drones

Article

A New Image Encryption Algorithm Based on DNA State Machine for UAV Data Encryption

Moatsum Alawida [1,*], Je Sen Teh [2] and Wafa' Hamdan Alshoura [2]

1 Department of Computer Sciences, Abu Dhabi University, Abu Dhabi 59911, United Arab Emirates
2 School of Computer Sciences, Universiti Sains Malaysia, George Town 11800, Malaysia
* Correspondence: moatsum.alawida@adu.ac.ae

Abstract: Drone-based surveillance has become widespread due to its flexibility and ability to access hazardous areas, particularly in industrial complexes. As digital camera capabilities improve, more visual information can be stored in high-resolution images, resulting in larger image sizes. Therefore, algorithms for encrypting digital images sent from drones must be both secure and highly efficient. This paper presents a novel algorithm based on DNA computing and a finite state machine (FSM). DNA and FSM are combined to design a key schedule with high flexibility and statistical randomness. The image encryption algorithm is designed to achieve both confusion and diffusion properties simultaneously. The DNA bases themselves provide diffusion, while the random integers extracted from the DNA bases contribute to confusion. The proposed algorithm underwent a thorough set of statistical analyses to demonstrate its security. Experimental findings show that the proposed algorithm can resist many well-known attacks and encrypt large-sized images at a higher throughput compared to other algorithms. High experimental results for the proposed algorithm include correlation coefficients of 0.0001 and Shannon entropy of 7.999. Overall, the proposed image encryption algorithm meets the requirements for use in drone-based surveillance applications.

Keywords: drone security; drone-based surveillance; DNA; finite state machine; image encryption; key scheduling; unmanned aerial vehicle; UAV

check for updates

Citation: Alawida, M.; Teh, J.S.; Alshoura, W.H. A New Image Encryption Algorithm Based on DNA State Machine for UAV Data Encryption. *Drones* **2023**, *7*, 38. https://doi.org/10.3390/drones7010038

Academic Editor: Vishal Sharma

Received: 28 November 2022
Revised: 30 December 2022
Accepted: 3 January 2023
Published: 5 January 2023

1. Introduction

Drones, or unmanned aerial vehicles (UAVs), are increasingly being used for various applications, such as surveillance and security, particularly in the rapidly expanding drone sector of industries such as oil and gas, where drones provide aerial assessments and surveillance of hazardous activities. In these industries, high-quality images captured by drones in critical sites are sent to a ground control station (GCS) for decision-making purposes, making image encryption an important aspect of drone usage. However, due to limited computational capabilities in terms of energy and memory, encryption algorithms for drones must not only be secure, but also highly efficient in terms of throughput. Drones offer several advantages over ground-based surveillance, such as the ability to monitor large and dangerous areas, maneuver quietly and efficiently, and track accurately, making them better suited for certain industrial applications. Additionally, drones are small, cheap, and more efficient than manned aircraft and can be difficult to detect using radar systems. With the use of various sensing and surveillance equipment, securing these drone systems is a growing area of interest.

There have been many image encryption algorithms proposed in the past that are based on various areas such as chaos [1–4], quantum computing [5], conventional encryption [6], and DNA [7–11]. These algorithms are used to encrypt digital images captured by various devices, including independent digital cameras, smartphones, laptops, and drones. Drones, in particular, are used for surveillance and are tasked with capturing high-resolution photos, which often have larger sizes and contain important details. Therefore, it is important to design encryption algorithms that are able to handle large images while also maintaining both security and speed.

Recently, DNA computing has been used in the design of encryption algorithms, utilizing dynamic rules to generate encryption keys. Digital images can also be represented as chains of DNA bases [7,8]. In general, DNA-based encryption involves encoding pixel values as DNA bases before performing the encryption process based on DNA operations. Once the encryption process is complete, these DNA bases are converted back into 8-bit pixel values (gray level) [9,10]. Chaos and DNA have been combined in a number of algorithms [7,10–13], including a recent chaos-based encryption algorithm with equilibrium key streams and DNA encryption, which was designed to address weaknesses of image ciphers against statistical attacks [9]. Another example is a medical image encryption algorithm [11] that combines DNA with a one-dimensional chaotic map, using the logistic map to control DNA rules and encrypting the most significant bits of the bit-planes using rotation and permutation operations. The secret key for this algorithm is derived from the plainimage using a hash function. This algorithm demonstrated better performance compared to other ciphers, but each image only has one secret key of limited size.

An encryption algorithm based on an arithmetic sequence scrambling model, DNA coding sequence, and one-dimensional logistic map was recently introduced in a study [14]. This algorithm uses hashing of the plainimage to generate chaotic variables as a secret key, making the encryption process dependent on the plainimage. The chaotic sequence is then converted into DNA sequences and used to achieve diffusion and confusion through DNA coding and operations. Another image cipher based on a compound sine–piecewise linear chaotic map and varying DNA coding was introduced in [15]. This proposed algorithm has a straightforward design, utilizing numerous arithmetic operations in its improved chaotic map. While chaos-based image encryption algorithms are prevalent, chaotic maps can be computationally expensive due to floating point operations and may have various security issues [16].

The aim of this work is to propose a novel image encryption algorithm that is suitable for use in drone surveillance systems. To achieve this, a new key scheduling algorithm based on DNA and a finite state machine (FSM) is proposed, which is able to provide a high level of security and encrypt images of any size. The round keys generated using the DNA-FSM method are dependent on a secret key with a flexible key space and exhibit a high level of randomness and sensitivity to small changes. Images of any size or type are processed as DNA bases to resist various attacks and eliminate correlations between pixels, and the cipher only requires two rounds of substitution and permutation operations to be performed in one pass. As far as the authors are aware, there has been no prior work focusing on drone-based image encryption, so comparisons were made against state-of-the-art image encryption algorithms designed for regular-sized images. The proposed algorithm meets the requirements of drone surveillance systems in industrial areas, which require high speed and security. The main contributions of this work can be summarized as follows:

- A new method, called DNA-FSM, is proposed for deriving round keys from the secret key to encrypt images of different sizes using the same key.
- A novel image encryption algorithm is proposed for use in secure surveillance drone systems to address industrial challenges. The algorithm is designed to achieve both diffusion and confusion characteristics in one pass, using key values to perform substitution and permutation operations on the plainimage pixels.

The structure of this paper is as follows: In Section 2, we discuss the proposed secure surveillance drone framework. In Section 4, we cover the preliminaries. In Section 4, we introduce the DNA-FSM method and analyze its performance. In Section 5, we present the DNA-FSM-based image encryption algorithm. In Section 6, we discuss the experimental results and performance. Finally, in Section 7, we conclude the paper with some final remarks.

2. Secure Surveillance Drone Framework

Drones have been widely used for surveillance by broadcasting a live video of the area over which they hover [17]. They are useful in industrial zones because they can access high-risk areas that need observation. They can traverse dangerous areas, that would otherwise be unsafe for personnel, to obtain information. Moreover, drones provide more flexibility, have a wider range of vision, and have the ability to track moving objects. However, there are also constraints to the drone's ability to broadcast, such as limited power and flight time. Thus, drone-based surveillance is used for very specific objectives. The drone operator needs to focus on specific areas and take high-resolution images with dimensions of up to $3 \times 2000 \times 1700$ pixels. As drone images contain sensitive information, they need to be encrypted before being sent over wireless communication technologies such as radio waves, ADS-B, satellite, cellular, Wi-Fi, and others.

Communication between the GCS and drone is mostly insecure and needs to be encrypted. To address this problem, a framework is required for encrypting drone images being communicated over insecure channels. A drone has limited power and captures images with high resolution. Thus, the encryption algorithm must be both power- and time-efficient to ensure that the drone can operate optimally. Figure 1 illustrates a secure surveillance drone framework for an industrial zone. Industrial zones are full of challenges, difficulties, and risks, so a drone helps to monitor dangerous situations. The framework consists of the drone, GCS, and communication link. An attacker has the capability to eavesdrop on an insecure communication link and perform a man-in-the-middle attack. Thus, an encryption algorithm is required to encrypt data being transmitted by the drone. This paper proposes an image encryption algorithm based on DNA and FSM to be used for this purpose.

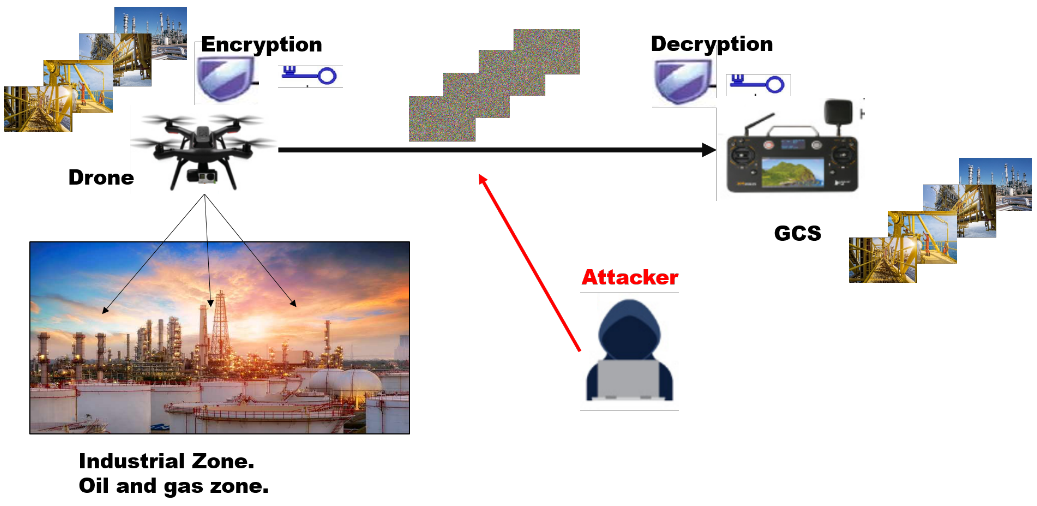


Figure 1. Secure surveillance drone framework.

3. Preliminaries

In this section, we provide some preliminary information about DNA and FSM, as both of these concepts play a major role in the proposed encryption algorithm.

3.1. DNA

Nucleic acids are biomolecules (or tiny biopolymers) that are required for all known living forms to exist. The term DNA is used to refer to both DNA and RNA, which are made up of nucleotides. DNA or RNA has four bases. DNA has adenine (A), cytosine (C), guanine (G), and thymine (T). Regarding RNA, the first three bases are similar, while the last is replaced by uracil (U). Each base has its complement; for example, A is the complement to T whereas C is the complement to G. A nucleotide sequence consists of biomolecules and symbolism, which can be expressed by a symbolic thread.

DNA concepts are widely utilized in computing fields such as data encryption. This is due to the ease of converting binary numbers into nucleotides to be processed. The four nucleotides can be represented using two bits each 00, 01, 10, 11. There are eight different ways to map these pairs of bits to nucleotides (referred to henceforth as encoding rules), as shown in Table 1. Various DNA operations can be performed on these four nucleotides. In DNA-based encryption methods, the XOR operation is often employed. Table 2 depicts the XOR result of all DNA nucleotide possibilities. DNA-XOR is one of the simplest and fastest logical operations and is used in encryption because it facilitates the recovery of the original plaintext after decryption. One of the main advantages of the DNA-XOR operation is that it can produce a uniform distribution between 0 and 1. Due to this property, DNA-XOR is employed in this paper to develop both the key schedule and encryption algorithm.

Table 1. DNA encoding rules.

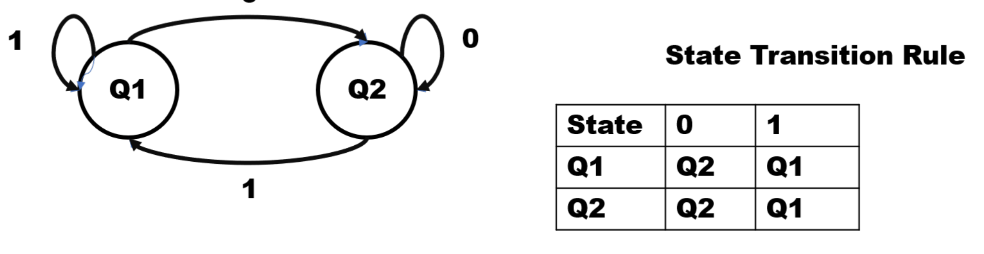
Binary	1	2	3	4	5	6	7	8
00	A	A	C	G	C	G	T	T
01	C	G	A	A	T	T	C	G
10	G	C	T	T	A	A	G	C
11	T	T	G	C	G	C	A	A

Table 2. DNA-XOR rules.

XOR	A	T	G	C
A	A	T	G	C
T	T	A	C	G
G	G	C	A	T
C	C	G	T	A

3.2. Finite State Machines

There are two types of FSMs—deterministic and nondeterministic. They are commonly used in computational linguistics [18]. FSMs have been used to design cryptographic algorithms to achieve higher security and encryption speed [13,19]. An FSM consists of a set of machine states or nodes and a set of links that connects them. These states may either be the initial, final, or connected states. The directed link between states is also called the arc or transition. Each transition has a label to distinguish it from other transitions associated with the states in the FSM. The relationship between states is governed by state transition rules. For a deterministic FSM, each transition is linked to each state. There is also no unlabeled transition between any two states. Figure 2 depicts an example of an FSM with two states and four transitions, each with its own labels.



State Transition Rule

State	0	1
Q1	Q2	Q1
Q2	Q2	Q1

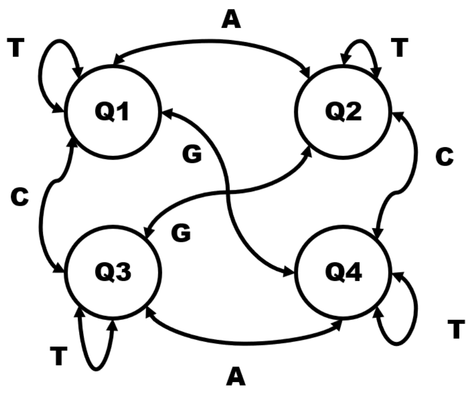
Figure 2. FSM with two states and its state transition rule.

An FSM can have a large number of states and transitions. The transition labels can also take on various symbols or values depending on the problem being solved. Therefore, the use of FSMs provides a high level of flexibility in terms of design and possible symbols for transitions. Each state can contain a mathematical or logical operation that will be performed on an initial condition, while transitions dictate the transition between states. For the proposed image cipher, we will use four states with distinct labels while each state has four transitions (three to transition to other states and one as a self-loop).

4. DNA-FSM

We propose DNA-FSM as a new method to design a key schedule capable of generating multiple round keys from a given secret key. Most existing methods rely on binary or integer-based operations to generate round keys. A key schedule is commonly found in conventional symmetric-key block ciphers. For example, the Tiny Encryption Algorithm (TEA) divides its 128-bit secret key into four 32-bit blocks which are used in consecutive rounds. The Data Encryption Standard (DES) key schedule divides its 56-bit key into two halves, which then undergo permutation operations to produce round subkeys.

In DNA-FSM, each state performs two logical operations, both of which are dependent on the labels of the four nucleotides A, C, T, and G. We utilize four nucleotides in a deterministic FSM. To minimize the complexity of the FSM, each state has four transitions that link to other states and itself. Each transition isbidirectional. Figure 3 shows the DNA-FSM method, which depicts four states that each have four incident transitions. The state transition rule is simplified to minimize latency when used in cryptographic algorithms.



State Transition Rule

State	Q1	Q2	Q3	Q4
Q1	T	A	C	G
Q2	A	T	G	C
Q3	C	G	T	A
Q4	G	C	A	T

Figure 3. FSM of four states and its state transition rule.

The logical operations used in the proposed DNA-FSM are DNA-XOR and left rotation, an operation commonly used in cryptographic algorithms due to its low computational overhead. DNA-FSM is controlled using a DNA sequence derived from the secret key. Transitions between the states are dictated by the secret key. When performing the DNA-XOR operation, each DNA base will be involved in a DNA-XOR operation with every single one of the secret key's DNA bases. The current or active state of the DNA-FSM is chosen based on the current DNA base. For each type of DNA base, the number of left rotations is fixed. For A, C, G, and T, the number of left rotations are 1, 2, 3, and 4, respectively. This operation contributes to the significant difference between round keys, thereby eliminating any correlation between subsequent keys.

The DNA-FSM method iterates multiple times, and each iteration involves computing both a DNA-XOR and left rotation operation. An updated DNA sequence is obtained and the next machine state is set. For the next iteration, the active DNA base is used to determine which DNA bases are used for the XOR operation. The state transitions are dictated by the secret key, resulting in the updated DNA sequence. In each iteration, we obtain one round key. This round key is of the same size as the original secret key. One of the main characteristics of the DNA-FSM key schedule is that it can process a secret key of any length to produce any number of round keys.

4.1. DNA-FSM Performance

The proposed DNA-FSM method can generate many round keys from the secret key. In this section, we evaluate the performance of DNA-FSM as a key scheduler based on bit sensitivity, histogram analysis, fuzzy entropy, the NIST statistical test suite, and the correlation coefficient (CC) [20,21]. These experiments and metrics were selected to showcase different aspects such as key sensitivity, distribution, complexity, randomness, and the relationship (or lack thereof) between round key values [22].

4.1.1. Key Sensitivity

Key sensitivity is measured by observing the impact of changing a single bit of the secret key to the output of the key schedule. In this experiment, one secret key bit is randomly selected and toggled (flipped from 0 to 1 or vice versa) to produce two keys that differ in only one bit. Each key is then used as an input to the DNA-FSM and their corresponding round keys are generated. The difference between the two round keys is calculated by determining if there is a difference between their DNA bases. If the DNA bases are the same, the value is zero, and if it is different, the value is one. The average of all differences between DNA bases for a total of 64 round keys is calculated. Values that are close to 1 indicate high sensitivity to each of the secret key bits. We repeat the experiment 64 times using various pairs of secret keys, the results of which are illustrated in Figure 4. We can see that the proposed method is highly sensitive because all values are close to 1 even after one DNA-FSM iteration. Thus, further increasing the number of iterations can lead to higher sensitivity.

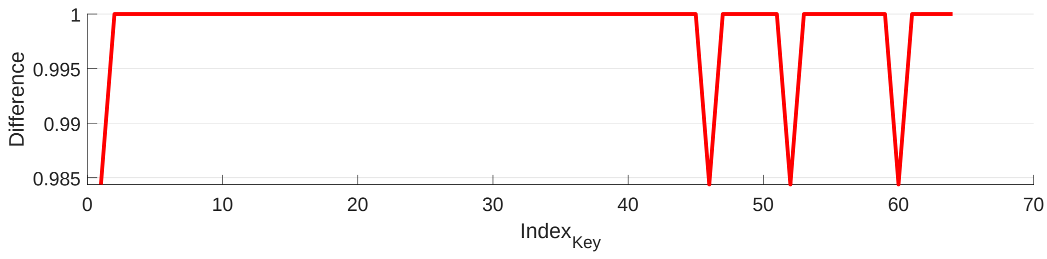


Figure 4. Key sensitivity of DNA-FSM.

4.1.2. Histogram Analysis

A histogram was used to study the distribution of the DNA bases for multiple round keys. The uniform distribution of the round keys reflects the lack of statistical biases and implies that an attacker would not be able to extract information about the original key value. Any bias towards one or two DNA bases may leak information. In this experiment, a histogram was produced for round keys by keeping count of each of the DNA bases. The ideal value for each DNA base is the size of the round key divided by four. In our experiment, the size of the round key is 128 bits, so the ideal value that represents a uniform distribution is 32. Figure 5 shows the histogram that depicts the average count for five iterations of the proposed DNA-FSM for a secret key DNA sequence that consists of *AATTCCGG* repeated eight times. Each of the four DNA bases has an average count of close to 32, which implies that the key bits are uniformly distributed.

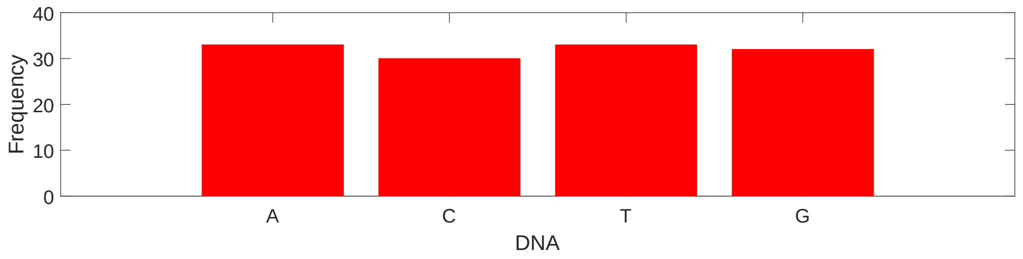


Figure 5. Histogram of 128 DNA bases after five iterations.

4.1.3. Fuzzy Entropy

Fuzzy entropy (FuzzyEN) can be used as a measure of system complexity [23], and uncertainty. A large value indicates high complexity and implies that an adversary will have difficulties in reconstructing the iterative patterns of the system. In our experiment, we generated 50 round keys with a length of 2000 DNA bases each (4000 bits) to identify if patterns exist in a long data sequence. The round keys were generated from secret keys with minor differences. In FuzzyEN, a Gaussian membership function is used as an alternative to the Heaviside function. The Gaussian function used to estimate FuzzyEN is defined as

$$\Theta(w_{i,j}^{m}, r) = exp(\frac{-(w_{i,j}^{m})^2}{r}), \tag{1}$$

where m is the embedding dimension, r is the tolerance value, and $w = \max_{i,j\,\in\,(0,m-1)} |x(i) - x(j)|$ is the maximum distance between two series equivalent to m. FuzzyEn was calculated and the results are illustrated in Figure 6. The results suggest that DNA-FSM is highly complex, thus patterns in its output sequences are difficult to detect even after just one iteration.

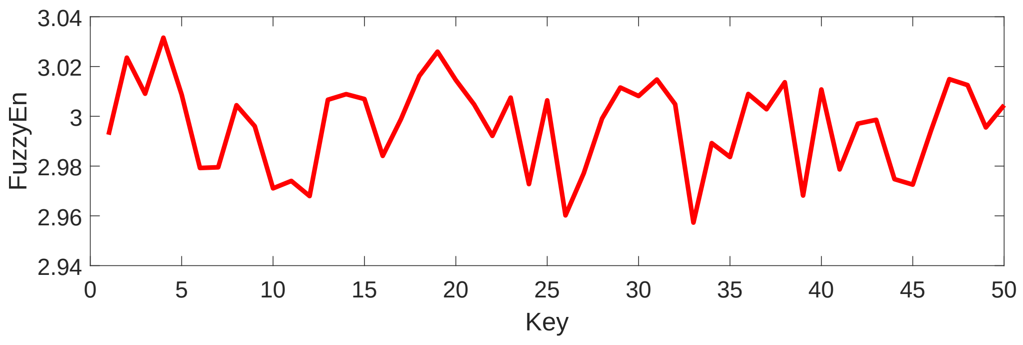


Figure 6. Fuzzy entropy analysis of DNA-FSM.

4.1.4. NIST SP 800-22

One important property of encryption keys in general is statistical randomness testing. To evaluate the statistical randomness of the round keys generated by DNA-FSM, we utilize the NIST SP 800-22 test suite. To generate sufficient bits for testing, we use 64 DNA bases as the secret key, then iterate the DNA-FSM method 125,000 times to generate one round key with a length of 10^6. In each iteration, we extract four DNA bases.

The round keys are converted into the binary and used as inputs to the NIST SP 800-22 test suite which has 15 subtests. A random sequence is considered to have passed the entire test suite if it successfully passes all subtests. To evaluate whether or not a sequence has passed a subtest, its P_{value} value must be greater than the significance value α, which is set as 0.01. Based on results shown in Table 3, the proposed method successfully passed all 15 subtests, implying that the round keys are sufficiently random, evenly distributed, and complex.

4.2. Correlation Coefficient

To study the relationship between round keys generated in each iteration with the original secret key, we can calculate their correlation coefficient. Correlation values that are near zero imply a lack of a statistical relationship between the secret key and its round key. In other words, the round keys are independent keys. We generate 50 round keys by iterating DNA-FSM 50 times; the original key is ACTG and repeated 32 times. CC is calculated after converting the DNA bases into bits as

$$C_r = \frac{COV(X,Y)}{\sqrt{D(X)D(Y)}}, \tag{2}$$

where $COV(x, y) = E([x - E(x)][y - E(y)])$, $E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i$ and $D(x) = \frac{1}{N}\sum_{i=1}^{N}[x_i - E(x)]^2$. x and y are the values of adjacent values in the sequence. The CC results in Figure 7 show that the round keys have no correlation to their original secret key, with values that are close to zero.

Table 3. NIST SP 800-22 results.

Subtests	PRNG
Frequency	0.7542
Block frequency	0.6523
Cumulative sums	0.8564
Runs test	0.3265
Longest run	0.1542
Binary matrix rank	0.6589
FFT	0.5421
Nonoverlapping template	0.3265
Overlapping template	0.6953
Overlapping template	0.2658
Universal	0.4526
Approximate entropy	0.3574
Random excursions	0.1594
Random excursions variant	0.6532
Serial	0.3254
Linear complexity	0.9658
Success Counts	15/15

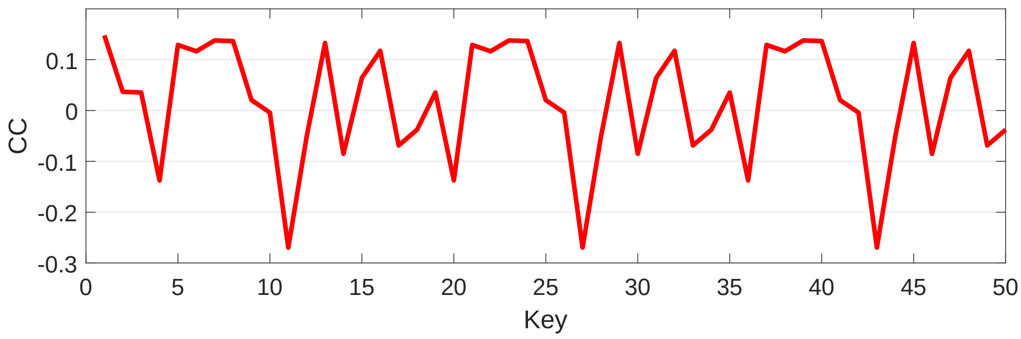


Figure 7. Correlation values of 50 round keys.

5. DNA-FSM-Based Image Encryption Algorithm

This section introduces the proposed image encryption algorithm based on DNA-FSM, which is used to generate round keys for each column or row of the plainimage. Figure 8 shows the three general steps of the proposed algorithm: round key generation, image conversion, and image encryption rounds. DNA-FSM is executed once to generate n round keys (represented as DNA bases and integers), where n is the total number of pixels in an image. Image conversion is where the image pixels are converted into DNA bases based on one of the DNA rules discussed in Section 3.1. There are a total of two rounds involved in image encryption. The encryption algorithm performs one of two scenarios depending on the dimension of the plainimage. If the plainimage has more columns than rows, then Round 1 performs column-wise encryption while Round 2 performs row-wise encryption. At the same time, the size of the round key will be based on the number of columns. If there are more rows than columns, these round operations are reversed, while the size of the round key is based on the number of rows. This ensures that the proposed algorithm is flexible enough to encrypt images of various dimensions. The final encrypted image is converted back to 8-bit pixels to display the cipherimage. Decryption involves performing the encryption steps in reverse, starting from round two to round one, then converting the result from DNA bases back to integers. The secret key (encryption key) must be exchanged in a secure manner between the communicating parties. Without the secret key, decryption will be unsuccessful. All steps of the proposed algorithm are detailed in the following subsections.

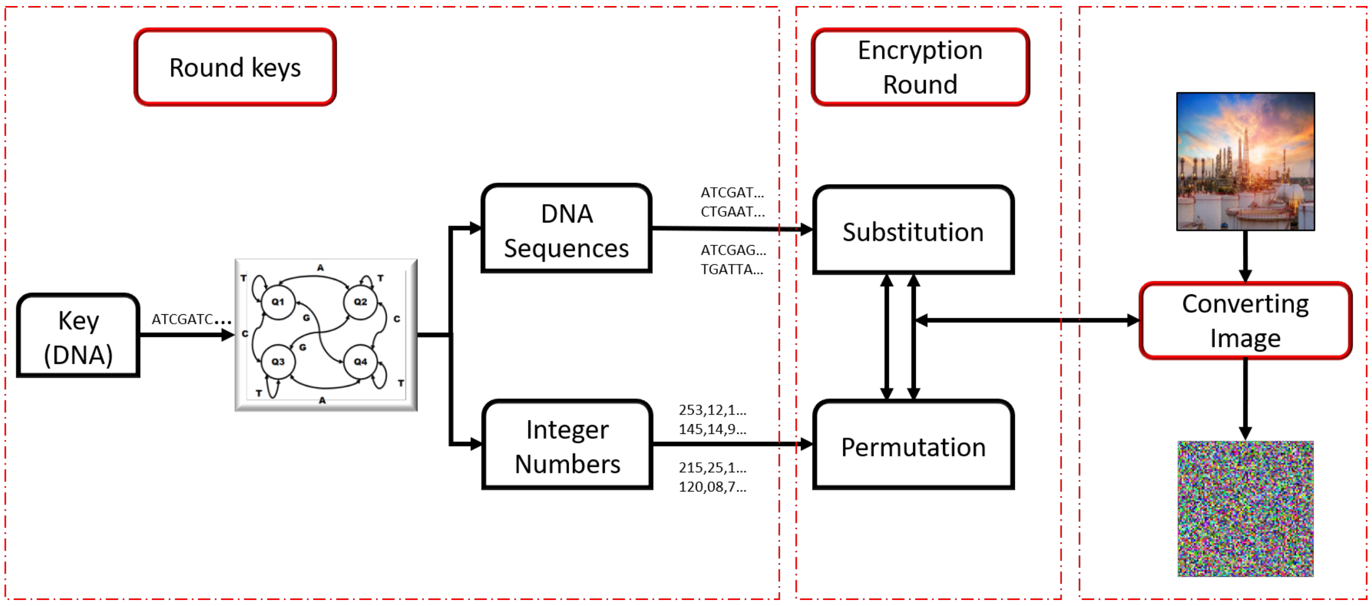


Figure 8. Overview of encryption steps.

DNA-FSM is different from conventional image encryption algorithms that are purely based on chaos or DNA computing. It is used to generate DNA bases and integers without having to depend on a nonlinear function such as a chaotic map. Instead, the secret key is used to produce highly random DNA and integer strings that are later used in substitution and permutation operations. DNA-FSM is well-suited for securing drone images as compared to other image encryption algorithms for the following reasons:

- DNA-FSM is structurally simple, unlike the high-dimensional chaotic maps with complicated mathematical operations that are commonly used for image encryption.
- DNA-FSM uses DNA-XOR and left rotation operations to generate round keys and the final key. Both operations are simple and have minimal computational overhead.
- DNA bases play a significant role in generating the final key, unlike other DNA image algorithms that use DNA bases only for the XOR operation.
- The proposed image encryption algorithm uses DNA bases of the image pixels and round keys to achieve diffusion while, at the same time, changing the positions of the DNA bases of the image pixels.
- Drone images have very large sizes and may contain sensitive information, so the drones need to secure the images and send them to the GCS for decryption. Therefore, a heavy focus will be on encryption rather than decryption because drones have limited computational resources. The encryption algorithm should have minimal latency, while the decryption algorithm will be on the GSC side which has ample computational resources.
- DNA operations and the FSM structure are both simple and have lower computing complexity as compared to other nonlinear functions.

5.1. Round Key Generation

In the image encryption algorithm, the secret key plays the biggest role in creating image-sized values. The keyspace should be higher than or equal to the recommended keyspace of 2^{128} to resist brute force attacks. In this paper, the smallest recommended key size is 128 bits, which is equivalent to 64 DNA bases. Users can choose to use larger key sizes based on the number of rows or columns of an image (whichever is larger). As discussed in Section 4, the secret key is assigned to each state, and with each iteration, we produce a new round key. The resulting round key is completely different from the original secret key, which is later showcased in Section 4.1. From each round key, 16 DNA base values are extracted and used to construct four columns (each column has four DNA bases) if column-wise encryption is performed. If row-wise encryption is to be performed, these DNA bases are used to construct four rows. In the rest of the following descriptions, we use column-wise encryption as an example.

A round key is one row of the final key of the algorithm. We select four bases of DNA for a column. These values are also converted integers. In the end, we have eight columns, four of which store DNA bases and the other four store integers. Regarding how the DNA bases are sampled from the round key, the first 4 DNA bases are allocated to the first, followed by the 5th to 8th DNA bases for the second, 50th to 53rd for the third, and 60th to 63rd for the fourth column. We selected these positions from the beginning to the end of the round key. The secret key is used to generate all round keys.

The round keys are random and uniformly distributed. They consist of two parts: DNA bases (four columns) and integer values (four columns). DNA bases are used to perform pixel substitution, whereas the integers are used for pixel permutation. These two operations play an important role in achieving confusion and diffusion properties, both of which are mandatory for encryption algorithms to ensure optimal security. Substitution involves modifying pixel values, while permutation changes or swaps pixel positions. Figure 9 illustrates the process involved in generating the round key.

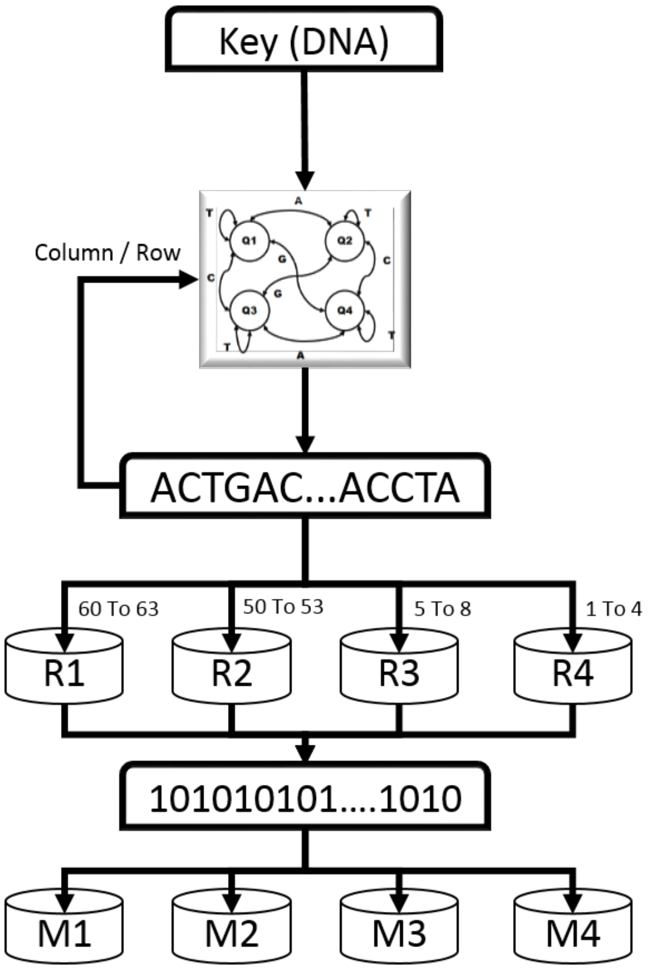


Figure 9. Round key generation.

5.2. Image Conversion

Image conversion needs to be performed before and after encryption/decryption. First, the pixel values are converted from 8-bit values to DNA bases before being encrypted or decrypted by the algorithm. After the encryption or decryption operations are completed, the result is converted from the DNA bases back into 8-bit pixels. Every 8-bit pixel is represented by four DNA bases. The number of columns after conversion is 4$\times$ the width of the image. We use the eighth DNA rule from Table 1 to convert pairs of bits into one DNA base. Every single pixel in the image will undergo the conversion process.

5.3. Image Encryption

The proposed DNA-FSM-based image encryption algorithm consists of two operations, substitution and permutation. The substitution operation modifies the DNA bases based on two other bases obtained from the round key and Val_{Pre}, which is a variable calculated from prior DNA bases. This is performed using the DNA-XOR operation. The final result is a new DNA base. Val_{Pre} is generated by using DNA-FSM with four DNA bases taken from the last four DNA bases that were encrypted. In the first round, the prior DNA bases are selected based on their indexes in ascending order, while in the second round, the prior DNA bases are selected based on their indexes in descending order. These four DNA bases are used to iterate DNA-FSM four times, resulting in four new DNA bases. This way, we ensure that all prior DNA bases will have a direct impact on the current DNA base being computed. A single change to any one of the DNA bases of the plainimage will diffuse throughout the remaining pixels of the plainimage in the first round. In the second round of encryption, these changes will again be diffused throughout the final encrypted image.

For the permutation operation, we use the round key integers, which are stored in the M1 to M4 columns or rows. First, each round key is assigned its own index (position). We then sort the round keys in ascending order and take note of how the index values are shuffled. The shuffled indexes will be used to dictate the permutation pattern. For both the first and second rounds, the first index is used for each column, while the second index is used for each row. Note that if there are more rows than columns, then this is reversed (first index for rows, second index for columns). For example, for an image P, if the first value in the sorted list of indexes is 15, the first value that will be encrypted is $P(15, i)$, and so on. Then, the permutation is then switched to either a column-wise or row-wise permutation, depending on which operation was used in the first round, e.g., if a column-wise permutation was performed in the first round, a row-wise permutation is performed in the second. The decision to start with column- or row-wise operations depends on the dimensions of the image (if there are more columns, column-wise operations are performed first, and vice versa). The remaining values of the round key are not involved in the second round.

Some important points to complete the algorithm are as follows:

- The first Val_{Pre} value for encrypting the first pixel in the plainimage is taken from the first four DNA bases of KEY.
- Converting the plainimage to DNA bases is performed once before encryption, and the resulting cipherimage is converted back from DNA bases into 8-bit pixels.
- If there are more columns than rows, encryption is performed in a column-wise manner in the first round, followed by a row-wise manner in the second round. If there are more rows than columns, this process is reversed.
- The final four DNA bases in the first round are used as prior values at the beginning of the second round to achieve full diffusion.

Algorithm 1 summarizes the important steps of the proposed algorithm. The decryption process performs all encryption steps in reverse (from the last pixel to the first and from the last DNA base in the round key to the first). As long as the recipient has a valid secret key, the decryption can be performed successfully.

Figure 10 shows the capability of the proposed algorithm in generating noise-like cipherimages. An effective encryption process is designed such that both confusion and diffusion are achieved in a single pass, each round. A single bit of change to the secret key will result in differing round keys whose effect will diffuse throughout the encrypted image. When dealing with color images, the three channels (red, green, and blue) are encrypted one by one. Then, the proposed algorithm is performed and the cipherimage is obtained by combining all three channels. The resulting colored image pixels are scrambled separately in each channel, effectively eliminating any statistical traces of the original plainimage.

Algorithm 1: Image encryption.

```
Data: Image, $P$ and Secret key, $KEY$
Result: Encrypted Image (C)
1  $[COL, ROW] = size(P, 1, 2);$
2  $Q = 1;$
3  for i=1 to Large(COL,ROW) do
4      $[KEY, Q] = DNA - FSM(KEY, Q);$
5      $R(1, i, :) = KEY(1:4), R(2, i, :) = KEY(5:8);$
6      $R(3, i, :) = KEY(50:53), R(4, i, :) = KEY(60:63);$
7      $M(1:4, i) = R(1:4, i, :);$
8  $[A1, B1] = sort(M(:, 1:COL));$
9  $[A1, B2] = sort(M(:, 1:ROW));$
10 $Val_{Pre} = KEY(1:4);$
11 for Channel=1 to size(P,3) do
12     for i=1 to COL do
13         $DNA_I = Image_{DNA}(P(:, i));$
14         for J=1 to ROW do
15             $F(1:4) = DNA_{Num}(DNA - FSM(Val_{Pre}, 1));$
16             $SCR(j, i, 1:4) = DNA_{XOR}(DNA_{XOR}(DNA_I(B1(j, 1), 1:4), R(1:4, i, F(1:4))), Val_{Pre}(1:4));$
17             $Val_{Pre}(1:4) = SCR(j, i, 1:4);$
18             $Val_{Pre} = DNA - FSA(Val_{Pre}, 1:4);$
19     for i=1 to ROW do
20         for J=1 to COL do
21             $F(1:4) = DNA_{Num}(DNA - FSM(Val_{Pre}, 1));$
22             $SCR1(j, i, 1:4) = DNA_{XOR}(DNA_{XOR}(SCR(B2(j, 1), 1:4), R(1:4, i, F(1:4))), Val_{Pre}(1:4));$
23             $Val_{Pre}(1:4) = SCR(j, i, 1:4);$
24             $Val_{Pre} = DNA - FSA(Val_{Pre}, 1:4);$
25     $C(:, :, Channel) = DNA_{Num}(SCR1, M);$
```

5.4. Discussion

A new image encryption algorithm based on the DNA methodology was proposed. DNA-FSM was developed to produce highly sensitive round keys derived from the secret key. These round keys play a major role in ensuring the security and randomness of the proposed algorithm. Encryption requires two rounds to achieve full diffusion of the key bits and plainimage pixels. The entire encryption process is based on DNA rules. A summary of the algorithm's important properties are as follows:

- The proposed algorithm has the flexibility to support any key length.
- Changes to just one bit of the plainimage will diffuse throughout the cipherimage, producing cipherimages that are completely different.
- Changes to just one bit of the secret key result in an entirely different cipherimage.
- Permutation and substitution are carried out in one pass. Thus, they both closely influence one another.

- The algorithm can support various image types and dimensions.
- The round keys generated from the secret key are used in the substitution process to provide a high level of confusion to the cipherimage, whereas the use of prior pixel values (in terms of DNA bases) improves the overall diffusion property of the algorithm.
- The proposed encryption can resist various attacks and statistical analyses such as the statistical, differential, and chosen/known-plaintext attacks. These are depicted in Section 6.

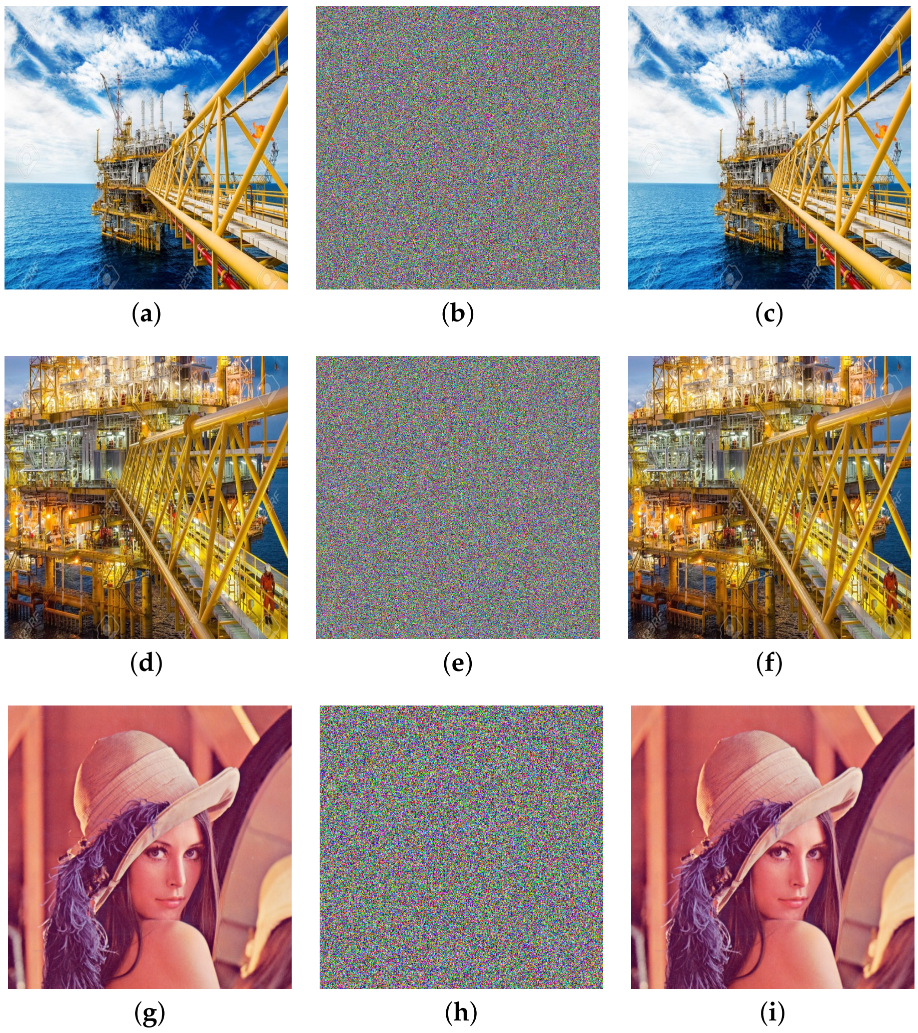


Figure 10. Four processes of the proposed image encryption scheme: (**a**) Oil rig (1) image size 7998 × 5332 × 3, (**b**) encrypted image, (**c**) decrypted image. (**d**) Oil rig (2) image size 6431 × 4272 × 3, (**e**) encrypted image, (**f**) decrypted image. (**g**) Lena image size 512 × 512 × 3, (**h**) encrypted Lena image, (**i**) decrypted Lena image.

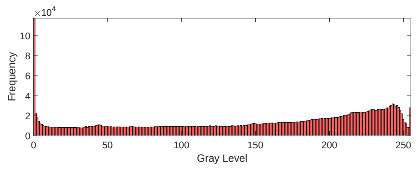
6. Experimental Results and Performance

In this section, all experiments were performed on Matlab R2021a to ease the processing of matrices and signals. The specification of the computing device includes an Intel(R) Core(TM) i5-10210U CPU@1.60 GHz 2.11 GHz and 16 GB RAM. We also study the efficiency of the proposed algorithm when implemented on a DJI Mavic Air 2 Drone, as shown in Figure 11. It is capable of 34 min of maximum flight time, records 4K/60 fps videos, and captures 48-megapixel photos. The images used in the experiments are divided into two types: standard images (such as Lena) that are commonly used to test state-of-the-art image encryption algorithms, and the second type consists of surveillance images taken from a drone in an industrial zone. The second category of images has high-resolution and large file sizes.

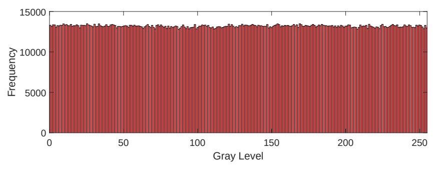
Figure 11. Drone DJI Mavic Air 2.

6.1. Histogram Analysis

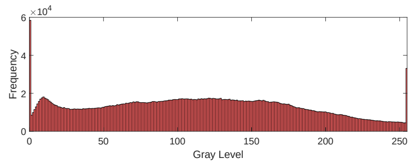
The histogram gives a visual description of the pixel distribution in an image. An image is essentially a two-dimensional matrix that consists of values (pixels) ranging between 0 and 255. The histogram illustrates the frequency of occurrences of each of the possible 256 pixel values. Ideally, cipherimages should have an evenly distributed histogram, which implies that each pixel occurs with equal probability or frequency. This implies that there are no observable patterns in the resulting cipherimage. Three plainimages were encrypted and their resulting histograms were generated before and after encryption, as shown in Figure 12. All three cipherimages have *flat* histograms, which indicates a lack of any identifiable biases or statistical patterns.



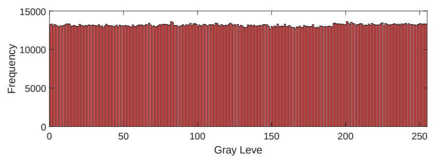
(**a**) Oil rig (1) (Plainimage)



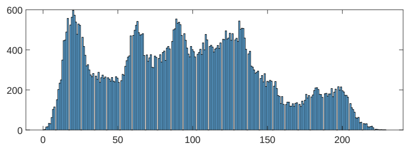
(**b**) Oil rig (1) (Cipherimage)



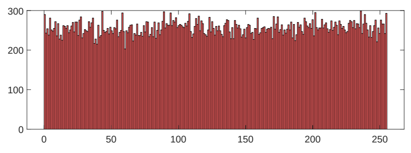
(**c**) Oil rig (2) (Plainimage)



(**d**) Oil rig (2) (Cipherimage)



(**e**) Lena (Plainimage)



(**f**) Lena (Cipherimage)

Figure 12. Histogram of different images.

6.2. Correlation Coefficients

CC is an important measure for studying the relationship between adjacent pixels in the image. Pixel values in the plainimage are always strongly correlated, unlike in cipherimages. The CC is calculated in three directions: vertical, horizontal, and diagonal. The proposed encryption algorithm eliminates any correlation between two adjacent pixels in only two rounds of encryption.

CC was calculated for all the test images in all three directions, as shown in Figure 13. Both the plainimage pixels and key values are randomly diffused throughout the cipherimages, leading to a lack of correlation between adjacent values. In contrast, the pixels in the plainimages are clearly correlated to one another, as depicted in Figure 13 and Table 4. Table 5 provides a numerical comparison of the average CC results of the proposed cipher against previously proposed algorithms. Note that the CC of the proposed algorithm is closer to zero, as compared to its peers. This confirms that the proposed algorithm can effectively eliminate any correlation between pixel values in the cipherimages.

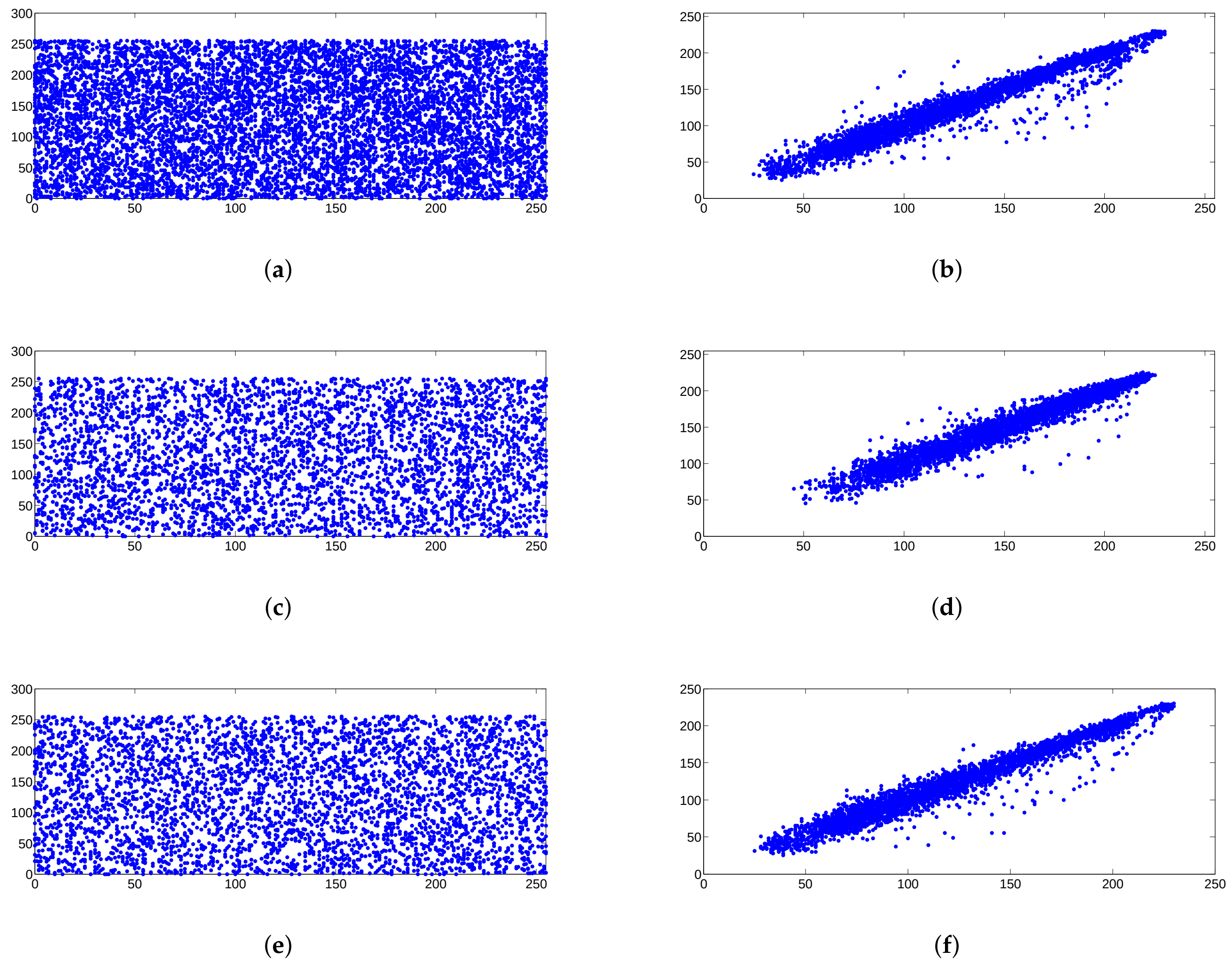


Figure 13. CC of Oil rig (1) image for red channel. (**a**) Horizontal cipherimage, (**b**) horizontal plainimage, (**c**) vertical cipherimage, (**d**) vertical plainimage, (**e**) diagonal plainimage, (**f**) diagonal cipherimage.

Table 4. Correlation coefficient for drone and Lena images.

Image	**Plainimage**			**Cipherimage**		
	D	**V**	**H**	**D**	**V**	**H**
Oil rig (1)	0.9475	0.8326	0.9542	−0.00031	−0.00542	−0.0026
Oil rig (2)	0.9124	0.8698	0.8547	−0.00075	−0.00214	−0.0042
Lena	0.8754	0.7542	0.9325	−0.00012	−0.00021	0.00123

Table 5. Correlation coefficient comparison of the encrypted Lena image.

Image	**Cipherimage**		
	D	**V**	**H**
Proposed algorithm	−0.00012	−0.00021	0.00123
Ref. [24]	0.0011	0.0012	0.016
Ref. [25]	0.0004	0.0011	0.0125
Ref. [26]	0.0015	0.0013	0.012

6.3. Chosen/Known Plaintext Attacks

In cryptanalysis, attackers try to find the relationship between the plain and the cipherimages to extract some recurring patterns that help recover the secret key or equivalent keys [27–29]. Completely black or white images are commonly used to identify these weaknesses. If the algorithm can still produce a statistically random cipherimage from a white or black plainimage, it can resist some of these attacks. Otherwise, cipherimages with observable patterns can be exploited by adversaries. Encryption processes must rely on secret key values as well as pixel values in the plainimage, but in the case of a black or white image, pixel values no longer have any effect on the encryption process, notably in the first round of encryption. However, the proposed cipher still produces a noise-like cipherimage after the second round of encryption. This analysis is illustrated in Figure 14, where one can see that the cipherimages are noise-like images that depict resistance against chosen or known plaintext attacks.

6.4. Contrast Analysis

To study the existence of persistent patterns or repetitions in different directions in an image, we perform contrast analysis. The differences between the local features of all pixels are measured across the four directions in the image. This can be computed mathematically as

$$C = \sum_{i,j} |i - j|^2 G(i, j), \tag{3}$$

where G is co-occurrence matrix of the gray level index, i, and j are 8-bit gray level pixels. According to [30], the ideal value of the contrast features can be calculated as

$$C_{theoretical} = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{|i - j|^2}{(256)^2}. \tag{4}$$

The $C_{theoretical}$ value is 10,922.5. Table 6 lists the results of our contrast analysis of the sample images. The differences between our results and the ideal values are calculated. The results confirm that the proposed algorithm does not have fixed patterns of local features between pixels in different directions.

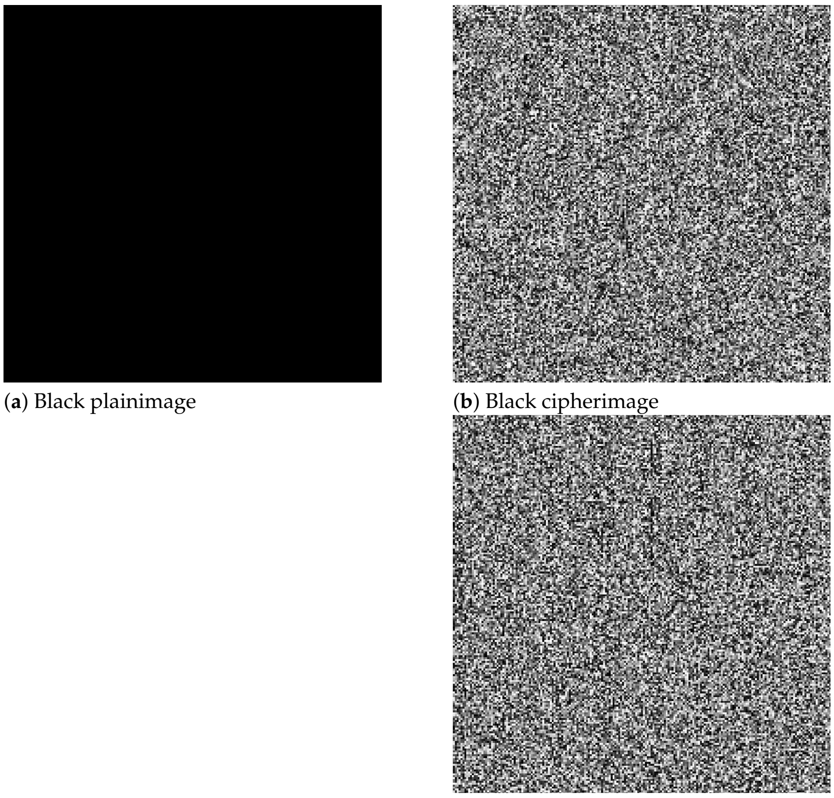

(**a**) Black plainimage (**b**) Black cipherimage (**c**) White plainimage (**d**) White cipherimage

Figure 14. Encryption results of black and white images.

Table 6. Contrast values.

Image	Contrast Value	Ideal Value	Difference
Oil rig (1)	10,865.6	10,922.5	56.868
Oil rig (2)	10,902.0	10,922.5	20.4644
Lena	10,893.4	10,922.5	29.1331

6.5. Local Shannon Entropy

The Shannon entropy is a measure of the uniform distribution of values in a series. In image encryption, it measures the uniform distribution of pixel values within the image. Pixel values that are equally distributed are an indicator that an image is highly random. This even distribution must occur for both the entire encrypted image and also subsets of the image to be selected in a random manner. To analyze Shannon entropy for smaller subsets of the overall image, we can calculate the local Shannon entropy (LSE) [31]. For an entire image I, Shannon entropy can be calculated as

$$H(I) = -\sum_{j=0}^{L-1} p(I_j) log_2 p(I_j), \tag{5}$$

where $p(I_j)$ is the discrete probability density function (PDF), I_j is the pixel value from 0 to 255, and L is gray level ($L = 256$ for an 8-bit gray image). For LSE calculation, nonoverlapping blocks are selected from the cipherimage with an equal number of pixels. Shannon entropy is calculated per block, then the average Shannon entropy value is calculated for all blocks. The LSE can be calculated as

$$H_{k,T_B}(I) = \sum_{i=1}^{k} \frac{H(I_{B_i})}{k}, \tag{6}$$

where $H(I_{B_i})$ is the Shannon entropy of nonoverlapping image blocks B_i. k and T_B denote the number of blocks and pixels in each block, respectively. The LSE test has critical limits, such that if the LSE values of the cipherimage are within this critical interval, it implies that the algorithm is successful in producing highly random images. This interval is within (7.901901305, 7.903037329) and was calculated according to a significance value $\alpha = 0.05$ and parameters $(k, TB) = (30, 1936)$ [31].

The resulting LSE for the three cipherimages produced by the proposed algorithm are as follows: Oil rig (1) 7.902736, Oil rig (2) 7.902865, and Lena 7.902924, all of which fall within the aforementioned critical interval. Therefore, the LSE results confirm that the proposed algorithm produces a high level of randomness even in small localized subsets of the image.

6.6. Differential Attack Analysis

To be even remotely capable of resisting a differential attack, cipherimages produced by a cipher must have high sensitivity to slight changes to the plainimage's pixel values. The use of the DNA-XOR operation ensures that any change in the DNA bases will result in an entirely different output. In addition, the encryption process relies on past encrypted values that will be diffused throughout the image. If changes to the plainimage are not completely diffused, the differences between cipherimages can be used to facilitate key recovery attacks. To analyze the resistance of image ciphers against differential attacks, the number of pixel change rate (NPCR) and the unified average change intensity (UACI) is commonly used. Two plainimages with a 1-bit difference and the same secret key are used. They are encrypted to obtain two cipherimages, C_1 and C_2. NPCR and UACI can be estimated as

$$NPCR(C_1, C_2) = \frac{\sum_{i,j}^{M,N} D(i,j)}{MN} \times 100\% \tag{7}$$

and

$$UACI(C_1, C_2) = \frac{1}{MN} \sum_{i,j}^{M,N} \frac{|C_1(i,j) - C_2(i,j)|}{L} \times 100\% \tag{8}$$

respectively, where MN is the total number of pixels in a plainimage, $L = 255$ is the maximum gray level value for an 8-bit pixel, and

$$D(i,j) = \begin{cases} 1, & \text{if } C_1(i,j) \neq C_2(i,j) \\ 0, & \text{if } C_1(i,j) = C_2(i,j), \end{cases} \tag{9}$$

Two tests have critical values to measure if the cipherimage has successfully passed the evaluation. These critical values are used as thresholds [32]. NPCR has a critical value (N^*_α) for different image sizes. N^*_α is defined as

$$N^*_\alpha = \frac{L - \Phi^{-1}(\alpha)\sqrt{L/MN}}{L+1}, \tag{10}$$

where α is the significance level, and Φ^{-1} is the inverse cumulative density function (CDF) of normal distribution $N(0,1)$. An image encryption scheme must have an NPCR value larger than N^*_α to pass the NPCR test. In the UACI test, two strict critical intervals $U^{*-}_\alpha, U^{*+}_\alpha$ can be calculated for different image sizes as

$$\begin{cases} U^{*-}_\alpha = \mu_u - \Phi^{-1}(\alpha/2) \times \sigma_u; \\ U^{*+}_\alpha = \mu_u + \Phi^{-1}(\alpha/2) \times \sigma_u, \end{cases} \tag{11}$$

where

$$\mu_u = \frac{L+2}{3L+3} \tag{12}$$

and

$$\sigma_u^2 = \frac{(L+2)(L^2+2L+3)}{18(L+1)^2 \times L \times MN}. \tag{13}$$

First, we set the importance level as $alpha$ =0.05 for each image used in the test. We select a random 256 × 256 subset of pixels from each cipherimage to compare with critical intervals. Thus, the critical values for the selected size are $N_\alpha^* \geq 99.5693\%$ and $U_\alpha^{*-}, U_\alpha^{*+} = (33.2824\%, 33.6447\%)$. NPCR and UACI are calculated for all three cipherimages, the results of which are shown in Table 7. For all cipherimages, the proposed algorithm successfully passed the evaluation criteria for three images with different sizes and dimensions.

Table 7. NPCR and UACI values.

Image	NPCR	UACI
Oil rig (1)	99.782	33.548
Oil rig (2)	99.634	33.446
Lena	99.635	33.458

6.7. Speed Analysis

Image encryption algorithms must be able to encrypt different image sizes with a high security margin. To achieve this goal, many image ciphers include complex mathematical operations that lead to an increase in latency. Thus, there is always a contradiction between algorithm speed and security. In this algorithm, the DNA rules were used to avoid dealing with bits directly. Moreover, round keys were used at one time, whereby the number of round keys is equal to the total number of pixels columns or rows (depending on which one is larger). In addition, the DNA rules process 2 bits at one time, which should cut latency in half. Thus, we minimize the computational complexity of the proposed cipher while maintaining its high security level. Table 8 shows the time required to encrypt and decrypt the various test images. These images are large and would require a significant amount of time to encrypt; however, the proposed algorithm encrypts can still encrypt them under 17 s, even for the largest image, which contains a total of approximately $2^{26.9}$ pixels.

Table 8. Speed analysis of encryption and decryption (second).

Image	Encryption	Decryption
Oil rig (1) (7998 × 5332 × 3)	16.124	16.895
Oil rig (2) (46,431 × 4272 × 3)	11.241	12.124
Lena (512 × 512 × 3)	1.141	1.926
Ref. [33] Lena (512 × 512 × 1)	0.9731	-
Ref. [34] Lena (512 × 512 × 1)	0.5510	-
Ref. [35] Lena (512 × 512 × 1)	1.3357	-

6.8. Comparison

In this section, a comparison between the proposed cipher and other algorithms is provided. This comparison is to benchmark the performance and efficiency of the proposed algorithm against the current state-of-the-art. We selected two image ciphers from previous studies that are highly cited recently. The selected algorithms were implemented on the same computer and on the same images to ensure that a fair comparison is performed. We chose a set of 28 widely used images in addition to the three images that were used before. All previous tests presented in the paper were implemented for algorithms involved in the comparison. For 28 images, we calculated the average for all metrics that include CC, contrast, and speed. Success values (those that fall within the required intervals) are counted for LSE, NPCR, and UACI. Table 9 summarizes all of these results which shows that the proposed algorithm can outperform its peers in both security and efficiency.

Table 9. Comparison of the proposed cipher against two other ciphers on the same computer.

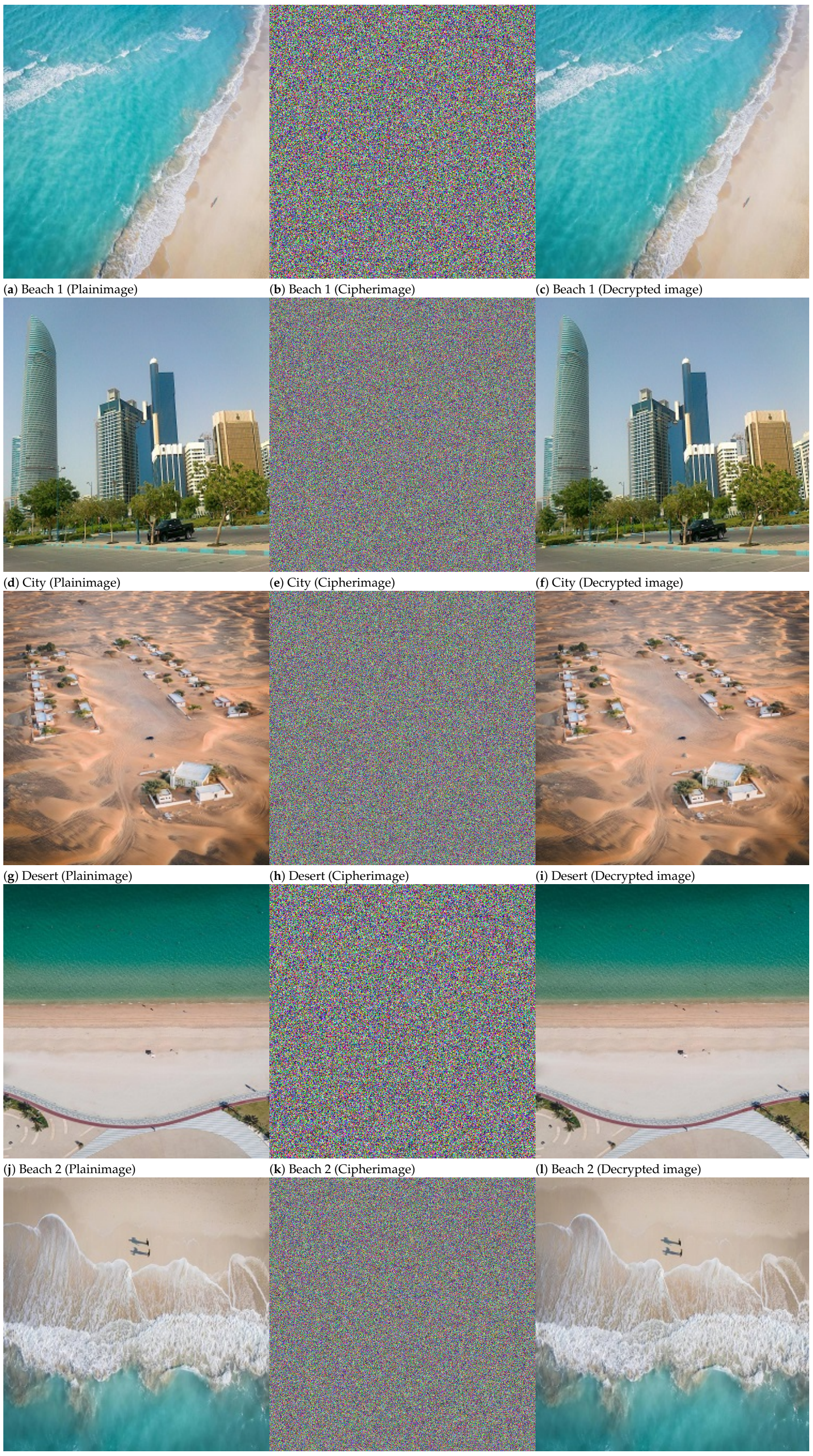
Algorithm	Image	Test					
		CC	LSE	Contrast	NPCR	UACI	Speed
Proposed Algorithm	Oil rig (1)	−0.00031	7.902736	10,865.6	99.782	33.548	16.124
	Oil rig (2)	−0.00075	7.902865	10,902.0	99.634	33.446	11.241
	Lena	−0.00012	7.902924	10,893.4	99.635	33.458	1.141
	28 images	0.00136	25/28	10,933.06	27/28	27/28	1.834
Ref. [24]	Oil rig (1)	0.00541	7.912542	10,903.4	99.714	33.452	4050
	Oil rig (2)	0.00143	7.924513	10,936.4	99.854	33.654	2600
	Lena	0.00042	7.901456	10,928.6	99.651	33.465	25.30
	28 images	0.00214	14/28	10,923.09	27/28	20/28	29.76
Ref. [26]	Oil rig (1)	0.00512	7.912543	10,909.1	99.424	31.652	1296
	Oil rig (2)	0.00152	7.91245	10,925.5	99.412	33.985	832
	Lena	0.00425	7.90324	10,918.2	99.36	28.41	8.61
	28 images	0.01245	16/28	10,923.9	4/28	4/28	22.20

We selected a set of recently proposed algorithms for comparison purposes. These algorithms represent the current state-of-the-art in image encryption and serve as a benchmark for our proposed solution. Table 10 summarizes the comparison of the proposed algorithm with 11 other image ciphers [36–46]. We selected several key metrics that allow to compare the algorithms from various perspectives. Firstly, Shannon entropy (SE) looks at the randomness and uniformity of pixel distribution in a cipher image. A uniform pixel range will not allow an attacker to trivially identify patterns that can be used in cryptanalysis attempts. The proposed algorithm has an SE score that closely approximates the ideal value of eight. This result was achieved because the two encryption rounds perform both diffusion and confusion operations simultaneously. In terms of other metrics, such as CC, NPCR, and UACI, results show that the proposed algorithm has stronger security (based on statistical properties) as compared to other algorithms. It also has a large key space. The proposed cipher is suitable for drone applications because it does not need significant resources for encryption. We use two simple structures, DNA and FSM, to generate highly random DNA and integer strings. In contrast, other classical chaotic image ciphers require a nontrivial number of mathematical operations to generate chaotic sequences to encrypt images.

Table 10. Comparison of the proposed cipher against other image ciphers.

Algorithms	SE	CC	NPCR	UACI	Key Space
DNA-FSM (Lena)	7.9999	0.0001	99.63	33.45	2^M(M can be any number)
Ref. [36]	7.9021	−0.0017	99.60	33.45	2^{400}
Ref. [37]	7.9977	−0.0007	99.60	33.45	2^{1096}
Ref. [38]	7.9970	0.0022	99.61	33.46	0.3×2^{158}
Ref. [39]	7.9971	0.0113	99.61	33.66	2^{624}
Ref. [40]	7.9978	−0.0012	99.60	33.45	2^{688}
Ref. [41]	7.9972	−0.0019	99.61	33.43	2^{320}
Ref. [42]	7.9973	0.0026	99.58	33.54	2^{236}
Ref. [43]	7.9999	0.0004	99.61	33.46	2^{512}
Ref. [44]	7.9991	−0.0634	99.34	33.51	2^{396}
Ref. [45]	7.9972	−0.0023	99.60	33.47	2^{299}
Ref. [46]	7.9993	0.02694	99.64	33.47	6.13×10^{501}

Figure 15 shows the number of plainimages and cipherimages captured by different drone models in different locations to study the proposed algorithm on the number of captured images by drones of various sizes. We confirmed that our proposal can encrypt and decrypt plainimages of any size with complete recovery. The results for these captured images are shown in Table 11. We used the same metrics that we used previously. It is important to note that our proposal can generate cipherimages with high statistical security.



(**a**) Beach 1 (Plainimage) (**b**) Beach 1 (Cipherimage) (**c**) Beach 1 (Decrypted image)
(**d**) City (Plainimage) (**e**) City (Cipherimage) (**f**) City (Decrypted image)
(**g**) Desert (Plainimage) (**h**) Desert (Cipherimage) (**i**) Desert (Decrypted image)
(**j**) Beach 2 (Plainimage) (**k**) Beach 2 (Cipherimage) (**l**) Beach 2 (Decrypted image)
(**m**) Beach 3 (Plainimage) (**n**) Beach 3 (Cipherimage) (**o**) Beach 3 (Decrypted image)

Figure 15. Drone images captured in various locations.

Table 11. Comparison of the proposed cipher with other image ciphers.

Images	SE	CC	NPCR	UACI
Beach 1	7.998	0.0021	99.65	33.43
City	7.9992	0.0015	99.67	33.46
Desert	7.9991	0.0017	99.63	33.47
Beach 2	7.9982	−0.0001	99.62	33.45
Beach 3	7.9981	0.0008	99.61	33.44

7. Conclusions

In this research, we introduced a new image encryption algorithm to meet the security and computational requirements of drone-based surveillance. A key scheduling algorithm was proposed based on DNA and FSM. The secret key was used to create a large number of round keys with flexible sizes. DNA bases were used to perform substitution while the integer representation of these DNA bases was used in the permutation operation. The cipher was designed to achieve both confusion and diffusion properties in one round. A total of two rounds is required to encrypt images that can be of any size and type, to produce noise-like images. The proposed key schedule and image encryption algorithms were subjected to a large set of statistical analyses. The experimental results show that the round keys are statistically random, and the proposed algorithm can encrypt large images within a relatively short amount of time. The proposed algorithm was also shown to be resistant to attacks. The proposed algorithm fulfilled the spatial requirements for drone applications; however, the proposed algorithm can still improve its encryption speed for large images. To address this drawback, future work can look into simplifying the DNA-FSM further to achieve higher efficiency.

Author Contributions: Conceptualization, methodology, formal analysis M.A.; writing—review and editing, writing—original draft preparation M.A.; supervision, project administration M.A; data curation, validation software J.S.T. and W.H.A.; funding acquisition M.A., and J.S.T.; investigation, resources M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was fully supported by Abu Dhabi University under Grant No. 19300635.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This article contains no data or material other than the articles used for the review and referenced..

Acknowledgments: The authors would like to express their gratitude to the Office of Research & Sponsored Programs at Abu Dhabi University in the United Arab Emirates for their support of this research article.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Laiphrakpam, D.S.; Thingbaijam, R.; Singh, K.M.; Al Awida, M. Encrypting Multiple Images With an Enhanced Chaotic Map. *IEEE Access* **2022**, *10*, 87844–87859. [CrossRef]
2. Xian, Y.; Wang, X. Fractal sorting matrix and its application on chaotic image encryption. *Inf. Sci.* **2021**, *547*, 1154–1169. [CrossRef]
3. Alawida, M.; Teh, J.S.; Mehmood, A.; Shoufan, A.; Alshoura, W.H. A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations. *J. King Saud-Univ.-Comput. Inf. Sci.* **2022**, *34*, 8136–8151. [CrossRef]
4. Alshoura, W.H.; Zainol, Z.; Teh, J.S.; Alawida, M.; Alabdulatif, A. Hybrid SVD-Based Image Watermarking Schemes: A Review. *IEEE Access* **2021**, *9*, 32931–32968. [CrossRef]
5. Ye, G.; Jiao, K.; Huang, X. Quantum logistic image encryption algorithm based on SHA-3 and RSA. *Nonlinear Dyn.* **2021**, *104*, 2807–2827. [CrossRef]
6. Zhang, Y. Test and Verification of AES Used for Image Encryption. *3D Res.* **2018**, *9*, 1–27. [CrossRef]
7. El-Khamy, S.E.; Mohamed, A.G. An efficient DNA-inspired image encryption algorithm based on hyper-chaotic maps and wavelet fusion. *Multimed. Tools Appl.* **2021**, *80*, 23319–23335. [CrossRef]

8. Hazra, A.; Ghosh, S.; Jash, S. A Review on DNA Based Cryptographic Techniques. *Int. J. Netw. Secur.* **2018**, *20*, 1093–1104.
9. Wu, J.; Liao, X.; Yang, B. Image encryption using 2D Hénon-Sine map and DNA approach. *Signal Process.* **2018**, *153*, 11–23. [CrossRef]
10. Zhou, S. A real-time one-time pad DNA-chaos image encryption algorithm based on multiple keys. *Opt. Laser Technol.* **2021**, *143*, 107359. [CrossRef]
11. Aouissaoui, I.; Bakir, T.; Sakly, A. Robustly correlated key-medical image for DNA-chaos based encryption. *IET Image Process.* **2021**, *15*, 2770–2786. [CrossRef]
12. Alawida, M.; Samsudin, A.; Alajarmeh, N.; Teh, J.S.; Ahmad, M.; Alshoura, W.H. A Novel Hash Function Based on a Chaotic Sponge and DNA Sequence. *IEEE Access* **2021**, *9*, 17882–17897. [CrossRef]
13. Alawida, M.; Teh, J.S.; Oyinloye, D.P.; Alshoura, W.H.; Ahmad, M.; Alkhawaldeh, R.S. A New Hash Function Based on Chaotic Maps and Deterministic Finite State Automata. *IEEE Access* **2020**, *8*, 113163–113174. [CrossRef]
14. Yan, X.; Wang, X.; Xian, Y. Chaotic image encryption algorithm based on arithmetic sequence scrambling model and DNA encoding operation. *Multimedia Tools Appl.* **2021**, *80*, 10949–10983. [CrossRef]
15. Zhang, S.; Liu, L. A novel image encryption algorithm based on SPWLCM and DNA coding. *Math. Comput. Simul.* **2021**, *190*, 723–744. [CrossRef]
16. Preishuber, M.; Hütter, T.; Katzenbeisser, S.; Uhl, A. Depreciating Motivation and Empirical Security Analysis of Chaos-Based Image and Video Encryption. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 2137–2150. [CrossRef]
17. Vacca, A.; Onishi, H. Drones: Military weapons, surveillance or mapping tools for environmental monitoring? The need for legal framework is required. *Transp. Res. Procedia* **2017**, *25*, 51–62. [CrossRef]
18. Ibias, A.; Núñez, M.; Hierons, R.M. Using mutual information to test from Finite State Machines: Test suite selection. *Inf. Softw. Technol.* **2021**, *132*, 106498. [CrossRef]
19. Alawida, M.; Samsudin, A.; Teh, J.S.; Alshoura, W. Deterministic chaotic finite-state automata. *Nonlinear Dyn.* **2019**, *98*, 2403–2421. [CrossRef]
20. Imdad, M.; Ramli, S.N.; Mahdin, H. An Enhanced Key Schedule Algorithm of PRESENT-128 Block Cipher for Random and Non-Random Secret Keys. *Symmetry* **2022**, *14*, 604. [CrossRef]
21. Zakaria, A.A.; Ab Halim, A.H.; Ridzuan, F.; Zakaria, N.H.; Daud, M. LAO-3D: A Symmetric Lightweight Block Cipher Based on 3D Permutation for Mobile Encryption Application. *Symmetry* **2022**, *14*, 2042. [CrossRef]
22. Hernandez-Castro, J.C.; Peris-Lopez, P.; Aumasson, J.P. On the key schedule strength of present. In *Data Privacy Management and Autonomous Spontaneus Security*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 253–263.
23. Xie, H.B.; Chen, W.T.; He, W.X.; Liu, H. Complexity analysis of the biomedical signal using fuzzy entropy measuremen. *Appl. Soft Comput.* **2011**, *11*, 2871–2879. [CrossRef]
24. Hua, Z.; Zhou, Y. Image encryption using 2D Logistic-adjusted-Sine map. *Inf. Sci.* **2016**, *339*, 237–253. [CrossRef]
25. Huang, L.; Cai, S.; Xiong, X.; Xiao, M. On symmetric color image encryption system with permutation-diffusion simultaneous operation. *Opt. Lasers Eng.* **2019**, *115*, 7–20. [CrossRef]
26. Diab, H. An Efficient Chaotic Image Cryptosystem Based on Simultaneous Permutation and Diffusion Operations. *IEEE Access* **2018**, *6*, 42227–42244. [CrossRef]
27. Li, M.; Lu, D.; Xiang, Y.; Zhang, Y.; Ren, H. Cryptanalysis and improvement in a chaotic image cipher using two-round permutation and diffusion. *Nonlinear Dyn.* **2019**, *96*, 31–47. [CrossRef]
28. Li, M.; Fan, H.; Xiang, Y.; Li, Y.; Zhang, Y. Cryptanalysis and Improvement of a Chaotic Image Encryption by First-Order Time-Delay System. *IEEE Multimed.* **2018**, *25*, 92–101. [CrossRef]
29. Li, M.; Lu, D.; Wen, W.; Ren, H.; Zhang, Y. Cryptanalyzing a Color Image Encryption Scheme Based on Hybrid Hyper-Chaotic System and Cellular Automata. *IEEE Access* **2018**, *6*, 47102–47111. [CrossRef]
30. Haralick, R.M.; Shanmugam, K.; Dinstein, I.H. Textural features for image classification. *IEEE Trans. Syst. Man Cybern.* **1973**, *6*, 610–621. [CrossRef]
31. Wu, Y.; Zhou, Y.; Saveriades, G.; Agaian, S.; Noonan, J.P.; Natarajan, P. Local Shannon entropy measure with statistical tests for image randomness. *Inf. Sci.* **2013**, *222*, 323–342. [CrossRef]
32. Wu, Y.; Noonan, J.P.; Agaian, S. Npcr and uaci randomness tests for image encryption. *J. Sel. Areas Telecommun.* **2011**, *4*, 31–38.
33. Diaconu, A.V. Circular inter–intra pixels bit-level permutation and chaos-based image encryption. *Inf. Sci.* **2016**, *355–356*, 314–327. .: 10.1016/j.ins.2015.10.027. [CrossRef]
34. Liao, X.; Lai, S.; Zhou, Q. A novel image encryption algorithm based on self-adaptive wave transmission. *Signal Process.* **2010**, *90*, 2714–2722. [CrossRef]
35. Ping, P.; Xu, F.; Mao, Y.; Wang, Z. Designing permutation–substitution image encryption networks with Henon map. *Neurocomputing* **2018**, *283*, 53–63. [CrossRef]
36. Tao, Y.; Cui, W.; Zhang, Z. Spatiotemporal chaos in multiple dynamically coupled map lattices and its application in a novel image encryption algorithm. *J. Inf. Secur. Appl.* **2020**, *55*, 102650. [CrossRef]
37. Zhou, S.; Wang, X.; Zhang, Y.; Ge, B.; Wang, M.; Gao, S. A novel image encryption cryptosystem based on true random numbers and chaotic systems. *Multimed. Syst.* **2022**, *28*, 95–112. [CrossRef]
38. Chen, C.; Sun, K.; He, S. An improved image encryption algorithm with finite computing precision. *Signal Process.* **2020**, *168*, 107340. [CrossRef]

39. Belazi, A.; Abd El-Latif, A.A.; Belghith, S. A novel image encryption scheme based on substitution-permutation network and chaos. *Signal Process.* **2016**, *128*, 155–170. [CrossRef]
40. Wang, M.; Wang, X.; Zhao, T.; Zhang, C.; Xia, Z.; Yao, N. Spatiotemporal chaos in improved cross coupled map lattice and its application in a bit-level image encryption scheme. *Inf. Sci.* **2021**, *544*, 1–24. [CrossRef]
41. Hosny, K.M.; Kamal, S.T.; Darwish, M.M.; Papakostas, G.A. New image encryption algorithm using hyperchaotic system and fibonacci q-matrix. *Electronics* **2021**, *10*, 1066. [CrossRef]
42. Wang, X.; Gao, S.; Ye, X.; Zhou, S.; Wang, M. A new image encryption algorithm with cantor diagonal scrambling based on the PUMCML system. *Int. J. Bifurc. Chaos* **2021**, *31*, 2150003.
43. Xian, Y.; Wang, X.; Teng, L.; Yan, X.; Li, Q.; Wang, X. Cryptographic system based on double parameters fractal sorting vector and new spatiotemporal chaotic system. *Inf. Sci.* **2022**, *596*, 304–320. [CrossRef]
44. Sayed, W.S.; Radwan, A.G.; Fahmy, H.A.; Elsedeek, A. Trajectory control and image encryption using affine transformation of lorenz system. *Egypt. Inform. J.* **2021**, *22*, 155–166. [CrossRef]
45. Xu, Q.; Sun, K.; Zhu, C. A visually secure asymmetric image encryption scheme based on RSA algorithm and hyperchaotic map. *Phys. Scr.* **2020**, *95*, 035223. [CrossRef]
46. Broumandnia, A. Designing digital image encryption using 2D and 3D reversible modular chaotic maps. *J. Inf. Secur. Appl.* **2019**, *47*, 188–198. [CrossRef]