A new RGB image encryption algorithm based on hyper-chaotic-discrete-wavelet-packet-transform

Manish Kumar

Department of Mathematics, Birla Institute of Technology and Science-Pilani, Hyderabad Campus, Hyderabad-500078, Telangana, India

Abstract

There are a few significant concerns for chaotic-based image encryption algorithms, such as small key space and weak security problems, etc. (see [1]). Motivated by this fact, we propose a new hyper-chaotic-discrete-wavelet-packet-transform encryption algorithm for RGB images to avoid these issues. The proposed algorithm consists of five steps. The first two steps are associated with 2D and 3D contemporary chaotic diffusion processes. The intermediate stage involves a multilevel discrete wavelet packet transform selected by users, providing immune support to the algorithm's infrequent attacks. The last two steps consist of chaotic diffusion and row-column matrix shuffling. The main advantage of using the hyper-chaotic system in the encryption process is giving a vast keyspace to make brute force attacks infeasible and provides an excellent speed to the proposed algorithm. The proposed scheme renders diffusion and confusion in 2D and 3D forms followed by Bit-XOR operation, including the wavelet packet transform as an intermediate step to avoid high redundancy and high correlation in the encrypted image. The proposed algorithm is compared with the existing compatible algorithms. Further, the performance and security analysis demonstrate the robustness and efficiency of the proposed algorithm.

Keywords:
Hyper-chaotic system, Discrete wavelet packet transform, 2D and 3D chaotic diffusion, Encryption, Decryption

1. Introduction

In today's world, we are sharing a lot of information in images from one end to the other. Sometimes this information becomes essential (confidential) and requires secure transmission so that none other than one we want to share can view this sensitive information. A middle person is an adversary who tries to fetch any partial or complete knowledge from these images for their financial gain or any illegal purpose. For many years, militaries and governments have been using encryption/decryption algorithms to facilitate secret communication. One can use encryption algorithms in various online platforms, such as education, marketing, trading, video telecasting, and advertisement. Hence, we need to protect this critical information through the encryption algorithm so that an adversary cannot steal any info as partial or complete. However, achieving this target is quite challenging as adversaries continuously investigate to break the existing algorithms.

Recently a review on chaos-based image encryption is reported in [2], and references therein are good to support their algorithms. Various issues related to chaos-based encryption have been reported in [3–6]. Image encryption algorithm using Fourier domain are given in [7–11]. However, in the Fourier domain, the data becomes complex, increasing processing time and storage problems. In contrast, the wavelet transform solves the shortcomings of the Fourier transform. Also, by compressing the data, wavelet transforms helps in reducing the data size. It would decrease the processing time and storage problems in low-power devices that work through wireless communications. Algorithms proposed in [13–15] provide high security offered in the wavelet domain. An image encryption algorithm, which uses the combination of wavelet and Fourier transforms, is given in [12]. In [16], the authors developed image encryption in the wavelet packet domain where the original image is decomposed using wavelet packets and shuffled using RMAC parameters.

We propose a new encryption model (algorithm) to enhance the secret of the original images using a hyper-chaotic system involving discrete wavelet packet transform. In the proposed algorithm, an original image is inter-shuffled between R, G, and B layers and diffused by applying 3D chaotic diffusion followed by 2D chaotic diffusion. Then finally, chaotic diffusion in the wavelet packet domain has been used. The main objective of using the

*Manish Kumar (Corresponding author)
email:manish.math.bhu@gmail.com; manishkumar@hyderabad.bits-pilani.ac.in.

wavelet packet transform after 2D chaotic diffusion is to provide coefficients effectively over random images. The time complexity of the proposed algorithm is excellent. The proposed keyspace is strong enough to resist brute force attacks infeasible. A comparison of the proposed algorithm with existing compatible methods demonstrates its robustness.

Organization of the paper: In section 2, we briefly state the 4D Lorenz hyper-chaotic system and the discrete wavelet packet transform. In section 3, we provide a procedure to generate secret keys for the encryption algorithm. Section 4 is devoted to the construction of the proposed algorithm. Section 5 involves the performance and security analysis of the proposed algorithm. We have provided a comparison with existing compatible methods in Section 6. Finally, the conclusion is available in the last section.

2. Preliminaries on 4D Lorenz hyper-chaotic system and discrete wavelet packet transform

This section discusses the two main ingredients of the proposed algorithm, such as the 4D Lorentz hyper-chaotic system and Discrete Wavelet Packet Transform (DWPT).

2.1. 4D Lorenz hyper-chaotic system

We use 4D Lorenz hyper-chaotic system to generate random sequences for key generation. The Lorenz 4D hyper-chaotic system is provided by:

$$\begin{aligned}\frac{dx}{dt} &= a(y-x)+u,\\ \frac{dy}{dt} &= bx-xz-y,\\ \frac{dz}{dt} &= xy-cz,\\ \frac{du}{dt} &= du-xz,\end{aligned}$$

where a, b, and c ($a = 10$, $b = 28$, $c = 8/3$) are constants and d ($d = 1.3$) is a control parameter. To demonstrate the key sensitivity of the hyper-chaotic system, we have plotted graphs for each sequence. For instance, the symbol $x_r(0)$ and $x_r(0)^*$ denotes the initial and marginal change in the key sequence used for red channel. Sensitivity to the initial conditions of the hyper-chaotic system up to 15th decimal places is depicted in figure 1.

2.2. Discrete wavelet packet transform

Discrete Wavelet Packet Transform (DWPT) decomposes the original image into various parts depending upon levels (for instance, an image is decomposed into four sub-images using a 2D-DWPT with level one, and each of them is a one of the fourth size of the original image. One of the sub-images is a low-frequency sub-band say Approximation (A), and the others are high-frequency sub-band in the Vertical (V), Horizontal (H), and Diagonal (D) directions.). Similarly, all four sub-images can be further divided again if we increase one more level to sixteen sub-images and so on see for details [21–24]. The DWPT is an extension of Discrete Wavelet Transform (DWT) wherein only low-frequency sub-band (A) decomposed continuously while increasing the wavelet levels. Unlike DWT [25, 26], DWPT involves further decomposition of the wavelet spaces, $W^n_{j,\psi}$ where $n, j \in \mathbb{Z}$. We can define 2D-DWPT of an image $f(n_1,n_2)$ of size $N_1 \times N_2$ (for $n = 0,1$) as follows

$$\begin{aligned}W^0_{\varphi}(j_0,k_1,k_2) &= \frac{1}{\sqrt{N_1N_2}}\sum_{n_1}\sum_{n_2} f(n_1,n_2)\varphi_{j_0,k_1,k_2}(n_1,n_2),\\ W^1_{\psi}(j_0,k_1,k_2) &= \frac{1}{\sqrt{N_1N_2}}\sum_{n_1}\sum_{n_2} f(n_1,n_2)\psi_{V,j_0,k_1,k_2}(n_1,n_2)\\ &+\frac{1}{\sqrt{N_1N_2}}\sum_{n_1}\sum_{n_2} f(n_1,n_2)\psi_{H,j_0,k_1,k_2}(n_1,n_2)\\ &+\frac{1}{\sqrt{N_1N_2}}\sum_{n_1}\sum_{n_2} f(n_1,n_2)\psi_{D,j_0,k_1,k_2}(n_1,n_2),\end{aligned}$$

where j_0 represents any starting scale.

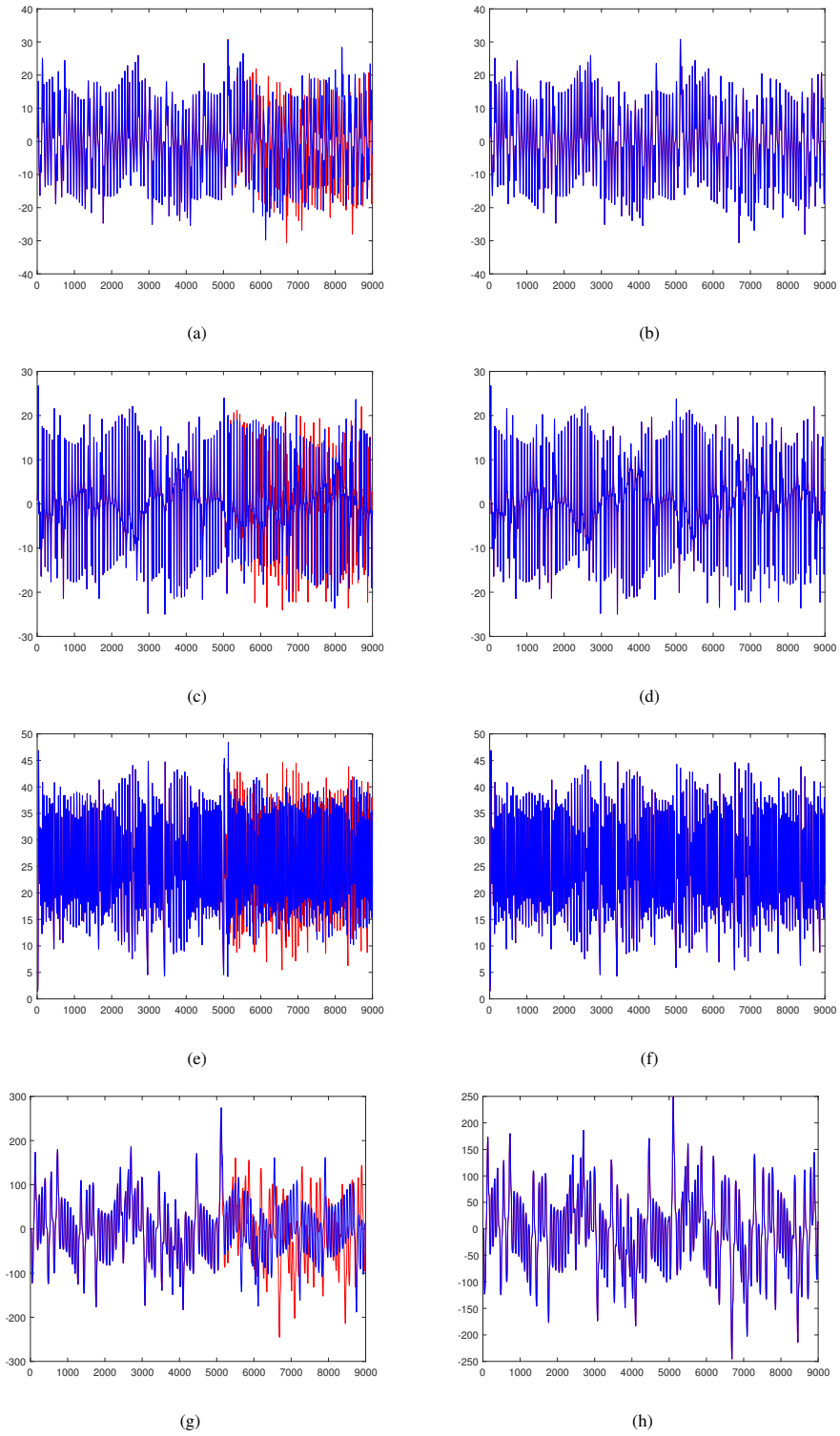

Figure 1: Key sequence analysis: Sensitive Dependence on Initial Condition (SIC) for the marginal change in (a) from $x_r(0) = 1.2$ to $x_r(0)^* = 1.2 + 1 \times 10^{-15}$, (b) from $x_r(0) = 1.2$ to $x_r(0)^* = 1.2 + 1 \times 10^{-16}$, (c) from $y_r(0) = 0.8$ to $y_r(0)^* = 0.8 + 1 \times 10^{-15}$, (d) from $y_r(0) = 0.8$ to $y_r(0)^* = 0.8 + 1 \times 10^{-16}$, (e) from $z_r(0) = 1.6$ to $z_r(0)^* = 1.6 + 1 \times 10^{-15}$, (f) from $z_r(0) = 1.6$ to $z_r(0)^* = 1.6 + 1 \times 10^{-16}$, (g) from $u_r(0) = 0.7$ to $u_r(0)^* = 0.7 + 1 \times 10^{-15}$, (h) from $u_r(0) = 0.7$ to $u_r(0)^* = 0.7 + 1 \times 10^{-16}$.

The original image can be obtained using Inverse Discrete Wavelet Packet Transform (IDWPT) expressed as below

$$
\begin{aligned}
f(n_1,n_2) = & \frac{1}{\sqrt{N_1N_2}}\sum_{k_1}\sum_{k_2} W_{\varphi}^{0}(j_0,k_1,k_2)\varphi_{j_0,k_1,k_2}(n_1,n_2) \\
& + \frac{1}{\sqrt{N_1N_2}}\sum_{j_0}\sum_{k_1}\sum_{k_2} W_{\psi}^{1}(j_0,k_1,k_2)\psi_{V,j_0,k_1,k_2}(n_1,n_2) \\
& + \frac{1}{\sqrt{N_1N_2}}\sum_{j_0}\sum_{k_1}\sum_{k_2} W_{\psi}^{1}(j_0,k_1,k_2)\psi_{H,j_0,k_1,k_2}(n_1,n_2) \\
& + \frac{1}{\sqrt{N_1N_2}}\sum_{j_0}\sum_{k_1}\sum_{k_2} W_{\psi}^{1}(j_0,k_1,k_2)\psi_{D,j_0,k_1,k_2}(n_1,n_2).
\end{aligned}
$$

The proposed algorithm uses 'db4' (Daubechies 4) wavelet function while doing decomposition of image. The original image is divided into sixteen parts after the application of a two-level discrete wavelet packet transform (as $n > 1$), as shown in figure 2.

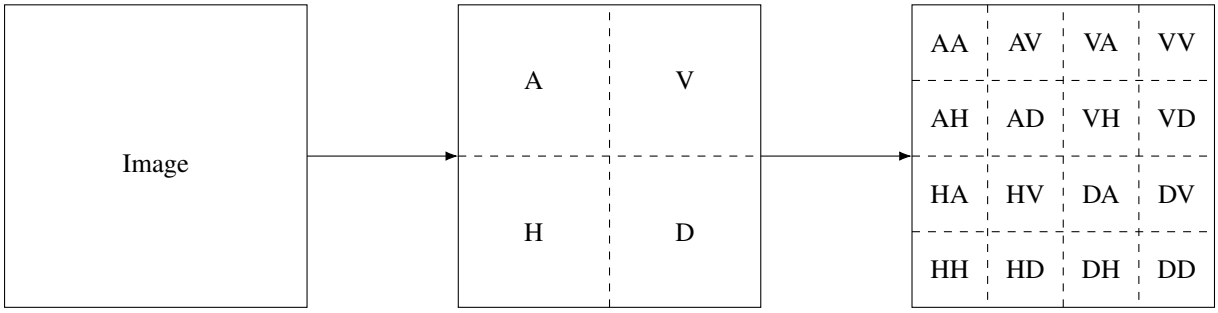

Figure 2: Second level 2D discrete wavelet packet decomposition of an image.

3. Key generation using original RGB image

In this section, twelve keys have been generated (i.e., Key_1,Key_2, Key_3,Key_4,Key_5,Key_6,Key_7,Key_8, Key_9, Key_{10}, Key_{11} and Key_{12}) using 4D Lorenz hyper-chaotic system. The proposed key generation is based on original image which avoid plain-image and cipher-image attacks. Here, each keys are highly sensitive and prone to attack. The procedure for key generation is as follows:

1. Suppose the original RGB image is of size $N_1 \times N_2 \times 3$ and decompose it into R, G, and B layers.
2. Sum over all the pixel values in R layer, take modulo 256 and then divide by 256 to produce a decimal number, which is a value of r_0 and do the same for G and B layers to get the values g_0 and b_0, respectively.
3. Generate three solution space by using 4D Lorenz hyper-chaotic system for $i = 1,2,\ldots,3N_1N_2$, as follows:
   (a) $x_r(i)$, $y_r(i)$, $z_r(i)$ and $u_r(i)$ with initial condition $x_r(0) = 1.2$, $y_r(0) = 0.8$, $z_r(0) = 1.6$ and $u_r(0) = r_0$.
   (b) $x_g(i)$, $y_g(i)$, $z_g(i)$ and $u_g(i)$ with initial condition $x_g(0) = 1.2$, $y_g(0) = 0.8$, $z_g(0) = 1.6$ and $u_g(0) = g_0$.
   (c) $x_b(i)$, $y_b(i)$, $z_b(i)$ and $u_b(i)$ with initial condition $x_b(0) = 1.2$, $y_b(0) = 0.8$, $z_b(0) = 1.6$ and $u_b(0) = b_0$.
4. **Generation of Key_1 matrix:** Take the sequence $\{x_r(1), x_r(2), x_r(3), \ldots, x_r(d^2)\}$, reshape this sequence into a matrix of size $d \times d$ and sort this matrix row wise (parameter d is defined in the step 1 of 4.1). Mark the previous index values before sorted, this matrix will be Key_1.
5. **Generation of Key_2, Key_3 and Key_4 matrices:** Let us generate three new sequences $\{y_r^{'}(1), y_r^{'}(2), y_r^{'}(3), \ldots, y_r^{'}(N_1N_2)\}$, $\{z_r^{'}(1), z_r^{'}(2), z_r^{'}(3), \ldots, z_r^{'}(3N_1N_2)\}$ and $\{u_r^{'}(1), u_r^{'}(2), u_r^{'}(3), \ldots, u_r^{'}(N_1N_2)\}$ where

$$
\begin{aligned}
y_r^{'}(i) &= round(|1000 \times y_r(i)|) \pmod{d-1} + 1, \quad \text{for } i = 1,2,\ldots,N_1N_2, \\
z_r^{'}(i) &= round(|1000 \times z_r(i)|), \quad \text{for } i = 1,2,\ldots,3N_1N_2, \\
u_r^{'}(i) &= round(|1000 \times u_r(i)|), \quad \text{for } i = 1,2,\ldots,N_1N_2.
\end{aligned}
$$

   Reshape these sequences $\{y_r^{'}(1), y_r^{'}(2), y_r^{'}(3), \ldots, y_r^{'}(N_1N_2)\}$, $\{z_r^{'}(1), z_r^{'}(2), z_r^{'}(3), \ldots, z_r^{'}(3N_1N_2)\}$ and $\{u_r^{'}(1), u_r^{'}(2), u_r^{'}(3), \ldots, u_r^{'}(N_1N_2)\}$ into matrices of sizes $N_1 \times N_2$, $r \times c \times d$ (parameters r and c are defined in step 1 of the Section 4.1) and $N_1 \times N_2$ will be Key_2,Key_3 and Key_4, respectively.

6. **Generation of permutation keys $Key_5, Key_6, Key_7, Key_8, Key_9$ and Key_{10}:** To generate permutations keys one can introduce a function $perm(T) = T'$ over the sequence $T = \{t_1, t_2, t_3, \ldots, t_{N_1}\}$ where $perm(T)$ sorts the sequence T and mark the previous index after sorting it will becomes $T^{'}$. Now, we compute $Key_5, Key_6, Key_7, Key_8, Key_9$ and Key_{10} as follows:

$$\begin{aligned} Key_5 &= perm(\{x_g(1), x_g(2), x_g(3), \ldots, x_g(N_1)\}), \\ Key_6 &= perm(\{y_g(1), y_g(2), y_g(3), \ldots, y_g(N_2)\}), \\ Key_7 &= perm(\{z_g(1), z_g(2), z_g(3), \ldots, z_g(N_1)\}), \\ Key_8 &= perm(\{u_g(1), u_g(2), u_g(3), \ldots, u_g(N_2)\}), \\ Key_9 &= perm(\{x_b(1), x_b(2), x_b(3), \ldots, x_b(N_1)\}), \\ Key_{10} &= perm(\{y_b(1), y_b(2), y_b(3), \ldots, y_b(N_2)\}). \end{aligned}$$

7. **Generation of Key_{11}, and Key_{12}:** We generate two new sequences $\{z_b^{'}(1), z_b^{'}(2), z_b^{'}(3), \ldots, z_b^{'}(n_1^{'}n_2^{'})\}$ and $\{u_b^{'}(1), u_b^{'}(2), u_b^{'}(3), \ldots, u_b^{'}(n_1^{'}n_2^{'})\}$, where $n_1^{'} \times n_2^{'}$ is size of the approximation coefficients at J^{th} level after applying discrete wavelet packet transform on the image

$$\begin{aligned} z_b^{'}(i) &= |1000 \times z_b(i)|, \quad i = 1, 2, \ldots, n_1^{'}n_2^{'}, \\ u_b^{'}(i) &= |1000 \times u_b(i)|, \quad i = 1, 2, \ldots, n_1^{'}n_2^{'}. \end{aligned}$$

Reshape the sequences $\{z_b^{'}(1), z_b^{'}(2), z_b^{'}(3), \ldots, z_b^{'}(n_1^{'}n_2^{'})\}$ and $\{u_b^{'}(1), u_b^{'}(2), u_b^{'}(3), \ldots, u_b^{'}(n_1^{'}n_2^{'})\}$ into matrices of size $n_1^{'} \times n_2^{'}$, sort them by column and row wise and mark previous index after sorting it becomes Key_{11} and Key_{12}, respectively.

4. The procedure of the proposed encryption/decryption scheme

The proposed encryption and decryption scheme has been elaborated as follows.

4.1. Encryption scheme

The proposed encryption procedure is provided in detail, and the flowchart of the encryption algorithm is shown in figure 3. There are five steps in the encryption algorithm and are elaborated as follows.

Step 1 **Chaotic 3D diffusion:** We introduced a new method of shuffling in 3D domain. At first, the original RGB image I of size $N_1 \times N_2 \times 3$ and reshape into an image I_1 size of $r \times c \times d$. (For instance, $r = 128$, $c = 64$ and $d = 96$ for an original image of size $512 \times 512 \times 3$). Then apply 3D chaotic diffusion on the image I_1 as below; for $i = 1, 2, \ldots, r$, $j = 1, 2, \ldots, c$ and $k = 1, 2, \ldots, d$ to get an image I_2 as follows

$$I_2(i, j, k) = I_1(i, j, key_1(key_2(i, j), k)) \oplus key_3(i, j, key_1(key_2(i, j), k)) \pmod{256}.$$

The above process provides inter-shuffling in R, G, and B layers of an original image associated with diffusion. Now reshape the obtained image I_2 into the size $N_1 \times N_2 \times 3$.

Step 2 **Chaotic 2D diffusion:** Now, decompose the image I_2 into three layers as I_2^R, I_2^G and I_2^B each of size $N_1 \times N_2$. Apply 2D chaotic diffusion on each I_2^R, I_2^G and I_2^B which shuffles image pixels only in 2D associated with diffusion process as follows:

$$\begin{aligned} I_3^R(i, j) &= I_2^R(key_5(i), key_6(j)) \oplus key_4(key_5(i), key_6(j)) \pmod{256}, \\ I_3^G(i, j) &= I_2^G(key_7(i), key_8(j)) \oplus key_4(key_7(i), key_8(j)) \pmod{256}, \\ I_3^B(i, j) &= I_B^B(key_9(i), key_{10}(j)) \oplus key_4(key_9(i), key_{10}(j)) \pmod{256}, \end{aligned}$$

for $i = 1, 2, \ldots, N_1$ and $j = 1, 2, \ldots, N_2$.

Step 3 **Chaotic diffusion:** Use discrete wavelet packet transform on I_3^R, I_3^G and I_3^B individually and apply chaotic diffusion in first block of the discrete wavelet packet coefficients (say, approximation coefficients AA in level 2 as shown in figure 2) J^{th} level as depicted in figure 3. At J^{th} level for R, G and B layers the

approximation wavelet packet coefficients are A_J^R, A_J^G and A_J^B respectively. The process of the chaotic diffusion process is given below:

$$A_J^{R'} = (int16(A_J^R) \oplus key_{11}) \oplus key_{12}) + A_J^R - round(A_J^R),$$
$$A_J^{G'} = (int16(A_J^G) \oplus key_{11}) \oplus key_{12}) + A_J^G - round(A_J^G),$$
$$A_J^{B'} = (int16(A_J^B) \oplus key_{11}) \oplus key_{12}) + A_J^B - round(A_J^B),$$

for $i = 1, 2, \ldots, n_1^{'}$ and $j = 1, 2, \ldots, n_2^{'}$. Where $int16$ and $round$ are the typecasting and rounding functions respectively in MATLAB.

Step 4 **Chaotic row-column matrix shuffling:** Replace the original approximation coefficients A_J^R, A_J^G and A_J^B by the obtained wavelet packet coefficients $A_J^{R'}$, $A_J^{G'}$ and $A_J^{B'}$ respectively to get the layers I_4^R, I_4^G and I_4^B. Then apply row-column matrix shuffling by the following process:

$$f_1(i,j) = I_4^R(i, key_{11}(i,j)),$$
$$g_1(i,j) = I_4^G(i, key_{11}(i,j)),$$
$$h_1(i,j) = I_4^B(i, key_{11}(i,j)),$$
$$I_5^R(i,j) = f_1(key_{12}(i,j), j),$$
$$I_5^G(i,j) = g_1(key_{12}(i,j), j),$$
$$I_5^B(i,j) = b_1(key_{12}(i,j), j).$$

Step 5 Compose each layer I_5^R, I_5^G and I_5^B to get an encrypted image I_6.

4.2. Decryption algorithm

We can obtain the decrypted image by performing backward steps in the encryption process, and inverse wavelet packet transform.

4.3. Encryption/decryption outputs

Outputs of the proposed algorithm on standard images (Baboon, Peppers, and Fruits) are shown in figure 4.

5. Performance and security analysis

From a cryptanalysis point of view, a secure encryption algorithm must resist various kinds of attacks (such as statistical, brute force, cipher-image, and plain-image attacks). The proposed algorithm has gone under multiple tests to check its robustness. Moreover, the efficiency of the proposed algorithm is shown through time complexity.

1. **Chaotic 3D diffusion:** This step involves shuffling and diffusion of pixel values in the 3D. It offers inter shuffling and diffusion between R, G, and B layers.

2. **2D chaotic diffusion:** Inter shuffled and diffused layers obtained from the above step are again shuffled and diffused at the second time once again using 2D chaotic diffusion.

3. **Chaotic diffusion:** We applied diffusion on the approximation part of the discrete wavelet packet coefficients, where most of the information about the image is available at each level. We ignore other wavelet packet coefficients, because most of the values are closed to zero, and almost all information about the image is available in the approximation part of the wavelet packet coefficients at each level. This property of wavelets tells a suitable environment made by wavelet packet transform for wireless communications.

4. **Chaotic row-column matrix shuffling:** To introduce an additional layer of security, all the coefficients are concatenated and shuffled according to permutation involved in the key matrix.

5.1. The time complexity of the proposed algorithm

To evaluate the time complexity of the proposed algorithm, we need to calculate the time involve in each step and then add them together to obtain the total time involved during the entire process: 2D & 3D chaotic diffusion takes time complexity $O(N_1 \times N_2 \times 3)$ and row-column diffusion in wavelet packet domain reserves time complexity $O(J \times N_1 \times N_2 \times \log_2(N_1 N_2))$. So the total time complexity is $O(J \times N_1 \times N_2 \times \log_2(N_1 N_2))$. The proposed encryption and decryption process are taking an average 0.672 and 0.602 seconds for an image of size $512 \times 512 \times 3$ in Intel® Core^{TM} i7-6700K CPU@ 4.00 GHz Processor, 32GB RAM, Windows 10 Pro and MATLAB R2021a. Hence, the proposed algorithm is good enough in terms of time complexity.

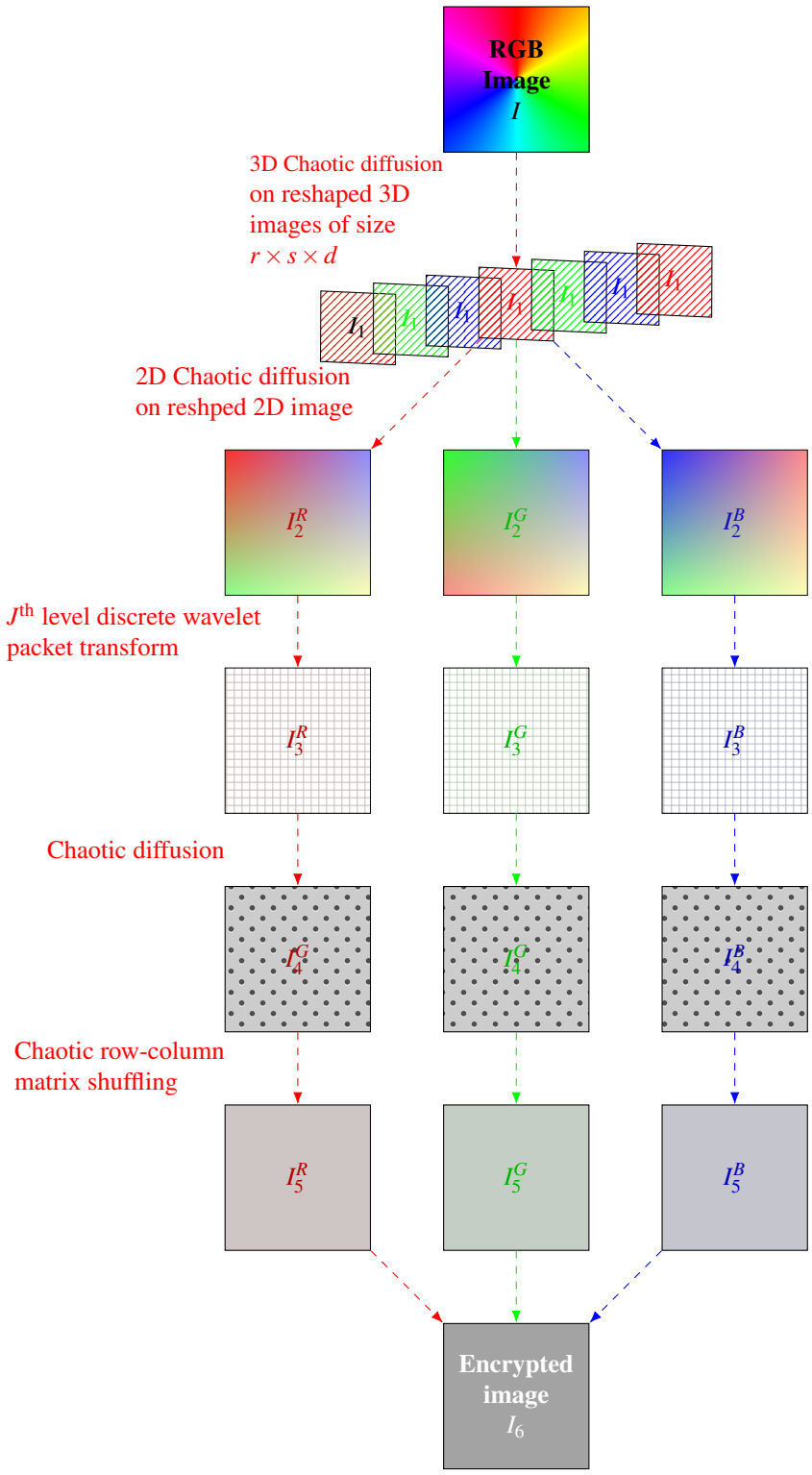

Figure 3: Flowchart of encryption algorithm.

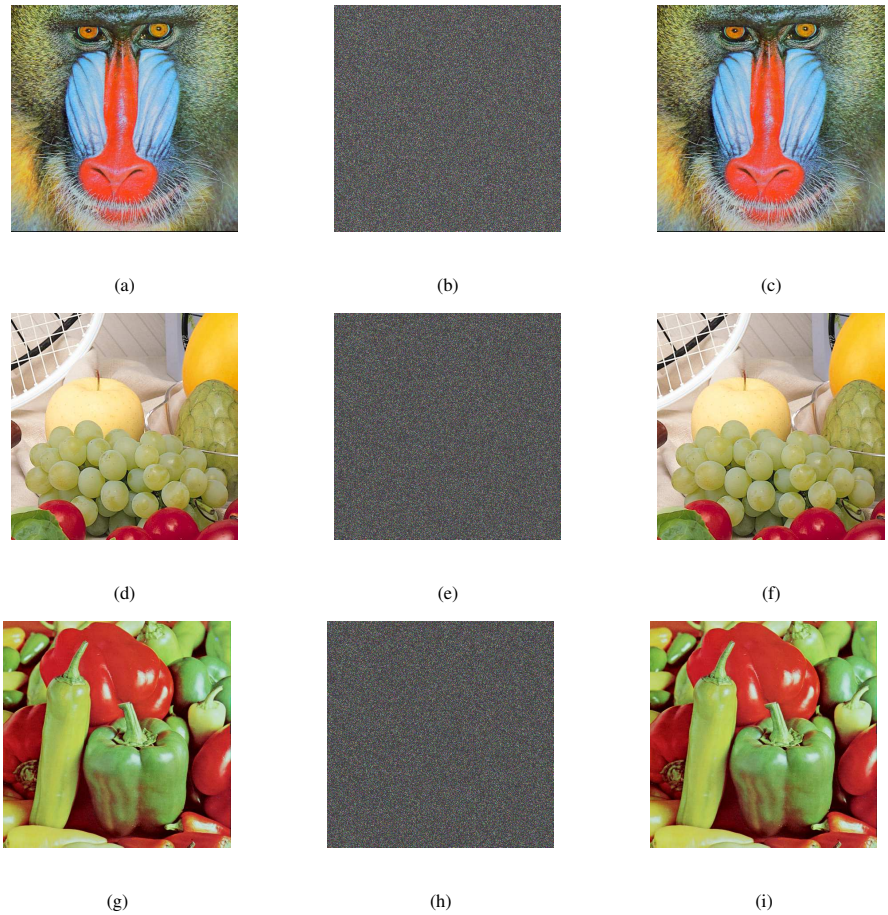

Figure 4: Outputs of the proposed algorithm on standard images:(a) Baboon original image, (b) Baboon encrypted image, (c) Baboon decrypted image, (d) Peepers original image, (e) Peepers encrypted image, (f) Peepers decrypted image, (g) Fruits original image, (h) Fruits encrypted image, (i) Fruits decrypted image.

5.2. Histogram analysis

A histogram is helpful for an adversary to analyze a graphical view of the data distribution of the encrypted and original images. Histogram analysis for Baboon, Fruits and Peppers are shown in figure 5-7, and these figures indicate that each distribution cannot be identifiable statistically. More importantly, the proposed encryption algorithm produces different histograms than the original image histogram, which would be pretty hard to get back the original image from the encrypted image histogram.

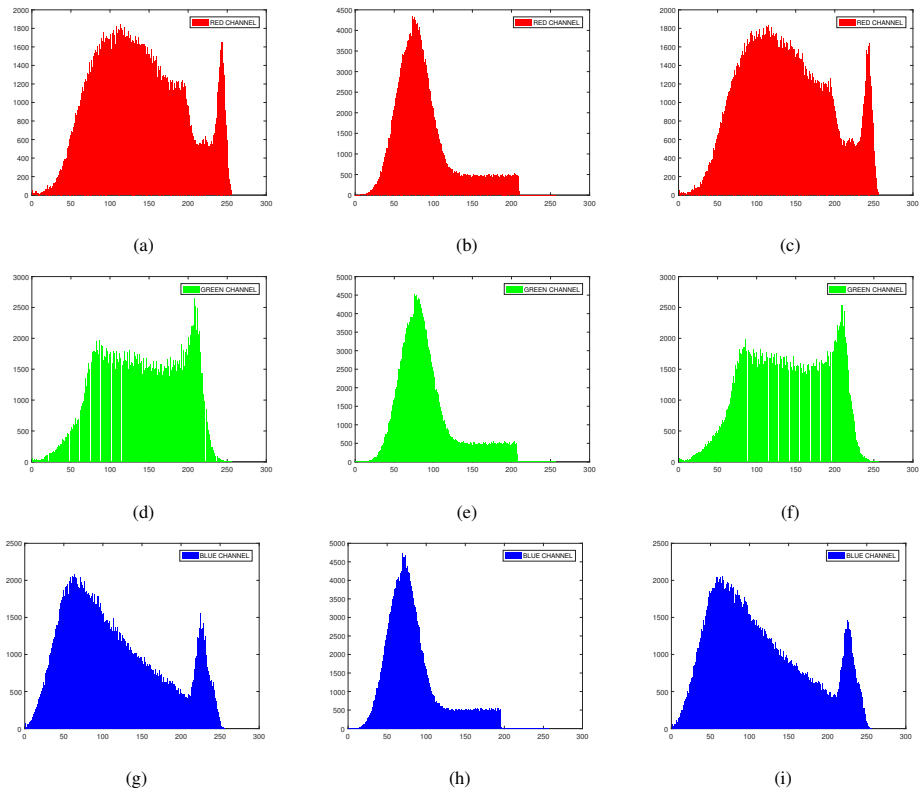

Figure 5: Histogram analysis: (a) Baboon original red plane, (b) Baboon encrypted red plane, (c) Baboon decrypted red plane, (d) Baboon original green plane, (e) Baboon encrypted green plane, (f) Baboon decrypted green plane, (g) Baboon original blue plane, (h) Baboon encrypted blue plane, (i) Baboon decrypted blue plane.

5.3. The mean square error and peak signal-to-noise ratio

The Mean Square Error (MSE) is an essential measure to understand the perfect recovery of the original image from the encrypted one, possibly valid for the proposed algorithm as the score is zero between the original and encrypted. Further, its good to compare the score of MSE for the original image and decrypted image to know the robustness of the proposed algorithm. The formula for calculating the MSE score is as follows

$$MSE = \frac{1}{N_1 \times N_2} \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} [f(i,j) - f_o(i,j)]^2$$

where (i, j) represents the position of pixels for $i = 1, 2, \ldots, N_1$ and $j = 1, 2, \ldots, N_2$. The intensity functions for encrypted/decrypted and original images are f and f_o, respectively. Data presented in Table 1 shows that the proposed algorithm is lossless.

Peak Signal-to-Noise Ratio (PSNR) is another crucial measure to analyze the ratio between a maximum value of a signal/image (power) and the power of distorting noise that affects the signal/image. One can calculate the PSNR between the original and encrypted/decrypted image using the formula as follows

$$PSNR = 20 \times log \frac{MAX_I^2}{\sqrt{MSE}},$$

where MAX_I is the maximum possible pixel value of the image. The PSNR values between original and encrypted images are shown in Table 1.

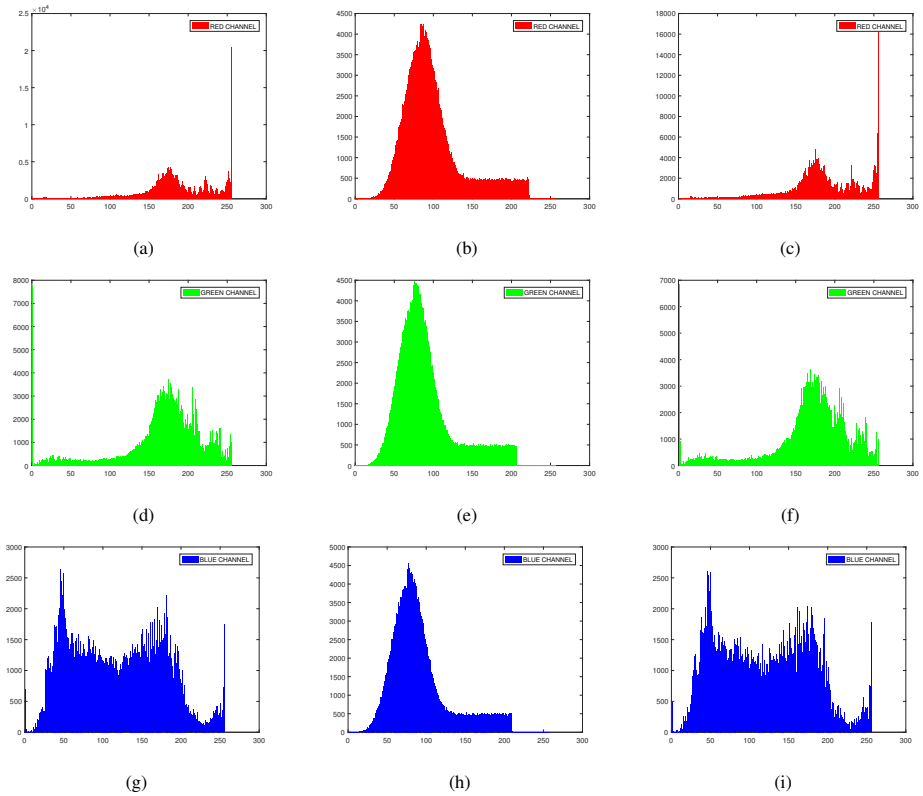

Figure 6: Histogram analysis: (a) Peepers original red plane, (b) Peepers encrypted red plane, (c) Peepers decrypted red plane, (d) Peepers original green plane, (e) Peepers encrypted green plane, (f) Peepers decrypted green plane, (g) Peepers original blue plane, (h) Peepers encrypted blue plane, (i) Peepers decrypted blue plane.

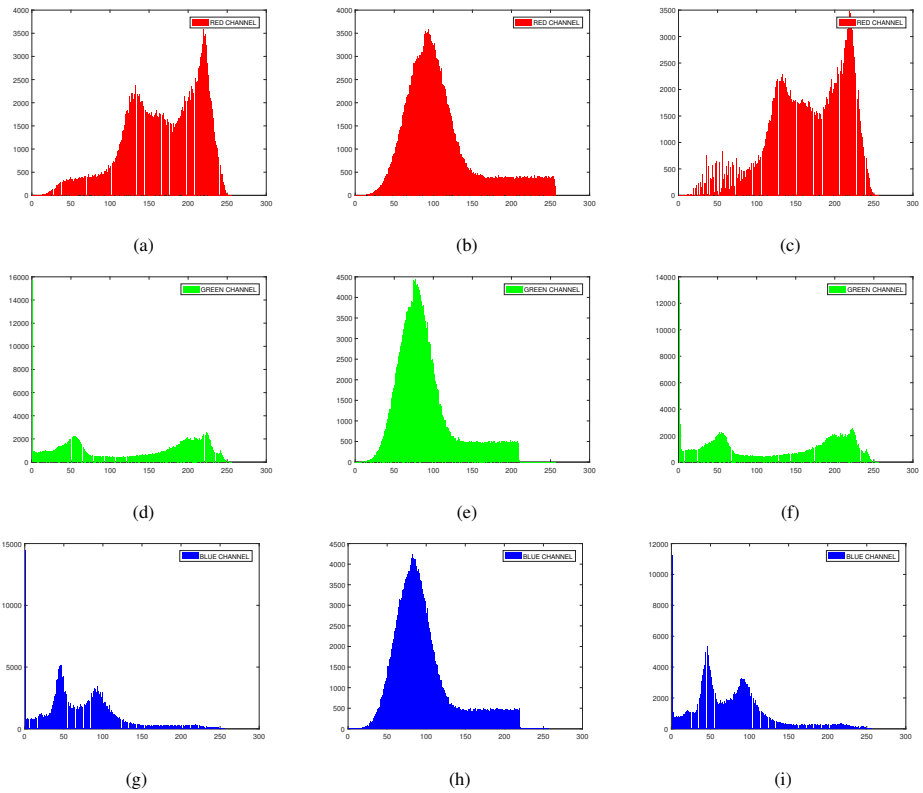

Figure 7: Histogram analysis: (a) Fruits original red plane, (b) Fruits encrypted red plane, (c) Fruits decrypted red plane, (d) Fruits original green plane, (e) Fruits encrypted green plane, (f) Fruits decrypted green plane, (g) Fruits original blue plane, (h) Fruits encrypted blue plane, (i) Fruits decrypted blue plane.

Table 1: Mean square error and peak signal to noise ratio between original and encrypted images of Baboon, Fruits, and Peppers, respectively.

Image planes	Baboon		Fruits		Peppers	
	MSE	PSNR	MSE	PSNR	MSE	PSNR
R	3.0859e+04	-2.8177	3.7970e+04	-3.7183	3.1185e+04	-2.8633
G	2.8245e+04	-2.4334	2.7834e+04	-3.1088	3.0156e+04	-2.7177
B	3.1428e+04	-2.9046	2.6223e+04	-1.8353	2.4972e+04	-2.8984

5.4. Size of the key space

The initial parameters used to generate the 4D hyper-chaotic system that are $x_r(0)$, $y_r(0)$, $z_r(0)$ and $u_r(0) = r_0$, $x_g(0)$, $y_g(0)$, $z_g(0)$, $u_g(0) = g_0$, $x_b(0)$, $y_b(0)$, $z_b(0)$ and $u_b(0) = b_0$. These initial parameters are secret keys of the proposed algorithm. All these parameters are sensitive up to 15th decimal places. And the type of wavelet name is also used as a secret key. So the size of the key space is $1 + 10^{12\times 15} = 10^{180} + 1 \approxeq 2^{597}$. The proposed key space is strong enough to resist brute force attack infeasible.

5.5. Marginal key change analysis

Suppose an adversary is trying to get the original image by guessing keys very close to the original keys. The output of the decryption process by the marginal change in keys is shown in figure 8. Suppose an adversary can access all the keys except only one key, then it is impossible to get the original image; see all cases in figure 8 except figure 8(m). Hence, as a result, with all the possible keys except only one corresponding key, the attacker cannot recover the correct image.

5.6. Cropping attack analysis

The cropping analysis plays a prime role while transferring confidential images from one place to another through a network. The encrypted image has been cropped and analyzed for data loss. The main advantage of doing this step is to check whether the decrypted image is visible or not after cropping the encrypted image by an adversary. From figure 9, we can claim that the proposed algorithm provides satisfactory results.

5.7. Communication channel noise analysis

The algorithm should provide excellent quality images on the receiver end when affected by noise during transmission. The proposed algorithm has been tested by salt & pepper noise. Hence, the results depicted in figure 10 confirm that the proposed algorithm is free from noise attacks.

6. Comparison

This section compares the proposed algorithm with the existing compatible algorithms for key space as shown in table 2 and corresponding to run-time are shown in tables 3 and 4.

Table 2: Comparison of the key space with the existing compatible algorithms.

Algorithm	Key space
Proposed algorithm	$10^{180} + 1$
Ref. [17]	10^{169}
Ref. [18]	$> 10^{78}$
Ref. [19]	$\approx 10^{63}$
Ref. [20]	$\approx 1.0368 \times 10^{114}$ (fixed size)

7. Conclusion

We propose a new secure RGB image encryption model based on 4D hyper-chaotic system involving discrete wavelet packet transform. The proposed algorithm offers an entirely efficient diffusion and confusion process on encrypted images. A formula for keyspace is provided, and it resists all types of attacks due to sensitivity to the initial conditions. The proposed algorithm is fast and efficient in terms of computation time. We tested the proposed algorithm for standard images (Baboon, Fruits, and Peppers). The data presented in table 3, and table 4 show that the proposed algorithm is fast, and sound compared to existing competing algorithms. The algorithm can also withstand cropping and noise attacks (can be seen in figure 9 and figure 10). Hence, the proposed algorithm can opt for practical use.

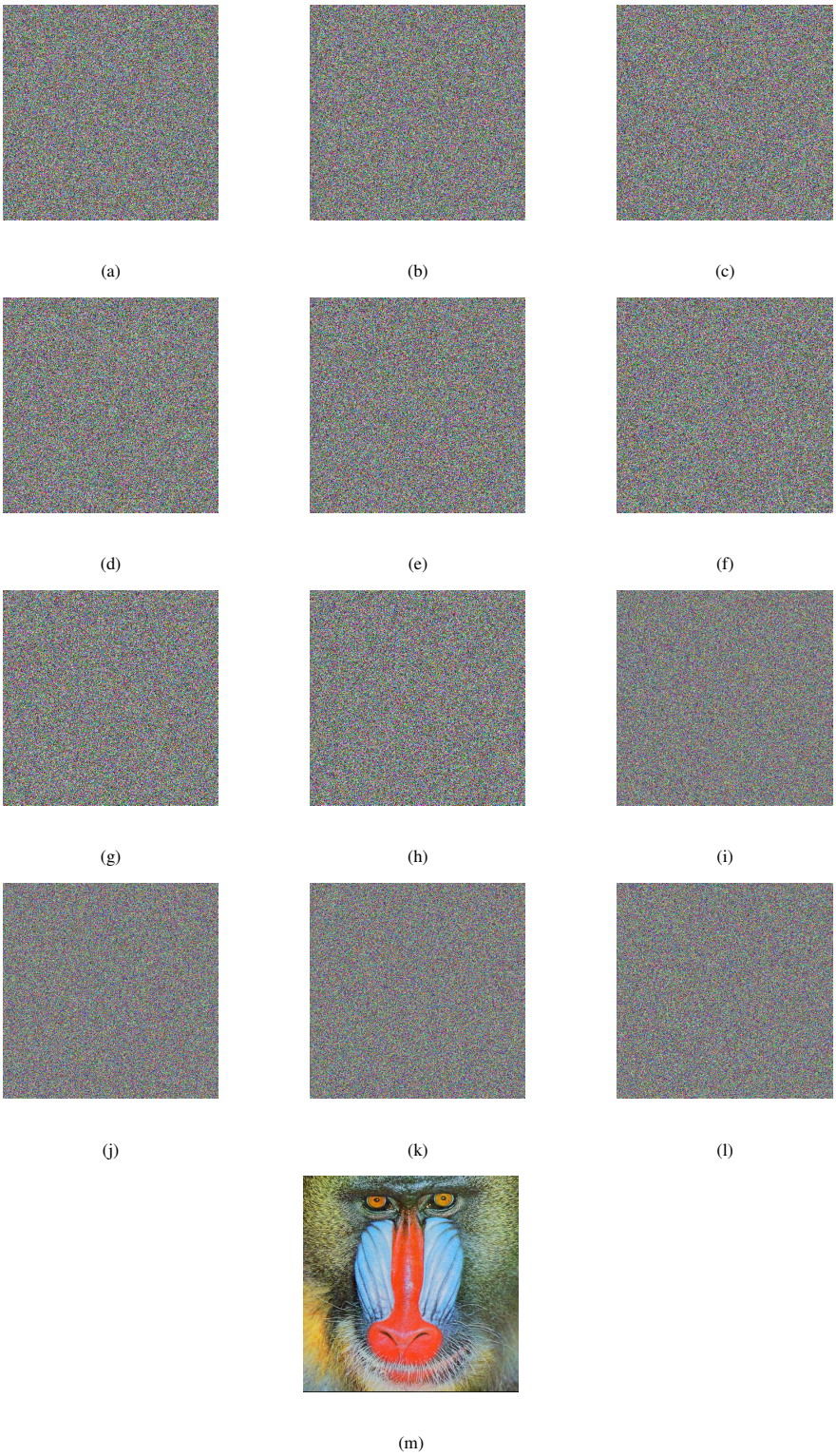

Figure 8: Marginal key change analysis: wrongly decrypted image with marginal change of 1^{-15} in (a) $x_r(0)$, (b) $y_r(0)$, (c) $z_r(0)$, (d) $u_r(0)$,, (e) $x_g(0)$, (f) $y_g(0)$, (g) $z_g(0)$, (h) $iu_g(0)$, (i) $x_b(0)$, (j) $y_b(0)$, (k) $z_b(0)$, (l) $u_b(0)$, (m) correctly decrypted image.

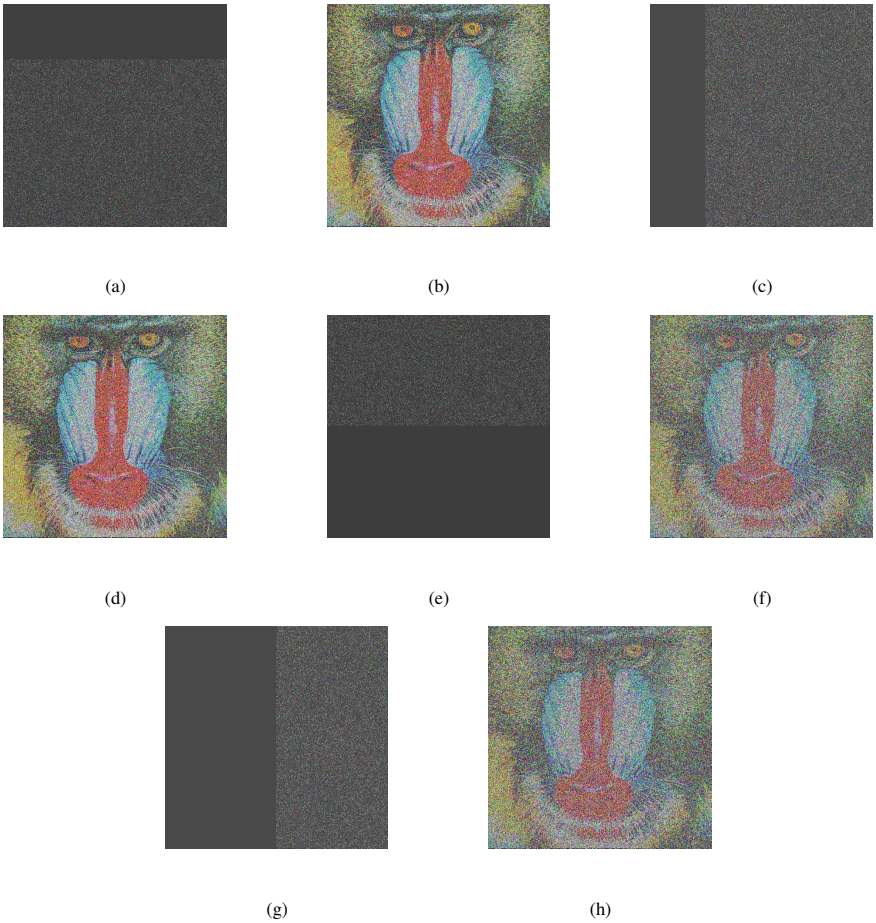

Figure 9: Cropping attack analysis: (a) 25% cropping of encrypted image in horizontal direction, (b) decrypted result, (c) 25% cropping of encrypted image in vertical direction, (d) decrypted result, (e) 50% cropping of encrypted image in horizontal direction, (f) decrypted result, (g) 50% cropping of encrypted image in vertical direction, (h) decrypted result.

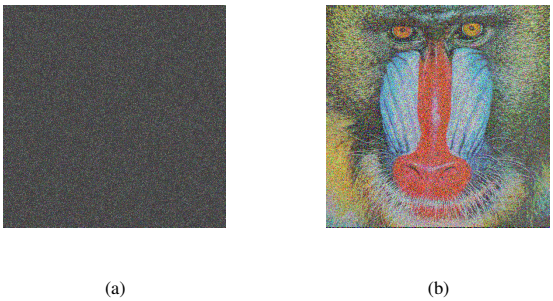

Figure 10: Noise attack analysis: (a) attacked encrypted image with Salt and Pepper noise, (d) decrypted image.

Table 3: Run-time of encryption/decryption algorithm for a set of wavelet packet functions in seconds.

Wavelet name	Baboon		Fruits		Peppers	
	Encryption	Decryption	Encryption	Decryption	Encryption	Decryption
db4	0.560292	0.563421	0.507620	0.568563	0.527547	0.565897
Sym1	0.588291	0.510592	0.521794	0.486568	0.501763	0.588728
Sym2	0.537672	0.541572	0.570864	0.556375	0.544631	0.518596
Coif2	0.580895	0.500256	0.509653	0.519567	0.534951	0.543181
fk18	0.524030	0.496448	0.517595	0.492555	0.516887	0.509851

Table 4: Comparison of the run-time of the proposed algorithm.

Algorithm	Encryption running time (in seconds)
Proposed algorithm	<0.6
Ref. [16]	38.292
Ref. [27]	1.686
Ref. [28]	0.719
Ref. [29]	3.704

Acknowledgment

The author is thankful to the Science & Engineering Research Board, Government of India, provides financial support through project file no. YSS/2015/000930.

Declaration of Competing Interest

The author declares that he has no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Preishuber, T. Hütter, S. Katzenbeisser, A. Uhl, Depreciating motivation and empirical security analysis of chaos-based image and video encryption, IEEE Transactions on Information Forensics and Security, 13(9), 2137–2150, (2018).

[2] M. Kumar, A. Saxena, S.S. Vuppala, A Survey on chaos-based image encryption techniques, in Multimedia Security Using Chaotic Maps: Principles and Methodologies, 1–26, (2020).

[3] S. Noshadian, A. Ebrahimzade, & J.S. Kazemitabar, Breaking a chaotic image encryption algorithm. Multimed Tools Appl, 79, 25635–25655, (2020).

[4] C. Li, D. Lin, B. Feng, J. Lü, F. Hao, Cryptanalysis of a chaotic image encryption algorithm based on information entropy, IEEE Access, 6, 75834–75842, (2018).

[5] G. Ye, C. Pan, X. Huang, Z. Zhao, J. He, A chaotic image encryption algorithm based on information entropy, Int. J. Bifurc. Chaos, 28, 1850010 (2018).

[6] Y. Dou, M. Li, Cryptanalysis of a new color image encryption using combination of the 1D chaotic map, Appl. Sci., 10(6), 2187, (2020).

[7] M. Kumar, A. Vaish, Encryption of color images using MSVD in DCST domain, Optics and Lasers in Engineering, 88, 51–59, (2017).

[8] J. M. Vilardy, S. Millán María, E. Pérez-Cabré, Non linear image encryption using a fully phase nonzero-order joint transform correlator in the Gyrator domain, Optics and Lasers in Engineering, 89, 88–94, (2017).

[9] L. Sui, M. Xin, A. Tian, H. Jin, Single-channel color image encryption using phase retrieve algorithm in fractional Fourier domain, Optics and Lasers in Engineering, 51, 1297–1309, (2013).

[10] S. Liu, J. T. Sheridan, Optical encryption by combining image scrambling techniques in fractional Fourier domains, Optics Communications, 287, 73–80, (2013).

[11] Y. Qin, Z. Wang, Q. Pan, Q. Gong, Optical color-image encryption in the diffractive-imaging scheme, Optics and Lasers in Engineering, 77, 191–202, (2016).

[12] D. Kong, X. Shen, Multiple-image encryption based on optical wavelet transform and multichannel fractional Fourier transform, Optics & Laser Technology, 57, 343–349, (2014).

[13] G. Bhatnagar, Q. M. J. Wua, Discrete fractional wavelet transform and its application to multiple encryption, Information Sciences, 223, 297–316, (2013).

[14] A. Kanso, M. Ghebleh, An algorithm for encryption of secret images into meaningful images, Optics and Lasers in Engineering, 90, 196–208, (2017).

[15] N. Taneja, B. Raman, I. Gupta, Selective image encryption in fractional wavelet domain, Int. J. Electron. Commun. (AEÜ), 65, 338–344, (2011).

[16] M. Kumar, K. L. S. Josyula. An interlaced secure algorithm for RGB image encryption in wavelet packet domain, International Journal of Wavelets, Multiresolution and Information Processing, 14, 1650008, (2016).

[17] X. Wu, D. Wang, J. Kurths, H. Kan, A novel lossless color image encryption scheme using 2D DWT and 6D hyperchaotic system, Information Sciences, 349–350 (2016).

[18] Y. Luo, M. Dub, J. Liu, A symmetrical image encryption scheme in wavelet and time domain, Commun Nonlinear Sci Numer Simulat, 20, 447–460, (2015).

[19] A. Belazi, A. A. A. El-Latif, A. V. Diaconu, R. Rhouma, S. Belghith, Chaos-based partial image encryption scheme based on linear fractional and lifting wavelet transforms, Optics and Lasers in Engineering, 88, 37–50, (2017).
[20] H. Liu, X. Wang, Color image encryption using spatial bit-level permutation and high-dimension chaotic system, Optics Communications, 284, 3895-3903, (2011).
[21] R. Coifman, Y. Meyer and M. V. Wickerhauser, Size properties of wavelet packets, *Wavelets and Their Applications*, Jones, and Bartlett, Boston, MA, 453–470, (1992).
[22] R. Coifman, Y. Meyer and M. V. Wickerhauser, Wavelet analysis and signal processing, *Wavelets and Their Applications*, Jones, and Bartlett, Boston, MA, 153–178, (1992).
[23] S. Mallat, *A wavelet tour of signal processing*, Academic Press, San Diego, 1999.
[24] D.K. Ruch, P.J. Van Fleet, Wavelet theory: an elementary approach with applications. New Jersey (NJ): Wiley; 2009.
[25] I. Daubechies, *Ten Lectures on Wavelets*, SIAM Philadelphia, Pennsylvania, 1992.
[26] R. S. Pathak, *The Wavelet Transform*, Atlantis Press/World Scientific, Amsterdam, Paris, 2009.
[27] C.K. Huang, H.H. Nien, Multi chaotic systems based pixel shuffle for image encryption, Opt. Commun., 282, 2123–2127, (2009).
[28] X. Zhang, G. Zhu, S. Ma, Remote-sensing image encryption in hybrid domains, Opt. Commun., 285(7), 1736–1743, (2012).
[29] S. Lian, J. Sun, Z. Wang, A block cipher based on a suitable use of chaotic standard map, Chaos Solitons Fractals, 26, 117–129, (2005).