Review Paper

Two Feistel rounds in image cryptography acting at the nucleotide level exploiting dna and rna property

Abdellatif JarJar[1]

Abstract
In recent years, a variety of chaos-based image encryption algorithm have been proposed. The majority of image encryption systems, employ the confusion-diffusion architecture and operate at the pixel level. In this paper, a new color image encryption algorithm at the nucleotide-level applied, will be proposed. The first block will be altered by confusion with a boot vector generated from the clear image. After that, two Feistel rounds will be imposed on each 12 nucleotides clear block. The first round is performed by chaotic displacement of the nucleotides, and, the second uses a confusion matrix created in preconceived designs from chaotic maps used. Ultimately, a diffusion will be implemented to maximize the avalanche impact and keep the system out of differential attacks. On output, a transition to the nucleotides complementary will be established, and a confusion with the chaos vector will be developed. Next, the RNA application can be used to synthesize amino acids, and as a result, the isolation of the exon genes from introns. Simulations made on a large database with images in a variety of sizes and formats, show that our strategy can be prevented from being attacked by any known attack.

Keywords $G_t = Z/tZ$ · Chaotic map · FEISTEL two-round diagram · DNA · RNA

Abbreviations

G_t^*	Set of invertible elements of the G_t-ring
DNA	Desoxi-ribonucleic acid
RNA	Ribonucleic acid
M(i:)	Line number i of matrix M
M(:I)	Column number i of matrix M

1 Introduction

Communications has always been an important aspect in the acquisition of new knowledge and the development of humanity. The need to be able to send a message securely is probably as old as the communications themselves.

From a historical point of view, it is during conflicts between nations that this need has been most acute lively. In our modern world, where various methods of communication are used regularly, the need for confidentiality is more present than ever in a multitude of levels. For example, it is normal for a firm to want to protect its new software against piracy, that banking institutions want to ensure that transactions are and that all individuals want their personal data protected. The need for secure communications has given rise to the science we call cryptology.

The encryption operation transforms plain text into encrypted text, called a cryptogram, using a key (called the encryption key). Decryption is the processing of data that transforms encrypted text into plain text. Cryptography is the science of creating such encryption systems. Cryptanalysis is the complementary science of determining certain properties of these systems in order to reconstruct the plain text, All accepted algorithms incline to Shanon's recommendation "confusing diffusion permutation" and also to Kerchof's principle "the algorithm must be known, the security of the algorithm is based on the confusion of the encryption key.

Cryptology includes cryptography and cryptanalysis, often in the absence of the necessary parameters for

✉ Abdellatif JarJar, abdoujjar@gmail.com | [1]Moulay Rachid High School, Taza, Morocco.

SN Applied Sciences (2019) 1:1411 | https://doi.org/10.1007/s42452-019-1305-7

Received: 30 June 2019 / Accepted: 18 September 2019 / Published online: 15 October 2019

decryption. Encryption is generally performed using a well-defined algorithm and a unique key.

With the development of innovative technologies in the field of information sciences and the digital world, all data is increasingly shared on the Internet or stored on magnetic media. On the other hand, unauthorized access to information or private information has become an issue in the virtual world. Security issues have raised many concerns not only among researchers, and people using the Internet. Encryption has become an effective way to avoid attacks. However, traditional encryption standards, such as DES and AES, are generally designed for textual information. They are considered unsuitable for information with a high correlation between data, such as image and video data.

The fundamental architecture of image encryption based on chaotic systems was first proposed by Fredrich in 1998. Within this structure, two unrelated steps are performed in a single encryption cycle. These are the phases of confusion and diffusion. First, a random permutation of all pixels leads to a large reduction in the correlation between adjacent pixels. However, confusion operations are carried out in a closed manner. Sometimes, a one-dimensional chaotic or random sequence is used to systematically modify the value of each pixel by an XOr operator and a chain with the already encrypted pixels. After this step, the basic elements of the image, the bit or pixel values, will be evenly distributed.

Since the image is introduced and processed in digital form, that its applications have continued to increase. It has become exploited by a wide public, both professionals and amateurs alike. But given the extent of the computer resources allowing the free circulation of information, the ease of transmission of confidential data, man has been pushed to improve more and more encryption algorithms to secure his confidential data. But all his techniques bow to Shanon's recommendations [1] For permutation, some approaches use Arnold's method [2], others use improvements to Hill's classic method [3], and still others use static permutation matrixes [4]. For confusion, most algorithms use the operator Xor noted between the plain text and the encryption key [5]. Few color image encryption methods use diffusion to avoid differential attacks. Today, faced with the great mathematical advance of chaos theory, we are experiencing a wave of algorithms based on the construction of recurring suites with a chaotic aspect [6] that is developing more and more at high speed. Chang'e Dong [7] offers color image encryption based on the construction of a coupled chaotic map. Wanga et al. [8] proposed a crypto system based on a multitude of chaotic maps that define an effective result. All these approaches use a Lyapunov exponent calculation [5] to check the installation of chaos and sensitivity to initial conditions. Most encryption algorithms operating on blocks used the Feistel scheme with several turns. RC4, RC6, DES used more than four towers [9]. In our approach, we will apply two Feistel towers on 24-bit.

Most encryption techniques and systems have been implemented on the modular arithmetic basis. However, these techniques have been extensively exploited and are under attack. This has encouraged researchers to invent other techniques based on the use of some DNA property. DNA cryptography is a new field of instinctive cryptography that has emerged from DNA computer research. Some algorithms used for DNA cryptography have limitations in that they still use modular arithmetic cryptography at some of their stages or are based on biological laboratory experiments, which is not appropriate in the digital computing environment. To fill this gap, we describe a new algorithm for color image encryption based on chaos, DNA transformation and the biological property of the RNA.

In 1994, Adleman [10] included the first scientist who conducted DNA experiments and as a result a new generation of DNA research was developed. Many researchers have retrieved the information side of DNA to use it in computer-based investigations. At the same time, DNA use in cryptography is beginning to surface and is being used as a medium in bioinformatics [11]. Gehani et al. [12] presented an image encryption algorithm for DNA strand cryptography.

Zhang et al. [13] have proposed a color image encryption using DNA characteristics and an enforcement of DNA sequence processing to encode the pixels in the image. They used chaotic and hyper-chootic maps to effectively carry out their approach [14].

Virtually all encryption schemes use chaotic systems combined with DNA performance and fixed value allocation for nucleotides [15] (A = 0; C = 1; G = 2; T = 3), this weakens the technique. However, in our approach, the nucleotide values will be selected pseudo randomly and will be related to the chaotic maps used. Others use the properties of genetic algorithms and DNA [16].

2 The proposed method

Based on chaos [17], our algorithm is an encryption system that takes as input a color image of size (n, m) and generates an encrypted image of the same size at the output. With this technology, a new encryption operating mode is being introduced based on the application of two Feistel rounds over each 12 nucleotides block. As a follow-up, a transition to nucleotides complementary and RNA property is applied for amino acid production. This initiative is structured into four main themes. The phases of this new methodology can be shown in the Fig. 1.

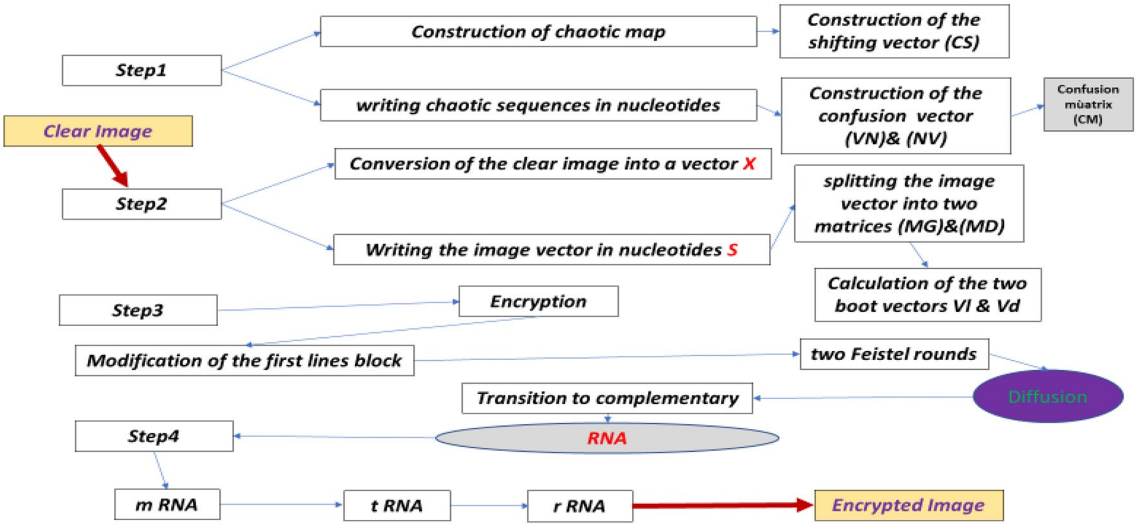

Construction of chaotic map
Construction of the shifting vector (CS)
Step1
writing chaotic sequences in nucleotides
Construction of the confusion vector (VN)& (NV)
Confusion mùatrix (CM)
Clear Image
Conversion of the clear image into a vector X
Step2
splitting the image vector into two matrices (MG)&(MD)
Writing the image vector in nucleotides S
Calculation of the two boot vectors Vl & Vd
Step3
Encryption
Modification of the first lines block
two Feistel rounds
Diffusion
Transition to complementary
Step4
RNA
m RNA
t RNA
r RNA
Encrypted Image

Fig. 1 Steps for the encryption of a color image

A. Step 1:

(1) Development of chaotic maps:

For the generation of encryption keys, our strategy uses the two most widely deployed chaotic maps used in color image encryption, the Logistics Map and the Skew Tent Maps. This operation is highlighted in Fig. 2.

(a) The logistics map

The use of the logistics map in image cryptography is due to the simplicity of its expression and the high sensitivity to initial condition. Its expression is described by Eq. 1

$$\begin{cases} \mathbf{u}_0 \in]0,51[, \mu \in [3,754] \\ \mathbf{u}_{\mathbf{n}+1} = \mu \mathbf{u}_{\mathbf{n}}(1-\mathbf{u}_{\mathbf{n}}) \end{cases} \tag{1}$$

It is known that the logistics map presents a chaotic aspect for $\mathbf{u}_0 \in]0,51[, \boldsymbol{\mu} \in [3,754]$. For the simulations of our approach, we have chosen, $u_0 = 0,7458121001$as the initial value and $\mu = 3.859412001$as a control parameter.

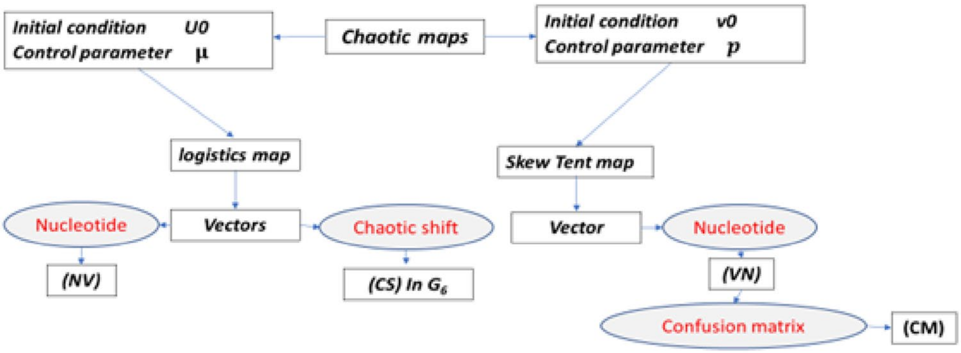
(b) The Skew Tent Map

The Skew tent map will be redefined as the Eq. (2)

$$\begin{cases} \mathbf{v}_0 \in]01[p \in]0,51[\\ \mathbf{v}_{\mathbf{n}+1} = \begin{cases} \frac{\mathbf{v}_\mathbf{n}}{\mathbf{p}} & if\ 0 \prec \mathbf{v}_\mathbf{n} \prec p \\ \frac{1-\mathbf{v}_\mathbf{n}}{1-\mathbf{p}} & if\ p \prec \mathbf{v}_{\mathbf{n}\prec 1} \end{cases} \end{cases} \tag{2}$$

It is known that The Skew Tent Map presents a chaotic aspect for $\boldsymbol{v}_0 \in]01[\boldsymbol{p} \in]0,51[$. For the simulations of our approach, we have chosen, $v_0 = 0.54865$ as the initial value, and $p = 0.8742120013$ as a control parameter. In the end, the key space of our algorithm is $u_0 = 0,7458121001, \mu = 3.859412001$, for logistic map, and $p = 0.8742120013, v_0 = 0,548652012$, for Skew tent map. In the simulation, we operate with a precision of 10-10,

Fig. 2 Building the encryptions keys



therefore, the global key size of the encryption key is significantly greater than 100 bits. This protects our algorithm from brutal attacks.

(c) Writing chaotic nucleotide sequences:

Many encryption schemes based on **DNA** property affect the initial nucleotide values saved in Table (***IV***):

(***IV***)	Initial values	0	1	2	3
	Nucleotide	A (adenine)	C (cytosine)	G (guanine)	T (thymine)
	Complementary	T	G	C	A

This allocation of values to nucleotides weaknesses the cryptographic system. To reinforce our systems against attacks, the values assigned to nucleotides will be taken in pseudo random ways and highly sensitive to the chaotic maps used.

For the rewriting of the sequences of the logistics map, we will apply a first permutation (***PR***) on the second line of the main nucleotide table (***IV***), obtained by a descending sorting in the broad sense of the chaotic vector (***PI***) whose coordinates are:

(***PI***)	*U*(*n*)	*U*(*m*)	*V*(*nm*)	*V*(2*nm*)

We obtain a new table (***NT***)) of the nucleotide values allocated to the logistic sequence specified by the algorithm 1:

Algorithm 1

```
for $i = 1$ to $4$
$NT(1;\boldsymbol{i}) = IV(1;i)$
$NT(2;\boldsymbol{i}) = IV(2;PR(\boldsymbol{i}))$
Next $i$
```

For the rewriting of the sequences of the SKTM, we will apply a permutation (***RP***) on the second line of the main nucleotide table (***NT***), obtained by a descending sorting in the broad sense of the chaotic vector (***IP***) whose coordinates are:

(***IP***)	*V*(*n*)	*V*(*m*)	*U*(*nm*)	*U*(2*nm*)

This transformation creates a third line of the table (***NT***) of nucleotide values for the **SKTM** sequence, given by the algorithm 2

Algorithm 2

```
for $i = 1$ to $4$
$NT(3;\boldsymbol{i}) = NT(2;RP(\boldsymbol{i}))$
Next $i$
```

Example:

(***PI***)	0.8858	0.87452	0.54215	0.92450
(***IP***)	0.31457	0.92450	0.8858	0.80745

The generated (***PR***) and (***RP***) permutation is

(***PR***)	4	1	2	3
(***PR***)	1	4	3	2

Finally

(***NT***)	Final value	0	1	2	3		
	Nucleotides	T	A	C	G	(***PR***)	For logistic map
	Complementary	T	G	C	A	(***RP***)	For SKTM

We confirm, that a change in the vector (***IP***) or vector (***PI***) would generate a new permutation and subsequently a new (***NT***) table of nucleotide values, and consequently, new nucleotide values will be allocated to the encryption keys.

(d) Conversion of chaotic values into nucleotides.

The transformation of the logistics sequence terms into nucleotides leads to the output of the size vector (***NV***) (1.12 nm), and the transition of the SKTM terms into nucleotides generates the size vector (***VN***) (1.12 nm). This routine is performed by the algorithm

Algorithm 3

```
for $i = 1$ to $12\,nm$
for $k = 4$ to $0$
if $u(\boldsymbol{i}) - \frac{\boldsymbol{k}}{4} \geq 0$ then
$NV(\boldsymbol{i}) = NT(2, \boldsymbol{k} + 1)$
$VN(\boldsymbol{i}) = NT(3, \boldsymbol{k} + 1)$
Exit for
end if
Next $k, i$
```

In parallel, the construction of the displacement vector (***CS***) used in the first Feistel round as an element of $\boldsymbol{G}_6$ is dictated by algorithm 4

Algorithm 4

```
for $i = 1$ to $nm$
$CS(\boldsymbol{i}) = mod\left(10^{12}\boldsymbol{u}(\boldsymbol{i}), 5\right) + 1$
Next $i$
$\forall\, i$ with $0 \leq i \leq nmCS(i) \neq 0$
```

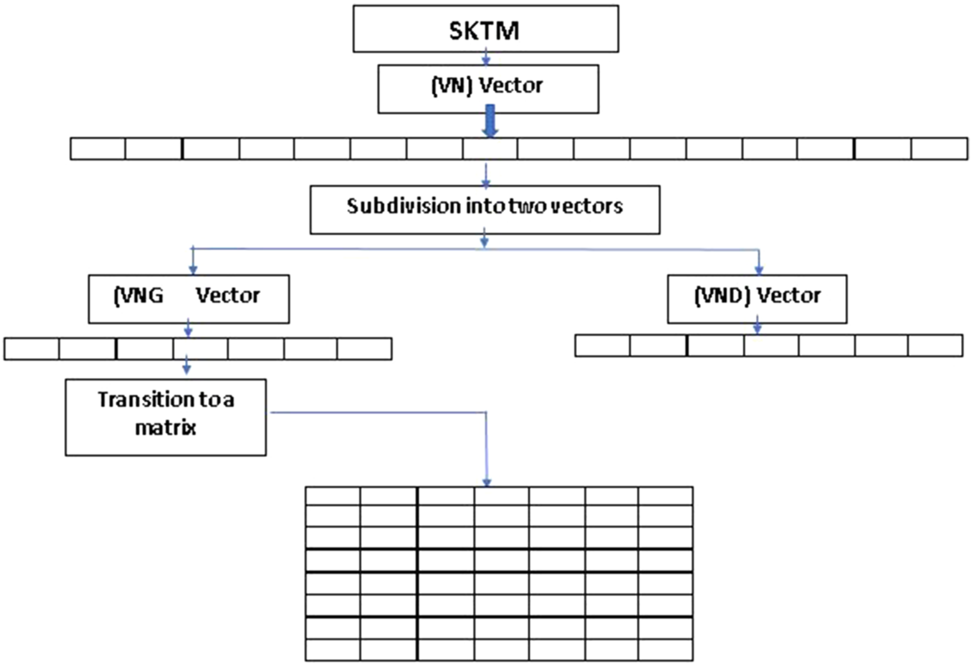


Fig. 3 Creation of the confusion matrix

Example:

Let's be the chaotic ($\boldsymbol{U}_i$) and ($\boldsymbol{V}_i$) sequence:

Chaotic ($\boldsymbol{U}_i$) sequence											
0.2925	0.68251	0.9658	0.0325	0.8952	0.4125	0.3215	0.5214	0.5602	0.7854	0.2153	0.6582
Chaotic ($\boldsymbol{V}_i$) sequence											
0.2925	0.68251	0.9658	0.0325	0.8952	0.4125	0.3215	0.5214	0.5602	0.7854	0.2153	0.6582

By applying algorithms 3, 4, we obtain the following values

Nucleotide writing

A	*C*	*G*	*T*	*G*	*A*	*A*	*C*	*C*	*C*	*G*	*T*
Nucleotide writing											
G	*C*	*A*	*T*	*A*	*G*	*G*	*C*	*C*	*C*	*A*	*T*

Therefore, the data of two identical vectors from two different chaotic maps, generates two different nucleotide vectors

(e) Construction of the (***CM***) confusing matrix

The first 6 nm vector measurements (***VN***) are transcribed into a matrix (***CM***) of size (nm, 6). This matrix is considered as a confession matrix in the second Feistel round. This initiative is exemplified in Fig. 3.

This matrix is used as a confusion template in the second round.

(f) Algebraic DNA operations

Our approach involves algebraic operation data on nucleotides. Submissive to the Kerchof principle; the three fundamental operations employed in this algorithm are provided in the Fig. 4

$$\textbf{\textit{We note that}}\begin{cases}\forall x, y, z in \{\boldsymbol{A}, \boldsymbol{G}, \boldsymbol{C}, \boldsymbol{T}\} \\ If z = x + y \; Then\, x = z - y \\ If z = x \otimes y \; Then\, x = z \otimes y \\ x \otimes x = A\end{cases}$$

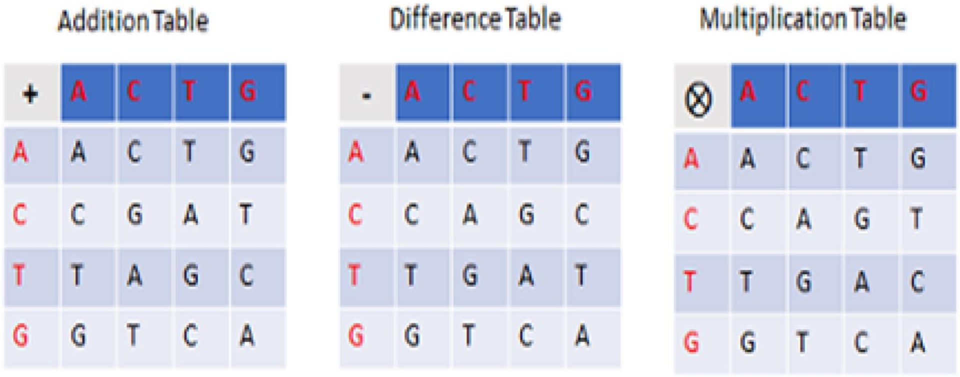
B. Step 2:

(a) Setting the image to be encrypted

Before starting the encryption process of the original color image, it is recommended to prepare the original color image. This phase is illustrated by the diagram in the Fig. 5.

(a) Translating the clear image to a vector

After extraction from the three (***RGB***) spectral channels, their vector conversion (***Vr***), (***Vg***), (***Vb***) we convert into the



Addition Table

+	A	C	T	G
A	A	C	T	G
C	C	G	A	T
T	T	A	G	C
G	G	T	C	A

Difference Table

-	A	C	T	G
A	A	C	T	G
C	C	A	G	C
T	T	G	A	T
G	G	T	C	A

Multiplication Table

⊗	A	C	T	G
A	A	C	T	G
C	C	A	G	T
T	T	G	A	C
G	G	T	C	A

Fig. 4 Algebraic DNA operations

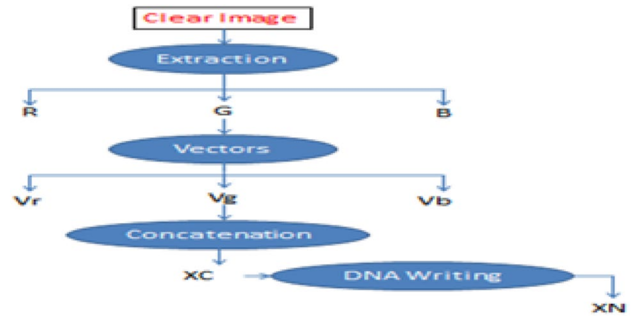


Fig. 5 Setting the image to be encrypted

$\boldsymbol{X}\left(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{3nm}\right)$ size vector (1.3 nm), this operation is defined by the algorithm 6:

Algorithm 5

```
{ $for\ i = 1\ to\ nm$
{ $X(3\boldsymbol{i} - 2) = \boldsymbol{V_r}(i)$
{ $X(3\boldsymbol{i} - 1) = \boldsymbol{V_g}(\boldsymbol{i})$
{ $X(3\boldsymbol{i}) = \boldsymbol{V_b}(i)$
{ $Next\ i$
```

(b) Writing clear pixels in nucleotides

To write the pixels of the image vector X in nucleotides, we apply the permutation (***PR***) on the second line of the table (***NT***). To determine the values of the nucleotide complements that will be assigned to the pixels, we apply the permutation (***RP***) on the second line of the table (***TN***). This phase is provided by the algorithm 7

Algorithm 6

```
{ $for\ i = 1\ to\ 4$
{ $TN(1;\boldsymbol{i}) = NT(1;\boldsymbol{i})$
{ $TN(2;\boldsymbol{i}) = NT(2;\boldsymbol{PR}(\boldsymbol{i}))$
{ $TN(3;\boldsymbol{i}) = NT(3;\boldsymbol{RP}(\boldsymbol{i}))$
{ $Next\ i$
```

In our example, the final nucleotide values for the clear image are displayed in table (***TN***)

(***TN***)	Final value	0	1	2	3
	Nucleotides	G	T	C	A
	Complementary	A	T	G	C

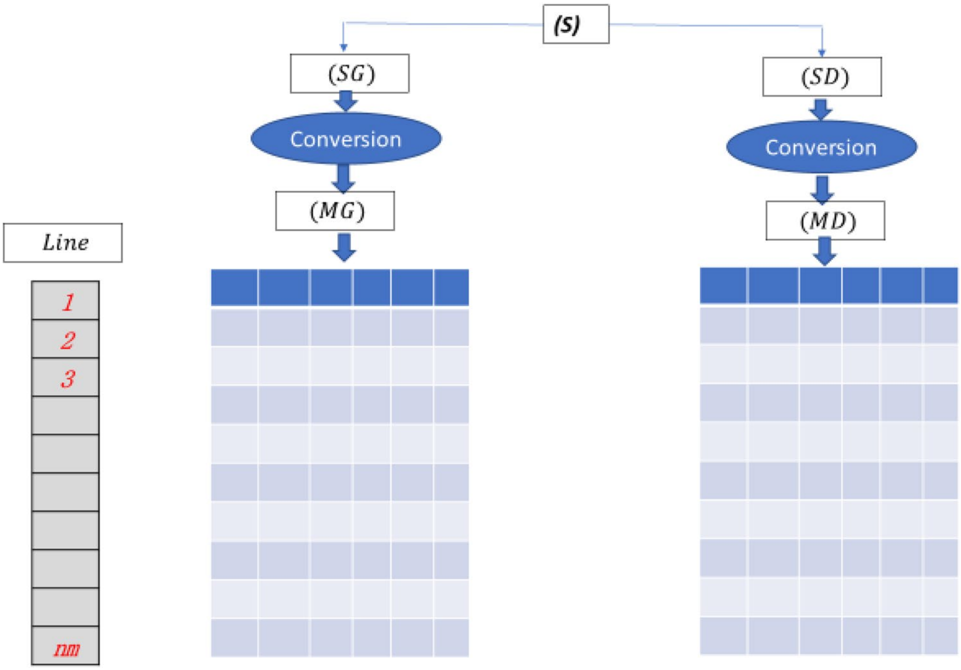
To simplify the transaction of the integer values of the pixels of the vector X into nucleotides, we will use the matrix (***MN***) described below

Value	***Writing in G_4***		***Nucleotide representation***	
0	**0**	**0**	**TN(2,1)**	**TN(2,1)**
1	**0**	**1**	**TN(2,1)**	**TN(2,2)**
2	**0**	**2**	**TN(2,1**	**TN(2,3)**
3	**0**	**3**	**TN(2,1)**	**TN(2,4)**
4	**1**	**0**	**TN(2,2)**	**TN(2,1)**
5	**1**	**1**	**TN(2,2)**	**TN(2,2)**
6	**1**	**2**	**TN(2,2)**	**TN(2,3)**
7	**1**	**3**	**TN(2,2)**	**TN(2,4)**
8	**2**	**0**	**TN(2,3)**	**TN(2,1)**
9	**2**	**1**	**TN(2 ,3)**	**TN(2,2)**
10(A)	**1**	**2**	**TN(2,3)**	**TN(2,3)**
11(B)	**2**	**3**	**TN(2,3)**	**TN(2,4)**
12(C)	**3**	**0**	**TN(2,4)**	**TN(2,1)**
13(D)	**3**	**1**	**TN(2,4)**	**TN(2,2)**
14(E)	**2**	**1**	**TN(2,4)**	**TN(2,3)**
15(F)	**3**	**1**	**TN(2,4)**	**TN(2,4)**

Matrix for converting pixels to nucleotide

Initially, every pixel $\left(\boldsymbol{x_k}\right)$ will be transformed into a hexadecimal value using the format $\left(\boldsymbol{G_k D_k}\right)$. The Eq. 3 explains this process.

Fig. 6 Construction of the left and right matrix



$$\begin{cases} \forall x \ With\ 0 \le x \le 255 \\ \boldsymbol{x_k} = \boldsymbol{G_k D_k} \\ With \\ \begin{cases} \boldsymbol{G_k} = E\left(\frac{x_k}{16}\right) \\ \boldsymbol{D_k} = \boldsymbol{x_k} - 16 * \boldsymbol{G_k} \end{cases} \end{cases} \quad (3)$$

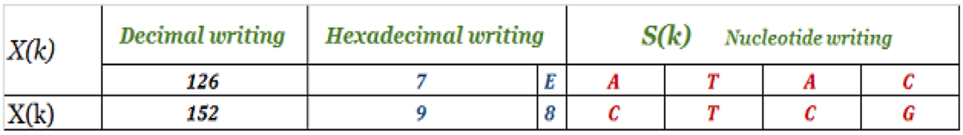
Later, each $(\boldsymbol{G_k})$ block and $(\boldsymbol{D_k})$. block will be transcribed into nucleotides values using the $(\boldsymbol{MN})$ matrix.

The decomposition of the pixel $(\boldsymbol{x_k})$ produces the vector $(\boldsymbol{S_k})$ is expressed in the algorithm

Algorithm 7

```
for $i = 1$ to $2$
$\boldsymbol{s_k}(\boldsymbol{i}) = MN(\boldsymbol{G_k}, \boldsymbol{i} + 2)$
$\boldsymbol{s_k}(\boldsymbol{i} + 2) = MN(\boldsymbol{D_k}, \boldsymbol{i} + 2)$
Next i
```

Example:



X(k)	Decimal writing	Hexadecimal writing		S(k) Nucleotide writing			
	126	7	E	A	T	A	C
X(k)	152	9	8	C	T	C	G

(c) Decomposition of the nucleotide image vector into two matrices

It is known that every Feistel round needs two blocks of the same size for execution. One block is called left and another one says right. For this reason, in our approach, we will disaggregate the vector $(\boldsymbol{S})$ obtained by converting pixels into nucleotides, partition it into two vectors $(\boldsymbol{SG})$ and (SD) of size (1.6 nm) each, then translate (SG) into a matrix $(\boldsymbol{MG})$ including the left block lines and the vector $(\boldsymbol{SD})$ into the matrix $(\boldsymbol{MD})$ of right blocks. Each matrix is of size (nm, 6). Therefore, this breakdown of the original image into two block nucleotide matrices is established by the scheme in Fig. 6:

(d) Building The Initialization Vectors

Two boot $(\boldsymbol{Vl})$ and $(\boldsymbol{Vd})$ vectors are built. The first from the $(\boldsymbol{MG})$ matrix and the second from the $(\boldsymbol{MD})$ matrix. This process is controlled by the algorithm 9

Algorithm 8

```
for $i = 1$ to $6$
$Vl(i) = VN(i + m)$
$Vd(i) = NV(i + n)$
for $k = 2$ to $nm$
$Vl(\boldsymbol{i}) = Vl(\boldsymbol{i}) + MG(k, i)$
$Vd(\boldsymbol{i}) = Vd(\boldsymbol{i}) + MD(k, i)$
Next $k$
Next $i$
```

The vector (***Vl***) will be constructed for modifying the value of the first line of the (***MG***) matrix; when the vector (***Vd***) will change the value of the first line of the (***MD***) matrix. We note that the boot vectors are sensitive to the initial conditions used.

A. Step 3:

(2) Encryption the clear image

Figure 6, emphasizes the process of encryption of a color image by our approach. After separating the clear image into blocks of 12 nucleotides and modifying the value of the first block by the initialization vector, built from the clear image and the chaotic maps used. Two Feistel rounds are performed on each block. The first round, is a chaotic displacement of nucleotides by the component-based vector (***CS***) in $\boldsymbol{G}_6$, while the second is provided by a (***CM***) confusing matrix. At the end of the encryption procedure, a broadcast with the following clear block is established, to maximize the avalanche effect and protect the system from known differential attacks. Then, a transition to complementary nucleotides, followed by an application of m-RNA on the whole vector for the construction of amino acids, and t-RNA for the isolation of EXONS blocks and INTRONS blocks. The encryption steps for the color image are outlined in the diagram in the Fig. 7.

The figure shown above can be translated by the algorithm 10

(a) Analytical expression on the encryption function

Algorithm 9

```
for$i = 1$to$nm$
$U(i) = IV + U(i)$
Do two Feistel rounds
Concatenate the two output blocks of the two towers
Do $IV = U'(i)$
Go back to the beginning step
Diffusion with the next clear block
transition to complementary
Applying $RNA_m$
Applying $RNA_t$
Cypher image
```

(a) First Feistel-Round

The first Feistel round applied to the nucleotide blocks is assisted by a chaotic displacement at the nucleotides level of a step derived from the coordinates of the vector (***CS***) previously fabricated as a component in the ring $\boldsymbol{G}_6$. It is displayed in the Fig. 8 below

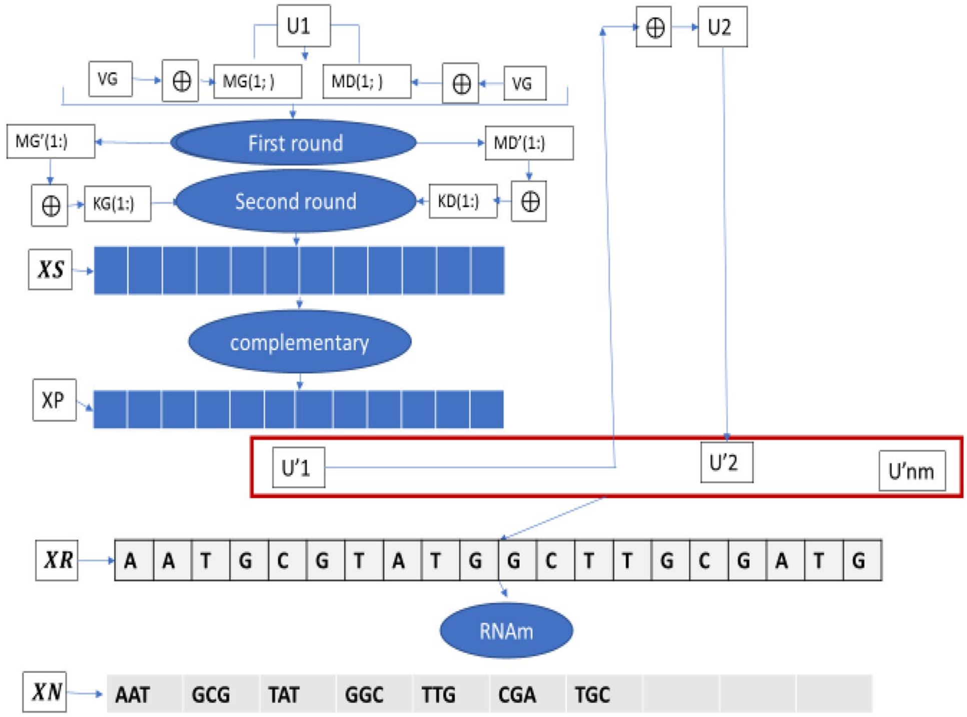


Fig. 7 Encryption of the original image

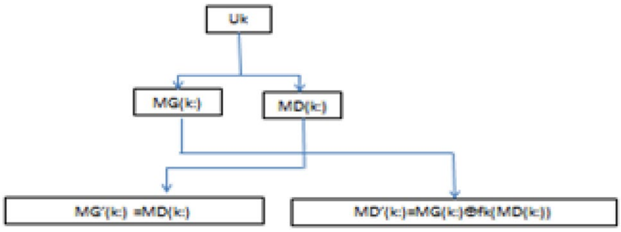


Fig. 8 First Feistel round

This round may be expressed by the following numerical expression

$$\varphi_k^1(\boldsymbol{MG}(\boldsymbol{k}\ :),\boldsymbol{MD}(\boldsymbol{k}\ :))=\begin{cases}MG'(k\ :)=MD(k\ :)\\ MD'(k\ :)=MG(k\ :)\otimes \boldsymbol{f_k}(\boldsymbol{MD}(\boldsymbol{k}\ :))\end{cases} \tag{4}$$

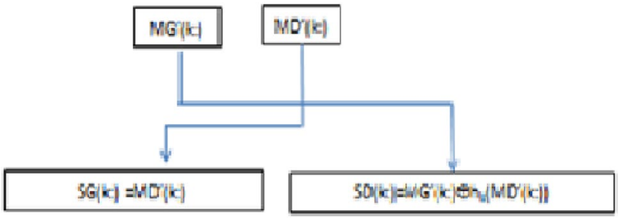


Fig. 9 Second round

$$\varphi_k^2\left(\boldsymbol{MG'}(\boldsymbol{k}\ :),\boldsymbol{MD'}(\boldsymbol{k}\ :)\right)=\begin{cases}SG(k\ :)=MD'(k\ :)\\ SD(k\ :)=MG'(k\ :)\otimes \boldsymbol{h_k}\left(\boldsymbol{MD'}(\boldsymbol{k}\ :)\right)\end{cases} \tag{5}$$

This round may be expressed by the following numerical expression in the algorithm 13

Algorithm 12

```
$\varphi_k^2\left(\boldsymbol{MG'}(\boldsymbol{k}\ :),\boldsymbol{MD'}(\boldsymbol{k}\ :)\right) =$
    $for\ i = 1\ to\ 6$
    $SG(k\ :\ i) = MD'(k\ :\ i)$
    $Next\ i$
    $for\ i = 1\ to\ 6$
    $SD(\boldsymbol{k}\ :) = \boldsymbol{MG'}(\boldsymbol{k}\ :) \otimes \left(CM(\boldsymbol{k},\boldsymbol{i}) + \boldsymbol{MD'}(\boldsymbol{k}\ :\ \boldsymbol{i})\right)$
    $Next\ i$
```

The mathematical expression of Eq. 4 is assigned by the algorithm 11

Algorithm 10

```
$for\ i = 1\ to\ 6$
$\boldsymbol{f_k}(MD(\boldsymbol{k}\ :\ ) = MD(k\ :\ mod(\boldsymbol{i} + \boldsymbol{LC}(\boldsymbol{k}), 6))$
$Next\ i$
```

Finally: the first-round mathematical expression is obtained through the algorithm 12

So, in our approach the mathematical expression of two Feistel's rounds applied to block $\boldsymbol{U_k}((\boldsymbol{MG}(\boldsymbol{k}\ :),\boldsymbol{MD}(\boldsymbol{k}\ :))$ is given by algorithm 14: We pose $\left\{\ FEistel2=\varphi_k^2 o\varphi_k^1)\ \right\}$.

At the end of the encryption mode using two Feistel rounds, a connection is established between the encrypted block and the next clear block. Such a technique greatly augments the avalanche impact and prevents known differential attacks on the system.

Algorithm 11

```
$\varphi_k^1(\boldsymbol{MG}(\boldsymbol{k}\ :),\boldsymbol{MD}(\boldsymbol{k}\ :)) =$
    $for\ i = 1\ to\ 6$
    $MG'(\boldsymbol{k},\boldsymbol{i}) = MD(k,i)$
    $Next\ i$
    $for\ i = 1\ to\ 6$
    $MD'(k;i) = MG(\boldsymbol{k},\boldsymbol{i}) \otimes MD(\boldsymbol{k},\boldsymbol{mod}(\boldsymbol{i} + \boldsymbol{LC}(\boldsymbol{k}), 6))$
    $Next\ i$
```

(b) Second Feistel-Round

The second Feistel round applied to the output block is accomplished by a confusion matrix (***CM***) at the nucleotide level by the operator (+). This is indicated in Fig. 9.

This may be seen as Eq. 5

(b) Transition to complementary

After concatenation of all the encrypted blocks, in such a way as to form the vector (***S***), a transition to the complementary is achieved. After this action a mistake is caused with the chaotic vector (***CL***). This is demonstrated by the algorithm 15

Algorithm 13

```
$for\ i = 1\ to\ 8\,nm$
$S(\boldsymbol{i}) = \overline{\boldsymbol{S}(\boldsymbol{i})} \otimes VN(\boldsymbol{i})$
$Next\ i$
```

(3) Applying m-RNA

m-RNA is the transformation of nucleotides into amino acids whose total number is 64. It is the grouping by three nucleotides. This procedure is documented through the algorithm 16

Algorithm 14

```
$for\ k = 1\ to\ 12nm$
$XR(\boldsymbol{k}) = S(3\boldsymbol{k} - 2)X(3\boldsymbol{k} - 1)S(3k)$
$Next\ k$
```

The amino acid vector value (***XR***) is a component in $(\boldsymbol{G}_{64})$.

Example

The transcribed gene is defined by a step shift equal to **4 nm-D5**, so the first codon of the unexplained gene is the **D5** codon.

The displacement of the codons in direction 5 is indicated by the algorithm 17

(b) Design the direction 3′

Algorithm 15

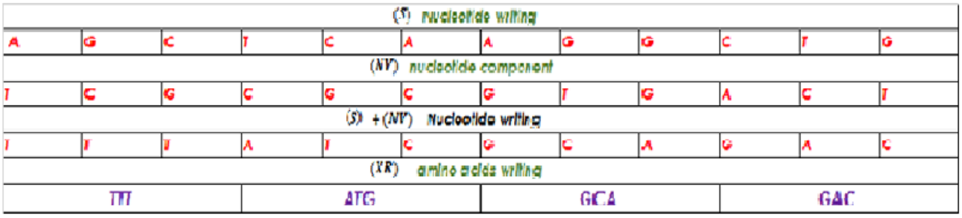
```
$for\ k = 1\ to\ 4\,nm$
$GN(\boldsymbol{k}) = XR((k + 4\,nm - D5)mod4\,nm)$
$Next\ k$
```

The reading direction 3′ of the t-RNA, represents the end of the transcribed gene. In our case, it is the (***GN***) codon value (4 nm).

So

$$\boldsymbol{D}3 = \boldsymbol{GN}(4\,\boldsymbol{nm}) \tag{7}$$

This 3′ direction is computed to affect the level of the stop codon which will delimit the Exon genes and Intron genes.



$(\bar{S})$ nucleotide writing											
A	G	C	T	C	A	A	G	G	C	T	G
(NV) nucleotide component											
T	C	G	C	G	C	G	T	G	A	C	T
$(S) + (NV)$ Nucleotide writing											
T	T	T	A	T	C	G	C	A	G	A	C
(XR) amino acids writing											
TTT			ATG			GCA			GAC		

(a) Applying t-RNA

The application of **t-RNA** on the (***XR***) vector is to isolate the **EXONS** from the **INTRONS** blocks. So, first it computes the two directions 5′ and 3′ then the stop codon by the following formulas

(a) Design the direction 5′

Direction 5′ is the departure point for the codons to be scanned by the t-RNA of the transcribed gene at the output of the encryption operating mode. This direction indicates the index value of the first codon in the displaced gene. Its value is calculated by the Eq. 6

$$\boldsymbol{D}5 = \boldsymbol{Int}\left(10^{10}\left(\frac{(3\boldsymbol{u}(\boldsymbol{n} + \boldsymbol{m}) + 2\boldsymbol{v}(\boldsymbol{m}) + \boldsymbol{u}(\boldsymbol{n}))}{6}\right)\boldsymbol{mod}3\boldsymbol{nm}\right) + 1 \tag{6}$$

(c) Compute the stop codon

The codon stop is the codon which delimits **Exons** and **Introns** in reading the output gene. The reading is made in the 5′ to 3′ direction. the codon stop value is determined by the two chaotic maps already used for the formula:

$$\boldsymbol{stop} = \boldsymbol{Int}\left(\left(\frac{(\boldsymbol{GN}(\boldsymbol{D}5) + \boldsymbol{D}3)}{5}\right)\boldsymbol{modulo}63\right) \tag{8}$$

We observe that the codon **stop** relies on the two directions of t-RNA readings. Reading the transcribed gene is performed by the t-RNA from the D5 codon to the first stop codon to form the first **EXON** block and continues until the second stop codon to form the **INTRON** first block and until the gene-end.

(b) Applying r-RNA

For **r-RNA**, the **EXONS blocks** identified include some genetic information, So for our approach, they will be dynamically offset, while the **INTRONS blocks** detected do not incorporate any genetic information and will not be affected by our algorithm.

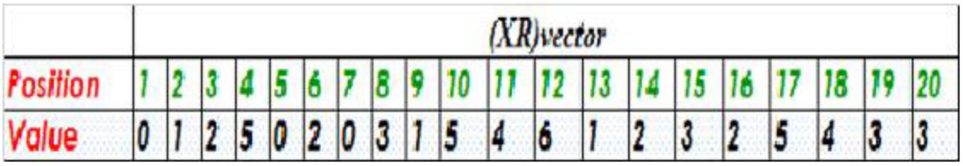
(a) Compute the step of the shift

In our simulation, the step of the element displacement offset of an **Exon** block, depends on its position in the gene. It is depicted by the following formula:

$$\boldsymbol{step}(\boldsymbol{i}) = \boldsymbol{CS}(\boldsymbol{i})\boldsymbol{modulo}(\boldsymbol{L}) \quad (9)$$

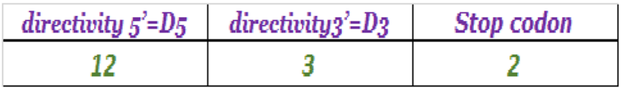
L: represents a length the **EXON** block of position i. We note that: $\forall i \boldsymbol{step}(\boldsymbol{i}) \neq 0$.

Example: In $\left(\boldsymbol{G}_8\right)$



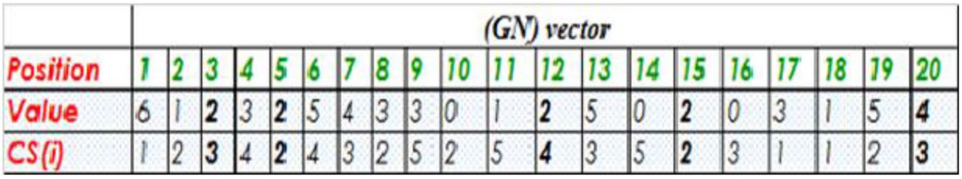
	(XR)vector																			
Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0	1	2	5	0	2	0	3	1	5	4	6	1	2	3	2	5	4	3	3

The calculation of the two directions and the stop codon is shown in the table



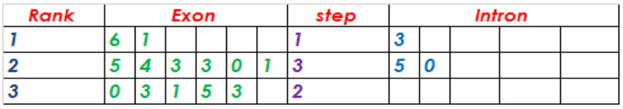
directivity 5'=D5	directivity3'=D3	Stop codon
12	3	2

this produces the gene shown in the vector (GN).



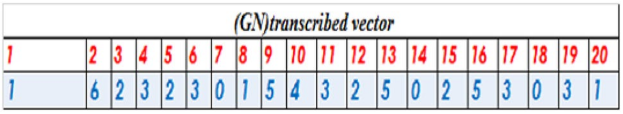
	(GN) vector																			
Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	6	1	**2**	3	**2**	5	4	3	3	0	1	**2**	5	0	**2**	0	3	1	5	**4**
CS(i)	1	2	**3**	4	**2**	4	3	2	5	2	5	**4**	3	5	**2**	3	1	1	2	**3**

Exon and Intron Genes Extraction is given in the table



Rank	Exon						step	Intron				
1	6	1					1	3				
2	5	4	3	3	0	1	3	5	0			
3	0	3	1	5	3		2					

The output genus is



(GN)transcribed vector																			
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	6	2	3	2	3	0	1	5	4	3	2	5	0	2	5	3	0	3	1

B. Step 4:

(4) Encrypted image decryption

Our approach is a symmetric encryption system. The encryption key will also be required to decrypt the color image from the encrypted image. We use the same key but with inversion of encryption procedures. After decomposing the encrypted image into 24-bit blocks. The decryption process begins with the last block and ends with the second block. A recalculation of the initialization vector is performed and the value of the first block is found. The reverse function of the two turns of Feistel applied in this algorithm is given by the Eq. 9

$$\begin{cases} Feistel2\left(\boldsymbol{U}_{\boldsymbol{k}}\right) = \boldsymbol{U'}_{\boldsymbol{k}} \\ So \\ \boldsymbol{U}_{\boldsymbol{k}} = \left(\varphi_k^2 \boldsymbol{o} \varphi_k^1\right)^{-1}\left(\boldsymbol{U'}_{\boldsymbol{k}}\right) = \left(\varphi_k^1\right)^{-1} o \left(\varphi_k^2\right)^{-1}\left(\boldsymbol{U'}_{\boldsymbol{k}}\right) \end{cases} \quad (9)$$

The reciprocal of the two Feistel tours is given by the algorithm 17

Algorithm 16

$$\boldsymbol{Fistel}2((\boldsymbol{MG}(\boldsymbol{k}\ :)\boldsymbol{MD}(\boldsymbol{k}\ :)) = (\boldsymbol{SG}(\boldsymbol{k}\ :), \boldsymbol{SD}(\boldsymbol{k}\ :)) = \begin{cases} SG(\boldsymbol{k}\ :) = MG(k\ :) \otimes (MS(\boldsymbol{k}\ :) + MD(\boldsymbol{k}\ :)) \\ SD(\boldsymbol{k}\ :) = MG(\boldsymbol{k}\ :) \otimes \boldsymbol{h_k} SG(k;) \end{cases}$$

the final expression of the inverse function is given by the algorithm 18

Algorithm 17

$$\boldsymbol{So}\left(\varphi_k^2 \boldsymbol{o} \varphi_k^1\right)^{-1} \begin{cases} MG(k\ :) = SG(k\ :) \otimes \boldsymbol{f_k}(SD(k\ :) \otimes \boldsymbol{h_k}(SG(\boldsymbol{k}\ :)) \\ MD(k\ :) = SD(k\ :) \otimes \boldsymbol{h_k}(\boldsymbol{SG}(\boldsymbol{k}\ :)) \end{cases}$$

This following table gives the original image and its histogram, as well as the encrypted image and its histogram

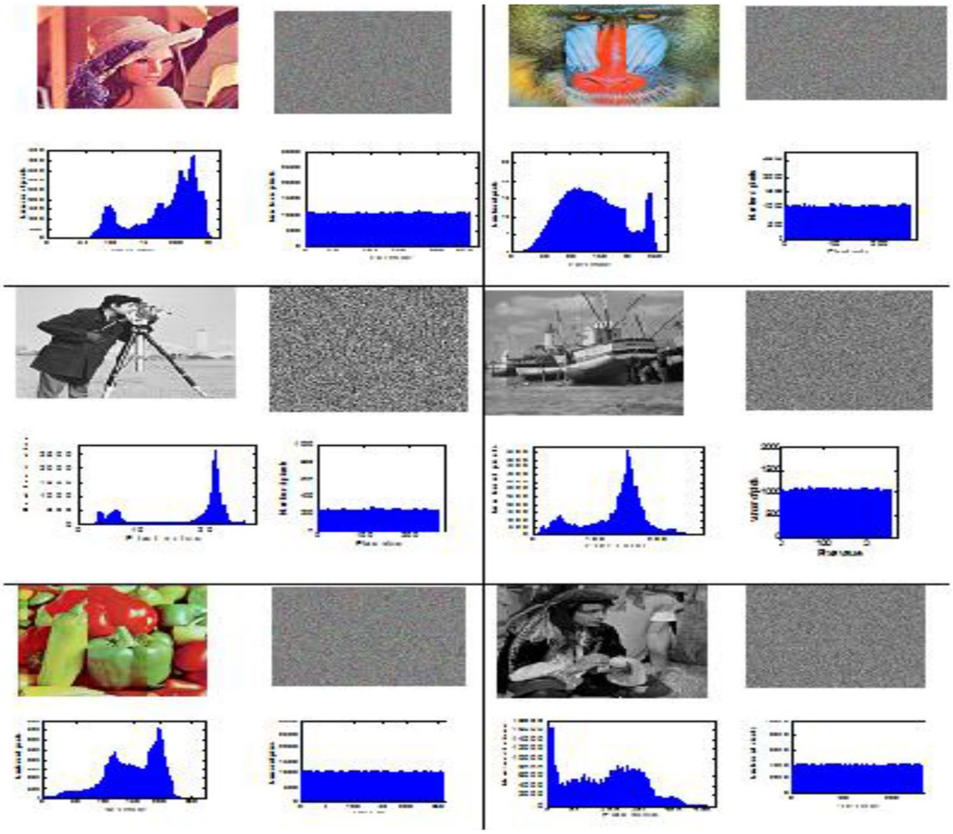
The following table illustrates the encryption of some of the most commonly used images in encryption theory

Example:

A good system crypto must face all known attacks. Our approach has been tested on a database of color images of different sizes and formats. All encrypted images obtained have a flat histogram. This ensures that our crypto system is safe from statistical attacks.

(a) Key space

If the precision of the computing is 10 decimal digits, then the size of the encryption key in our approach is $10^{40} \approx 2^{120} \gg 2^{100}$ which is more than enough to protect our method from brutal attacks.



We note that all the histograms of the images tested in our approach have been flattened. Obtaining the uniform histograms of the encrypted images ensures that our system is protected from statistical attacks.

C. Step 5:

(1) Investigation of cryptosystem performance

(b) Secret key's sensitivity Analysis

The high sensitivity of the encryption keys used in our system indicates that a very slight degradation of the encryption key automatically leads to an image that is so different from the original image. This confirmation can be viewed below the scheme in the Fig. 10.

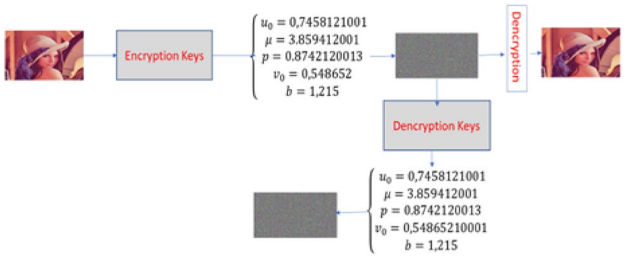


Fig. 10 Secret key's sensitivity

We note that a 10^{-15} change in a single encryption parameter of this technology is incapable of restoring the clear image by the same decryption process.

c) Entropy Analysis

According to SHANON, information entropy is the amount of information encompassed or released by a random information source. In particular, the more redundant the source, the less information it contains. Therefore, the entropy is maximum for a source whose symbols are all equiprobable. The entropy expression is given by SHANNON by Eq. 10. For an (MC) image of size (n, m), we pose $(t = nm)$ one gets:

$$\textbf{\textit{Entropy H}}\,(\boldsymbol{MC}) = \frac{1}{\boldsymbol{t}}\sum_{\boldsymbol{i}=1}^{\boldsymbol{t}} -\boldsymbol{p}(\boldsymbol{i})\log_2(\boldsymbol{p}(\boldsymbol{i})) \tag{10}$$

Simulations performed on 70 identical-sized color images are represented graphically by the Fig. 11.

The entropy values of the images encrypted by our algorithm are around 8, it is the maximum value for a color image encoded on 8 bits. It confirms the uniformity of the histograms. This proves that this approach is safe from entropy attack.

Simulations conducted on 70 color images of varying sizes are represented in graphical format in Fig. 12.

(d) Correlation analysis

Correlation is a technique that compares two images to estimate the displacement of pixels in one image relative to another reference image. Adjacent pixels of a standard image of a clear image have a strong correlation. A good crypto image system must remove such correlation in order to avoid any statistical attack. The correlation expression is defined by Eq. 11

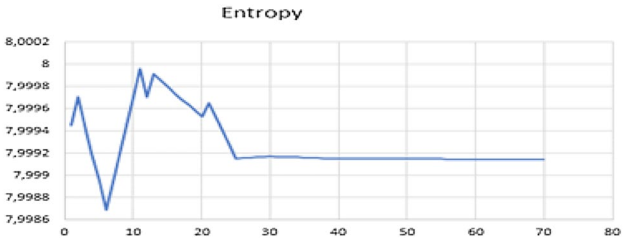


Fig. 11 Entropy of 70 images of the same size

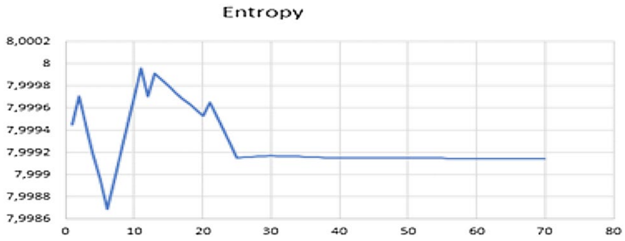


Fig. 12 Entropy of 70 images of the varying sizes

$$\textbf{\textit{correlation r}} = \frac{\mathrm{cov}(\boldsymbol{x},\boldsymbol{y})}{\sqrt{\boldsymbol{V}(\boldsymbol{x})}\sqrt{\boldsymbol{V}(\boldsymbol{y})}} \tag{11}$$

(a) Horizontal correlation

Simulations made on 70 images of the database gave the horizontal correlation scores are displayed in Fig. 13.

Figure 10 shows that the horizontal correlation values of the encrypted images are close to zero. This ensures high security against correlation attacks.

(b) Vertical correlation

Simulations made on 70 images of the database gave the vertical correlation scores are displayed in Fig. 14.

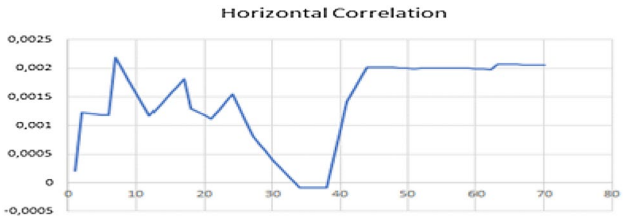


Fig. 13 Horizontal correlation of 70 images of the varying sizes

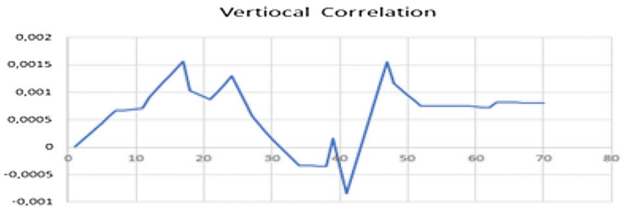


Fig. 14 Vertical correlation of 70 images of the varying sizes

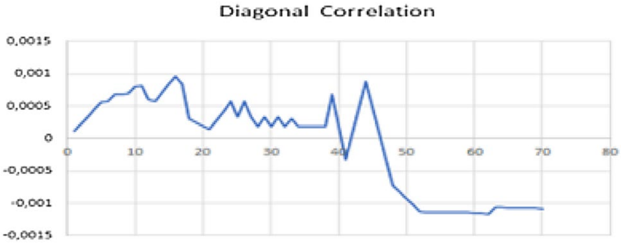


Fig. 15 Diagonal correlation of 70 images of the varying sizes

Figure 11 shows that the vertical correlation values of the encrypted images are close to zero. This ensures high security against correlation attacks.

(c) Diagonal correlation

Simulations made on 70 images of the database gave the diagonal correlation scores are displayed in Fig. 15

Figure 11 shows that the diagonal correlation values of the encrypted images are close to zero. This ensures high security against correlation attacks.

(2) Differential analysis

In general, an attacker can make a very small change on the clear image (for example, change only one bit, then study the change in the result obtained. In so doing, it may be able to find a relevant relationship with the original image and the encrypted image. If a small change in the clear image may cause a large change in the encrypted image, in terms of diffusion and confusion, in this case this differential attack would become ineffective and virtually useless. To test the influence of pixel change on the entire encrypted image by the proposed algorithm, two common measures were used: the pixel change rate **(NPCR)** and the unified average pixel change intensity **(UACI)**. Note two encrypted images, whose corresponding free-to-air images differ by only one pixel, from $\left(\boldsymbol{C}_1\right)$ and $\left(\boldsymbol{C}_2\right)$, respectively. The expressions of these two statistical constants are given by Eqs. 12 and 13, for an image size (n, m)

The NPCR mathematical analysis of an image is given by the Eq. 12

$$\boldsymbol{NPCR} = \left(\frac{1}{\boldsymbol{nm}}\sum_{\boldsymbol{i,j}=1}^{\boldsymbol{nm}}\boldsymbol{D}(\boldsymbol{i},\boldsymbol{j})\right) * 100 \qquad (12)$$

$$\boldsymbol{With\ D}(\boldsymbol{i},\boldsymbol{j}) = \begin{cases} 1 \ \ if\ \boldsymbol{C}_1(\boldsymbol{i},\boldsymbol{j}) \neq \boldsymbol{C}_2(\boldsymbol{i},\boldsymbol{j}) \\ 0 \ \ if\ \boldsymbol{C}_1(\boldsymbol{i},\boldsymbol{j}) = \boldsymbol{C}_2(\boldsymbol{i},\boldsymbol{j}) \end{cases}$$

The UACI mathematical analysis of an image is given by the Eq. 13

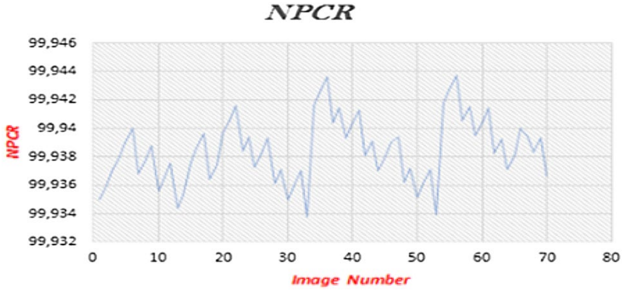


Fig. 16 NPCR of 70 images of the varying sizes

$$\boldsymbol{UACI} = \left(\frac{1}{\boldsymbol{nm}}\sum_{\boldsymbol{i,j}=1}^{\boldsymbol{nm}}\boldsymbol{Abs}\left(\boldsymbol{C}_1(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{C}_2(\boldsymbol{i},\boldsymbol{j})\right)\right) * 100 \qquad (13)$$

The figure (Fig. 16) shows the NPCR calculation of 70 images of different sizes.

All detected values are inside the confidence interval [99, 93299, 946]. These values are largely **sufficient** to affirm that our crypto system is protected from known differential attacks

The figure (Fig. 17) shows the UACI calculation of 70 images of different sizes

All detected values are inside the confidence interval [33;3433, 35]. These values are largely sufficient to affirm that our crypto system is protected from known differential attacks.

(a) Avalanche effect

The avalanche effect is a required property in virtually all cryptographic hash functions and block coding algorithms. It causes progressively more important changes as the data is propagating in the structure of the algorithm. Therefore, by perturbing a single bit at the input, we can obtain a very different output, (about 1 bit out of 2 changed) explaining the name of this phenomenon. The avalanche effect makes it more difficult to reverse the function due to its chaotic properties (if well designed).

This constant determines the avalanche impact of the cryptographic structure in place. It is approximated by the Eq. 14

$$AE = \left(\frac{\sum_i bit\ change}{\sum_i bit\ total}\right) * 100 \qquad (14)$$

Figure 13 depicts the evaluation of the AE score for 70 images examined by our approach.

All values returned from the AE by our method are all in the range of residual values [73, 96 74;02]. This

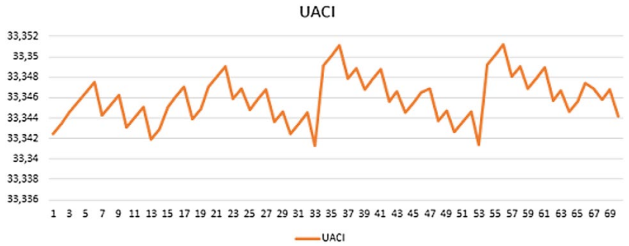


Fig. 17 UACI of 70 images of the varying sizes

guarantees that a one bit change in the clear image will be reflected by a change of at least 78% of the encrypted image's bits.

b) Signal-to-peak noise ratio (PSNR)

(a) MSE

The image quality estimation to be based on the pixel change was obtained by processing the PSNR values and the MSE (Fig. 18). These are the error metrics used to compare the image and the cipher image.

Mean Square Error (MSE): This is the cumulative square deviation between the original image and the additional noise image. When the MSE level is reduced, the error is reduced.

This constant measure the distance between the pixels of the clear image and the encrypted image. It is calculated by the Eq. 15

$$\boldsymbol{MSE} = \sum_{\boldsymbol{i},\boldsymbol{j}} (\boldsymbol{P}(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{C}(\boldsymbol{i},\boldsymbol{j}))^2 \tag{15}$$

$(P(i,j))$; pixel of the clear image
$(C(i,j))$: pixel of the cypher image

(b) PSNR

The signal-to-peak noise ratio, often abbreviated PSNR, is a engineering term for the ratio between a signal's maximum possible power and the power of distorted noise that affects the precision of its display. Since many signals have a very large dynamic range, the PSNR is generally stated in terms of the logarithmic decibel scale. The PSNR mathematical analysis of an image is given by the Eq. 16

$$\boldsymbol{PSNR} = 20\boldsymbol{Log}_{10}\left(\frac{\boldsymbol{I}_{\boldsymbol{max}}}{\sqrt{\boldsymbol{MSE}}}\right) \tag{16}$$

For RGB color images, the definition of PSNR is the same except that the MSE is the sum of all square value changes. In the alternative, for color images, the image is transcoded into a separate color space and the PSNR is displayed for each channel in that color space. The acceptable PSNR values are the real numbers in the domain (5, 10). Simulations made on over 70 images of various magnitudes and formats returned the same results as depicted in the Fig. 19.

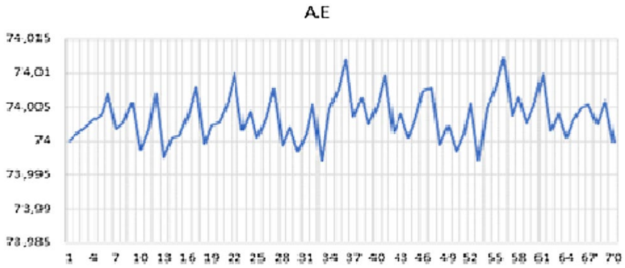


Fig. 18 Avalanche effect

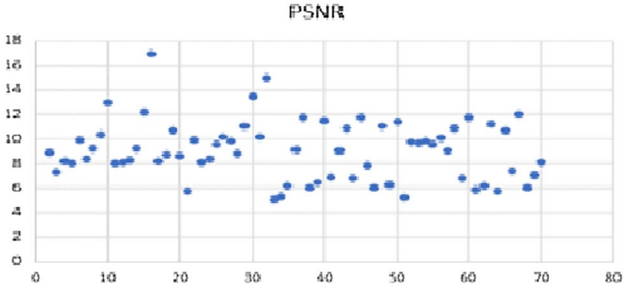


Fig. 19 SNR of 70 images of the varying sizes

All values returned from the **PSNR** by our method are all in the range of residual values [5;412;10].

(c) Speed analysis

Assuming that the traditional DES and AES encryption algorithms operate in ECB mode, they are vulnerable to statistical attacks and selected plain text attacks. In addition, these two systems require no linking on clear and encrypted blocks, and are consequently deficient in the face of differential attacks. In this sense, we will compare the time complexity for reference images with these two crypto systems. In addition to safety parameters, runtime is an important factor in evaluating image encryption system performance. To approve and document the quality of our methodology in a timely fashion. And finally, thanks to these properties, we have selected the "Lena" grayscale image with three different sizes (256 × 256, 512 × 512 and 1024 × 1024). . The results are presented in the Table 1

We compare our results with the two classical algorithms AES and DES, we can affirm that the time of execution is reasonable. The test was performed on other images of different sizes, and we obtained acceptable

Table 1 Execution time (in second)

Image	Our method	DES	AES
Lena (256 × 256)	0.09644	0.639772	5.687244e−002
Lena (512 × 512)	0.17469	7.449005	0.347506
Lena (1024 × 1024)	0.48421	29.11398	1.152980

scores. This is due to the low algorithm complexity of the implemented algorithms in our strategy.

(1) Bio security

(a) Nucleotide statistical analyses

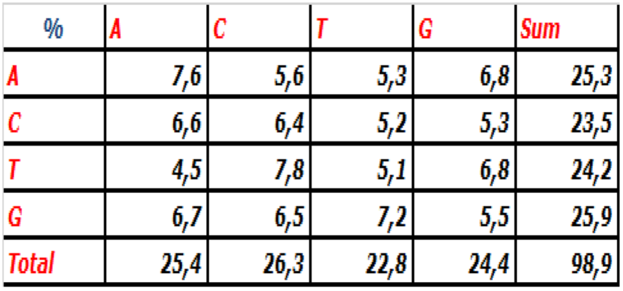
In this part we will study the nucleotide presence frequency in the encrypted image.

(a) Percentage nucleotides isolated

%	A	C	T	G
	24	26	25	25

There is nearly uniformity in the distribution of nucleotides in the encrypted image.

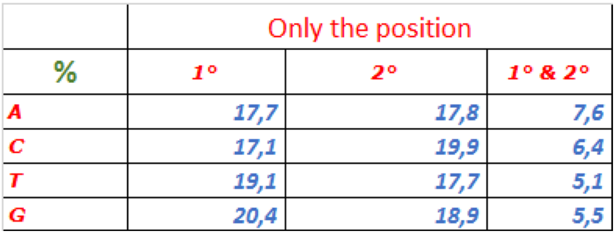
(b) Percentage of nucleotides used in pairs



%	A	C	T	G	Sum
A	7,6	5,6	5,3	6,8	25,3
C	6,6	6,4	5,2	5,3	23,5
T	4,5	7,8	5,1	6,8	24,2
G	6,7	6,5	7,2	5,5	25,9
Total	25,4	26,3	22,8	24,4	98,9

We notice that the template is not symmetrical.

In a regrouping of two nucleotides we notice that:

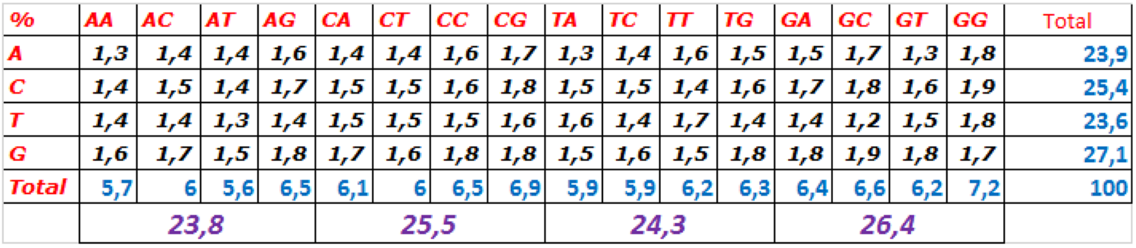


	Only the position		
%	1°	2°	1° & 2°
A	17,7	17,8	7,6
C	17,1	19,9	6,4
T	19,1	17,7	5,1
G	20,4	18,9	5,5

The frequency of occurrence of a nucleotide in the first position is not identical to its incidence in the second position.

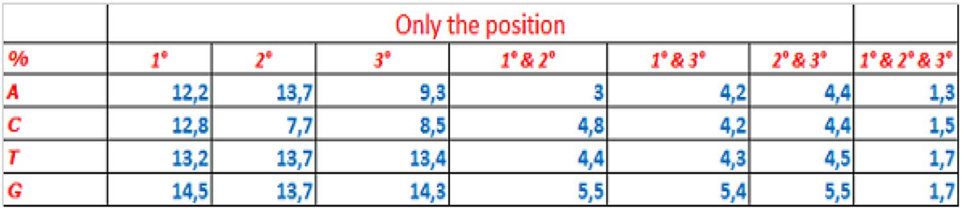
(c) Percentage of codons

In a regrouping of nucleotides into codons, we get the following results.



%	AA	AC	AT	AG	CA	CT	CC	CG	TA	TC	TT	TG	GA	GC	GT	GG	Total
A	1,3	1,4	1,4	1,6	1,4	1,4	1,6	1,7	1,3	1,4	1,6	1,5	1,5	1,7	1,3	1,8	23,9
C	1,4	1,5	1,4	1,7	1,5	1,5	1,6	1,8	1,5	1,5	1,4	1,6	1,7	1,8	1,6	1,9	25,4
T	1,4	1,4	1,3	1,4	1,5	1,5	1,5	1,6	1,6	1,4	1,7	1,4	1,4	1,2	1,5	1,8	23,6
G	1,6	1,7	1,5	1,8	1,7	1,6	1,8	1,8	1,5	1,6	1,5	1,8	1,8	1,9	1,8	1,7	27,1
Total	5,7	6	5,6	6,5	6,1	6	6,5	6,9	5,9	5,9	6,2	6,3	6,4	6,6	6,2	7,2	100
	23,8				25,5				24,3				26,4				

from this table, we extract the frequency of the nucleotide position in the codon:



	Only the position						
%	1°	2°	3°	1° & 2°	1° & 3°	2° & 3°	1° & 2° & 3°
A	12,2	13,7	9,3	3	4,2	4,4	1,3
C	12,8	7,7	8,5	4,8	4,2	4,4	1,5
T	13,2	13,7	13,4	4,4	4,3	4,5	1,7
G	14,5	13,7	14,3	5,5	5,4	5,5	1,7

We notice that the template is not symmetrical. There is nearly uniformity in the distribution of codons in the encrypted image.

In our approach, we pseudo-randomly assigned nucleotides to the integer values of the original image, and to chaotic patterns without going through a DNA gene bank, which made it very difficult or indeed impossible to detect the gene of the DNA. In addition, the complementary values are also assigned in a pseudo-random way and do not obey the calculated classical complementary, so any reconstruction of these values requires knowledge of the encryption keys.

(b) Math security

The large size of our encryption key protects the system from brutal attacks. In parallel, the different pseudo random choice of nucleotides for the transcription of chaotic maps and pixels of the original image complicates the detection and screening of nucleotides and their complementary ones. In addition, the key mask used in conjunction with (One Time Pad) OTP is the same sized as the clear image. It is widely believed that OTP is secured.

3 Conclusion

This article outlines a new method of color image encryption using several biological **DNA** and **RNA** property. Such a technique models the mathematical problem into a biological question. First, two Feistel rounds of 12 nucleotides have been applied to each block. The first is a chaotic shift of nucleotides, while the second is a confusion caused by a random matrix, followed by a diffusion with the next clear block, to increase avalanche response. In a second stage, a confusion of the output vector is established. A transition to complementary nucleotides is implemented, with amino acid formation. Finally, the 5′ and 3′ directions are computed and the stop codons are evaluated, the EXONS blocks are isolated from the **INTRONS** blocks. **EXONS** have been substituted, while **INTRONS** have been chaotically shifted. Simulations conducted on a significant number of color images allow us to ensure that our approach can overcome any known attack.

Compliance with ethical standards

Conflict of interest I am the alone author of this article, and therefore no conflict. To finalize this document, I did not receive any assistance funds from any organization. This document does not contain any studies or experiments on animals.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the author.

References

1. Shanon CE (1949) Communication theory of secrecy systems. Bell Syst Tech J 28(4):656–715
2. Hillion A (1986) Les théories mathématiques des populations. PUF, Paris
3. Jarjar A (2017) Improvement of hill's classical method in image cryptography. Int J Stat Appl Math A 2(3):37–43
4. Hraoui S, Gmira F, Jarar AO, Satori K, Saaidi A (2013) Benchmarking AES and chaos based logistic map for image encryption. In: 2013 ACS international conference computer systems and applications (AICCSA)
5. Zhang G, Liu Q (2011) A novel image encryption method based on total shuffling scheme. Opt Commun 284(12):2775–2780
6. Feng X, Tian X, Xia S (2011) An improved image scrambling algorithm based on magic cube rotation and chaotic sequences. In: 4th international congress on image and signal processing. IEEE Press, pp 1021–1024
7. Dong C (2014) Color image encryptionusing one-time keys and coupledchaoticsystems. J Signal Process Image Commun 29(5):628–640
8. Wanga X-Y, Gua S-X, Zhangab Y-Q (2015) Novel image encryptionalgorithmbased on cycle shift and chaotic system. J Signal Process Image Commun 68:126–134
9. Suzaki T, Minematsu K (2010) Improving the generalized Feistel. In: Hong S, Iwata T (eds) Proceedings of the 17th international workshop fast software encryption, vol 6147, Seoul, Korea, pp 19–39
10. Adleman LM (1994) Molecular computation of solutions of combinatiorial problems. Science 266:1021–1024
11. Xiao GZ, Lu MX, Qin L, Lai XJ (2006) New field of cryptography: DNA cryptography. Chin Sci Bull 51(12):1413–1420
12. Gehani A, LaBean TH, Reif JH (2000) DNA-based cryptography. DIMACS series in discrete mathematics. Theor Comput Sci 54:233–249
13. Liu Z, Gong M, Dou Y, Liu F, Lin S, Ahmad MA et al (2012) Double image encryption by using Arnold transform and discrete fractional angular transform. Opt Lasers Eng 50:248–255
14. Abdullah AH, Enayatifar R, Lee M (2012) A hybrid genetic algorithm and chaotic function model for image encryption. AEU Int J Electron Commun 66:806–816
15. Niu Y, Zhang X, Han F (2017) Image encryption algorithm based on hyperchatic map and nucleotide sequence database. Comput Intell Neurosci 2017:1–9.
16. Handy M, Mousa I-J (2006) DNA, genetic encryption technique. Comput Netw and Inf Secur 7:1–9
17. Liu L, Zhang Q, Wei X (2012) A RGB image encryption algorithm based on DNA encoding and chaos map. Comput Electr Eng 38(5):1240–1248

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.