IET Image Processing

WILEY

The Institution of Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

A DNA-Dynamic Permutation-Diffusion Algorithm for Image Encryption Using Scaling Chaotification Models and Advanced DNA Operations

Mustafa Kamil Khairullah[1] | Mohd Zafri Bin Baharuddin[2] | Reema Thabit[3] | Mohammad Ahmed Alomari[4] | Gamal Alkawsi[5] | Faten A. Saif[6]

[1]Institute of Sustainable Energy (ISE), Universiti Tenaga Nasional, Selangor, Malaysia | [2]College of Engineering, Universiti Tenaga Nasional, Selangor, Malaysia | [3]School of Computer Science, Faculty of Innovation and Technology, Taylor's University, Selangor, Malaysia | [4]Fakulti Teknologi dan Kejuruteraan Elektronik dan Komputer (FTKEK), Universiti Teknikal Malaysia Melaka (UTeM), Melaka, Malaysia | [5]Institute of Informatics and Computing in Energy, Universiti Tenaga Nasional, Kajang, Malaysia | [6]Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Serdang, Malaysia

Correspondence: Gamal Alkawsi (gamalalkawsi@tu.edu.ye)

Received: 6 April 2025 | **Revised:** 1 July 2025 | **Accepted:** 4 August 2025

Funding: The authors received no specific funding for this work.

ABSTRACT

The rise in cyber threats to digital images over networks is a primary problem for both private and government organisations. Image encryption is considered a useful way to secure the digital image; however, it faces critical challenges such as weak key generation, chosen-plaintext attacks, high overhead, and scalability. To overcome these challenges, this paper proposes the DNA-Dynamic Concurrent Permutation-Diffusion Algorithm (DNA-DCP-DA), which introduces four advanced encryption mechanisms. Firstly, new scaling chaotification models are introduced to enhance chaotic properties, achieving superior results in bifurcation, Lyapunov Exponent (LE), Sample Entropy (SEn), Kolmogorov Entropy (KEn) and key generation. Secondly, a Key Vectorisation Method (KVM) is proposed to optimise execution time and reduce the computational overhead of chaotic map iterations. Thirdly, robust non-commutative DNA operations are introduced, including DNA hybrid and circular shift operations to enhance encryption security. Finally, integrate permutation and dynamic diffusion processes, strengthening security and improving efficiency. To evaluate the proposed algorithm, extensive experiments have been conducted, and results have been compared with the latest encryption algorithms. This shows the proposed encryption algorithm is better, with superior results for correlation results close to zero and Information Entropy (IE) larger than 7.999. The Number of Pixel Change Rates (NPCR) exceeds 99.6%, and the Uniform Average Change Intensity (UACI) is above 33.4%. The algorithm encrypts an image of size 256 × 256 in 0.1255 s, with a key space reaching 2^{697}. As a result, the proposed system establishes a new benchmark for secure and efficient image encryption against cyber threats.

1 | Introduction

1.1 | Background

Cyber threats to digital images are escalating, with the global cost of cybercrime projected to surge from $9.22 trillion in 2024 to $13.82 trillion by 2028 [1]. Image encryption is crucial for mitigating these risks, ensuring image confidentiality, integrity, and privacy against unauthorised access, data breaches, manipulation and other cyber threats in different applications, such as IoT devices. [2, 3]. In response, numerous encryption algorithms have been proposed, based on diverse theoretical

foundations such as chaos theory [4], fuzzy theory [5] and S-box theory [6].

Chaos theory, exemplified by chaotic maps, is especially valuable in encryption due to its properties of extreme sensitivity to parameters, high pseudo-randomness and ergodicity, making chaotic systems a primary choice for encryption and steganography [7–9]. Chaotic maps in encryption are typically classified as one-dimensional (1D) or multi-dimensional. 1D chaotic maps are straightforward to implement in software and hardware [10–13]. Nonetheless, they have limitations, such as a limited chaotic range, low Lyapunov Exponent (LE), periodic windows and insufficient pseudo-randomness [10]. Multi-dimensional chaotic maps, in contrast, exhibit greater chaotic complexity and random pseudo-sequences but require more complex and costly hardware/software implementation, as they only reach chaotic states after extensive operations.

Chaotification models provide a general framework to maximise the chaotic behaviour of dynamical systems [14–16]. In general, chaotification can generate a chaotic state within a system that is originally nonchaotic. It is also applied to chaotic maps to further enhance their properties, making these maps more secure, effective and suitable for secure cryptographic applications. Therefore, chaotification models are strong candidates for addressing the limitations of chaotic maps. Despite the effectiveness of the chaotification models, they can suffer from some drawbacks, such as periodic windows in parameter spaces, low LE and inadequate randomness, which can compromise their application in cryptography and security. Moreover, the chaotic maps generated from chaotification models involve multiple complex mathematical operations; thus, extensive iterations of chaotic map-based key generators result in high computational overhead. Few studies are available that discuss the time consumption of chaotification models, which is quite significant in enhancing efficiency [17].

Chaotic image encryption schemes typically consist of two phases: confusion and diffusion [18–22]. In the confusion phase, pixel locations are chaotically rearranged according to chaotic sequences to disrupt the correlation between adjacent pixels. At this stage, the histogram remains unchanged, as pixel values are not modified. As a result, the confusion phase alone is vulnerable to statistical attacks. In the diffusion phase, pixel values are altered by directly applying a chaotic sequence, transforming the confused image into an incomprehensible, diffused image.

Alongside chaotic encryption techniques, DNA encryption techniques have emerged with advantages including low power consumption and massive parallelism [23–25]. DNA image encryption generally consists of three main stages: DNA encoding, DNA operations and DNA decoding [26–29]. The first stage converts the original image into a DNA sequence based on specific encoding rules. In the second stage, DNA operations are applied to manipulate the DNA sequence. Finally, the third stage decodes the processed DNA sequence back into a binary format according to decoding rules.

Integrating chaotic maps and DNA technology offers a promising strategy for enhancing both security and efficiency of image encryption [30, 31]. However, this integration can suffer from certain disadvantages, such as a lack of resistance to chosen-plaintext/ciphertext attacks. The vulnerability to chosen-plaintext/ciphertext attacks is heightened by the limited types of existing DNA operations. Most systems rely on a few fixed DNA operations (e.g., XOR, XNOR, subtraction and addition), which can weaken the cryptosystem against attacks if either the ciphertext or the key is exposed [32, 33]. Moreover, most DNA-based image cryptosystems typically perform permutation and diffusion in separate steps, which weakens security [34]. When the image is uniformly black or white, security relies entirely on diffusion, leaving the system vulnerable. In that case, attackers can use chosen plaintext attacks to bypass permutation by targeting the diffusion phase directly.

In addition to these security challenges, efficiency issues also persist, as many existing schemes struggle to balance robust protection with low computational cost. This highlights the importance of developing encryption algorithms that can simultaneously deliver high security, strong resistance to cryptanalysis and practical performance suitable for real-time applications.

1.2 | Related Work

Recently, researchers have focused on developing 1D chaotic maps and chaotification models to generate higher-quality chaos while addressing the trade-off between security and efficiency. These advancements in chaotic strategies directly contribute to improving the security of image encryption systems. In [35], a technique was introduced that uses the tent map sequence as input for the Chebyshev map, producing pseudo-random sequences effective in encryption systems. Similarly, [36] presented a model combining multiple 1D chaotic maps, resulting in increased randomness. The study in [37] proposed a chaotification model using buffeting and modulo operations to strengthen chaos in existing maps. The work in [38] introduced an improved chaotic map based on cascading methods, which demonstrated effective chaotic behaviour but required substantial computational resources. Another technique includes [39], which developed a fractional sine chaotification model (FSCM) that raised chaos performance, though at a high implementation complexity. In [40], a cosine transformation-based chaotification framework was proposed. Although the chaotification framework comes with good performance, the performance gradually decreases when the parameter is close to zero. Further, [41] suggested a 1D enhancement framework combining exponential, logarithmic and sine functions to improve chaotic effects. The work in [42] adapted the logistic map with perturbations to overcome its inherent limitations, achieving satisfactory chaotic outcomes.

In addition to enhancements in 1D chaotic maps, 2D chaotic maps have also been explored to balance security and efficiency in image encryption. For example, in [43], a novel 2D multiple collapse chaotic map is proposed, leveraging arctangent functions to achieve enhanced chaotic characteristics and improved uniformity in the chaotic sequence distribution. The work in [44] proposes a novel 2D Salomon chaotic map, inspired by the classical Salomon function. This map integrates the cosine and modular functions to significantly enhance the complexity of its behaviour. The study in [45] presents a novel two-dimensional infinite logic map with enhanced chaotic properties, aimed at improving security and performance in cryptographic

applications. In [46], a new 2D chaotic map was proposed by combining the logistic map, sine map and a hyperbolic function. This design aimed to enhance chaotic complexity and balance security with computational efficiency in image encryption applications.

In parallel, considerable efforts have focused on developing DNA encryption techniques. For example, [47] presented an image encryption algorithm combining DNA techniques with spatiotemporal chaos, though authors in [48] found it remained vulnerable to specific attacks. The authors in [49] proposed robust encryption algorithms that resist attacks by utilising DNA techniques and two chaotic maps: the 1D logistic map and the 2D logistic map. However, authors in [50] demonstrated their susceptibility to chosen-plaintext attacks. In [51], a greyscale image cryptosystem based on DNA techniques and a 2D Hénon-Sine map was introduced, but the authors in [52] successfully breached it using chosen-plaintext attacks. In [53], a method combining a hyperchaotic map, Hill cipher, Feistel network and DNA coding technique was proposed, but [54] analysed the algorithm and showed that it could be compromised with a specific chosen-plaintext attack. Another approach in [55] employed dynamic DNA techniques with a 4D memristive hyperchaotic system, applying chaotic sequences and a dynamic confusion-diffusion mechanism to secure images. A novel image encryption scheme using fractional chaotic systems, DNA coding and mutation mechanisms was proposed in [56], demonstrating high randomness and resilience against attacks. In [26], the authors introduced two new DNA operations: the right-circular shift operation and the left-circular shift operation, and combined them with hash functions to enhance security against chosen-plaintext attacks. In [57], an image encryption method using a 4D hyperchaotic system, SHA-512 and DNA operations showed strong security, but the 4D system led to slower encryption speeds. In [32], an encryption method using a hyperchaotic map and a DNA triploid mutation was proposed to solve the limitation of DNA operations. Another approach in [58] used hash functions (MD5, SHA-256) for key shaping, a memristor hyperchaotic system for random sequence generation, Arnold's transform for image scrambling and DNA operations for pixel value modification. The analysis confirmed the algorithm's resistance to various attacks. The work in [33] proposes a new encryption system that utilises new DNA operations and a chaotic map. The DNA operations are proposed to address the limitations of existing DNA operations, while the hyperchaotic map is employed to generate strong key sequences. The authors in [59] developed a chaotic map incorporating DNA strand exchange and DNA strand diffusion to satisfy the confusion and diffusion properties, respectively. The authors in [60] proposed a cryptosystem based on DNA and a hyberchaotic map. A bidirectional spiral transformation mechanism is used to rearrange the image pixels. To implement the dynamic DNA technique, several DNA operations are applied, incorporating both DNA bases and their lowercase forms. In [61], an encryption system using a hyperchaotic map and reversible DNA subtraction was designed to address security issues in traditional DNA operations. While this system demonstrated strong security, it required extensive key sequences from the hyperchaotic map, resulting in slower encryption speeds and high computational overhead. The work in [62] develops an S-Box design that leverages the crossover and mutation operators of genetic algorithms. This design is integrated with a 2D Styblinski–Tang chaotic map to construct a resilient image cryptosystem. A new colour image encryption scheme is introduced in [63] to ensure image security, based on a 5D fractional-order hyper-chaotic map, an extended DNA encoding/decoding scheme and four new DNA operation methods. The algorithm employs a block-based encryption mechanism, dividing the image into sub-blocks for processing. In [64], a colour medical image compression-encryption scheme is proposed, combining compressive sensing and DNA coding operations. This method integrates position scrambling, reduced-stiffness operations and DNA-based pixel-level transformations for the protection of medical imaging data. This work in [65] presents an image encryption algorithm based on a delayed chaotic system and DNA coding with cross-layer techniques. The initial values of the chaotic system are selected based on the image's entropy and correlation to strengthen resistance against chosen-plaintext attacks. The DNA encoding step is jointly applied with cross-layer position rearrangement, followed by a cross-layer diffusion mechanism to complete the encryption.

1.3 | Contribution

This work presents new chaotification models and a new encryption system based on these chaotification models and new DNA operations. The contributions of this study are summarised as follows:

1. Enhance chaotic maps: This study introduces new chaotification models that strengthen chaotic maps by ensuring control parameters cover all parameter spaces. This approach eliminates periodic windows, improves randomness and increases sensitivity, thereby generating secure and unpredictable key sequences.
2. Superior chaotic performance: The proposed maps demonstrate enhanced chaotic properties, including improved LE and increased sensitivity and randomness. Experimental results confirm that these maps outperform traditional chaotic models in chaotic performance metrics.
3. Improved encryption speed: To address the computational overhead of chaotic maps iteration, a Key Vectorisation Method (KVM) is introduced. This innovative approach effectively reduces the number of iterations required by chaotic maps, thereby streamlining the encryption process and substantially decreasing execution time.
4. Concurrent permutation-diffusion process: This paper proposes a new technique to combine permutation and diffusion processes into one integrated step. This integration not only reduces the processing steps but also disables targeted attacks, ensuring that the permutation effect remains active even when a white or black image is selected.
5. New non-commutative DNA operations: This study proposes innovative non-commutative DNA operations designed to manipulate pixel values. These new operations preserve the benefits of traditional DNA operations, such as low computational overhead, while enhancing encryption effectiveness and resistance to attacks.

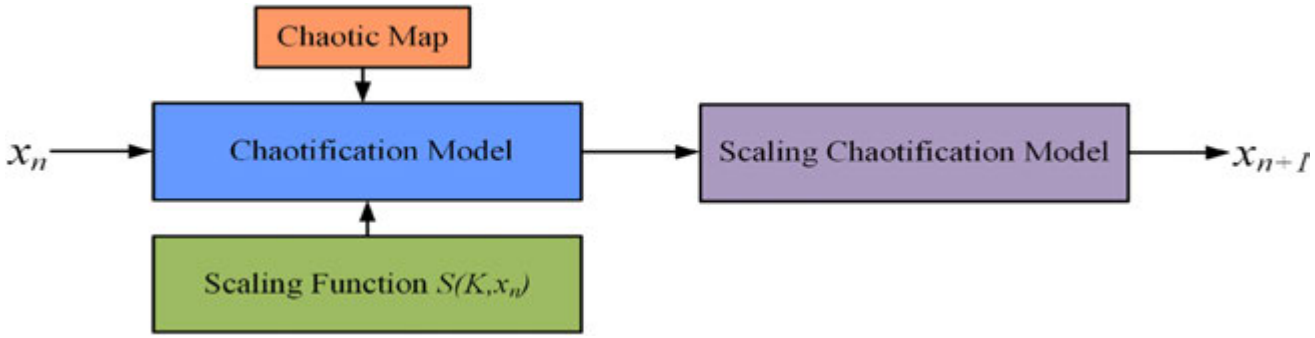

FIGURE 1 | Principal diagram of the scaling chaotification model.

1.4 | Organisation

This paper is structured as follows. Section 2 introduces the proposed chaotification models and details their enhanced chaotic properties. Section 3 describes the new DNA operations that improve encryption performance. Section 4 outlines the proposed image encryption algorithm, while Section 5 presents the performance evaluation results and comparative analyses with existing systems. Finally, Section 6 concludes the paper.

2 | Scaling Chaotification Models

Driven by the aforementioned factors, it is evident that chaotification models are essential to ensure a high Lyapunov index and a broad parameter range that exhibits continuous chaotic behaviour without periodic windows, a phenomenon known as 'robust chaos'. Recently, four chaotification models have been proposed to enhance chaos: the Sine Chaotification Model (SCM) [66], the Modular Chaotification Model (MCM) [67], the Sine Transform-Based Chaotic System (STBCS) [68] and the Cosine Transform-Based Chaotic System (CTBCS) [69]. However, these models exhibit certain limitations, such as a finite chaotic range, periodic behaviour and low random behaviour within specific subsets of their parameters.

This paper proposes new scaling chaotification models by adding a scaling function to the SCM, MCM, STBCS and CTBCS. The scaling function is motivated by scaling and shifting properties in trigonometric functions such as sine and cosine functions, and can be mathematically described as:

$$S(K, \boldsymbol{x_n}) = 2^K \times x_n + \frac{2^K}{x_n} \tag{1}$$

where K is a scaling parameter. The scaling function is a nonlinear equation that scales the input of the sine, cosine and modular functions, aiming to intensify the chaotic behaviour of these functions. The scaling function is designed to induce dual scaling terms: the first, a linear expansion term $(2^K \times x_n)$; the second, a reciprocal expansion term $(\frac{2^K}{x_n})$. This interaction significantly enhances both unpredictability and sensitivity by two cases:

1. First term $(2^K \times x_n)$: a linear expansion that scales proportionally with the current state x_n , enhancing divergence when x_n is moderately valued.
2. Second term $(\frac{2^K}{x_n})$: a reciprocal amplification that dominates when x_n is small, introducing strong nonlinearity and preventing convergence.

This dual-term design ensures that chaotic behaviour is reinforced regardless of whether the input is large or near zero, introducing adaptive scaling across the domain, significantly increasing the system's sensitivity and chaotic behaviour. The effectiveness of these proposed models is evaluated using bifurcation, LE, Sample Entropy (SEn) and Kolmogorov Entropy (KEn) analyses. Figure 1 illustrates the basic structure of the scaling chaotification model.

2.1 | Scaling Sine Chaotification Model (S-SCM)

The SCM is used to enhance the chaotic properties of the available maps by adding the sine transform to the output of the chaotic maps. The mathematical expression of SCM is as follows [66]:

$$x_{n+1} = \sin(\pi \times C(a, x_n)) \tag{2}$$

where a illustrates the control parameter, $C(a, x_n)$ illustrates one of the existing chaotic maps. The new scaling chaotification model of SCM, named S-SCM, is generated by adding the scaling function to the SCM. The mathematical expression of S-SCM is

$$x_{n+1} = \sin(\pi \times (C(a, x_n) \times S(K, x_n))) \tag{3}$$

where a is in the range of $(0, \infty)$. The output sequence (x_n) is bounded between the range of [-1, 1] by the sine function.

2.2 | Scaling Modular Chaotification Model (S-MCM)

The MCM is a kind of chaotification framework used for the purpose of improving the chaotic complexity of the current maps [67]. The MCM is defined as follows:

$$x_{n+1} = C(a, x_n) \bmod P \tag{4}$$

where a is the controlling parameter of MCM and mod represents modular function. The scaling representation of MCM, namely

TABLE 1 | Chaotic maps.

Input (Seed) map	Chaotification model	Generated map	Map equation
Logistic map (LM)	SCM	SCM-LM	$x_{n+1} = \sin(\pi \times a \times x_n \times (1 - x_n))$
	S-SCM	S-SCM-LM	$x_{n+1} = \sin\left(\pi \times (a \times x_n \times (1 - x_n)) \times 2^K \times x_n + \frac{2^K}{x_n}\right)$
Sine map (SM)	SCM	SCM-SM	$x_{n+1} = \sin(\pi \times a \times \sin(\pi \times x_n))$
	S-SCM	S-SCM-SM	$x_{n+1} = \sin\left(\pi \times ((a \times \sin(\pi \times x_n))) \times 2^K \times x_n + \frac{2^K}{x_n}\right)$
Logistic map (LM)	MCM	MCM-LM	$x_{n+1} = (a \times x_n \times (1 - x_n)) \bmod 3$
	S-MCM	S-MCM-LM	$x_{n+1} = \left((a \times x_n \times (1 - x_n)) \times 2^K \times x_n + \frac{2^K}{x_n}\right) \bmod 3$
Sine map (SM)	MCM	MCM-SM	$x_{n+1} = (\mathrm{a} \times \sin(\pi \times x_n)) \bmod 3$
	S-MCM	S-MCM-SM	$x_{n+1} = \left((\mathrm{a} \times \sin(\pi \times x_n)) \times 2^K \times x_n + \frac{2^K}{x_n}\right) \bmod 3$
Logistic map (LM) and sine map (SM)	STBCS	STBCS-LSM	$x_{n+1} = \sin(\pi \times (4 \times a_1 \times x_n \times (1 - x_n) + a_2 \times \sin(\pi \times x_n)))$
	S-STBCS	S-STBCS-LSM	$x_{n+1} = \sin\left(\pi \times (a_1 \times x_n \times (1 - x_n) + a_2 \times \sin(\pi \times x_n)) \times 2^K \times x_n + \frac{2^K}{x_n}\right)$
Logistic map (LM) and sine map (SM)	CTBCS	CTBCS-LSM	$x_{n+1} = \cos(\pi \times (4 \times a_1 \times x_n \times (1 - x_n) + a_2 \times \sin(\pi \times x_n) - 0.5))$
	S-CTBCS	S-CTBCS-LSM	$x_{n+1} = \cos\left(\pi \times (a_1 \times x_n \times (1 - x_n) + a_2 \times \sin(\pi \times x_n)) \times 2^K \times x_n + \frac{2^K}{x_n}\right)$

S-MCM, can be defined in the following expression:

$$x_{n+1} = (C(a, x_n) \times S(K, x_n)) \bmod P \tag{5}$$

where a is in the range of $(0, \infty)$ and $P \in n+$. The output sequence (x_n) is in the $[0, P]$ range.

2.3 | Scaling Sine Transform Based Chaotic System (S-STBCS)

The STBCS is a general framework that combines two chaotic maps' outputs and then applies sine transform to the result of the combination [68]. The mathematical expression of STBCS is as follows:

$$x_{n+1} = \sin(\pi \times (C_1(a_1, x_n) + C_2(a_2, x_n))) \tag{6}$$

where $C_1(a, x_n)$ and $C_2(a, x_n)$ are two existing maps, a_1 and a_2 represent control parameters. After adding the scaling function to the STBCS, the new scaling STBCS named S-STBCS is defined as follows:

$$x_{n+1} = \sin(\pi \times ((C_1(a_1, x_n)) + (C_2(a_2, x_n))) \times (S(K, x_n))) \tag{7}$$

$C_1(a, x_n)$ and $C_2(a, x_n)$ are two seed maps, a_1 and a_2 are control parameters in the range of $(0, +\infty)$. x_n is the output sequence.

2.4 | Scaling Cosine Transform Based Chaotic System (S-CTBCS)

Similar to STBCS, the CTBCS is a general framework that applies the cosine function to the result of addition between two available maps namely seed maps [69]. The CTBCS equation is as follows:

$$x_{n+1} = \cos(\pi \times (B + C_1(a_1, x_n) + C_2(a_2, x_n))) \tag{8}$$

$C_1(a_1, x_n)$ and $C_2(a_2, x_n)$ are two existing maps. B represents a constant shift. a_1 and a_2 are two parameters. After adding the scaling function to the CTBCS, the scaling CTBCS named S-CTBCS can be defined as:

$$x_{n+1} = \cos(\pi \times ((C_1(a_1, x_n)) + (C_2(a_2, x_n))) \times (S(K, x_n))) \tag{9}$$

$C_1(a, x_n)$ and $C_2(a, x_n)$ represent two seed maps, a_1 and a_2 are their control parameters in the range of $(0, \infty)$. x_n is the output sequence of S-CTBCS.

2.5 | Examples of New Chaotic Maps

Using the proposed models, users can select different 1D chaotic maps as input maps to produce many new chaotic maps. To illustrate the efficiency of the proposed chaotification models, the logistic map (LM) $x_{n+1} = \mathrm{LM}(a, x_n) = a \times x_n \times (1 - x_n)$ and the sine map (SM) $x_{n+1} = \mathrm{SM}(a, x_n) = a \times \mathrm{Sin}(\pi x_n)$ are chosen as seed maps in this subsection. Setting the logistic and sine maps to be seed maps, we can obtain new chaotic maps, their mathematical definitions are listed in Table 1. The proposed models aim to produce chaos that can exhibit chaotic behaviour in the whole range of parameters along with high LE (i.e., extreme chaos complexity). The chaotic maps introduced by the proposed models still have the advantages of 1D chaotic maps and can be easily executed by software/hardware devices.

In the original chaotification models, the output of the seed map has bounded behaviour before being applied to the sine, cosine or modular transformations. The difference between the output values of the seed map is not much, as in a logistic map where the output value is confined to the range of [0, 1]. On the contrary, the proposed models amplify the values of chaotic maps before applying sine, cosine, or modular transformations via the scaling function. To ensure a strong amplification effect, the scaling parameter K is recommended to be within the range [8, 24].

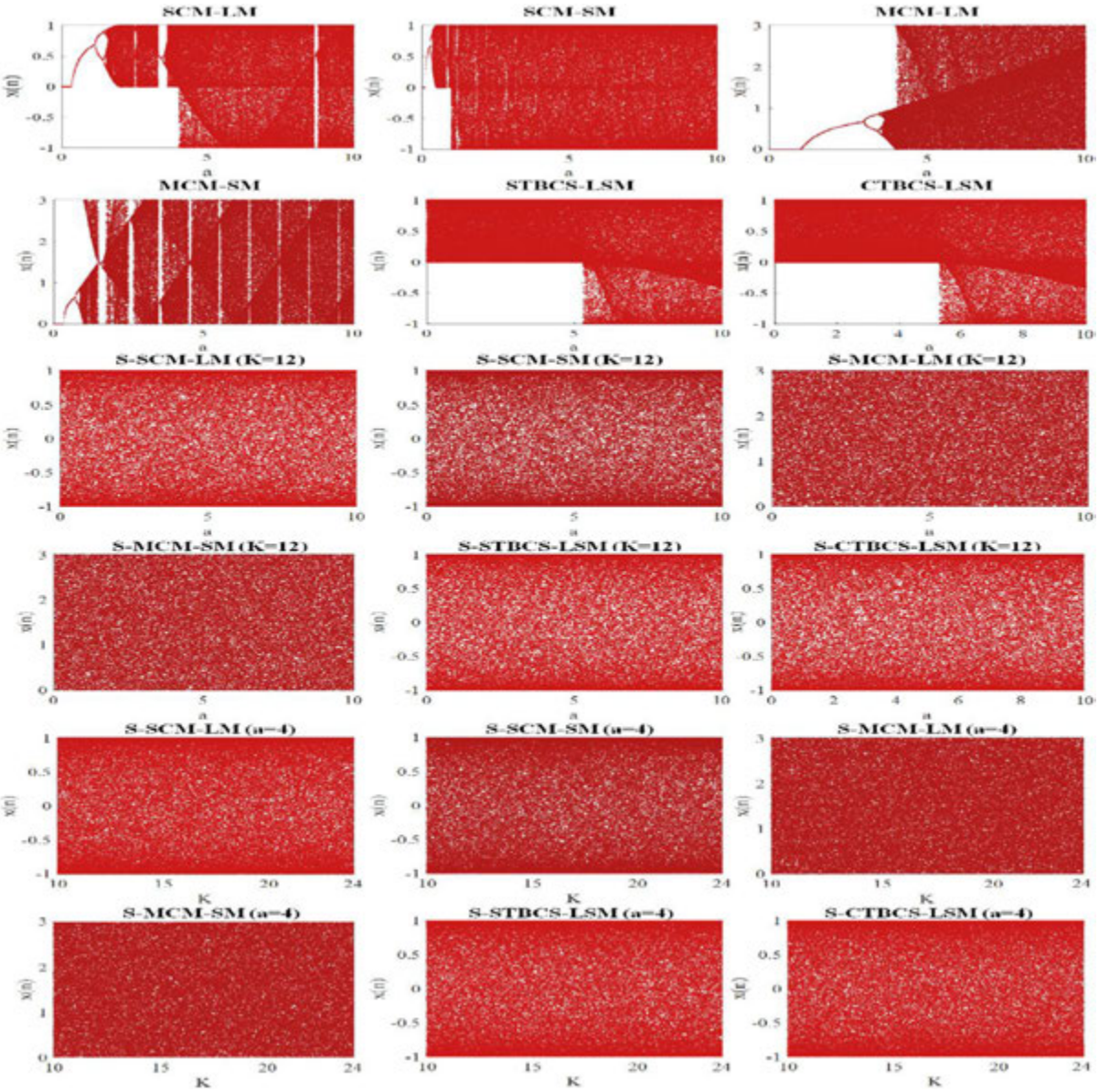

FIGURE 2 | Bifurcation diagrams.

2.6 | Performance Analysis of Chaotic Maps

This section presents an examination of the characteristics of the proposed chaotification models using several evaluation techniques. As suggested in the original work to simplify the performance analysis, for the chaotic maps STBCS-LSM and CTBCS-LSM, the parameters are set as $a_1 = a$ and $a_2 = 1 - a$. The same applies to the proposed maps S-STBCS-LSM and S-CTBCS-LSM. To highlight the strengths of our proposed chaotic maps, we compare them with the Enhanced Logistic Map (ELM) [70] and the Improved Sine Tangent Map (IST) [12].

2.6.1 | Bifurcation Analysis

Bifurcation is a significant factor in determining the dynamic behaviour of chaotic maps [71, 72]. The bifurcation illustrates the generation steps of the chaos state by exhibiting a reflection of the state for chaotic maps. The chaotic performance of chaotic maps is mainly determined by their control parameters. There is a narrow range of chaotic parameter values for several chaotic maps that led to aperiodic behaviour, making the maps enter a nonchaotic state. The periodic windows can be identified using a bifurcation diagram in which the periodic windows are revealed as narrow distinct bands concentrating at specific values. Figure 2 presents the bifurcation diagrams. In comparison with the maps generated by the original chaotification systems, the maps generated by the proposed scaling models show no periodic windows and are uniformly distributed in the [-1,1] range in all parameter spaces. On the contrary, the periodic windows are easy to visually distinguish in the bifurcation diagrams of original chaotification systems' maps. Moreover, the bifurcation diagrams with respect to the scaling parameter K exhibit a complete absence of periodic windows, indicating a consistently chaotic regime across the range from 10 to 24.

Thus, the proposed chaotic maps overcome the problems related to periodic windows. Hence, they show better ergodicity and performance.

2.6.2 | Lyapunov Exponent (LE) Analysis

LE is used to determine whether the map is chaotic or not, where the positive values of the LE determine the chaotic areas of maps and vice versa. Moreover, the larger value of the exponent illustrates a better chaotic effect. LE of a 1D chaotic map can be mathematically defined by the following equation [73]:

$$LE = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x_i)| \tag{10}$$

$f'(x_i)$ represents the time series created by a chaotic map, while n represents the length of the time series. LE is able to quantify the complexity of the chaos as well as assess whether there is chaotic behaviour in a chaotic map. The results are depicted in Figure 3. One can see that the improved models retain positive values across the whole parameter space. On the other hand, the original chaotification models have some negative values across parameter space. Except the maps generated from STBCS, and CTBCS can retrieve a positive LE over all parameter ranges, but are relatively low (i.e., limited chaotic complexity and randomness). The low positive values of the LE even with indicating chaotic states, are still predictable and vulnerable. Additionally, the proposed maps gain much larger values of LE, demonstrating unbreakable robust chaos in the overall values of the parameters even when they are close to zero. The LE with respect to the parameter K is also evaluated in Figure 3. It can be observed that parameter K yields consistently high and positive LE values across the range from 10 to 24. Moreover, a high positive LE with increasing K indicates that the system becomes more chaotic and sensitive to initial conditions, due to the influence of the scaling function. Consequentially, the results confirm that the problem of the Low LE is addressed in the proposed chaotification models.

2.6.3 | Sample Entropy (SEn)

SEn is one of the most effective metrics used to measure the time series' degree of randomness and complexity. It can quantify chaos, unpredictability and disorder within a system by characterising the similarity relations of sequences generated by dynamical systems [74]. SEn is used to verify the complexity of the proposed chaotification model. A higher value of SEn reflects a high level of randomness. The SEn results of different chaotification models are depicted in Figure 4. The results show that the proposed improved chaotic maps clearly outperform other maps in terms of SEn. Our improved maps achieve significantly higher SEn values across the entire parameter range compared to those generated by other strategies. As a consequence, the maps produced by the proposed scaling chaotification models obtain highly complex unpredictable behaviour, beating the defects of low complex behaviour.

2.6.4 | Kolmogorov Entropy (KEn)

KEn is an entropy metric that provides a mathematical measure of complexity and randomness in finite dynamical systems. It quantifies the amount of additional information needed to accurately predict the $(\boldsymbol{t}+1)$th output of a trajectory based on its previous $\boldsymbol{t}$ outputs [75]. A higher KEn value indicates greater unpredictability and complexity in the map's behaviour. The KEn results of different chaotification models are depicted in Figure 5. The results indicate that the improved chaotic maps achieve better KEn values compared to other maps, indicating better unpredictability.

3 | DNA Technique

In this section, DNA bases (the four nucleotides) and the new DNA operations are discussed.

3.1 | DNA Encoding/Decoding Rules

DNA comprises four nucleic acids: adenine (A), cytosine (C), guanine (G) and thymine (T), in which A complements T and so do C and G. According to the binary rule, 1 complements 0, and thus 11 complements 00, as are 01 and 10. By encoding the DNA bases A, C, G and T into binary format 00, 01, 10 and 11, we can get 24 encoding rules. However, there are only 8 DNA encoding types that satisfy the complement rules of Watson–Crick [76, 77]. Additionally, DNA can be represented in the quaternary system in which 0, 1, 2 and 3 can represent A, C, G and T respectively, as well as the complementary rules, where 0 and 3 are complementary and so are 1 and 2 according to rule 1. In the proposed method, every pixel of the input grey image is in the range of [0, 255]. Consequently, we can represent this value as 4 digits in the quaternary system (8 digits in the binary system). After that, encode the 4 digits into a DNA sequence using the listed encoding rules in Table 2. As an example, if a pixel equals 228 in the decimal system, its quaternary form is 3210 (its binary form is 11100100). Then, we can encode this number into 8 rules as follows:

$$TGCA, TCGA, AGCT, ACGT, GTAC, GATC, CTAG, CATG$$

or, equivalently in quaternary form:

$$3210, 3120, 0213, 0123, 2301, 2031, 1302, 1032.$$

DNA decoding is a reverse version of DNA encoding. There are two types of encoding that are commonly used: static encoding and dynamic encoding. In static encoding, the encoding scheme is fixed, so all image pixels are encoded by applying one encoding rule. In dynamic encoding, the image pixels are encoded by applying different encoding rules, in which each pixel has its own encoding rule depending on the encoding rule matrix. This matrix is usually generated by the use of a chaotic map. Dynamic encoding is considered more secure since the choice of encoding rule depends on chaos. For example, suppose a pixel has a DNA value of $TCGA$ (quaternary 3120) and a corresponding chaotic sequence that determines the encoding rules: 2458. The first digit is encoded using Rule 2, the second digit using Rule 4, the third digit using Rule 5 and the last digit using Rule 8. The final result is $TGTG$ (quaternary 3232).

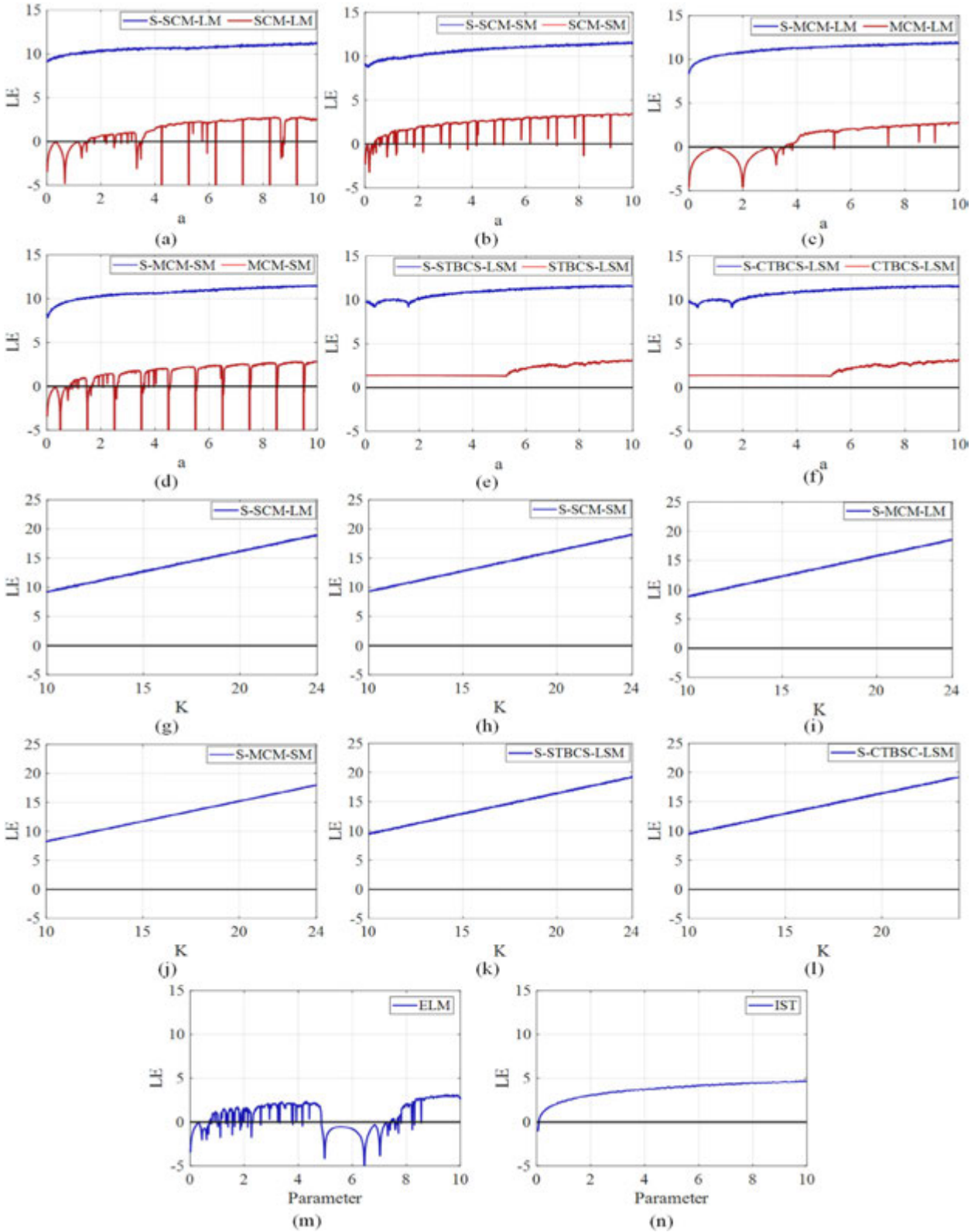

FIGURE 3 | LE of: (a) SCM-LM and S-SCM-LM for K = 12, (b) SCM-SM and S-SCM-SM for K = 12, (c) MCM-LM and S- MCM-LM for K = 12, (d) MCM-SM and S- MCM-SM for K = 12, (e) STBCS-LSM and S- STBCS-LSM for K = 12, (f) CTBCS-LSM and S- CTBCS-LSM for K = 12, (g) S-SCM-LM for a = 4, (h) S-SCM-SM for a = 4, (i) S-MCM-LM for a = 4, (j) S-MCM-SM for a = 4, (k) S- STBCS-LSM for a = 4, (l) S- CTBCS-LSM for a = 4, (m) ELM and (n) IST.

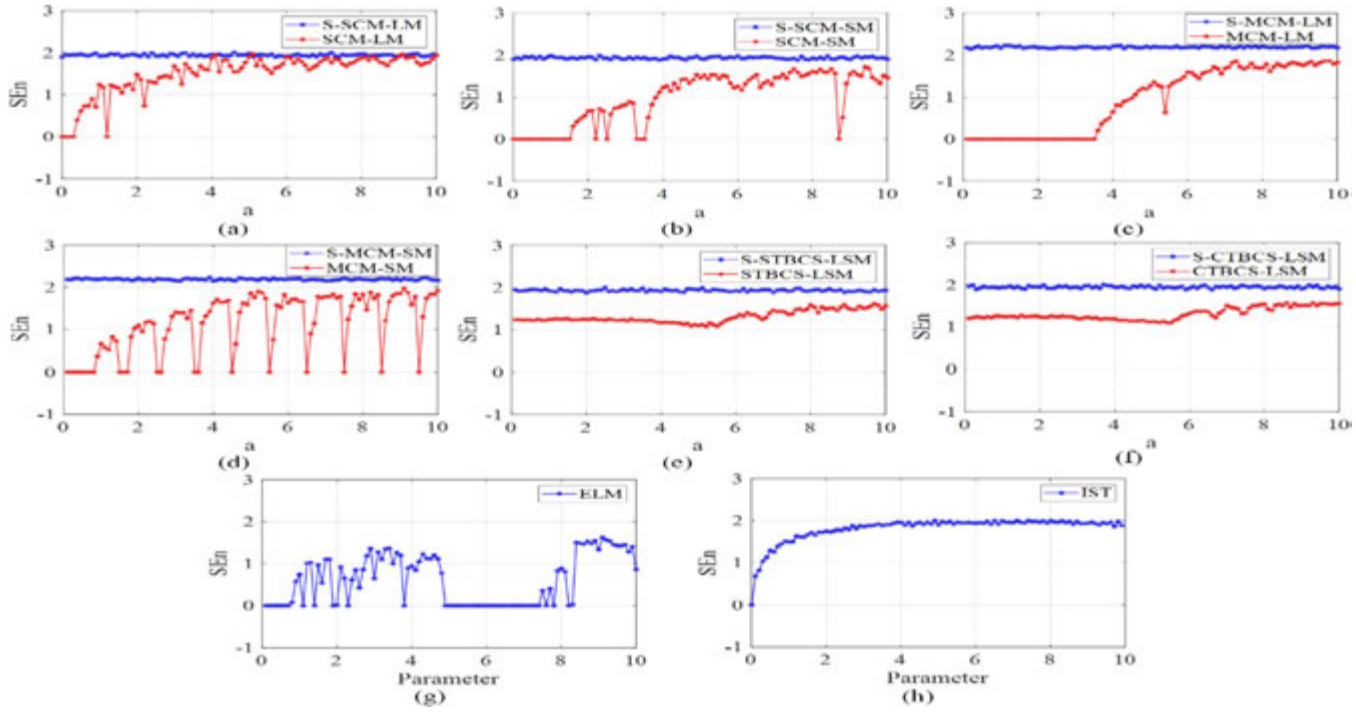

FIGURE 4 | SEn of: (a) SCM-LM and S-SCM-LM for K = 12, (b) SCM-SM and S-SCM-SM for K = 12, (c) MCM-LM and S- MCM-LM for K = 12, (d) MCM-SM and S-MCM-SM for K = 12, (e) STBCS-LSM and S-STBCS-LSM for K = 12, (f) CTBCS-LSM and S-CTBCS-LSM for K = 12, (g) ELM and (h) IST.

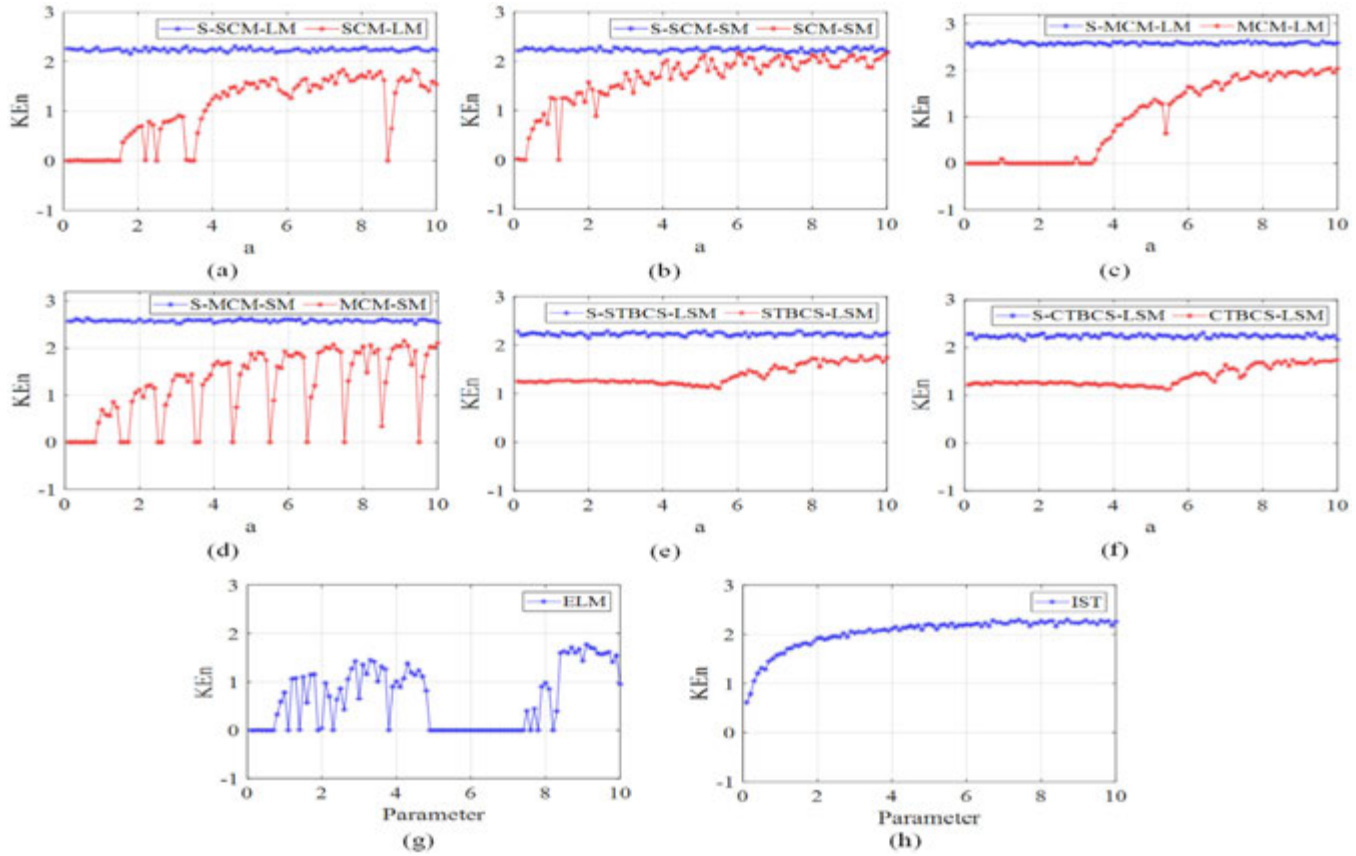

FIGURE 5 | KEn of: (a) SCM-LM and S-SCM-LM for K = 12, (b) SCM-SM and S-SCM-SM for K = 12, (c) MCM-LM and S-MCM-LM for K = 12, (d) MCM-SM and S-MCM-SM for K = 12, (e) STBCS-LSM and S-STBCS-LSM for K = 12, (f) CTBCS-LSM and S-CTBCS-LSM for K = 12, (g) ELM and (h) IST.

TABLE 2 | DNA encoding/decoding rules.

Binary	Quaternary	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
00	0	A	A	T	T	C	C	G	G
01	1	C	G	C	G	A	T	A	T
10	2	G	C	G	C	T	A	T	A
11	3	T	T	A	A	G	G	C	C

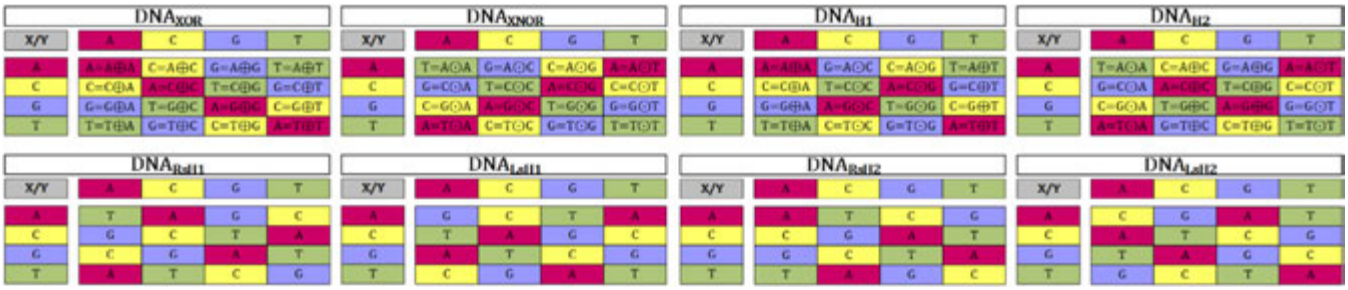

FIGURE 6 | DNA operations.

3.2 | DNA Sequence Operations

Different operations can be performed on the sequence of DNA, such as the operations of XOR, XNOR, addition and subtraction. DNA operations have pros such as low power consumption, huge storage capacity and huge parallelism [78]. However, these operations are basic, commutative and have been proven to be vulnerable. Following the principles in [79], the security of encryption relies heavily on nonlinear and non-commutative operations, such as substitution through S-boxes. Inspired by this, we propose non-commutative DNA operations that are designed to break the algebraic symmetry of traditional DNA-based logic operations, thereby improving resistance to cryptanalysis.

3.2.1 | DNA Exclusive-Or (DNA_{XOR}) and DNA Exclusive-Nor (DNA_{XNOR}) Operations

DNA operations are based on logical operations, including DNA exclusive-or (DNA_{XOR}) and DNA exclusive-nor (DNA_{XNOR}) operations according as shown in Figure 6. The image is encrypted using DNA operations between the chaotic sequence and the encoded image. The main purpose of the DNA operation rules is to change the value of the pixels in the DNA diffusion step. For example, suppose the DNA sequence of a pixel is $X = ACGT$ and the DNA key is $Y = CGAT$. For the encryption process, if we apply $Z = DNA_{XOR}\ (X, Y)$, the result is $CTGA$ ($A \oplus C = C, C \oplus G = T, G \oplus A = G, T \oplus T = A$). Then, for decryption process, $X = DNA_{XOR}\ (Z, Y)$, the result is $ACGT$ ($C \oplus C = A, T \oplus G = C, G \oplus A = G, A \oplus T = T$).

3.2.2 | New DNA Hybrid Operations

Two new hybrid operations are presented in this subsection, in which every single hybrid operation is formed by two different operations. In the first hybrid operation (DNA_{H1}), the first and fourth columns are selected the same as DNA_{XOR}, while the second and third columns are selected the same as DNA_{XNOR}. For example, if we perform the DNA_{H1} between $ACGT$ and $CGAT$ ($A \odot C = G, C \odot G = A, G \oplus A = G, T \oplus T = A$), the encryption result is $GAGA$, and the decryption is performed between $GAGA$ and $CGAT$ ($G \odot C = A, A \odot G = C, G \oplus A = G, A \oplus T = T$) to result in $ACGT$. In the second hybrid operation (DNA_{H2}), the first and fourth columns are the same as DNA_{XNOR}, while the second and third columns are the same as DNA_{XOR}. The hybrid operations are shown in Figure 6.

3.2.3 | New DNA Circular Shift Operations

The circular shift operation is a function that is applied to rearrange the components of a row. The right circular shift (Rs) moves the final element to the first position and shifts all other elements to the next position. The left circular shift (Ls) moves the last element to the first position and moves the other elements to the next left position.

Rs and Ls can be applied to the DNA hybrid operations to move the locations of DNA bases in a row and hence generate new DNA operations. Assume that the shifting amount is equal to 1, then the Rs of $DNA_{H1}(DNA_{RsH1})$ and Ls of $DNA_{H1}(DNA_{\mathrm{LSH1}})$ can be represented as follows:

$$DNA_{RsH1} = Rs\,(DNA_{H1}) = \begin{bmatrix} Rs\{A,G,C,T\} \\ Rs\{C,T,A,G\} \\ Rs\{G,A,T,C\} \\ Rs\{T,C,G,A\} \end{bmatrix} = \begin{bmatrix} T,A,G,C \\ G,C,T,A \\ C,G,A,T \\ A,T,C,G \end{bmatrix} \quad (11)$$

$$DNA_{LsH1} = Ls\,(DNA_{H1}) = \begin{bmatrix} Ls\{A,G,C,T\} \\ Ls\{C,T,A,G\} \\ Ls\{G,A,T,C\} \\ Ls\{T,C,G,A\} \end{bmatrix} = \begin{bmatrix} G,C,T,A \\ T,A,G,C \\ A,T,C,G \\ C,G,A,T \end{bmatrix} \quad (12)$$

Algorithm 1 | Key Vectorization Method (KVM)

Input: Vector $\boldsymbol{A}$ of size $\boldsymbol{N}1$, vector $\boldsymbol{B}$ of size $\boldsymbol{N}2$
Output: Matrix $\boldsymbol{EX}$ of size $\boldsymbol{N}1 \times \boldsymbol{N}2$
$[\boldsymbol{BS}, \boldsymbol{BI}]$ = Sort $(\boldsymbol{B})$; //$\boldsymbol{BS}$ is the sorted version of $\boldsymbol{B}$ and $\boldsymbol{BI}$ contains the original indices of $\boldsymbol{B}$.
$\boldsymbol{a} = [0, 1, 2, \ldots, \boldsymbol{N}1 - 1]^T$; // $\boldsymbol{a}$ is column vector representing row positions.
$\boldsymbol{b} = [\boldsymbol{BI}_1, \boldsymbol{BI}_2, \boldsymbol{BI}_3, \ldots, \boldsymbol{BI}_{\boldsymbol{N}2}]$; // $\boldsymbol{b}$ is a row vector.
$\boldsymbol{S} = (\boldsymbol{a} + \boldsymbol{b})$ mod $\boldsymbol{N}1$; // Apply element-wise addition to generate the shifted indices matrix $\boldsymbol{S}$ of size $\boldsymbol{N}1 \times \boldsymbol{N}2$.
$\boldsymbol{EX} = \boldsymbol{A}(\boldsymbol{S})$;

The circular shift functions can also be applied to DNA_{H2} to form two new operations, namely DNA_{RSH2} and DNA_{LSH2}. The results are shown in Figure 6. If we alter the shifting amount to equal 2 or 3, we can get different circular shift operations. As there are eight encoding rules, likewise, there available eight kinds of every single DNA operation. The DNA_{XOR} and DNA_{XNOR} are commutative, while the hybrid operations and circular shift operations are non-commutative.

3.3 | Key Vectorisation Method (KVM)

Vectorisation is a process of applying an operation to multiple data values concurrently, thus allowing for efficient data processing without the need for explicit loops that perform individual scalar operations. This technique can be integrated with single instruction multiple data (SIMD) extensions in modern CPUs to enhance execution speed by leveraging parallel execution to accelerate tasks [17]. In most DNA encryption techniques, the key sequence dimensions must align with the input image dimensions. Thus, instead of iterating the chaotic maps until they fit the input image dimensions, which is time-consuming, we define the KVM as a method to expand the length of chaotic sequences by leveraging vectorisation and the SIMD technique.

Suppose a set A consists of $N1$ elements, we can get a different set if we randomly shuffle the elements of A according to another set B using the sort and circular shift functions. The KVM can be expressed as follows:

$$EX\left(N1, N2\right) = KVM\left(A\left(N1\right), B\left(N2\right)\right) \tag{13}$$

A represents a vector to be optimised, B represents a shifting vector, and $N1$ and $N2$ represent the number of elements in A and B respectively. Algorithm 1 illustrates the KVM.

4 | Propsed Encryption System

The DNA-DCP-DA includes four styles of processes; key sequence generation process, DNA encoding process, DNA permutation-diffusion process and DNA decoding process. The block diagram of DNA-DCP-DA is shown in Figure 7. Users can select any chaotic maps generated by proposed models, as their efficiency is proven, and any DNA operations to be directly used in the proposed encryption algorithm. In this proposed work, we used four new chaotic maps (S-SCM-LM, S-MCM-LM, S-STBCS-LM and S-CTBCS-LM) and six new DNA operations (DNA_{H1}, DNA_{H2}, DNA_{RsH1}, DNA_{LsH1}, DNA_{RsH2} and DNA_{LsH2}). Motivated by the numeric example in [80], the encryption process is demonstrated on a small-sized image for clarity in Figure 8.

4.1 | Key Sequence Generation Process

This subsection illustrates the generation process of chaotic sequences according to the pixels of the input image and then sequence vectorisation.

4.1.1 | Hash Sequence

Hash functions have an important part in the process of generating chaotic sequences. Because the hash functions are irreversible, they endure such plaintext and ciphertext attacks. Message Digest Algorithm (MD5) is a hash function that generates 128-bit hash values [81]. The proposed method uses a double-layer MD5 Algorithm to raise the security level. The proposed method generates the hash value based on the input image and the initial values of the chaotic maps.

Let the input image $(E1)$ be a 2D matrix in the size of $M \times N$, we generate three vectors; $V1$, $V2$ and $V3$. $V1$ represents the summation of $E1$, $V2$ of the length M represents the row-wise summation of E_1 and $V3$ of the length N represents the column-wise summation of E_1. Applying the double-layer MD5 process, where the MD5 function is firstly applied to the $V1$, $V2$ and $V3$ separately to generate three individual hash values. These individual hash values are combined and hashed again using the MD5 function to result in the final hash value (C). The hash value (C) with a length of 128 bits is split into 8-bit blocks in the decimal format as follows:

$$\begin{cases} C = MD5\left(MD5\left(V1\right), \mathrm{MD5}\left(V2\right), \mathrm{MD5}\left(V3\right)\right) \\ C = \{c_1, c_2, c_3, \ldots, c_{16}\} \end{cases} \tag{14}$$

After that, we generate four values that work as image keys as follows:

$$\begin{cases} S_1 = (c_1 \oplus c_2 \oplus c_3 \oplus c_4) \\ S_2 = (c_5 \oplus c_6 \oplus c_7 \oplus c_8) \\ S_3 = (c_9 \oplus c_{10} \oplus c_{11} \oplus c_{12}) \\ S_4 = (c_{13} \oplus c_{14} \oplus c_{15} \oplus c_{16}) \end{cases} \tag{15}$$

Let the initial parameter keys $(\mathrm{x'}_1, \mathrm{x'}_2, \mathrm{x'}_3, \mathrm{x'}_4)$ be selected randomly. Then, the initial values of the chaotic maps are selected based on hash values of the input image as follows:

$$\begin{cases} x_1\left(0\right) = \left(x'_1 + 0.1 \; mod \; S_1\right)/256 \\ x_2\left(0\right) = \left(x'_2 + 0.1 \; mod \; S_2\right)/256 \\ x_3\left(0\right) = \left(x'_3 + 0.1 \; mod \; S_3\right)/256 \\ x_4\left(0\right) = \left(x'_4 + 0.1 \; mod \; S_4\right)/256 \end{cases} \tag{16}$$

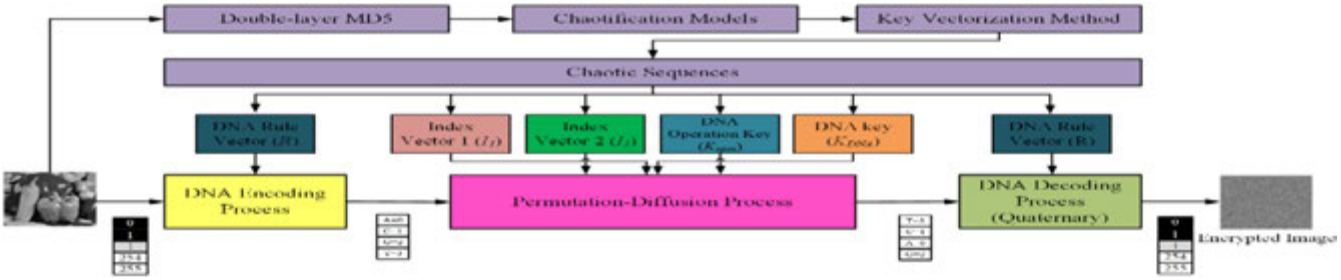

FIGURE 7 | Block diagram of DNA-DCP-DA.

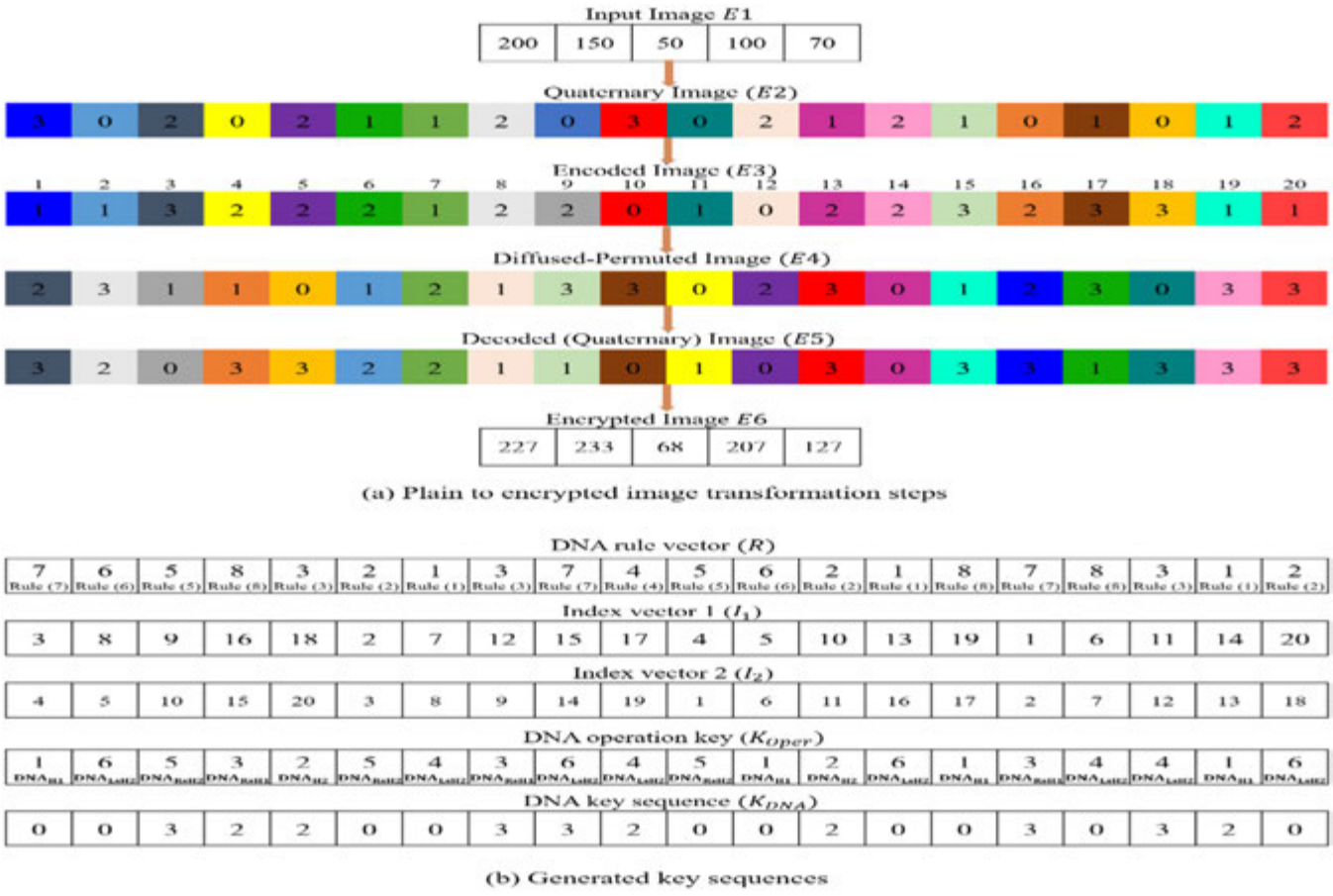

FIGURE 8 | Illustration of the DNA-DCP-DA encryption process using a 1×4 image.

4.1.2 | Generating Chaotic Sequences

Firstly, define the parameters $a^1, a^2, a_1^1, a_2^1, a_1^2, a_2^2, K^1, K^2, K^3$ and K^4 of the chaotic maps. Secondly, iterate S-SCM-LM with parameters a^1 and K^1 and initial value $x_1(0)$, and iterate S-STBCS-LM with parameters a_1^1, a_2^1 and K^2 and initial value $x_2(0)$, each for M iterations. Thirdly, iterate S-MCM-LM with parameter a^2 and K^3 and initial value $x_3(0)$, and iterate S-CTBCS-LM with parameters a_1^2, a_2^2 and K^4 and initial value $x_4(0)$, each for $4N$ iterations. Lastly, using the KVM to generate four sequences of size $M \times 4N$ to fit the DNA encryption needs as in the following equations:

$$EX_1(M, 4N) = \text{KVM}\ (x_1(M), x_3(4N)) \tag{17}$$

$$EX_2(M, 4N) = \text{KVM}\ (x_1(M), x_4(4N)) \tag{18}$$

$$EX_3(M, 4N) = \text{KVM}\ (x_2(M), x_3(4N)) \tag{19}$$

$$EX_4(M, 4N) = \text{KVM}\ (x_2(M), x_4(4N)) \tag{20}$$

here, KVM is the same as Algorithm 1. Then, EX_1, EX_2, EX_3 and EX_4 are reshaped into a 1D vector of size $1 \times 4MN$.

4.2 | DNA Encoding Process

The input image $E1$ is a 2D matrix with the size $M \times N$. In the step of encoding, the input image is dynamically transformed into DNA encoded image $E3$ with a size of $1 \times 4MN$. The complete steps of the DNA encoding are as follows:

Step 1: Input Image $E1$ with a size of $M \times N$ is reshaped into 1D vector with a size of $1 \times MN$.

Step 2: Transform each element in $\boldsymbol{E1}$ into 4 digits of quaternary format to form the quaternary image $E2$ with a size of $1 \times 4MN$.

Step 3: Calculate the DNA rule vector R with a size of $1 \times 4MN$ by using the chaotic sequence EX_1 as illustrated in the following equation.

$$R(i) = \text{floor}\left(EX_1 \times 1000\right) \bmod 8 + 1 \tag{21}$$

i ranges from 1 to $4MN$. $R(i)$ takes integer values from the set $\in \{1, 2, 3, 4, 5, 6, 7, 8\}$, corresponding to the number of rules in Table 2.

Step 4: Encode each element in the quaternary image $E2$ into one of the DNA bases (0, 1, 2, 3) according to the rule vector R, to form the encoded image $E3$ with a size $1 \times 4MN$. For each index $\boldsymbol{i}$, the encoding rule applied is determined by the value of $R(i)$, and the mapping follows a predefined set of rules in Table 2. The encoding process for $\boldsymbol{E}2$ can be expressed as:

$$E3(i) = \begin{cases} \text{Encode}\left(E2(i), Rule\ 1\right), & if\ R(i) = 1 \\ \text{Encode}\left(E2(i), Rule\ 2\right), & if\ R(i) = 2 \\ \text{Encode}\left(E2(i), Rule\ 3\right), & if\ R(i) = 3 \\ \text{Encode}\left(E2(i), Rule\ 4\right), & if\ R(i) = 4 \\ \text{Encode}\left(E2(i), Rule\ 5\right), & if\ R(i) = 5 \\ \text{Encode}\left(E2(i), Rule\ 6\right), & if\ R(i) = 6 \\ \text{Encode}\left(E2(i), Rule\ 7\right), & if\ R(i) = 7 \\ \text{Encode}\left(E2(i), Rule\ 8\right), & if\ R(i) = 8 \end{cases} \tag{22}$$

4.3 | Permutation-Diffusion Process

The concurrent permutation-diffusion process is important to protect the system from different attacks, such as plaintext attacks. The complete steps are summarised as follows:

Step 1: Calculate two index vectors as follows:

$$[XS_1, I_1] = \text{sort}\left(EX_1\right) \tag{23}$$

$$[XS_2, I_2] = \text{sort}\left(EX_2\right) \tag{24}$$

I_1 and I_2 are the index values of EX_1 and EX_2, respectively. XS_1 and XS_2 are the sorted sequences.

Step 2: Compute the DNA operation key K_{Oper} with a size of $1 \times 4MN$ as follows:

$$K_{Oper}(i) = \text{floor}\left(EX_3(i) \times 1000\right) \bmod 6 + 1 \tag{25}$$

where i ranges from 1 to $4MN$ and $K_{Oper}(i) \in \{1, 2, 3, 4, 5, 6\}$.

Algorithm 2 | Permutation-diffusion process

Input: Encoded image $\boldsymbol{E}3$ of size $1 \times 4\boldsymbol{MN}$, vector $\boldsymbol{EX}_1$ of size $1 \times 4\boldsymbol{MN}$, vector $\boldsymbol{EX}_2$ of size $1 \times 4\boldsymbol{MN}$, vector $\boldsymbol{EX}_3$ of size $1 \times 4\boldsymbol{MN}$, vector $\boldsymbol{EX}_4$ of size $1 \times 4\boldsymbol{MN}$

Output: Diffused image $E4$ of size $1 \times 4\boldsymbol{MN}$

$[\boldsymbol{XS}_1, \boldsymbol{I}_1] =$ sort($\boldsymbol{EX}_1$(i)); // $\boldsymbol{I}_1$ is the first index vector.

$[\boldsymbol{XS}_2, \boldsymbol{I}_2] =$ sort($\boldsymbol{EX}_2$); // $\boldsymbol{I}_2$ is the second index vector.

$\boldsymbol{K_{Oper}}\ (i) = \text{floor}(\boldsymbol{EX}_3(i) \times 1000) \bmod 6 + 1$; // $\boldsymbol{K_{Oper}}(i)$ is the DNA operation vector.

$\boldsymbol{K_{DNA}}\ (i) = \text{floor}(\boldsymbol{EX}_4(i) \times 1000 - \text{floor}(\boldsymbol{EX}_4(i) \times 1000)) \times 4$; // $\boldsymbol{K_{DNA}}(i)$ is the DNA key vector.

$$\boldsymbol{E}4\,(i) = \begin{cases} DNA_{H1}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(i) = 1 \\ DNA_{H2}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(\boldsymbol{i}) = 2 \\ DNA_{RsH1}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(i) = 3 \\ DNA_{LsH1}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(i) = 4 \\ DNA_{RsH2}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(i) = 5 \\ DNA_{LsH2}(\boldsymbol{E}3(\boldsymbol{I}1(i)), \boldsymbol{K_{DNA}}(\boldsymbol{I}2(i))), & if\ \boldsymbol{K_{Oper}}(i) = 6 \end{cases}$$

// $\boldsymbol{E}4$ is the diffused-permuted image.

Step 3: Compute the DNA key sequence K_{DNA} with a size of $1 \times 4MN$ as follows:

$$K_{DNA}(i) = \text{floor}\left(EX_4(i) \times 1000 - \text{floor}\left(EX_4(i) \times 1000\right)\right) \times 4 \tag{26}$$

where $i = 1, 2, \ldots, 4MN$ and K_{DNA} is designed to generate quaternary sequences, where $K_{DNA} \in \{0, 1, 2, 3\}$.

Step 4: Execute the permutation-diffusion process by applying the DNA operations between the encoded image $E3$ and the K_{DNA} sequence according to K_{Oper} to obtain the diffused-permuted image $E4$ with the size of $1 \times 4MN$ as shown in the following Equation:

$$E4(i) = \begin{cases} \text{DNA}_{\text{H1}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 1 \\ \text{DNA}_{\text{H2}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 2 \\ \text{DNA}_{\text{RsH1}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 3 \\ \text{DNA}_{\text{LsH1}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 4 \\ \text{DNA}_{\text{RsH2}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 5 \\ \text{DNA}_{\text{LsH2}}\left(E3\left(I1(i)\right), K_{DNA}\left(I2(i)\right)\right), & if\ K_{Oper}(i) = 6 \end{cases} \tag{27}$$

$i = 1, 2, \ldots, 4MN$. The K_{Oper} sequence is used to determine which DNA operation is to be applied, ensuring a dynamic impact. $I1$ and $I2$ are used to dynamically select elements from the encoded image $E3$ and K_{DNA}, hence creating a permutation effect. The DNA operations ensure diffusion by mixing their combined influence. This approach strengthens encryption by integrating both diffusion and permutation, while reducing computational overhead. The permutation-diffusion process is outlined in Algorithm 2.

4.4 | DNA Decoding Process

The decoding step converts the diffused image $E4$ into the encrypted greyscale image $E6$ with a size of $M \times N$. The steps of decoding are as follows:

Step 1: Decode the diffused image $E4$ into quaternary image $E5$ with a size of $1 \times MN$ according to R.

$$E5(i) = \begin{cases} \text{Decode}\,(E4(i), Rule\ 1), & if\ R(i) = 1 \\ \text{Decode}\,(E4(i), Rule\ 2), & if\ R(i) = 2 \\ \text{Decode}\,(E4(i), Rule\ 3), & if\ R(i) = 3 \\ \text{Decode}\,(E4(i), Rule\ 4), & if\ R(i) = 4 \\ \text{Decode}\,(E4(i), Rule\ 5), & if\ R(i) = 5 \\ \text{Decode}\,(E4(i), Rule\ 6), & if\ R(i) = 6 \\ \text{Decode}\,(E4(i), Rule\ 7), & if\ R(i) = 7 \\ \text{Decode}\,(E4(i), Rule\ 8), & if\ R(i) = 8 \end{cases} \tag{28}$$

Step 2: Transform the quaternary image $E5$ into encrypted image $E6$ with the size of $1 \times MN$ by converting every 4 digits to an integer value ranging from 0 to 255.

Step 3: Reshape the image $E6$ to a size of $M \times N$ to form the final encrypted image.

4.5 | Decryption Process Details

The decryption algorithm is the same as the encryption algorithm but in reverse way with the same encryption keys. The entire steps are as follows:

Step 1: Reshape the encrypted image $E6$ to the size of $1 \times MN$.

Step 2: Compute the rule vector R.

Step 3: Transform the encrypted image $E6$ into quaternary format to obtain the quaternary image $E5$.

Step 4: Encode the quaternary image $E5$ as in Equation (22) to recover the diffused-permuted image $E4$.

Step 5: Calculate I_1, I_2, K_{DNA} and K_{Oper} vectors.

Step 6: Apply the inverse permutation-diffusion process to recover $E3$ as detailed in the following equation.

$$E3\,(I1(i)) = \begin{cases} \text{DNA}_{\text{H1}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 1 \\ \text{DNA}_{\text{H2}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 2 \\ \text{DNA}_{\text{RsH1}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 3 \\ \text{DNA}_{\text{LsH1}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 4 \\ \text{DNA}_{\text{RsH2}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 5 \\ \text{DNA}_{\text{LsH2}}\,(E4(i), K_{DNA}\,(I2(i))), & if\ K_{Oper}(i) = 6 \end{cases} \tag{29}$$

Step 7: Decode the encoded image $E3$ as in Equation (28) to recover the quaternary image $E2$.

Step 8: Convert every 4 digits of the quaternary image $E2$ to an integer value with a range of 0 to 255 to form $E1$. Then, resize the $E1$ to the $M \times N$ size to recover the original image.

4.6 | Colour Image Encryption

This work can be extended to colour images by splitting the image into its red (R), green (G) and blue (B) channels, applying the same encryption process as for greyscale images to each channel separately, and then recombining the channels to produce the encrypted colour image (see Figure 9).

5 | Performance and Security Analysis

Many experiments and analyses are conducted in this section to accurately evaluate the proposed image encryption method. The experiments are conducted using the MATLAB R2021a, which is a platform on Windows 10. The test images with size of 512 × 512 ('peppers', 'baboon', 'camera man', 'boat', 'peppers (colour)', 'baboon (colour)') are chosen from the CVG-UGR database (https://ccia.ugr.es/cvg/dbimagenes/) and USC-SIPI (https://sipi.usc.edu/database/) to be used as plain images. Figure 10 shows the plain images with their corresponding encrypted images. To demonstrate the superiority of the proposed algorithm, we make a comparison with some recent algorithms that used DNA technology with chaos in [26, 63, 82–86], S-box technology in [62, 87] and elliptic curve technology with chaos in [88].

5.1 | Key Security Analysis

In this subsection the security is analysed using key space analysis, key sensitivity and key sequence analysis to prove that the proposed algorithm can satisfy the key security requirements and can exhibit effective performance.

5.1.1 | Key Space

The encryption algorithm is capable of withstanding brute force attacks when it possesses a substantial key space. Generally, the encryption algorithm resistant to brute force attacks must own the key space that exceeds 2^{100} [89]. In the proposed scheme, the chaotic map's initial values $x_1(0), x_2(0), x_3(0)$ and $x_4(0)$ and parameters $a^1, a^2, a_1^1, a_2^1, a_1^2, a_2^2, K^1, K^2, K^3$ and K^4 are considered secret keys. Each of S-SCM-LM and S-MCM-LM has two parameters and one initial value. S-STBCS-LM has three parameters and one initial value, and so does S-CTBCS-LM. Therefore, we own ten parameters and four initial values. If the precision of initial values and parameters is equivalent to 10^{-15}, the key space is $10^{15\times14} = 10^{210} \approx 2^{697.4}$ which is larger than 2^{100}. So, the proposed scheme proposes a highly efficient key space. The key space if we use the maps generated from original chaotification models (SCM-LM, MCM-LM, STBCS-LM, CTBCS-LM) is equal to $10^{15\times10} = 10^{150} \approx 2^{498.2}$, because we can own 6 parameters and 4 initial values. Thus, the proposed maps can not only generate high random sequences but also increase the key space. Table 3 lists the results. Compared to the other works in Table 3, the proposed work has a higher key space, and that proved the high efficiency of the proposed maps. Increasing the key space is a predominant requirement in modern cryptosystems to ensure resistance against brute-force attacks. In our design, we theoretically propose a large key space,

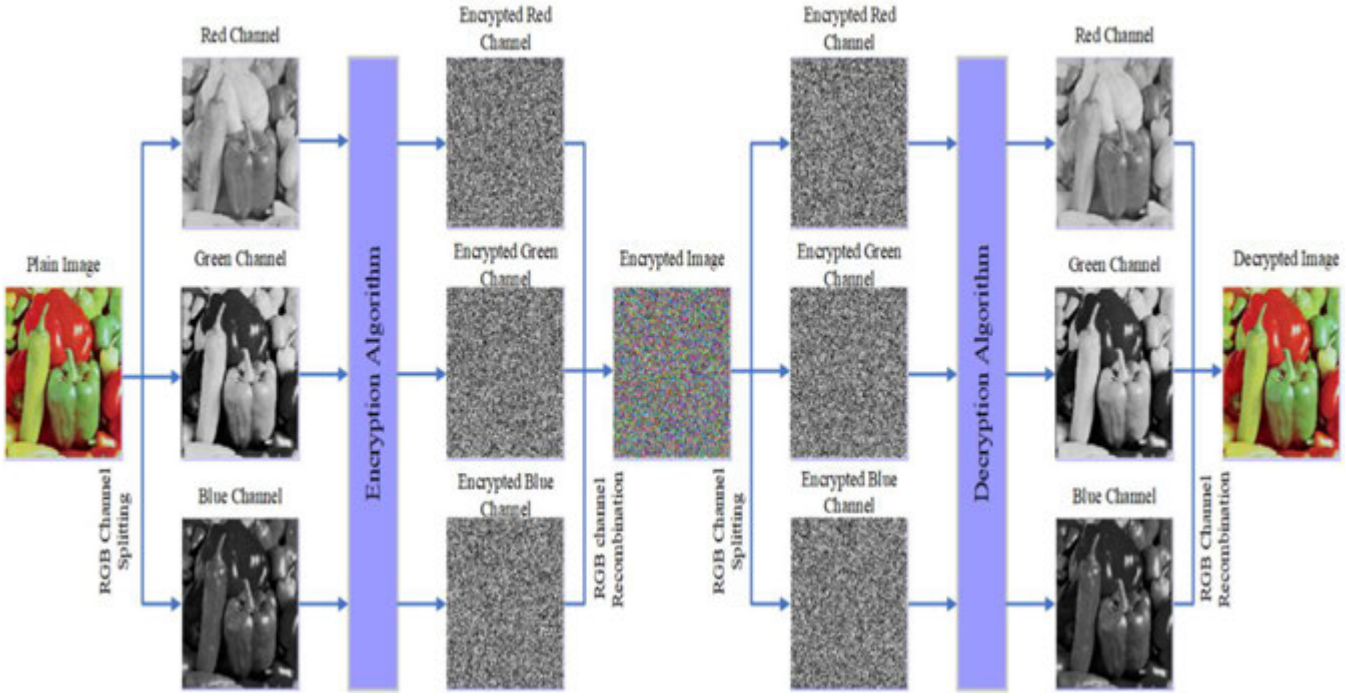

FIGURE 9 | Block diagram of the RGB encryption and decryption process.

FIGURE 10 | Simulation results of the test images with their corresponding encrypted images.

TABLE 3 | Key space results.

Algorithm	Proposed	Proposed with original chaotification models	Ref. [26]	Ref. [82]	Ref. [83]	Ref. [84]	Ref. [86]	Ref. [62]	Ref. [88]	Ref. [63]	Ref. [87]
Key space	$10^{210} \approx 2^{697.6}$	$10^{150} \approx 2^{498.6}$	2^{512}	$10^{75} \approx 2^{249}$	$10^{141} \approx 2^{468.3}$	$10^{105} \approx 2^{349}$	$10^{79} \approx 2^{263}$	$10^{120} \approx 2^{398}$	2^{512}	2^{360}	$10^{120} \approx 2^{398}$

reaching up to $2^{697.4}$, to enhance security. Although this level of precision can be supported in theory, practical implementations of chaotic maps on finite precision computers may experience reduced effective precision due to rounding errors or computer limitations, particularly in floating-point representations [90]. Therefore, proposing a high theoretical key space provides an additional security margin to compensate for such practical limitations.

5.1.2 | Key Sensitivity

A key sensitivity test is an important parameter used to ensure that a very tiny modification in the key can result in a huge variation in both encrypted/decrypted images [91]. The key sensitivity test is to decrypt an encrypted image with a decryption key that varies from the encryption key by 10^{-15}. In this method, the initial values and the parameters are secret keys. We add a tiny change

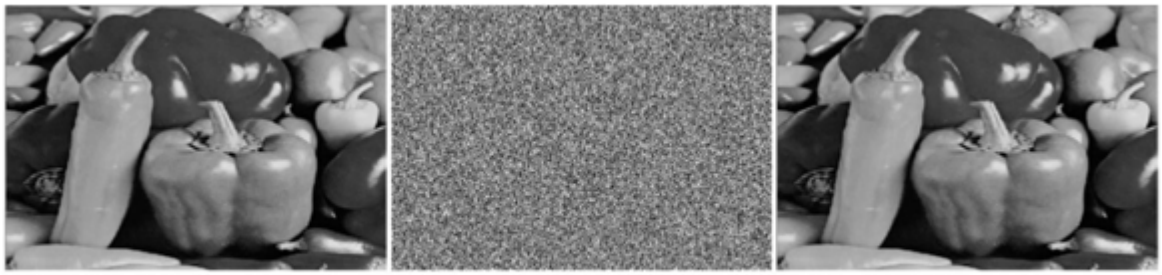

FIGURE 11 | Key sensitivity test.

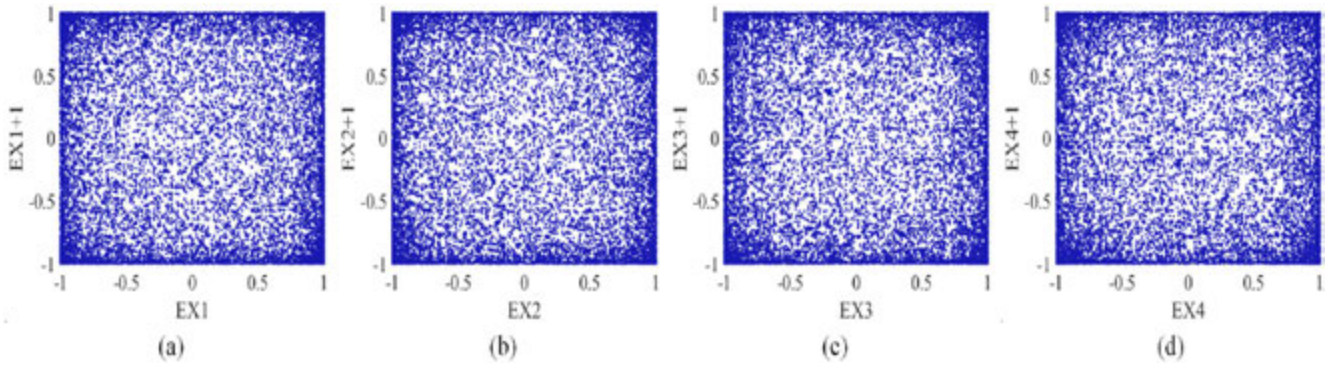

FIGURE 12 | Phase trajectories of the key sequences generated by KVM: (a) EX1 sequence, (b) EX2 sequence, (c) EX3 sequenceand (d) EX4 sequence.

TABLE 4 | SEN and KEN of key sequences generated by KVM.

Sequences	EX1	EX2	EX3	EX4
SEn	1.92419	1.92423	1.91729	1.91726
KEn	2.25011	2.25030	2.23398	2.23404

to one of the parameters in the decryption process, and the result can be shown in Figure 11. The results obtained demonstrate a significant disparity between the restored decrypted image and the original image when decrypting the image with a slightly altered key.

5.1.3 | Key Sequences Analysis

The key sequences expanded by KVM were analysed using phase trajectory, SEn, and KEn. Phase trajectory was employed to assess randomness and to investigate whether the sequences' trajectories introduce exploitable patterns. The absence of recognisable patterns confirms that the KVM produces random sequences. The result is shown in Figure 12. The results show that the trajectories of the key sequences explore the entire state space without bias, indicating strong randomness. In addition, SEn and KEn are also invested in quantifying the complexity of the key sequences. The results are reported in Table 4, showing the average of 10 executions with varying initial values. The results show that the key sequences have high SEn and KEn values, close to those of the proposed chaotic maps, demonstrating that the KVM does not compromise security.

5.2 | Statistical Analysis

In this work, histogram analysis, correlation analysis and IE are the three indications used in evaluating the suggested algorithm with respect to statistical attacks.

5.2.1 | Histogram Analysis

The histogram describes the information about an image regarding the intensities of pixel distribution [92]. The input image shows the histogram with a nonuniform shape because it has a unique shape. Conversely, the encrypted image should exhibit a high uniformity level to prevent information from being extracted. The attackers can effortlessly examine the encrypted image regularity when the encrypted image has an uneven histogram and then make statistical attacks. We analysed the histogram of the encrypted image, as it is shown in Figure 13. As can be noticed, the encrypted image histogram has an even distribution, illustrating a preferable performance in terms of histogram.

5.2.2 | Correlation Analysis

A secured encryption method is expected to defeat the high-level correlation that occurs between the adjacent pixels of the input images [93]. Statistical attacks can exploit the correlation of adjacent encrypted pixels to retrieve some beneficial information regarding the plain image. The correlation value within a close range of 1 illustrates a high correlation and an unsecured encryp-

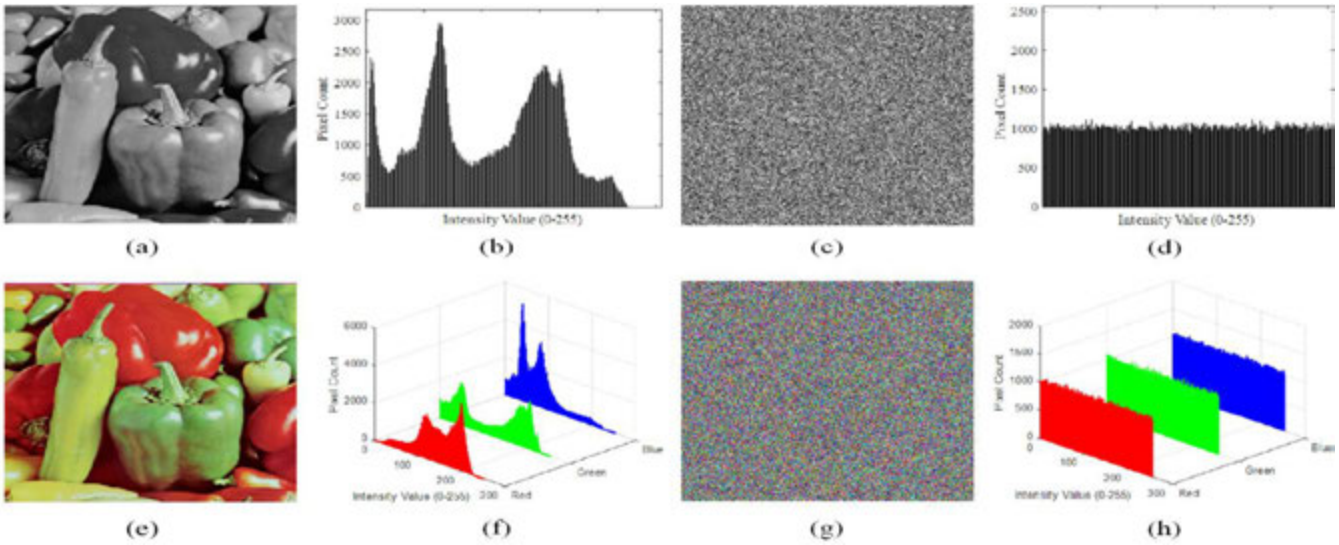

FIGURE 13 | Histogram analysis. (a) Original greyscale image, (b) histogram of the original greyscale image, (c) encrypted greyscale image, (d) histogram of the encrypted greyscale image, (e) original colour image, (f) histogram of the original colour image, (g) encrypted colour image and (h) histogram of the encrypted colour image.

tion algorithm. Conversely, the correlation value within a close range of 0 indicates that the encryption method defeats the correlation and secured the encryption algorithm. The mathematical expression of correlation is given as:

$$\begin{cases} H(x) = \frac{1}{N}\sum_{i-1}^{N} x_i \\ V(x) = \frac{1}{N}\sum_{i-1}^{N} (x_i - H(x))^2 \\ C(x,y) = \frac{1}{N}\sum_{i-1}^{N} (x_i - H(x))(y_i - H(y)) \\ E_{xy} = \frac{C(x,y)}{\sqrt{V(x)}\sqrt{V(y)}} \end{cases} \tag{30}$$

N represents the total number of adjacent pixel pairs; x and y represent two locationally adjacent pixels; $H(x)$ represents the mean; $V(x)$ represents the variance and $C(x, y)$ represents the covariance. Table 5 shows the results for greyscale images, and Table 6 shows the results for colour images. The results represent the mean over 10 trials using different initial keys for the proposed method. Both Tables 5 and 6 demonstrate that the correlation coefficients of the encrypted images approached nearly 0 in three orientations. Furthermore, Tables 5 and 6 showcase the outcomes of additional references, thereby affirming the superior efficacy of the proposed methodology in comparison with those of other references. Consequently, the results substantiate that the neighbouring pixels within the encrypted image exhibit a lack of correlation, and the standard deviation of the correlation coefficients across 10 trials is very small (approaching zero), confirming that the proposed method consistently eliminates correlation between adjacent pixels regardless of the initial key. Visually, Figure 14 reveals that the distribution of pixels of the input image is located near $x = y$, whereas the pixels of the encrypted image are distributed evenly in three directions. Consequentially, the proposed encryption method eliminates the correlation in three directions.

5.2.3 | Information Entropy (IE)

The IE is applied as an assessing measure to determine the distribution of pixels in the encrypted image. The ideal value of IE equals 8 [94]. An effective algorithm should suggest a random encrypted image; thus, its entropy reaches close to 8. The IE is expressed as:

$$\text{IE}\ (E) = \sum_{i=0}^{2^H-1} P(E_i) \log_2 \frac{1}{P(E_i)} \tag{31}$$

H refers to total length of pixel in bits and $P(E_i)$ refers to probability of E_i in image E. Tables 7 and 8 tabulate the results for greyscale images and colour images. The results represent mean ± standard deviation over 10 trials with varying initial keys for the proposed method. The results obtained in Tables 7 and 8 closely approximate the ideal value. Furthermore, in comparison to alternative schemes, the proposed algorithm demonstrated commendable outcomes.

5.3 | Diffusion and Permutation Performance Analysis

The differential attacks and plaintext attacks are two references to measure the effectiveness of the encryption algorithm in terms of diffusion effect, while the permutation test is a reference to measure the permutation effect.

5.3.1 | Differential Attacks

The differential attack is a type of plaintext attack that examines how a change in the plain images can affect the encrypted images [95]. Firstly, the attackers encrypt the input image to form the first encrypted image. Secondly, they apply a small modification to the input image and then encrypt the image to create the

TABLE 5 | Correlation analysis results of encrypted images (greyscale image).

Image	Direction	Encrypted image							
		Proposed	Ref. [26]	Ref. [83]	Ref. [84]	Ref. [85]	Ref. [86]	Ref. [62]	Ref. [88]
Baboon	Horizontal	−0.00172	0.0124	−0.0068	0.0119	−0.0009	−0.0136	−0.00004	0.0011
	Vertical	−0.00191	−0.0118	−0.0082	0.0014	−0.0130	−0.0075	0.00052	−0.0012
	Diagonal	−0.00059	−0.0215	0.0036	−0.0055	0.0186	0.0156	0.00053	0.0014
Peppers	Horizontal	0.00162	0.0049	0.0131	−0.0061	0.0052	−0.0095	0.00050	0.0017
	Vertical	−0.00246	0.0099	0.0022	0.0002	0.0039	−0.0115	0.00088	−0.0012
	Diagonal	0.00142	0.0068	0.0030	−0.0219	0.0215	0.0118	0.00035	−0.0021
Cameraman	Horizontal	0.00330	−0.0061	0.0102	0.0119	−0.0077	−0.0337	−0.00091	−0.0015
	Vertical	−0.00056	0.0058	0.0066	0.0175	0.0061	0.0023	−0.00065	0.0019
	Diagonal	−0.00212	0.0166	−0.0202	−0.0179	0.0083	0.0101	−0.00099	0.0018
Boat	Horizontal	−0.00002	—	—	—	—	—	—	—
	Vertical	−0.00002	—	—	—	—	—	—	—
	Diagonal	0.00162	—	—	—	—	—	—	—

TABLE 6 | Correlation analysis results of encrypted images (colour images).

Image	Direction	Channel	Encrypted image		
			Proposed	Ref. [63]	Ref. [87]
Baboon (colour)	Horizontal	R	−0.00067	0.01478	0.0003
		G	0.00206	0.02172	0.0001
		B	−0.00067	−0.01131	0.0001
	Vertical	R	0.00145	−0.00589	0.0004
		G	−0.00242	0.00829	−0.0002
		B	−0.00404	−0.01634	−0.0004
	Diagonal	R	−0.00378	−0.03433	0.0004
		G	−0.00070	0.00278	0.0001
		B	−0.00077	0.00660	−0.0002
Peppers (colour)	Horizontal	R	0.00114	−0.00317	−0.0001
		G	−0.00157	0.00775	−0.0004
		B	0.00227	−0.00385	−0.0003
	Vertical	R	−0.00043	0.01419	−0.0002
		G	−0.00276	0.01451	0.0003
		B	−0.00396	0.00297	−0.0001
	Diagonal	R	−0.00010	0.01163	−0.0001
		G	0.00118	−0.00338	−0.0001
		B	−0.00014	0.00159	0.0002

TABLE 7 | Information entropy (IE) results (greyscale images).

Image	Information entropy (IE)							
	Proposed	Ref. [26]	Ref. [83]	Ref. [84]	Ref. [85]	Ref. [86]	Ref. [62]	Ref. [88]
Baboon	7.9993 ± 0.00006	7.9993	7.9993	7.9992	7.9992	7.9994	9.9994	7.9997
Peppers	7.9993 ± 0.00008	7.9993	7.9992	7.9993	7.9978	7.9993	9.9994	7.9993
Camera man	7.9993 ± 0.00005	7.9992	7.9991	7.9993	7.9983	7.9992	9.9994	7.9990
Boat	7.9993 ± 0.00007	—	—	—	—	—	—	—

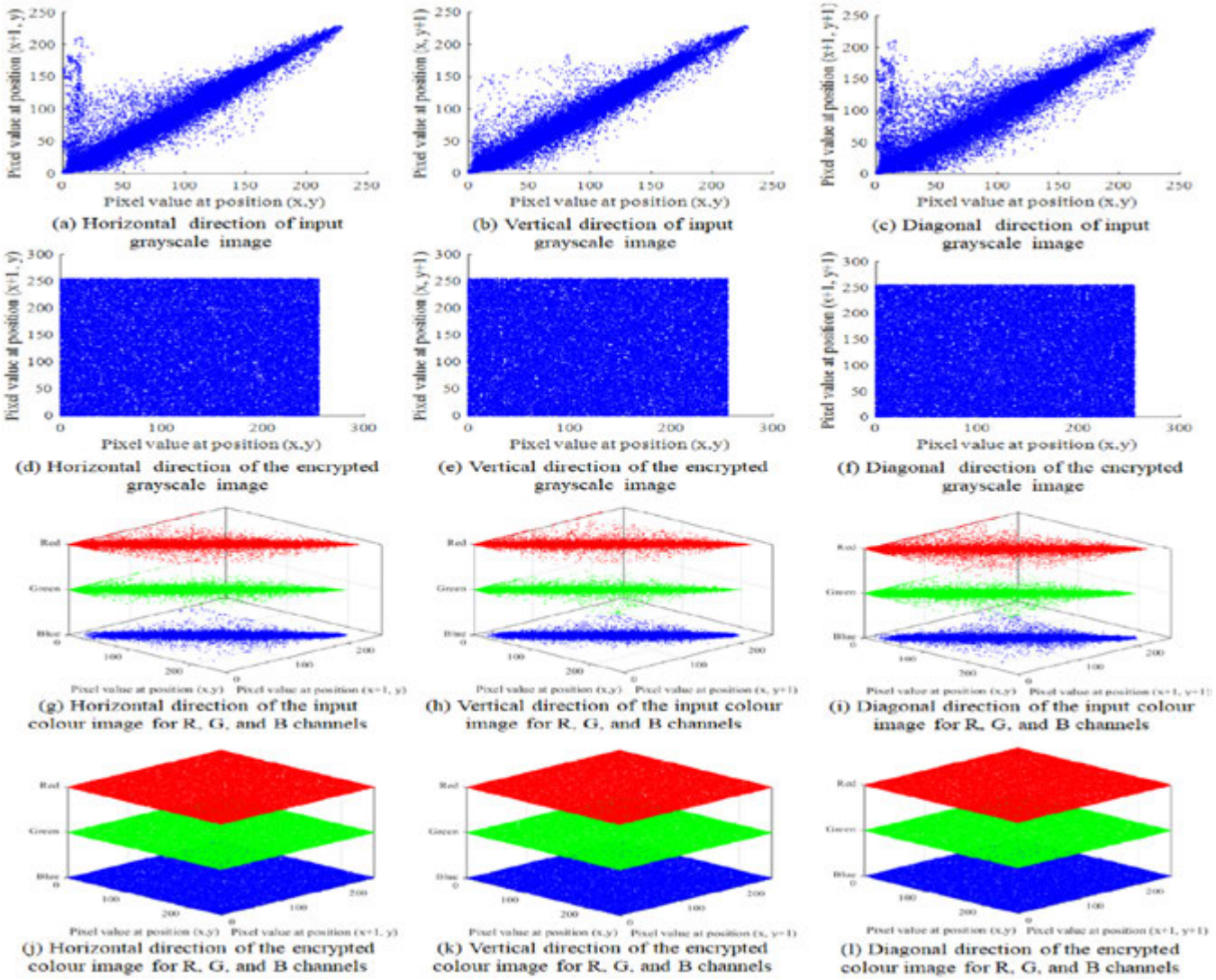

FIGURE 14 | Correlation analysis results.

TABLE 8 | Information entropy (IE) results (colour images).

Image	Channel	Information entropy (IE)		
		Proposed	Ref. [63]	Ref. [87]
Baboon	R	7.9993 ± 0.00008	7.9993	7.9993
	G	7.9994 ± 0.00004	7.9993	7.9993
	B	7.9994 ± 0.00003	7.9992	7.9994
Peppers	R	7.9993 ± 0.00010	7.9993	7.9994
	G	7.9993 ± 0.00010	7.9991	7.9993
	B	7.9993 ± 0.00008	7.9992	7.9993

second image. Thirdly, they compared the two encrypted images to reveal the interconnection of the plain and the encrypted images. The efficient algorithm should have a high degree of sensitivity to any tiny change that occurs in the input image, which means the original image possesses different encryption results in case a single bit is changed. This is also referred to as plaintext sensitivity. Encryption algorithms with better plaintext sensitivity can hold better resistance to differential attacks and hold a better diffusion property. The number of pixel change rate (NPCR) and uniform average change intensity (UACI) are used to test the resistance of an encryption algorithm towards differential attacks. The definitions of NPCR and UACI are as follows:

$$\begin{cases} D_{H_1,H_2}(i,j) = \begin{cases} 1, H_1(i,j) = H_2(i,j) \\ 0, H_1(i,j) \neq H_2(i,j) \end{cases} \\ NPCR = \sum_{i,j} \dfrac{D_{H_1,H_2}(i,j)}{M \times N} \times 100\% \end{cases} \tag{32}$$

$$\text{UACI} = \frac{1}{M \times N} \sum_{i,j} \frac{|H_1(i,j) - H_2(i,j)|}{255} \times 100\% \tag{33}$$

H_1 refers to the original encrypted image, and H_2 refers to the encrypted image after changing one pixel in its plain image. For a secured algorithm, NPCR and UACI should be within 99.6% and 33.4%, respectively [26]. Tables 9 and 10 present the results for greyscale and colour images, respectively. Values represent mean ± standard deviation over 10 trials with varying initial keys for the proposed method. The results are close the expected values.

TABLE 9 | NPCR and UACI results (greyscale images).

Image	Test	Proposed	Ref. [26]	Ref. [82]	Ref. [83]	Ref. [84]	Ref. [85]	Ref. [86]	Ref. [62]	Ref. [88]
Baboon	NPCR (%)	99.60 ± 0.016	99.61	99.63	99.62	99.61	60.28	99.61	99.60	99.64
	UACI (%)	33.42 ± 0.020	33.47	33.30	33.51	33.46	21.40	33.39	33.44	33.18
Peppers	NPCR (%)	99.60 ± 0.015	99.60	99.61	99.62	99.62	51.64	99.60	99.60	99.64
	UACI (%)	33.41 ± 0.014	33.52	33.42	33.46	33.35	20.31	33.45	33.45	33.49
Camera man	NPCR (%)	99.63 ± 0.005	99.61	—	99.60	99.62	76.26	99.61	99.60	99.64
	UACI (%)	33.50 ± 0.027	33.47	—	33.46	33.44	28.30	33.43	33.44	33.47
Boat	NPCR (%)	99.61 ± 0.012	—	—	—	—	—	—	—	—
	UACI (%)	33.54 ± 0.016	—	—	—	—	—	—	—	—

TABLE 10 | NPCR and UACI results (colour images).

Image	Test	Channel	Proposed	Ref. [63]	Ref. [87]
Baboon	NPCR (%)	R	99.60 ± 0.016	99.62	99.61
		G	99.59 ± 0.016	99.60	99.60
		B	99.61 ± 0.013	99.61	99.60
	UACI (%)	R	33.41± 0.021	33.51	33.46
		G	33.59 ± 0.011	33.53	33.43
		B	33.43 ± 0.040	33.49	33.46
Peppers	NPCR (%)	R	99.61 ± 0.010	99.60	99.60
		G	99.60 ± 0.016	99.59	99.60
		B	99.60 ± 0.015	99.61	99.61
	UACI (%)	R	33.46 ± 0.032	33.42	33.46
		G	33.50 ± 0.023	33.42	33.47
		B	33.47 ± 0.041	33.48	33.48

Consequently, our method achieves effective security against differential attacks.

5.3.2 | Plaintext Attacks Analysis

The traditional attacks of cryptanalysis are chosen ciphertext attacks, chosen-plaintext attacks, ciphertext-only attacks and known plaintext attacks. The chosen-plaintext attack types are the most hazardous, surpassing the other three types. Thus, if the encryption algorithm can resist chosen-plaintext attacks, it can also resist the other three types of attacks [96]. In the chosen-plaintext attack, the cryptanalyst endeavours to encrypt specific visuals to scrutinise their corresponding encrypted representations. Subsequently, the aim is to ascertain the secret key or approximate the decryption procedure to reconstitute the original image devoid of necessitating access to the secret key. To verify the resistance of the proposed method against chosen-plaintext attacks, several experimental analyses are conducted.

First, black and white image tests are commonly used because such images have uniform pixel values, which can disable or weaken the effect of the permutation process and make weaknesses in the encryption algorithm easier to detect. Table 11 shows the value of adjacent correlation, IE, NPCR and UACI for the white image and the black image. Figure 15 depicts the histogram for both white and black images. As evident, for the encrypted white image and the encrypted black image, the information entropy is close to the optimal value; the adjacent correlation is so close to 0; NPCR and UACI are close to the expected value, and the histogram is even. The obtained results indicate high resistance against chosen-plaintext attacks in terms of black image and white image attacks. The proposed DNA-DCP-DA ensures that the permutation and diffusion steps are closely linked, working together seamlessly. This guarantees that the permutation step can't be destroyed and protects against chosen plaintext attacks from specifically targeting the diffusion.

Second, the evaluation method used in Ref. [97] and Ref. [98] is adopted, where the security of the image cryptosystem is analysed using the following equation:

$$PI_1(i,j) \oplus PI_2(i,j) \neq CI_1(i,j) \oplus CI_2(i,j) \tag{34}$$

where PI_1 and PI_2 are two plain images, and CI_1, and CI_2 are their corresponding cipher images. For encryption schemes that rely on simple operations such as XOR, the cipher image can be expressed as $CI = PI \oplus K$, where PI is the plain image and K is the key. In this case, the variation between two plain images (PI_1, PI_2) can be directly deduced from their cipher images (CI_1, CI_2). Specifically, the difference between ciphertexts is as follows:

$$CI_1 \oplus C\,I_2 = (PI_1 \oplus K) \oplus (PI_2 \oplus K) = PI_1 \oplus PI_2 \tag{35}$$

As can be seen, the K is cancelled in Equation (35), and the attacker can compute the plain image difference from the cipher image difference without the need to recover K. This can create a serious security vulnerability in chosen-plaintext attack scenarios. As visualised in Figure 16, the results of the XOR operation on two plain images and their corresponding cipher images are obviously different, Thus, the vulnerability that is exploited by this type of attack is effectively disabled.

Third, the approach outlined in [50] is employed. In this approach, the input image (PI_1) is first divided into equally sized blocks. In each block, the first pixel contains a special pixel value (α), while the remaining pixels are filled with fixed values. This

TABLE 11 | Statistical results for black and white images.

Image	Information entropy (IE)	Correlation analysis			NPCR	UACI
		Horizontal	Vertical	Diagonal		
Black (1080, 256)	7.9993	0.00037	0.00100	−0.00497	99.62%	33.49%
White (256, 1080)	7.9994	−0.00316	0.00050	−0.00316	99.62%	33.45%

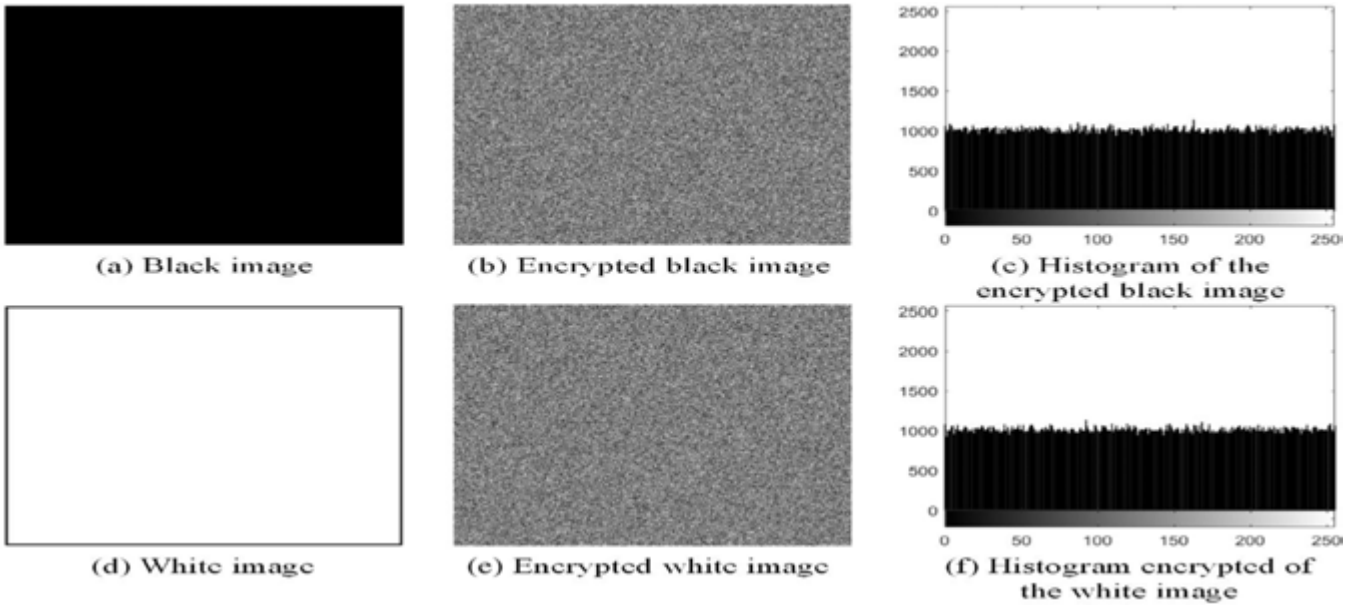

FIGURE 15 | Results of black and white image test.

FIGURE 16 | Results of the chosen plaintext attack method in [97]. (a) Plain peppers image (PI_1), (b) plain boat image (PI_2), (c) $PI_1 \oplus PI_2$ and (d) $CI_1 \oplus CI_2$.

image is encrypted to produce the first cipher image (CI_1). Next, two pixels in the block are modified: the pixel with the special value (α) is increased, while another is decreased by the same amount to preserve the overall sum of all pixels, resulting in a second plain image (PI_2). Preserving the sum of all pixels ensures that the key derived from this sum remains consistent, making the key identical even after modifying the first pixels. PI_2 is then encrypted to generate a second cipher image (CI_2). Lastly, the XOR of CI_1 and CI_2 reveals the affected pixel position, while the other pixels appear as zeros, thereby exposing the pixel's transformation. While the original method utilises 16×16 blocks in a 256×256 image, for illustration purposes, we adapt this approach to a 4×4 image without the need for block division. Furthermore, the initial key values are fixed when constructing CI_1 and I_2 . The main goal of this attack is to take advantage of the lack of an intricate relationship between the key and the plaintext to crack the encryption system. The results are presented in Figure 17. The result reveals that all cipher pixel values are non-zero, illustrating that the special pixel (α) transformation is not detected. Therefore, DNA-DCP-DA can effectively resist this type of chosen plaintext attack because it employs row-wise, column-wise and total pixel summation combined with a double-layer

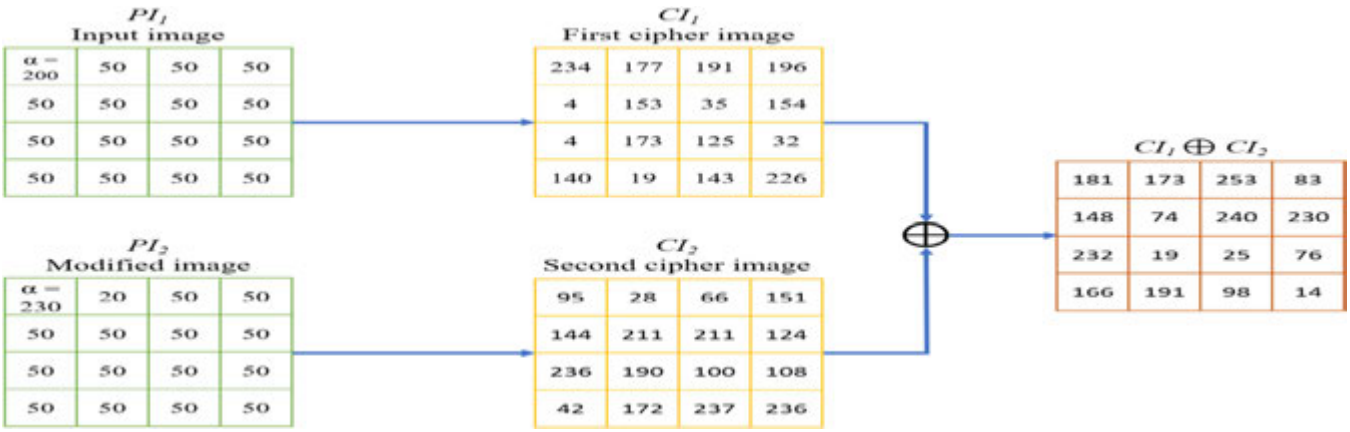

FIGURE 17 | Results of the chosen plaintext attack method in [50].

MD5 hash, making the relationship between the plain image and the generated key more difficult to detect.

Fourth, the proposed DNA-DCP-DA utilises four chaotic maps (S-SCM-LM, S-MCM-LM, S-STBCS-LM and S-CTBCS-LM), offering an exceptionally large key space. With such a vast key space, the encryption scheme becomes nearly impossible to break under a ciphertext-only attack [99].

5.3.3 | Security Analysis of the Proposed DNA Operations

To test the proposed DNA operations, the plain image is first transformed into a quaternary sequence. Then, the proposed encryption operations are directly applied to this quaternary representation using the DNA key. The first test analyses the distribution of the quaternary values after the operations are applied to verify whether the DNA-based operations effectively randomise the sequence by producing an equal distribution of 0, 1, 2 and 3. The second test involves converting the encrypted quaternary sequence back into pixel values to form the encrypted image and examining the histogram. Both of these tests provide a comprehensive evaluation of the proposed operations' ability to obscure both quaternary patterns and pixel intensity distributions. The results are shown in Figure 18, indicating that the proposed DNA operations produced ciphertexts with even histograms and uniform distributions of the quaternary symbols (0, 1, 2 and 3). This demonstrates that the proposed operations do not compromise security in terms of ciphertext randomness. Moreover, the proposed DNA operations are designed to be non-commutative. In contrast to commutative operations (e.g., DNA_{XOR} or DNA_{XNOR}, where $DNA_{XOR}(X,Y) = DNA_{XOR}(Y,X)$), the non-commutative operations (e.g., DNA_{RsH1}) depend on the operand order: $DNA_{RsH1}(X,Y) \neq DNA_{RsH1}(Y,X)$. This introduces operand-order dependencies, which increase resistance to linear cryptanalysis by complicating the relationships between the plaintext, ciphertext and key. This can also prevent algebraic recovery of the key. For example, let the plain image sequence be $PI = ACGT$ and the key be $K = AGCG$. Applying DNA_{XOR}, we obtain the cipher image $CI = DNA_{XOR}(PI,K) = ATTC$. If the attacker knows PI and CI, the K can be recovered as $K = DNA_{XOR}(PI,CI) = AGCG$. Now, consider applying the proposed non-commutative DNA operation DNA_{RsH1} with same PI and K, the ciphertext is $CI = DNA_{RsH1}(PI,K) = TTGC$. When attempting to recover the key using $K = DNA_{RsH1}(PI,CI)$ the result is $CAAT$; or using $K = DNA_{RsH1}(CI,PI)$, the result is $ATAA$. In both cases, the result does not match the actual key. This demonstrates that the non-commutative operations complicate algebraic key recovery based on known plaintext-ciphertext pairs. In addition to all the aforementioned advantages, the proposed operations can be applied using different DNA encoding rules. As a result, each operation can appear in eight possible forms depending on the chosen encoding rule. Generating new DNA operations can further empower dynamic DNA encryption schemes by providing a greater variety of operations. This is able to increase the complexity of potential cryptanalysis by making the operation set less predictable and more resistant to attacks.

5.3.4 | Permutation Test

The permutation process has the goal of preventing attackers from recognising image details. In this method, the permutation effect is evaluated by cropping a square from the middle of the encrypted image and replacing it with a white square. Then, visually observe how the white square spreads across the whole decrypted image. The result in Figure 19 shows that the white square is randomly spread across the whole decrypted image, illustrating the strong permutation effect of DNA-DCP-DA. Furthermore, to evaluate the separability of the concurrent permutation-diffusion process, we conducted a simulation in which the DNA operations were reversed (diffusion effect), while the permutation pattern remained unreversed. The result is shown in Figure 20. It reveals that the decrypted image remained visually unintelligible and exhibits a uniform histogram, providing robust evidence that the permutation step is critical and cannot be isolated from the diffused data.

5.4 | Noise and Data Loss Attacks Analysis

When images are electronically transmitted over long distances via wired or wireless channels, they are often degraded by noise or distortion [100]. Thus, the noise attacks and the data loss attacks are analysed in this subsection. The analysis of noise and data

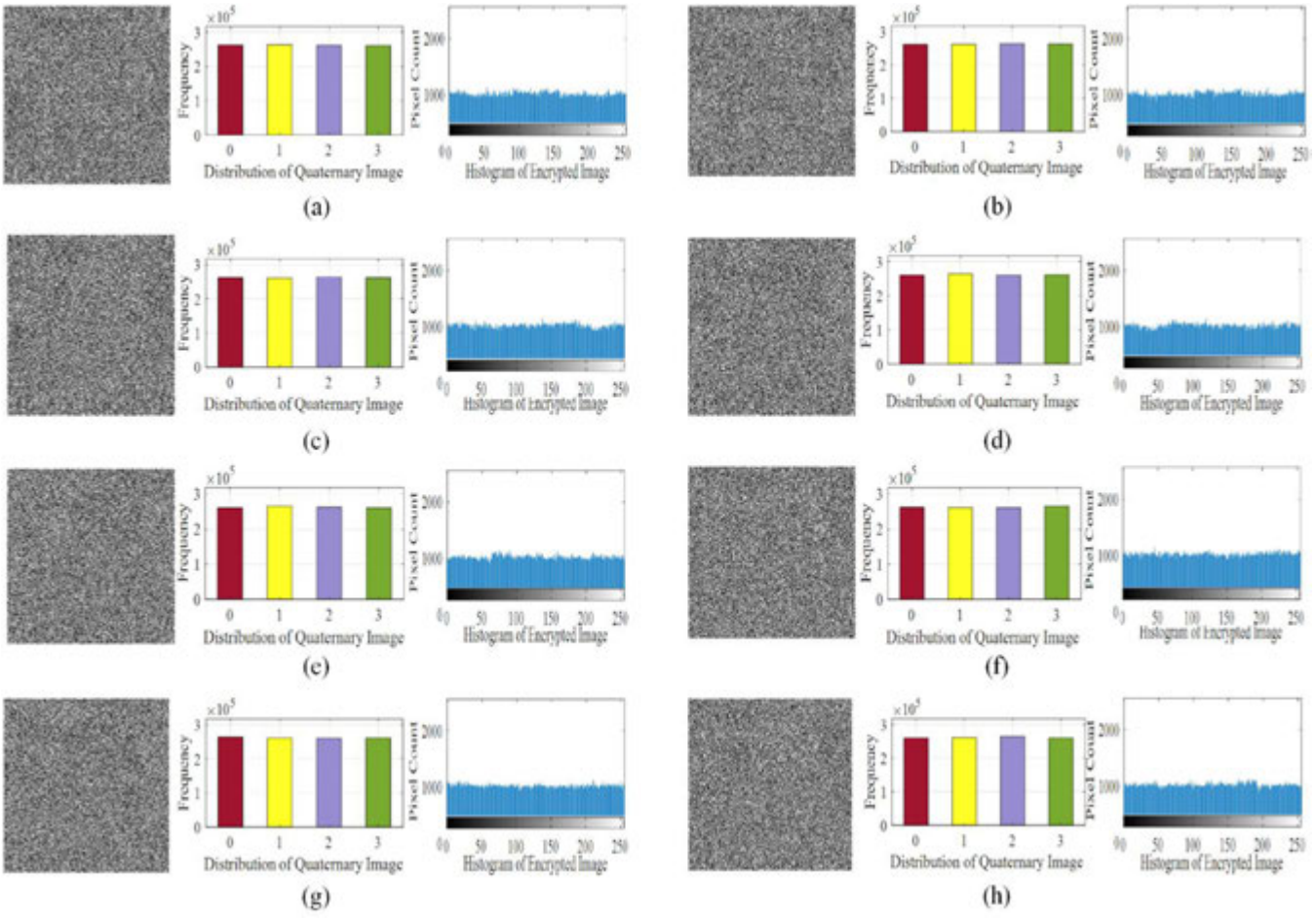

FIGURE 18 | Security analysis of the DNA operations. (a) DNA_{XOR}, (b) DNA_{XNOR}, (c) DNA_{H1}, (d) DNA_{H2}, (e) DNA_{RsH1}, (f) DNA_{LsH1}, (g) DNA_{RsH2} and (h) DNA_{LsH2}.

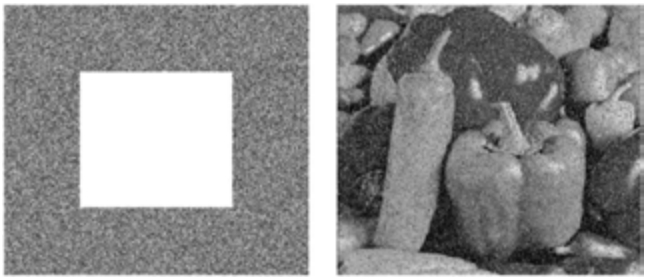

FIGURE 19 | Permutation test.

loss attacks is independent of chaotic sequences, it depends on the structure of the encryption algorithm regardless of whether it uses chaotic sequences.

5.4.1 | Noise Attacks

A strong algorithm should decrease the impact of noise during the decryption process when receiving the encrypted image. As a consequence of examining the strength of the proposed method toward the noise attack, the following procedure is performed: the input image is encrypted by the proposed method, and then various densities of noise are added to the encrypted image. After that, we tried to decrypt the encrypted image to recover the original image. Figure 21 depicts the results of adding noise with densities of 0.5%, 5% and 10% to the encrypted image. The decrypted image in Figure 21 is still recognisable. To further evaluate the effect of noise on the proposed DNA-DCP-DA, the quality metric peak signal-to-noise ratio (PSNR) between the plain images and the noisy decrypted images is computed, and the results are presented in Table 12. Visually, the results show that the decrypted noisy images preserve fair information. Moreover, the PSNR values demonstrate that the proposed method can still preserve essential image details.

5.4.2 | Data Loss Attacks

During the transmission, the encrypted image is prone to loss of data. Such a loss may lead to significant degradation of the recovered image during the decryption process. Consequently, minimising the effect of data loss is an important factor for the encryption algorithm. Diverse sizes of data are cropped from the encrypted images by replacing the data with zeros. Having obtained that, we tried to decrypt the encrypted images with data loss. Figure 22 shows the result with a 1/16 crop ratio, a 1/4 crop ratio and a 1/2 crop ratio. The PSNR is also calculated between the plain images and the decrypted images with data loss, and the results are listed in Table 13. The results demonstrate that

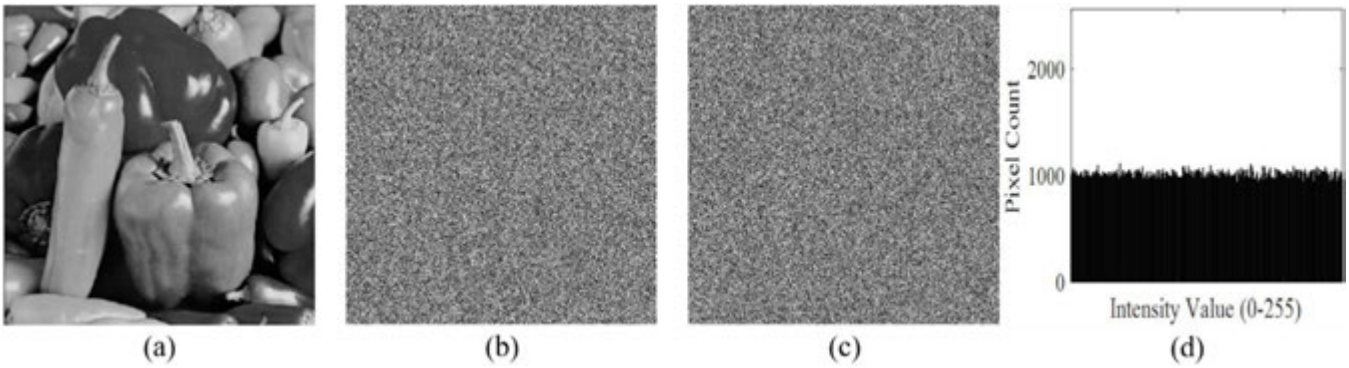

FIGURE 20 | Result of reversing diffusion with unknown permutation pattern. (a) Plain image, (b) encrypted image, (c) decrypted image with unknown permutation pattern and (d) histogram of decrypted image.

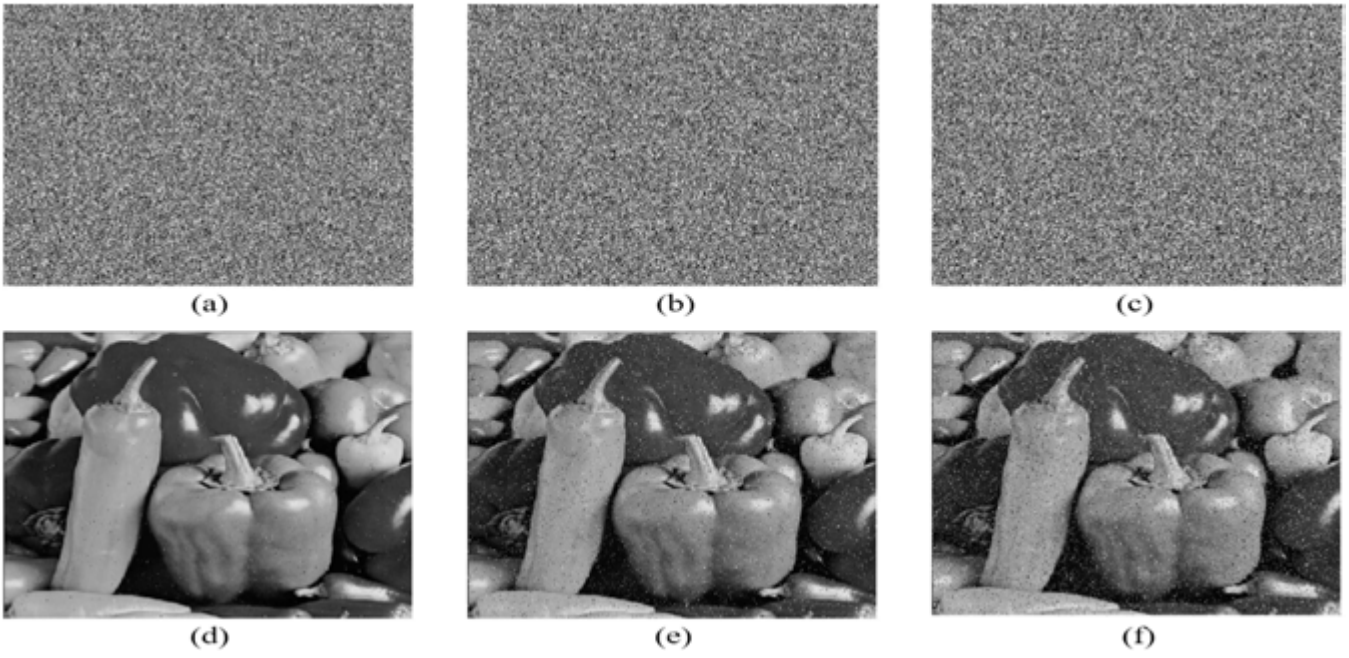

FIGURE 21 | Results of noise attacks. Encrypted images with (a) 0.5% noise ratio, (b) 5% noise ratio, (c) 10% noise ratio, (d) decrypted version of (a), (e) decrypted version of (b) and (f) decrypted version of (c).

the decrypted images retain meaningful information even when 50% of the pixels have lost data. As a consequence, the proposed DNA-DCP-DA is capable of enduring data loss attacks.

5.5 | Efficiency Analysis

This subsection evaluates the computational efficiency of the proposed encryption scheme. We report the overall execution time of the full algorithm and the resource performance of the KVM, including its execution time, CPU usage, and memory consumption.

5.5.1 | Execution Time

The speed of encryption is an important metric to evaluate encryption system performance in real-time applications. The experimental platform is MATLAB R2021a with an Intel Core i7-4600 CPU at 2.7 GHz and 8 GB RAM on the Windows 10 operating system. The comparisons for 256 × 256 greyscale and colour images are reported in Tables 14 and 15, respectively. The results of the proposed scheme without using the KVM are also reported in Tables 14 and 15, where the four chaotic maps are each iterated M × N × 4 times to generate four sequences that match the size of the DNA image. The results of the proposed algorithm have the lowest execution time compared to the other algorithms from other references. The computational complexity of the four chaotic maps without employing KVM is $4 \times O\ (M \times N \times 4) = O\ (16MN)$, whereas with the application of KVM, the complexity reduces to $O\ (M \times 4 + N \times 4 \times 2) = O\ (4M + 8N)$. Furthermore, Table 16 reports the results of the proposed algorithm with and without KVM for different image sizes. The results show that using KVM can noticeably decrease the execution time, especially for large-sized images, demonstrating the high effectiveness of KVM in reducing processing time.

TABLE 12 | PSNR between plain and noisy decrypted images.

Image	Noise ratio		
	0.5%	5%	10%
Baboon	32.2826	22.1547	19.2222
Peppers	31.4050	21.1797	18.2513
Camera man	31.3594	21.6653	18.5824
Boat	31.6853	22.0591	18.8860
Baboon (colour) R / G / B	31.7008 / 32.2111 / 31.0705	21.5242 / 21.9324 / 20.9668	18.5219 / 18.8499 / 18.181
Peppers (colour) R / G / B	31.7559 / 30.1450 / 30.4237	21.5861 / 20.4222 / 20.5174	18.4916 / 17.4455 / 17.6567

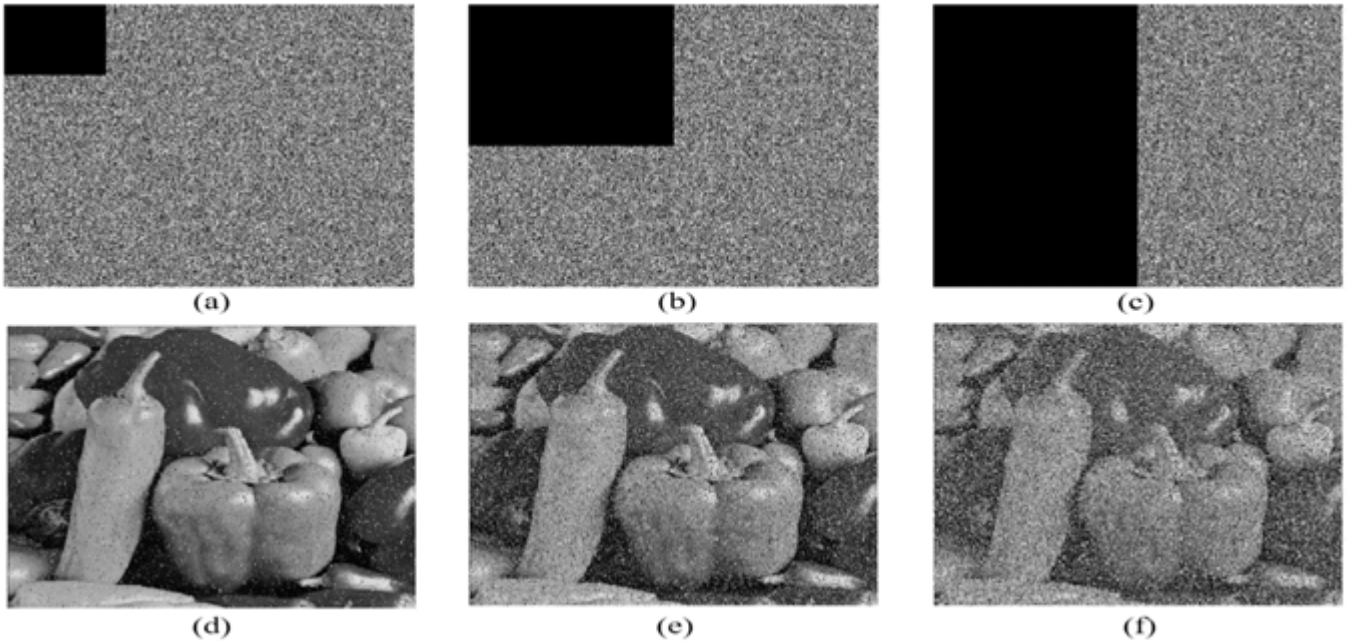

FIGURE 22 | Results of data loss attacks. Encrypted images with (a) 1/16 loss ratio, (b) 1/4 loss ratio, (c) 1/2 loss ratio, (d) decrypted version of (a), (e) decrypted version of (b) and (f) decrypted version of (c).

TABLE 13 | PSNR between plain and decrypted images with data loss.

Image	Loss ratio		
	1/16	1/4	1/2
Baboon	21.1972	15.2474	12.3351
Peppers	20.2968	14.3326	11.3471
Camera man	20.6660	14.6532	11.5483
Boat	21.0694	15.0528	12.1249
Baboon (colour) R / G / B	20.6029 / 20.9307 / 20.1477	14.6266 / 14.9466 / 14.1420	11.6574 / 12.032 / 11.2134
Peppers (colour) R / G / B	20.5547 / 19.5405 / 19.6847	14.6679 / 13.5111 / 13.6647	11.7742 / 10.548 / 10.6558

5.5.2 | KVM Execution and Resource Utilisation

The key sequence generation step is isolated and analysed independently to evaluate the computational impact of the KVM on the overall process. The performance of this step is measured using three primary indicators: execution time, average CPU usage during execution and memory utilisation. These metrics provide a comprehensive view of its efficiency. The results for the key sequence generation step with KVM are presented in Table 17, along with the results of the traditional chaotic map iterations

TABLE 14 | Execution time (s) for a 256 × 256 greyscale image.

Algorithm	Proposed	Proposed without using KVM	Ref. [26]	Ref. [82]	Ref. [83]	Ref. [84]	Ref. [86]	Ref. [62]	Ref. [88]
Encryption	0.1255	0.3276	0.2177	0.6007	2.3545	1.5210	4.3920	—	—
Decryption	0.2227	0.3301	0.3551	0.3804	2.3296	1.4506	3.6577	—	—
Encryption per pixel	1.91×10^{-6}	4.99×10^{-6}	3.32×10^{-6}	9.16×10^{-6}	35.92×10^{-6}	23.20×10^{-6}	67.01×10^{-6}	—	—
Decryption per pixel	3.39×10^{-6}	5.03×10^{-6}	5.41×10^{-6}	5.80×10^{-6}	35.54×10^{-6}	22.13×10^{-6}	55.81×10^{-6}	—	—
Processor speed	2.70 GHz	2.70 GHz	2.60 GHz	3.00 GHz	2.60 GHz	2.60 GHz	2.60 GHz	—	—
Platform	MATLAB	MATLAB	MATLAB	MATLAB	MATLAB	MATLAB	MATLAB	—	—

TABLE 15 | Execution time (s) for a 256 × 256 colour image.

Algorithm	Proposed	Proposed without using KVM	Ref. [63]	Ref. [87]
Encryption	0.3003	0.74359	1.24	—
Decryption	0.3342	0.70592	1.28	—
Encryption per pixel	1.52×10^{-6}	3.78×10^{-6}	6.30×10^{-6}	—
Decryption per pixel	1.69×10^{-6}	3.59×10^{-6}	6.51×10^{-6}	—
Processor speed	2.70 GHz	2.70 GHz	1.61 GHz	—
Platform	MATLAB	MATLAB	MATLAB	—

TABLE 16 | Execution time (s) for different image sizes.

		256 × 256		512 × 512		1080 × 1080		2160 × 2160	
Image size		Greyscale	Colour	Greyscale	Colour	Greyscale	Colour	Greyscale	Colour
Proposed	Encryption	0.1255	0.3003	0.45529	1.3376	1.8862	5.7534	7.9068	24.1369
	Decryption	0.2227	0.33278	0.46159	1.3398	1.9784	5.8557	7.9814	24.2499
Proposed without using KVM	Encryption	0.3276	0.74359	2.8490	3.4123	8.9832	12.7394	37.1597	54.2940
	Decryption	0.3301	0.70592	2.4265	3.2701	8.8845	12.6784	36.2785	53.9390

method (without using KVM) for comparison. The results show that the method with KVM achieves significant acceleration with decreased memory utilisation. However, a corresponding increase in CPU usage was observed during KVM execution. This is because KVM effectively leverages more processing cores simultaneously to achieve faster computation. As a result, the processor works harder during execution but completes the execution in a shorter time. Moreover, GPU execution of the KVM is also investigated to further evaluate the parallelisation efficiency of the proposed KVM. The GPU used in this study is the NVIDIA GeForce RTX 4060 Laptop GPU. Table 18 shows the comparison of resource utilisation on CPU and GPU platforms. The results in Table 18 show that GPU execution for generating the four key sequences using chaotic maps was actually slower than CPU execution. This is because the nature of chaotic maps requires sequential iterations, where the key generation process demands M iterations for row-dependent key sequences and $4 \times N$ iterations column-dependent sequences, with each next state $x\ (i+1)$ depending on the current state $x\ (i)$. This data dependency significantly limits the parallelism of GPUs, as GPUs are most efficient when independent computations can be distributed across many cores simultaneously. In contrast, GPU execution of the KVM shows significantly improved parallel efficiency and provides faster execution compared to the CPU. This is because the KVM eliminates sequential dependencies and employs vectorised operations, such as modular shifting and sorting, which GPUs can process in parallel. These findings confirm that the KVM is inherently more suitable for GPU acceleration.

6 | Conclusion

The new scaling chaotification models are defined by adjusting the existing chaotification models that mainly use sine, cosine, or module functions. The performance analysis of the proposed models proves that the periodic windows are eliminated, meaning the chaotic performance cannot be degraded to nonchaotic,

TABLE 17 | KVM resource utilisation.

Sequence size		256 × 256	512 × 512	1080 × 1080	2160 × 2160
Execution time for generating four sequences (s)	Using KVM	0.00700	0.015391	0.07431	0.27739
	Using chaotic map iterations	0.2871	1.6003	8.0330	29.2544
Average CPU usage (%)	Using KVM	35	35	34	37
	Using chaotic map iterations	25	25.91	21.57	26.08
Memory usage (mb)	Using KVM	8.405	32.9769	149.2419	597.2787
	Using chaotic map iterations	8.405	33.5012	268.3863	1069.1338

TABLE 18 | Comparison of KVM resource utilisation (CPU versus GPU).

Sequence size	256 × 256	512 × 512	1080 × 1080	2160 × 2160
CPU execution time for generating four sequences (s)	0.007	0.015	0.074	0.277
GPU execution time for generating four sequences (s)	0.88	2.14	3.85	8.33
CPU execution time for KVM (s)	0.00460	0.010134	0.06940	0.26565
GPU execution time for KVM (s)	0.00196	0.003462	0.01149	0.04445
Average CPU usage (%)	35	35	34	37
Average GPU usage (%)	16.6	17.7	22.2	24.2

and the values of LE and entropy are increased. Increasing the LE and entropy results means increasing the sensitivity of the initial conditions, and that enhances unpredictability, expands the key space, secures the key generation and enhances the encryption step. These models are directly involved in generating secret keys for the proposed DNA-DCP-DA. We utilised the concurrent permutation-diffusion method to perform permutation and diffusion steps simultaneously, preventing targeted attacks on the diffusion step and increasing efficiency. We also define new DNA computing operations, and the performance analysis demonstrates the efficiency of the new DNA operations. Moreover, the experimental histogram is even; the key space is enormous, reaching $2^{697.4}$; the correlation analysis is near 0; the IE is larger than 7.999, and noise and data loss attacks on the encrypted image are ineffective. The NPCR is greater than 99.60% and the UACI is greater than 33.46%. Many experiments validate the endurance against chosen-plaintext attacks. Therefore, the results prove that our algorithm achieves high-level security outcomes. Having worked with the major aim of beating the shortcomings of consuming an excessive amount of time and computational overhead, we defined a new KVM to reduce the computational overhead and the long execution time to achieve 0.1255 s for a 256 × 256 image by minimising the number of iterations of the chaotic maps. Lower computational complexity and execution time aid the applicability of the image encryption system based on DNA technique in real-time applications that allow fast data protection along with maintaining strong security, such as image transmission security, medical image security, cloud storage security, digital watermarking security and IoT device security. Future work will focus on analysing the proposed scheme's robustness against AI-based cryptanalysis, investigating the impact of finite precision and rounding errors in hardware implementations, and conducting a comparative study of various scaling functions to develop more effective chaotification models.

Author Contributions

Mustafa Kamil Khairullah: conceptualisation, data curation, formal analysis, software, visualisation, writing – original draft, **Mohd Zafri Bin Baharuddin:** conceptualisation, methodology, supervision, writing – review and editing, **Reema Thabit:** conceptualisation, supervision, validation, **Mohammad Alomari:** validation, visualisation, writing – review and editing, **Gamal Alkawsi:** project administration, validation, writing – review and editing, **Faten Saif:** writing – review and editing.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The datasets used in this study are publicly available from the Computer Vision Group (CVG) at the University of Granada (https://ccia.ugr.es/cvg/dbimagenes/) and the Signal and Image Processing Institute (SIPI) at the University of Southern California (https://sipi.usc.edu/database/).

References

1. "Expected Cost of Cybercrime Until 2027," Statista, published May 30, 2023, https://www.statista.com/chart/28878/expected-cost-of-cybercrime-until-2027/.

2. M. A. Alomari, M. N. Al-Andoli, M. Ghaleb, et al., "Security of Smart Grid: Cybersecurity Issues," *Potential Cyberattacks, Major Incidents, and Future Directions Energies* 18, no. 1 (2025): 141, https://doi.org/10.3390/en18010141.

3. P. Mahadevappa, R. K. Murugesan, R. Al-Amri, R. Thabit, A. H. Al-Ghushami, and G. Alkawsi, "A Secure Edge Computing Model Using Machine Learning and IDS to Detect and Isolate Intruders," *MethodsX* 12 (2024): 102597, https://doi.org/10.1016/j.mex.2024.102597.

4. X. Y. Wang, L. Yang, R. Liu, and A. Kadir, "A Chaotic Image Encryption Algorithm Based on Perceptron Model," *Nonlinear Dynamics* 62 (2010): 615–621, https://doi.org/10.1007/s11071-010-9749-8.

5. H. Liu, X. Wang, and A. Kadir, "Colour Image Encryption Using Choquet Fuzzy Integral and Hyper Chaotic System," *Optik-International Journal for Light and Electron Optics* 124, no. 18 (2013): 3527–3533, https://doi.org/10.1016/j.ijleo.2012.10.068.

6. M. Vijayakumar and A. Ahilan, "An Optimized Chaotic S-Box for Real-Time Image Encryption Scheme Based on 4-Dimensional Memristive Hyperchaotic Map," *Ain Shams Engineering Journal* 15, no. 4 (2024): 102620, https://doi.org/10.1016/j.asej.2023.102620.

7. X. Jiang, G. Jiang, Q. Wang, and D. Shu, "Image Encryption Algorithm Based on 2D-CLICM Chaotic System," *IET Image Processing* 17, no. 7 (2023): 2127–2141, https://doi.org/10.1049/icp.2024.2825.

8. K. M. Hosny, Y. M. Elnabawy, A. M. Elshewey, S. M. Alhammad, D. S. Khafaga, and R. Salama, "New Method of Colour Image Encryption Using Triple Chaotic Maps," *IET Image Processing* 18 (2024): 3262–3276, https://doi.org/10.1049/ipr2.13171.

9. R. Thabit, N. I. Udzir, S. M. Yasin, A. Asmawi, and A. A. A. Gutub, "CSNTSteg: Colour Spacing Normalization Text Steganography Model to Improve Capacity and Invisibility of Hidden Data," *IEEE Access* 10 (2022): 65439–65458, https://doi.org/10.1109/ACCESS.2022.3182712.

10. Y. Wan, S. Gu, and B. Du, "A New Image Encryption Algorithm Based on Composite Chaos and Hyperchaos Combined With DNA Coding," *Entropy* 22, no. 2 (2020): 171, https://doi.org/10.3390/e22020171.

11. R. Li, Q. Liu, and L. Liu, "Novel Image Encryption Algorithm Based on Improved Logistic Map," *IET Image Processing* 13, no. 1 (2019): 125–134, https://doi.org/10.1049/iet-ipr.2018.5900.

12. A. Belazi, S. Kharbech, M. N. Aslam, et al., "Improved Sine-Tangent Chaotic Map With Application in Medical Images Encryption," *Journal of Information Security and Applications* 66 (2022): 103131, https://doi.org/10.1016/j.jisa.2022.103131.

13. Q. Liang and C. Zhu, "A New One-Dimensional Chaotic Map for Image Encryption Scheme Based on Random DNA Coding," *Optics & Laser Technology* 160 (2023): 109033, https://doi.org/10.1016/j.optlastec.2022.109033.

14. Q. Wu, "Cascade-Sine Chaotification Model for Producing Chaos," *Nonlinear Dynamics* 106 (2021): 2607–2620, https://doi.org/10.1007/s11071-021-06885-3.

15. W. Liu, K. Sun, H. Wang, and B. Li, "The Modular Modulation Chaotification Map and Its Hardware Implementation," *IEEE Transactions on Instrumentation and Measurement* 73 (2024): 1–9, https://doi.org/10.1109/tim.2024.3368470.

16. N. Charalampidis, C. Volos, L. Moysis, and I. Stouboulos, "A Chaotification Model Based on Modulo Operator and Secant Functions for Enhancing Chaos," *Chaos Theory and Applications* 4, no. 4 (2022): 274–284, https://doi.org/10.51537/chaos.1214569.

17. B. Ge, X. Chen, G. Chen, and Z. Shen, "Secure and Fast Image Encryption Algorithm Using Hyper-Chaos-Based Key Generator and Vector Operation," *IEEE Access* 9 (2021): 137635–137654, https://ieeexplore.ieee.org/document/9562561.

18. R. Matthews, "On the Derivation of a "Chaotic" Encryption Algorithm," *Cryptologia* 13, no. 1 (1989): 29–42, https://doi.org/10.1080/0161-118991863745.

19. A. Al-Hyari, C. Obimbo, and I. Altaharwa, "Generating Powerful Encryption Keys for Image Cryptography With Chaotic Maps by Incorporating Collatz Conjecture," *IEEE Access* 12 (2024): 4825–4844, https://doi.org/10.1109/access.2024.3349470.

20. D. Herbadji, A. Belmeguenai, N. Derouiche, and H. Liu, "Colour Image Encryption Scheme Based on Enhanced Quadratic Chaotic Map," *IET Image Processing* 14, no. 1 (2020): 40–52, https://doi.org/10.1049/iet-ipr.2019.0123.

21. H. Çelik and N. Doğan, "A Hybrid Colour Image Encryption Method Based on Extended Logistic Map," *Multimedia Tools and Applications* 83, no. 5 (2024): 12627–12650, https://doi.org/10.1007/s11042-023-16215-x.

22. J. X. Chen, Z. L. Zhu, L. B. Zhang, C. Fu, and H. Yu, "An Efficient Diffusion Scheme for Chaos-Based Digital Image Encryption," *Mathematical Problems in Engineering* 2014, no. 1 (2014): 427349, https://doi.org/10.1155/2014/427349.

23. P. Li, J. Qian, and T. T. Xu, "New Chaotic Systems and Application in DNA Coloured Image Encryption," *Multimedia Tools and Applications* 83, no. 17 (2024): 50023–50045, https://doi.org/10.1007/s11042-023-17605-x.

24. H. Liu and X. Wang, "Image Encryption Using DNA Complementary Rule and Chaotic Maps," *Applied Soft Computing* 12, no. 5 (2012): 1457–1466, https://doi.org/10.1016/j.asoc.2012.01.016.

25. X. Chai, X. Fu, Z. Gan, Y. Lu, and Y. Chen, "A Colour Image Cryptosystem Based on Dynamic DNA Encryption and Chaos," *Signal Processing* 155 (2019): 44–62, https://doi.org/10.1016/j.sigpro.2018.09.029.

26. E. Z. Zefreh, "An Image Encryption Scheme Based on a Hybrid Model of DNA Computing, Chaotic Systems and Hash Functions," *Multimedia Tools and Applications* 79, no. 33 (2020): 24993–25022, https://doi.org/10.1007/s11042-020-09111-1.

27. R. Liu, S. Wang, Z. Wang, X. Gong, S. Zhang, and J. Zhou, "A New Colour Image Encryption Scheme Based on DNA Sequence Operations and Novel Hyper-Chaotic System," *Physica Scripta* 100 (2025): 025025, https://10.1088/1402-4896/ada098.

28. H. Wen and Y. Lin, "Cryptanalyzing an Image Cipher Using Multiple Chaos and DNA Operations," *Journal of King Saud University-Computer and Information Sciences* 35, no. 7 (2023): 101612, https://doi.org/10.1016/j.jksuci.2023.101612.

29. L. M. Adleman, "Molecular Computation of Solutions to Combinatorial Problems," *Science* 266, no. 5187 (1994): 1021–1024, https://doi.org/10.1126/science.7973651.

30. Q. Zhang, L. Liu, and X. Wei, "Improved Algorithm for Image Encryption Based on DNA Encoding and Multi-Chaotic Maps," *AEU-International Journal of Electronics and Communications* 68, no. 3 (2014): 186–192, https://doi.org/10.1016/j.aeue.2013.08.007.

31. W. Wen, K. Wei, Y. Zhang, Y. Fang, and M. Li, "Colour Light Field Image Encryption Based on DNA Sequences and Chaotic Systems," *Nonlinear Dynamics* 99 (2020): 1587–1600, https://doi.org/10.1007/s11071-019-05378-8.

32. 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* 162 (2022): 112456, https://doi.org/10.1016/j.chaos.2022.112456.

33. J. Yu, K. Peng, L. Zhang, and W. Xie, "Image Encryption Algorithm Based on DNA Network and Hyperchaotic System," *The Visual Computer* 40 (2024): 8001–8021, https://doi.org/10.1007/s00371-023-03219-9.

34. E. Z. Zefreh, "PSDCLS: Parallel Simultaneous Diffusion–Confusion Image Cryptosystem Based on Latin Square," *Journal of Information Security and Applications* 83 (2024): 103785, https://doi.org/10.1016/j.jisa.2024.103785.

35. L. C. Cao, Y. L. Luo, S. H. Qiu, and J. X. Liu, "A Perturbation Method to the Tent Map Based on Lyapunov Exponent and Its Application," *Chinese Physics B* 24, no. 10 (2015): 100501, https://doi.org/10.1088/1674-1056/24/10/100501.

36. Y. P. K. Nkandeu, J. R. Mboupda Pone, and A. Tiedeu, "Image Encryption Algorithm Based on Synchronized Parallel Diffusion and New Combinations of 1D Discrete Maps," *Sensing and Imaging* 21 (2020): 1–36, https://10.1007/s11220-020-00318-y.

37. Z. Zhang, H. Zhu, P. Ban, Y. Wang, and L. Y. Zhang, "Buffeting Chaotification Model for Enhancing Chaos and Its Hardware Implementation," *IEEE Transactions on Industrial Electronics* 70, no. 3 (2022): 2916–2926, https://doi.org/10.1109/tie.2022.3174288.

38. F. Yuan, Y. Deng, Y. Li, and G. Chen, "A Cascading Method for Constructing New Discrete Chaotic Systems With Better Randomness," *Chaos: An Interdisciplinary Journal of Nonlinear Science* 29, no. 5 (2019): 053120, https://doi.org/10.1063/1.5094936.

39. Y. Li, X. He, and W. Zhang, "The Fractional Difference Form of Sine Chaotification Model," *Chaos, Solitons & Fractals* 137 (2020): 109774, https://doi.org/10.1016/j.chaos.2020.109774.

40. M. Alawida, A. Samsudin, and J. S. Teh, "Digital Cosine Chaotic Map for Cryptographic Applications," *IEEE Access* 7 (2019): 150609–150622, https://doi.org/10.1109/access.2019.2947561.

41. D. Li, J. Li, and X. Di, "A Novel Exponential One-Dimensional Chaotic Map Enhancer and Its Application in an Image Encryption Scheme Using Modified ZigZag Transform," *Journal of Information Security and Applications* 69 (2022): 103304, https://doi.org/10.1016/j.jisa.2022.103304.

42. M. Alawida, "Enhancing Logistic Chaotic Map for Improved Cryptographic Security in Random Number Generation," *Journal of Information Security and Applications* 80 (2024): 103685, https://doi.org/10.1016/j.jisa.2023.103685.

43. C. Yang, X. Wei, and C. Wang, "S-Box Design Based on 2D Multiple Collapse Chaotic Map and Their Application in Image Encryption," *Entropy* 23, no. 10 (2021): 1312, https://doi.org/10.3390/e23101312.

44. Q. Lai, G. Hu, U. Erkan, and A. Toktas, "A Novel Pixel-Split Image Encryption Scheme Based on 2D Salomon Map," *Expert Systems With Applications* 213 (2023): 118845, https://doi.org/10.1016/j.eswa.2022.118845.

45. Z. Xu, E. Zheng, H. Han, X. Dong, X. Dang, and Z. Wang, "A Secure Healthcare Data Sharing Scheme Based on Two-Dimensional Chaotic Mapping and Blockchain," *Scientific Reports* 14, no. 1 (2024): 23470, https://doi.org/10.1038/s41598-024-73554-x.

46. Z. Man, Y. Dai, T. Liu, and X. Meng, "Exploration of AI Security: Application of a Novel Chaotic Map 2D-SCD in the Two-Dimensional Spatial Mirror Projection Algorithm," *Optics Communications* 591 (2025): 132023, https://doi.org/10.1016/j.optcom.2025.132023.

47. C. Song and Y. Qiao, "A Novel Image Encryption Algorithm Based on DNA Encoding and Spatiotemporal Chaos," *Entropy* 17, no. 10 (2015): 6954–6968, https://doi.org/10.3390/e17106954.

48. H. Wen, S. Yu, and J. Lü, "Breaking an Image Encryption Algorithm Based on DNA Encoding and Spatiotemporal Chaos," *Entropy* 21, no. 3 (2019): 246, https://doi.org/10.3390/e21030246.

49. A. Jain and N. Rajpal, "A Robust Image Encryption Algorithm Resistant to Attacks Using DNA and Chaotic Logistic Maps," *Multimedia Tools and Applications* 75 (2016): 5455–5472, https://doi.org/10.1007/s11042-015-2515-7.

50. Y. Dou, X. Liu, H. Fan, and M. Li, "Cryptanalysis of a DNA and Chaos Based Image Encryption Algorithm," *Optik* 145 (2017): 456–464, https://doi.org/10.1016/j.ijleo.2017.08.050.

51. J. Wu, X. Liao, and B. Yang, "Image Encryption Using 2D Hénon-Sine Map and DNA Approach," *Signal Processing* 153 (2018): 11–23, https://doi.org/10.1016/j.sigpro.2018.06.008.

52. J. Chen, L. Chen, and Y. Zhou, "Cryptanalysis of a DNA-Based Image Encryption Scheme," *Information Sciences* 520 (2020): 130–141, https://doi.org/10.1016/j.ins.2020.02.024.

53. X. Zhang, Z. Zhou, and Y. Niu, "An Image Encryption Method Based on the Feistel Network and Dynamic DNA Encoding," *IEEE Photonics Journal* 10, no. 4 (2018): 1–14, https://doi.org/10.1109/jphot.2018.2859257.

54. W. Feng, Z. Qin, J. Zhang, and M. Ahmad, "Cryptanalysis and Improvement of the Image Encryption Scheme Based on Feistel Network and Dynamic DNA Encoding," *IEEE Access* 9 (2021): 145459–145470, https://doi.org/10.1109/access.2021.3123571.

55. Z. Liu, C. Wu, J. Wang, and Y. Hu, "A Colour Image Encryption Using Dynamic DNA and 4-D Memristive Hyper-Chaos," *IEEE Access* 7 (2019): 78367–78378, https://doi.org/10.1109/access.2019.2922376.

56. J. Hao, H. Li, H. Yan, and J. Mou, "A New Fractional Chaotic System and Its Application in Image Encryption With DNA Mutation," *IEEE Access* 9 (2021): 52364–52377, https://doi.org/10.1109/access.2021.3069977.

57. Y. Hui, H. Liu, and P. Fang, "A DNA Image Encryption Based on a New Hyperchaotic System," *Multimedia Tools and Applications* 82, no. 14 (2023): 21983–22007, https://doi.org/10.1007/s11042-021-10526-7.

58. N. H. Sharkawy, Y. M. Afify, W. Gad, and N. Badr, "Gray-Scale Image Encryption Using DNA Operations," *IEEE Access* 10 (2022): 63004–63019, https://doi.org/10.1109/access.2022.3182329.

59. C. Zou, X. Wang, C. Zhou, S. Xu, and C. Huang, "A Novel Image Encryption Algorithm Based on DNA Strand Exchange and Diffusion," *Applied Mathematics and Computation* 430 (2022): 127291, https://doi.org/10.1016/j.amc.2022.127291.

60. Q. Wang, X. Zhang, and X. Zhao, "Colour Image Encryption Algorithm Based on Bidirectional Spiral Transformation and DNA Coding," *Physica Scripta* 98, no. 2 (2023): 025211, https://doi.org/10.1088/1402-4896/acb322.

61. L. Li, "Image Encryption Algorithm Based on Hyperchaos and DNA Coding," *IET Image Processing* 18, no. 3 (2024): 627–649, https://doi.org/10.1049/ipr2.12974.

62. D. Ustun, S. Sahinkaya, and N. Atli, "Developing a Secure Image Encryption Technique Using a Novel S-Box Constructed Through Real-Coded Genetic Algorithm's Crossover and Mutation Operators," *Expert Systems With Applications* 256 (2024): 124904, https://doi.org/10.1016/j.eswa.2024.124904.

63. F. Q. Meng and G. Wu, "A Colour Image Encryption and Decryption Scheme Based on Extended DNA Coding and Fractional-Order 5D Hyper-Chaotic System," *Expert Systems With Applications* 254 (2024): 124413, https://doi.org/10.1016/j.eswa.2024.124413.

64. X. Xue, H. Jin, and C. Zhou, "Compressive Sensing and DNA Coding Operation: Revolutionary Approach to Colour Medical Image Compression-Encryption Algorithm," *IET Image Processing* 18, no. 14 (2024): 4589–4606, https://doi.org/10.1049/ipr2.13270.

65. C. Zhao, Z. Zhai, and B. Zeng, "Delayed Chaotic Image Encryption Algorithm Using Cross-Layer and DNA Coding Techniques," *Computer Standards & Interfaces* 93 (2025): 103974, https://doi.org/10.1016/j.csi.2025.103974.

66. Z. Hua, B. Zhou, and Y. Zhou, "Sine Chaotification Model for Enhancing Chaos and Its Hardware Implementation," *IEEE Transactions on Industrial Electronics* 66, no. 2 (2018): 1273–1284, https://doi.org/10.1109/tie.2018.2833049.

67. Z. Hua, B. Zhou, Y. Zhang, and Y. Zhou, "Modular Chaotification Model With FPGA Implementation," *Science China Technological Sciences* 64, no. 7 (2021): 1472–1484, https://doi.org/10.1007/s11431-020-1717-1.

68. Z. Hua, B. Zhou, and Y. Zhou, "Sine-Transform-Based Chaotic System With FPGA Implementation," *IEEE Transactions on Industrial Electronics* 65, no. 3 (2017): 2557–2566, https://doi.org/10.1109/tie.2017.2736515.

69. Z. Hua, Y. Zhou, and H. Huang, "Cosine-Transform-Based Chaotic System for Image Encryption," *Information Sciences* 480 (2019): 403–419, https://doi.org/10.1016/j.ins.2018.12.048.

70. R. Wang, M. Y. Li, and H. J. Luo, "Exponential Sine Chaotification Model for Enhancing Chaos and Its Hardware Implementation," *Chinese Physics B* 31, no. 8 (2022): 080508, https://doi.org/10.1088/1674-1056/ac6335.

71. Y. Zhou, L. Bao, and C. P. Chen, "A New 1D Chaotic System for Image Encryption," *Signal Processing* 97 (2014): 172–182, https://doi.org/10.1016/j.sigpro.2013.11.012.

72. X. Wang and M. Wang, "A Hyperchaos Generated From Lorenz System," *Physica A: Statistical Mechanics and Its Applications* 387, no. 14 (2008): 3751–3758, https://doi.org/10.1016/j.physa.2008.02.020.

73. G. C. Wu and D. Baleanu, "Jacobian Matrix Algorithm for Lyapunov Exponents of the Discrete Fractional Maps," *Communications in Nonlinear Science and Numerical Simulation* 22, no. 1-3 (2015): 95–100, https://doi.org/10.1016/j.cnsns.2014.06.042.

74. J. S. Richman, D. E. Lake, and J. R. Moorman, "Sample Entropy," in *Methods in Enzymology* (Academic Press, 2004), 172–184.

75. P. Grassberger and I. Procaccia, "Estimation of the Kolmogorov Entropy From a Chaotic Signal," *Physical Review A* 28, no. 4 (October 1983): 2591–2593, https://doi.org/10.1103/physreva.28.2591.

76. N. N. Abdullah, N. H. Zakaria, A.b Halim, et al., "A Theoretical Comparative Analysis of DNA Techniques Used in Dna Based Cryptography," *Journal of Sustainability Science and Management* 17, no. 5 (2022): 165–178, https://doi.org/10.46754/jssm.2022.05.014.

77. X. Chai, Z. Gan, Y. Lu, Y. Chen, and D. Han, "A Novel Image Encryption Algorithm Based on the Chaotic System and DNA Computing," *International Journal of Modern Physics C* 28, no. 05 (2017): 1750069, https://doi.org/10.1142/s0129183117500693.

78. T. Head, G. Rozenberg, R. S. Bladergroen, C. K. D. Breek, P. H. M. Lommerse, and H. P. Spaink, "Computing With DNA by Operating on Plasmids," *Bio Systems* 57, no. 2 (2000): 87–93, https://doi.org/10.1016/s0303-2647(00)00091-5.

79. J. Daemen and V. Rijmen, "The Design of Rijndael: AES – The Advanced Encryption Standard," in *Information Security and Cryptography* (Springer, 2002), 1–238, https://doi.org/10.1007/978-3-662-04722-4.

80. U. Erkan, A. Toktas, S. Memiş, Q. Lai, and G. Hu, "An Image Encryption Method Based on Multi-Space Confusion Using Hyperchaotic 2D Vincent Map Derived From Optimization Benchmark Function," *Nonlinear Dynamics* 111, no. 21 (2023): 20377–20405, https://doi.org/10.1007/s11071-023-08859-z.

81. R. Rivest, "The MD5 Message-Digest Algorithm," *Internet Engineering Task Force*, April 1, 1992.

82. S. Zhu, X. Deng, W. Zhang, and C. Zhu, "Image Encryption Scheme Based on Newly Designed Chaotic Map and Parallel DNA Coding," *Mathematics* 11, no. 1 (2023): 231, https://doi.org/10.3390/math11010231.

83. X. Chai, Z. Gan, K. Yuan, Y. Chen, and X. Liu, "A Novel Image Encryption Scheme Based on DNA Sequence Operations and Chaotic Systems," *Neural Computing and Applications* 31 (2019): 219–237, https://doi.org/10.1007/s00521-017-2993-9.

84. T. Hu, Y. Liu, L. H. Gong, and C. J. Ouyang, "An Image Encryption Scheme Combining Chaos With Cycle Operation for DNA Sequences," *Nonlinear Dynamics* 87 (2017): 51–66, https://doi.org/10.1007/s11071-016-3024-6.

85. K. Zhan, D. Wei, J. Shi, and J. Yu, "Cross-Utilizing Hyperchaotic and DNA Sequences for Image Encryption," *Journal of Electronic Imaging* 26, no. 1 (2017): 013021, https://doi.org/10.1117/1.jei.26.1.013021.

86. X. Wang and C. Liu, "A Novel and Effective Image Encryption Algorithm Based on Chaos and DNA Encoding," *Multimedia Tools and Applications* 76 (2017): 6229–6245, https://doi.org/10.1007/s11042-016-3311-8.

87. D. Ustun and S. Sahinkaya, "Design of Secure S-Boxes via Novel 2D-Zettle Chaotic Map and ABC Algorithm for Robust Image Encryption," *Mathematics and Computers in Simulation* 235 (2025): 175–204, https://doi.org/10.1016/j.matcom.2025.03.019.

88. K. Pandey and D. Sharma, "Novel Image Encryption Algorithm Utilizing Hybrid Chaotic Maps and Elliptic Curve Cryptography With Genetic Algorithm," *Journal of Information Security and Applications* 89 (2025): 103995, https://doi.org/10.1016/j.jisa.2025.103995.

89. G. Alvarez and S. Li, "Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems," *International Journal of Bifurcation and Chaos* 16, no. 8 (2006): 2129–2151, https://doi.org/10.1142/s0218127406015970.

90. B. M. El-den, S. Aldosary, H. Khaled, T. M. Hassan, and W. Raslan, "Leveraging Finite-Precision Errors in Chaotic Systems for Enhanced Image Encryption," *IEEE Access* 12 (2024): 176057–176069, https://doi.org/10.1109/ACCESS.2024.3462807.

91. W. Feng, J. Zhang, Y. Chen, et al., "Exploiting Robust Quadratic Polynomial Hyperchaotic Map and Pixel Fusion Strategy for Efficient Image Encryption," *Expert Systems With Applications* 246 (2024): 123190, https://doi.org/10.1016/j.eswa.2024.123190.

92. S. Etemadi Borujeni and M. Eshghi, "Chaotic Image Encryption System Using Phase-Magnitude Transformation and Pixel Substitution," *Telecommunication Systems* 52 (2013): 525–537, https://doi.org/10.1007/s11235-011-9458-8.

93. Y. Wang, K. W. Wong, X. Liao, and G. Chen, "A New Chaos-Based Fast Image Encryption Algorithm," *Applied Soft Computing* 11, no. 1 (2011): 514–522, https://doi.org/10.1016/j.asoc.2009.12.011.

94. A. Shafique and J. Shahid, "Novel Image Encryption Cryptosystem Based on Binary Bit Planes Extraction and Multiple Chaotic Maps," *The European Physical Journal Plus* 133, no. 8 (2018): 331, https://doi.org/10.1140/epjp/i2018-12138-3.

95. G. Zhang and Q. Liu, "A Novel Image Encryption Method Based on Total Shuffling Scheme," *Optics Communications* 284, no. 12 (2011): 2775–2780, https://doi.org/10.1016/j.optcom.2011.02.039.

96. C. Zhu, G. Wang, and K. Sun, "Improved Cryptanalysis and Enhancements of an Image Encryption Scheme Using Combined 1D Chaotic Maps," *Entropy* 20, no. 11 (2018): 843, https://doi.org/10.3390/e20110843.

97. C. Lakshmi, K. Thenmozhi, J. B. B. Rayappan, and R. Amirtharajan, "Hopfield Attractor-Trusted Neural Network: An Attack-Resistant Image Encryption," *Neural Computing and Applications* 32, no. 15 (2020): 11477–11489, https://doi.org/10.1007/s00521-019-04637-4.

98. Y. Wang, L. Teng, and X. Wang, "An Image Encryption Algorithm Based on Circular Rotation and Generalized Feistel Structure," *Soft Computing* 28, no. 5 (2024): 4335–4358, https://doi.org/10.1007/s00500-023-08747-z.

99. L. Liu, D. Jiang, T. An, and Y. Guan, "A Plaintext-Related Dynamical Image Encryption Algorithm Based on Permutation-Combination-Diffusion Architecture," *IEEE Access* 8 (2020): 62785–62799, https://doi.org/10.1109/ACCESS.2020.2983716.

100. A. Khan, I. Alam, M. F. Khan, et al., "A Comprehensive Analysis of Adaptive Image Restoration Techniques in the Presence of Different Noise Models," *Foundation University Journal of Engineering and Applied Sciences* 1, no. 2 (2020): 6–13, https://doi.org/10.33897/fujeas.v1i2.322.