IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 July 2023, accepted 30 July 2023, date of publication 2 August 2023, date of current version 15 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3301460

RESEARCH ARTICLE

Image Encryption via Base-*n* PRNGs and Parallel Base-*n* S-Boxes

MOHAMED GABR[1], **(Member, IEEE), RIMON ELIAS**[2], **(Senior Member, IEEE), KHALID M. HOSNY**[3], **(Senior Member, IEEE), GEORGE A. PAPAKOSTAS**[4], **AND WASSIM ALEXAN**[5,6], **(Senior Member, IEEE)**

[1]Computer Science Department, Faculty of Media Engineering and Technology, German University in Cairo, Cairo 11835, Egypt
[2]Digital Media Department, Faculty of Media Engineering and Technology, German University in Cairo, Cairo 11835, Egypt
[3]Department of Information Technology, Zagazig University, Zagazig 44519, Egypt
[4]MLV Research Group, Department of Computer Science, International Hellenic University, 65404 Kavala, Greece
[5]Communications Department, Faculty of Information Engineering and Technology, German University in Cairo, Cairo 11835, Egypt
[6]Mathematics Department, German International University (GIU), Cairo 13507, Egypt

Corresponding author: George A. Papakostas (gpapak@cs.ihu.gr)

This work was supported by the Hellenic Academic Libraries (HEAL)-Link in OA mode.

ABSTRACT The fast-paced advancement in multimedia production and exchanges over unsecured networks have led to a dire need to develop security applications. In this regard, chaos theory has much to offer, especially high-dimensional (HD) chaotic functions of fractional order. The authors propose a new symmetric, secure and robust image encryption method in this research work. In this method, the authors hybridize the Chen and Chua chaotic functions with a Memristor circuit to benefit from the strengths of each. Such a hybridization of systems allows for the generation of pseudo-random numbers which are used to develop encryption keys and substitution boxes (S-boxes). For the application of the generated encryption keys towards carrying out data diffusion, instead of the most commonly used approach of bit-stream level *XOR*, this work utilizes different logical and arithmetic operations, which is made possible by performing this process over variable numerical bases. Moreover, multiple S-boxes of varying base-*n* are generated and utilized in a parallel fashion, carrying out data confusion. The computed numerical results reflect the superior capabilities of the proposed image encryption technique, signifying resilience and robustness against various attacks.

INDEX TERMS Base-*n* key, base-*n* S-box, parallel S-boxes, chaos theory, Chen, Chua, fractional-order, hyperchaotic map, memristor, image cryptosystem, image encryption.

I. INTRODUCTION

Modern wireless communication networks and big data applications have made security issues crucial [49]. Hence, research and development efforts into putting in place data security measures like cryptography [7], [12], [14], steganography [6], [15], watermarking, as well as their combined use [3], [56] has become a very hot topic in the last decade. Well-established cryptosystems protected private and sensitive data for years. DES, Triple DES, and AES were popular and dependable cryptographic algorithms. However, it became clear over time that not all cryptosystems are suitable for multimedia like 2D and 3D images and videos [2], [62]. This is because such multimedia inherently have sheer redundancy and high cross-correlation among their pixels. This, in part, explains the exponential increase in literature in recent years in relation to image steganography and cryptography. On examining the literature on image encryption algorithms, one repeatedly encounters the utilization of various mathematical operations and constructs originating from chaos theory [4], [7], [14], [27], [35], [58], cellular automata [2], [5], [19], DNA encoding [20], [46], [51], [54], electric circuits [11], [40], and elliptic curves [9], [22], [52].

Chaos theory, in particular, has been widely investigated and applied in relation to image encryption. This is because of the array of characteristics inherent in dynamical functions

The associate editor coordinating the review of this manuscript and approving it for publication was Yassine Maleh.

of chaotic behavior. Those include ergodicity, sensitivity to initial values, pseudo-randomness, and periodicity [24]. Such functions are generally classified into low-dimensional (LD) functions and high-dimensional (HD) ones, with each class displaying one or more desirable qualities [14]. While LD chaotic functions result in simpler software and hardware implementations, their utilization in image encryption does not provide sufficient cryptographic strength [25]. On the other hand, HD chaotic functions while being more complex, requiring more computing resources and circuitry, are effectively capable of providing very high security. On examining hyperchaotic functions, one easily realizes the large number of control parameters involved [37]. This means that their utilization in image encryption algorithms directly translates into a much larger key space, effectively mitigating any chances for the success of brute-force attacks [20]. Attempting to solve hyperchaotic systems at a fractional-order allows for a further increase in the number of control variables, and subsequently, a further increase in the key space. Furthermore, it is possible to reach optimal or near-optimal periods of PRNGs generated from hyperchaotic systems through careful design choices of the systems, combinations of maps and systems, initialization, and post-processing of the generated raw chaotic sequence [48], [59].

Nevertheless, it is important to note that utilization of hyperchaotic functions in image encryption algorithms could also have its disadvantages. For example, hyperchaotic functions are often non-linear, which can make them difficult to analyze. This makes it difficult to prove that chaotic functions are secure against cryptanalytic attacks. Additionally, chaotic functions can be computationally expensive to evaluate, which can make them impractical for use in real-time applications. Finally, chaotic functions can be sensitive to initial conditions, which can make it difficult to ensure that the encrypted image is consistent with the original image. However, this work mitigates the problem of computational complexity through the employment of state-of-the-art parallel processing techniques in the software implementation of the proposed image cryptosystem [57].

The image processing community has recently found an interest in fractional-order dynamical systems that exhibit chaotic behaviors [50]. More specifically, their applications in image encryption have been gaining momentum due to the better performance they offer, in comparison to their integer-order counterparts [7], [10], [26], [28], [41], [44]. In [7], numerical solutions are obtained for the fractional-order hyperchaotic 4D Chen system, and are jointly employed with those of the sine chaotic map to generate PRNGs. Next, hybrid DNA coding is utilized to implement an efficient image encryption algorithm. The authors of [7] carry out an extensive security analysis over a large array of images, providing a very solid work. The authors of [10] suggest the use of a fractional-order logistic map in their proposed image cryptosystem and compare its use with the classical logistic map. Various analyses are carried out to gauge the performance of their proposed cryptosystem, however, no execution time is provided. An image encryption technique with a large key space is proposed in [26], where a 4D Chen hyperchaotic map of fractional-order is employed in conjunction with a Fibonacci Q-matrix. While the proposed fractional-order hyperchaotic system is shown to perform well in terms of image encryption, the authors of [26] also do not provide an execution time analysis. A lightweight image encryption technique is proposed in [28], where switching between 3 different fractional-order systems takes place. While the algorithm proposed in [28] is shown to be very efficient, providing an encryption rate of 8.32 Mbps, however, very limited security analysis is provided otherwise. This makes it hard to gauge its performance. The authors of [41] propose an image encryption algorithm that makes use of the solutions of chaotic fractional-order fuzzy cellular neural networks (FOFCNN). Analysis of chaoticity of the proposed FOFCNNs are well presented and discussed, promising excellent application to image encryption. However, apart from a key space of 10^{300}, a pixel cross-correlation analysis and an entropy analysis, no further security or efficiency metrics are reported in [41]. In [44], a scheme utilizing smoothed sliding modes state observers for chaotic systems of fractional-order is proposed as a secure image and text encryption design. The authors provide an array of analyses, showcasing the efficiency of their design, its security, and resistivity to known and chosen plaintext attacks. The common denominator in many image cryptosystems that are based on the solutions of fractional-order systems is their use of PRNGs. Such PRNGs are generated by making use of either the modulus operation or the comparison of decimal values against a preset threshold to generate the required encryption key (which takes the form of a bit-stream).

Design and implementation of pseudo-random number generators (PRNG) are at the core of research efforts in the field of cryptography. This is because both key generation and S-box design benefit from a randomly-distributed bit-stream [5]. Various instances in the literature showcase the use of a PRNG in image encryption algorithms. For example, the authors of [17] make use of the Lucas sequence to generate an S-box for their proposed 3-stage image encryption algorithm. While the proposed algorithm in [17] is shown to be robust, it is not efficient in comparison with the state-of-the-art. In [13], the authors generate encryption keys by utilizing the Rössler system and the Recamán's sequence, in another low-efficiency algorithm. Similarly, the authors of [30] make use of the Fibonacci sequence, a chaotic tan function, as well as a Bessel function, to generate PRNGs as encryption keys. The authors of [1] research elliptic curves and use them to design a PRNG, then utilize it in conjunction with the Arnold map to carry out image encryption. The generated PRNG is shown to pass all NIST tests, as well as other statistical and differential tests, and thus its use in image encryption provides good security. An FPGA implementation of a PRNG is proposed in [60], where a memristive Hopfield neural network (MHNN) with a special activation gradient is utilized. The proposed MHNN is computer simulated and dynamically analyzed before its actual implementation on an FPGA. Based on the proposed MHNN, a PRNG is generated and used as an encryption key. The Mersenne Twister is made use of in [18], where the authors employ it as one of the encryption keys in an efficient multi-stage algorithm. The proposed algorithm in [18] is shown to be highly resistive to various types of attacks. The authors of [31] utilize a discretized version of the chaotic sine map to design an S-box, as well as the hyperchaotic Lü system as a PRNG, in a multi-stage image encryption algorithm. Their proposed substitution-permutation network operates in the cipher block chaining mode, and is shown to provide excellent security performance. This is because the adopted Lü system is highly non-linear and generates discrete values with lengthy orbits.

The previous paragraphs carried out a brief literature review on the need for image encryption, the importance of chaos theory to this field, the use of PRNGs in the design of encryption keys and S-boxes, as well as the recent trend of utilizing the solutions of fractional-order dynamical functions to realize better performing security measures. In this research work, a multi-faceted research gap is identified in the literature and an attempt is made to fill it, as follows. First, when it comes to key application, it is usually performed over the bit-stream level. As a commonly employed reversible operation at this level, *XOR* has been the default operation in use [2], [5], [7], [19], [20], [60], which made it predictable by adversaries. This may eventually lead to some levels of predictability. Hence, the attempted solution in this work is to apply the key on different levels (decimal, among others, for example) to expand the range of available applicable operations, which can be later performed in an alternating manner. The second solution this work attempted at is redefining the notion of the S-box on 2 levels. Instead of only considering 8-bit S-boxes (where every element $\in [0, 255]$), more ranges are experimented on. Moreover, multiple S-boxes are applied in a parallel fashion as well. Third, this work identified multiple recent works on image encryption that do not provide any information related to execution time, rendering it impossible to gauge their propositions suitability for real-time applications. In this work, execution time is provided and shown to outperform the state-of-the-art, rendering it suitable for real-time applications.

This research work proposes the following:

1) A multi-stage symmetric image encryption technique is proposed. In the first and final stages, encryption keys are generated based on the solution of different hyperchaotic functions of fractional-order, which are applied over different bases with multiple operations. The center encryption stage utilizes the solutions of yet another hyperchaotic function such that parallel S-boxes of different dimensions and bases are applied in a parallel fashion.
2) The proposed design is shown to satisfy Shannon's ideas of confusion and diffusion [8], and is thus highly secure.
3) A large array of testing metrics are computed and their values compared against the ideal benchmarks, showcasing not only the security but also the robustness capabilities of the proposed design against visual, statistical, entropy and known plain text attacks.
4) Since the center stage incorporates the use of multiple S-boxes that are designed based on the solutions of a hyperchaotic function, this allows for the theoretical expansion of the key space to infinity, resisting any possible brute force attacks.
5) The proposed design is shown to be ultra-efficient, encrypting images at a rate of 4.01 Mbps.
6) Output encrypted images' data under the proposed design are shown to successfully pass all the NIST-800 randomness tests.

The remainder of this research work is organized as follows. Section II provides a discussion on chaotic fractional-order differential systems as pseudo-random bit-stream generators. Section III discusses the application of encryption keys at different levels. Section IV discusses seed-based, base-*n*, parallel S-box generation and application. Section V outlines the proposed color image encryption technique. Section VI reports the computed visual and numerical results, as well as conducts a comparative analysis with counterpart image encryption algorithms from the literature. Finally, Section VII draws the conclusions and suggests a couple of future research works.

II. CHAOTIC FRACTIONAL DIFFERENTIAL SYSTEMS AS PSEUDO-RANDOM BIT-STREAM GENERATORS

Fraction calculus, as a mathematical phenomenon, has been well established since 1695 [23]. However, such a phenomenon has not been involved in many applications until recent years. Accordingly, chaotic fractional-order differential systems, in recent years, are starting to be utilized as means for PRNG bit-streams generation techniques. Beside randomness being fulfilled by the chaotic behaviour of the system, another property which makes fractional differential systems especially more fitting for image encryption is the existence of many control coefficients, which in turns contribute to enhancing the key space of the overall encryption technique. In this work, 3 systems are used, namely, Chen system (Section II-A), Chua system (Section II-B), and Memristor system (Section II-C). Alongside these section, a discussion about the method applied to convert the numerical sequences produced by these systems into bit-streams is presented in Section II-D.

A. CHEN SYSTEM

The fractional order 4D Chen system [23], [55] is a hyperchaotic fractional differential system. Hyperchaotic functions are a logical mathematical evolution towards better production of PRNG sequences. This is clear because hyperchaotic systems produces more than one positive Lyapunov exponent, which reemphasises on the sudo-randomness of numerical sequences generated by them (which are later turned into bit-streams). Alongside randomness, as a 4D system, many control variables are involved in the Chen system, which as mentioned presents it as a good candidate for image encryption techniques.

The Chen system is modeled using the following equations:

$$D^{\alpha_1}x = a(y-x)+u, \tag{1}$$
$$D^{\alpha_2}y = \gamma x - xz + cy, \tag{2}$$
$$D^{\alpha_3}z = xy - bz, \tag{3}$$

and

$$D^{\alpha_4}u = yz + du. \tag{4}$$

In (1), (2), (3), and (4), 3 groups of control variables are utilized to control the system, and change the produces sequence in accordance. The first group, the initial values for x, y, z, and u, (or x_0, y_0, z_0, and u_0), are the representation of the initial point in the 4D space from which the rest of the system is emitted. The second group, a, b, c, γ, and d, are the scale coefficients for the 4 equations. Finally, $\alpha 1$, $\alpha 2$, $\alpha 3$, and $\alpha 4$ are the fractional differential orders. All 3 groups combined introduce a total of 13 variables.

For demonstration, Fig. 1 shows an example plot for the fractional-order 4D Chen system. In Fig. 1, the system is calculated in the 4D space, hence initial values are needed for the four axes. However, for plotting purposes, one axis is ignored in each illustration. Further analysis of the system's hyperchaotic behavior can be carried out through examining its bifurcation plots against various parameters, as illustrated in Fig. 2 and Fig. 3, for b and c, respectively. Moreover, the 4 Lyapunov characteristic exponents (LCEs), which give the rate of exponential divergence from perturbed initial conditions, are plotted in Fig. 4.

B. CHUA SYSTEM

Another well-known fractional-order differential system which exhibits a chaotic behaviour is the Chua system [38]. As a chaotic system, its PRNG behaviour, alongside sensitivity to control parameters, makes the Chua system a good candidate for PRNG seed-based bit-stream generation. The Chua, as a 3D system, is equated as follows:

$$D^{\alpha_1}x = p(y-x-f(a,b,x)), \tag{5}$$
$$D^{\alpha_2}y = x-y+z, \tag{6}$$
$$D^{\alpha_3}z = -qy, \tag{7}$$

given that,

$$f(a,b,x) = bx + \frac{1}{2}(a-b)(|x+1|-|x-1|) \mid a < b < 0. \tag{8}$$

As presented in (5), (6), (7), and (8), there is a total of 10 control variables, which can be divided into 3 groups. First group contains the initial values for x,y, and z. The second group consists of scale factors, p and q for the main axis equations ((5), (6), and (7)), and a and b for (8). The third group, α_1, α_2, and α_3 are the fractional differential orders.

As an illustration, Fig. 5 displays an example plot for the fractional-order 3D Chua system. The figure shows the solution of the regular system, however, in application, the system is further solved in fractional-order. Further analysis of the system's hyperchaotic behavior can be carried out through examining its bifurcation plots against various parameters, as illustrated in Fig. 6 and Fig. 7, for b and c, respectively. Moreover, the 4 Lyapunov characteristic exponents (LCEs), which give the rate of exponential divergence from perturbed initial conditions, are plotted in Fig. 8.

C. MEMRISTOR SYSTEM

As one of the most recent fractional-order differential systems, a Memristor system remodeling we presented in [43]. Maintaining the same previously mentioned advantages in Chen and Chua systems, the Memristor system proposed exhibited a chaotic behaviour which is controllable with many tenability variables. The Memristor system is equated as:

$$D^{\alpha_1}x = ax + b(y-x)\times u^2, \tag{9}$$
$$D^{\alpha_2}y = -z - cy - d(y-x)\times u^2, \tag{10}$$
$$D^{\alpha_3}z = y, \tag{11}$$

and

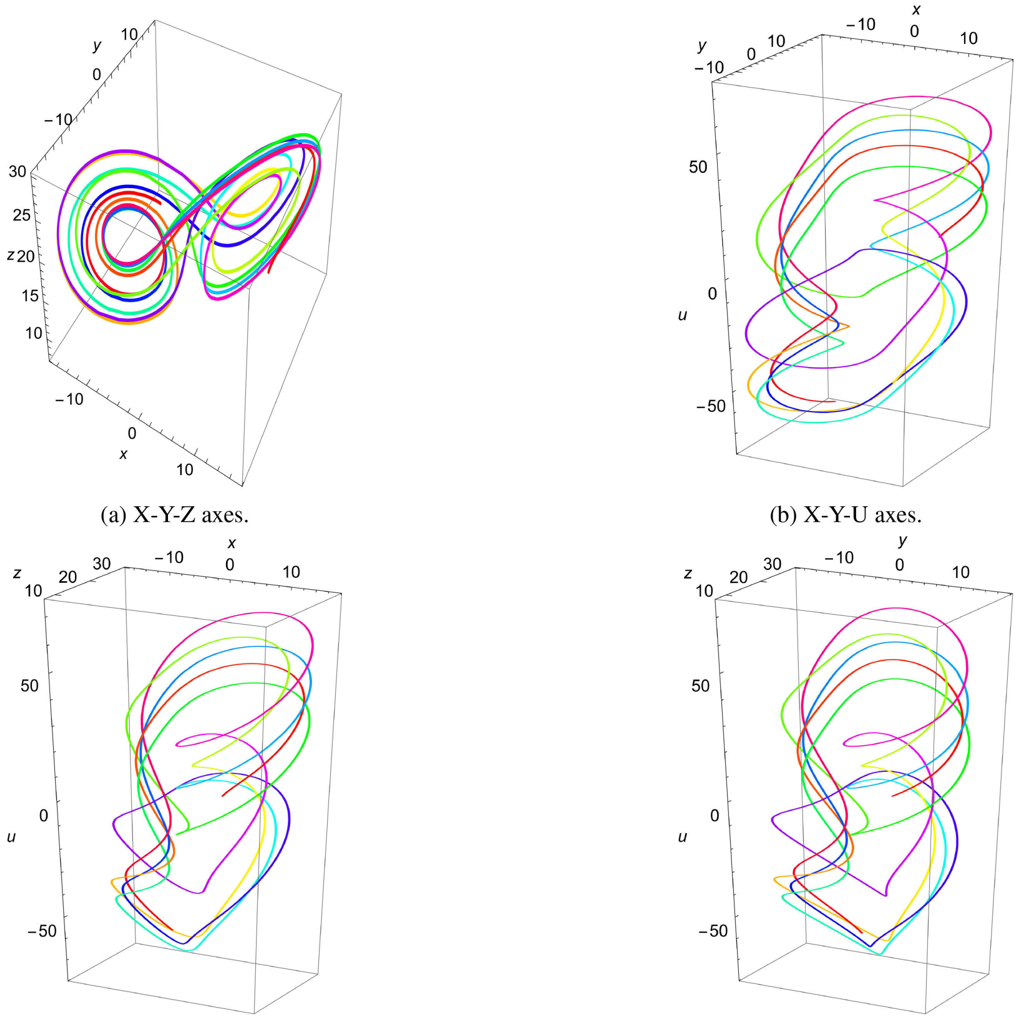
$$D^{\alpha_4}u = eu + f(y-x) - gu(y-x). \tag{12}$$

As the previously discussed systems, 3 sets of control variables are present in (9), (10), (11), and (12). The first set would contain the initial values for the axis x, y, z, and u. The second set contains 7 scaling factors which are: a, b, c, d, e, f, and g. The last set, α_1, α_2, α_3, and α_4, are the fractional differential orders. The total number of control variables is 15.

As an illustration, Fig. 9 displays an example plot for the fractional-order 4D Memristor system. As mentioned in the cases of Chen and Chua, the figure shows the solution of the regular system. However, in application, the system is further solved in fractional-order. Further analysis of the system's hyperchaotic behavior can be carried out through examining its bifurcation plots against various parameters, as illustrated in Fig. 10 and Fig. 11, for b and c, respectively. Moreover, the 4 Lyapunov characteristic exponents (LCEs), which give the rate of exponential divergence from perturbed initial conditions, are plotted in Fig. 12.

D. SYSTEMS SOLUTIONS TO BIT-STREAMS

Given a set of a sufficient number of control variables, one of the systems presented in Sections II-A, II-B, or II-C can be solved. The result of such a process, for a kD system, can be modeled as k sequences of real numbers ($k = 4$ for Chen and



(a) X-Y-Z axes.

(b) X-Y-U axes.

(c) X-Z-U axes.

(d) Y-Z-U axes.

FIGURE 1. 3D plots for the fractional order 4D Chen system over different combinations of axes where $\{x, y, z, u\} = 0.3$, $a = 35$, $b = 3$, $c = 12$, $\gamma = 28$, $d = 0.5$, and $\alpha = 0.97$ (the system is calculated in the 4D space, hence initial values are needed for the four axes. However, for plotting purposes, one axis is ignored in each illustration).

Memristor for example). Generically, 2 main processes are needed to transform a kD set of real numbers sequences into a 1D bit-stream. These 2 processes are a flattening process (to result in a 1D sequence), and a Boolean checking process (to convert real numbers into 0's and 1's). The flattening of the k sequences into a single sequence is performed using the following relation:

$$\begin{gathered}\{\{D_1^1, D_2^1, \ldots\}, \ldots, \{D_1^k, D_2^k, \ldots\}\} \\ \downarrow \\ \{D_1^1, \ldots, D_1^k, D_2^1, \ldots, D_2^k, \ldots\}.\end{gathered} \tag{13}$$

Given the flattened 1D sequence, and a threshold value λ, a bit-stream can be produced using the relation:

$$\begin{gathered}\{D_1^1, \ldots, D_1^k, D_2^1, \ldots, D_2^k, \ldots\} \\ \downarrow \\ \{C(D_1^1), \ldots, C(D_1^k), C(D_2^1), \ldots, C(D_2^k), \ldots\},\end{gathered} \tag{14}$$

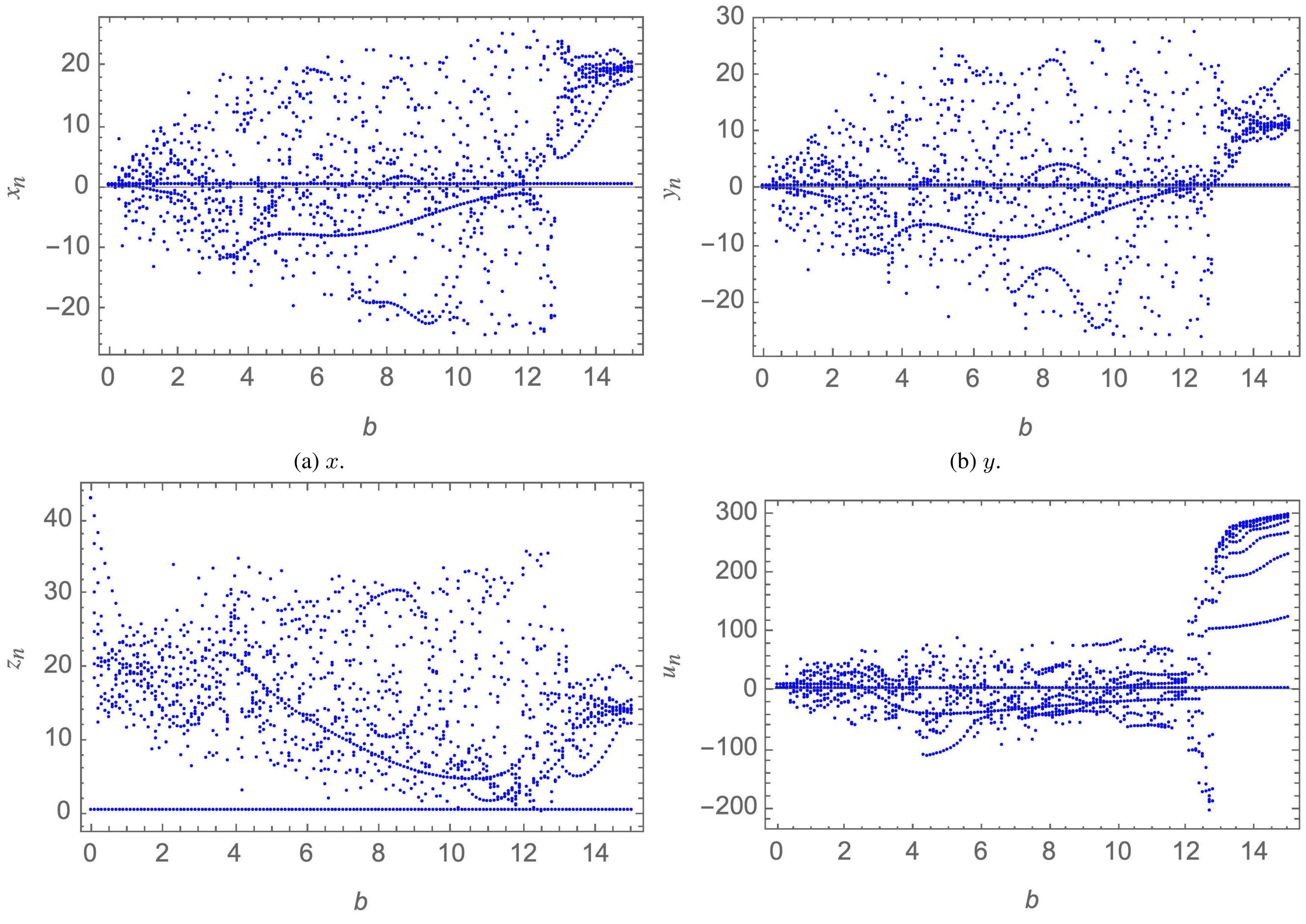
such that,

$$C(n) = \begin{cases} 1, & n > \lambda \\ 0, & otherwise, \end{cases} \tag{15}$$

where λ is a threshold that would result in a balanced bit-stream (as aforementioned, having an equal numbers of 0's and 1's). The value of λ would be the median of the 1D sequence of real numbers, as the median value presents the exact middle value of a sequence (unlike the mean for example).

III. ENCRYPTION KEY APPLICATION

Applying an encryption key is a default stage in any encryption algorithm. In image encryption, in particular, it is most common to utilize a key in 2 out of 3 stages of encryption [5], [13], [18], [19], [20]. This is due to the fact that such a stage performs the task of diffusion of an external key into the data to be encrypted. In the most basic form, applying an



(a) x.

(b) y.

(c) z.

(d) u.

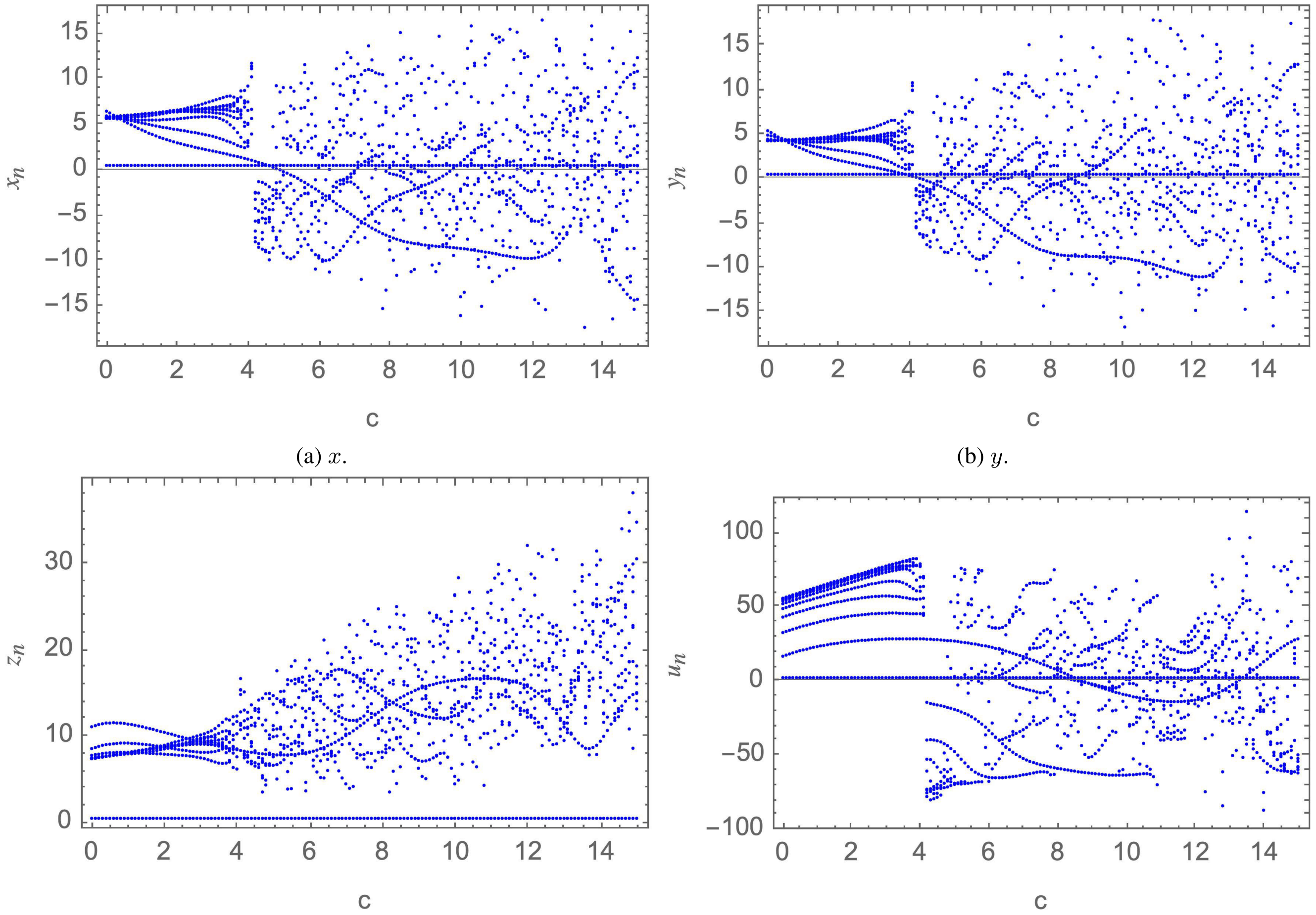
FIGURE 2. Bifurcation plots of the fractional order 4D Chen system for *x*, *y*, *z* and *u* against *b*.

encryption key is the operation which is used first to fuse both the data and the key into one encrypted result. Moreover, it is demanded to be possible to perform other operations on the encrypted result and the key resulting in reproducing the original input data, defining what is commonly known as reversible processes. Hence, a reversible process can be looked at, for the sake of this argument, as a mathematical/logical function. Evaluating a function, 2 aspects are analyzed, namely, the domain (containing both input data and key for encryption, or encrypted data alongside key for decryption), and the range (presenting the result of encryption or decryption respectively). In the case of image encryption or decryption, it is expected that both are of the same numerical base (bits for example). In other words, naturally, both the key and the input are operated on in the same base for the same size of data. Accordingly, the number of possible operations applicable (especially reversible ones) is limited by the the numerical base of operations, as further elaborated. Hence, in Sections III-A, III-B, and III-C, the numerical base, alongside examples of applicable reversible operations performed in that base are further discussed in Section III-D.

A. BIT-LEVEL OPERATION

On bit-level (representing data in binary space as sequences of 0's and 1's), only a few operations (logical gates) can operate in a reversible manner, for example, the *XOR*, *NOT* and the *CCNOT* (also known as the Toffoli gate [16]). The main cause of the reversibility of both these operations is the lack of collision of output with respect to the input. For example, considering the *AND* logic gate as a non-reversible operation, for a second input bit of 0 (as the second input position is usually reserved for the key bit), regardless of the first bit values (the image data bit), the output bit would be a 0, indicating the loss of the image data bit. Therefore, only the *XOR* and the *NOT* operations are considered when there is a need for reversibility, as in the case of image encryption. Moreover, alongside being reversible, as the operation presents an interaction between 2 bits (a data bit and a key bit), the *NOT* logic gate can only be utilized in junction with the *XOR* gate, presenting the *XNOR* gate.

According to the discussion provided above, in the binary numerical base (bit-level), only the *XOR*, and the *XNOR* as the negated variance, are utilizable in the image encryption



(a) x.

(b) y.

(c) z.

(d) u.

FIGURE 3. Bifurcation plots of the fractional order 4D Chen system for *x*, *y*, *z* and *u* against *c*.

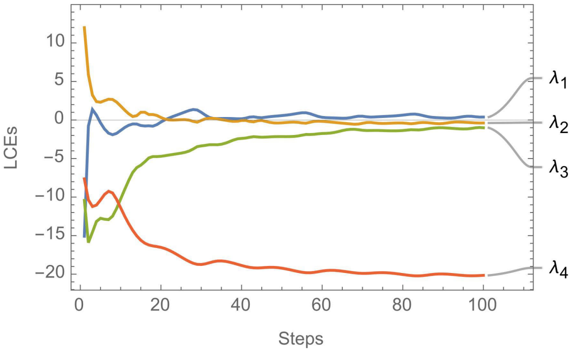


FIGURE 4. A plot of the 4 Lyapunov characteristic exponents of the fractional order 4D Chen system.

applications as they are binary and reversible. Such lack of diversity in applications resulted in key application on the bit-level to be subject to vulnerability issues. Assessing the main cause of such shortcoming, the core cause is rooted at the nature of the numerical base in use, which is the bit-level. As mentioned above, for operations being evaluated as mathematical functions, the domain and the range are the main aspects upon which evaluations are based. Being limited to the binary space, alongside respecting the constraints of being binary and reversible, results in making the possible number of ranges of functions to be exactly 2. In accordance to this, DNA key application (Section III-B), attracted much research interest as it remodeled the key embedding process by increasing the possible ranges space, as discussed next.

B. DNA-LEVEL OPERATION

DNA encoding, as a method utilized in key application in image cryptography, has proved foundational usefulness over the past years [20], [33], [42], [54], [61]. As a variant application, it added a layer of confusion to data diffusion as a result of introducing other key application (DNA-level) operations than the classical (bit-level) *XOR*. In other words, besides the availability of the previously mentioned operations (*XOR* and *XNOR*) which are applicable on the bit-level (base-2), other (binary and reversible) operations were introduced as a result of scaling up the presentation numerical domain to DNA-level (base-4). Towards further elaboration of such an enhancement, the DNA key application is discussed in steps,

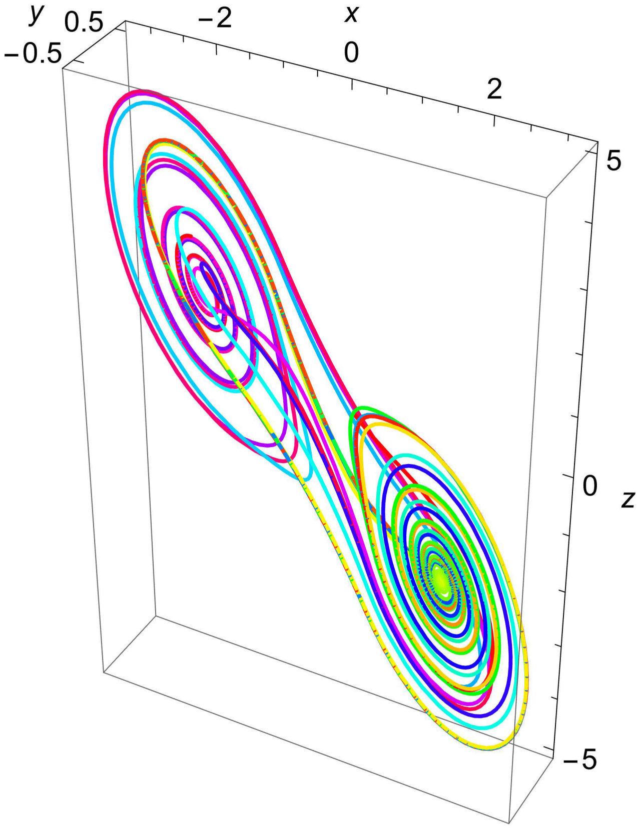


FIGURE 5. 3D plot for the fractional order Chua system, shown here for $a = -1.27$, $b = -0.68$, $p = 10$ and $q = 14.87$.

modeling data as DNA proteins (A, T, C, and G), alongside their numerical base-4 counterparts (0, 1, 2 and 3).

DNA key embedding is performed over 3 steps:

1) Transform image data and Key data into DNA sequences.
2) Perform a DNA-level operation on transformed data.
3) Transform data back to the original numerical space (as a reverse of the first step).

Starting with the first step, transformation of data into DNA is performed as a substitution relation between pairs of bits and DNA protein. Building such relation, variations of translations get presented. Numerically speaking, there are 4 permutations of pairs of bits, which are 00, 01, 10, and 11, and 4 possible substitutions of proteins, which are A, T, C, and G. Taking these possible values in consideration, a total of ${}_4P_4 = 24$ possible substitution sets can be used. However, in previous research, 8 substitution sets have been commonly utilized, which are shown in Table 1 [20], [54]. In base-4, representation of values is performed as regular base transformation, which is performed as per the relation:

$$\{(00_2 \rightarrow 0_4), (01_2 \rightarrow 1_4), (10_2 \rightarrow 2_4), (11_2 \rightarrow 3_4)\}. \quad (16)$$

Post transformation to DNA (taking the equivalent base-4 translation in perspective) performed on both image data and key data, an operation is to be performed. As previously mentioned, the operations or functions of desire are those which are both binary (taking 2 inputs), and reversible. For elaboration sake, out of the 8 mentioned substitutions permutations, the first substitution and the base-4 equivalent are used in the rest of the discussion, which is given by the relation:

TABLE 1. DNA to bit pairs assignment permutations.

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

TABLE 2. *XOR* operation over DNA and base-4 representations.

	(A,0)	(T,1)	(C,2)	(G,3)
(A,0)	(A,0)	(T,1)	(C,2)	(G,3)
(T,1)	(T,1)	(A,0)	(G,3)	(C,2)
(C,2)	(C,2)	(G,3)	(A,0)	(T,1)
(G,3)	(G,3)	(C,2)	(T,1)	(A,0)

TABLE 3. *XNOR* operation over DNA and base-4 representations.

	(A,0)	(T,1)	(C,2)	(G,3)
(A,0)	(G,3)	(C,2)	(T,1)	(A,0)
(T,1)	(C,2)	(G,3)	(A,0)	(T,1)
(C,2)	(T,1)	(A,0)	(G,3)	(C,2)
(G,3)	(A,0)	(T,1)	(C,2)	(G,3)

$$(00 \rightarrow (A, 0)), (01 \rightarrow (T, 1)),$$
$$(10 \rightarrow (C, 2)), (11 \rightarrow (G, 3)). \quad (17)$$

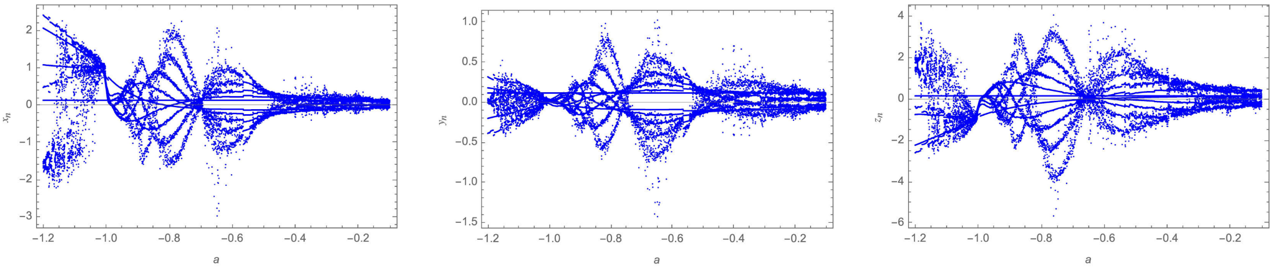
Starting with the operations applicable on the bit-level, the *XOR* and the *XNOR* can be performed in DNA and base-4 as regular bit-level operations, as shown in Tables 2 and 3. As this operation is a one-to-one mapping from the bit-level operation, there is not much added value in using it in DNA-level. On the other hand, it can be better utilized in an interleaving manner (as alternating between *XOR* and *XNOR*), or in combination with other operations, as later discussed. Nevertheless, *XOR* and *XNOR* can be seen as examples of self-reversing operations, as in, for example:

$$XOR(x, y) = z \rightarrow XOR(z, y) = x. \quad (18)$$

As a representation of the other group of operations, addition and subtraction represent the pair-reversing operations such that:

$$Add(x, y) = z \rightarrow Sub(z, y) = x. \quad (19)$$

Tables 4 and 5 demonstrates the results of DNA and base-4 addition and subtraction. Moreover, as in the case of bit-level, the NOT gate can be used in succession to the addition and subtraction processes, introducing more applicable operations. However, the main conclusion here is that as a direct result to the increase in numerical presentation, from base-2 in bit-level to base-4 in DNA-level, the possible ranges of

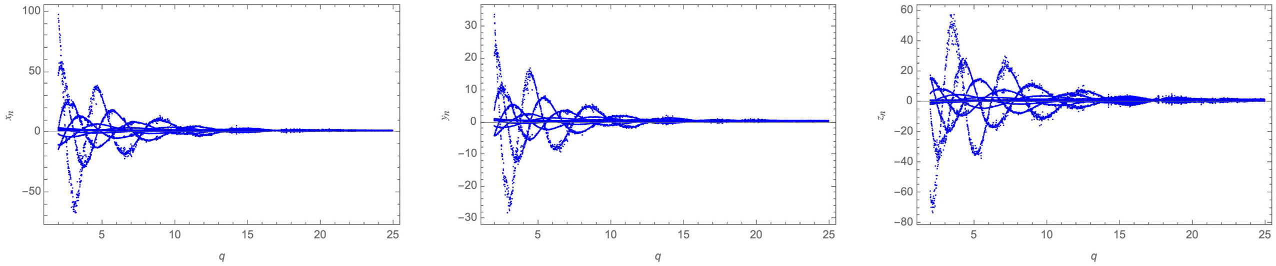


(a) x.

(b) y.

(c) z.

FIGURE 6. Bifurcation plots of the Chua system for x, y and z against a.



(a) x.

(b) y.

(c) z.

FIGURE 7. Bifurcation plots of the Chua system for x, y and z against q.

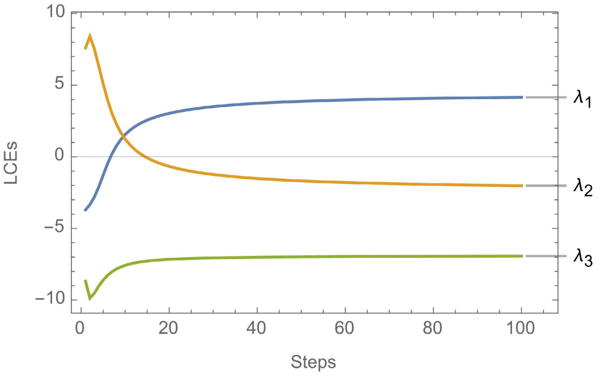


FIGURE 8. A plot of the 3 Lyapunov characteristic exponents of the Chua system.

TABLE 4. Addition operation over DNA and base-4 representations.

	$(A,0)$	$(T,1)$	$(C,2)$	$(G,3)$
$(A,0)$	$(A,0)$	$(T,1)$	$(C,2)$	$(G,3)$
$(T,1)$	$(T,1)$	$(C,2)$	$(G,3)$	$(A,0)$
$(C,2)$	$(C,2)$	$(G,3)$	$(A,0)$	$(T,1)$
$(G,3)$	$(G,3)$	$(A,0)$	$(T,1)$	$(C,2)$

functions increased as well. Furthermore, the availability of multiple operations adds a layer of confusion in contrast to the most commonly used *XOR* logic gate. This is especially true if various operations are used in an alternating way, as later discussed. Hence, the enlargement of the numerical base and the increase in the number of applicable operations is the logical next step, as discussed in Section III-C.

TABLE 5. subtraction operation over DNA and base-4 representations.

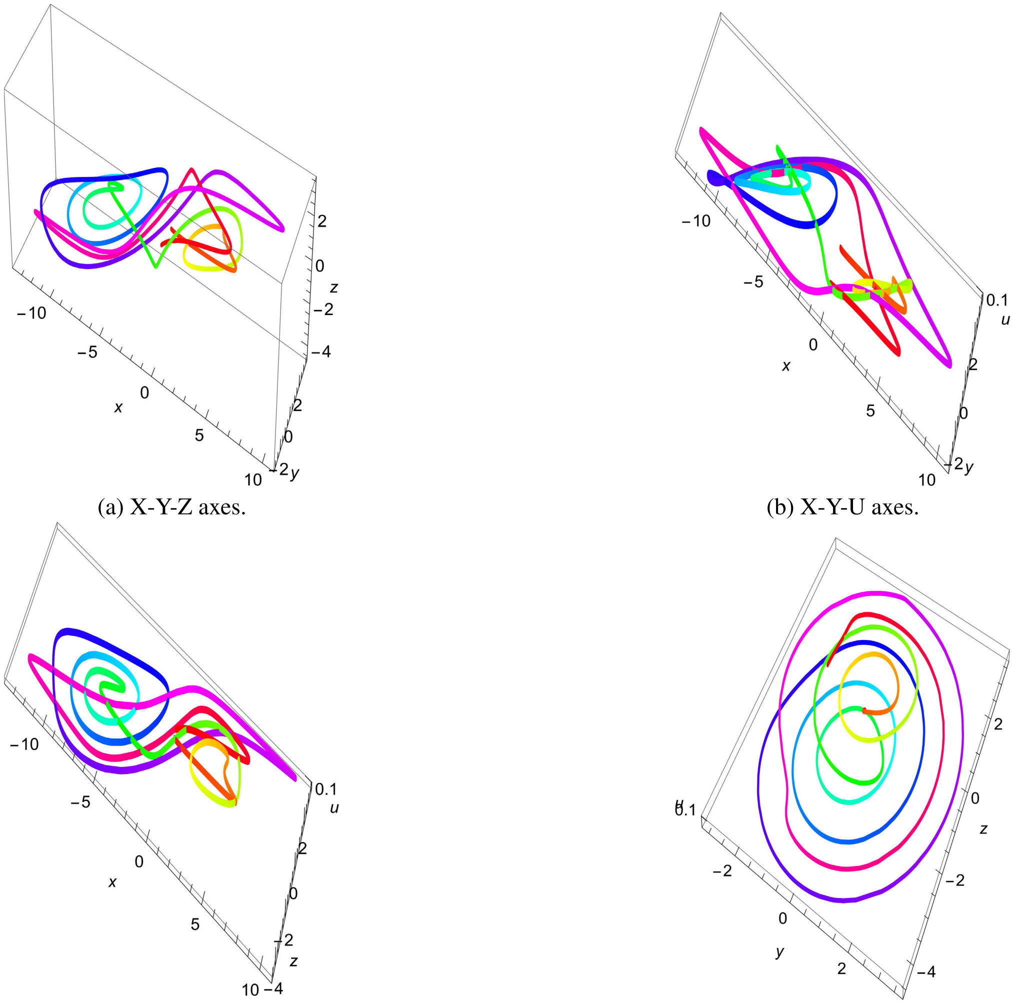
	$(A,0)$	$(T,1)$	$(C,2)$	$(G,3)$
$(A,0)$	$(A,0)$	$(G,3)$	$(C,2)$	$(T,1)$
$(T,1)$	$(T,1)$	$(A,0)$	$(G,3)$	$(C,2)$
$(C,2)$	$(C,2)$	$(T,1)$	$(A,0)$	$(G,3)$
$(G,3)$	$(G,3)$	$(C,2)$	$(T,1)$	$(A,0)$

C. BASE-n LEVEL OPERATIONS

As discussed in the previous sections, increasing the numerical base of the data representation allows for more flexibility in operations design. This comes as a direct result of the increase taking effect on the total number of possible domain-to-range permutations, treating operations as functions. Hence, the main scope of this work is to explore the possibility of scaling up the numerical base dynamically. This is discussed in terms of, given the image data and the key data transformed to a certain base-n, 5 example operations. Needless to say, these operations are binary (taking image data and key data as input), and reversible (as self-reversing or pair-reversing). Additionally, there are no doubts that these 5 operations are showcase examples, which are:

1) *XOR*.
2) *XNOR*.
3) *Add*/*Sub*.
4) *Mod*.
5) Rot_r/Rot_l

As in the case of Section III-B, *XOR*, *XNOR*, Add and Sub (Tables 2 and 4, respectively) can be scaled up to base-n naturally. Starting with *XOR* and *XNOR* as examples 1 and



(a) X-Y-Z axes.

(b) X-Y-U axes.

(c) X-Z-U axes.

(d) Y-Z-U axes.

FIGURE 9. 3D plots for the fractional order Memristor system over different combinations of axes where $\{x, y, z, u\} = 0.3$, $a = 1.5$, $b = 360$, $c = 0.0326$, $d = 36$, $e = -1.5$, $f = -0.0213$, $g = 0.08$, and $\alpha = 0.97$ (the system is calculated in the 4D space, hence initial values are needed for the four axes. However, for plotting purposes, one axis is ignored in each illustration).

2, they are to be applied as the case they are applied on a bit-level. As in the case of bit-level and DNA-level, both of them are self-reversing. Similarly, addition and subtraction, which are example 3, are pair-reversing, as in the case of DNA. However, for a more generic *n*-level implementation, for image data and key in base-*n*, namely I_n and K_n, addition and subtraction are performed circularly, as per (20) and (21) respectively.

$$Add(I_n, K_n, n) = \begin{cases} I_n + K_n, & I_n + K_n < n \\ I_n + K_n - n, & otherwise, \end{cases} \tag{20}$$

$$Sub(I_n, K_n, n) = \begin{cases} I_n - K_n, & I_n - K_n > 0 \\ I_n - K_n + n, & otherwise, \end{cases} \tag{21}$$

For example 4, a self-reversing modulus-based arithmetic operation is applied. The operation is equated as:

$$Mod(I_n, K_n, n) = (n - (I_n + K_n)\%n)\%n \tag{22}$$

For example 5, bit-wise rotation-based pair-reversing operations are utilized. Due to the availability of more bits per input, bit-wise cyclic rotation provides a larger range of results. Rotation-based operations, namely Rot_r, for rotations to the right, and Rot_l, for rotations to the left, are equated as in (23) and (24) respectively.

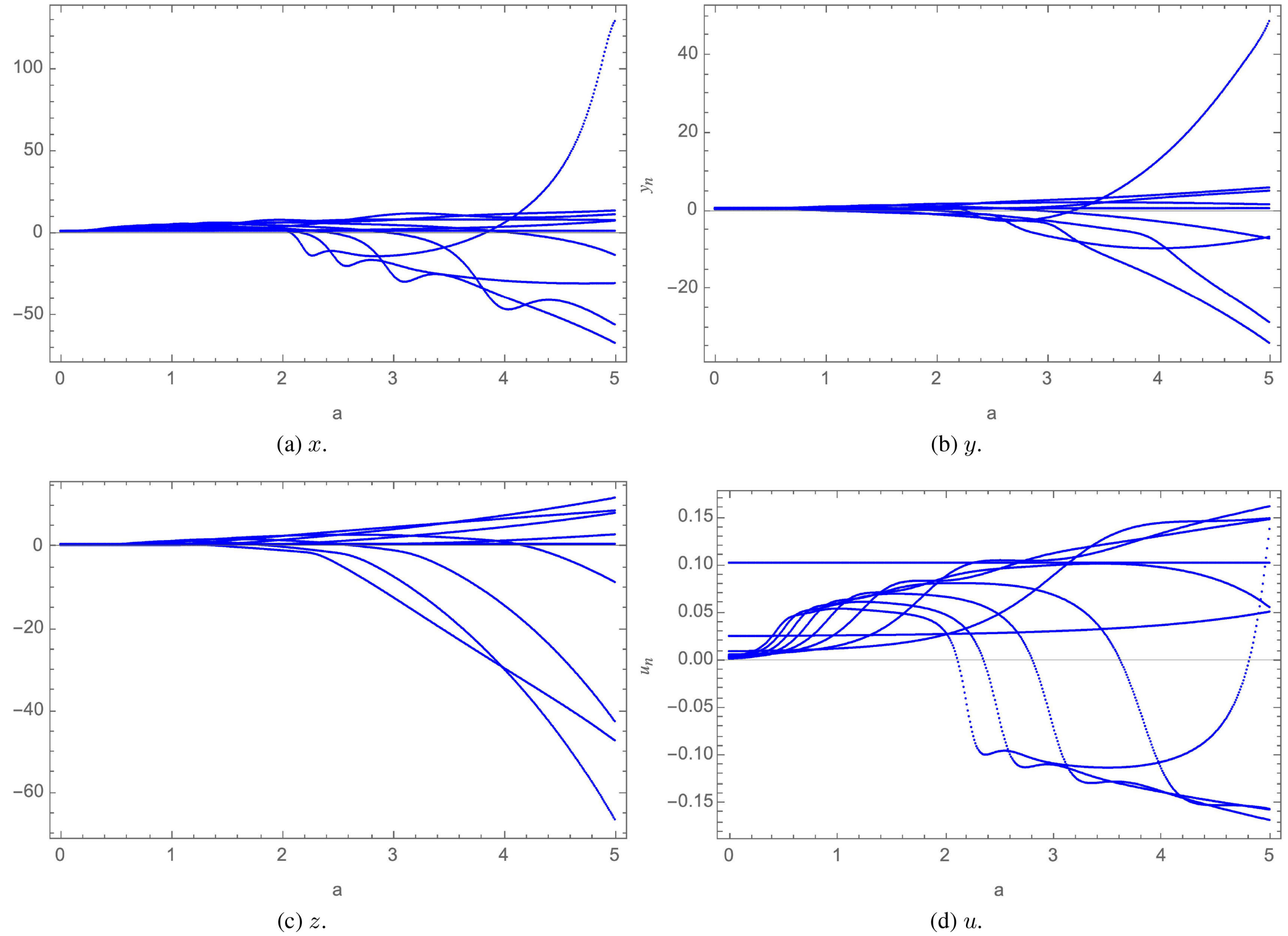
$$Rot_r(I_n, K_n, n) = I_n \gg (K_n\%n) \tag{23}$$

$$Rot_l(I_n, K_n, n) = I_n \ll (K_n\%n) \tag{24}$$

Such that $x \gg y$ means shift x to the right by y positions, and $x \ll y$ means shift x to the left by y positions. Using these 5 operations, 2 sets are formed, one for encryption operations, the other is for the decryption ones, as shown in (25) and (26) respectively.

$$EncOp = \{XOR, XNOR, Add, Mod, Rot_r\} \tag{25}$$

$$DecOp = \{XOR, XNOR, Sub, Mod, Rot_l\} \tag{26}$$



(a) x.

(b) y.

(c) z.

(d) u.

FIGURE 10. Bifurcation plots of the Memristor system for x, y, z and u against a.

As shown in (25) and (26), each operation in $EncOp_i$ has its reversing operation in the same position $DecOp_i$, in order to handle both self-reversing and pair-reversing operations alike.

D. KEY APPLICATION PROCEDURE

Based upon discussion in subsection III-C, instead of always resorting to XOR use, other operations can be applied instead. However, in this work, the followed approach makes use of all operations concurrently. This gets to be feasible by defining an operation selection mechanism. Hence, the encryption process is performed as per these steps:

1) Both the image and key are transformed into 1D bit-streams (base-2) producing 2 sets I_2 and K_2 both of the same size s.
2) Both I_2 and K_2 are transformed to the same base-n (such that n becomes one of the seeds of the process), producing

$$I_n = \{i_1, i_2, \ldots, i_{(s/n)}\}_n$$

and

$$K_n = \{k_1, k_2, \ldots, k_{(s/n)}\}_n$$

both of the same size s/n.
3) Given a set of operations $EncOp$ (for example, (25)) of size s_{op} ($s_{op} = 5$ in (25)), a seed $seed_{op}$ is used to generate a set of numbers Sel_{op} such that

$$Sel_{op} = \{sel_1, sel_2, \ldots sel_{(s/n)}\} | sel_w \in [1, s_{op}].$$

4) Using the encryption method:

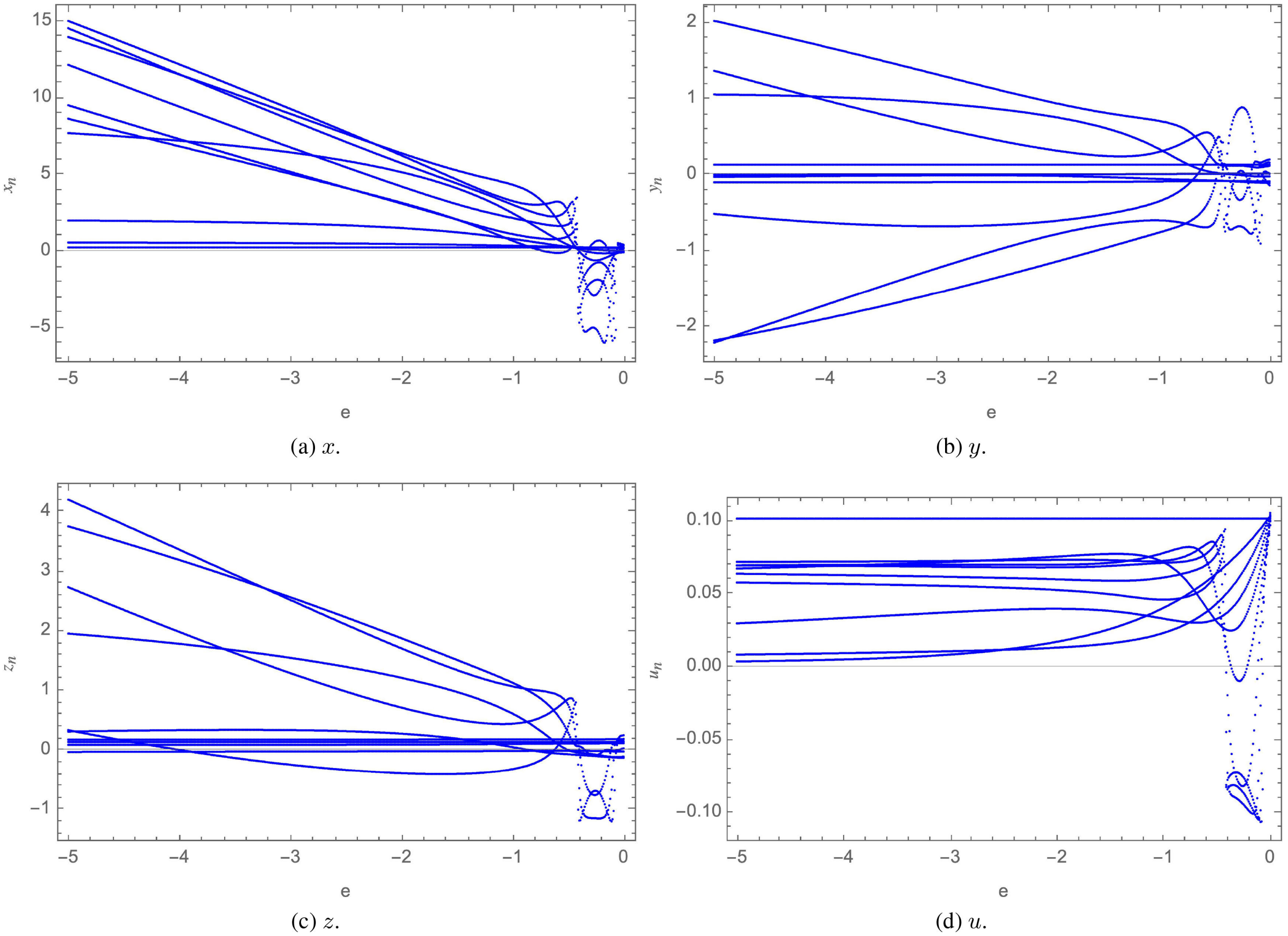
$$Enc(i_j, k_j, n, sel_j) = EncOp_{sel_j}(i_j, k_j, n), \tag{27}$$

I'_n is equated as:

$$I'_n = \left\{ \begin{array}{c} Enc(i_1, k_1, n, sel_1), \\ Enc(i_2, k_2, n, sel_2), \\ \ldots, \\ Enc\left(i_{(s/n)}, k_{(s/n)}, n, sel_{(s/n)}\right) \end{array} \right\}_n .$$

As this procedure would be used as one of the encryption stages, I'_n would be later transformed to the target base of the stage to follow. For decryption, given I'_n, K_n, and Sel_{op}, and using the decryption method:

$$Dec(i'_j, k_j, n, sel_j) = DecOp_{sel_j}(i'_j, k_j, n), \tag{28}$$



(a) x.

(b) y.

(c) z.

(d) u.

FIGURE 11. **Bifurcation plots of the Memristor system for *x*, *y*, *z* and *u* against *e*.**

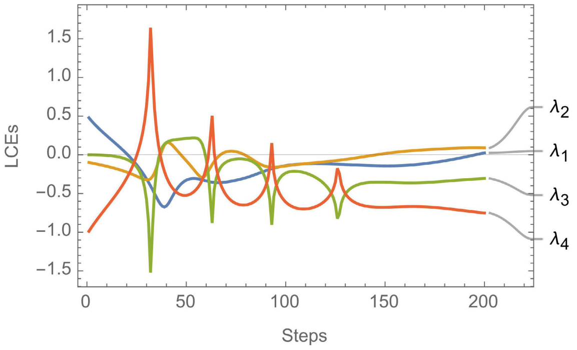


FIGURE 12. **A plot of the 4 Lyapunov characteristic exponents of the Memristor system.**

knowing that both *EncOp* and *DecOp* are of the same size s/n, I_n is reconstructed as:

$$I_n = \left\{ \begin{array}{c} Dec(i'_1, k_1, n, sel_1), \\ Dec(i'_2, k_2, n, sel_2), \\ \dots, \\ Dec\left(i'_{(s/n)}, k_{(s/n)}, n, sel_{(s/n)}\right) \end{array} \right\}_n .$$

IV. SEED-BASED S-BOX GENERATION AND APPLICATION

As the main source for non-linearity, and in accordance to Shannon's property of confusion [47], an S-box is considered a component of prime importance in image encryption algorithms. Therefore, over the years, many S-boxes have been proposed and evaluated towards achieving the highest possible levels of confusion [45]. Accordingly, most research efforts in this area focused mainly on generating 16 × 16 S-boxes, then subject them to a number of evaluations in order to showcase their confusion capabilities [45]. It is worth noting here that 16×16, which are a total of 256 unique values covering the range between 0 and 255, is most commonly used as they cover all the possible values in an image (as grayscale intensities take on values $\in [0, 255]$, in each of the RGB color channels). Hence, it is used as a means for changing the values inside the image by substituting for them by other values within the same range.

In this work, a similar discussion as in Section III-C is proposed, which is for confusion to be performed over a variable base (instead of the traditional base-8, where its elements $\in [0, 255]$). Moreover, instead of relying on a single

S-box, the proposal of alternating between multiple S-boxes is introduced as well (partially similar to operation selection, Section III-D). Both of these propositions aim at increasing the level of confusion applied by the S-box, as changing the base of S-box and providing multiple possibilities changes the confusion mechanism as a whole. Towards that, 2 concepts are discussed, a variable base S-box instead of the classical 16 × 16 approach (Section IV-A), and the application of parallel S-boxes instead of 1 (Section IV-B). Moreover, Section IV-C demonstrates the incorporation of the 2 concepts into one S-box application technique.

A. BASE-*n* S-BOXES

In this section, an S-box generation technique is defined. As demanded in this work, there are 2 properties required out of the S-box generation mechanism. The first requirement, which is directly addressed here, is that the S-box generated is to be of a variable base-n. The second requirement is, as multiple S-boxes are required to be applied in parallel (as discussed in Section IV-B), the generation mechanism should be able to generate varying S-boxes. Fulfilling both of these requirements, the proposed S-box generation mechanism aims more towards transforming a bit-stream into an S-box. Taking into consideration that such a design passively transforms this technique into a seed-based S-box generation technique as the given bit-stream is seed-based (as in the case of Sections II-A, II-B, and II-C).

As per the previous discussion, given a bit-steam S_2 of size $n \times 2^n$, where n is the desired base, Algorithm 1 is followed to generate an S-box in base-n given S_2. For an unsorted list (containing repeated values) of numbers in base-n, $S_n \in [0, 2^n - 1]$, and a sorted set (not containing repeated values) $L = [0, 1, 2, \dots, 2^n - 1]$, $S_n[m]$ denotes the element to move from L to the resulting S-box. Moreover, to ensure that $S_n[m]$ is within the size of L, the modulus operation is employed. Accordingly, Algorithm 1 acts as a controlled reshuffle technique of L, such that controlling the reshuffling is imposed by using the seed-based stream S_n as the selection sequence. Hence, different S_2's result in producing different S_n's, which in turn results in different S-boxes, achieving variability. Moreover, the reconstruction of a certain S-box is possible using the same bit-stream.

B. PARALLEL S-BOXES

In Section IV-A, given n as the base of S-box application, and a bit-stream, an S-box is generated. Correspondingly, the idea in this section is simply to provide m number of bit-streams resulting in generating m S-boxes, all for the same base-n, which is equated as:

$$\begin{aligned} &\{S_1, S_2, \dots, S_m\} \\ &\qquad \downarrow_{Algorithm\ 1} \\ &S\text{-}boxList = \{S\text{-}box_1, S\text{-}box_2, \dots, S\text{-}box_m\}, \end{aligned} \tag{29}$$

for $\{S_1, S_2, \dots, S_m\}$ being a set of bit-streams, each of of size $n \times 2^n$. Beside $S\text{-}boxList$, a list of the inverses of these S-boxes is needed to regenerate the original input, which is equated as:

$$S\text{-}boxList^{-1} = \{S\text{-}box_1^{-1}, S\text{-}box_2^{-1}, \dots, S\text{-}box_m^{-1}\}, \tag{30}$$

Unlike in normal S-boxes, where each value v has one substitution $S\text{-}box[v]$, in parallel S-boxes v has m possible substitutions (where m is the total number of S-boxes), as per the equation:

$$S\text{-}boxList[v, u] = S\text{-}box_u[v] | u \in [1, m]. \tag{31}$$

Respectively, the inverse of $S\text{-}boxList[v, u]$ is equated as:

$$S\text{-}boxList^{-1}[v, u] = S\text{-}box_u^{-1}[v] | u \in [1, m]. \tag{32}$$

The main aim of this is, for the same value of v, different substitutions would take place in different occasions, based on different selections of S-boxes i. As a result for the same value v getting different substitutions, non-linearity is increased, which results in enhancing the confusion aspect.

C. APPLICATION OF BASE-*n* PARALLEL S-BOXES

As discussed in Sections IV-A, and IV-B, given a value for the base, n, and a set of m bit-streams, a set of S-boxes is generated using (29). Respectively, for an image in base-n,

$$I_n = \{i_1, i_2, \dots, i_j\},$$

a selections set is needed of the same size as I_n,

$$Sel_{S\text{-}box} = \{sel_1, sel_2, \dots, sel_j\} | sel_w \in [1, m],$$

which is a list of random numbers generated using a seed value $seed_{S\text{-}boxList}$. Accordingly, applying the S-box (using (31)) is performed as:

$$I'_n = \left\{ \begin{matrix} S\text{-}boxList[i_1, sel_1], \\ S\text{-}boxList[i_1, sel_2], \\ \dots, \\ S\text{-}boxList[i_j, sel_j] \end{matrix} \right\}_n .$$

Similarly, I_n is reconstructed by applying the inverses of the S-boxes in (32) as:

$$I_n = \left\{ \begin{matrix} S\text{-}boxList^{-1}[i'_1, sel_1], \\ S\text{-}boxList^{-1}[i'_1, sel_2], \\ \dots, \\ S\text{-}boxList^{-1}[i'_j, sel_j] \end{matrix} \right\}_n .$$

Algorithm 1 Generate S-Box Given S_2 and n

```
1) $L = [0, 1, 2, \dots, 2^n - 1]$
2) S-box= {}
3) convert $S_2$ of size $n \times 2^n$ to base-$n$ generating $S_n$ of size
   $2^n$ (same size as $L$)
4) $m = 0$
5) $i = S_n[m]\%Length(L)$
6) append $L[i]$ to S-box
7) delete $L[i]$ from $L$
8) $m = m + 1$
9) if $Length(L) > 0 : GoTo(5)$
10) return S-box
```

V. PROPOSED IMAGE ENCRYPTION TECHNIQUE

In this work, many components which can be utilized for constructing symmetric image encryption techniques were presented. These components can be summarized as:

- Bit-Streams Generation:
  - Chen System (Section II-A).
  - Chua System (Section II-B).
  - Memristor System (Section II-C).
- Key application on base-n (Section III-D).
- Parallel base-n S-box application (Section IV-C).

Utilizing these components, a 3-stage encryption technique is proposed, as per the following steps:

1) Stage 1: Chen system key diffusion.
   a) First, the input color image, I_c, of dimensions $M \times N$, is converted into a 1D bit-stream to produce the set I.
   b) Given a set of seeds for the Chen system of size 13, a bit-stream is generated of the same length as I.
   c) For a base-n_{s1}, and a set of operations $EncOp$, and a seed for operations selections $seed_{op1}$, the Key application in Section III-D is applied, generating I_{s1}.
2) Stage 2: Memristor parallel base-n S-box Application.
   a) Given a base-n_{s2}, and a set of seeds of size $m \times 15$, a set of m bit-streams is generated, which are used in generating a set of m S-boxes in base-n_{s2}, as discussed in Section IV-B.
   b) Transform I_{s1} to base-n_{s2}, then apply the generated S-box as discussed in Section IV-C, generating $I_{s1,s2}$.
3) Stage 3: Chua system key diffusion.
   a) For a base-n_{s3}, $I_{s1,s2}$ is transformed to that base.
   b) Given a set of seeds for the Chua system of size 10, a bit-stream is generated of the same length as I, then transformed to base-n_{s3}.
   c) For a set of operations $EncOp$, and a seed for operations selections $seed_{op3}$, the key application in Section III-D is applied, generating $I_{s1,s2,s3}$.

After performing the 3 stages, reshaping $I_{s1,s2,s3}$ back into a 2D image (of dimensions $M \times N$) results in the encrypted image I'. Fig. 13 shows the flowchart of the proposed image encryption technique. It is worth noticing here that, as per the definition of the key application technique (as it relies on any bit-stream), and the parallel S-box generation mechanism (which operates using any bit-stream as well), other encryption procedures can be created using the defined components.

Accordingly, for the decryption process to be performed, given I' and the encryption seeds, these steps are to be followed:

1) Stage 3: Chua system key diffusion.
   a) First, the input color image, I'_c, of dimensions $M \times N$, is converted into a 1D bit-stream to produce the set $I_{s1,s2,s3}$.
   b) For a base-n_{s3}, $I_{s1,s2,s3}$ is transformed to that base.
   c) Given a set of seeds for the Chua system of size 10, a bit-stream is generated of the same length as $I_{s1,s2,s3}$, then transformed to base-n_{s3}.
   d) For a set of operations $DecOp$, and a seed for operations selections $seed_{op3}$, the key application in Section III-D is applied, generating $I_{s1,s2}$.
2) Stage 2: Memristor parallel base-n S-box Application.
   a) Given a base-n_{s2}, and a set of seeds of size $m \times 15$, a set of m bit-streams is generated, which are used in generating a set of m S-boxes in base-n_{s2}, as discussed in Section IV-B.
   b) Transform $I_{s1,s2}$ to base-n_{s2}, then apply the generated S-box as discussed in Section IV-C, generating I_{s1}.
3) Stage 1: Chen system key diffusion.
   a) Given a set of seeds for the Chen system of size 13, a bit-stream is generated of the same length as $I_{s1,s2,s3}$.
   b) For a base-n_{s1}, and a set of operations $DecOp$, and a seed for operations selections $seed_{op1}$, the Key application in Section III-D is applied, generating I.

Re-transforming the resulting set I into a 2D image of size $M \times N$ will produce the colored image I_c back, concluding the decryption process. Fig. 14 shows the flowchart of the proposed image decryption technique.

VI. PERFORMANCE EVALUATION AND NUMERICAL RESULTS

This section encompasses a number of tests and computation of metrics that aim at showcasing the performance of the proposed color image encryption technique. This includes testing against visual, statistical, differential, randomness, entropy and brute-force cryptanalysis measures. Testing is carried out in the Wolfram language v.13.1 on a machine that employs macOS Catalina v.10.15.7, equipped with a 2.9 Ghz 6-Core Intel Core i9 processor and 32 GB of 2400 MHz DDR4 RAM. The graphics card is a Radeon Pro Vega 20 with 4 GB, supplemented with an Intel UHD Graphics 630 card with 1535 MB. A set of images that are popular within the image processing community is utilized, all of dimensions 256×256, unless otherwise stated. These are: Lena, Mandrill, Peppers, House, House2, Girl, Sailboat and Tree. The chosen values of the variables used in the keys generation are:

1) Stage 1: Chen system key diffusion.
   a) Number of operations: 5.
   b) Base: 16.
   c) Operations selection seed: 4444 on Mersenne Twister.

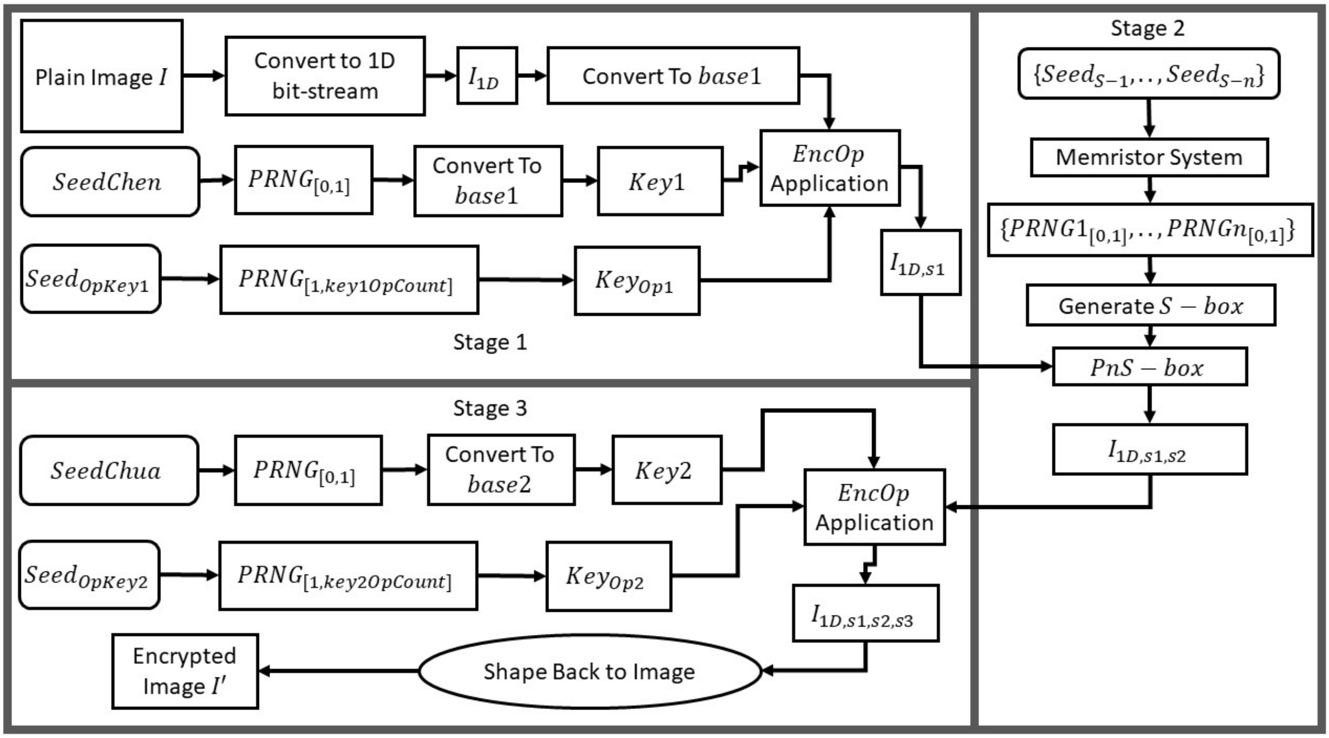


FIGURE 13. Flowchart of the encryption algorithm of the proposed image encryption technique.

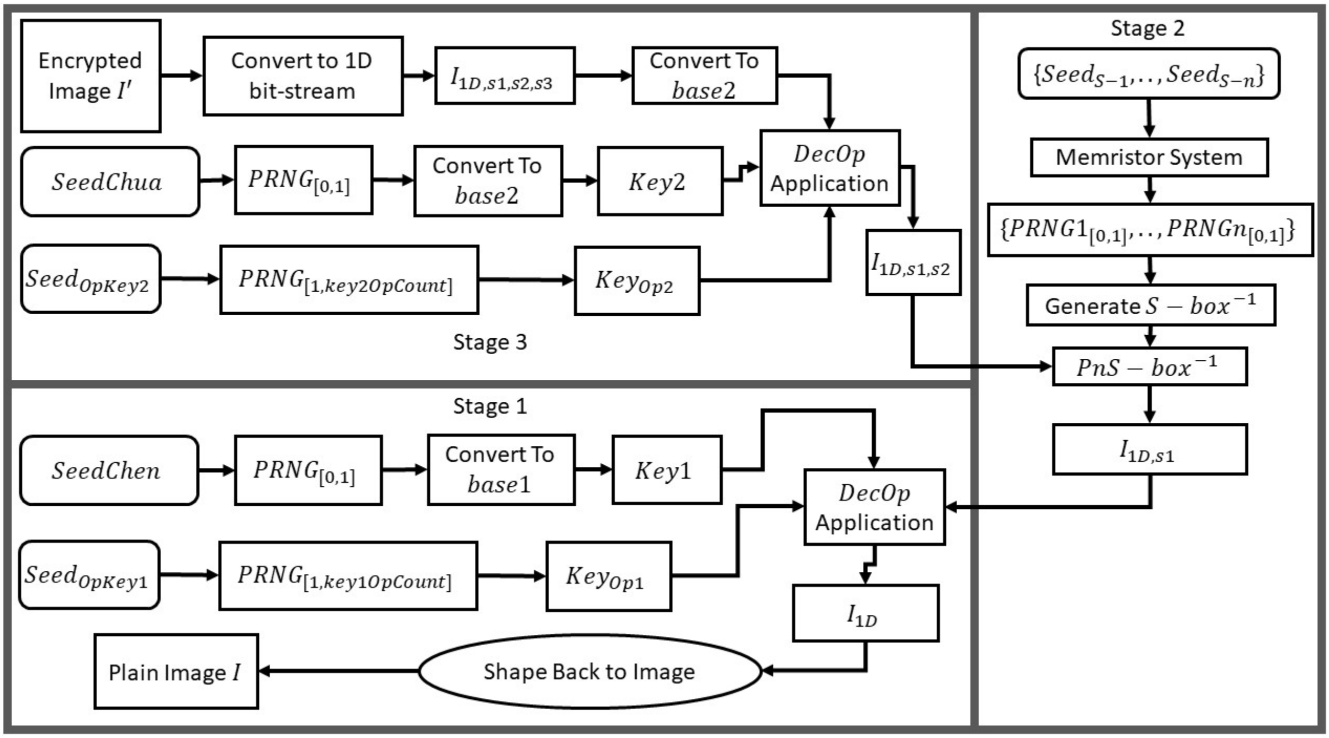


FIGURE 14. Flow chart of the decryption algorithm of the proposed image encryption technique.

   d) The set of seeds: $\{x, y, z, u\} = 0.3, a = 35, b = 3, c = 12, \gamma = 7, d = 0.5, \alpha_1 = 0.85, \alpha_2 = 0.7, \alpha_3 = 0.55$, and $\alpha_4 = 0.95$.

2) Stage 2: Memristor parallel base-*n* S-box generation and application.

   a) Number of parallel S-boxes: 10.
   b) Base: 12.
   c) S-box selections seed: 5555 on Mersenne Twister.
   d) The set of seeds: Shown in Table 6.

3) Stage 3: Chua system key diffusion.

a) Number of operations: 5.
b) Base: 8.
c) Operations selection seed: 3333 on Mersenne Twister.
d) The set of seeds: $\{x, y, z\} = 0.3$, $a = 1.27$, $b = -0.68$, $p = 10$, $q = 14.87$, $\alpha_1 = 0.7$, $\alpha_2 = 0.55$, and $\alpha_3 = 0.95$.

The carried out security analyses are:

1) Visual and Histogram Analysis (Section VI-A).
2) Mean Squared Error (Section VI-B).
3) Peak Signal to Noise Ratio (Section VI-C).
4) Mean Absolute Error (Section VI-D).
5) Information Entropy (Section VI-E).
6) Fourier Transformation Analysis (Section VI-F).
7) Correlation Coefficient Analysis (Section VI-G).
8) Differential Attack Analysis (Section VI-H).
9) The National Institute of Standards and Technology Analysis (Section VI-I).
10) Key Space Analysis (Section VI-J).
11) Histogram Dependency Tests (Section VI-K).
12) Execution Time Analysis (Section VI-L).
13) Further Tests on Resistivity to Attacks (Section VI-M)

A. VISUAL AND HISTOGRAM ANALYSIS

The simplest measure of how well a cryptosystem performs can be carried out using the human visual system (HVS). Subfigures (a) and (b) in Fig. 15 – Fig. 19, display a number of plain images and their corresponding encrypted versions. It is clear that no visual cues are recognized from any of the encrypted images. Further inspection by the HVS to the histograms in subfigures (c) and (d) of Fig. 15 – Fig. 17 also indicate, statistically, that no information can be inferred from the histograms of the encrypted images. This is because all the histograms of the encrypted images show a uniform distribution of values.

B. MEAN SQUARED ERROR

The mean squared error (MSE) is a simple metric that showcases the difference between 2 data sets. In this case, it is used to showcase the difference between the pixel values of a plain image I and its encrypted version I'. This means that the higher the MSE value, the better the encryption algorithm's ability at removing any similarity between the 2 images. The MSE is mathematically expressed as

$$MSE = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I_{(i,j)} - I'_{(i,j)})^2}{M \times N}, \tag{33}$$

where the dimensions of each image is $M \times N$. Table 7 provides a comparison of MSE values reported in the literature and those computed for the proposed technique. It is clear the proposed technique exhibits comparable performance.

The literature typically reports values of MSE and Peak Signal to Noise Ratio (PSNR) alongside one another, since the mathematical expression for PSNR, in (34), depends on the MSE value. But the authors of [32] only report PSNR values, without their corresponding MSE values. This the reason behind the N/A entries in Table 7 under the heading of [32].

C. PEAK SIGNAL TO NOISE RATIO

The peak signal to noise ratio (PSNR) relates the error margin in a signal (in this case, an encrypted image) to the peak signal value. Since, this metric is applied to images in this research work, the peak signal value is the maximum pixel intensity (i.e. 255). Hence, the PSNR is mathematically expressed as

$$PSNR = 10 \log \left(\frac{I_{max}^2}{MSE}\right), \tag{34}$$

where I_{max} is the maximum pixel intensity in an image I. As is clear from (34), the PSNR is inversely proportional to the MSE. This means that for the purposes of image encryption efforts, a lower PSNR value signifies better encryption abilities of an image cryptosystem. Table 8 provides a comparison of PSNR values reported in the literature and those computed for the proposed technique. As was the case with MSE, it is clear the proposed technique exhibits comparable PSNR performance.

D. MEAN ABSOLUTE ERROR

The mean absolute error (MAE) is a metric that maintains the linearity of the pixel error distribution behavior between 2 images, I and I'. It is expressed mathematically as

$$MAE = \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |I_{(i,j)} - I'_{(i,j)}|}{M \times N}. \tag{35}$$

In a similar fashion to the MSE, maximizing the MAE signifies better encryption ability of an image cryptosystem. Table 9 provides a comparison of MAE values reported in the literature and those computed for the proposed technique. It is clear that a comparable MAE performance with the literature is achieved.

E. INFORMATION ENTROPY

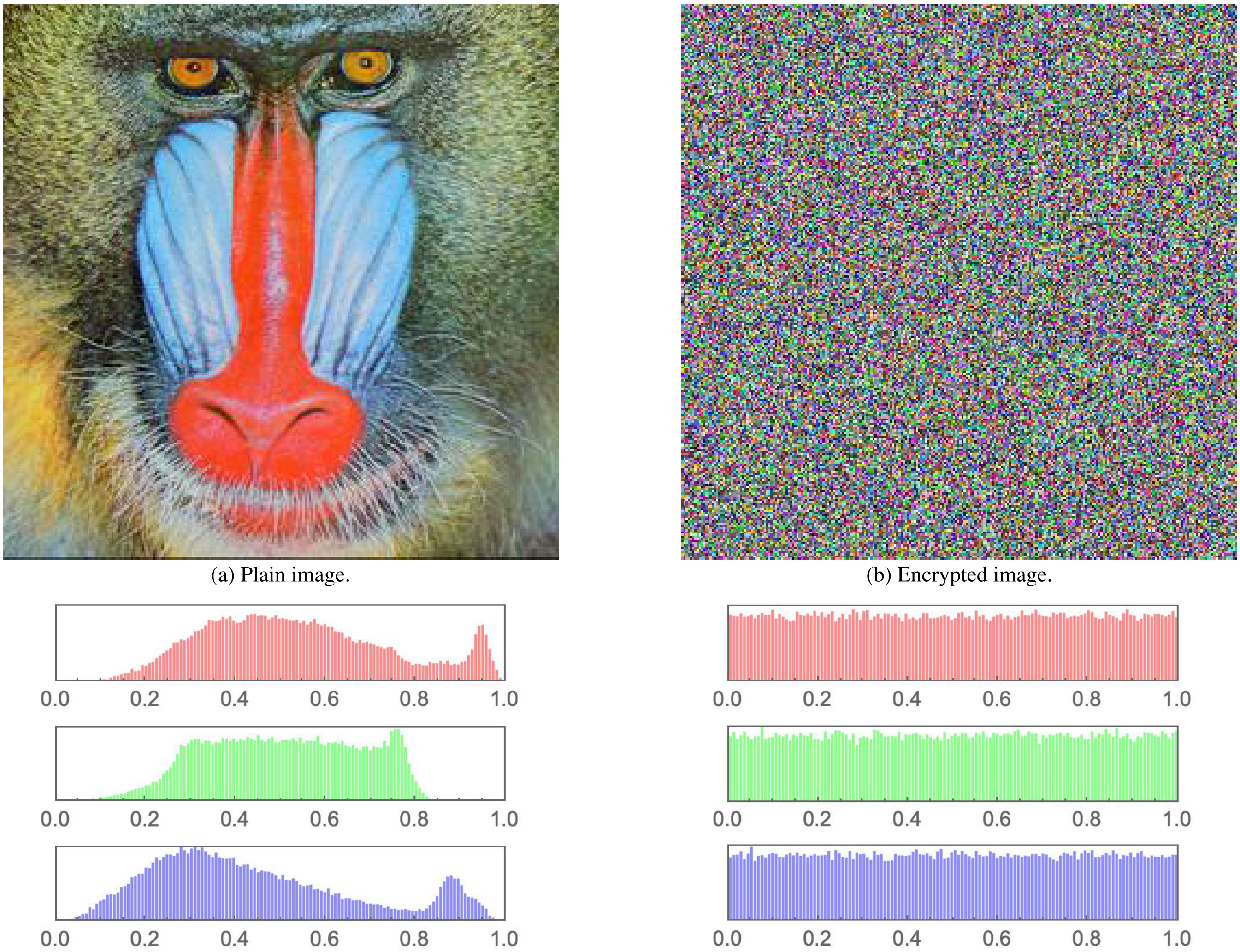
The information entropy, $H(m)$, is a metric that is utilized to measure the degree of randomness in a distribution of gray color intensity in a grayscale image. Thus, an RGB image is first color-separated into its 3 constituent grayscale images and the information entropy is then computed for each, then their average is reported. It is mathematically expressed as

$$H(m) = \sum_{i=1}^{M} p(m_i) \log_2 \frac{1}{p(m_i)}, \tag{36}$$

where $p(m_i)$ refers to the probability of occurrence of symbol m, while M represents the total number of bits for each symbol. The ideal information entropy value of an encrypted image is 8, with practical implementations of cryptosystems coming very close to this value. Table 10 provides a comparison of information entropy values reported in the literature and those computed for the proposed technique. It is clear the proposed technique exhibits comparable performance.

TABLE 6. **Seed values for the Memristor system used for generating the 10 S-boxes used in the performance evaluation and numerical results (all values for *c*, *f* and *y* are multiplied by 10^{-3}, and all values of *e* and *f* are negatively signed).**

S	a	b	c	d	e	f	g	x	y	z	u	α_1	α_2	α_3	α_4
1	1.5	360	32	36	1.5	21	0.08	0.1	1	0.05	0.01	0.9	0.7	0.95	0.9
2	1.8	500	62	42	1.7	24	0.05	0.5	2	0.07	0.08	0.3	0.9	0.45	0.7
3	1.3	439	12	55	1.9	94	0.09	0.1	5	0.02	0.19	0.4	0.4	0.65	0.3
4	1.2	537	52	67	1.4	34	0.13	0.4	23	0.15	0.16	0.5	0.2	0.23	0.2
5	1.3	338	25	52	1.6	65	0.34	0.7	1	0.02	0.76	0.2	0.3	0.43	0.3
6	1.6	732	32	64	1.3	53	0.64	0.3	53	0.03	0.28	0.6	0.1	0.87	0.4
7	1.5	655	53	77	1.5	63	0.67	0.2	76	0.04	0.63	0.7	0.8	0.47	0.5
8	1.1	813	84	12	1.2	35	0.16	0.5	32	0.03	0.22	0.2	0.4	0.92	0.6
9	1.9	908	75	28	1.4	65	0.34	0.4	87	0.02	0.76	0.8	0.9	0.28	0.8
10	1.4	621	82	92	1.6	75	0.76	0.3	32	0.04	0.97	0.9	0.2	0.74	0.9



(a) Plain image.

(b) Encrypted image.

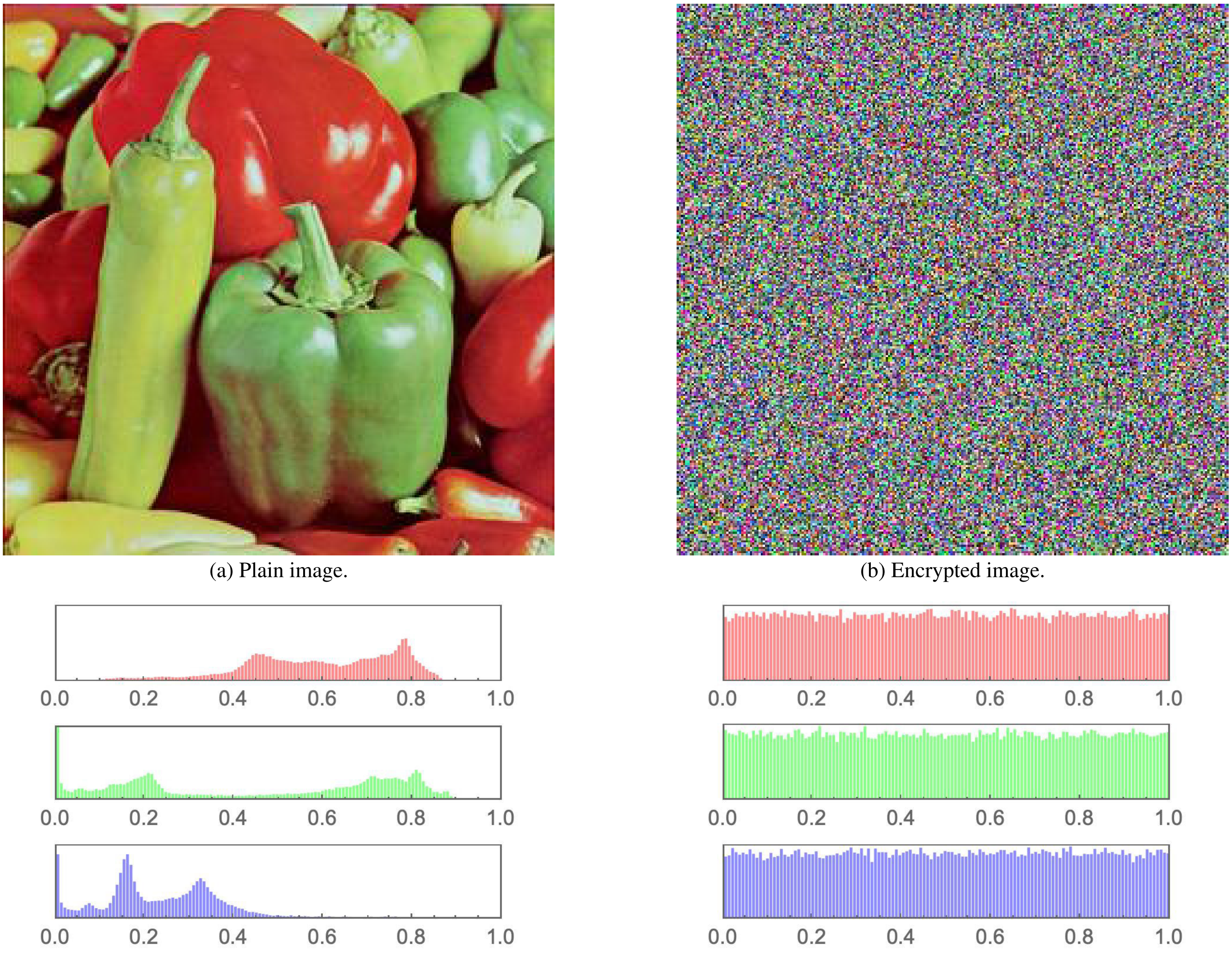
(c) Histogram of the plain image.

(d) Histogram of the encrypted image.

FIGURE 15. **Mandrill image and histogram comparison pre- and post-encryption.**

F. FOURIER TRANSFORMATION ANALYSIS

Pixel cross-correlation pre- and post-encryption is easily examined through applying the Discrete Fourier Transform (DFT) and visually examining the images. The resultant DFT of encrypted images would not convey structural information, unlike that of a plain image which usually displays a structure resembling a plus sign in the center of the image. This is because spatial features of a plain image, such as edges and corners, separate into different frequencies. The larger frequencies of the sine and cosine trigonometric functions



(a) Plain image.

(b) Encrypted image.

(c) Histogram of the plain image.

(d) Histogram of the encrypted image.

FIGURE 16. Peppers image and histogram comparison pre- and post-encryption.

TABLE 7. Comparison of MSE values with other algorithms from the literature.

Image	Proposed	[21]	[4]	[36]	[59]	[5]	[33]	[15]	[26]
Lena	8882.4	9112.1	8927	10869.7	4859	8888.9	N/A	8972.8	8867.4
Mandrill	8316.4	8573.4	8290.9	10930.3	6399.1	8295.2	N/A	8352.8	N/A
Peppers	10081.3	10298.7	10045.1	N/A	7274.4	10092.3	N/A	10069.1	10119.5
House	8310.2	8427	8351.6	N/A	N/A	N/A	N/A	N/A	N/A
House2	9137.1	9374.7	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Girl	12175	12450.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Sailboat	10021.7	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Tree	9927.4	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Average	9606.38	9706.1	8903.6	10900	6177.5	9088.8	N/A	9131.6	9493.5

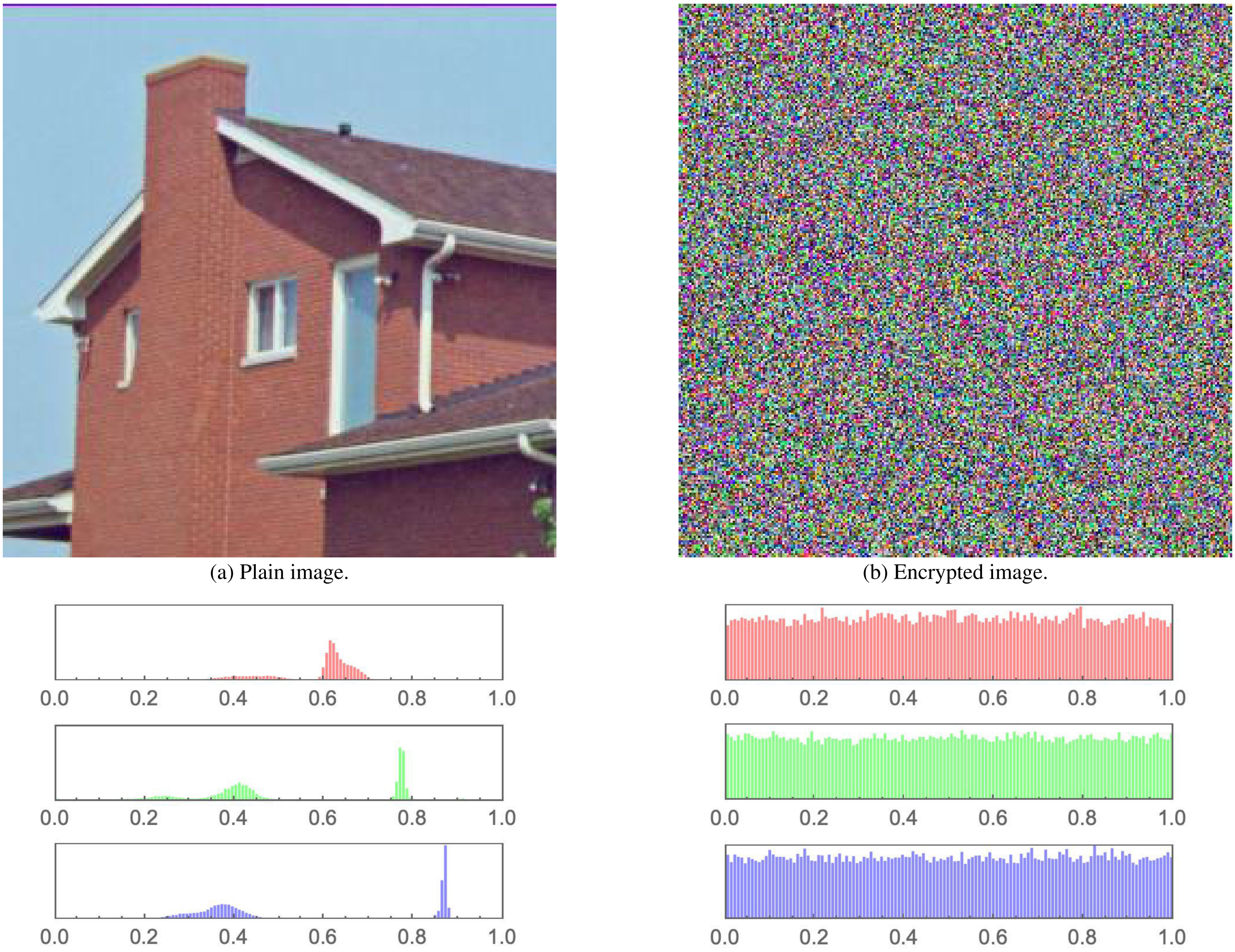
are visually easier to identify in the frequency domain, as a result of proximity of pixel values in the spatial domain. For a square image signal $f(a, b)$, of dimensions $N \times N$, in the spatial domain, its equivalent frequency domain representation would be mathematically expressed as

$$F(k, l) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i, j)e^{-i2\pi(\frac{ki}{N}+\frac{li}{N})}, \tag{37}$$

where every point $F(k, l)$ in the Fourier space is represented in terms of the product of $f(i, j)$ with an exponential basis function. It is clear from Fig. 18 (c), the presence of the white plus sign structure in the center of the DFT of the plain image, unlike Fig. 18 (d) which displays the DFT of the encrypted version, and has no such structure. This signifies the absence of any spatial features in the encrypted image.

G. CORRELATION COEFFICIENT ANALYSIS

A correlation coefficient analysis is carried out to check the cohesion of pixels locally in an image. That is to say, to what



(a) Plain image.

(b) Encrypted image.

(c) Histogram of the plain image.

(d) Histogram of the encrypted image.

FIGURE 17. House image and histogram comparison pre- and post-encryption.

TABLE 8. Comparison of PSNR [dB] values with other algorithms from the literature.

Image	Proposed	[21]	[4]	[36]	[59]	[5]	[33]	[15]	[26]
Lena	8.64553	8.53462	8.6237	7.7677	11.3	8.64233	8.5674	8.6554	9.15
Mandrill	8.93143	8.79929	8.9448	7.7447	10.10	8.94253	10.0322	8.9272	N/A
Peppers	8.09565	8.00296	8.11128	N/A	9.55	8.94253	N/A	8.13789	8.13
House	8.93472	8.87405	8.91309	N/A	N/A	N/A	N/A	N/A	N/A
House2	8.52272	8.41125	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Girl	7.27613	7.17879	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Sailboat	8.12137	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Tree	8.16246	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Average	8.40153	8.30016	8.64822	7.7562	10.3167	8.84246	9.2998	8.5735	8.64

extent nearby pixels share the same or nearly the same colors. For plain images, neighboring pixels mostly share the same color values and thus the evaluation of this metric results in values very close to 1. On the contrary, well-encrypted images should possess a pixel distribution that does not result in any form of similarity among adjacent or nearby pixels. In such a case, this metric would result in values very close to zero. The pixel correlation coefficient is mathematically expressed as

$$\rho(x, y) = \frac{cov(x, y)}{\sqrt{\sigma(x)}\sqrt{\sigma(y)}}, \tag{38}$$

where,

$$cov(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu(x))(y_i - \mu(y)), \tag{39}$$

TABLE 9. Comparison of MAE values with other algorithms from the literature.

Image	Proposed	[21]	[5]	[36]	[37]	[33]	[2]
Lena	77.3307	78.3564	77.3752	87	77.35	77.96	77.4877
Peppers	81.9253	82.3273	81.7740	N/A	74.71	N/A	81.9832
Mandrill	75.0986	81.913	75.1659	92	73.91	67.85	75.1632
House	75.0542	N/A	N/A	N/A	N/A	N/A	75.4983
House2	78.2498	N/A	N/A	N/A	N/A	N/A	78.3327
Girl	90.2537	N/A	N/A	N/A	N/A	N/A	89.9807
Sailboat	81.7614	N/A	N/A	N/A	N/A	N/A	82.1003
Tree	81.4288	N/A	N/A	N/A	N/A	N/A	81.1623
Average	81.4288	80.8656	78.105	89.5	75.6567	72.905	80.2136

TABLE 10. Comparison of information entropy values with other algorithms from the literature.

Image	Proposed	[21]	[4]	[36]	[40]	[59]	[5]	[33]	[15]
Lena	7.9989	7.9856	7.999	7.999	7.997	7.996	7.997	7.9972	7.9989
Mandrill	7.9990	7.9905	7.999	7.999	7.999	N/A	7.996	7.9969	7.9987
Peppers	7.9990	7.9951	7.999	7.9991	N/A	7.997	7.9969	N/A	7.9992
House	7.9985	7.9577	7.999	N/A	N/A	N/A	N/A	N/A	N/A
House2	7.9988	7.9847	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Girl	7.9989	7.9789	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Sailboat	7.9990	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Tree	7.9987	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Average	7.9987	7.9821	7.999	7.999	7.998	7.9965	7.9963	7.9971	7.9989

$$\sigma(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu(x))^2, \tag{40}$$

$$\mu(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i). \tag{41}$$

Table 11 demonstrates the computed adjacent pixel cross-correlation values achieved by the proposed image encryption technique for various images, both plain and encrypted, in 3 dimensions (horizontal, diagonal and vertical). It is clear that plain images' adjacent pixels exhibit high cross-correlation, with values approaching 1. On the contrary, a value approaching 0 is computed for encrypted images. Furthermore, Table 12 provides a comparison with the literature. It is clear that a comparable performance is indeed achieved, with all algorithms, including the proposed one, reporting values that approach 0.

H. DIFFERENTIAL ATTACK ANALYSIS

A differential attack analysis is carried out through comparing a plain and an encrypted image on a pixel-by-pixel basis. The purpose here is to compute a percentage of difference in color intensities resulting from the encryption process. This translate into 2 metrics. The first is the number of pixels changing rate (NPCR), while the second is the unified average change intensity (UACI). For 2 images, I_1 and I_2, of dimensions $M \times N$, such a difference per pixel, $D(x, y)$, is mathematically expressed as

$$D(x,y) = \begin{cases} 0 & I_1(x,y) = I_2(x,y) \\ 1 & Otherwise \end{cases} \Bigg|\, x \in [1,M] \& y \in [1,N] \tag{42}$$

This allows one to express the NPCR as

$$NPCR = \frac{\sum_{x=1}^{M}\sum_{y=1}^{N} D(x,y)}{M \times N} \times 100. \tag{43}$$

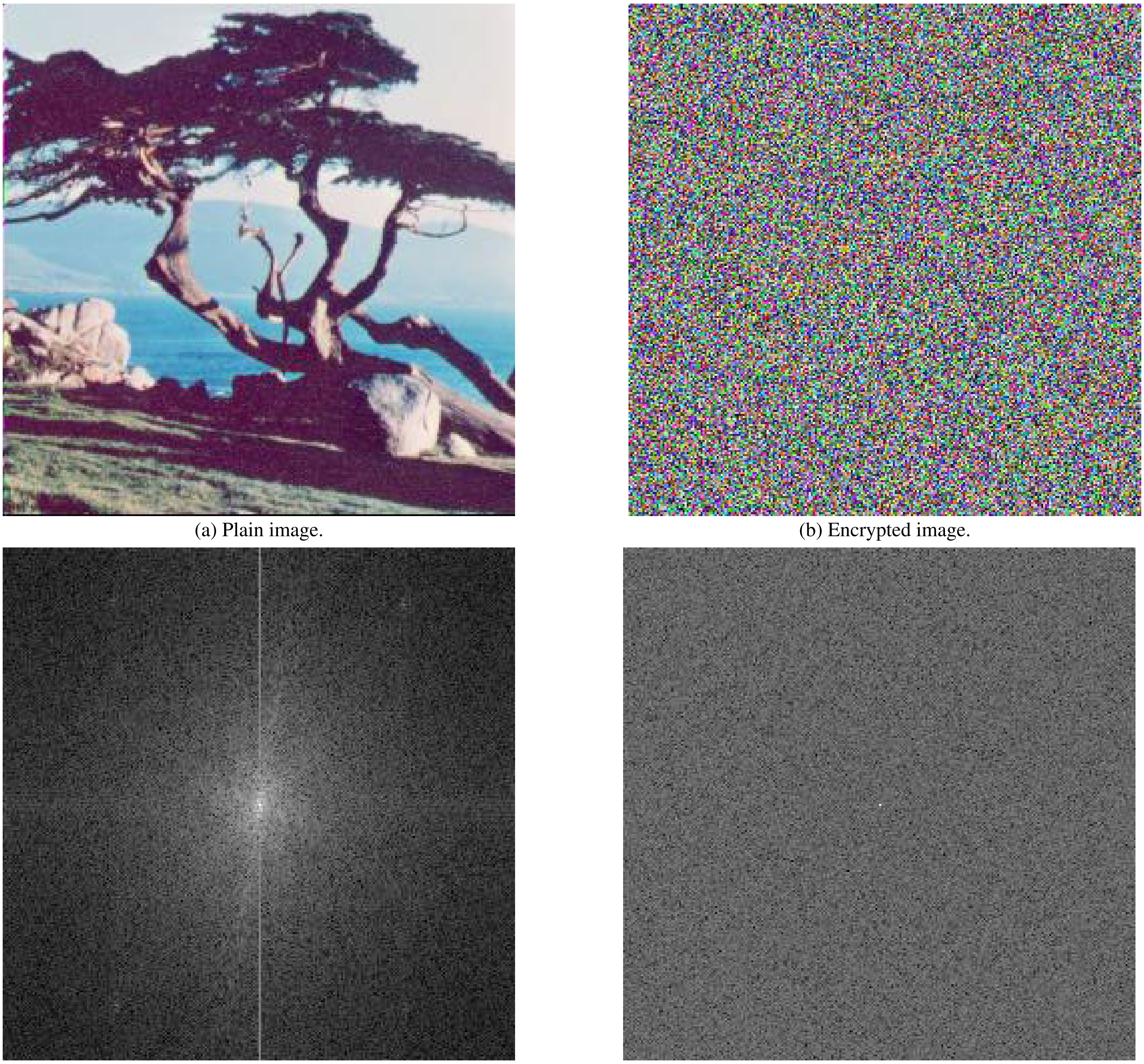
For a well-encrypted image, the NPCR is expected to pass the threshold of 99%. On the other hand, the UACI aims to compute the difference between 2 images through their mean averages. This means that the UACI is mathematically expressed as

$$UACI = \frac{1}{M \times N}\sum_{x=1}^{M}\sum_{y=1}^{N}\frac{|I_1(x,y) - I_2(x,y)|}{255} \times 100. \tag{44}$$

The literature identifies a UACI value of 33% as an indication of a well-encrypted image. Table 13 displays the computed NPCR and UACI values of the proposed color image cryptosystem. It is clear that an average value of the NPCR is ideal, while an average value of the UACI is very close to the ideal value. Moreover, Table 14 provides a comparison with the literature, showcasing very close proximity of the various algorithms.

I. THE NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY ANALYSIS

A trustworthy technique for assessing the randomness of encrypted images is the SP-800 analysis from the National Institute of Standards and Technology (NIST). It consists of a series of tests run on a bit-stream to gauge the degree of randomness provided by a PRNG. A bit-stream's probability, or p-value, needs to be greater than 0.01 in order to pass any of the tests. We show that a lengthy bit-stream created by concatenating the rows of an encrypted image passes the NIST suite of tests for our proposed image encryption



(a) Plain image.

(b) Encrypted image.

(c) Plain image DFT.

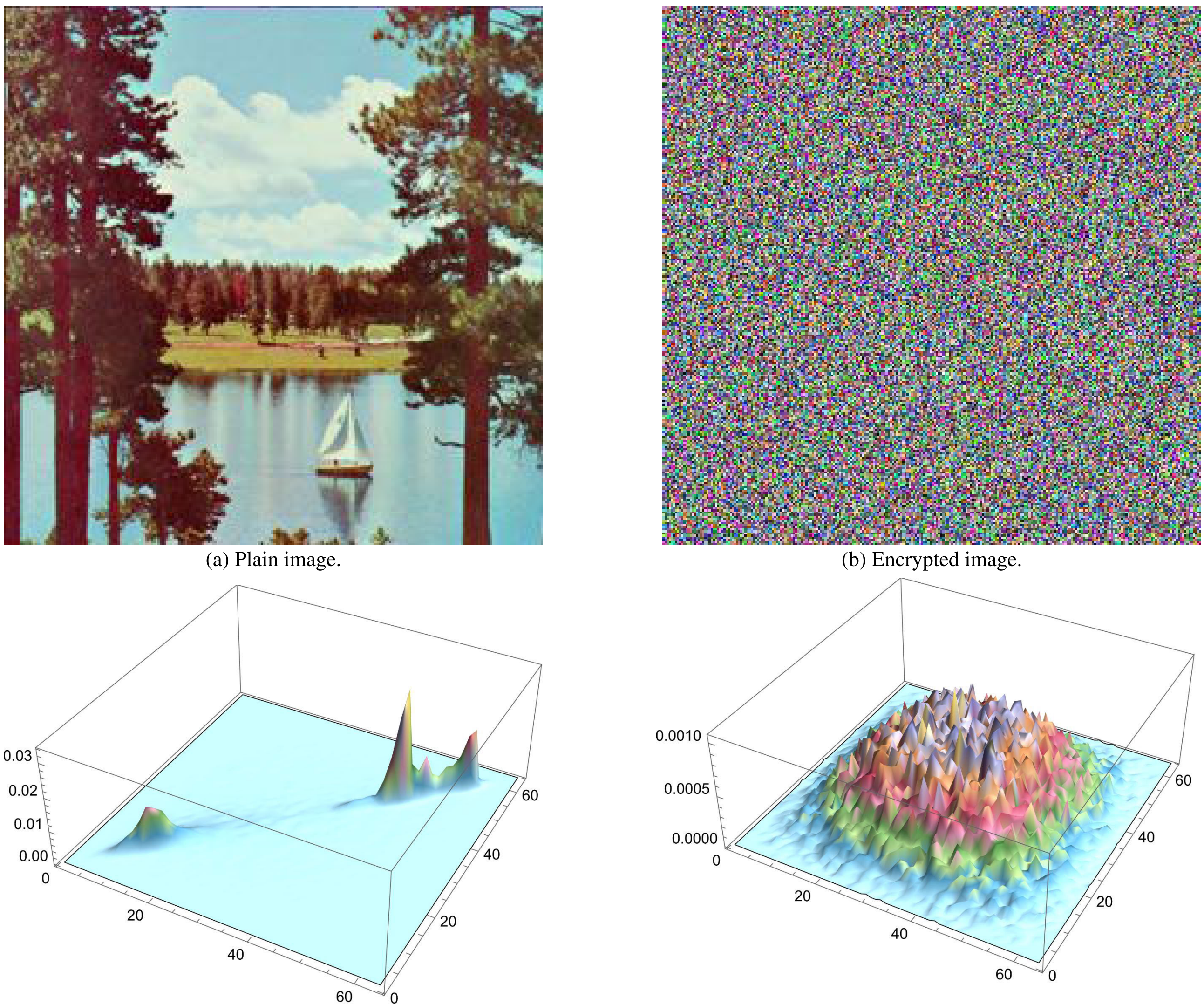
(d) Encrypted image DFT.

FIGURE 18. Tree image and DFT comparison pre- and post-encryption.

TABLE 11. Comparison between correlation coefficients of plain and encrypted images.

	Plain Image			Encrypted Image		
	Correlation Coefficient			Correlation Coefficient		
Image	Horizontal	Diagonal	Vertical	Horizontal	Diagonal	Vertical
Lena	0.938611	0.913175	0.96833	0.00112809	0.00166795	0.0033841
Peppers	0.959422	0.930426	0.966795	−0.0021352	−0.0037003	−0.011349
Mandrill	0.848778	0.750624	0.79088	−0.0017954	0.0031452	0.0020511
House	0.978232	0.936044	0.952926	0.00442727	0.0042474	0.008562
House2	0.907075	0.850782	0.923091	0.0042833	0.00400797	0.00045403
Girl	0.974013	0.951471	0.965671	−0.0034344	0.0041862	0.00018418
Sailboat	0.952381	0.919872	0.950138	0.00281799	0.0024538	−0.002807
Tree	0.968153	0.929967	0.94515	0.00308483	−0.0080665	0.0015145
Average	0.939355	0.897095	0.932873	0.00058474	0.00099272	0.00030635

technique. For illustrative purposes, Table 15 demonstrates the numerical results of a such an analysis as performed on a 256 × 256 image of Lena. All of the computed values in Table 15 are exceeding 0.01 as a proof that the proposed



(a) Plain image.

(b) Encrypted image.

(c) 3D plot of the pixel co-occurrence matrix of the plain image.

(d) 3D plot of the pixel co-occurrence matrix of the encrypted image.

FIGURE 19. Sailboat image and 3D plot of the pixel co-occurrence matrix comparison pre- and post-encryption.

TABLE 12. Comparison of correlation coefficients of the encrypted Lena image among various schemes from the literature.

Scheme	Horizontal	Diagonal	Vertical
Proposed	0.00112809	0.00166795	0.0033841
[2]	0.0064113	−0.0015143	0.000568333
[5]	0.002287	−0.00132	−0.00160
[7]	0.0079784	−0.00012531	0.0011584
[15]	0.00144	−0.00151	0.00795
[21]	0.003265	−0.00413	0.002451
[33]	−0.0061	−0.0018	0.0067
[36]	0.0054	0.0054	0.0016
[54]	0.000199	0.003705	−0.000924

image encryption technique fulfills the NIST analysis benchmark requirements.

J. KEY SPACE ANALYSIS

The proposed image encryption technique is formed by combining 3 stages consecutively. Each of these stages contains a large set of variables. These variables are:

1) Stage 1: Chen system key diffusion.
   a) Number of operations: an integer.
   b) Base: an integer.
   c) Operations selections seed: an integer.
   d) The set of seeds: 13 real numbers.
2) Stage 2: Memristor parallel base-*n* S-box application.
   a) Number of parallel S-boxes: an integer m.
   b) Base: an integer.
   c) Operations selections seed: an integer.
   d) The set of seeds: $m \times 15$ real numbers.
3) Stage 3: Chua system key diffusion.
   a) Number of operations: an integer.
   b) Base: an integer.
   c) Operations selections seed: an integer.
   d) The set of seeds: 10 real numbers.

Cumulatively, there are 9 integer values, 23 real numbers, in addition to 15 real numbers for each S-box. Hence, for

TABLE 13. NPCR and UACI of various images.

Metric	Image	Result
NPCR	Lena	99.6272
	Peppers	99.5967
	Mandrill	99.6231
	House	99.6089
	House2	99.6206
	Sailboat	99.586
	Tree	99.6094
	Girl	99.5977
	Average	99.6087
UACI	Lena	30.3258
	Peppers	32.1276
	Mandrill	29.4504
	House	29.433
	House2	30.6862
	Sailboat	32.0633
	Tree	31.9328
	Girl	35.3936
	Average	31.4266

TABLE 14. Comparison of the average NPCR and UACI of the Lena image among various schemes from the literature.

Scheme	NPCR	UACI
Proposed	99.6272	30.3258
[2]	99.5855	30.3873
[5]	99.63	30.3432
[15]	99.62463	30.56810
[21]	99.65	30.4567
[33]	99.61	33.5160
[36]	99.52	26.7933

TABLE 15. NIST analysis of the data bit-stream of an encrypted Lena image of dimensions 256 × 256.

Test Name	p-value	Remarks
Frequency	0.301427	Success
Block Frequency	0.444596	Success
Runs	0.971420	Success
Longest run of ones	0.286803	Success
Rank	0.686914	Success
Spectral FFT	0.575147	Success
Non overlapping T.M.	0.719975	Success
Overlapping T.M.	0.429492	Success
Maurer's Universal	0.253374	Success
Linear complexity	0.379766	Success
Serial	0.655061	Success
Approx. entropy	0.221568	Success
Cum. sums forward	0.369008	Success
Cum. sums reverse	0.496751	Success
Random ex. 1	0.282195	Success
Random ex. 2	0.444486	Success
Random ex. 3	0.837840	Success
Random ex. 4	0.729269	Success
Random ex. 5	0.671262	Success
Random ex. 6	0.809745	Success
Random ex. 7	0.553379	Success
Random ex. 8	0.527259	Success
Random ex. var. 1	0.900795	Success
Random ex. var. 2	0.731239	Success
Random ex. var. 3	0.756366	Success
Random ex. var. 4	0.674977	Success
Random ex. var. 5	0.518936	Success
Random ex. var. 6	0.599128	Success
Random ex. var. 7	0.588620	Success
Random ex. var. 8	0.564595	Success
Random ex. var. 9	0.545396	Success
Random ex. var. 10	0.449744	Success
Random ex. var. 11	0.834085	Success
Random ex. var. 12	0.871108	Success
Random ex. var. 13	0.890930	Success
Random ex. var 14	0.903743	Success
Random ex. var. 15	0.862494	Success
Random ex. var. 16	0.781999	Success
Random ex. var. 17	0.719528	Success
Random ex. var. 18	0.713889	Success

TABLE 16. Key space values comparison.

Scheme	Key space
Proposed	∞
[2]	$10^{499} \approx 2^{1658}$
[5]	$10^{128} \approx 2^{425}$
[7]	$10^{244} \approx 2^{744}$
[15]	$10^{167} \approx 2^{554}$
[21]	$10^{112} \approx 2^{372}$
[26]	$10^{35} \approx 2^{116}$
[30]	$10^{66} \approx 2^{219}$
[35]	∞
[42]	$10^{300} \approx 2^{996}$
[54]	$10^{56} \approx 2^{187}$

a single S-box: 9 integers and 38 real numbers. This means that for 10 S-boxes (as in Section VI): 9 integers and 173 real numbers. In theory, this translates into an infinite key space. For comparative purposes, Table 16 provides key space values for various algorithms from the literature. The only work attaining a similarly large key space is that of [34]. The attained key space effectively means that the proposed image encryption technique is resistant to brute-force attacks.

K. HISTOGRAM DEPENDENCY TESTS

On the histogram level, correlation between a plain and an encrypted image is evaluated by applying a linear dependency test between the histograms of the images before and after encryption [20]. For a dependency coefficient ranging between -1 and 1, 0 is the preferred value for a well-performing encryption technique. The main cause of preferring 0 as a result is that a 0 signifies no dependency, a while 1 signifies a strong dependency. On the other hand, a -1 signifies strong inverse dependency. To perform this evaluation, 5 different linear correlation evaluation techniques are applied in this research work: Blomqvist β, Goodman-Kruskal γ, Kendall τ, Spearman ρ, and Pearson correlation r.

With respect to the medians of the distributions, Blomqvist assesses the correlation between 2 histogram distributions (X and Y) as a medial correlation coefficient (for medians $\bar{x}$ and $\bar{y}$). Blomqvist correlation is equated as follows:

$$\beta = \{(X - \bar{x})(Y - \bar{y}) > 0\} - \{(X - \bar{x})(Y - \bar{y}) < 0\}. \quad (45)$$

The assessment of the Goodman-Kruskal, which presents a pairwise measure of monotonic association, is based on the relative order of consecutive elements in a given pair of histograms. Pairs of elements of both distributions are either promoting or inhibiting the linear correlation, which are counted. Next, the overall evaluation is based upon these

TABLE 17. Histogram dependency tests for various images, in separate color channels and combined RGB.

Image	Color	β (45)	γ (46)	τ (47)	ρ (48)	r (49)
Lena	R	0.0275056	0.0662174	0.064069	0.0949286	0.075699
	G	−0.0625	−0.0451215	−0.0445375	−0.0637035	−0.0798137
	B	−0.0078745	−0.0044192	−0.0041629	−0.0025057	0.0213273
	RGB	0.0510818	0.00377896	0.00375828	0.0015246	0.00769476
Peppers	R	0.0320204	0.0377274	0.0369005	0.0540886	0.0515456
	G	0.019725	−0.0078883	−0.007803	−0.0099396	0.0328761
	B	−0.0514905	−0.0300921	−0.0296082	−0.0438476	−0.0400148
	RGB	−0.0629941	−0.0266129	−0.026449	−0.0369429	−0.0195293
Mandrill	R	−0.015625	−0.0649895	−0.0643068	−0.0987726	−0.0987726
	G	−0.047247	−0.0237717	−0.02334	−0.0343666	−0.0248794
	B	0.0199727	0.00275138	0.00272362	0.0022772	0.00243885
	RGB	0.0236235	−0.0461824	−0.0459304	−0.0674272	−0.0627675
House	R	0.0866195	0.125008	0.121654	0.180534	0.0575833
	G	−0.109375	−0.0986904	−0.0975915	−0.140344	−0.153031
	B	0.00789068	0.03845	0.0368305	0.0574686	0.0672775
	RGB	0.141178	0.0637533	0.0633851	0.0973743	0.011198
House2	R	−0.046875	−0.0356827	−0.0351172	−0.0526773	−0.0550206
	G	−0.03125	−0.0364932	−0.0361378	−0.0528161	−0.0307434
	B	0.00410356	0.0323847	0.0318583	0.0486502	0.0488306
	RGB	−0.0555783	−0.0265048	−0.026365	−0.0365267	−0.0160983
Sailboat	R	−0.0196469	−0.0097551	−0.0093728	−0.0125805	0.0084438
	G	0	0.0106499	0.0105482	0.0144764	0.0376543
	B	−0.03125	0.00512595	0.00505942	0.0104705	0.0820164
	RGB	−0.015625	−0.0708974	−0.0704962	−0.106562	−0.0929196
Tree	R	0.071149	0.0316351	0.0310719	0.0479491	0.0189841
	G	0.00396081	0.0182702	0.0180021	0.0261575	0.0281026
	B	0.158795	0.167422	0.161098	0.232229	0.165729
	RGB	0.09375	0.125613	0.124842	0.185802	0.21756
Girl	R	−0.0013045	−0.0065005	−0.0054828	−0.0073383	−0.0279095
	G	−0.0152091	0.0250861	0.0207607	0.0268033	0.02566
	B	−0.0655474	−0.0753534	−0.0609495	−0.0842475	−0.100816
	RGB	0	−0.0535714	−0.0511755	−0.0762662	−0.0656636

TABLE 18. Execution time of the proposed image encryption technique, at various image dimensions.

Image Dimensions	t_{Enc} [s]	t_{Dec} [s]	t_{Add} [s]
64 × 64	0.04	0.05	0.09
128 × 128	0.15	0.14	0.29
256 × 256	0.44	0.48	0.92
512 × 512	1.65	1.63	3.28
1024 × 1024	6.28	6.24	12.52

TABLE 19. Encryption time comparison of the Lena image of dimensions 256 × 256.

Scheme	Time [s]	Machine specifications (CPU and RAM)
Proposed	0.44	3.3 GHz AMD® Ryzen 9 5900HX, 32 GB
AES	0.71	3.3 GHz AMD® Ryzen 9 5900HX, 32 GB
[5]	2.582389	2.9 GHz Intel® Core™ i9, 32 GB
[21]	1.42545	2.9 GHz Intel® Core™ i9, 32 GB
[22]	1.1168	3.4 GHz Intel® Core™ i7, 8 GB
[30]	3.45	N/A
[64]	1.112	3.4 GHz Intel® Core™ i3, 4 GB
[15]	3.0019	3.4 GHz Intel® Core™ i7, 8 GB

2 counts (namely n_c and n_d). Goodman-Kruskal correlation is equated as:

$$\gamma = \frac{n_c - n_d}{n_c + n_d}. \tag{46}$$

Kendall correlation evaluation relates the sample size to the counts of concordant pairs and discordant pairs, equating the correlation as:

$$\tau = \frac{n_c - n_d}{\frac{n(n-1)}{2}}. \tag{47}$$

Spearman, as a rank correlation test, compares the sorted positions of elements forming the histogram to the mean rank. Spearman rank correlation is equated as:

$$\rho = \frac{\sum(R_{ix} - \overline{R}_x)(R_{iy} - \overline{R}_y)}{\sqrt{\sum(R_{ix} - \overline{R}_x)^2 \sum(R_{iy} - \overline{R}_y)^2}}. \tag{48}$$

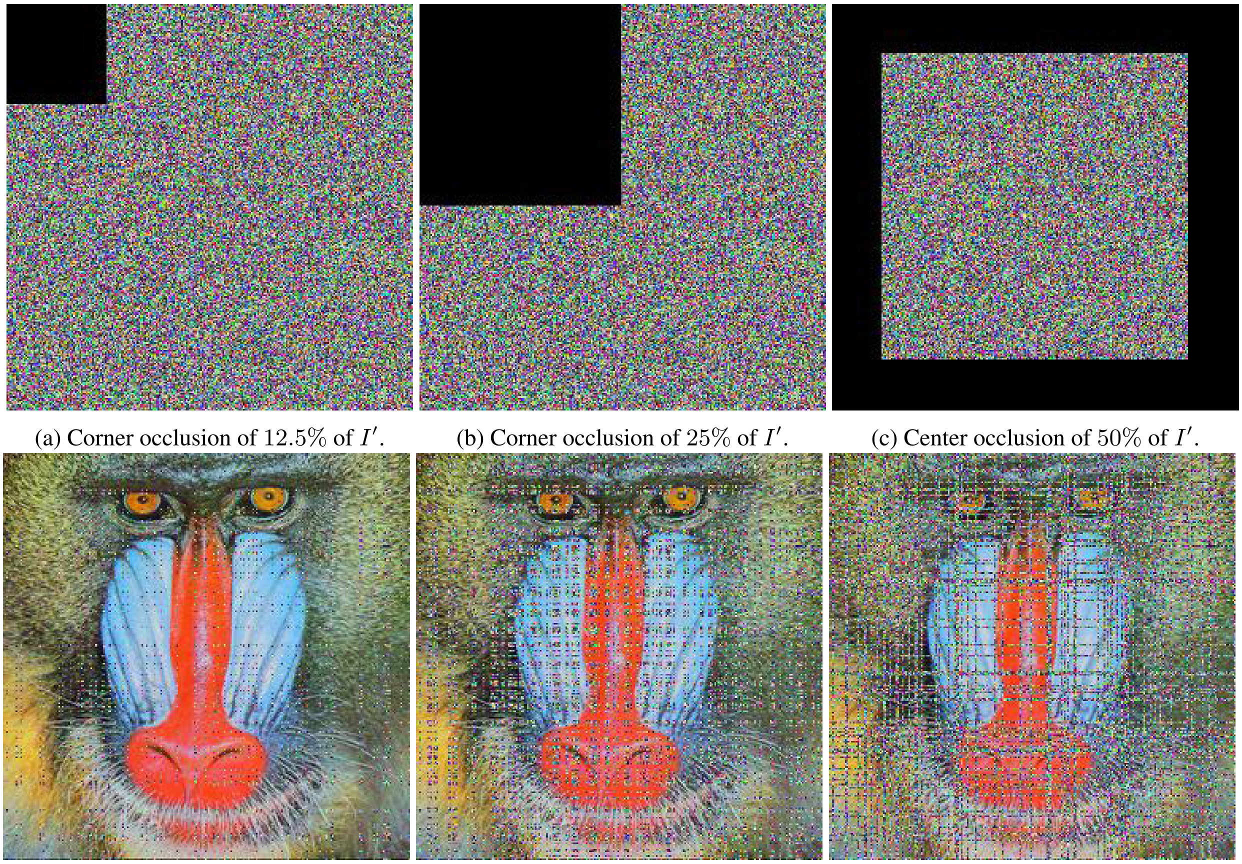
Last but not least, as the most common evaluation mechanism, Pearson correlation simply compares values of the distributions to the mean averages of these distributions. It is equated as:

$$r = \frac{\sum(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum(X_i - \overline{X})^2 \sum(Y_i - \overline{Y})^2}}. \tag{49}$$

The evaluation results of the 5 tests for various test images are displayed in Table 17. Since all scores are approaching 0, there is a clear lack of correlation between the input and encrypted versions of the images in terms of histograms for all colour channels.

L. EXECUTION TIME ANALYSIS

Table 18 shows the execution time of the proposed image encryption technique at various image dimensions. While the



(a) Corner occlusion of 12.5% of I'.

(b) Corner occlusion of 25% of I'.

(c) Center occlusion of 50% of I'.

(d) Corner occlusion of 12.5% of I_c.

(e) Corner occlusion of 25% of I_c.

(f) Center occlusion of 50% of I_c.

FIGURE 20. Three instances of occlusion attacks on encrypted images {(a) ,(b),(c)}, and their decrypted versions {(d),(e),(f)}.

time increases in relation to increases in image dimensions. The proposed image encryption technique is shown to be very efficient, encrypting images at a rate of 4.01 Mbps. Table 19 carries out a comparative analysis among a number of image encryption algorithms from the literature, as well as the AES, and the proposed image encryption technique. It is clear that the proposed technique outperforms its counterparts, including the AES, irrespective of the computing environment. This in turn makes it a favorable technique of choice for small portable devices of limited computing abilities.

M. FURTHER TESTS ON RESISTIVITY TO ATTACKS

This subsection attempts to showcase the ability of the proposed image encryption technique to resist various types of attacks, including occlusion attacks, salt and pepper attacks, as well as Gaussian noise attacks. Furthermore, it provides a short discussion on linear attacks and the resistivity of the proposed technique to them.

An occlusion attack on an encrypted image involves an adversary blocking out or obscuring a contiguous region of the encrypted image before decryption by setting those pixel values to black or a uniform value. This effectively erases the original pixel values in that area, resulting in loss of image content when decrypted. The location, shape, and size of the occluded region can vary depending on the acceptable degradation aimed for by the attacker. Occlusion destroys the pixel values in the blocked area, unlike noise addition which just distorts the values. This attack is effective against spatial encryption schemes that shuffle or scramble pixels, while frequency domain encryption has some inherent occlusion resistance. Countermeasures include redundancy, encryption diffusion techniques, reconstruction algorithms, and authentication to detect tampering. Since the proposed algorithm involves multiple stages of confusion and diffusion, such an attack is shown in Fig. 20 to be ineffective. It is clear in Fig. 20 that even by increasing the occluded area to be 50% of the encrypted image, its decrypted versions are nevertheless still identifiable as those of the Mandrill image.

Salt and pepper attacks are a type of image manipulation attack that can be performed on encrypted images. In this type of attack, the attacker adds random pixels to the encrypted image (salt) or removes pixels from the encrypted image (pepper) in order to modify the image in a way that is not detectable by the encryption algorithm. This can result in the image being distorted or corrupted, making it difficult or impossible to decipher. Salt and pepper attacks can be

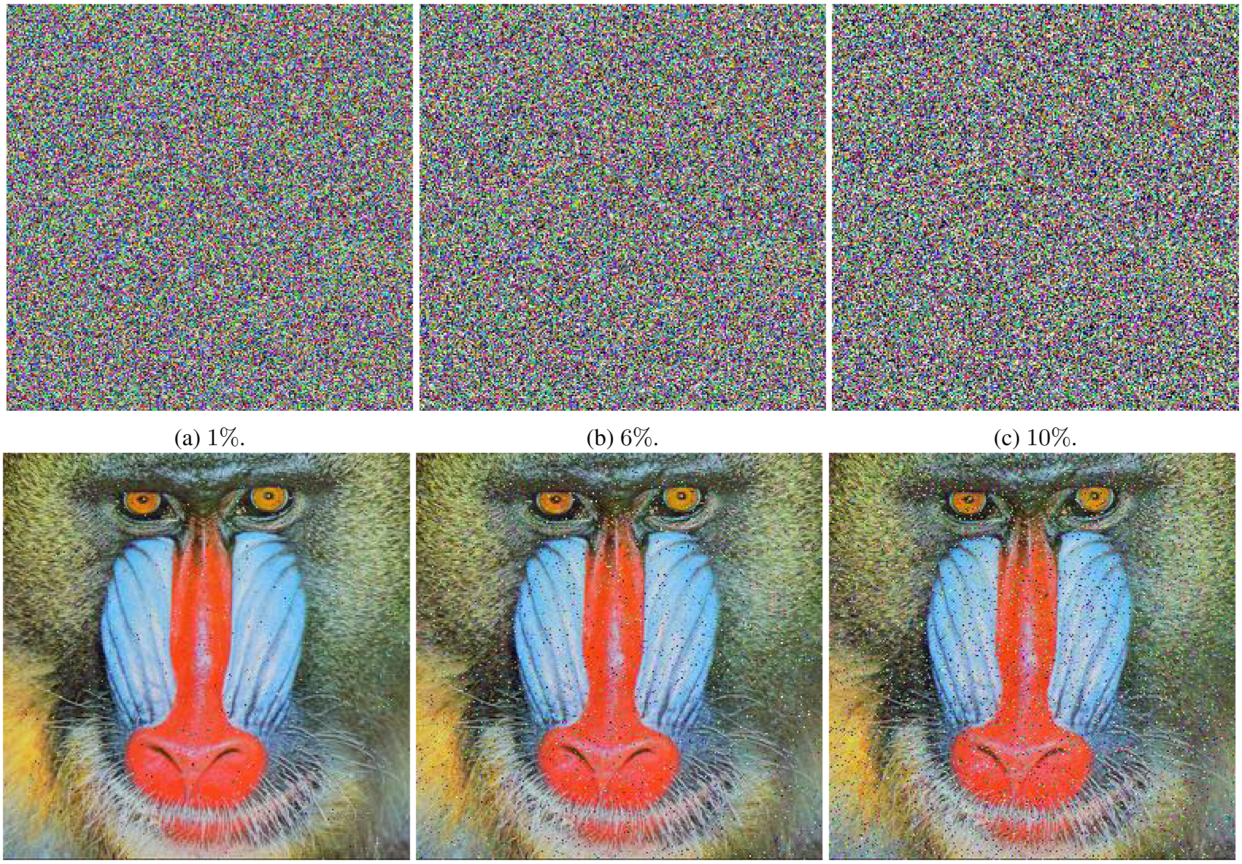

(a) 1%.
(b) 6%.
(c) 10%.
(d) 1%.
(e) 6%.
(f) 10%.

FIGURE 21. The effect of various instances of salt and pepper noise attacks on encrypted images {(a) ,(b),(c)}, and their decrypted counterparts {(d),(e),(f)}.

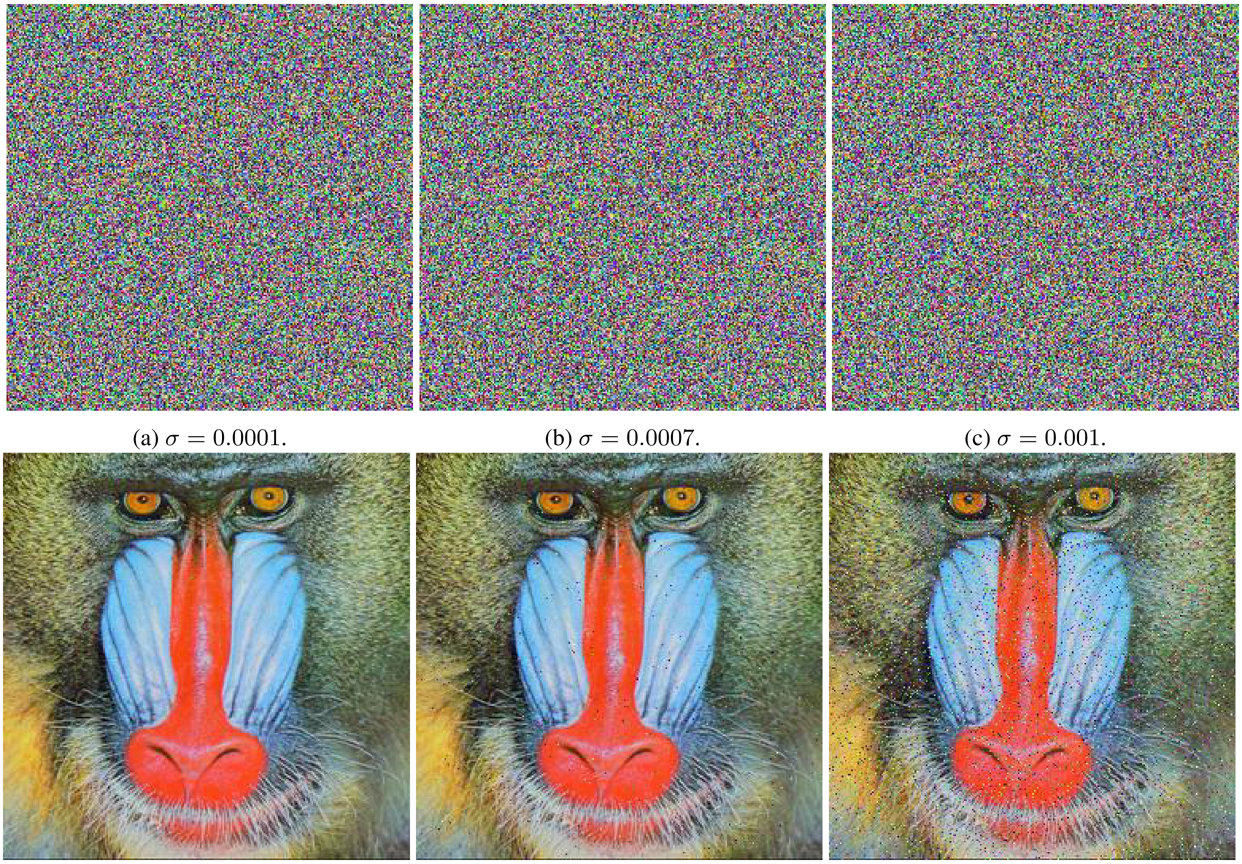
particularly effective against weak encryption algorithms or when the encryption key is compromised. To prevent salt and pepper attacks, stronger encryption algorithms and secure key management practices should be used. It is clear in Fig. 21 that even by increasing the strength of the attack from 1% to 10% the decrypted images are still identifiable as the Mandrill image.

Gaussian noise attacks are another type of image manipulation attack that can be performed on encrypted images. In this type of attack, the attacker adds random noise to the encrypted image, following a Gaussian distribution. The noise added to the image can be subtle and difficult to detect, but can significantly impact the quality and integrity of the image. Gaussian noise attacks can also be used to weaken the encryption algorithm, making it easier for the attacker to decipher the image. To prevent Gaussian noise attacks, secure encryption algorithms with strong key management practices should be used. Additionally, image processing techniques such as denoising can be used to remove any noise added to the encrypted image. It is clear in Fig. 22 that even by increasing the strength of the attack from $\sigma = 0.0001$ to $\sigma = 0.001$ the decrypted images are still identifiable as the Mandrill image.

Linear attacks on image encryption algorithms are a class of attacks that exploit the linear properties of an encryption algorithm to recover the original image or the encryption key. In a linear attack, the attacker constructs a set of linear equations using the plaintext and ciphertext pairs, and then solves for the encryption key. This type of attack is particularly effective against weak encryption algorithms that have linear properties.

One example of a linear attack is the known-plaintext attack, in which the attacker has access to both the original image and its encrypted version. The attacker then constructs a set of linear equations using the plaintext and ciphertext pairs and solves for the encryption key. Another example is the chosen-plaintext attack, in which the attacker can choose the plaintext to be encrypted and obtain the corresponding ciphertext. The attacker can then use these pairs to construct linear equations and solve for the encryption key.

To prevent linear attacks, encryption algorithms should be designed to have strong non-linear properties. Non-linear encryption algorithms make it difficult for attackers to construct linear equations and solve for the encryption key. Since the proposed image encryption technique involves the construction and application of S-boxes, which are highly



(a) $\sigma = 0.0001$. (b) $\sigma = 0.0007$. (c) $\sigma = 0.001$.

(d) $\sigma = 0.0001$. (e) $\sigma = 0.0007$. (f) $\sigma = 0.001$.

FIGURE 22. The effect of zero-mean Gaussian noise attacks on different standard deviations σ on encrypted images as well as the effect on their decrypted counterparts.

non-linear in nature, it becomes impossible for any sort of cryptanalysis effort utilizing linear attacks to be successful.

VII. CONCLUSION AND FUTURE WORK

This research work identified the prime importance of both key and S-box generation from PRNGs, then capitalized on this knowledge by proposing a multi-stage symmetric image encryption technique. Solutions of the Chen, Chua and Memristor hyperchaotic systems of fractional-order were employed to generate the needed PRNGs for this research work. In the first and third stages, an encryption key is generated and employed to carry out data diffusion. However, unlike conventional image cryptosystems, multiple logical and arithmetic operations were made use of while applying the key to the image data. In the second encryption stage, multiple S-boxes of various dimensions were generated and employed in a parallel fashion to carry out data confusion. This is unlike conventional image cryptosystems where a single S-box, in most cases of dimensions 16×16 is made use of. Next, a large number of performance evaluation metrics were computed and their values compared to the state-of-the-art. For many of the metrics, the achieved values were shown to be superior to their counterparts. The adoption of fractional-order systems was shown to be most beneficial in relation to the key space. Since having a very large number of control variables allows for a key space of theoretically infinite length.

A limitation of this work is that since the generated S-boxes were of various dimensions, traditional S-box evaluation metrics were no longer viable to apply in such a novel case. A future work could make use of the ideas proposed in this research work, still applying various S-boxes in a parallel fashion, however, utilizing only S-boxes of standard dimensions. This would allow for the computation of S-box performance evaluation metrics, as in [7], [20], and [45], as well as for comparison with other well-established S-boxes in the literature.

REFERENCES

[1] S. H. AbdElHaleem, S. K. Abd-El-Hafiz, and A. G. Radwan, "A generalized framework for elliptic curves based PRNG and its utilization in image encryption," *Sci. Rep.*, vol. 12, no. 1, p. 13278, Aug. 2022.

[2] W. Alexan, N. Alexan, and M. Gabr, "Multiple-layer image encryption utilizing fractional-order Chen hyperchaotic map and cryptographically secure PRNGs," *Fractal Fractional*, vol. 7, no. 4, p. 287, Mar. 2023.

[3] W. Alexan, A. Ashraf, E. Mamdouh, S. Mohamed, and M. Moustafa, "IoMT security: SHA3–512, AES-256, RSA and LSB steganography," in *Proc. 8th NAFOSTED Conf. Inf. Comput. Sci. (NICS)*, Dec. 2021, pp. 177–181.

[4] W. Alexan, M. ElBeltagy, and A. Aboshousha, "Image encryption through Lucas sequence, S-box and chaos theory," in *Proc. 8th NAFOSTED Conf. Inf. Comput. Sci. (NICS)*, Dec. 2021, pp. 77–83.

[5] W. Alexan, M. ElBeltagy, and A. Aboshousha, "RGB image encryption through cellular automata, S-box and the Lorenz system," *Symmetry*, vol. 14, no. 3, p. 443, Feb. 2022.

[6] W. Alexan, A. Elkhateeb, E. Mamdouh, F. Al-Seba'Ey, Z. Amr, and H. Khalil, "Utilization of corner filters, AES and LSB steganography for secure message transmission," in *Proc. Int. Conf. Microelectron. (ICM)*, Dec. 2021, pp. 29–33.

[7] W. Alexan, M. Gabr, E. Mamdouh, R. Elias, and A. Aboshousha, "Color image cryptosystem based on sine chaotic map, 4D Chen hyperchaotic map of fractional-order and hybrid DNA coding," *IEEE Access*, vol. 11, pp. 54928–54956, 2023.

[8] D. R. Anderson, *Model Based Inference in the Life Sciences: A Primer on Evidence*, vol. 31. Cham, Switzerland: Springer, 2008.

[9] N. A. Azam, U. Hayat, and M. Ayub, "A substitution box generator, its analysis, and applications in image encryption," *Signal Process.*, vol. 187, Oct. 2021, Art. no. 108144.

[10] Y.-R. Bai, D. Baleanu, and G.-C. Wu, "A novel shuffling technique based on fractional chaotic maps," *Optik*, vol. 168, pp. 553–562, Sep. 2018.

[11] J.-J. Chen, D.-W. Yan, S.-K. Duan, and L.-D. Wang, "Memristor-based hyper-chaotic circuit for image encryption," *Chin. Phys. B*, vol. 29, no. 11, Nov. 2020, Art. no. 110504.

[12] W. El-Shafai, I. M. Almomani, and A. Alkhayer, "Optical bit-plane-based 3D-JST cryptography algorithm with cascaded 2D-FrFT encryption for efficient and secure HEVC communication," *IEEE Access*, vol. 9, pp. 35004–35026, 2021.

[13] M. ElBeltagy, W. Alexan, A. Elkhamry, M. Moustafa, and H. H. Hussein, "Image encryption through Rössler system, PRNG S-box and Recamán's sequence," in *Proc. IEEE 12th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2022, pp. 0716–0722.

[14] M. T. Elkandoz and W. Alexan, "Image encryption based on a combination of multiple chaotic maps," *Multimedia Tools Appl.*, vol. 81, no. 18, pp. 25497–25518, Jul. 2022.

[15] S. Farrag and W. Alexan, "Secure 3D data hiding technique based on a mesh traversal algorithm," *Multimedia Tools Appl.*, vol. 79, nos. 39–40, pp. 29289–29303, Oct. 2020.

[16] A. Fedorov, L. Steffen, M. Baur, M. P. da Silva, and A. Wallraff, "Implementation of a Toffoli gate with superconducting circuits," *Nature*, vol. 481, no. 7380, pp. 170–172, Jan. 2012.

[17] M. Gabr, W. Alexan, and K. Moussa, "Image encryption through CA, chaos and Lucas sequence based S-box," in *Proc. Signal Process., Algorithms, Archit., Arrangements, Appl. (SPA)*, Sep. 2022, pp. 34–39.

[18] M. Gabr, W. Alexan, K. Moussa, B. Maged, and A. Mezar, "Multi-stage RGB image encryption," in *Proc. Int. Telecommun. Conf. (ITC-Egypt)*, Jul. 2022, pp. 1–6.

[19] M. Gabr, H. H. Hussein, and W. Alexan, "A combination of decimal- and bit-level secure multimedia transmission," in *Proc. Workshop Microw. Theory Techn. Wireless Commun. (MTTW)*, Oct. 2022, pp. 177–182.

[20] M. Gabr, H. Younis, M. Ibrahim, S. Alajmy, I. Khalid, E. Azab, R. Elias, and W. Alexan, "Application of DNA coding, the Lorenz differential equations and a variation of the logistic map in a multi-stage cryptosystem," *Symmetry*, vol. 14, no. 12, p. 2559, Dec. 2022.

[21] L. Gong, K. Qiu, C. Deng, and N. Zhou, "An image compression and encryption algorithm based on chaotic system and compressive sensing," *Opt. Laser Technol.*, vol. 115, pp. 257–267, Jul. 2019.

[22] U. Hayat, N. A. Azam, H. R. Gallegos-Ruiz, S. Naz, and L. Batool, "A truly dynamic substitution box generator for block ciphers based on elliptic curves over finite rings," *Arabian J. Sci. Eng.*, vol. 46, no. 9, pp. 8887–8899, Sep. 2021.

[23] A. S. Hegazi and A. E. Matouk, "Dynamical behaviors and synchronization in the fractional order hyperchaotic Chen system," *Appl. Math. Lett.*, vol. 24, no. 11, pp. 1938–1944, Nov. 2011.

[24] K. M. Hosny, *Multimedia Security Using Chaotic Maps: Principles and Methodologies*, vol. 884. Cham, Switzerland: Springer, 2020.

[25] K. M. Hosny, S. T. Kamal, and M. M. Darwish, "A color image encryption technique using block scrambling and chaos," *Multimedia Tools Appl.*, vol. 81, pp. 505–525, Sep. 2022.

[26] K. M. Hosny, S. T. Kamal, and M. M. Darwish, "Novel encryption for color images using fractional-order hyperchaotic system," *J. Ambient Intell. Humanized Comput.*, vol. 13, no. 2, pp. 973–988, Feb. 2022.

[27] K. M. Hosny, S. T. Kamal, and M. M. Darwish, "A novel color image encryption based on fractional shifted Gegenbauer moments and 2D logistic-sine map," *Vis. Comput.*, vol. 39, no. 3, pp. 1027–1044, Mar. 2023.

[28] J. Hou, R. Xi, P. Liu, and T. Liu, "The switching fractional order chaotic system and its application to image encryption," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 2, pp. 381–388, Apr. 2017.

[29] X. Hu, L. Wei, W. Chen, Q. Chen, and Y. Guo, "Color image encryption algorithm based on dynamic chaos and matrix convolution," *IEEE Access*, vol. 8, pp. 12452–12466, 2020.

[30] H. H. Hussein, W. Alexan, M. ElBeltagy, and A. Aboshousha, "Visual data security incorporating Fibonacci sequence, S-box, and chaos theory," in *Proc. Int. Conf. Smart Syst. Power Manage. (ICSPM)*, Nov. 2022, pp. 85–90.

[31] B. Idrees, S. Zafar, T. Rashid, and W. Gao, "Image encryption algorithm using S-box and dynamic Hénon bit level permutation," *Multimedia Tools Appl.*, vol. 79, nos. 9–10, pp. 6135–6162, Mar. 2020.

[32] N. Iqbal, R. A. Naqvi, M. Atif, M. A. Khan, M. Hanif, S. Abbas, and D. Hussain, "On the image encryption algorithm based on the chaotic system, DNA encoding, and castle," *IEEE Access*, vol. 9, pp. 118253–118270, 2021.

[33] B. Jasra and A. H. Moon, "Color image encryption and authentication using dynamic DNA encoding and hyper chaotic system," *Exp. Syst. Appl.*, vol. 206, Nov. 2022, Art. no. 117861.

[34] K. C. Jithin and S. Sankar, "Colour image encryption algorithm combining Arnold map, DNA sequence operation, and a Mandelbrot set," *J. Inf. Secur. Appl.*, vol. 50, Feb. 2020, Art. no. 102428.

[35] M. Khan and F. Masood, "A novel chaotic image encryption technique based on multiple discrete dynamical maps," *Multimedia Tools Appl.*, vol. 78, no. 18, pp. 26203–26222, Sep. 2019.

[36] M. Khan and T. Shah, "An efficient chaotic image encryption scheme," *Neural Comput. Appl.*, vol. 26, no. 5, pp. 1137–1148, Jul. 2015.

[37] M. Kumari and S. Gupta, "Performance comparison between chaos and quantum-chaos based image encryption techniques," *Multimedia Tools Appl.*, vol. 80, no. 24, pp. 33213–33255, Oct. 2021.

[38] C. P. Li, W. H. Deng, and D. Xu, "Chaos synchronization of the Chua system with a fractional order," *Phys. A, Stat. Mech. Appl.*, vol. 360, no. 2, pp. 171–185, Feb. 2006.

[39] H. Liu, B. Zhao, and L. Huang, "Quantum image encryption scheme using Arnold transform and S-box scrambling," *Entropy*, vol. 21, no. 4, p. 343, Mar. 2019.

[40] Z. Liu, C. Wu, J. Wang, and Y. Hu, "A color image encryption using dynamic DNA and 4-D memristive hyper-chaos," *IEEE Access*, vol. 7, pp. 78367–78378, 2019.

[41] P. Mani, R. Rajan, L. Shanmugam, and Y. Hoon Joo, "Adaptive control for fractional order induced chaotic fuzzy cellular neural networks and its application to image encryption," *Inf. Sci.*, vol. 491, pp. 74–89, Jul. 2019.

[42] F. Masood, J. Masood, L. Zhang, S. S. Jamal, W. Boulila, S. U. Rehman, F. A. Khan, and J. Ahmad, "A new color image encryption technique using DNA computing and chaos-based substitution box," *Soft Comput.*, vol. 26, pp. 7461–7477, Dec. 2021.

[43] M. Messadi, K. Kemih, L. Moysis, and C. Volos, "A new 4D memristor chaotic system: Analysis and implementation," *Integration*, vol. 88, pp. 91–100, Jan. 2023.

[44] J. J. Montesinos-García and R. Martinez-Guerra, "Colour image encryption via fractional chaotic state estimation," *IET Image Process.*, vol. 12, no. 10, pp. 1913–1920, Oct. 2018.

[45] F. Özkaynak, "An analysis and generation toolbox for chaotic substitution boxes: A case study based on chaotic labyrinth Rene Thomas system," *Iranian J. Sci. Technol., Trans. Electr. Eng.*, vol. 44, no. 1, pp. 89–98, Mar. 2020.

[46] S. Vinay, A. Pujar, H. Kedlaya, and V. S. Shahapur, "Implementation of DNA cryptography based on dynamic DNA sequence table using cloud computing," *Int. J. Eng. Res. Technol.*, vol. 7, no. 8, Jun. 2019.

[47] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.

[48] Y. Si, H. Liu, and Y. Chen, "Constructing a 3D exponential hyperchaotic map with application to PRNG," *Int. J. Bifurcation Chaos*, vol. 32, no. 7, Jun. 2022, Art. no. 2250095.

[49] N. Sun, C.-T. Li, H. Chan, B. D. Le, M. Z. Islam, L. Y. Zhang, M. R. Islam, and W. Armstrong, "Defining security requirements with the common criteria: Applications, adoptions, and challenges," *IEEE Access*, vol. 10, pp. 44756–44777, 2022.

[50] M. S. Tavazoei, "Fractional order chaotic systems: History, achievements, applications, and future challenges," *Eur. Phys. J. Special Topics*, vol. 229, nos. 6–7, pp. 887–904, Mar. 2020.

[51] N. H. UbaidurRahman, C. Balamurugan, and R. Mariappan, "A novel string matrix data structure for DNA encoding algorithm," *Proc. Comput. Sci.*, vol. 46, pp. 820–832, 2015.

[52] I. Ullah, N. A. Azam, and U. Hayat, "Efficient and secure substitution box and random number generators over mordell elliptic curves," *J. Inf. Secur. Appl.*, vol. 56, Feb. 2021, Art. no. 102619.

[53] Y. Wang, C. Wu, S. Kang, Q. Wang, and V. Mikulovich, "Multi-channel chaotic encryption algorithm for color image based on DNA coding," *Multimedia Tools Appl.*, vol. 79, pp. 18317–18342, Mar. 2020.

[54] D. Wei and M. Jiang, "A fast image encryption algorithm based on parallel compressive sensing and DNA sequence," *Optik*, vol. 238, Jul. 2021, Art. no. 166748.

[55] Z. Yan, "Controlling hyperchaos in the new hyperchaotic Chen system," *Appl. Math. Comput.*, vol. 168, no. 2, pp. 1239–1250, Sep. 2005.

[56] S. Yasser, A. Hesham, M. Hassan, and W. Alexan, "AES-secured bit-cycling steganography in sliced 3D images," in *Proc. Int. Conf. Innov. Trends Commun. Comput. Eng. (ITCE)*, Feb. 2020, pp. 227–231.

[57] E. Yavuz, "A new parallel processing architecture for accelerating image encryption based on chaos," *J. Inf. Secur. Appl.*, vol. 63, Dec. 2021, Art. no. 103056.

[58] I. Younas and M. Khan, "A new efficient digital image encryption based on inverse left almost semi group and Lorenz chaotic system," *Entropy*, vol. 20, no. 12, p. 913, Nov. 2018.

[59] F. Yu, L. Li, B. He, L. Liu, S. Qian, Z. Zhang, H. Shen, S. Cai, and Y. Li, "Pseudorandom number generator based on a 5D hyperchaotic four-wing memristive system and its FPGA implementation," *Eur. Phys. J. Special Topics*, vol. 230, nos. 7–8, pp. 1763–1772, Aug. 2021.

[60] F. Yu, Z. Zhang, H. Shen, Y. Huang, S. Cai, and S. Du, "FPGA implementation and image encryption application of a new PRNG based on a memristive Hopfield neural network with a special activation gradient," *Chin. Phys. B*, vol. 31, no. 2, Jan. 2022, Art. no. 020505.

[61] J. Yu, W. Xie, Z. Zhong, and H. Wang, "Image encryption algorithm based on hyperchaotic system and a new DNA sequence operation," *Chaos, Solitons Fractals*, vol. 162, Sep. 2022, Art. no. 112456.

[62] X. Zhang, X. Fan, J. Wang, and Z. Zhao, "A chaos-based image encryption scheme using 2D rectangular transform and dependent substitution," *Multimedia Tools Appl.*, vol. 75, no. 4, pp. 1745–1763, Feb. 2016.

[63] X. Zhang, L. Wang, Y. Wang, Y. Niu, and Y. Li, "An image encryption algorithm based on hyperchaotic system and variable-step Josephus problem," *Int. J. Opt.*, vol. 2020, pp. 1–15, Oct. 2020.

MOHAMED GABR (Member, IEEE) was born in Cairo, Egypt, in 1989. He received the B.Sc., M.Sc., and Ph.D. degrees in computer science and engineering from German University in Cairo (GUC), Egypt, in 2011, 2013, and 2023, respectively.

He has been with the Computer Science and Engineering Department, since 2011. He is teaching various courses in relation to computer vision, artificial intelligence, compilers, theory of computation, and computer graphics. He is the author or coauthor of various journal articles and conference papers. His research interests include computer vision and information security.

RIMON ELIAS (Senior Member, IEEE) received the M.C.S. and Ph.D. degrees in computer science from the University of Ottawa, Ottawa, ON, Canada, in 1999 and 2004, respectively.

He is the author of *Digital Media* (Springer) and *Modeling of Environments* (Lambert Academic Publishing); and the several book chapters, encyclopedia, journals, and conference papers. His interests embrace different image-related fields, including computer vision, image processing, computer graphics, and visualization.

KHALID M. HOSNY (Senior Member, IEEE) was born in Zagazig, Egypt, in 1966. He received the B.Sc., M.Sc., and Ph.D. degrees from Zagazig University, Egypt, in 1988, 1994, and 2000, respectively. From 1997 to 1999, he was a Visiting Scholar with the University of Michigan, Ann Arbor, MI, USA, and the University of Cincinnati, Cincinnati, OH, USA. He is currently a Professor of information technology with the Faculty of Computers and Informatics, Zagazig University. He has published four edited books and more than 150 papers in international journals. His research interests include image processing, pattern recognition, multimedia, and computer vision. He is a Senior Member of ACM. He is an editor and a scientific reviewer of more than 60 international journals. He was among the top 2% of scientists according to the Stanford ranking from 2020 to 2022.

GEORGE A. PAPAKOSTAS received the Diploma, M.Sc., and Ph.D. degrees in electrical and computer engineering from the Democritus University of Thrace (DUTH), Greece, in 1999, 2002, and 2007, respectively. He has 15 years of experience in large-scale systems design, as a senior software engineer and a technical manager. Currently, he is the Head of the Machine Learning and Vision (MLV) Research Group. He is also a Tenured Full Professor with the Department of Computer Science, International Hellenic University, Greece. He has (co) authored more than 200 publications in indexed journals, international conferences, book chapters, one book (in Greek), two edited books, and six journal special issues. His publications have over 3000 citations with an H-index of 32 (Google Scholar). His research interests include machine learning, computer/machine vision, pattern recognition, and computational intelligence. He is a member of ACM, IAENG, MIR Laboratories, EUCogIII, and the Technical Chamber of Greece (TEE). He served as a reviewer for numerous journals and conferences.

WASSIM ALEXAN (Senior Member, IEEE) was born in Alexandria, Egypt, in 1987. He received the B.Sc., M.Sc., and Ph.D. degrees in communications engineering and the M.B.A. degree from German University in Cairo (GUC), Egypt, in 2010, 2012, 2017, and 2019, respectively.

He was with the Mathematics Department, from 2010 to 2017. Since 2017, he has been an Assistant Professor with the Faculty of Information Engineering and Technology, GUC, teaching various courses in relation to wireless communications, modulation and coding, digital logic design, circuit theory, and mathematics. He is also an Adjunct Assistant Professor with the Mathematics Department, German International University (GIU), New Administrative Capital, Egypt, since its inception in 2019. He is the author or coauthor of more than 70 journal articles and conference papers. His research interests include wireless communications, information security, and image and signal processing.

Dr. Alexan is a member of ACM. He received the Best Paper Award at the 19th IEEE Conference on Signal Processing Algorithms, Architectures, Arrangements, and Applications (SPA'2015), Poznan, Poland; the AEG Writer of the Year Award from American University in Cairo (AUC), Egypt, in 2019; and the Best Poster Award at the 37th IEEE National Radio Science Conference, Cairo, Egypt, in 2020.