ECTI Association

**ECTI Transactions on Computer and Information Technology**
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

# A Novel Image Encryption Technique with Four Stage Bit-Interspersing and A 4D-Hyperchaotic System

Subhashish Pal[1], Ansuman Mahanty [2], Arghya Pathak[3], Jayashree Karmakar[4], Hrishikesh Mondal [5] and Mrinal Kanti Mandal[6]

## ABSTRACT

This paper introduces a new technique of color image encryption that includes four stages of bit interspersing and a 4D-hyperchaotic system. At first, the pixel values for different RGB channels of a plane color image were represented by an 8-bit binary number. The bit interspersing operation has been implemented on the bit stream generated by considering a particular block size for each color channel and reshaping it to the original size. The cipher image was then constructed by performing a bit XOR operation on the resultant image and the chaotic sequences generated by a 4D hyperchaotic system. The initial state variables of the said chaotic system have been developed from a 32-character secret key. This operation has been repeated three times, considering different block sizes of the newly generated cipher image after each stage. The main strengths of the proposed algorithm are the bit interspersing and image-dependent chaotic key base pixel substitution by bitwise XOR operation. A set of standard security tests are conducted to test the reliability of the suggested encryption method. Comparing the crypto-parameters to other recent works, we find that the proposed algorithm is better than the other methods.

*Article information:*

**Keywords:** Bit interspersing, Cryptography, Chaos, Hyperchaotic system, Encryption, Decryption

**Article history:**
Received: August 29, 2022
Revised: December 9, 2022
Accepted: January 21, 2023
Published: March 4, 2023 (Online)

**DOI:** 10.37936/ecti-cit.2023171.249733

## 1. INTRODUCTION

The growth of the internet and communication infrastructure has increased the volume and variety of shared data, making it more difficult for cryptosystem designers to guarantee the privacy of transmitted information. Many researchers have reported essential contributions to data security. Cryptography is a subset of data security systems that encrypts data before sending it across a network so that only the intended recipient can decipher it. In cryptography, keys are essential tools for encrypting and decrypting information. When both the keys are the same, that type of encryption is termed symmetric key cryptography, and if they are different, it is called asymmetric key cryptography [1].

In modern cryptography, the encryption algorithm plays a pivotal role in determining the quality of the cryptosystem. The more complex the encryption algorithm leads to a robust cryptosystem against attacks. In the last two decades, chaos-based cryptography [2] has drawn the attention of researchers as the pseudorandom sequences produced by such nonlinear chaotic systems are hard to understand and anticipate because of their structural complexity. Stochasticity, sensitivity to their beginning circumstances, and control parameters contribute to their popularity in such applications. As far back as 1998, Fridrich came up with the concept of chaotic encryption [3]. Permutation and diffusion are the two main components working behind such designs; hence, many cryptography researchers prefer chaotic systems [4]. In the last few years, using hyperchaotic systems in cryptography has become more popular than a standard chaotic system. Hyperchaotic systems are those chaotic systems that have more than one positive

[1,3,6] The authors are with Department of Physics, National Institute of Technology, Durgapur 713209, India, E-mail: sp.20ph1501@phd.nitdgp.ac.in, ap.18ph1102@phd.nitdgp.ac.in and mrinalkanti.mandal@phy.nitdgp.ac.in

[1,2] The authors are with Department of Physics, Dr. B. C. Roy Engineering College, Durgapur 713206, India, E-mail: subhashish.pal@bcrec.ac.in and ansuman.mahanty@bcrec.ac.in

[4] The author is with MUSE Lab, Indian Institute of Technology, Gandhinagar-382355, India, E-mail: jk.16ph1102@phd.nitdgp.ac.in

[5] The author is with Department of Physics, Durgapur Government College, Durgapur 713214, India, E-mail: hm.13ph1505@phd.nitdgp.ac.in

Lyapunov exponent.

Many authors have reported different types of encryption algorithms that are comprised of a hyperchaotic system along with logistic maps [4-6], S-box [7-8], discrete wavelet transforms (DWT) [9-10], genetic codes [11-15], sparse matrices [16], etc. Nazir et al. [15] demonstrated RGB color image encryption using S-box substitution in the DNA domain. They have used a 4D hyperchaotic system to design the S-box for a better encryption algorithm. Chai et al. [17] reported image encryption based on image matrix semi-tensor product (STP) and improved genetic algorithm using 6D hyperchaos. Refs. [18-19] Presented color image encryption technique based on DNA encoding and spatiotemporal chaos. The image pixel scrambling in DNA domain has been done before pixel confusion by a mixed linear-nonlinear coupled map lattices (MLNCML) system [18]. On the other hand, in [19], coupled map lattices and hash function SHA-256 are used to design the encryption algorithm.

We begin our proposed symmetric key-based image encryption algorithm by reading the pixel intensities for different color channels of a 256×256 color image. Then we divide each color channel into four identically sized blocks with a dimension of 128×128. All the pixel information in each block is then converted into 8-bit binary numbers, and a bit array of sizes (8×128×128) has been constructed by placing each row side by side. The bit interspersing operation is now done by dividing the whole array into two equal halves and placing the successive elements of the left half in between the two successive elements of the right half from the left, as discussed in Section 3. This new bit array is then reshaped into a 128×128 matrix by accumulating 8-bit in a single element, representing the distorted pixel information. The same operation is also performed for the other three blocks. From the dynamical equations of a 4D-hyperchaotic system [20] discussed in Section 2, 256×256×3 numbers of chaotic sequences have been generated for each state variable by considering the four initial values of the state variables generated from a 32-character security key as discussed in Section 2. An XOR operation is then performed with the chaotic sequences of the x-state variable to get the cipher image for the next stage. The same operation is then performed by considering the blocks of 64×64, 32×32, and 16×16 dimensions successively and doing a bit XOR operation with the chaotic sequences for the y, z, and w state variables, respectively, on the newly generated cipher image after each stage of the operation to get the final cipher image. The details of this encryption algorithm are discussed in Section 4.

To test the quality and strength of our proposed image encryption algorithm, we have performed the standard tests like texture analysis, histogram analysis, correlation analysis, mean square error (MSE), peak signal-to-noise ratio (PSNR), information entropy, key space, occlusion attack, and differential attack. The results of these tests are given in Section 5. In this section, we have also compared our test results with recently published works.

## 2. 4D HYPERCHAOTIC SYSTEM

The 4D hyperchaotic system [21] that we have used to generate the chaotic sequences for the diffusion of the pixel information is given by equation (1). This system shows chaotic behavior for the values of its four control parameters: $a = 2$, $b = 2$, $c = 0.5$, and $d = 14.5$. The 3D phase portraits of this system are shown in Fig. 1. The Lyapunov exponents (LE) of this system are $L_1 = 0.5841$, $L_2 = 0.000127664$, $L_3 = -0.4873$, and $L_4 = -3.5968$. Here we see that two of the LEs are positive out of four. Thus, this system behaves as a hyperchaotic one.

$$\begin{cases} \dot{x} = -ax - byz \\ \dot{y} = -x + cy + cw \\ \dot{z} = d - y^2 - z \\ \dot{w} = x - w \end{cases} \tag{1}$$

Stepwise details for the generation of the chaotic sequences from this system are as follows:

**Step 1:** The ASCII values of 32-character secret key is stored in an array $K_{1,32}$ of 32 elements.

**Step 2:** An array C of zeros of size 13×4 is initiated. Using the equation (2) a matrix A of size 12×3 is generated by taking $n$=7.

$$A^{i+1,j} = A^{i,j} + (-1)^i K[nj + i] \tag{2}$$

where, $i$ varies from 0 to 11 and $j$ from 0 to 2. By leaving the positions $i = 0$ and $j = 4$ of $C$, elements of $A$ are placed in $C$.

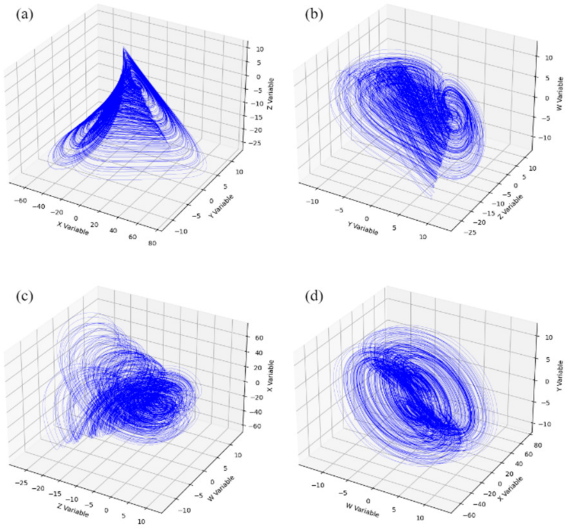


*Fig.1:* *Attractors of 4D Hyperchaotic System. (a) in xyz Plane, (b) in yzw Plane (c) in zwx Plane (d) in wxy Plane.*

**Step 3:** Average pixel value of the plane colour image divided by 2.56 is inserted at position $i = 13$ and $j = 4$ of $C$.
**Step 4:** From $C$, the last row $C_{13,j}$ is extracted.
**Step 5:** Initial conditions $x_0 = \frac{C_{13,1}}{100}$, $y_0 = \frac{C_{13,2}}{100}$, $z_0 = \frac{C_{13,3}}{100}$ and $w_0 = \frac{C_{13,4}}{100}$ are generated.
**Step 6:** Chaotic sequences are generated from equation (1) with the initial conditions. The corresponding iterative solutions are reshaped in 3-dimensional matrices. The values of the generated chaotic sequences are converted within the range 0 to 255 by taking modulo division to get $x$, $y$, $z$, and $w$, each of size 256×256×3.

## 3. BIT INTERSPERSING SCHEME

Scrambling the pixel information of an image in encryption is one of the most important parts of developing a robust cryptosystem [22]. In our proposed scheme, the diffusion of pixel information of the test images has been performed at the bit level. Therefore, the proposed algorithm provides better security than the technique where the pixels are interspersed. For an image matrix, the pixel values are in the range of 0 to 255, and for bit-level interspersing, we have converted the image information into its binary form. In our case, an array of binary elements is first constructed by placing each row side by side. The bit array is then divided into two equal halves, and the elements of the second half of the array are interspersed between the successive bits of the first half, starting from the left. Fig. 2 depicts the proposed mechanism for an image block of size 2×2. In our encryption scheme, we have taken the blocks of sizes 128×128, 64×64, 32×32, and 16×16 from the image matrix for each color channel at four different stages.

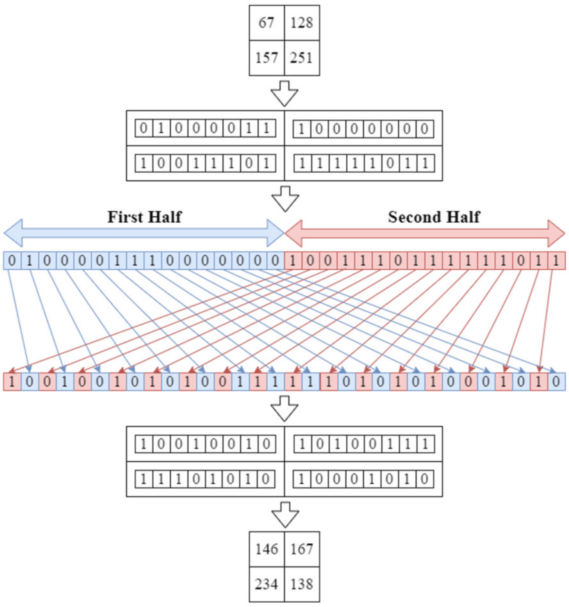


***Fig.2:*** *Bit Interspersing Scheme for 2×2 Block Size.*

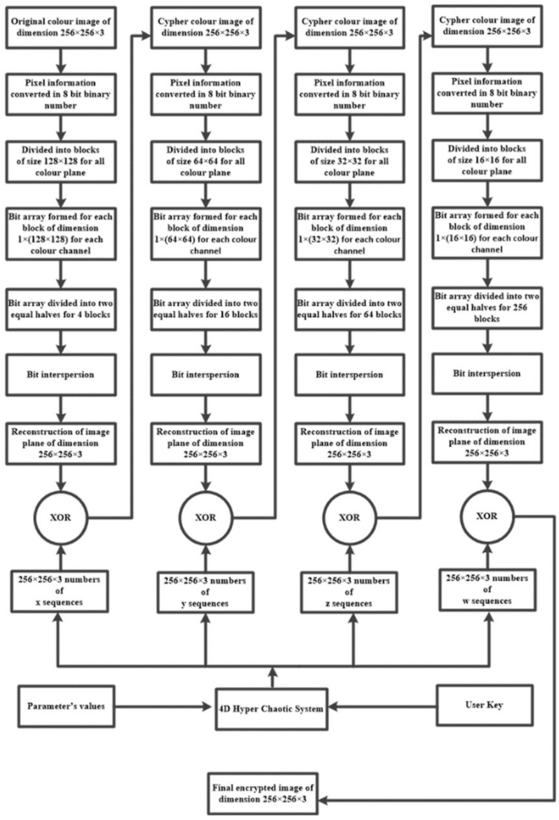

Original colour image of dimension 256×256×3
Cypher colour image of dimension 256×256×3
Cypher colour image of dimension 256×256×3
Cypher colour image of dimension 256×256×3
Pixel information converted in 8 bit binary number
Divided into blocks of size 128×128 for all colour plane
Divided into blocks of size 64×64 for all colour plane
Divided into blocks of size 32×32 for all colour plane
Divided into blocks of size 16×16 for all colour plane
Bit array formed for each block of dimension 1×(128×128) for each colour channel
Bit array formed for each block of dimension 1×(64×64) for each colour channel
Bit array formed for each block of dimension 1×(32×32) for each colour channel
Bit array formed for each block of dimension 1×(16×16) for each colour channel
Bit array divided into two equal halves for 4 blocks
Bit array divided into two equal halves for 16 blocks
Bit array divided into two equal halves for 64 blocks
Bit array divided into two equal halves for 256 blocks
Bit interspersion
Reconstruction of image plane of dimension 256×256×3
XOR
256×256×3 numbers of x sequences
256×256×3 numbers of y sequences
256×256×3 numbers of z sequences
256×256×3 numbers of w sequences
Parameter's values
4D Hyper Chaotic System
User Key
Final encrypted image of dimension 256×256×3

***Fig.3:*** *Flow Chart for Image Encryption Scheme.*

## 4. ENCRYPTION AND DECRYPTION ALGORITHM

The schematic diagram of the encryption algorithm is shown in Fig. 3. The details of this algorithm for the image of size 256×256×3 is discussed stepwise below. However, a user can also apply this scheme for different sizes of images and by considering different block sizes by partial modification.
**Step 1:** A plane color image $I^P$ of dimension 256×256×3, is taken.
**Step 2:** Three color planes $I_R^P$, $I_G^P$ and $I_B^P$ are extracted from $I^P$ for red (R), green (G), and blue (B) color channels, respectively.
**Step 3:** Each $I_i^P$, (i = R, G, B) is then divided into four equal blocks of size 128×128, and all the elements are then converted into 8-bit binary numbers.
**Step 4:** Bit arrays have been formed by placing the consecutive rows side-by-side for each block for individual color channels.
**Step 5:** A new bit sequence has been formed by bit interspersing operation in which the bit array is first divided into two equal halves and placed each successive element of the left half in between the two consecutive elements of the right half from the left.
**Step 6:** From this new bit array, a 128×128 block of byte (8-bit) has been formed for each block of all color channels.
**Step 7:** Combine all four such blocks again to form

the 2D image $N_i^P$ ($i$ = R, G, B) of size 256×256 for the individual colour channel.

**Step 8:** Now, bitwise XOR operation has been carried out with $N_R^P$, $N_G^P$ and $N_B^P$ and the $x$ chaotic sequence of size 256×256×3 generated from equation (1) according to the process discussed in section 2.

**Step 9:** Repeat step 1 - step 8 by considering the newly generated $I_i^P$ ($i =$ R, G, B) for block size 64×64 followedby bit XOR operations with the sequence $y$. A similar operation has been executed by considering 32×32 blocks followed by bit XOR with $z$ and 16×16 blocks followed by bit XOR with w sequentially, as described in Fig. 3.

**Step 10:** Finally, $N_R^P$, $N_G^P$ and $N_B^P$ are combined to form the final cipher image after stage IV as mentioned in Step 9.

To return the original image from the final encrypted image, we have followed the reverse algorithm as shown in Fig. 4. The proposed encryption and decryption algorithm has been tested in Python code. The processing time for the whole process is very fast and finished within a few seconds.

## 5. RESULTS AND TEST ANALYSIS

Six sample color images were taken to test the efficacy of the proposed encryption algorithm, which are shown in Fig. 5, along with the corresponding encrypted images.

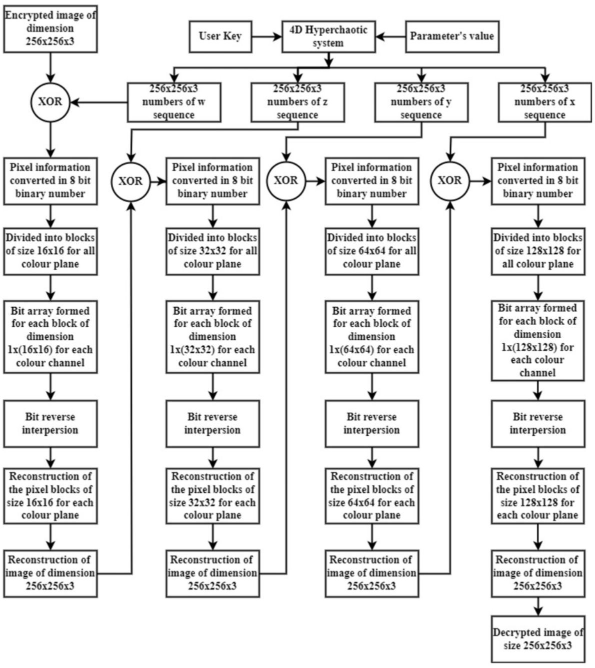

Encrypted image of dimension 256x256x3
User Key
4D Hyperchaotic system
Parameter's value
256x256x3 numbers of w sequence
256x256x3 numbers of z sequence
256x256x3 numbers of y sequence
256x256x3 numbers of x sequence
XOR
Pixel information converted in 8 bit binary number
Divided into blocks of size 16x16 for all colour plane
Bit array formed for each block of dimension 1x(16x16) for each colour channel
Bit reverse interpersion
Reconstruction of the pixel blocks of size 16x16 for each colour plane
Reconstruction of image of dimension 256x256x3
XOR
Pixel information converted in 8 bit binary number
Divided into blocks of size 32x32 for all colour plane
Bit array formed for each block of dimension 1x(32x32) for each colour channel
Bit reverse interpersion
Reconstruction of the pixel blocks of size 32x32 for each colour plane
Reconstruction of image of dimension 256x256x3
XOR
Pixel information converted in 8 bit binary number
Divided into blocks of size 64x64 for all colour plane
Bit array formed for each block of dimension 1x(64x64) for each colour channel
Bit reverse interpersion
Reconstruction of the pixel blocks of size 64x64 for each colour plane
Reconstruction of image of dimension 256x256x3
XOR
Pixel information converted in 8 bit binary number
Divided into blocks of size 128x128 for all colour plane
Bit array formed for each block of dimension 1x(128x128) for each colour channel
Bit reverse interpersion
Reconstruction of the pixel blocks of size 128x128 for each colour plane
Reconstruction of image of dimension 256x256x3
Decrypted image of size 256x256x3

***Fig.4:*** *Flow Chart for Decryption Scheme.*

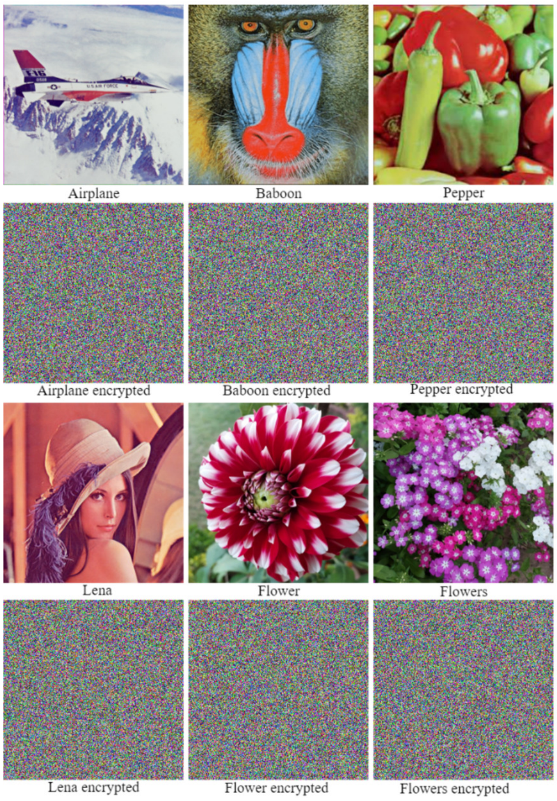


***Fig.5:*** *Original and their Corresponding Encrypted Images.*

### 5.1 Texture Analysis

The grey-level co-occurrence matrix (GLCM) is a statistical technique for analyzing texture that considers the spatial relationship between different pixels. The GLCM determines how frequently pairs of pixels within a particular spatial relationship occur in an image. Building a GLCM and then extracting statistical measures from this matrix to define an image texture. The normalized GLCM can be used to derive measurements such as homogeneity (Ho), contrast (Co), and energy (E).

The following equations are used for the above-mentioned texture analysis of the images:

$$Ho = \sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1+(i-l)^2} \tag{3}$$

$$Co = \sum_{i,j=0}^{N-1} P_{i,j}(i-j)^2 \tag{4}$$

$$E = \left\{ \sum_{i,j=0}^{N-1} (P_{i,j})^2 \right\}^{1/2} \tag{5}$$

Where $(i, j)$, $P_{i,j}$, and $N$ represent the spatial coordinates, GLCM of the image, and the Gray level, respectively. Results for the plane images and their corresponding ciphered images are shown in Table 1. In Table 2, the results are compared with respect to the recently reported work.

***Table 1:*** *Texture Analysis Results.*

| Image | | Ho | Co | E |
|---|---|---|---|---|
| Airplane | Original | 0.8147 | 1.2307 | 0.3873 |
| | Encrypted | 0.1683 | 42.6031 | 0.0621 |
| Baboon | Original | 0.6165 | 2.3950 | 0.1535 |
| | Encrypted | 0.1674 | 42.5684 | 0.0621 |
| Pepper | Original | 0.8482 | 0.5251 | 0.2531 |
| | Encrypted | 0.1674 | 42.5524 | 0.0622 |
| Lena | Original | 0.8380 | 0.6620 | 0.2563 |
| | Encrypted | 0.1675 | 42.5329 | 0.0622 |
| Flower | Original | 0.7866 | 1.4101 | 0.2480 |
| | Encrypted | 0.1682 | 42.6325 | 0.0622 |
| Flowers | Original | 0.6785 | 2.2377 | 0.1670 |
| | Encrypted | 0.1678 | 42.5722 | 0.0622 |

***Table 2:*** *Texture Analysis Comparison.*

| References | Texture Parameters | For the Image | | |
|---|---|---|---|---|
| | | Baboon | Pepper | Lena |
| Proposed method | Ho | 0.167356 | 0.167418 | 0.167508 |
| | Co | 42.56836 | 42.55243 | 42.53286 |
| | E | 0.062144 | 0.062186 | 0.062179 |
| Ref. [15] | Ho | 0.389631 | 0.38986 | 0.389141 |
| | Co | 10.49067 | 10.48289 | 10.51534 |
| | E | 0.015628 | 0.015629 | 0.015629 |

## 5.2 Histogram Analysis

An image histogram displays the frequency with which a specific pixel appears in the image. On an excellent cryptosystem, the encrypted image histogram should be consistent regardless of the nature of the original image to avoid the frequency of a specific pixel value being predictable. In Fig. 6, the histogram plot for each color component of the plane image and encrypted image of Lena is shown. The encrypted image histogram is relatively homogeneous with respect to the plane image histogram. Similar types of observations were made for other images, too. These outcomes supported the validity of the encryption algorithm.

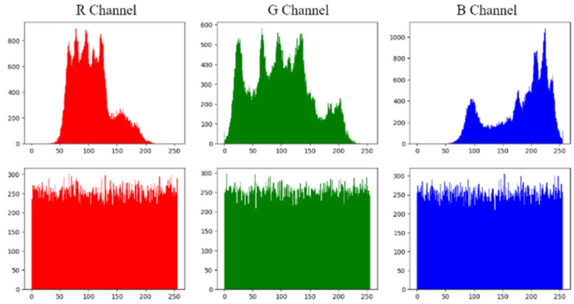


***Fig.6:*** *Histogram of Original (at the top row) and encrypted Lena (at the bottom row) image.*

The histogram of the encrypted image serves as evidence of the effectiveness of the encryption method for statistical study, but it is insufficient to confirm the accuracy of the pixel values in a decrypted image [16]. We have used the chi-square test as a metric to calculate the monotony of the histogram. The chi-square is described as follows:

$$X^2 = \sum\nolimits_{i=0}^{0(\max)} \left\{ \frac{(o_i - e_i)^2}{e_i} \right\} \tag{6}$$

$$\text{here, } e_i = \frac{m \times n}{o(\max)} \tag{7}$$

Here, $o(\max)$ is the maximum pixel value. In this case, it is 255, and $o_i$ is the observed pixel value at index $i$ in the histogram. The $e_i$ is the expected pixel value which is the same at every index $i$.

If the chi-square test score for the histogram of the encrypted image is lower than the theoretical value (293), then the score is acceptable [20]. In Table 3, the estimated chi-square values for the encrypted images are lower than the threshold of 293; hence, the proposed method passes the test.

***Table 3:*** *Texture Analysis Comparison.*

| Image | Color Channels | | | Remarks |
|---|---|---|---|---|
| | R | G | B | |
| Airplane | 250.85156 | 247.11719 | 245.22656 | Pass |
| Baboon | 260.70313 | 271.41406 | 255.4375 | Pass |
| Pepper | 265.67969 | 260.28906 | 230.4375 | Pass |
| Lena | 277.94531 | 268.46094 | 264.01563 | Pass |
| Flower | 247.09375 | 252.82031 | 270.50781 | Pass |
| Flowers | 224.49219 | 243.96875 | 226.57031 | Pass |

## 5.3 Correlation Analysis

Each pixel in a plane image strongly correlates with the pixels immediately next to it, whether those pixels are aligned horizontally, vertically, or diagonally. An effective encryption method should be able to remove these pixel correlations from the plain images and produce encrypted images that resemble noise and have suitably low correlations [22-23]. The following methods are used to figure out the correlation coefficients:

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D_x}\sqrt{D_y}} \tag{8}$$

$$E_x = \frac{1}{N}\sum\nolimits_{i=1}^{N} x_i \tag{9}$$

$$D_x = \frac{1}{N}\sum\nolimits_{i=1}^{N} (x_i - E(x))^2 \tag{10}$$

$$cov(x,y) = \frac{1}{N}\sum\nolimits_{i=1}^{N} (x_i - E(x))(y_i - E(y)) \tag{11}$$

$N$ is the total number of pixel pairs taken from the test image, where x, and y represent the values of the adjacent pixels. Different test images for each color component are shown in Table (4 - 6), with correlation coefficients for each color component in various orientations. Table 7 compares the correlation coefficients to those from a recent study that was also published.

Pixel correlation measures the dependence between two neighboring pixels in the context of an image. Three colors make up a color image, so correlation analysis is performed for each color component along horizontal, vertical, and diagonal directions. The correlation scatters plot for all three channels, considering the color image and its encrypted counterpart for different directions, is shown in Figs. (7 – 9). This strewn diagram shows that nearby pixels in a plain image have a strong association, whereas the correlation coefficient in an encrypted image rapidly declines.

***Table 4:*** *Correlation Coefficients in H-Direction.*

| Image | | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| Airplane | Original | 0.92951 | 0.916487 | 0.921575 |
| | Encrypted | 0.005174 | 0.008076 | -0.007934 |
| Baboon | Original | 0.931113 | 0.889474 | 0.927766 |
| | Encrypted | -0.004016 | -0.002938 | -0.001601 |
| Pepper | Original | 0.964497 | 0.980551 | 0.968931 |
| | Encrypted | 0.002227 | 0.004463 | -0.002864 |
| Lena | Original | 0.920665 | 0.945693 | 0.949692 |
| | Encrypted | -0.003602 | -0.002123 | -0.00322 |
| Flower | Original | 0.955372 | 0.958693 | 0.957467 |
| | Encrypted | -0.007829 | -0.004023 | 0.004488 |
| Flowers | Original | 0.951515 | 0.923458 | 0.948231 |
| | Encrypted | -0.001248 | -0.00575 | 0.00439 |

***Table 5:*** *Correlation Coefficients in V-Direction.*

| Image | | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| Airplane | Original | 0.910839 | 0.933904 | 0.934337 |
| | Encrypted | 0.000579 | -0.009795 | 0.001933 |
| Baboon | Original | 0.916308 | 0.852292 | 0.911202 |
| | Encrypted | -0.004139 | 0.001185 | 0.003468 |
| Pepper | Original | 0.968691 | 0.985193 | 0.97195 |
| | Encrypted | -0.001971 | 0.001657 | -0.001089 |
| Lena | Original | 0.952317 | 0.970962 | 0.970941 |
| | Encrypted | 0.002683 | 0.005589 | 0.001406 |
| Flower | Original | 0.96308 | 0.967722 | 0.96703 |
| | Encrypted | -0.005762 | 0.010598 | 0.014352 |
| Flowers | Original | 0.951825 | 0.920205 | 0.948684 |
| | Encrypted | -0.002226 | 0.001674 | -0.00403 |

***Table 6:*** *Correlation Coefficients in D-Direction.*

| Image | | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| Airplane | Original | 0.8688 | 0.86851 | 0.860913 |
| | Encrypted | -0.002 | 0.003154 | 0.001211 |
| Baboon | Original | 0.897361 | 0.833094 | 0.898375 |
| | Encrypted | -0.003254 | 0.003421 | 0.001122 |
| Pepper | Original | 0.930131 | 0.963305 | 0.937601 |
| | Encrypted | 0.003852 | 0.001162 | 0.001804 |
| Lena | Original | 0.890906 | 0.920797 | 0.926433 |
| | Encrypted | 0.001672 | -0.00189 | -0.000787 |
| Flower | Original | 0.933458 | 0.941716 | 0.938434 |
| | Encrypted | 0.002546 | 0.007882 | -0.004967 |
| Flowers | Original | 0.924476 | 0.875378 | 0.917346 |
| | Encrypted | -0.004761 | 0.006062 | 0.00138 |

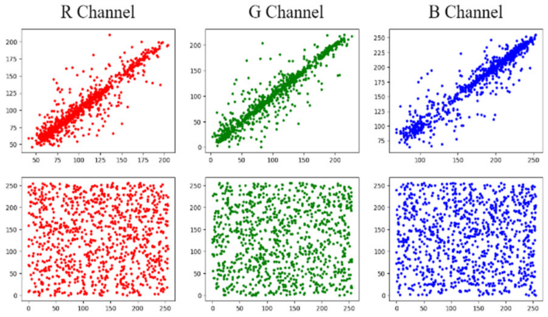


***Fig.7:*** *Correlation Scatter Plot in H-direction for Lena.*

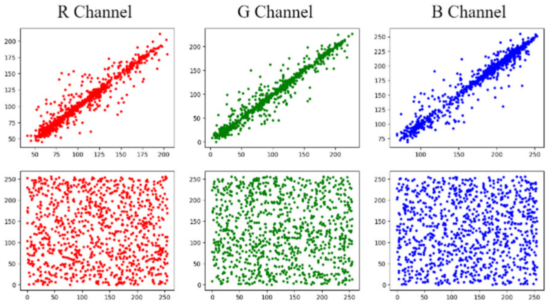


***Fig.8:*** *Correlation Scatter Plot in V-direction for Lena.*

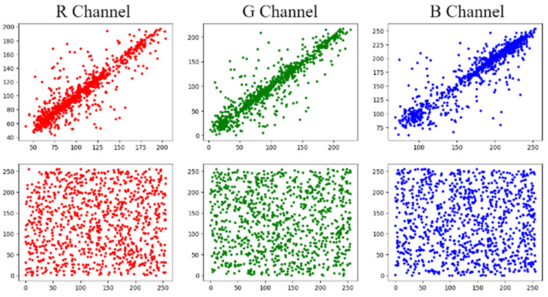


***Fig.9:*** *Correlation Scatter Plot in D-direction for Lena.*

***Table 7:*** *Comparison of Correlation Coefficients of Encrypted Lena.*

| Method | Direction | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| Proposed | Horizontal | -0.00360 | -0.00212 | -0.0032 |
| | Vertical | 0.00268 | 0.00559 | 0.00141 |
| | Diagonal | 0.00167 | -0.00189 | -0.00079 |
| Ref. [17] | Horizontal | 0.0094 | 0.0018 | 0.0019 |
| | Vertical | -0.0011 | -0.0076 | -0.0042 |
| | Diagonal | 0.0009 | 0.0006 | 0.0022 |
| Ref. [18] | Horizontal | 0.0092 | 0.0002 | 0.0076 |
| | Vertical | 0.0203 | -0.0025 | 0.0006 |
| | Diagonal | -0.0073 | -0.0131 | 0.0111 |
| Ref. [19] | Horizontal | 0.0137 | -0.0246 | -0.0137 |
| | Vertical | -0.0237 | -0.017 | 0.0023 |
| | Diagonal | 0.0109 | -0.0133 | -0.0013 |
| Ref. [26] | Horizontal | 0.0090 | -0.0013 | -0.0025 |
| | Vertical | -0.0027 | -0.0051 | -0.0103 |
| | Diagonal | -0.0155 | -0.0078 | 0.0099 |

***Table 8:*** *Comparison of Correlation Coefficients of Encrypted Lena.*

| **Test** | **Image** | **Channels** | | |
|---|---|---|---|---|
| | | **R** | **G** | **B** |
| **MSE** | Airplane | 9039.0797 | 9021.1972 | 9037.2584 |
| | Baboon | 9036.8136 | 9046.9348 | 9032.6218 |
| | Pepper | 9008.3035 | 9068.2824 | 8971.6749 |
| | Lena | 9020.7367 | 9036.3269 | 9012.0009 |
| | Flower | 9029.6316 | 9042.036 | 9023.0653 |
| | Flowers | 9036.8292 | 9050.396 | 9023.8678 |
| **PSNR** | Airplane | 8.569561 | 8.578162 | 8.570437 |
| | Baboon | 8.57065 | 8.565789 | 8.572665 |
| | Pepper | 8.584374 | 8.555553 | 8.602068 |
| | Lena | 8.578384 | 8.570884 | 8.582591 |
| | Flower | 8.574103 | 8.568141 | 8.577263 |
| | Flowers | 8.570643 | 8.564128 | 8.576876 |

***Table 9:*** *Comparison of Correlation Coefficients of Encrypted Lena.*

| Image | Proposed method | | Ref. [14] | Ref. [8] | | Ref. [13] |
|---|---|---|---|---|---|---|
| | MSE (Av) | PSNR (Av) | PSNR | MSE | PSNR | PSNR |
| Airplane | 9033 | 8.5727 | - | - | - | 7.9808 |
| Baboon | 9039 | 8.5697 | 8.7237 | 6691 | 8.1478 | 8.7979 |
| Pepper | 9016 | 8.5807 | 8.0422 | 8290 | 7.9731 | 8.1281 |
| Lena | 9023 | 8.5773 | 8.5645 | 7677 | 8.6430 | 8.6588 |

## 5.4 MSE and PSNR Analysis

A successfully encrypted image should be noticeably different from the plane image. We compute the Mean Square Error (MSE) between the original image and the encrypted image in order to get an idea of how well each image is encrypted. The mathematical definition of MSE [24] follows:

$$MSE = \frac{1}{M \times N} \sum\nolimits_{i,j} (O(i,j) - E(i,j))^2. \qquad (12)$$

Here, $O(i,j)$ and $E(i,j)$ represent the pixel values at the $i^{\text{th}}$ row and $j^{\text{th}}$ column of the original and encrypted image, respectively. The MSE number should ideally be as high as possible for optimal encryption security [14]. In addition, the quality of the encrypted image is assessed by employing a metric known as Peak Signal-to-Noise Ratio (PSNR) [22], which is given below:

$$PSNR = 10 \left( \frac{I_{\max}^2}{MSE} \right). \qquad (13)$$

Where $I_{\max}$ refers to the highest possible value for the image. It is recommended that the PSNR be set to a low value [14], resulting in a significant contrast between the encrypted image and the original image. The performance of the suggested approach is measured in terms of MSE and PSNR for each sample image, and the results are shown in Table 8. A comparison of MSE and PSNR results for our proposed method with some other techniques is shown in Table 9.

## 5.5 Information Entropy Analysis

An information entropy indicates the randomness of a system. In this context, Shannon [25] was the one who came up with the term entropy. The following equation may be used to determine the information entropy of a given information source denoted by $m$.

$$En(m) = \sum\nolimits_{i=0}^{2^{N-1}} P(m_i) \log_2 \frac{1}{P(m_i)}. \qquad (14)$$

Where, $P(m_i)$ denotes the probability of the symbol $m_i$. If $E(m) = N$, the output of a source that emits $2^N$ symbols will be completely arbitrary. The optimum value for $E(m)$ is 8 because each symbol in our system is represented by 8 bits. This implies that the source is random. The results of performing an entropy analysis on the three-color components of the test images are shown in Table 10. It is clear from this Table that the entropy is near the ideal value of 8. Table 11 compares the information entropy acquired using the suggested technique with other reported methods.

***Table 10:*** *Information Entropy of the Images.*

| Image | | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| Airplane | Original | 6.1987 | 6.8167 | 6.7232 |
| | Encrypted | 7.9982 | 7.9983 | 7.9983 |
| Baboon | Original | 6.9771 | 6.8275 | 6.9466 |
| | Encrypted | 7.9981 | 7.9980 | 7.9982 |
| Pepper | Original | 7.1925 | 7.6775 | 7.3669 |
| | Encrypted | 7.9981 | 7.9981 | 7.9985 |
| Lena | Original | 6.9716 | 7.5976 | 7.2688 |
| | Encrypted | 7.9976 | 7.9980 | 7.9981 |
| Flower | Original | 7.7604 | 7.0554 | 7.7236 |
| | Encrypted | 7.9983 | 7.9982 | 7.9980 |
| Flowers | Original | 7.8446 | 7.5341 | 7.7684 |
| | Encrypted | 7.9985 | 7.9975 | 7.9985 |

***Table 11:*** *Comparison of Information Entropy of Encrypted Lena.*

| **Method** | **Channels** | | |
|---|---|---|---|
| | **R** | **G** | **B** |
| Proposed | 7.9976 | 7.9980 | 7.9981 |
| Ref. [17] | 7.9972 | 7.9968 | 7.9976 |
| Ref. [18] | 7.998 | 7.9979 | 7.9978 |
| Ref. [19] | 7.9892 | 7.9898 | 7.9899 |
| Ref [26] | 7.9971 | 7.9974 | 7.9973 |

## 5.6 Key Space

We have used a new 4D Hyperchaotic system and bit-interspersing technique in the proposed algorithm. The system is governed by its four initial conditions. Because of this, if any of the original circumstances are altered even slightly, we will end up with a completely new chaotic sequence. Four initial conditions are generated from the 32 characters of the key, and these conditions are then input into the 4D Hyperchaotic system for generating the chaotic sequences. The larger the value of key space, the less will be the chances of attack. For our proposed system, the key space is $2^{256} \approx 1.158 \times 10^{77}$, and it is sufficiently big to render meaningless the results of any kind of brute force attack. A little shift in the input key (a single bit) will cause a whole new chaotic sequence to be produced. Fig. 10 depicts that the encrypted image cannot be decrypted to its original form with a change in a single character of the key.

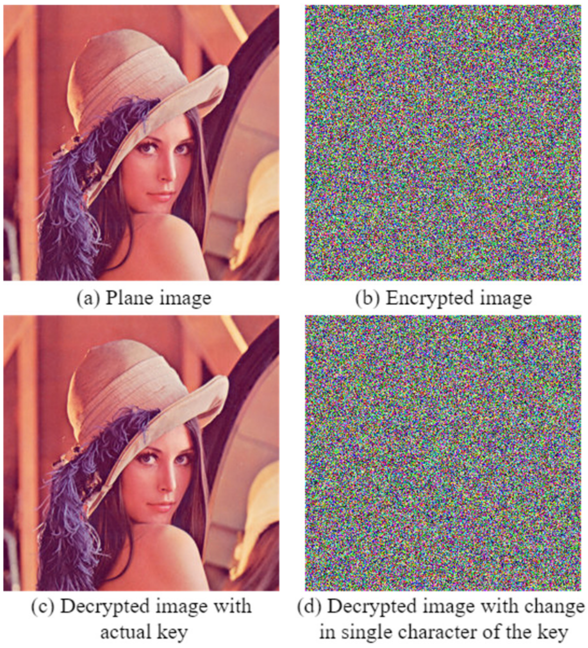

(a) Plane image
(b) Encrypted image
(c) Decrypted image with actual key
(d) Decrypted image with change in single character of the key

***Fig.10:*** *Effect of Change in key Character.*

## 5.7 Occlusion Attack

An encrypted image is subjected to an occlusion attack, which involves partly or entirely concealing the image with certain fixed-valued pixels. A block is eliminated from the encrypted image, and then the corresponding decrypted image is obtained. It can be seen from Fig. 11, that the encrypted photographs are still recognizable and still include most of the information of the original images. It demonstrates that the encryption mechanism that we have provided is resistant to occlusion attacks.

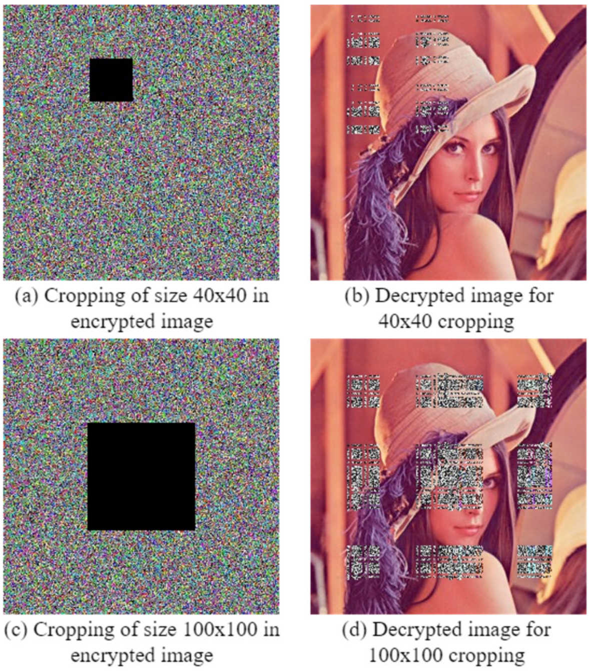

(a) Cropping of size 40x40 in encrypted image
(b) Decrypted image for 40x40 cropping
(c) Cropping of size 100x100 in encrypted image
(d) Decrypted image for 100x100 cropping

***Fig.11:*** *Occlusion Effect on Encrypted Lena.*

## 5.8 Differential Attack Analysis

By slightly changing the pixels of the plane image and comparing the corresponding encrypted images, the attacker may establish a correlation to breach the cryptosystem. The number of pixel change rates (NPCR) and unified average changing intensity (UACI) [27] are commonly used to check the strength of an encryption algorithm against a differential attack. The following are the formulas for calculating NPCR and UACI:

$$NPCR = \frac{1}{L}\sum_{i,j} D(i,j) \times 100\%, \qquad (15)$$

$$UACI = \frac{1}{L}\sum_{i,j} \frac{|E_1(i,j) - E_2(i,j)|}{255} \times 100\%. \qquad (16)$$

Where $L$ refers to the total number of pixels in the image and $E_1$ and $E_2$ are the two encrypted images corresponding to the original image and the one-pixel changed image. $D(i,j)$ may be derived using the principles listed below.

If $E_1(i,j) \neq E_2(i,j)$, then $D(i,j) = 1$,
if $E_1(i,j) = E_2(i,j)$, then $D(i,j) = 0$.

In this experiment, we modified one random pixel of the original image and carry out the test 10 times with one round of encryption to acquire the average values shown in Table 11 for both NPCR and UACI. The findings demonstrate that the suggested method has mean values of NPCRs and UACIs that are more than 99.6 percent and close to 33.3 percent, respectively. This indicates that the value is sufficiently big to withstand the differential attack. A comparison of these tests is presented in Table 12 and Table 13.

***Table 12:*** *NPCR and UACI Test Results.*

| Test | Image | Channels | | |
|---|---|---|---|---|
| | | R | G | B |
| NPCR | Airplane | 99.6162 | 99.6559 | 99.6081 |
| | Baboon | 99.6498 | 99.6254 | 99.6116 |
| | Pepper | 99.6300 | 99.6071 | 99.5964 |
| | Lena | 99.6254 | 99.6239 | 99.6330 |
| | Flower | 99.6361 | 99.6223 | 99.6208 |
| | Flowers | 99.6071 | 99.6055 | 99.6467 |
| UACI | Airplane | 33.5104 | 33.5243 | 33.3232 |
| | Baboon | 33.5596 | 33.5509 | 33.5475 |
| | Pepper | 33.5852 | 33.5248 | 33.4598 |
| | Lena | 33.5304 | 33.5039 | 33.5343 |
| | Flower | 33.2649 | 33.5755 | 33.3473 |
| | Flowers | 33.5223 | 33.4602 | 33.4828 |

***Table 13:*** *Comparison of NPCR for Lena.*

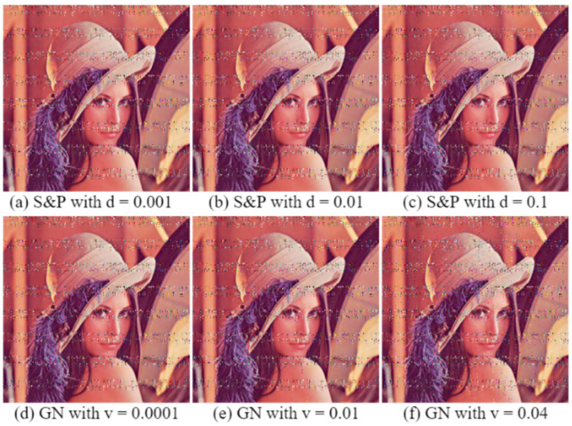
| Method | NPCR | | |
|---|---|---|---|
| | R | G | B |
| Proposed | 99.6254 | 99.6239 | 99.6330 |
| Ref. [17] | 99.6491 | 99.6163 | 99.6324 |
| Ref. [18] | 99.6531 | 99.6522 | 99.6518 |
| Ref. [19] | 99.6137 | 99.6053 | 99.6079 |
| Ref [26] | 99.63 | 99.6 | 99.6 |

***Table 14:*** *Comparison of UACI for Lena.*

| Method | NPCR | | |
|---|---|---|---|
| | R | G | B |
| Proposed | 33.5304 | 33.5039 | 33.5343 |
| Ref. [17] | 33.3827 | 33.3661 | 33.4577 |
| Ref. [18] | 33.4572 | 33.4715 | 33.4384 |
| Ref. [19] | 33.4655 | 33.4781 | 33.4746 |
| Ref [26] | 33.6 | 33.3 | 33.4399 |

## 5.9 Noise attack

In the process of sending the data over the communication channel, the quality of the decrypted image can get distorted as a result of induced noise. On the encrypted image, we applied noise modeled after salt and pepper (S&P) with a range of densities (d), as well as Gaussian noise (GN) with variances (v), to evaluate the efficacy of the suggested approach. The reference to Fig. 12 implies that the original image was retrieved, albeit with some modifications. These results prove that the proposed method is resilient in the presence of interference from the communication channel.



(a) S&P with d = 0.001 (b) S&P with d = 0.01 (c) S&P with d = 0.1

(d) GN with v = 0.0001 (e) GN with v = 0.01 (f) GN with v = 0.04

***Fig.12:*** *Decrypted Lena after introducing noise .*

## 5.10 NIST test

The purpose of the NIST SP800-22 test is to evaluate the performance of the pseudorandom number generator [28]. It can determine whether a chaotic binary sequence is acceptable for image encryption. The NIST SP800-22 test consists of fifteen tests. The P-value can be utilized to quantify the unpredictability of the test sequence. If $P \geq 0.01$, the sequence is random and passes the test [29]. If P is less than 0.01, the sequence is not random and fails the test. If $P = 1$, the sequence is entirely random. If $P = 0$, then the sequence is not at all random. We have used this test suite to check the randomness of the outputs of the hyperchaotic system as well as that of the encrypted image obtained using the proposed encryption technique. The test results shown in Table 15 indicate that the binary stream created by the hyperchaotic system can pass all sub-tests, indicating that this chaotic system is appropriate for image encryption. Test results shown in Table 16 for the encrypted image of Lena also indicate that the binary stream of the encrypted image is random, and hence the proposed encryption scheme successfully passes the test.

***Table 15:*** *NIST Test Results for Hyperchaotic system.*

| **Statistical Test** | **P-value** | **Result** |
|---|---|---|
| Monobit-Frequency | 0.790882 | Pass |
| Block Frequency | 0.36175 | Pass |
| Cumulative Sums | 0.628057 | Pass |
| Runs | 0.71961 | Pass |
| Longest Run | 0.274257 | Pass |
| Rank | 0.852838 | Pass |
| FFT | 0.90718 | Pass |
| Non-overlapping template | 0.304056 | Pass |
| Overlapping Template | 0.253165 | Pass |
| Random Excursions variant | 0.223946 | Pass |
| Serial | 0.918841 | Pass |
| Universal | 0.985246 | Pass |
| Linear Complexity | 0.598174 | Pass |
| Approximate entropy | 0.663217 | Pass |

***Table 16:*** *NIST Test Results for Hyperchaotic system.*

| Statistical Test | P-value | Result |
|---|---|---|
| Monobit-Frequency | 0.198672 | Pass |
| Block Frequency | 0.229519 | Pass |
| Cumulative Sums | 0.388308 | Pass |
| Runs | 0.636702 | Pass |
| Longest Run | 0.414871 | Pass |
| Rank | 0.153418 | Pass |
| FFT | 0.170258 | Pass |
| Non-overlapping template | 0.414734 | Pass |
| Overlapping Template | 0.389543 | Pass |
| Random Excursions variant | 0.813203 | Pass |
| Serial | 0.077309 | Pass |
| Universal | 0.982752 | Pass |
| Linear Complexity | 0.1667 | Pass |
| Approximate entropy | 0.121657 | Pass |

### 5.11 Encryption and Decryption time

Execution time is an essential component of every cryptosystem. The encrypting and decryption process should take as little time as feasible for a cryptosystem to be considered useful. Encryption and decryption time depend on the encryption and decryption algorithm and the system configuration on which it is running. In this regard, we have avoided time-consuming mathematical operations in our cryptosystem. Instead, we have restricted ourselves to using only fundamental XOR operations, modulus operations, and the generation of larger dimension matrices in a single loop. The algorithm has been tested and implemented in Python 3.9. It is being run on a computer with a Processor AMD Ryzen 7 5700U with Radeon Graphics, 1801 Mhz, 8 Core(s), 16 Logical Processor(s), 8Gb RAM, and an operating system that is Windows 11 Home Single Language 64-bit. The encryption and decryption technique is executed 10 times for each image, after which the average amount of time required to encrypt and decrypt the image is computed and displayed in Table 17.

***Table 17:*** *Encryption and decryption time for test images.*

| Image | Encryption time in seconds | Decryption time in seconds |
|---|---|---|
| Airplane | 1.1332 | 1.1349 |
| Baboon | 1.2134 | 1.2150 |
| Pepper | 1.1431 | 1.1445 |
| Lena | 1.1176 | 1.1189 |
| Flower | 1.2038 | 1.2042 |
| Flowers | 1.2043 | 1.2052 |

## 6. CONCLUSIONS

This study proposes a new encryption scheme technique based on four-stage bit interspersing and a 4D hyperchaotic system. Test results demonstrate that this strategy effectively protects the image against statistical attacks. The system is highly key-sensitive, and rapid key change leads to substantial changes in the decrypted image. The histogram of the encrypted image has a uniform distribution. Chi-square tests were employed to support the regularity of different histograms. Analysis of correlation coefficients between adjacent pixels in the plain image indicated a considerable drop in correlation coefficients between adjacent pixels when encryption was applied. MSE and PSNR were used to evaluate the differences between the original and corresponding encrypted images. The results of the information entropy tests show that the entropy values are quite near the theoretical value of 8. As a result, the entropy attack does not compromise the examined encryption scheme. Using NPCR and UACI, we could measure the effectiveness of our defenses against differential attacks. The results of the NIST test sequence on the chaotic key and on the cipher image proved the validity of the proposed algorithm. These important statistical analysis results have been made possible due to the suitable bit interspersing and original image-dependent chaotic key generation from a hyperchaotic system. Overall, the encryption mechanism is simple and on par with the recently reported work. This method can be further modified by changing the block sizes of the images and using hyperchaotic systems with more than four variables.

## References

[1] M.A. Al-Shabi, “A survey on symmetric and asymmetric cryptography algorithms in information security,” *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, no. 3, pp.576-589. 2019.

[2] L. Kocarev, “Chaos-based cryptography: a brief overview,” in *IEEE Circuits and Systems Magazine*, vol. 1, no. 3, pp. 6-21, 2001.

[3] J. Fridrich, “Symmetric ciphers based on two-dimensional chaotic maps,” *International Journal of Bifurcation and chaos*, vol. 8, no. 6, pp. 1259-1284, 1998.

[4] L. Kocarev and S. Lian, “Chaos-based cryptography: Theory, algorithms and applications,” *Springer Science & Business Media*, vol. 354, 2011.

[5] L. Li-Hong, B. Feng-Ming and H. Xue-Hui,, “New image encryption algorithm based on logistic map and hyper-chaos,” in *2013 International Conference on Computational and Information Sciences*, pp. 713-716, 2013.

[6] J. Xu, P. Li, F. Yang and H. Yan,, “High intensity image encryption scheme based on quantum logistic chaotic map and complex hyperchaotic system,” in *IEEE Access*, vol. 7, pp.167904-167918, 2019.

[7] Y. Liu, X. Tong and J. Ma, “Image encryption algorithm based on hyper-chaotic system and

dynamic S-box," *Multimedia Tools and Applications*, vol. 75, no. 13, pp.7739-7759, 2016.

[8] M.D. Gupta and R.K. Chauhan, "Secure image encryption scheme using 4D-hyperchaotic systems based reconfigurable pseudorandom number generator and S-box," *Integration*, vol. 81, pp.137-159, 2021.

[9] X. Wu, D. Wang, J. Kurths and H. Kan, "A novel lossless colour image encryption scheme using 2D DWT and 6D hyperchaotic system," *Information Sciences*, vol. 349, pp. 137-153, 2016.

[10] L. Oteko Tresor and M. Sumbwanyambe, "A selective image encryption scheme based on 2d DWT, Henon map and 4d Qi hyper-chaos," in *IEEE Access*, vol. 7, pp.103463-103472, 2019.

[11] Y. Liu and J. Zhang, "A multidimensional chaotic image encryption algorithm based on DNA coding," *Multimedia Tools and Applications*, vol. 79, no. 29, pp.21579-21601, 2020.

[12] M. Kar, A. Kumar, D. Nandi and M.K. Mandal, "Image encryption using DNA coding and hyperchaotic system," *IETE Technical Review*, vol. 37, no. 1, pp.12-23, 2020.

[13] X. Ouyang, Y. Luo, J. Liu, L. Cao and Y. Liu, "A color image encryption method based on memristive hyperchaotic system and DNA encryption.," *International Journal of Modern Physics B*, vol. 34, no. 4, p.2050014, 2020.

[14] B. Jasra and A.H. Moon, "Color image encryption and authentication using dynamic DNA encoding and hyperchaotic system," *Expert Systems with Applications*, vol. 206.p.117861, 2022.

[15] H. Nazir, I.S. Bajwa,, S. Abdullah, R. Kazmi, and M. Samiullah, "A Colour Image Encryption Scheme Combining Hyperchaos and Genetic Codes," *IEEE Access*, vol. 10, pp.14480-14495, 2022.

[16] J. Karmakar, N. Debashis and M.K. Mandal, "A novel hyper-chaotic image encryption with sparse-representation based compression," *Multimedia Tools and Applications*, vol. 79 no. 37, pp. 28277-28300, 2020.

[17] X. Chai, X. Zhi, Z. Gan, Y. Zhang, Y. Chen, and J. Fu, "Combining improved genetic algorithm and matrix semi-tensor product (STP) in colour image encryption," *Signal Processing*, vol. 183, p.108041, 2021.

[18] K. Xuejing and G. Zihui, "A new colour image encryption scheme based on DNA encoding and spatiotemporal chaotic system," *Signal Processing: Image Communication*, vol. 80, p.115670, 2020.

[19] X.J. Wu, K.S. Wang, X.Y. Wang, H.B. Kan, J. Kurths, "Colour image DNA encryption using NCA map-based CML and one-time keys," *Signal Process*, vol.148, pp.272–287, 2018.

[20] S. Ma, Y. Zhang, Z. Yang, J. Hu and X. Lei, "A new plaintext-related image encryption scheme based on chaotic sequence," in *IEEE Access*, vol. 7, pp.30344-30360, 2019.

[21] R.D. Méndez-Ramírez, A. Arellano-Delgado, M.A. Murillo-Escobar and C. Cruz-Hernández, "A New 4D Hyperchaotic System and Its Analog and Digital Implementation," *Electronics*, vol. 10, no.15, pp. 1793, 2021.

[22] Z. Zhong, J. Chang, M. Shan and B. Hao,, "Double image encryption using double pixel scrambling and random phase encoding," *Optics Communications*, vol.285 ,no. 5, pp.584-588, 2012.

[23] Q. Zhang, L. Guo and X. Wei, "Image encryption using DNA addition combining with chaotic maps," *Mathematical and Computer Modelling*, vol.52, no. 11-12, pp.2028-2035, 2010.

[24] O. E. Rössler, "An equation for continuous chaos original research article," Phys Lett A, vol. 57, no. ), pp.397–398, 1976.

[25] C.E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*. vol. 27, pp. 623-656, 1948.

[26] X.Y. Wang, H.L. Zhang and X.M. Bao,"Colour image encryption scheme using CML and DNA sequence operations," *Biosystems*, vol. 144, pp.18-26. 2016.

[27] Y. Wu, J.P. Noonan and S. Agaian, "NPCR and UACI randomness tests for image encryption," *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, vol. 1, no. 2, pp.31-38, 2011.

[28] C. Fan, Q. Ding and C. K. Tse, "Counteracting the dynamical degradation of digital chaos by applying stochastic jump of chaotic orbits," *International Journal of Bifurcation and Chaos*, vol. 29, no. 8, p.1930023, 2019.

[29] L. Bassham, A. Rukhin, J. Soto, J. Nechvatal, M. Smid, S. Leigh, M. Levenson, M. Vangel, N. Heckert and D. Banks, *SP800-22 Rev. 1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, Tech. Rep., SP800-22, National Institute of Standards & Technology, 2010.

**Subhashish Pal** received the B.Sc. and M.Sc. degrees from the C.C.S University, Meerut, Uttar Pradesh, India in 2000 and 2002 respectively. Presently working as an Assistant Professor of Physics in Dr. B. C. Roy Engineering College (BCREC), Durgapur, West Bengal, India and pursuing Ph.D. from NIT Durgapur, West Bengal, India. He has published two papers in International Journals. His area of interest is image processing, data security and cryptography.

**Ansuman Mahanty** received the MCA and M. Tech (CST) degrees from BIT Mesra, Ranchi and MAKAUT in 2001 and 2011 respectively. Presently working as an Assistant Professor in the Department of Computer Applications (MCA) in Dr. B. C. Roy Engineering College (BCREC), Durgapur, West Bengal, India. His area of interest is image processing, data security and cryptography.

**Arghya Pathak** received his B.Sc. degree in Physics from J. K. College Purulia (University of Burdwan), India in 2012 and M.Sc. degree from National Institute of Technology Durgapur, India in 2014. Presently he is pursuing his Doctoral research work. His research interests include image processing, sparse representation of signals, data security and cryptography. He has published two papers in International Journals.

**Jayashree Karmakar** received her B.Sc. degree in Physics from the University of Burdwan, India in 2014 and M.Sc. degree from National Institute of Technology Durgapur, India in 2016. She has completed her PhD degree from NIT Durgapur in the year 2021. Presently she is pursuing her Post-Doctoral research work at MUSE Lab IIT Gandhinagar. Her research interests include image processing, sparse representation of signals, data compression and security. She has published five papers in International Journals.

**Hrishikesh Mondal** received the B.Sc. and M.Sc. degrees from the University of Burdwan, Burdwan, West Bengal, India in 1998 and 2000 respectively. Presently working as an Assistant Professor of Physics in Durgapur Government College and completed Ph.D. from NIT, Durgapur (in 2022) West Bengal, India. He has published five research papers in national and International Journals.

**Mrinal Kanti Mandal** received the B.Sc. degree in Physics (Hons.) from Burdwan University, India in 1998, and the M.Sc. and Ph.D. degrees from the same University in 2000 and 2008 respectively. Presently he is an Associate Professor in the Department of Physics, National Institute of Technology, Durgapur. He has published more than 60 international journal papers and 50 conference papers in proceeding. His research interests include Design of Electronic Circuits & Systems, Nonlinear Dynamics & Chaos, Cryptography and Image Processing. He is a reviewer of Nonlinear Dynamics, Security and Communication Networks, International Journal of Electronics, Indian Journal of Pure and Applied Physics, Indian Journal of Physics, etc. He is a life member of IEEE, IETE, IPS and IAPT. His bio-data has been included in the database of science and Technology chapter of “Marquis Who’s Who” since 2010 edition.