Access* **2017**, *5*, 6429–6436.
82. Van den Assem, R.; van Elk, W. A chosen-plaintext attack on the microsoft basic protection. *Comput. Secur.* **1986**, *5*, 36–45. [CrossRef]
83. 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]
84. Wu, X.; Kan, H.; Kurths, J. A new color image encryption scheme based on DNA sequences and multiple improved 1D chaotic maps. *Appl. Soft Comput. J.* **2015**, *37*, 24–39. [CrossRef]
85. Fu, C.; Chen, J.; Zou, H.; Meng, W.; Zhan, Y.; Yu, Y. A chaos-based digital image encryption scheme with an improved diffusion strategy. *Opt. Express* **2012**, *20*, 2363–2378. [CrossRef] [PubMed]
86. Boopathy, D.; Sundaresan, M. A novel multi-dimensional encryption technique to secure the grayscale images and color images in public cloud storage. *Innov. Syst. Softw. Eng.* **2019**, *15*, 43–64. [CrossRef]
87. Liu, H.; Wang, X.; Kadir, A. Color image encryption using Choquet fuzzy integral and hyper chaotic system. *Opt. Int. J. Light Electron. Opt.* **2013**, *124*, 3527–3533. [CrossRef]

88. Kadir, A.; Hamdulla, A.; Guo, W. Color image encryption using skew tent map and hyper chaotic system of 6th-order CNN. *Optik* **2014**, *125*, 1671–1675. [CrossRef]
89. Rhouma, R.; Meherzi, S.; Belghith, S. OCML-based colour image encryption. *Chaos Solitons Fract.* **2009**, *40*, 309–318. [CrossRef]
90. Ahmad, J.; Hwang, S.O. A secure image encryption scheme based on chaotic maps and affine transformation. *Multimed. Tools Appl.* **2016**, *75*, 13951–13976. [CrossRef]
91. Liu, X.; Xiao, D.; Xiang, Y. Quantum Image Encryption Using Intra and Inter Bit Permutation Based on Logistic Map. *IEEE Access* **2019**, *7*, 6937–6946. [CrossRef]
92. Taneja, N.; Raman, B.; Gupta, I. Combinational domain encryption for still visual data. *Multimed. Tool Appl.* **2012**, *59*, 775–793. [CrossRef]
93. Wang, X.; Teng, L.; Qin, X. A novel colour image encryption algorithm based on chaos. *Signal Process.* **2012**, *92*, 1101–1108. [CrossRef]
94. Wu, Y.; Zhou, Y.; Saveriades, G.; Agaian, S.; Noonan, J.P.; Natarajan, P. Local Shannon entropy measure with statistical tests for image randomness. *Inf. Sci.* **2013**, *222*, 323–342. [CrossRef]
95. 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]
96. Liu, H.; Wang, X. Color image encryption using spatial bit level permutation and high-dimension chaotic system. *Opt. Commun.* **2011**, *284*, 3895–3903. [CrossRef]
97. Kalpana, J.; Murali, P. An improved color image encryption based on multiple DNA sequence operations with DNA synthetic image and chaos. *Opt. Int. J. Light Electron. Opt.* **2015**, *126*, 5703–5709. [CrossRef]
98. Luo, Y.; Zhou, R.; Liu, J.; Qiu, S.; Cao, Y. An efficient and self-adapting colour image encryption algorithm based on chaos and interactions among multiple layers. *Multimed. Tools Appl.* **2018**, *77*, 26191–26217. 018-5844-5. [CrossRef]
99. Pareek, N.K.; Patidar, V.; Sud, K.K. Image encryption using chaotic logistic map. *Image Vis. Comput.* **2006**, *24*, 926–934. [CrossRef]
100. Moafimadani, S.S.; Chen, Y.; Tang, C. A New Algorithm for Medical Color Images Encryption Using Chaotic Systems. *Entropy* **2019**, *21*, 577. [CrossRef]
101. Wang, M.; Wang, X.; Zhang, Y.; Zhou, S.; Zhao, T.; Yao, N. A novel chaotic system and its application in a color image cryptosystem. *Opt. Lasers Eng.* **2019**, *121*, 479–494. [CrossRef]
102. Zhang, W.; Zhu, Z.; Yu, H. A Symmetric Image Encryption Algorithm Based on a Coupled Logistic-Bernoulli Map and Cellular Automata Diffusion Strategy. *Entropy* **2019,** *21*, 504. [CrossRef]
103. Patro, K.A.K.; Acharya, B. An efficient colour image encryption scheme based on 1-D chaotic maps. *J. Inf. Secur. Appl.* **2019**, *46*, 23–41. [CrossRef]