Medical Imaging Process & Technology 2024, 7(1), 3001.
https://doi.org/10.24294/mipt.v7i1.3001

Article

Multiple image encryption acting at the RNA level

Mariem Jarjar, Abid Abdellah, Hicham Rrghout, Mourad Kattass, Abdellatif Jarjar*, Abdellhamid Benazzi

MATSI Laboratory, Mohamed First University, Oujda 60000, Morocco
*** Corresponding author:** Abdellatif Jarjar, abdoujjar@gmail.com

CITATION

Jarjar M, Abdellah A, Rrghout H, et al. Multiple image encryption acting at the RNA level. Medical Imaging Process & Technology. 2024; 7(1): 3001. https://doi.org/10.24294/mipt.v7i1.3001

ARTICLE INFO

Received: 10 October 2023
Accepted: 15 December 2023
Available online: 20 February 2024

Abstract: The purpose of this research is to develop a new method for encrypting multiple superimposed or side-by-side images. The process begins by extracting the red, green, and blue channels from each image and converting them into vectors that combine to produce a single image that undergoes an advanced pixel-level Vigenere transform. In the next step, a pseudorandom transition occurs at the nucleotide, followed by a passage to codons for genetic crossover implementation specifically designed for image scrambling. The latter process is controlled by many random tables developed from selected chaotic maps, which ensures a high degree of flexibility and security in the encryption method. To evaluate the effectiveness and security of this innovative multi-image encryption algorithm, extensive simulations were performed using a large number of images randomly selected from the database. The simulation results prove the reliability and robustness of the method.

Keywords: chaotic map; nucleotide writing; codon writing; substitution table; new Vigenere scheme; genetic operators

1. Introduction

In a variety of domains, including the military, natural disaster monitoring, traffic surveillance, weather forecasting, e-government, and personal matters, a vast quantity of images are being generated. Preserving the security of multimedia content, especially images, has emerged as a significant obstacle for both academic researchers and industry. As our world becomes increasingly interconnected and data is transmitted over the Internet, concerns about its susceptibility are on the rise. The inherent openness of information exchange protocols exposes them to potential security breaches, providing malicious actors with the opportunity to intercept and access sensitive data belonging to others. Consequently, cryptography has taken on paramount importance as a field, providing the sole recognized means of protection against such malicious attacks.

1.1. Related works

Cryptography's roots can be traced back to ancient times, and its initial steps are often attributed to figures such as Caesar. Over the course of history, it has undergone significant development and has evolved into a contemporary scientific discipline. In the present day, it remains the cornerstone of secure communication. To illustrate the fundamental encryption process, we can refer to **Figure 1**.

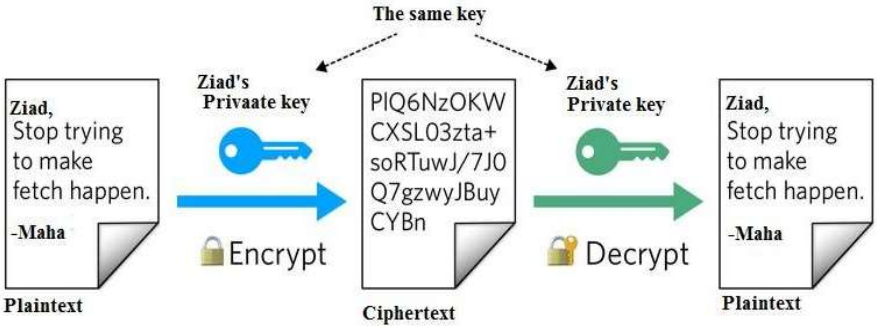


Figure 1. General encryption system.

It remained for a long time reserved for the military domain. With the progress of mathematics, cryptography quickly invaded the world of security. And we have seen the emergence of some standards, such as Vigenere [1], Hill [2], DES [3], Feistel, AES [4], and others. The development of chaos theory, and the availability of multiple chaotic maps facilitate the construction of encryption algorithms. Moreover, with Shannon's prescriptions [5], and the use of chaining in the encryption mode gives a strong robustness to new cryptographic systems. With the entry of images in the transfer between individuals, the cryptography remains the safest way for security. 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. Liu et al. [7] offers color image encryption based on the construction of a coupled chaotic map. Hoang [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 [9] to check the installation of chaos and sensitivity to initial conditions. Most encryption algorithms operating on blocks used the Feistel [10,11] scheme with several turns. RC4, RC6, DES used more than four towers [12–14].

1.2. Problematic

The majority of conventional systems remain vulnerable to frequency and statistical attacks. Moreover, in the absence of any chaining between encrypted blocks and next clear blocks, subsequently, all these classical systems are also exposed to differential attacks.

1.3. Our contributions

In most classical cipher systems, the pixel is the central image encryption algorithm element. In our new technique, we shift towards DNA and then RNA, considering them as central elements and integrating some genetic operators tailored for image encryption. This approach allows us to confront any known attack effectively.

2. The proposed methodology

Based on chaos [15,16], this method is articulated on several main axes, cited below:

2.1. Axe 1: Chaotic sequences selection

Our system is built upon the utilization of the three most extensively employed chaotic maps in cryptography [17,18]. These maps are renowned for their extreme sensitivity to initial conditions. They are generated using a sufficiently large key, rendering them immune to brute-force attacks. The three chaotic maps chosen for our algorithm are described below:

2.1.1. The logistics map

The logistic map(u_n) [19,20] is a recurrent sequence described by a simple polynomial of second degree defined by the following equation:

$$\begin{aligned} &u_0 \in]0{,}5\ 1[\,, \mu \in [3{,}75; 4] \\ &u_{n+1} = \mu u_n (1 - u_n) \end{aligned} \tag{1}$$

It is the most popular and widely used chaotic map in cryptography due to its high sensitivity to initial conditions and easy configuration in any cryptosystem.

2.1.2. A. J chaotic map

This new (W_n) A. J chaotic map [21] is defined by using a piecewise linear function. This chaotic sequence is given by the following formula:

$$\begin{cases} w_0 \in \left[\dfrac{1}{(1+p)}\ \dfrac{p}{(1+p)}\right] p \in [1{,}47; \varphi] \\ f(w_n) = w_{n+1} \begin{cases} p^2 w_n\ if\ 0 \le w_n \le \dfrac{1}{1+p} \\ p - p w_n\ if\ \dfrac{1}{1+p} \le w_n \le 1 \end{cases} \end{cases} \tag{2}$$

This new map is chosen for its great sensitivity to the initial conditions proved by the high value of its Lyapunov exponent which is $\lambda = \mathrm{p} \times \mathrm{Log}(2)$.

2.1.3. The skew tent map (SKTM)

The Skew tent map (V_n) [22,23] will be redefined as the next equation:

$$\begin{cases} \mathrm{v}_0 \in]0\ 1[\mathrm{p} \in]0{,}5\ 1[\\ \mathrm{v}_{n+1} = \begin{cases} \dfrac{\mathrm{v}_n}{\mathrm{p}}\ \text{if}\ 0 \prec \mathrm{v}_n \prec \mathrm{p} \\ \dfrac{1 - \mathrm{v}_n}{1 - \mathrm{p}}\ \text{if}\ \mathrm{p} \prec \mathrm{v}_{n<1} \end{cases} \end{cases} \tag{3}$$

The integration of these three chaotic maps will be employed to derive all the essential parameters required for the efficient and effective operation of our innovative technology. Our new algorithm consists of two rounds, the first, operates at the pixel level, while the second operates at the RNA level.

2.2. Axe 2: Pixel level encryption settings

This round, operates at the pixel level by applying a new, improved Vigenere technique, requiring the parameters construction below

- Confusion Tables (CT)
- Binary Control Tables (BT)
- (SW) strong substitution table.

2.2.1. (CT) table generation

The table (CT) of size (3 knm; 3) with coefficients in (G_{256}) is intended to act as aliasing and diffusion acting on the original image pixels level. Building such a table is described by the following algorithm:

Algorithm 1 (*CT*) table design

1 For i = 1 to 3 knm

2 $CT(i;1) = mod(E(|u(i) - v(i) + w(i)| \times 10^{10}),\quad 253) + 3)$

3 $CT(i;2) = mod\left(E\left(\left(u(i) \times w(i) + v(i)\right) \times 10^{10}\right), 253\right) + 2\right)$

4 $CT(i;3) = mod(E((u(i) \times v(i) + w(i) \times 10^{11}),\quad 253) + 1$

5 Next *i*

2.2.2. (BT) binary tables construction

The (BT) binary table of size (12 knm; 2), used to control any cryptographic operations used in our system. This control table is developed by the following algorithm:

Algorithm 2 (*BT*) table design

1 $For\ i = 1\ to\ 12\ knm$

2 $if\ u(i) < v(i)$ Then

3 $BT(i;1) = 0$ Else $BT(i;1) = 1$

4 $if\ w(i) > \dfrac{2 \times u(i) + 3 \times v(i)}{5}$ then

5 $BT(i;2) = 0$ Else $BT(i;2) = 1$

6 end if

7 Next i

2.2.3. (SW) substitution table design

(*SW*) is a table of pixel values change according to a confusion function which will be defined later. this matrix construction of size (256; 256) is determined by the following steps [23,24].

- The first three (*SW*) lines construction
- Determination of subsequent lines

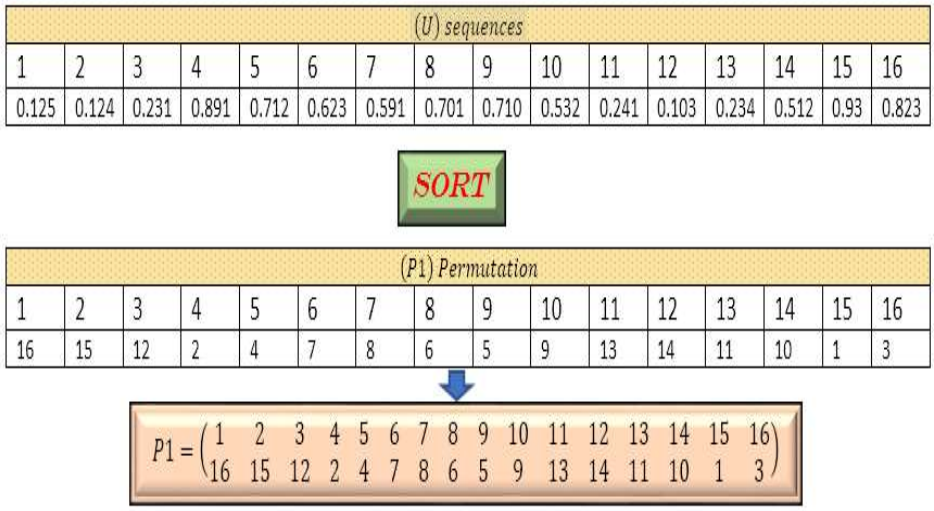
a) First three rows construction

The first three lines represent pseudo-random permutations in (G_{256}) are defined as follows:

- The first line is the permutation (P1) obtained by sorting on the first (256) (U) values
- The second line is the permutation (P2) obtained by sorting on the first (256) (V) values

- The third line is the permutation (P3) obtained by sorting on the first (256) (W) values

Example: Building a permutation in (G_{16}):



These first three (SW) S-Box rows are given by the following algorithm:

Algorithm 3 (SW) first three rows

1 $For\ i = 1\ to\ 256$
2 $SW(1; i) = P1(i)$
3 $SW(2; i) = P2(i)$
4 $SW(3; i) = P3(i)$
5 Next i

b) Following rows definition

The rank line $(i > 3)$ is generated by combining the line $(i - 1)$ and the line $(i - 3)$ or combining the line the line $(i - 2)$ and the line $(i - 1)$, depending on the value of the control vector $(BT(: 2))$. All the following rows are elucidated by the algorithm below.

Algorithm 4 (SW) following rows

1 $For\ i = 4\ to\ 256$
2 $For\ j = 1\ to\ 256$
3 $If\ BT(i; 2) = 0\ Then$
4 $SW(i; j) = SW(i - 1; SW1(i - 3; j))$
5 $Else$
6 $SW(i; j) = SW(i - 2; SW(i - 1; j))$
7 $Next\ j, i$

Example: (SW) in (G_8)

(SW)	1	2	3	4	5	6	7	8	(BT)
P1	2	5	1	8	3	6	4	7	
P2	5	1	8	3	6	2	7	4	0
P3	6	5	7	1	3	2	4	8	1
P4	2	5	7	6	8	1	3	4	1
P5	5	3	4	2	8	6	7	1	1
P6	8	4	2	3	1	6	7	5	0
P7	1	3	4	2	8	6	7	5	1
P8	8	4	2	3	5	6	7	8	0

This new approach uses a pseudorandom obfuscation function to alter pixel values and a chaotic diffusion function to establish connections between scrambled and clear pixels to maximize the avalanche effect impact and protect the system from any attacks.

2.2.4. C/D function used

Two functions have been determined for the first-round application acting at the pixel level, providing confusion and diffusion to protect the system from any known attack:

- Confusion function (VG)
- Diffusion function (DF)

The two used functions expression is given by the following formula:

$$\begin{gathered}\text{VG confusion function (pixel value change)} \\ VG(X(i)) = Y(i) = SW(CT(i;1); SW(CT(i;2); X(i) \oplus CT(i;3)) \\ \text{DF diffusion function (linking)} \\ DF(X(i)) = SW(CT(i;3); Y(i-1)) \oplus X(i)\end{gathered} \tag{4}$$

Now that all the first-round encryption parameters for this first lap, are created, we must prepare the original image to be encrypted.

2.3. Axe 3: Getting the image ready for encryption

Bowing to Kirchhoff's principle, firstly, in this section, we will describe all the necessary steps for preparing the original image before proceeding to the encryption process [25,26].

Given a superposition of (k) images of the same size arranged in a table (M) of size $(1; k)$ like indicated by **Figure 2**:

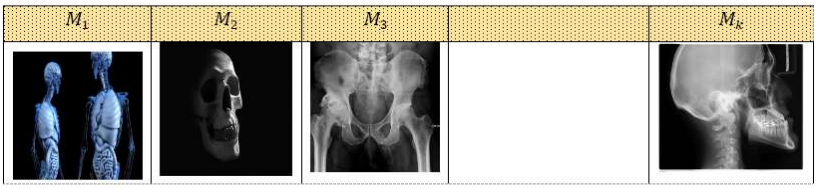


Figure 2. Arranging original images.

From each image (Mi), we extract the three-color channels (RGB) then convert them into vectors $(V_{3i-2}; V_{3i-1}; V_{3i})$. Stacking of vector sequences is performed to obtain a matrix (MG) of size (3k; nm), each row (i) of which, is represented by a vector (Vi) (**Figure 3**).

(MG)

V1	20			120			235		
V2								210	
						40			
									113
		28			14				
			39						
V3k	56			68	68	145			

Figure 3. (MG) development.

The matrix (MG) is defined by the following equation:

$$MG(i;j) = V_i(j)$$
$$i \in [\![1;3k]\!] \text{ and } j \in [\![1;nm]\!] \quad (5)$$

The vector (X) construction is seen under the scheme of **Figure 4.**

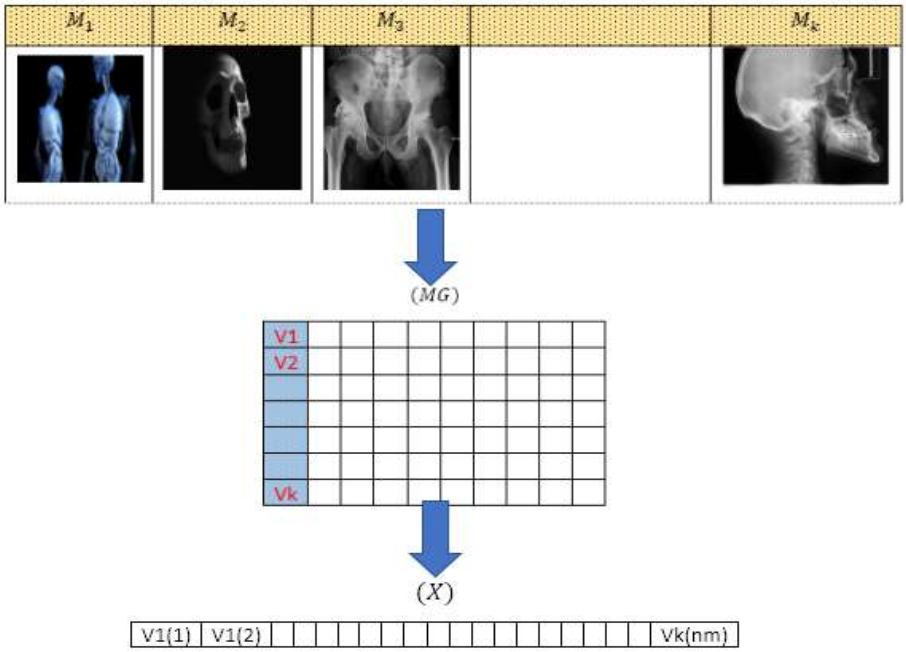


Figure 4. (X) development.

First round encryption

The vector (XC) is an original vector (X) transformation by an advanced Vigenere technique using large pseudorandom S-boxes and incorporating confusion and diffusion functions. However, in order to set the enhanced encryption mode, an initialization value $(v1)$ needs to be calculated.

a) $(v1)$ Starting value estimate

The initialization value $(v1)$ is computed under the binary control vector $(BT(:1))$ is used to taint the starting pixel by the confusion table (CT) and to start the encryption process. This value is given by the algorithm below:

Algorithm 5 Initialization value

1 $for\ i = 2\ to\ 3\ knm$

2 $If\ BT(i;1) = 0\ Then$

3	$v_1 = v_1 \oplus X(i) \oplus CT(i;2)$
4	$Else$
5	$v_1 = v_1 \oplus X(i) \oplus CT(i;3)$
6	$Next\ i$

This first round, uses improved Vigenere's technique and uses a confusion function (VG) and a diffusion function (DF), as shown in **Figure 5.**

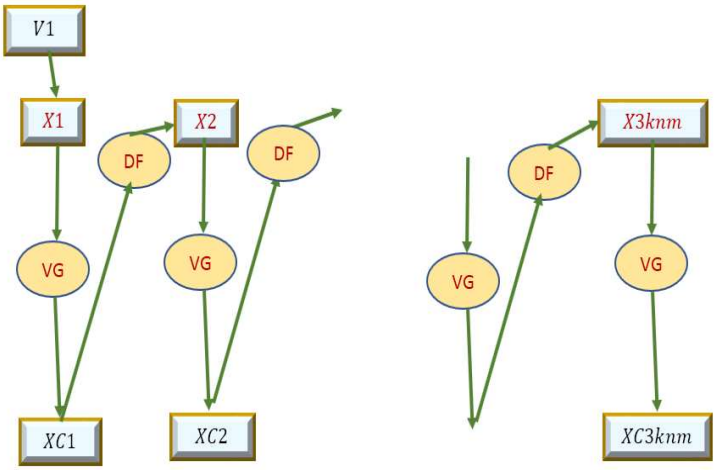


Figure 5. First lap encryption.

This first encryption phase, acting at the pixel level, is determined by the following algorithm:

Algorithm 6 First round

	$Bootstrap\ pixel\ tampering$
1	$XC(1) = VG(v1 \oplus X(1)) \oplus CT(1;3)$
	$First\ encryption\ process$
2	$For\ i = 2\ to\ 3\ knm$
3	$\omega = \mathrm{DF}(X(i))$: diffusion
4	$XC(i) = \mathrm{VG}(\omega) \oplus CT(i;2)$: confusion
5	$Next\ i$

The simulations obtained by applying this first circuit on reference images offer encouraging results as indicated by **Figure 6**.

All statistical parameters values of this first round are conform to international standards to better resist statistical attacks. Likewise, **Figure 7** proves that our system from the first round can cope with differential attacks:

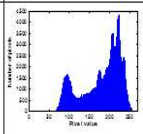
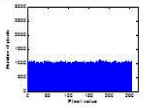
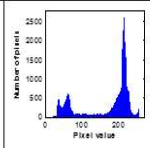
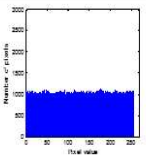
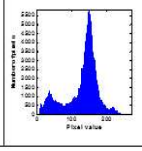
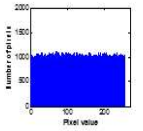
N°	Image	Size	Correlation			Entropy	Clear histogram	Cypher histogram
			Vertical	*Horizontal*	*Diagonal*			
Img1		256*256	0,0010	0,0002	-0,00021	7,9996		
Img2		406*536	0,0012	-0,00031	-0,00012	7,9998		
Img3		361*517	0,00201	-0,00011	-0,000103	7,9997		

Figure 6. Statistical parameters values ($k = 1$).

NPCR and UACI values.

Image						
NPCR	99.61	99.61	99.60	99.63	99.61	99.62
UACI	33.42	33.45	33.44	33.45	33.47	33.43

Figure 7. Differential parameters values ($k = 1$).

In addition, only the replacement is required for the first round, so the execution time complexity is constant and equal to $o(1)$. which encourages submitting the output vector $\mathrm{XC}(\mathrm{XC}_1, \mathrm{XC}_2, \ldots\ldots, \mathrm{XC}_{3\ \mathrm{knm}})$ to a second transformation.

2.4. Axe 4: Passage from pixels to nucleotides

To maximize attack time complexity, a second encryption lap is imposed following the steps below

- (G_4) Passage;
- Transition to nucleotide notation;

2.4.1. (G_4) transition

Each pixel of the vector (XC) will be transformed into four components in (G_4) for the formation of the vector (XB) of size $(1; 12\ \mathrm{knm})$. This transformation is described by the algorithm below:

Algorithm 7 Transition to (G_4)

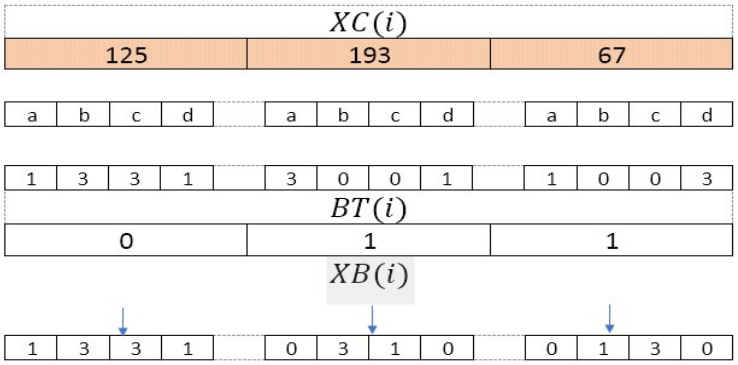
1 $For\ i = 1\ to\ 3\ knm$
$(G_{16}\ Passage)$

2 $x = E\left(\frac{XC(i)}{16}\right)$

3 $y = XC(i) - 16 \times x$

	$(G_4\ Passage)$
4	$a = E\left(\frac{x}{4}\right)$
5	$b = x - 4 \times a$
6	$c = E\left(\frac{y}{4}\right)$
7	$d = y - 4 \times c$
8	$If\ BT(i;1) = 0\ then$
9	$XB(4i-3) = a$
10	$XB(4i-2) = b$
11	$XB(4i-1) = c$
12	$XB(4i) = d$
13	*Else*
14	$XB(4i-3) = c$
15	$XB(4i-2) = a$
16	$XB(4i-1) = d$
17	$XB(4i) = b$
18	*End if*
19	*Next i*

Example: Switch from (XC) to (XB)



The vector (XB) stores the numerical values of each pixel of the vector (XC) in (G_4), will be migrated to DNA

2.4.2. Migration to (DNA)

The human body consists of numerous DNA genes, and their unique arrangement gives each individual a distinct identity referred to as DNA code or genetic makeup. There are four nucleotides in DNA [27,28], depending on the nature of the nitrogenous base:

- deoxyadenosine (adenine base); (A)
- deoxythymidine (thymine); (T)
- deoxyguanosine (guanine); (G)

- and deoxycytidine (cytosine) (C)

a) Nucleotide values storage

In our approach, the nucleotides notation is stored in the table (NC) of dimensions $(r; 4)$, where (r) is a pseudo-random value derived from the utilized tables. This guarantees that the same values in (G_4) can be associated with different nucleotides.

b) Calculating constant (r)

The constant (r), which establishes the height of the table (NC) containing nucleotides, is defined by the following algorithm.

Algorithm 8 First round

1	$x = 0$
2	$For\ i = 1\ to\ nm$
3	$For\ i = 1\ to\ nm$
4	$x = x + CT(i; 1)$
5	$Else$
6	$x = x + CT(i; 3)$
7	$End\ if$
8	$x = mod(x; 17)$
9	$Next\ i$
10	$r = x + 8$

We note that in our simulations:

$$8 \leq r \leq 24 \tag{6}$$

The maximum value of the height of the nucleotide table is $(4! = 24)$

2.4.3. (NC) nucleotide values table

The table (NC) of size $(r; 4)$ contains the likely nucleotides values. The first line contains the nucleotides $(A; C; T; G)$ taken as reference values. The line $(i > 1)$ is a displacement of line $(i - 1)$ of rank $VD(i)$.

a) (VD) displacement vector development

The displacement vector (VD) with coefficient in (G_4) of size $(1; r)$; is determined by the following algorithm:

Algorithm 9 (VD) vector design

1	$for\ i = 1\ to\ r$
2	$If\ BT(i; 1) = 0\ then$
3	$VD(i) = Mod(CT(i; 1); 3) + 1$
4	$Else$
5	$VD(i) = Mod(CT(i; 2); 3) + 1$
6	$End\ if$
7	$Next\ i$

Example: $r = 8$ $(Minimum\ height)$

(NC) Table					(VD)
Value	0	1	2	3	
Nucleotide1	A	C	T	G	1
Nucleotide2	G	T	A	C	2
Nucleotide3	T	A	C	G	3
Nucleotide4	G	T	A	C	1
Nucleotide5	A	C	G	T	2
Nucleotide6	C	G	T	A	3
Nucleotide7	A	C	G	T	1
Nucleotide8	G	T	A	C	2

The vector (VD) is only used to build the table (NC). Any change to a vector coordinate (VD) will regenerate a different nucleotide table value.

2.4.4. (*XN*) Vector nucleotides construction

The passage of the elements of (G_4) in DNA notation requires the construction of the vector (VE) to component in (G_r) by the following algorithm.

Algorithm 10 (*VD*) vector design

1 $for\ i = 1\ to\ 12\ knm$
2 $If\ BT(i;1) = 0\ then$
3 $VE(i) = Mod(CT(i;3);r) + 1$
4 $Else$
5 $VE(i) = Mod(CT(i;2);r) + 1$
6 $End\ if$
7 $Next\ i$

The transformation and transition from vector (XB) to (XN) is described by the following algorithm.

Algorithm 11 (*XN*) vector design

1 $for\ i = 1\ to\ 12\ knm$
2 $XN(i) = NC(VE(i); XB(i) + 1))$
3 $Next\ i$

All these transitions are seen under **Figure 8.**

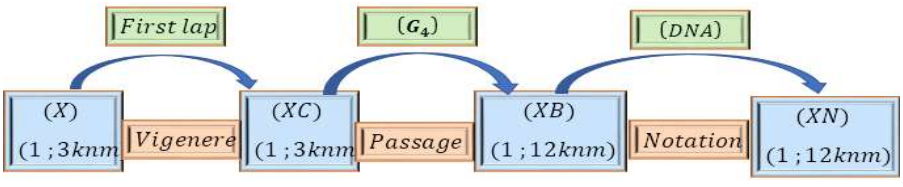


Figure 8. Transition to DNA writing.

(XN) vector storing the alphanumeric values of nucleotides select from the table (NC), while (XB) stores their numerical correspondences in (G_4).

Example: transition from vector (XB) with coefficients in (G) to vector (XN) with coefficients in (DNA) with $(r = 8)$ (**Figure 9**).

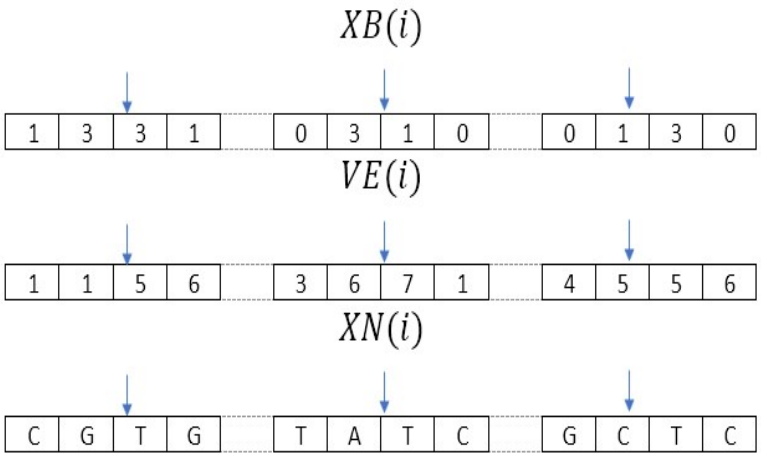


Figure 9. Transition to DNA writing.

We note that this approach is a polyalphabetic transformation, which makes the system more efficient and more robust.

2.5. Axe 5: (RNA) switching

A codon is the assembly of three nucleotides selected in (NC) table under the control of the binary decision table (BT); there are 64 codons in nature, and 20 of amino acid for the production of the protein necessary for all living things.

2.5.1. Codon storage vector

In order to create a (RNA) for the synthesis of proteins, three nucleotides are concatenated to generate a codon. In theory there are 64 different codons, which we will be stored in a (CO) vector of size $(1; 64)$ by the following algorithm:

Algorithm 12 Reference codons (CO)

1	$q = 1$
2	$For\ i = 0\ to\ 3$
3	$For\ j = 0\ to\ 3$
4	$For\ k = 0\ to\ 3$
5	$CO(q) = NC(1; i)\&NC(2; j)\&NC(3; k)$
6	$NN(q) = i \times 16 + j \times 4 + k$
7	$q = q + 1$
8	$Next\ k, j, i$

The table (CO) is called the codon reference table. The vector (CO) stores the 64 different values of the codons as a sequence of alphanumeric nucleotides while the vector (NN) stores their numerical references values.

2.5.2. Switching to codons writing

The vector (XN) is split into three vectors $(XN1)$, $(XN2)$, and $(XN3)$, each measuring $(1; 4\ \text{knm})$, which will later be combined to produce an (ARN). While the vector (*XB*) will also be subdivided into three sub-vectors $(XB1)$; $(XB2)$ and $(XB3)$ to store the numerical values of the codons obtained within the vector (NR) with coefficient in (G_{64}). These three sub-vectors will be subjected to a genetic crossing controlled by the table (CP) of size $(4\ \text{knm}; 4)$

b) (CP) Cross table development

The first column is the permutation obtained by an ascending sort on the 4 knm values of the vector $(BT(:1))$.

The second column is the permutation obtained by an ascending sort on the 4 knm values of the vector $(BT(:2))$.

The third column is the permutation obtained by an ascending sort on the 4 knm values of the vector $(CT(:1))$.

The fourth column is the permutation obtained by an ascending sort on the 4 knm values of the vector $(CT(:3))$.

The construction of such columns is described below:

- The first column indicates the index of a nucleotide of the table $(XN1)$ to take in crossing in the first position
- The second column indicates the index of the nucleotide of the table $(XN2)$ to be taken in crossing in the second position
- The third column indicates the index of the nucleotide of the table $(XN3)$ to take in crossing in the last position
- The fourth column indicates the index of the location of the codon obtained in the vector (RN)

The components of the vectors (*RN*) and (*RN*) are determined by the algorithm below:

Algorithm 13 Switching to (*RNA*) and numeric values

1	$For\ i = 1\ to\ 4\ knm$
2	$RN(CP(4; i) = XN1(CP(i; 1))\&XN2(CP(i; 2))\&XN3(CP(I; 3))$
3	$NR(CP(4; i)) = XB1(CP(i; 1)) \times 16 + XB2(CP(i; 2)) \times 4 + XB3(CP(I; 3))$
4	$Next\ i$

The vector (RN) stores the codons generated from the crossing of the three sub-vectors, while the vector (NR) stores the corresponding numerical values.

Example (**Figure 10**):

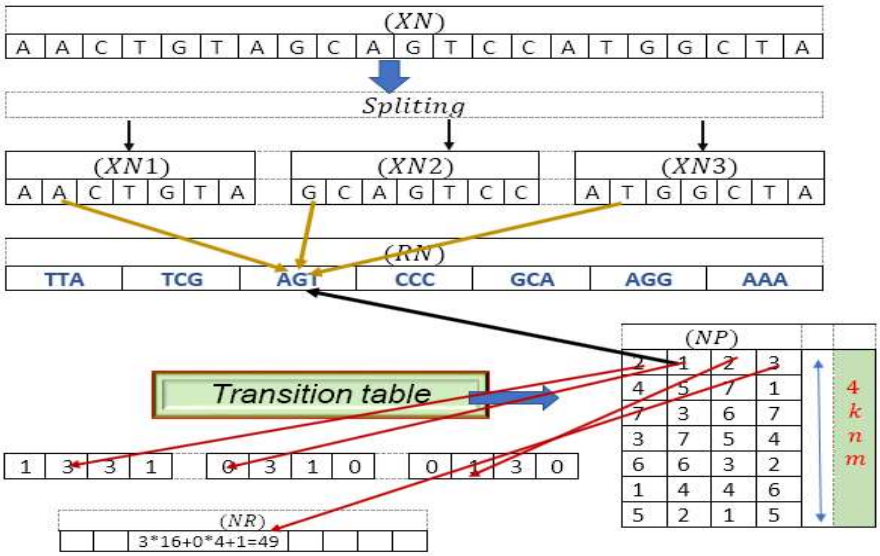


Figure 10. Transition to RNA writing.

The codon vector (RN) will be coupled to the chaotic vector (CA) for the reproduction of a new child vector (Y) inheriting the properties of the two vectors.

2.5.3. (CA) pseudo-random codon vector generating

To create the table of pseudo-random codons (CA) with dimensions $(1; 4\ \text{knm})$, we need to generate the vector (RR) of size $(1; 4\ \text{knm})$ using coefficients from (G_{64}). This process is determined by the following algorithm.

a) (RR) and (CA) vectors construction

The vector (CA) is the RNA confusion vector there are, given by the following algorithm:

Algorithm 14 Vector codons (CA)

1	$For\ i = 1\ to\ 4\ knm$
2	$If\ BT(i; 3) = 0\ then$
3	$RR(i) = mod(CT(I; 2); 64) + 1$
4	$else$
5	$RR(i) = mod(CT(i; 1); 64) + 1$
6	$End\ if$
7	$CA(i) = CO(RR(i))$
8	$Next\ i$

Example: (CA) development:

(CO)									
1	2	3	4	5		50		63	64
AAT	ACG	CCG	TTA	TAC		AAA		ATA	GGT

(RR)												
1	2	3	4	5		50		63	64		230	
3	4	63	50	1		5		2	5		50	

(CA)												
1	2	3	4	5		50		63	64		230	
CCG	TTA	ATA	AAA	AAT		TAC		ACG	GGT		AAA	

2.6. Axe 5: (RNA) storage

To apply algebraic operations on codons, we are going to develop a table (CS) of size $(64; 64)$ of two codons combination.

Codon S-Box

The table (SC) is constructed through a series of steps to guarantee the summation of two codons.

a) Matrix construction (*SC*)

The following stages provide a description of how the substitution table (SC) was developed:

- The first line is the $(Q1)$ obtained by sorting on the first 64 values of the vector $(CT(:1))$
- The rank line $(i > 1)$ is a step shift $(RR(i))$

This construction ensuring the first algebraic operation is given by the following algorithm.

Algorithm 15 (*SC*) substitution matrix development

1 $For\ i = 1\ to\ 64$

2 $SC(1, i) = Q1(i)$

3 $Next\ i$

4 $For\ i = 2\ to\ 64$

5 $For\ j = 1\ to\ 64$

6 $C(i, j) = SC\big(i - 1, (j \oplus RR(i))\big)$

7 $next\ j, i$

c) $(\boxplus)$ and $(\otimes)$ operators expression

The codons image vector (RN) obtained, will be combined with the chaotic codon vector (CA) by the following algorithm.

Algorithm 16 Sum of two codon

1 $CA(i) \boxplus RN(j) = CO\big(SC(RR(i); NR(j))\big)$

2 $CA(i) \otimes RN(j) = \big(SC(mod(3 \times RR(i) + 7; 64) + 1; NR(j))\big)$

These two algebraic operators applied to the second round are illustrated by **Figure 11.**

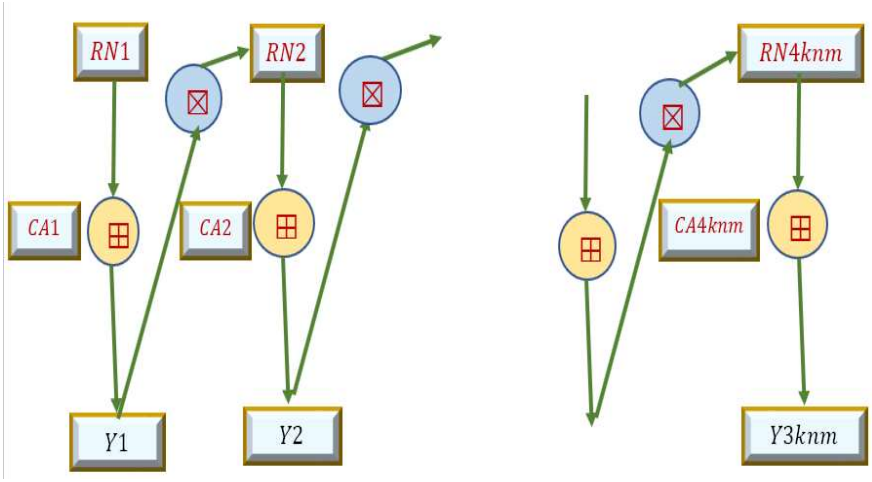


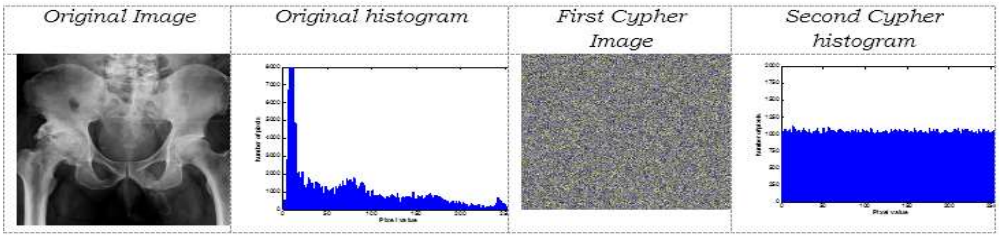
Figure 11. Second round.

This second encryption round is determined by the algorithm below:

Algorithm 17 Second round	
1	$Y(i) = CA(1) \boxplus RN(1)$
2	$For\ i = 2\ to\ 4\ knm$
3	$x = Y(i-1) \otimes RN(i)$
4	$Y(i) = CA(i) \boxplus x$
5	$Next\ i$

This last circuit has increased the supplement and robustness of our system. Some simulations prove that this technique is safe from any known attack.

d) second round simulation



The percentage of codons representing the vector (VS) is given by the following figures:

RNA distribution: **Figure 12** indicates the uniform codon distribution within the encrypted image

%	AA	AC	AT	AG	CA	CC	CT	CG	TA	TC	TT	TG	GA	GC	GT	GG	TOTAL
A	1,5	1,5	1,6	1,5	1,6	1,5	1,5	1,5	1,6	1,5	1,6	1,3	1,5	1,6	1,5	1,5	24,3
C	1,4	1,6	1,7	1,6	1,5	1,6	1,5	1,6	1,5	1,5	1,5	1,7	1,5	1,5	1,5	1,6	24,8
T	1,6	1,6	1,6	1,6	1,7	1,6	1,6	1,5	1,6	1,6	1,7	1,5	1,5	1,7	1,5	1,6	25,5
G	1,6	1,6	1,5	1,6	1,6	1,5	1,7	1,6	1,4	1,5	1,6	1,6	1,6	1,6	1,6	1,4	25
TOTAL	6,1	6,3	6,4	6,3	6,4	6,2	6,3	6,2	6,1	6,1	6,4	6,1	6,1	6,4	6,1	6,1	99,6

Figure 12. RAN %.

DNA distribution: **Figure 13** shows the uniform distribution of nucleotides with respect to their position in the codon.

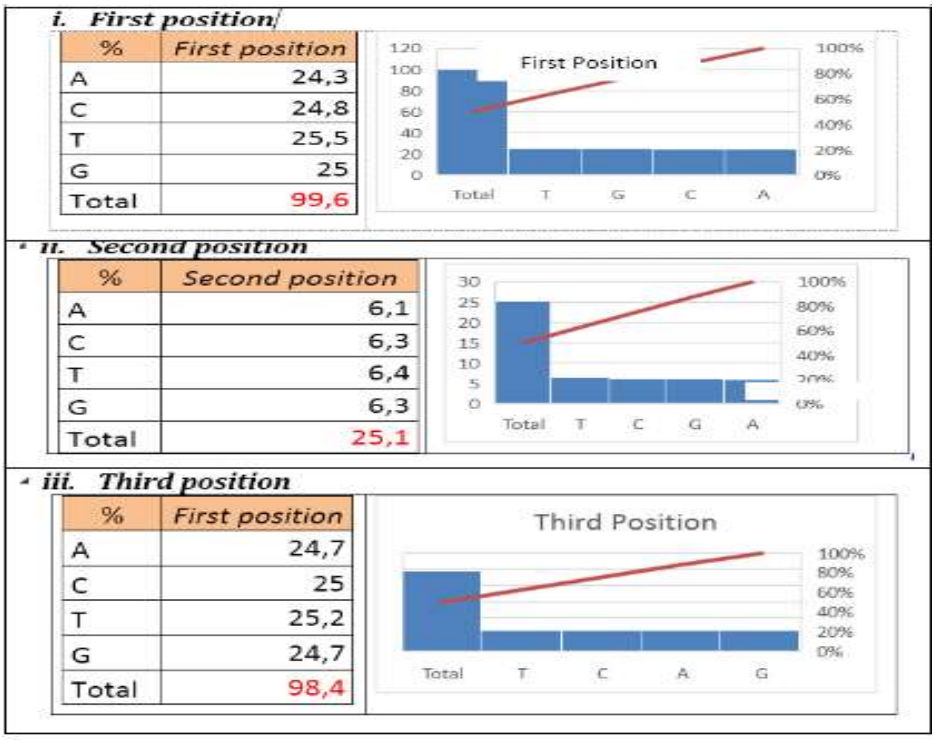


Figure 13. DNA distribution.

This uniform distribution affirms the robustness of our system.

2.7. Axe 6: Decryption stage

The proposed system is a symmetric encryption algorithm that employs extremely sensitive chaotic maps to initial conditions, managed by a large private key. This system utilizes confusion functions, as well as chaining between encrypted blocks and plaintext blocks by implementing a diffusion function. This requires us, during the decryption process, to begin with the final operations and also with the last encrypted block. The steps of the decryption process are listed below:

1) Switching the image to vector

2) RNA writes through following the encryption steps
3) Application of the reverse crossover
4) Matrix transition
5) Application of the inverse of the Vigenere's turns
6) Recovering the original image

3. Examples and simulations

A good cryptosystem is an encryption algorithm that can withstand any known attack. To analyze the new approach robustness, a large color images number of the same size were arbitrarily selected from a database and grouped into sets of five images placed side by side, as indicated by **Figure 14**:

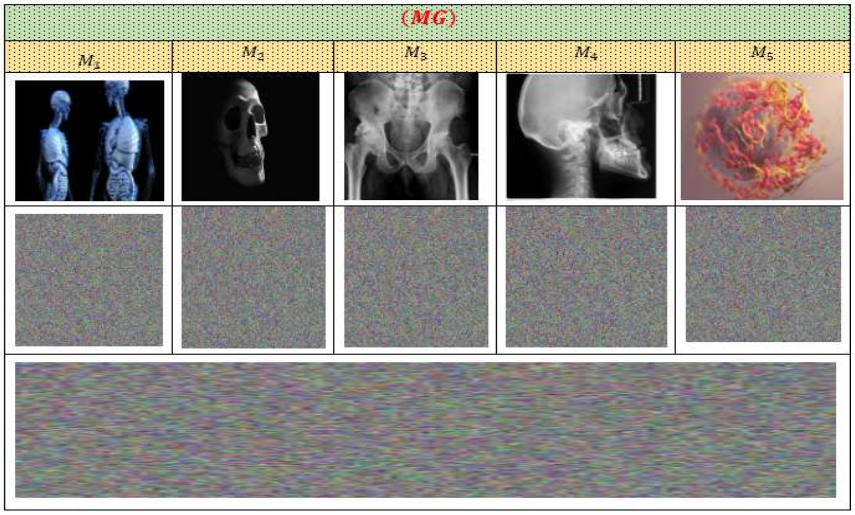


Figure 14. (*MG*) construction.

The most well-known and overcoming attacks are:

3.1. Brutal assauts

With key size knowledge, this technique involves reconstructing the private encryption key through an iterative procedure. Following Kirchhoff's principles, the system can be entirely revealed.

a) Key-space analysis

In academic literature, a key with a size of fewer than 100 bits can be reconstructed using an iterative procedure. Our algorithm employs three chaotic maps with a combined key size well exceeding 128 bits, safeguarding our method against brute-force attacks.

b) Secret key's sensitivity analysis

The chaotic maps employed in this system demonstrate a significant responsiveness to initial conditions. This assurance ensures that modifying even a solitary parameter within the private key will lead to the creation of a fresh chaotic sequence, consequently influencing other cipher parameters. This exceptional susceptibility of the key within our novel technology is visually represented in **Figure 15**.

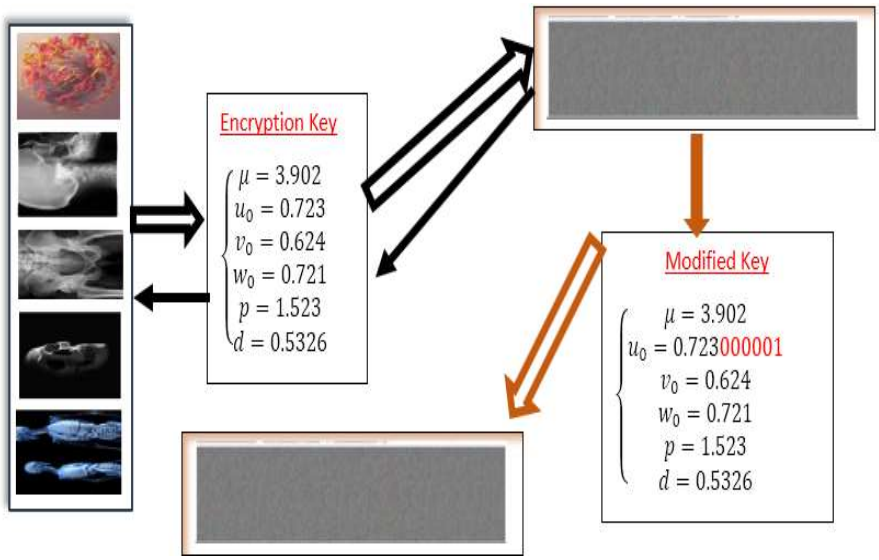


Figure 15. Encryption key sensitivity.

We note that a perturbation of the order of (10^{-9}) on a single parameter, is not able to reconstruct the original image.

3.2. Statistics attack security

To establish the resilience of our novel encryption method for color images against statistical attacks, a series of tests have been conducted on various images, highlighting the most noteworthy ones.

c) Histogram analysis

The histogram represents the gray level distribution of the encrypted image (**Figure 16**).

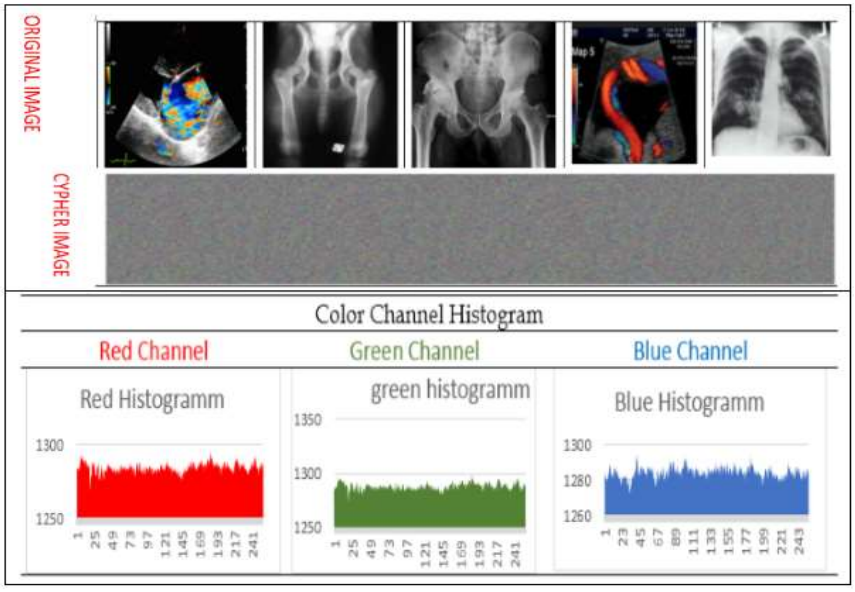


Figure 16. Histogram analysis.

We notice that the distribution of gray levels is uniform, which provides better protection against any histogram attack.

d) Entropy analysis

The equation below estimates the entropy of an image with size (n, m).

$$H(MC) = \sum_{i=1}^{t} -p(i)\, log_2(p(i)) \tag{7}$$

$p(i)$, is the probability of occurrence of level (i) in the original image. **Figure 7** shows the entropy of the images tested by our technique.

The entropy value is approaching the maximum value of 8, offering defense against potential entropy-based attacks (**Figure 17**).

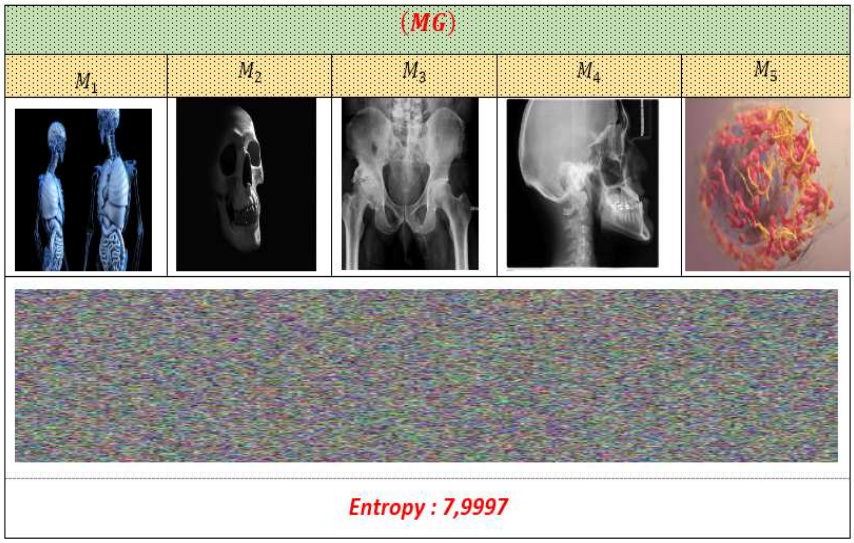


Figure 17. Entropy analysis.

3.3. Correlation analysis

The equation considers all pixels in an image of size (n, m) to determine the correlation.

$$r = \frac{cov(x, y)}{\sqrt{V(x)}\sqrt{V(y)}} \tag{8}$$

This relationship describes the degree of linear connection between neighboring pixels.

It is noted that the value of the correlation is close to zero, which guarantees the absence of any linear correlation linking the neighboring pixels and offers better protection against attacks by correlation (**Figure 18**).

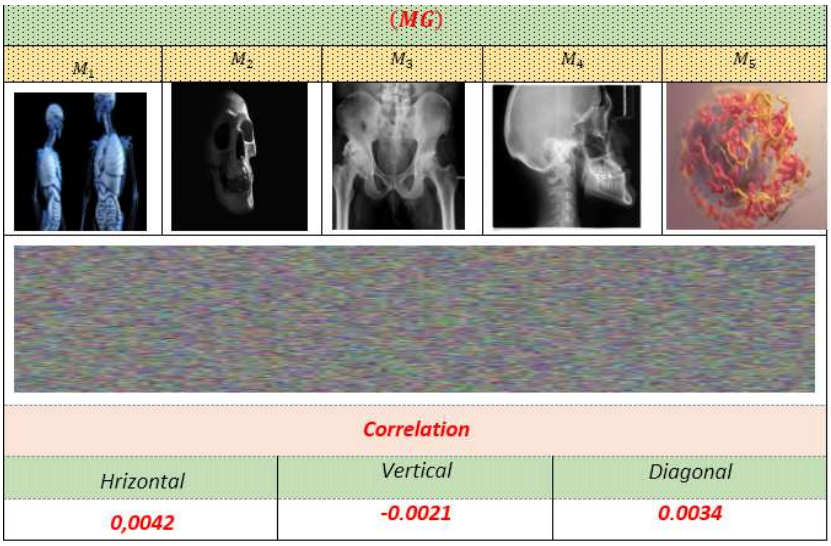


Figure 18. Correlation coefficients.

e) Differential analysis

Let be two encrypted images, whose corresponding free-to-air images differ by only one pixel, from (C_1) and (C_2), respectively. The $NPCR$ mathematical analysis of an image is given by the equation below:

$$NPCR = \left(\frac{1}{nm}\sum_{i,j=1}^{nm} D(i,j)\right) \times 100$$
$$With\ D(i,j) = \begin{cases} 1\ if\ C_1(i,j) \neq C_2(i,j) \\ 0\ if\ C_1(i,j) = C_2(i,j) \end{cases} \quad (9)$$

The $UACI$ mathematical analysis of an image is given by the below

$$UACI = \left(\frac{1}{nm}\sum_{i,j=1}^{nm} |C_1(i,j) - C_2(i,j)|\right) \times 100 \quad (10)$$

The computed differential values for the benchmark images analyzed using our novel technology conform to global standards. This results in an *NPCR* value nearing 99.95% and a *UACI* value surpassing 34.35%. These metrics establish the security of our encryption system against differential attacks, primarily attributed to the initial round setup. To further illustrate, the subsequent table offers a comparative overview with other contemporary techniques, reaffirming the resilience of our system (**Figure 19**).

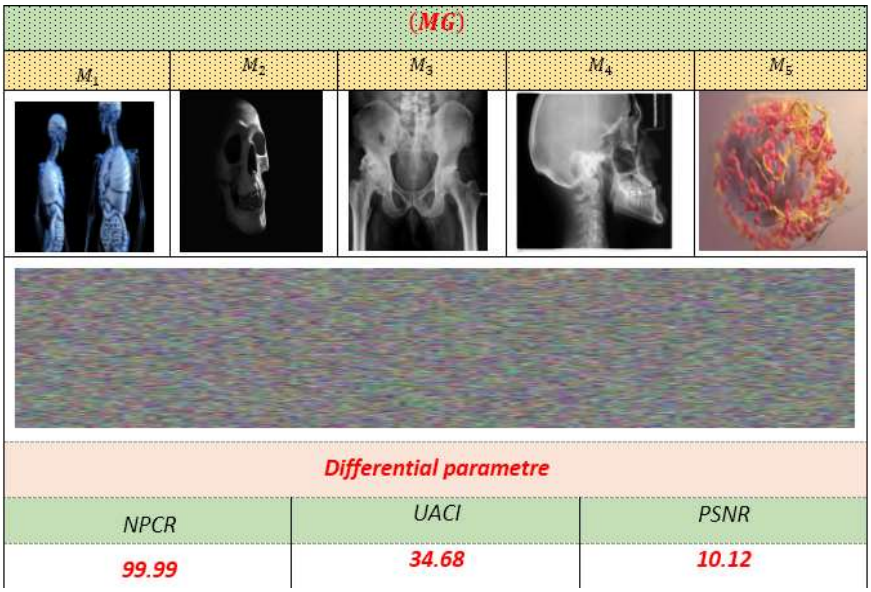


Figure 19. *NPCR* and *UACI* for different images.

3.4. Avalanche effect

The avalanche effect stands as a crucial attribute sought after in nearly all cryptographic hash functions and block coding algorithms. It guarantees that even slight modifications in the input data result in substantial and unpredictable alterations throughout the algorithm's entire structure. This characteristic determines the reach of avalanche influences within the cryptosystem. The measure of the avalanche effect can be estimated using the equation presented herein.

AE value

$$\left(AE = \frac{\sum_i bit\ change}{\sum_i bit\ total}\right) \times 100 \tag{11}$$

The avalanche effect values from the images tested by our technology provide strong protection against differential attacks (**Figure 20**).

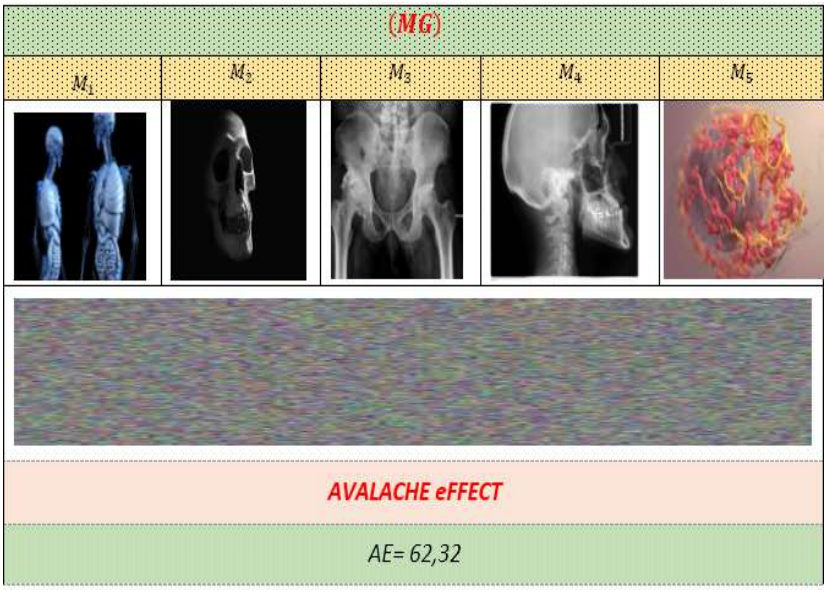


Figure 20. Avalanche effect.

3.5. Advantages of this process

This method encapsulates several advantages of which we mention

- Encryption keys from chaotic maps are extremely sensitive to initial conditions, making it difficult to recover the real key used.
- The chaotic transition to binary notation is a promising new approach.
- The S-Box structure and its use under the control of a chaotic decision vector increases the attack complexity of our technique.
- The transition to the codons increases the temporal complexity of attack and causes ambiguity on the values of the codons.
- Noncommutative algebraic operations give robustness to the system
- Our technique is applicable to any image of arbitrary size and format.

3.6. Approach limitation

The boundaries of our approach are heavily contingent upon the limitations of the selected chaotic maps and the design of the S-Boxes. In specific scenarios, the conversion to binary form can be enhanced, while transitioning into *DNA* or *RNA* may involve intricate genetic crossovers to bolster the resilience of the suggested system.

4. Conclusion

This article introduces a swift and dependable algorithm for the encryption of multiple images, employing dynamic and chaotic systems. To assess this approach, groupings of five images were merged into composite forms, considering the limitations imposed by the utilized materials. By incorporating enhancements through a novel and refined Vigenère technique at the pixel level, alongside the implementation of genetic operations tailored for image encryption at the RNA level, under the guidance of a diverse set of pseudo-random tables, the system gained notable resilience and maintained a consistent level of intricacy. The outcomes gleaned from extensive testing involving a diverse array of images showcased notable satisfaction and potential, instilling a sense of assurance in the plausible utilization of this technique for securing video content.

Author contributions: All authors carried out the experiment. All authors helped supervise the project and conceived the original idea. AJ and AB: reviewed the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: No funding was received to assist with the preparation of this manuscript.

Conflict of interest: All authors of this article confirming the absence of any conflict between them, and there are no private or public organizations or laboratories to fund this research, thus avoiding any expected conflicts. This document does not contain any research or experiments conducted on animals or humans.

References

1. JarJar A. Vigenere and genetic cross-over acting at the restricted ASCII code level for color image encryption. Medical & Biological Engineering & Computing. 2022, 60(7): 2077-2093. doi: 10.1007/s11517-022-02566-4

2. Qobbi Y, Jarjar A, Essaid M, et al. New Image Encryption Scheme Based on Dynamic Substitution and Hill Cipher. WITS 2020. Published online July 22, 2021: 797-808. doi: 10.1007/978-981-33-6893-4_72
3. Jarjar A. Improvement of hill's classical method in image cryptography. International Journal of Statistics and Applied Mathematics. 2017, 2(3 Part A).
4. Jarjar M, Najah S, Zenkouar K, et al. Further improvement of the HILL method applied in image encryption. In: Proceedings of 2020 1st international conference on innovative research in applied science, engineering and technology (IRASET); 16–19 April 2020; Meknes, Morocco. pp. 1–6. doi: 10.1109/IRASET48871.2020.9092046
5. Hraoui S, Gmira F, Jarar AO, et al. Benchmarking AES and chaos based logistic map for image encryption. 2013 ACS International Conference on Computer Systems and Applications (AICCSA). Published online May 2013. doi: 10.1109/aiccsa.2013.6616441
6. Kiran, Parameshachari BD, Panduranga HT. Medical Image Encryption Using SCAN Technique and Chaotic Tent Map System. Recent Advances in Artificial Intelligence and Data Engineering. Published online November 1, 2021: 181-193. doi: 10.1007/978-981-16-3342-3_15
7. Liu Z, Zhu D, Zhou C, et al. Chaotic image encryption method based on Zigzag scrambling and DNA coding. 2022 4th International Conference on Frontiers Technology of Information and Computer (ICFTIC). Published online December 2, 2022. doi: 10.1109/icftic57696.2022.10075195
8. Hoang TM. A novel design of multiple image encryption using perturbed chaotic map. Multimedia Tools and Applications. 2022, 81(18): 26535-26589. doi: 10.1007/s11042-022-12139-0
9. Xu L, Zhang J. A Novel four - Wing chaotic system with multiple attractors based on hyperbolic sine: Application to image encryption*. Integration. 2022, 87: 313-331. doi: 10.1016/j.vlsi.2022.07.012
10. Abdallah AA, Farhan AK. A New Image Encryption Algorithm Based on Multi Chaotic System. Iraqi Journal of Science. Published online January 30, 2022: 324-337. doi: 10.24996/ijs.2022.63.1.31
11. Suzaki T, Minematsu K. Improving the Generalized Feistel. Lecture Notes in Computer Science. Published online 2010: 19-39. doi: 10.1007/978-3-642-13858-4_2
12. JarJar A. Improvement of Feistel method and the new encryption scheme. Optik. 2018, 157: 1319-1324. doi: 10.1016/j.ijleo.2017.12.065
13. Abid A, Qobbi Y, Benazzi A, et al. Two Enhanced Feistel Steps for Medical Image Encryption. 2022 IEEE 3rd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS). Published online December 1, 2022. doi: 10.1109/icecocs55148.2022.9982938
14. Ge R, Yang G, Wu J, et al. A Novel Chaos-Based Symmetric Image Encryption Using Bit-Pair Level Process. IEEE Access. 2019, 7: 99470-99480. doi: 10.1109/access.2019.2927415
15. Shah A. Enhancing Security of Vignere Cipher using Modified RC4. International Journal of Computer Applications. 2016, 136(5): 38-41. doi: 10.5120/ijca2016908428
16. Zhang L, Zhang X. Multiple-image encryption algorithm based on bit planes and chaos. Multimedia Tools and Applications. 2020, 79(29-30): 20753-20771. doi: 10.1007/s11042-020-08835-4
17. Zhang Y, He Y, Zhang J, et al. Multiple Digital Image Encryption Algorithm Based on Chaos Algorithm. Mobile Networks and Applications. 2022, 27(4): 1349-1358. doi: 10.1007/s11036-022-01923-9
18. Zhang X, Wang X. Multiple-image encryption algorithm based on DNA encoding and chaotic system. Multimedia Tools and Applications. 2018, 78(6): 7841-7869. doi: 10.1007/s11042-018-6496-1
19. Abd Ali SM, Hasan HF. Novel encryption algorithm for securing sensitive information based on feistel cipher. Test Engeenering Managenment. 2019. 19(80): 10-16.
20. Alexan W, Alexan N, Gabr M. Multiple-Layer Image Encryption Utilizing Fractional-Order Chen Hyperchaotic Map and Cryptographically Secure PRNGs. Fractal and Fractional. 2023, 7(4): 287. doi: 10.3390/fractalfract7040287
21. Kumari M, Pawar V, Kumar P. A Novel Image Encryption Scheme with Huffman Encoding and Steganography Technique. International Journal of Network Security & Its Applications. 2019, 11(4): 49-73. doi: 10.5121/ijnsa.2019.11404
22. Kumar Patro KA, Acharya B. An efficient colour image encryption scheme based on 1-D chaotic maps. Journal of Information Security and Applications. 2019, 46: 23-41. doi: 10.1016/j.jisa.2019.02.006
23. Mansoor S, Sarosh P, Parah SA, et al. Adaptive Color Image Encryption Scheme Based on Multiple Distinct Chaotic Maps and DNA Computing. Mathematics. 2022, 10(12): 2004. doi: 10.3390/math10122004

24. Younus ZS, Hussain M K. Image steganography using exploiting modification direction for compressed encrypted data. *Journal of King Saud University-Computer and Information Sciences* 2022; 34(6): 2951–2963. doi: 10.1016/j.jksuci.2019.04.008
25. Yuan X, Zhang L, Chen J, et al. Multiple-image encryption scheme based on ghost imaging of Hadamard matrix and spatial multiplexing. Applied Physics B. 2019, 125(9). doi: 10.1007/s00340-019-7286-9
26. Liansheng S, Xiao Z, Chongtian H, et al. Silhouette-free interference-based multiple-image encryption using cascaded fractional Fourier transforms. Optics and Lasers in Engineering. 2019, 113: 29-37. doi: 10.1016/j.optlaseng.2018.10.002
27. Zhang L, Yuan X, Wang K, et al. Multiple-Image Encryption Mechanism Based on Ghost Imaging and Public Key Cryptography. IEEE Photonics Journal. 2019, 11(4): 1-14. doi: 10.1109/jphot.2019.2923705
28. Gao X, Mou J, Xiong L, et al. A fast and efficient multiple images encryption based on single-channel encryption and chaotic system. Nonlinear Dynamics. 2022, 108(1): 613-636. doi: 10.1007/s11071-021-07192-7