Tuna Swarm Optimization with 3D-chaotic map and DNA encoding for image encryption with lossless image compression based on FPGA

Original Scientific Paper

Sunil B. Hebbale
Faculty, KLECET Chikodi, Dt. Belagavi,
Karnataka 591201, India.
sunilhebbale123@gmail.com

V. S. Giridhar Akula
KORM College of Engineering,
Thadigotla, Kadapa, Andhra Pradesh, India
giridharakula456@gmail.com

Parashuram Baraki
Department of CS&E Smt. Kamala and
Sri.Venkappa M. Agadi College of Engineering and Technology,
Lakshmeshwar, Dist. Gadag, Karnataka, India
parashurambaraki456@gmail.com

Abstract *– Images and video-based multimedia data are growing rapidly due to communication network technology. During image compression and transmission, images are inevitably corrupted by noise due to the influence of the environment, transmission channels, and other factors, resulting in the damage and degradation of digital images. Numerous real-time applications, such as digital photography, traffic monitoring, obstacle detection, surveillance applications, automated character recognition, etc are affected by this information loss. Therefore, the efficient and safe transmission of data has become a vital study area. In this research, an image compression–encryption system is proposed to achieve security with low bandwidth and image de-noising issues during image transmission. The Chevrolet transformation is proposed to improve image compression quality, reduce storage space, and enhance de-noising. A 3D chaotic logistic map with DNA encoding and Tuna Swarm Optimization is employed for innovative image encryption. This optimization approach may significantly increase the image's encryption speed and transmission security. The proposed system is built using the Xilinx system generator tool on a field-programmable gate array (FPGA). Experimental analysis and experimental findings show the reliability and scalability of the image compression and encryption technique designed. For different images, the security analysis is performed using several metrics and attains 32.33 dB PSNR, 0.98 SSIM, and 7.99721 information entropy. According to the simulation results, the implemented work is more secure and reduces image redundancy more than existing methods.*

Keywords*: Image encryption, chaotic maps, decryption algorithm, optimization algorithm, DNA encoding, Logical operations, image compression*

1. INTRODUCTION

The Internet has evolved into an essential sort of technology in the modern day. It applies to various fields. Multimedia is a prevalent form of communication in modern society. Numerous real-world applications, such as medical imaging systems, radar transmission, military systems, etc., may use this technology. Multimedia storage and transmission are still crucial. The transmission of data over the Internet must also be secure, which is a key factor [1-3]. Currently, security technologies such as data encryption, digital signature, and trusted routing are utilized to transmit data securely. Images are a popular kind of media in all sectors. Consequently, encrypted communication is vital as well as of great interest. The three main areas of study are compression of images, encryption, and image processing. De-noising, encryption, and image compression are all subject areas of this article, which will deal with image compression and de-noising [4-6].

The primary goal of compressing images is to reduce both storage space and network traffic. The source encoder transforms the input image into a minimal representation. They are built using several transforms, including DFT, DCT, and DWT. The DWT is a computationally efficient method, and the JPEG-2000 standard has embraced it [7, 8]. Even though the DWT can reach efficient compression ratios, it has significant disadvantages, such as a lack of directionality, aliasing, shift variance, and oscillations. The complicated WT may overcome these restrictions but at a high computational cost. In addition, the FWPT may be applied to signals with large fractional frequency components, although it has a greater computing cost than the DWT [9, 10].

In addition to compression, images should be encrypted before transferring through insecure lines of communication. Since the encryption process destroys the link between pixels, it is often performed after compression [11, 12]. There are several accessible encryption protocols, and there is always a trade-off between computational cost and encryption strength. The encryption of images based on a chaotic system was extensively explored, along with its intrinsic properties revealing the intimate link between cryptography and chaotic systems and cryptography.

There are two types of cryptosystems based on chaotic maps: Two-dimensional (2D) and one-dimensional (1D) chaotic maps. 1D chaotic map-based encryption approaches are insufficiently secure due to various constraints. Their basic orbits and starting and control parameters were predictable [13-15]. Therefore, traditional 1D chaotic systems are inappropriate for encrypting images. Recently, a complex chaotic system based on Chebyshev and 1D Sine maps and a Sine Cosine composite function system is described. 2D chaotic maps feature more complicated architecture and better chaos than 1D maps, making them suited for data encryption. 2D chaotic maps include the 2D Logistic-Modulated-Sine Coupling-Logistic Chaotic Map, two-dimensional Logistic-Adjusted-Sine map, and the 2D sine chaotification system. The implementation of these maps was feasible [16, 17]. These suggested 2D maps perform better and have broader chaotic ranges than the traditional existing mapping. Additionally, greater chaotic behaviors are seen in hyper-chaotic systems. Furthermore, a random and dynamic combination of the hybrid hyper-chaotic map is suggested to attain excellent encryption performance [18-20].

The existing methods that were presented simply encrypt the plain image. The most efficient and secure approach for transmitting data is compression followed by encryption with noise reduction. To provide noiseless, secure, and rapid image transmission, the proposed model conducts image de-noising, compression, and security. To enhance image quality, minimize storage space, and achieve superior de-noising, tetrolet transformation-based image compression is recommended in this research. Then, encryption and decryption of grayscale images using 3D chaotic maps and the DNA encoding technique are proposed. In addition, Tuna Swarm Optimization is presented, which may significantly increase the image's encryption speed and transmission security.

In digital signal processing systems, FPGA has become the primary paradigm for high-performance system implementation, particularly in image processing applications. In addition, FPGA can build high-performance signal processing system designs at fast speeds. In other words, the Xilinx system generator (XSG) increases the capability of FPGA and offers efficient tools for designing an image encryption model in a manner that is compatible with MATLAB/Simlinkin.

The practical design of the proposed image encryption and compression algorithm is the main goal of this work. As a result, a high level of security must be guaranteed. Every system or piece of work already in existence in this subject has to do with software or DSP. Therefore, the particular system's portability is not very simple. Here, the proposed approach is applied to the huge VLSI domain. We are concentrating on the suggested work's front-end design and putting it into FPGA. This will increase speed and effectiveness while decreasing area, energy, latency, and expense. Therefore, an encryption-based chaotic system is built in this research using FPGA. The following is a summary of the important contributions to this paper:

- De-noising and lossless compression of image is implemented by Chevrolet transformation.
- Proposed 3D chaotic logistic map for the diffusion of pixels during encryption.
- The initial condition for the 3D employs an ASCII key with 64 bits. Bits undergo "bit-XOR" to obtain a decimal value.
- A cryptographic hash function SHA-512 is utilized to produce a hash value.
- The proposed Tuna Swarm Optimization can greatly improve encryption speed and secure image transmission.

The remaining sections are structured as follows: The relevant work reported by various researchers is summarized in section 2. The suggested image compression and encryption technique are detailed in Section 3. Section 4 describes the experimental implementation of both the chaotic system and the proposed encryption architecture using Xilinx. In section 5, the conclusion and future work are explored.

2. LITERATURE SURVEY

There are several image compression and encryption technique exists. However, achieving both strong security and compression at the same time is still a difficult challenge since both are conflicting. In this part, the idea of image encryption using chaotic maps, as well as image denoising and compression, is discussed.

Bhargava and Gangadharan [21] developed an HRRBF-based FABF on an FPGA utilizing RLG technology with integrated geometric and photometric weight computation and kernel result calculation techniques. RPGs were used to build the reversibly modified carry select adder, the multiplier, and the reversible adder subtractor which enhances speed and lowers delay, power, area, and execution time. TCAM was used as the main memory to speed up processing.

Maazouz, Toubal, Bengherbia, Houhou, and Batel [22] constructed a new discrete chaotic system using the hybrid classification approach; the state variables of this system were then utilized to build a new S-box substitution matrix. The significance of this matrix in establishing the algorithm's degree of security cannot be overstated. The advantages led further to the improvement of the cryptographic properties of the developed algorithm, which included the Feistel model and some AES components.

Kumar, Reddy, Rinaldi, Parameshachari, and Arunachalam [23] provided low-power, high-speed hardware solutions for the FPGA implementation of AES to protect data. This work does not use LUTs to implement AES SubBytes and InvSubBytes transformations. This design uses combinational logic circuits to convert SubBytes and InvSubBytes. This architecture removes unwanted delays and adds sub-pipelining to improve AES algorithm speed by not using LUTs. The modified positive polarity reed muller (MPPRM) design reduces total hardware needs. With MPPRM architecture in SubBytes phases, a mixed column architecture is implemented.

Zodpe and Sapkal [24] suggested a method for improving the encryption quality of the AES algorithm, as well as its implementation on FPGA. Initially, the modified AES algorithm's S-box values are created using the PN Sequence Generator. Second, the first encryption/decryption key is derived from the output of the PN Sequence Generator.

Hafsa, Gafsi, Malek, and Machhou [25] suggested combining a complicated chaotic PRNG with MAES. For a high-quality encryption key, a chaotic PRNG was recommended. Unpredictability, entropy, and complexity characterize the created key. Using four S-boxes enhanced the complexity of the MAES sub-bytes operation. Mix-columns and shift-rows transformations were removed to enhance complexity. Only four encryption rounds are done in a loop, saving time. The encryption data route executes a 32-byte block in parallel and one clock cycle.

Kumar, Balmuri, Marchewka, Bidare Divakarachari, and Konda [26] improved AES's key expansion to speed up subkey generation. The fork-join model of key expansion (FJMKE) structure was created to increase the speed of subkey creation while minimizing the hardware requirements of AES by eliminating the frequent calculation of secret keys. JFK-AES produces subkeys in half the time of the usual design.

Bhargava and Gangadharan [27] developed FPGA-based MRBF-based FBF architecture to reduce computing complexity, space, and power. MRBF was implemented using a quantum-dot cellular automaton-based carry select adder, which reduces space, power consumption, and latency. In addition, image de-noising utilizing MRBF-FBF was tested.

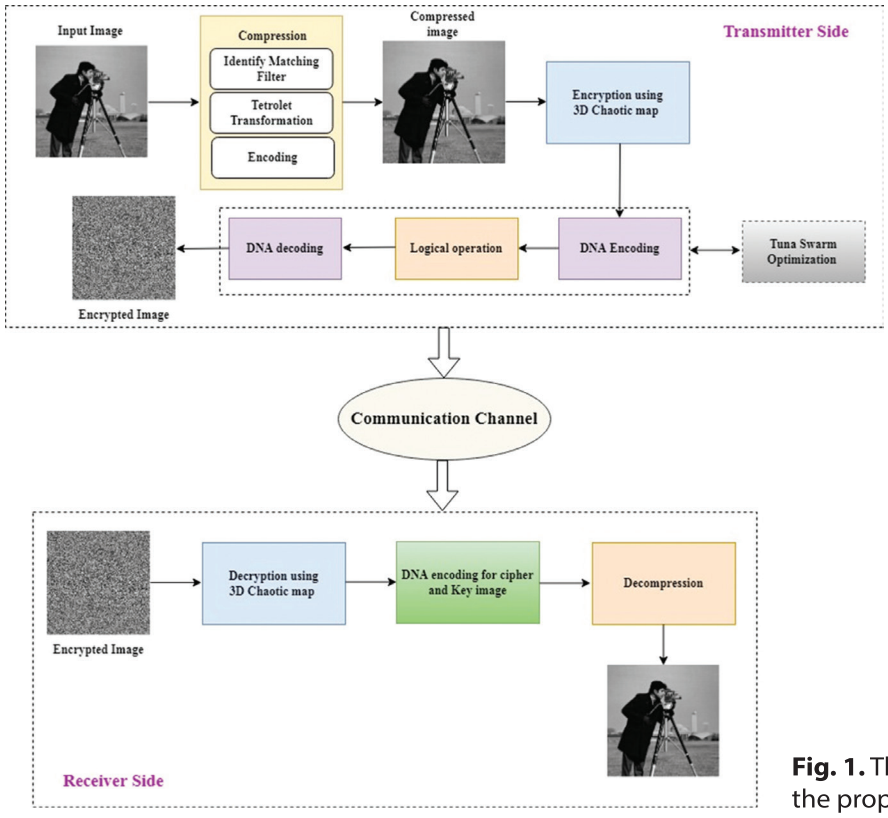


Fig. 1. The architecture of the proposed methodology

3. METHODOLOGY

The quality of digital images is diminished by noise interference. As image quality degrades, image diagnosis becomes more challenging. The transmission of massive data sets is the greatest hurdle in digital image transmission. The compression approach decreases the image's file size by decreasing the number of pixels, making it more suitable to store and deliver information across a variety of communication channels. Additionally, reducing network traffic will increase efficiency. The Chevrolet transformation is employed to remove noise and redundancy from digital images. This compressed image has been encrypted using chaotic maps. Fig. 1 depicts the block diagram for the proposed framework.

The primary components of the proposed work are the input image, image compression, compressed image, encryption module, encrypted image, inverse process, and reconstruction of the image. Haar model becomes Chevrolet by matching and rearranging tiles. The image is partitioned into 4x4 blocks. Each block's tetromino division reflects its image geometry. By combining four similar squares that are linked at one edge.

3.1. MATCHING TILE IDENTIFICATION

The Chevrolet transform is based on the Chevrolet decomposition method. The 4x4 input matrix of the image undergoes a standard Haar transformation.

The sum of the twelve detail coefficients is denoted as

$$CD_{sum} = \left|s(diagonal)\right| + \left|s(horizontal)\right| + \left|s(vertical)\right| \quad (1)$$

Where CD_{sum} represents the present information. The subsequent tile is used to determine the new tiles. Using the Haar transform, new information is acquired and stored in NDsum. After reviewing every tile, the procedure concludes. After that, tiles that match are chosen.

3.2. TETROLET TRANSFORM

For Chevrolet coefficients, the input image is divided into 4x4 pixels. Level-1 Haar approximation contains matched tetromino tiles of sub-images. Diagonal, horizontal, and vertical image transformation coefficients are approximations. Repeat the process to generate the images using the Tetrolet transform [28] is expressed as

$$I =\subset Z_2 \quad (2)$$

$$a = (a[x,y])(x,y) \in I \quad (3)$$

Where $N=2J, J \in N$

$$N_4(x,y) := 1,y),(i+1,y),(x,y-1),(x,y+1) \quad (4)$$

An edge index has three neighbors, but a vertex index has just two.

$$J : I \rightarrow \text{ with } J((x,y)) := jN + x \quad A\ set\ E = r \in N \quad (5)$$

$$\forall (x,y) \in I_v \exists (x,y) \in I_v : (x,y) \in N_4(x,y) \quad (6)$$

Here, I_v= let L: I_v →be the condition I_v = 4 for every set.

3.2.1 Tilings by tetrominoes

The tetrominoes are formed using the four-unit squares. This is joined around its edges, not simply at its corners. There are five distinct forms independent of rotations and reflections; these are known as free tetrominoes. The conventional 2D-Haar wavelet decomposition yields a unique tetromino partition.

Haar Wavelet model handles sample or native sample average changes in the input image matrix. Except for $k=l=0$, Haar transform coefficients reflect rows and columns of native pixel averages. This square measurement resulted in multiple "edge extractions" The second Haar transformation processes images as rows and columns. The transition involves adding and subtracting level one's parts. For stage one, the transformation can just be a straightforward component addition. However, the fundamental elements of the transformation matrix are present at levels greater than one. These parts result in decimal numbers (pixel values are integers), and even small decimal numbers can have a significant impact on higher-level value changes. To circumvent this limitation, the higher-level Approximation matrix is used.

$$k=2a+b-1 \quad (7)$$

Based on the aforementioned formula, a and b are defined clearly for k. Hence, the approximation matrix for level 3 (N=8) comprises 15 coefficients.

$$Haar\ transform = \begin{bmatrix} \begin{bmatrix} A & V \\ H & D \\ A & V \\ H & D \end{bmatrix} & \begin{bmatrix} A & V \\ H & D \\ A & V \\ H & D \end{bmatrix} \end{bmatrix} \quad (8)$$

Therefore, the grayscale image is compressed and noise free. These images are then encrypted further using 3D chaotic maps encoded with DNA.

3.3. IMAGE ENCRYPTION

The proposed encryption algorithms are detailed in this section. This method consists of the following encryption process:

- 3D chaotic map with a generation of symmetric keys
- Permutation
- Diffusion
- Key image generation using 3D and pixel confusion

3.3.1. 3D chaotic map with symmetric keys generation

The proposed method initializes the p, q, and r dimensions of a 3D-chaotic map using three symmetric keys. Symmetric keys are identical for both encryption and decryption. This method requires the sender and the recipient to have the same secret key. Consequently, in the proposed work, the receiver must hold identical keys to decode the encrypted image.

Three separate ways for creating symmetric map initialization keys, each with its approach, are discussed. Each method initializes the p, q, and r dimensions of a 3D chaotic logistic map with a random decimal integer between 0 and 1.

3.3.1.1 SHA-512

In the proposed methodology, SHA-512 [29] is utilized and the SHA-512 hash function of the compressed image generates the initial values and intermittent parameters of the chaotic system. The hash sequence of the plain image with 512 bits is obtained when the compressed image is given: $K = [k_1, k_2, \ldots, k_{64}]$.

$$Checksum = 2 + \frac{mod((h1+h2+h3+h4)*10^{14},256)}{255} \tag{9}$$

Where h1, h2, h3, and h4 are 8-bit sequences generating SHA-512 64 bits overall. Modulo functions as 'mod'. Then, the chaotic system is generated by the initial values.

3.3.1.2 Chebyshev key

This key initializes the 3D chaotic map's x dimension. Chebyshev's polynomial chaos map [30] is a chaos map. Orthogonal Chebyshev polynomials are used. This map's chaos is utilized for encryption. n-degree polynomials are defined by the equation (10):

$$k_{n+1} = \cos(p^* \arccos(k_n)) \tag{10}$$

Applying the Chebyshev polynomial formula provided in Equation 10 using the aforementioned parameters k_1 and p for n times. This map then produces a 1 × 65,356 -dimensional vector with chaotic negative and positive decimal values. The vectors are then transformed from decimal to binary values and shifted by one unit circularly. The matrix is transformed back to its original decimal form. The logical operation bit XOR is performed between the matrices. All the acquired values are put together, and the sum is transformed into a 64-bit binary. All the acquired values are put together, and the sum is transformed into a 64-bit binary. The acquired 64-bit binary is subdivided into eight sectors, and every eight bits is transformed into a decimal number. Applying the SHA-512 hash algorithm and utilizing Equation [10] generates a checksum result. The checksum value generated by the Chebyshev key generation process is utilized to initialize the 'p' parameter of the 3D chaotic map.

3.3.1.3 Prime key

The primary key [31] initializes the second dimension 'y' of the 3D chaotic mapping. Enter ASCII 64-bit characters. The characters entered are transformed into a 128-bit binary value. Only 64 bits are extracted and split into eight halves. Every bit is transformed into a decimal number. Applying the SHA-512 hash algorithm and utilizing Equation [9] generates a checksum result. The ASCII key generation method's checksum value is used to initialize the 'z' parameter of a 3D chaotic mapping. After initializing the values, the map is iterated T_2 times to produce chaotic values. Equation 11 normalizes p, q, and r vectors.

$$\begin{cases} x = \left\lfloor \mathrm{mod}\left((x^* T_2), image\ height\right) \right\rfloor \\ y = \left\lfloor \mathrm{mod}\left((y^* T_2), image\ height\right) \right\rfloor \\ z = \left| \mathrm{mod}\left((z^* T_2), image\ height\right) \right| \end{cases} \tag{11}$$

Where Image height = 256 for image size 512 x 512 and T_2 is a large number

3.3.2 Permutation of pixels

Pixel position permutation is a technique for unusually repositioning an image's pixels. This procedure may be carried out either arbitrarily or by employing a permutation key. The proposed approach employs the one and two dimension sequences of Level 1's 3D chaotic map as permutation keys for the permutation of columns and rows of the compressed image respectively.

3.3.2.1 Row permutation

To explain the permutation process, an example with eight-by-eight matrices is shown. The permutation of a row has four stages:

Step 1: Load the original image A and obtain its width and height, denoted by W and H, respectively.

Step 2: Iterate H times the x sequence present in equation (11) to get the permutation key of rows.

Step 3: Sort x in ascending order depending on its index to get the row permutation vector PROW = (Pr_1, Pr_2,..., Pr_H).

Step-4: Generate the permuted image for the row I utilizing the permutation vector P_{ROW} as specified in equation (7):

$$P(i, j + k(i)) = I(i, j) \tag{12}$$

Here, the permutated matrix row-wise is P, the input matrix is I, and the rows and columns index of the input matrix is (i, j).

3.3.2.2 Permutation of column

In column permutation, the same steps are taken as in row permutation:

Step 1: To construct the column permutation key, repeat the y sequence defined in equation (12) W times.

Step 2: Obtain the column permutation vector P_{Column} = (Pc_1, Pc_2... Pc_W) by sorting the series depending on its index in ascending order.

Step 3: Using the permutation vector P_{Column} and the equation below, obtain the permuted image for column C:

$$C(i + l(j), j) = P(i, j) \tag{13}$$

3.3.3 DNA encoding and decoding

DNA is a high molecular weight, double-stranded polymer with a double helix shape. Including adenine (A), thymine (T), guanine (G), and cytosine (C) as nitrogen-containing bases (C). A and T, as well as G and C, have complementary relationships.

8-bit binary represents each grayscale pixel. 00 and 11 are binary complements, as are 01 and 10. Each pixel in a grayscale image may be turned into a four-bit DNA sequence. Table 1 shows how a DNA nucleotide may be encoded using the complementary pairing concept.

Table 1. Encoding and decoding rules of DNA

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

3.3.4 XOR logical operation

DNA-encoded images and key images are XORed to encrypt and randomize the pixels. The key image is generated from a 3D chaotic map's 'z' dimension. The dimension 'z's-generated chaotic sequence is an [M × N] matrix. Where the input image is represented as M and N. Third step of the proposed encryption procedure encodes the key image with DNA using eight complementary principles. The DNA-encoded key image and the Level 3 image are logically XORed.

3.3.5 DNA diffusion and Tuna Swarm Optimization for optimized encrypted image

TSO algorithm [32] is proposed to obtain the channel's optimum DNA-encoded key image. Tuna is a marine carnivorous fish. Sizes of tuna species vary significantly. Tuna is an ocean-top predator that eats midwater and surface fish. They have an efficient and unique swimming technique (called fishtail form) that the body stays rigid while the tail swings swiftly. Despite swimming fast, the single tuna is slower than the agile fish. Tuna employ "group travel" to hunt. Intelligence helps them find and attack the target. These species have evolved smart foraging approaches.

The primary strategy is spiral foraging. When feeding, tuna swim in a spiral pattern to drive their prey into shallower water, where they may be successfully attacked. Next, is referred to as parabolic foraging. Each tuna swims in succession, creating a parabolic curve that encircles its target.

3.3.5.1 Initialization

TSO begins the optimization process by randomly creating starting populations in the search space,

$$X_i^{\text{int}} = rand.(ub - 1b) + 1b, \quad i = 1,2,\dots,NP, \tag{14}$$

Where, lower and upper boundaries of search space as 1b and ub, with the initial individual as X_i^{int}, rand is a random vector from 0 to 1 with uniform distribution and tuna populations represented as NP.

3.3.5.2 Spiral Foraging

When predators attack herring, sardines, and other small schooling fish, the entire group creates a dense, continually shifting structure, making it difficult to target a single individual. The tuna swarm creates a spiral to hunt. Most fish in the group lack directional sense, but when a few fish swims together in a given direction, the nearby fish will adjust their route sequentially until they form a large group with the same goal and begin hunting. Along with circling after food, tuna groups communicate. Each tuna follows the previous one, so they may exchange information. Based on these assumptions, the spiral foraging formula is:

$$X_i^{t+1} = \begin{cases} \alpha_1 \cdot \left(X_{best}^t + \beta \cdot \left|X_{best}^t - X_i^t\right|\right) + \alpha_2 \cdot X_i^t, & i = 1, \\ \alpha_1 \cdot \left(X_{best}^t + \beta \cdot \left|X_{best}^t - X_i^t\right|\right) + \alpha_2 \cdot X_{i-1}^t, & i = 2,3,\dots,NP, \end{cases} \tag{15}$$

$$\alpha_1 = a + (1-a) \cdot \frac{t}{t_{\max}}, \tag{16}$$

$$\alpha_2 = (1-a) - (1-a) \cdot \frac{t}{t_{\max}}, \tag{17}$$

$$\beta = e^{bl} \cdot \cos(2\pi b), \tag{18}$$

$$l = e^{3\cos(((t_{\max}+1/t)-1)\pi)}, \tag{19}$$

where b is a 0-1 random number, Maximum iterations t_{max}, iteration number is t, a determines how closely tuna follow the ideal individual in the first phase,), α_1 and α_2 regulate the likelihood of individuals to migrate to the optimum and preceding individuals, current best (food) as X_{best}^t, X_i^{t+1} is the ith individual in the iteration $t + 1$.

Tuna may use the search area around food as they forage in a spiral. Following the best forager is not as effective as group foraging when the best fails to find food. To start a spiral search, we generate a random search coordinate. Each member may explore a wider region, and TSO can explore globally. The particular mathematical model is presented here:

$$X_i^{t+1} = \begin{cases} \alpha_1 \cdot \left(X_{rand}^t + \beta \cdot \left|X_{rand}^t - X_i^t\right|\right) + \alpha_2 \cdot X_i^t, & i = 1, \\ \alpha_1 \cdot \left(X_{rand}^t + \beta \cdot \left|X_{rand}^t - X_i^t\right|\right) + \alpha_2 \cdot X_{i-1}^t, & i = 2,3,\dots,NP, \end{cases} \tag{20}$$

where X_{rand}^t is a random search space reference.

Metaheuristic algorithms initially explore globally before focusing on local exploitation. As the number of iterations increases, TSO adjusts spiral foraging reference locations from random to optimal. It is given in equation (21),

$$X_i^{t+1} = \begin{cases} \alpha_1 . \left(X_{rand}^t + \beta . \left| X_{rand}^t - X_i^t \right| \right) + \alpha_2 . X_i^t, & \mathrm{i} = 1 \quad if\ rand < \dfrac{t}{t_{\max}} \\ \alpha_1 . \left(X_{rand}^t + \beta . \left| X_{rand}^t - X_i^t \right| \right) + \alpha_2 . X_{i-1}^t, & i = 2,3,\ldots, NP, \\ \alpha_1 . \left(X_{rand}^t + \beta . \left| X_{rand}^t - X_i^t \right| \right) + \alpha_2 . X_i^t, & i = 1 \\ \alpha_1 . \left(X_{rand}^t + \beta . \left| X_{rand}^t - X_i^t \right| \right) + \alpha_2 . X_{i-1}^t, & i = 2,3,\ldots, NP, if\ rand \geq \dfrac{t}{t_{\max}} \end{cases} \qquad (21)$$

3.3.5.3 Parabolic Foraging

Additionally, eating in a spiral pattern, tunas can feed cooperatively in a parabolic shape. Concerning food, tuna creates a parabolic pattern. Moreover, tuna look for food by scanning their surroundings. Assuming that the chance of selection is 50% for each of these methods, they are executed concurrently. It is mathematically expressed as:

$$X_i^{t+1} = \begin{cases} X_{best}^t + rand . \left(X_{best}^t - X_i^t \right) + TF . p^2 . \left(X_{best}^t - X_i^t \right), & if\ rand < 0.5, \\ TF . p^2 . X_i^t, & if\ rand \geq 0.5, \end{cases} \qquad (22)$$

$$P = \left(1 - \frac{t}{t_{\max}} \right)^{(t / t_{\max})}, \qquad (23)$$

Where TF is a random number with a value of 1 or – 1.

3.3.5.4 Optimization through TSO

TSO is used in this stage to acquire the optimum mask sequence. At the beginning of the TSO optimization procedure, the search space population is generated at random. In each cycle, each individual chooses randomly between two foraging methods or regenerates the search space with probability *r*. In parameter setup simulation experiments, *r* will be examined. All TSO individuals are updated and computed till the completion of the optimization procedure, then the optimal tuna and its fitness value are obtained. Ultimately, the vector with the highest fitness value creates the optimized masks.

3.3.6 Final Encryption

The application of logical operations yields a new DNA-encoded matrix that is subsequently DNA decoded to decimal values. Using the same DNA complementary principles, the pixels are turned into binary bits, which may then be translated into decimal integers. After applying four stages of encryption, the resulting image is an optimal cipher image that is broadcast across the channel to the destination.

3.3.7 Decryption

To recover the image at the receiving end, reverse procedures are done. As symmetric keys are used in the proposed approach, identical keys are used at both the receiver and transmitting ends.

3.4 FPGA Implementation

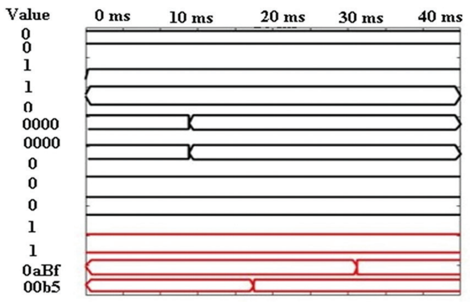
The complete design was encoded using the Verilog programming language. This design used 30820 slices and 61088 LUTs. Over 50 to 80 clock cycles of delay are required for the initial round. After that, we get the output at each clock cycle. The proposed method achieves a minimum clock period of 4.246 nanoseconds and a high clock frequency of 256.022 MHz, resulting in an efficiency of 14.15 Mbps/slice.

4. SIMULATION RESULTS AND DISCUSSIONS

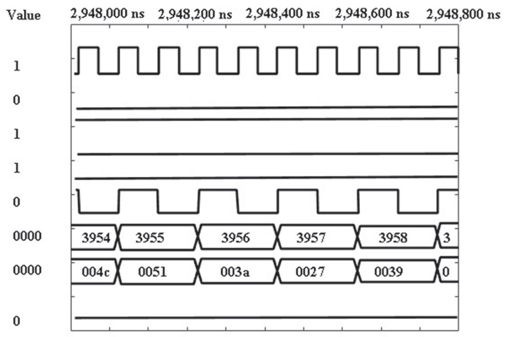
In software, the proposed compression and encryption technique was implemented, as well as security and compression performance evaluations. MATLAB is used for software implementation, with the Xilinx tool for FPGA implementation. The simulation results are shown in Figure 2 (1)-(c).

4.1 IMPLEMENTATION DETAILS

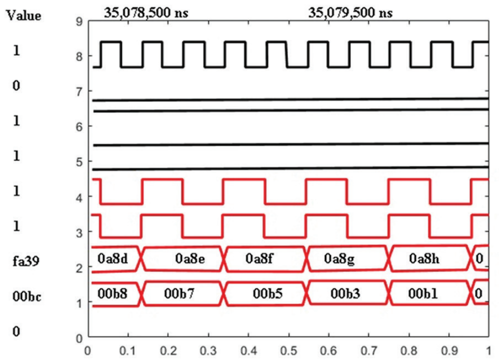
The proposed encryption is implemented in MATLAB version 10 using the Windows 7 operating system and an Intel Core i7 processor. The following analysis has been conducted. This section presents the evaluation results, highlighting image quality analysis, statistical analysis, and performance of the algorithm.



(a)



(b)



(c)

Fig.2. (a-c): Xilinx simulation results

The input gray image is encrypted using 3D chaos, permutation of pixels, optimum DNA encoding, and XOR logic. Chebyshev, prime, and ASCII of 64-bit are employed to create the 3D chaotic mapping during permutation. Permutation of a column and row pixels is performed to the chaotic map's initial two dimensions, p, and q. To diffuse an image, additional DNA encoding rules are employed. This optimization approach may significantly enhance the image's encryption speed and transmission security. The chaotic map's third dimension 'z' is employed for XOR operation. The key images are DNA-encoded. The confusion matrix is generated by XOR the encoded input and the key image.

4.2. STATISTICAL ANALYSIS

The implementation is carried out on eight 512×512 grayscale images with 256 gray levels. The images include Lena, a pepper, a baboon, a yacht, a boat, an airplane, Barbara, and a clock. Our methodology provides very secure denoised image compression, as shown by the outcomes of our experiments. The encryption and decryption of three images are shown in Table 2, and it reveals that the cipher image does not disclose any data from the compressed image. The following are subsections detailing the experimental results.

4.2.1. Compression ratio

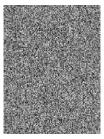
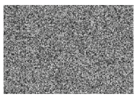
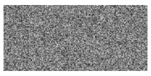
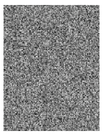
The PSNR value is the objective vertex that assesses the image quality after compression and decryption, and its mathematical description is as follows:

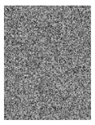
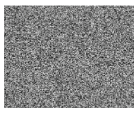
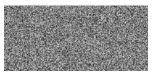
$$MSE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} (F(x,y) - f(x,y))^2$$
$$PSNR = 10.\log_{10}\left(\frac{255^2}{MSE}\right) \tag{24}$$

Where width and length are denoted as *W* and *H*, the original image as *f (x, y)*, and decrypted image as *F(x, y)*.

Table 2 shows the PSNR values for the various compression ratio values of various images. The image becomes increasingly distorted when PSNR decreases. When the CR is 4:3 or 2:1, the recovered image is essentially identical to the original image. When the CR is 4:1, the decrypted image is still recognizable, hence the image transmission is satisfied by the proposed compression encryption approach. Moreover, the performance of the proposed approach is increased by the use of the FPGA than using MATLAB. As the compression ratio increases, image quality degrades. Low compression improves image quality. The proposed tetrolet transform compresses normal visual data effectively.

Table 2. PSNR values for various CRs

Original image	Compression ratio	Encryption image	Decryption Image	PSNR with FPGA	PSNR with MATLAB
	4:3			42.5943	40.8791
	2:1			39.0267	37.6532
	4:1			37.6125	35.9912
	4:3			43.8712	40.9977
	2:1			41.9264	38.6541
	4:1			39.5187	35.7889

	4:3			40.816	37.5637
	2:1			38.1024	35.2908
	4:1			35.0013	33.3356

4.2.2 Histogram Analysis

Histogram analysis of a cipher image shows the strength of its security. The histogram of encrypted images has a uniform distribution of pixel values, whereas the histogram of cipher images is nonlinear. As a result, it can be said that the encryption is secure.

The varied bar heights in the histogram image represent the various occurrence frequencies of the data. Figure 3 displays the results of a histogram analysis of the suggested encryption and decryption procedure. It can be seen from the image that the plain image's non-uniform histogram has more common pixels in the 200–50 pixel range, meaning that certain pixels are less frequent than the rest. The distribution is uniform and distinct from the histograms of a plain image while the histogram of the encrypted image has an equal count of each pixel's occurrence. When an image's histogram is flat, it offers no important data to attackers. With the use of the histograms in Figure 3, we can state with confidence that the suggested encryption method is remarkably resistant to assaults.

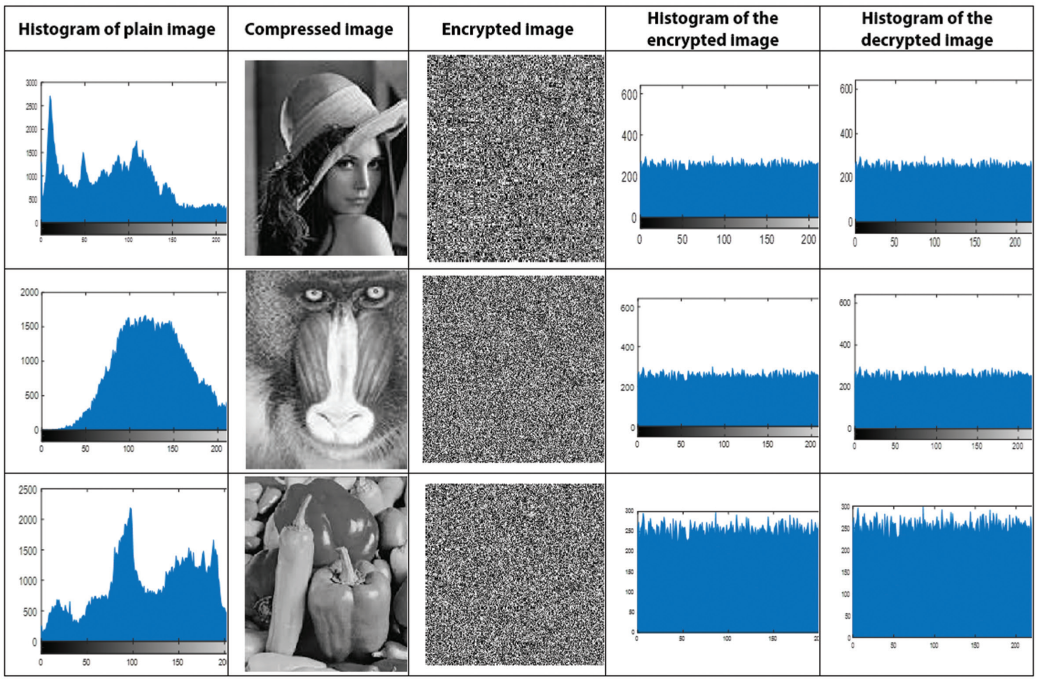


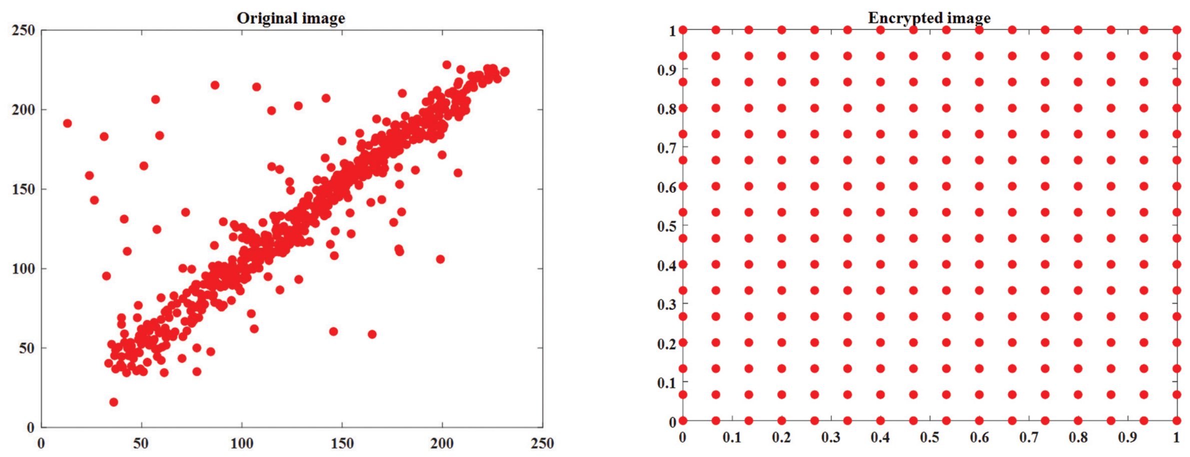
Fig. 3. Histogram analysis for the Encryption and Decryption process

4.2.3. Correlation coefficient test

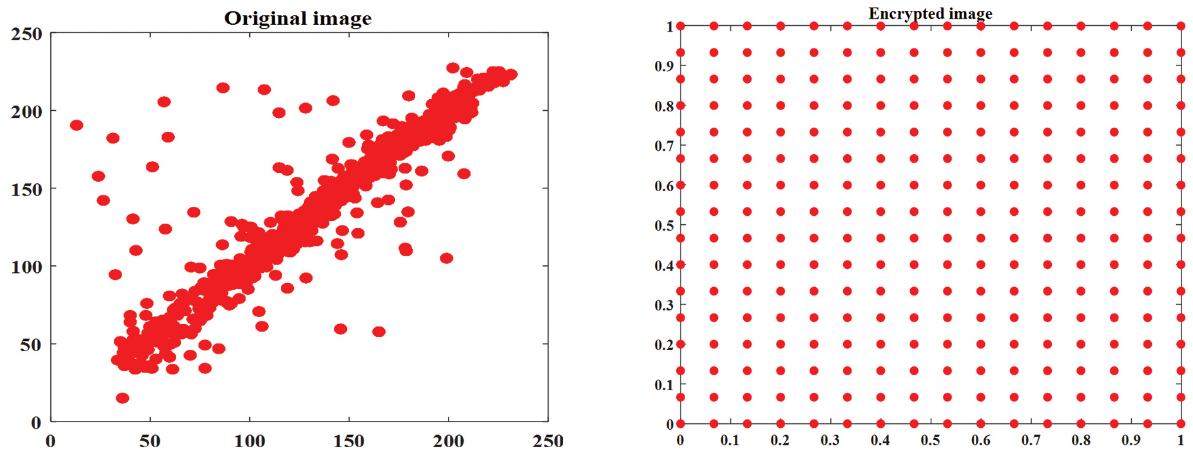
Another important test in statistical analysis is the correlation coefficient. The correlation coefficient indicates the degree of similarity between neighboring pixels in a particular image. In cryptography, the dependency between neighboring pixels must be destroyed; hence the CC near 0 is always desired in a cryptosystem. The outcome of the proposed technique for CC is shown in Figure 4. Using the following formula, the correlation coefficient can be determined:

$$CC = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \times \sum_{i=1}^{N}(y_i - \bar{y})^2}} \tag{25}$$

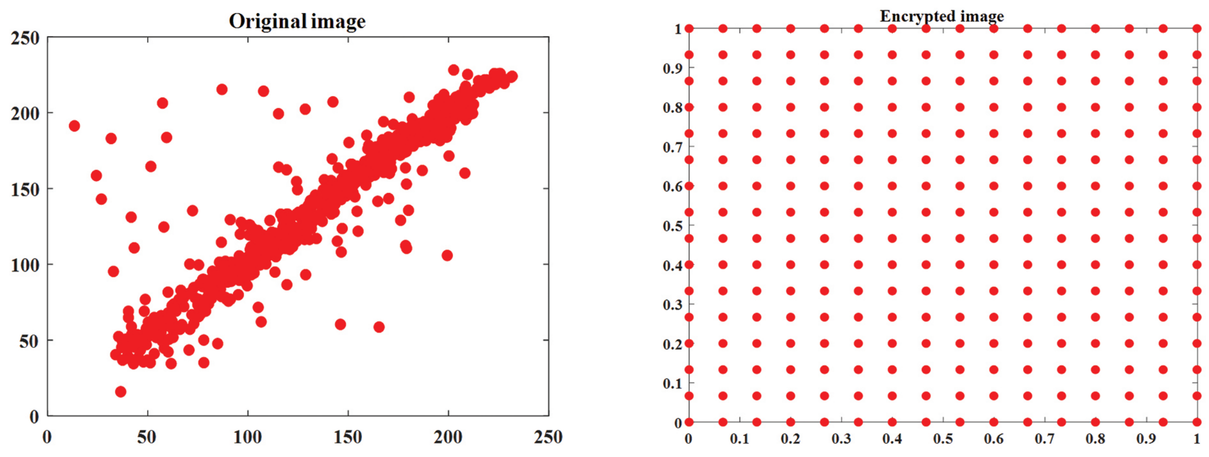
Where two adjacent pixels as x_i *and* y_i, *and the* number of pixels is *N*.



(a) CC of Lena



(b) CC of Baboon



(c) CC of Pepper

Fig. 4. (a) Correlation of Lena's image. (b) Correlation of Baboon image. (c) Correlation of Pepper image

4.2.4. Keyspace analysis

This value impacts the resilience of a cryptographic algorithm against brute-force attacks. It estimates the key sample space used to choose the encryption key. Therefore, to reduce brute force's feasibility, the sample area for the key should be made extremely spacious. The proposed method employs the SHA-512 function, which provides a key space of size 2^{512}, which seems to be strong enough to resist brute-force attacks.

4.2.5. Entropy test

Entropy (H) is the average quantity of information in each pixel, as per information theory as is expressed as:

$$H = \sum_{i=1}^{N} p(x_i) \times \log_2 \left(\frac{1}{p(x_i)} \right) \tag{26}$$

where the total number of pixels is N.

A perfect cryptosystem generates ciphers with equal probabilities, resulting in an entropy of eight pixels encoded on 8 bits (pixel values between 0 and 255). Table 3 lists the various entropy values for the evaluated test images. This table indicates that the entropy of the encrypted images is quite near to the ideal value, demonstrating the algorithm's resilience to entropy threats.

Table 3. Results of information entropy compared with state-of-art Methods

Test Image	Proposed using MATLAB	Proposed using FPGA	Ref. [33]	Ref. [34]	Ref. [35]	Ref. [36]
Lena	7.9991	7.9993	7.9891	7.9991	-	-
Pepper	7.9995	7.9998	7.9929	7.9987	-	-
Baboon	7.9990	7.9992	-	7.9990	7.9969	-
Yacht	7.9984	7.9986	-	-	7.9977	-
Boat	7.9987	7.9989	-	-	7.9979	-
Airplane	7.9962	7.9964	-	-	-	7.9980
Barbara	7.9950	7.9952	-	-	-	7.9969
Clock	7.9900	7.9902	-	-	-	7.9980

4.2.6. Strength against noise attack

In communication environments, cipher text images are frequently easily disrupted and destroyed by noise. Therefore, it must be able to ensure the resilience of encryption text while sending it. Thus, even if the cipher text is altered by interference, it is still possible to extract some relevant information.

To measure the effectiveness of our de-noising approach, the tetrolet transformation, we corrupt the grayscale image with varying quantities of salt and pepper noise. Using DCSR, GSR, wavelet, and NESTA, Table 4 displays the PSNR and SSIM quantitative evaluation results related to the proposed approach for various noise level values.

Table 4. Comparative results for image denoising with existing algorithms using FPGA

Methods	Noise Level	PSNR	SSIM
Proposed	20%	35.21	0.99
	50%	33.69	0.99
	60%	30.56	0.99
	70%	29.89	0.95
NESTA Algorithm	20%	18.88	0.39
	50%	16.71	0.38
	60%	16.33	0.37
	70%	16.03	0.36
Wavelet denoising	20%	31.68	0.89
	50%	28.61	0.86
	60%	26.69	0.79
	70%	24.84	0.74
GSR Algorithm	20%	26.70	0.80
	50%	24.66	0.78
	60%	23.61	0.74
	70%	22.40	0.69
DCSR Algorithm	20%	33.97	0.99
	50%	30.08	0.95
	60%	27.03	0.93
	70%	25.24	0.92

Due to the tetrolet transform's superior sparse representation for image geometrical structural properties and excellent power focusing capacity, it has a significant denoising capability. With these advantages, the performance of the proposed approach is superior to other existing techniques. Among all the techniques, the DCSR algorithm achieves similar results to the proposed approach when the noise level is lower (20%). However, when the noise level is increased, the performance of this approach is lower than the proposed approach. Figures 5 and 6 exhibit fluctuations in PSNR and SSIM output in response to varying noise levels, demonstrating the efficacy of our approach.

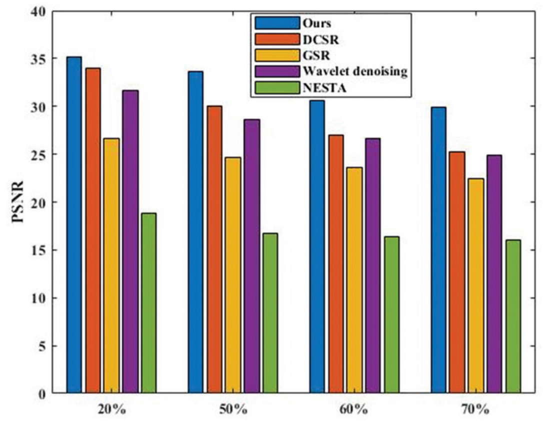


Fig.5. Analysis of PSNR values for various noise reduction techniques using FPGA

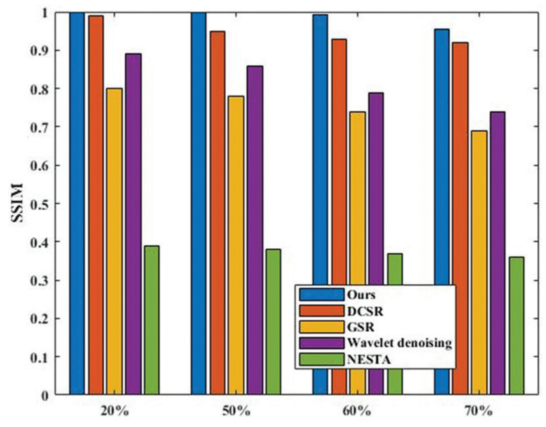


Fig. 6. Analysis of SSIM values for various noise reduction techniques using FPGA

These PSNR and SSIM results reveal that the demonstrated technique consistently receives the best scores, proving that the restoration result obtained by it is the best both practically and graphically.

Table 5. Comparative results for image denoising with existing algorithms using MATLAB

Methods	Noise Level	PSNR	SSIM
Proposed	20%	33.88	0.97
	50%	31.23	0.97
	60%	28.48	0.97
	70%	26.12	0.93
NESTA Algorithm	20%	16.54	0.36
	50%	14.98	0.35
	60%	12.51	0.35
	70%	14.93	0.34
Wavelet denoising	20%	28.43	0.86
	50%	26.12	0.85
	60%	24.51	0.76
	70%	22.90	0.71
GSR Algorithm	20%	24.89	0.78
	50%	23.71	0.75
	60%	20.12	0.72
	70%	18.39	0.65
DCSR Algorithm	20%	30.78	0.97
	50%	28.76	0.92
	60%	25.43	0.91
	70%	22.98	0.89

From Tables 4 and 5, it is observed that the performance of the proposed approach is enhanced using FPGA. The PSNR and SSIM values are quite decreased in MATLAB implementation. Finally, we concluded that the results acquired by FPGA are more clear and more accurate than the outcomes obtained by MATLAB.

4.2.7. Time complexity Analysis

Another crucial metric to examine when evaluating the efficiency of the encryption system is time complexity. The plain image is compressed and encrypted as part of the encryption process, and the encrypted image is decrypted and rebuilt as part of the decryption phase. Table 5 displays the time complexity taken for reconstruction, decryption, encryption, and compression.

Table 6. Encryption and decryption time (sec)

Encryption process		
Compression	Encryption	Total
0.84	0.18	1.02
Decryption process		
Decryption	Reconstruction	Total
0.19	29.08	29.27

5. CONCLUSION AND FUTURE WORKS

This research proposes a novel approach for encrypting compressed grayscale images utilizing optimized DNA encoding and 3D-chaotic logistics mapping. Proposed the Chevrolet algorithm to compress the grayscale image with excellent quality and low noise. Here, several steps in the encryption process are carried out using a 3D chaotic map. DNA encoding is done using complementary rules on the input and key images. The encoded input image and key image are now combined logically using the XOR method. Proposed a Tuna Swarm Optimization that might significantly improve the image's encryption speed and transmission security. This study tested the proposed method on eight images and analyzed the outcomes. Our approach is tested on salt-and-pepper-noise-corrupted images. Comparing Chevrolet transformation with cutting-edge PSNR and SSIM algorithms. Histogram analysis, key space analysis, entropy, correlation coefficient, and time complexity supported the proposed algorithm's performance. The proposed method outperforms state-of-the-art methods with 32.33dB PSNR, 0.98% SSIM, and 7.99721 information entropy, demonstrating it is more secure using FPGA implementation. In the future, we may design more complex algorithms using various basic quantum chaotic systems to improve the encryption process. We can do this by utilizing chaotic quantum systems, hyper-chaos, etc.

Conflict of Interests: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We declare that this manuscript is original, has not been published before, and is not currently being considered for publication elsewhere.

Funding: The authors received no specific funding for this study

Availability of Data and Material: Not applicable

Code Availability: Not applicable

Authors' Contributions: The authors confirm sole responsibility for the following: study conception and design, data collection, analysis and interpretation of results, and manuscript preparation.

Ethics Approval: This material is the authors' original work, which has not been previously published elsewhere. The paper reflects the authors' own research and analysis truthfully and completely.

6. REFERENCES

[1] W. Sirichotedumrong, H. Kiya, “Grayscale-based block scrambling image encryption using ycbcr color space for encryption-then-compression systems”, APSIPA Transactions on Signal and Information Processing, Vol. 8, No. 1, 2019, pp. 12-23.

[2] Y. Naseer, T. Shah, A. Javeed, “Advanced image encryption technique utilizing compression, dynamical system, and S-boxes”, Mathematics and

Computers in Simulation, Vol. 178, No. 2, 2020, pp. 207-217.

[3] J. S. Khan, S. K. Kayhan, "Chaos and compressive sensing-based novel image encryption scheme", Journal of Information Security and Applications, Vol. 58, No. 1, 2021, p. 102711.

[4] E. Setyaningsih, R. Wardoyo, A. K. Sari, "Securing color image transmission using a compression-encryption model with a dynamic key generator and efficient symmetric key distribution", Digital Communications and Networks, Vol. 6, No.4, 2020, pp. 486-503.

[5] M. Shi, S. Guo, X. Song, Y. Zhou, E. Wang, "Visual secure image encryption scheme based on compressed sensing and regional energy", Entropy, Vol. 23, No. 5, 2021, p. 570.

[7] X. Chai, J. Bi, Z. Gan, X. Liu, Y. Zhang, Y. Chen, "Color image compression and encryption scheme based on compressive sensing and double random encryption strategy", Signal Processing, Vol. 176, 2020, pp. 107-684.

[8] V. Upadhyaya, M. Salim, "Joint approach based quality assessment scheme for compressed and distorted images", Chaos, Solitons & Fractals, Vol. 160, No.1, 2022, pp. 112-278.

[9] V. Upadhyaya, M. Salim, "Compressive Sensing: An Efficient Approach for Image Compression and Recovery", Recent Trends in Communication and Intelligent Systems, Vol.1, No.1, 2020, pp. 25-34.

[10] M. Lahdir, H. Hamiche, S. Kassim, M. Tahanout, K. Kemih, S. A. Addouche, "A novel robust compression-encryption of images based on SPIHT coding and fractional-order discrete-time chaotic system", Optics & Laser Technology, Vol. 109, No.1, 2019, pp.534-546.

[11] Y. Xie, J. Yu, S. Guo, Q. Ding, E. Wang, "Image encryption scheme with compressed sensing based on the new three-dimensional chaotic system", Entropy, Vol. 21, No. 9, 2019, p. 819.

[12] Y. G. Yang, B. W. Guan, Y. H. Zhou, W. M. Shi, "Double image compression-encryption algorithm based on fractional order hyperchaotic system and DNA approach", Multimedia Tools and Applications, Vol. 80, No. 1, 2021, pp. 691-710.

[13] H. Hu, Y. Cao, J. Xu, C. Ma, H. Yan, "An image compression and encryption algorithm based on the fractional-order simplest chaotic circuit", IEEE Access, Vol. 9, No.1, 2021, pp. 22141-22155.

[14] K. Wang, X. Wu, T. Gao, "Double color image compression–encryption via compressive sensing", Neural Computing and Applications, Vol. 33, No. 19, 2021, pp. 12755-12776.

[15] W. Huang, D. Jiang, Y. An, L. Liu, X. Wang, "A novel double-image encryption algorithm based on Rossler hyperchaotic system and compressive sensing", IEEE Access, Vol. 9, No.1, 2021, pp. 41704-41716.

[16] A. Sahasrabuddhe, D. S. Laiphrakpam, "Multiple image encryption based on 3D scrambling and hyper-chaotic system", Information Sciences, Vol. 550, No. 1, 2021, pp. 252-267.

[17] J. Mou, F. Yang, R. Chu, Y. Cao, "Image compression and encryption algorithm based on the hyperchaotic map", Mobile Networks and Applications, Vol. 1, No. 1, 2019, pp. 1-13.

[18] B. K. Shukur, S. Hadi, W. Al-Hameed, "An Efficient Approach Implementation to Image Compression and Encryption", Eurasian Journal of Engineering and Technology, Vol. 8, No. 1, 2022, pp. 1-13.

[19] L. Gong, K. Qiu, C. Deng, N. Zhou, "Image compression and encryption algorithm based on a chaotic system and compressive sensing", Optics & Laser Technology, Vol. 115, No. 1, 2019, pp. 257-267.

[20] J. Karmakar, D. Nandi, M. K. Mandal, "A novel hyperchaotic image encryption with sparse-representation-based compression", Multimedia Tools and Applications, Vol. 79, No. 37, 2020, pp. 28277-28300.

[21] H. Dong, E. Bai, X. Q. Jiang, Y. Wu, "Color image compression-encryption using the fractional-order hyperchaotic system and DNA coding", IEEE Access, Vol. 8, No.1, 2020, pp. 163524-163540.

[22] G. U. Bhargava, S. V. Gangadharan, "FPGA implementation of modified recursive box filter-based fast bilateral filter for image denoising", Circuits, Systems, and Signal Processing, Vol. 40, No. 3, 2021, pp. 1438-1457.

[23] M. Maazouz, A. Toubal, B. Bengherbia, O. Houhou, N. Batel, "FPGA implementation of a chaos-based

image encryption algorithm", Journal of King Saud University-Computer and Information Sciences, Vol. 1, No. 1, 2022, pp. 23-45.

[24] T. M. Kumar, K. S. Reddy, S. Rinaldi, B. D. Parameshachari, K. Arunachalam, "A low area high-speed FPGA implementation of AES architecture for cryptography application. Electronics, Vol. 10, No. 16, 2021, pp. 20-23.

[25] H. Zodpe, A. Sapkal, "An efficient AES implementation using FPGA with enhanced security features", Journal of King Saud University-Engineering Sciences, Vol. 32, No. 2, 2020, pp. 115-122.

[26] A. Hafsa, M. Gafsi, J. Malek, M. Machhout, "FPGA implementation of improved security approach for medical image encryption and decryption", Scientific Programming, Vol. 3, No. 1, 2021, pp. 34-56.

[27] T. M. Kumar, K. R. Balmuri, A. Marchewka, P. B. Divakarachari, S. Konda, "Implementation of Speed-Efficient Key-Scheduling Process of AES for Secure Storage and Transmission of Data", Sensors, Vol. 21, No. 24, 2021, p. 8347.

[28] G. U. Bhargava, S. V. Gangadharan, "FPGA implementation of modified recursive box filter-based fast bilateral filter for image denoising", Circuits, Systems, and Signal Processing, Vol. 40, No. 3, 2021, pp. 1438-1457.

[29] J. Krommweh, "Tetrolet transform: A new adaptive Haar wavelet algorithm for sparse image representation", Journal of Visual Communication and Image Representation, Vol. 21, No. 4, 2010, pp. 364-374.

[30] M. McLoone, J. V. McCanny, "Efficient single-chip implementation of SHA-384 and SHA-512", Proceedings of the IEEE International Conference on Field-Programmable Technology, Hong Kong, China, 16-18 December 2002, pp. 311-314.

[31] L. Kocarev, Z. Tasev, "Public-key encryption based on Chebyshev maps", Proceedings of the International Symposium on Circuits and Systems, Bangkok, Thailand, 25-28 May 2003.

[32] A. ShokouhSaljoughi, H. Mirvaziri, "A new method for image encryption by 3D chaotic map", Pattern Analysis and Applications, Vol. 22, No. 1, 2019, pp. 243-257.

[33] M. Tan, Y. Li, D. Ding, R. Zhou, C. Huang, "An Improved JADE Hybridizing with Tuna Swarm Optimization for Numerical Optimization Problems", Mathematical Problems in Engineering, Vol. 1, No. 1, 2022.

[34] Y. Naseer, T. Shah, A. Javeed, "Advanced image encryption technique utilizing compression, dynamical system, and S-boxes", Mathematics and Computers in Simulation, Vol. 178, No.1, 2020, pp. 207-217.

[35] W. Alexan, M. ElBeltagy A. Aboshousha, "Rgb image encryption through cellular automata, s-box, and the Lorenz system", Symmetry, Vol. 14, No. 3, 2022, p. 443.

[36] J. Hao, H. Li, H. Yan, J. Mou, "A New Fractional Chaotic System and its application in image encryption with DNA mutation", IEEE Access, Vol. 9, No.1, 2021, pp. 52364-52377.

[37] R. Guesmi, M. A. Farah, "A new efficient medical image cipher based on a hybrid chaotic map and DNA code", Multimedia tools and applications, Vol. 80, No. 2, 2021, pp. 1925-1944.

Appendix-1

Acronyms	Abbreviations
FPGA	Field-Programmable Gate Array
DFT	Discrete Fourier Transform
DCT	Discrete Cosine Transform
DWT	Discrete Wavelet Transform
FWPT	Full Wavelet Packet Transform
XSG	Xilinx System Generator
HRRBF	Hybrid Recursive Reversible Box Filter
FABF	Fast Adaptive Bilateral Filter
RLG	Reversible Logic Gates
AES	Advanced Encryption Standard
LUT	Look Up Table
MPPRM	Modified Positive Polarity Reed-Muller
PRNG	Pseudorandom Number Generator
MAES	Modified Advanced Encryption Standard
FJMKE	Fork-Join Model Of Key Expansion
MRBF	Modified Recursive Box Filter
FBF	Fast Box Filter
TSO	Tuna Swarm optimization