Improved vigenere using affine functions surrounded by two genetic crossovers for image encryption

Hamid El Bourakkadi¹, Abdelhakim Chemlal¹, Hassan Tabti², Mourad Kattass¹, Abdellatif Jarjar¹, Abdelhamid Benazzi¹
¹MATSI Laboratory, Mohammed First University, Oujda, Morocco
²LSIA Laboratory, Sidi Mohamed Ben Abdellah University, Fez, Morocco

ABSTRACT

This paper presents an improved method for encrypting color images, surpassing the effectiveness of genetic crossover and substitution operations. The technique incorporates dynamic random functions to enhance the integrity of the resulting vector, increasing temporal complexity to thwart potential attacks. The improvement involves integrating genetic crossover and utilizing two extensive pseudorandom replacement tables derived from established chaotic maps in cryptography. Following the controlled vectorization of the original image, our approach initiates with a first genetic crossover inspired by deoxyribonucleic acid (DNA) behavior at the pixel level. This genetic crossover is succeeded by a confusion-diffusion lap, reinforcing the connection between encrypted pixels and their neighboring counterparts. The confusion-diffusion process employs dynamic pseudorandom affine functions at the pixel level. Then a second genetic crossover operator is applied. Simulations conducted on a diverse set of images with varying sizes and formats showcase the robustness of our method against statistical, brute-force, and differential attacks.

Keywords:
Chaotic map
Genetic crossover
Hybrid circuit
Pseudorandom functions
S-Box

This is an open access article under the CC BY-SA license.

Corresponding Author:
Hamid El Bourakkadi
MATSI Laboratory, Mohammed First University
Oujda, Morocco
Email: hamid.elbourakkadi.d23@ump.ac.ma

1. INTRODUCTION

Ensuring the security of information during network transmission has become a widely explored research area. In this context, encryption technology plays a crucial role, with symmetric and asymmetric encryption algorithms applied in cryptography [1]. Symmetric encryption, known for its efficiency, heightened security, and rapid encryption speed with a large key, relies on preserving the ciphering key for its security. The algorithm involves minimal computation and ensures a high protection level and ciphering speed when utilizing a lengthy key. Data security transmission is contingent upon safeguarding the encryption key. On the other hand, while offering high security, asymmetric ciphering comes with significant encryption and decryption time, making it suitable for limited data encryption such as passwords. The security of data transmission in this case relies on both the key and the algorithm. With the principle of Kirchhoff, the key system's protection is closely related to the ciphering key rather than the algorithm. In addressing these challenges, numerous image ciphering methods utilize the principles of symmetry theory, and this paper similarly embraces a strategy centered on the same type of algorithms.

Despite researchers' efforts to develop more secure ciphering methods, numerous image encryption methods have been successfully compromised [2], [3]. In pursuit of heightened security, many scholars have turned to multi-round encryption methods [4], [5], though this approach involves a significant amount of
time. Some authors have suggested encryption methods focused solely on the relevant characteristics in an image [6]-[8]. Hussain et al. [7] introduced a ciphering architecture for medical images centered on TetraVex game theory method, demonstrating flexibility and reliability in protecting medical images against any attack. Çelik and Doğan [9], suggested an encryption image architecture using data hiding and logistic maps, exhibiting satisfactory security performance in experimental results. Drawing inspiration from these studies, we propose utilizing existing technology to recognize the facial contour area when encrypting human images, enabling the individual encryption of these regions. After encrypting the facial part, the entire image undergoes another layer of encryption. Compared to traditional one-round encryption, this approach demonstrates superior encryption effectiveness. Even in the event of an algorithmic attack, the private aspects, such as the face, remain indistinguishable and unrecoverable. In contrast to multi-round encryption schemes, these techniques involve shorter periods and higher speeds to encrypt and decrypt data.

As chaos theory continues to evolve, researchers have systematically explored the attributes of pseudorandomness, and sensitivity to initial values [10], [11]. Additionally, various chaotic image encryption algorithms have emerged, integrating principles from diverse disciplines such as genetic algorithms rules such as quantum maps and perceptron-like networks [12], [13]. Chatterjee et al. [14], Chatterjee et al designed an encryption architecture that involves replacement-diffusion operation using standard and logistic maps, addressing the issue of dynamic degradation. Li et al. [15] demonstrate that the use of chaotic sequences as keys can compromise the algorithm’s security. Efficiently, chaotic methods have various variations, with some, such as the PWLCM, logistic, Henon, and skew tent maps [16]-[19], being renown image encryption. Focusing on the logistic and PWLCM maps, they offer several advantages, including enhanced sensitivity and randomness, resulting in more secure, chaotic, and unique sequences.

The difficulty arises from the dependence of such algorithms on independent block encryption without the intervention of plaintext blocks, making them vulnerable to statistical and frequency attacks. Additionally, without the incorporation of diffusion and chaining functions between the encrypted and plaintext blocks, these techniques remain susceptible to differential attacks.

Our contribution to addressing anomalies raised in conventional research has driven us to develop a new image encryption system using two large substitution matrices generated from two chaotic maps highly sensitive to initial conditions, incorporating pseudo-random affine functions for confusion and diffusion restoration. This process is surrounded by two deep genetic crossover operations specifically tailored for image encryption to increase the complexity of attacks against our innovative system. Simulations results and comparisons with other algorithms conducted on randomly selected images from SIPI database reveal a high level of robustness, making our new technique impervious to differential and statistical attacks.

This research paper is divided into various sections, including a section describing the proposed method, explaining the basis of chaotic sequences, the classical Vigenere and affine techniques as well as detailing keys generation axis, revealing the nuances of the encryption and decryption process; a section devoted to results and discussions, presenting research results and discussion; and a section summarizing the findings and proposing research directions.

2. PROPOSED METHOD

Our approach enhances the traditional Vigenere method by incorporating extensive substitution tables alongside novel pseudorandom functions. Furthermore, reversible functions have been seamlessly integrated into the encryption process. However, this technique is divided into four main axes, the used chaotic sequences and functions, different keys generation steps, and the encryption and decryption axes.

2.1. Axis 1: used chaotic sequences and functions

The selected chaotic maps to construct different subkeys for encryption and decryption processes are highly sensitive to initial conditions and easy to implement in a cryptographic system. Additionally, our system incorporates random functions into classical Vigenere method. However, this subsection is divided into four subtites.

2.1.1. PWLCM map

The first chaotic sequence will be generated by the PLWCM map [16] as defined by (1).

\[
h_{n+1} = f(h_n) = \begin{cases} 
    h_0 \in ]0;1[, k \in ]0;5[, 4[ & \\
    k^{-1} h_n \; \text{if} \; 0 < h_n < k & \\
    (0.5 - k)^{-1}(h_n - k) \; \text{if} \; k < h_n < 0.5 & \\
    f(1 - h_n) \; \text{otherwise} & 
\end{cases}
\]  

(1)
The parameters \((h_0)\) and \((k)\) represent the initial state and its control parameter, respectively.

2.1.2. Logistic map

The logistic map [17] is a widely used mathematical function that models population growth over time within a limited space. It is a common tool in chaos theory and cryptography. The expression of such map is depicted in (2).

\[
\begin{cases}
    l_0 \in [0,5]; 1[ et \, \delta \in [3,75; 4] \\
    l_{n+1} = \delta \cdot l_n(1 - l_n)
\end{cases}
\]

(2)

2.1.3. Classical Vigenere method

The classical Vigenere cipher system is based on a matrix \((M)\) of fixed dimensions \((26,26)\) reserved for text encryption only. The encryption and decryption algorithms associated with the classical Vigenere method are given in Algorithm 1.

Algorithm 1. Classical Vigenere encryption and decryption algorithms

//Encryption
for \(i = 1\) to \(n\)  
\(C_k_i = M(Ck_i, Ke_i) = Pk_i + Ke_i \mod 26\)  
end for

//Decryption
for \(i = 1\) to \(n\)  
\(P_k_i = M^{-1}(P_k_i, Ke_i) = C_k_i - Ke_i \mod 26\)  
end for

Where \((P_k)\) is the plain message, \((C_k)\) is the cipher message, \((Ke)\) is the encryption key, \((M)\) is the Vigenere matrix, and \((n)\) be the length of the plain message.

2.1.4. Affine functions in \((Z/nZ)\)

Let \((f)\) be an affine function defined in the ring \((Z/nZ)\) by (3).

\[
\begin{cases}
    f: Z/nZ \rightarrow Z/nZ \\
    x \mapsto \text{mod}(ax + b; n)
\end{cases}
\]

\((a, b) \in \mathbb{Z}/n\mathbb{Z}\)

(3)

The function \((f)\) is a bijective function in \((Z/nZ)\) if and only if \((a)\) is invertible and \((b)\) is any.

Indeed, we have \(y = \text{mod}(ax + b; n)\)

Then, \(ax = \text{mod}(y - b; n)\) and \(x = \text{mod}(a^{-1}, (y - b); n)\) Where \((a^{-1})\) is the inverse of \((a)\) in ring \((Z/nZ)\).

Or, we know that \((a)\) is invertible in \((Z/nZ)\) if and only if \(a \land n = 1\).

Particular case: \(n = 2^k, k \in \mathbb{N}\)

In a particular case, \((a)\) is invertible in ring \((Z/2^kZ)\) if and only if \((a)\) is odd.

2.2. Axis 2: keys generation

This new technique uses the two most widely deployed chaotic maps in the field of cryptography [18], [19] by integrating large S-boxes incorporating strong pseudorandom affine functions for the confusion-diffusion process. The confusion-diffusion process is encapsulated by two genetic crossovers. This technique is structured around the subsections described below.

2.2.1. Pseudorandom vectors generation

Our system is a cryptographic architecture that necessitates the generation of subkeys. These subkeys are generated from two chaotic sequences that are extremely sensitive to initial conditions. The below subsections detail such process.

2.2.2. Used chaotic sequences

Two chaotic sequences \((h)\) and \((l)\) were generated based on PWLCM and Skew tent chaotic maps described in sub section (3.1). These sequences, used in our approach, are extremely sensitive to the initial conditions and easy to implement in any cryptosystem.

2.2.3. Sub keys construction

Seven pseudorandom vectors \((Vc1, Vc2, Vc3, Vr, Ve, Va, Vb)\) with coefficients in the ring \((Z/256Z)\) are generated by Algorithm 2.

Algorithm 2. Pseudorandom vectors generation

\[
\text{Vel}(i) = [E((\text{max}(h(i);i)) \cdot 10^{31}) \mod 253] + 2 \quad \text{// First confusion vector}
\]

\[
Vc2(i) = [E((\text{max}(h(i) + 2 \times i(i))/3) \cdot 10^{12}) \mod 254] + 1 \quad \text{// Second confusion vector}
\]

Improved vigenere using affine functions surrounded by two genetic ... (Hamid El Bourakkadi)
2.2.7. \textit{Algorithm} Substitution tables generation, $i \in \{1, 2, 3\}$

//Binary vectors construction

\begin{verbatim}
for i ← 1 to 3nm
if h(i) > l(i) then
    Ba1(i) ← 0
    Ba3(i) ← 1
else:
    Ba1(i) ← 1
    Ba3(i) ← 0
end if
if h(i) > 0.5 then:
    Ba2(i) ← 0
end for
\end{verbatim}

2.2.4. Generation of genetic crossover table (GC)

This operation is a genetic crossover adapted to the encryption of color images that will be accompanied by a table (GC) of size $(3nm, 2)$. The construction of this table is given by the steps below:

- The 1st column is the arrangement ($P$) obtained by a decreasing sort on the first $(3nm)$ values of the sequence ($h$).
- The second column is the permutation ($P'$) obtained by an increasing sort on the first $(3nm)$ values of the sequence ($l$).

2.2.5. Substitution tables generation

Our algorithm necessitates the creation of two novel tables, denoted as ($M1$) and ($M2$). These tables are used in confusion/diffusion process as substitution matrices. Each table is sized at $(256; 256)$ and operates with coefficients within the ring ($Z/256Z$).

2.2.6. ($M1$) S-BOX generation

The main mission of this section is to construct the new Vigenere substitution matrix, called ($M1$), with a size of $(256; 256)$, following the instructions provided below.

- The first row of the table ($M1$) is the permutation ($Pr1$) of the first 256 values of the vector ($Vc1$), obtained by sorting them in decreasing order.
- For ranks higher than 1, the rank line is a rank shift $Vc2(i)$ or $Vc3(i)$, depending on the control vector $Ba1(i)$. This table was generated by Algorithm 4.

\begin{verbatim}
Algorithm 4. ($M1$) Substitution box generation
for i ← 1 to 256 / First line
    if Ba1(i) = 0 then
        M1(i, j) ← Pr1(j)
    else
        M1(i, j) ← M1(i-1, mod(j + Vc2(i), 256))
end for
for j ← 2 to 256 / Next lines
    for i ← 1 to 256
        end if: end for: end for
\end{verbatim}

2.2.7. ($M2$) S-BOX generation

The construction of the new substitution matrix ($M2$) of size $(256; 256)$ is described by:

- The 1st line is the rearrangement ($Pr1$) obtained by a broad ascending order on the first 256 values of the vector ($Vc3$).
- The 2nd line is the rearrangement ($Pr2$) obtained by a broad ascending order on the first 256 values of the vector ($Vc2$).
- The 3rd line is the rearrangement ($Pr3$) obtained by a broad ascending order on the first 256 values of the vector ($Vc1$).
- The $i$th line ($i > 3$) is the composition of the functions of row ($i - 2$) and ($i - 3$) or ($i - 3$) and ($i - 1$), depending on the value of the control vector $Ba2(i)$.

These steps are illustrated in Algorithm 5.

\begin{verbatim}
Algorithm 5. ($M2$) Substitution box generation
for i ← 1 to 256 /3 first lines
    for i ← 4 to 256 /Next lines
\end{verbatim}
2.3. Axis 3: encryption phase

The innovative encryption process improves upon the traditional Vigenere technique by incorporating enhanced methods. It utilizes dynamic functions for confusion and diffusion, adding a unique twist to the Vigenere method. Additionally, two genetic mutations specifically designed for encrypting color images complement this approach. The process is organized into the following stages:

2.3.1. Original image vectorization

This phase involves uploading the original image of dimensions \((n,m)\) and then extracting the (RGB) channel vectors \((R), (G), \) and \((B)\), which are concatenated under the control of the binary vector \((Ba1)\) into a single vector \((XD)\) of dimensions \((1,3nm)\). The mathematical formulation of this phase is described in Algorithm 6.

Algorithm 6. Original image vectorization algorithm

\[
\text{for } j \leftarrow 1 \text{ to } nm \text{ \hspace{1cm} else} \\
\text{if } Ba1(j) = 0 \text{ \hspace{1cm} XD}(3j - 2) \leftarrow R(j) \oplus Vc1(j) \\
XD(3j - 2) \leftarrow R(j) \oplus Vc1(j) \text{ \hspace{1cm} XD}(3j - 1) \leftarrow G(j) \oplus Vc1(j) \\
XD(3j - 1) \leftarrow G(j) \oplus Vc1(j) \text{ \hspace{1cm} XD}(3j) \leftarrow B(j) \oplus Vc1(j) \\
XD(3j) \leftarrow B(j) \oplus Vc1(j) \text{ \hspace{1cm} end if \hspace{1cm} end for}
\]

2.3.2. First genetic crossover operation

After image preparation above, the genetic crossover will be applied to the integrity of the output vector \((XD)\). This operation will be subject to the control of the table \((GC1)\) and given by (4).

\[
X(i) = XD(GC1(i, 1)) \oplus Vc1(GC1(i, 2)), i \in [1, 3nm]
\]

The first column of the table \((GC1)\) indicates the rank of the pixel to be modified, while its second column indicates the rank of the chaotic value chosen for the confusion. This algorithm is illustrated by the following Figure 1. The first column of the table \((GC1)\) indicates the rank of the pixel to be modified, while its second column indicates the rank of the chaotic value chosen for the confusion.

![Figure 1. First genetic crossover operation](image)

2.3.3. Confusion and diffusion process

a. Expression of the pseudorandom functions

Let \((f_i)\) be the family of affine functions acting on the pixels. These functions are defined by (5).

\[
\begin{align*}
  f_i: Z/256Z & \rightarrow Z/256Z \\
  x & \mapsto \{ \mod(Va(i) \ast X(i) + Ve(i); 256) \text{ si } Ba2(i) = 0 \\
  & \quad \mod(Vb(i) \ast X(i) + Vr(i); 256) \text{ si } Ba2(i) = 1 \}
\end{align*}
\]

Since the elements \((Va(i))\) and \((Vb(i))\) are invertible in ring \((Z/256Z)\), the functions \((f_i)\) are reversible for all \(i \in [1; 3nm]\).
b. Confusion and diffusion function expression

To enhance the security of our system against potential differential attacks, we're incorporating confusion and diffusion functions that utilize pseudo-random vectors and reversible dynamic functions. This chaining technique involving replacement tables \((M1)\) and \((M2)\) amplifies the avalanche effect. The execution of diffusion functions is outlined in Algorithm 7.

Algorithm 7. \((Fv)\) confusion and diffusion function expression

\[
\begin{align*}
Z(i) &= Fv(X(i)) \\
\text{if } Ba2(i) = 0 &\text{ then: } Z(i) \leftarrow M1(Vc1(i), M2(Vc2(i)); mod(Va(i) + X(i) + Ve(i); 256)) \\
\text{else } &\text{ : } Z(i) \leftarrow M2(Vc3(i), M1(Vc1(i)); mod(Vb(i) + X(i) + Vr(i); 256)) \\
\text{end if}
\end{align*}
\]

This algorithm is illustrated in Figure 2 below.

c. Initialization value calculation

This improved Vigenere lap starts by calculating the initialization value \((In)\), which is closely linked to the plain image and is intended to change the value of the starting pixel and launch the encryption phase. This value is calculated by Algorithm 8 below.

Algorithm 8. Initialization value calculation

\[
\begin{align*}
In &= 0 & \text{else} \\
\text{for } i &= 2 \text{ to } 3nm & \text{In} = ln \oplus X(i) \oplus Vc3(i) \\
\text{if } Ba3(i) &= 0 & \text{end if} \\
In &= ln \oplus X(i) \oplus Vc2(i) & \text{end for}
\end{align*}
\]

This algorithm is illustrated in Figure 2 below.

d. Confusion and diffusion circuit

To overcome any differential attack, we first perform a diffusion round using the chaotic confusion vectors and a chaining between the ciphered pixels and the following plain pixels using the bijective affine functions. The diffusion process is illustrated by Algorithm 9.

Algorithm 9. Confusion and diffusion circuit

\[
\begin{align*}
/\text{First pixel encryption} & \quad \text{if } Ba3(i) = 0 & Z(i) = Fv(\alpha @ Vc2(i)) \\
Z(1) &= Fv(X(1) @ ln @ Vc1(1)) & \text{else } Z(i) = Fv(\alpha @ Vc3(i)) \\
/\text{Next pixels encryption} & \quad \text{end if} \\
\text{for } i &= 2 \text{ to } 3nm & \text{end for} \\
\beta &= f_1(X(i)) @ Z(i - 1)
\end{align*}
\]

This algorithm can be interpreted in Figure 2.

2.3.4. Second genetic crossover operation

After image preparation above, the genetic crossover will be applied to the integrity of the output vector \((Z)\). This operation will be subject to the control of the table \((GC2)\) and given by (6).

\[
T(i) = Z(GC2(i, 1)) \oplus Vc2(GC2(i, 2)), i \in [1, 3nm]
\] (6)

The first column of the table \((GC2)\) indicates the rank of the pixel to be modified, while its second column indicates the rank of the chaotic value chosen for the confusion. This algorithm is illustrated by the following Figure 3.

Figure 2. New Hybrid circuit incorporating dynamic pseudorandom affine functions
Improved vigenere using affine functions surrounded by two genetic ... (Hamid El Bourakkadi)

The first column of the table ($GC2$) indicates the rank of the pixel to be modified, while its second column indicates the rank of the chaotic value chosen for the confusion. The resulting vector ($T$) represents the cipher image.

2.4. Axis 4: phase of decryption

The suggested encryption system is symmetric and employs two diffusion functions, necessitating that the decryption process commences by applying the inverse functions to the last operation. The encrypted image transforms a vector ($Z$) with dimensions $(1; 3nm)$, upon which the subsequent steps are carried out:
- Application of the reverse for the second genetic crossover operation.
- Inverse of the pseudorandom functions and inverse of the confusion and diffusion circuit;
- Application of the reverse for the first crossover operation.

3. RESULTS AND DISCUSSION

All the simulations were implemented in Python on the Windows 10 operating system with a hardware environment consisting of an i7 processor laptop, a 1 TB hard drive, and 32 GB of RAM. The main test image “Lena”, as well as its encrypted and decrypted images, as well as all the plain images we used, were taken from SIPI database [20]. These image samples. The keys and other experimental parameters are generated from the chaotic maps described above. Prior to initiating the decryption process, the secret key needs to be securely transmitted to the recipient through a protected channel.

3.1. Statistical attacks

Examining encrypted data to reveal details about the encryption key or the plaintext images constitutes a statistical attack. These assaults encompass histogram attacks, entropy attacks, and correlation attacks, among others. The subsequent subsections delineate the impacts of these assaults on our system.

3.1.1. Analysis of possible keys space

Our algorithm uses two chaotic maps generated by four real parameters represented by 32 bits each. So, the size of our key is equal to 128 bits and encompasses 120 bits. This ensures that our system is resistant to any brute-force attack.

3.1.2. Key strength analysis

Our system uses two of the most widely utilized chaotic maps in the field of cryptography. Because they are highly sensitive to initial conditions, this guarantees significant responsiveness to our encryption key. This can be seen in the diagram in Figure 4. So, any modification of the encryption key will lead to two different ciphered images being obtained during the encryption stage. In addition, two decrypted images across the decryption stage will have completely different shapes. This confirms that our cryptosystem is safe from any brute-force attack.

Figure 4. Key strength analysis
3.1.3. Analysis of histograms

Histograms of images showing the distribution of pixel values. Figures 5 (a) to (d) illustrate the RGB histograms of all the test images used, while Figures 6 (a) to (d) show the RGB histograms of the corresponding encrypted images by our method. It is observable that the histograms of encrypted images generated by our system are almost uniformly distributed. This ensures better protection against any statistical attack.

Figure 5. RGB histograms of the original images (a) histogram of Lena plain image, (b) histogram of baboon plain image, (c) histogram of peper plain image and (d) histogram of house plain image
Improved vigenere using affine functions surrounded by two genetic...

(a) histogram of lena encrypted image, (b) histogram of baboon encrypted image, (c) histogram of peper encrypted image and (d) histogram of house encrypted image.

3.1.4. Analysis of entropy

The entropy of an image of size \((n, m)\) serves as a metric for assessing the security of that image encryption is given by (7).

\[
S(MC) = -\frac{1}{3nm} \sum_{i=1}^{3nm} p(i) \log_2(p(i))
\]  

(7)

Where \(p(i)\) represents the probability of occurrence of level \((i)\) in the plain image.

Table 1 illustrates a comparison between our approach and some similar approaches. This comparison proves that our approach is significantly better than the other compared algorithms [21]-[23]. This confirms that our cryptosystem can withstand entropy image attack.

Table 1. Comparison of encrypted image entropy with other methods: Lena (L), peppers (Pe), House (H)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Images</th>
<th>Red</th>
<th>Green</th>
<th>Blue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>Lena</td>
<td>7.9975</td>
<td>7.9975</td>
<td>7.9974</td>
</tr>
<tr>
<td></td>
<td>Peppers</td>
<td>7.9995</td>
<td>7.9995</td>
<td>7.9996</td>
</tr>
<tr>
<td>[21]</td>
<td>Lena</td>
<td>7.9972</td>
<td>7.9973</td>
<td>7.9970</td>
</tr>
<tr>
<td></td>
<td>Peppers</td>
<td>7.9993</td>
<td>7.9994</td>
<td>7.9994</td>
</tr>
<tr>
<td>[22]</td>
<td>Lena</td>
<td>7.9730</td>
<td>7.9750</td>
<td>7.9710</td>
</tr>
<tr>
<td></td>
<td>Peppers</td>
<td>7.9974</td>
<td>7.9974</td>
<td>7.9971</td>
</tr>
<tr>
<td>[23]</td>
<td>Lena</td>
<td>7.9993</td>
<td>7.9994</td>
<td>7.9992</td>
</tr>
</tbody>
</table>

This research was conducted at...
3.1.5. Correlation analysis

In (8) provides the correlation of an image with dimensions (n, m).

\[ \text{corr} = \frac{\text{cov}(x,y)}{\sqrt{\text{var}(x)\cdot\text{var}(y)}} \tag{8} \]

Table 2 illustrates a comparison between our approach and some similar approaches. This comparison proves that our approach is significantly better than the other compared algorithms. This verifies the security of our cryptosystem against statistical attacks. Table 3 illustrates a comparison between our approach and some similar approaches [21], [23], [24]. These results validate the resistance of our cryptosystem to statistical attacks.

| Table 2: Correlations between pixels for Lena, Apricot and Panda images |
|-----------------|----------|----------|----------|----------|----------|----------|----------|
|                  |           |          |          |          |          |          |          |
|                  | Original image | Encrypted image |
|                  | Horizontal | Vertical | Diagonal | Horizontal | Vertical | Diagonal |          |
| Lena Red        | 0.95580   | 0.96480  | 0.93220  | -0.000376 | -0.00015 | -0.00131 |
| Green           | 0.93556   | 0.95756  | 0.91902  | -0.00298 | 0.00912   | 0.00673 |
| Blue            | 0.90773   | 0.93930  | 0.89130  | -0.00145 | -0.00672 | 0.00064 |
| Apricot Red     | 0.98385   | 0.96944  | 0.98629  | -0.00137 | -0.00188 | -0.00541 |
| Green           | 0.97883   | 0.98511  | 0.96537  | -0.00107 | 0.00150   | 0.00234 |
| Blue            | 0.99153   | 0.98348  | 0.98724  | 0.00489  | -0.00571 | -0.00118 |
| Panda Red       | 0.95175   | 0.96552  | 0.93161  | 0.00513  | -0.00777 | -0.00495 |
| Green           | 0.95215   | 0.96436  | 0.93066  | 0.00788  | -0.00080 | 0.00027  |
| Blue            | 0.95542   | 0.97086  | 0.94265  | 0.000071 | 0.01097   | 0.00106  |

| Table 3: Correlation between ciphered “Lena” pixels |
|-----------------|----------|----------|----------|----------|
|                  |           |          |          |          |
|                  | Horizontal | Vertical | Diagonal |          |
| Proposed [21]   | -0.0027336| 0.0035   | -0.0024696|          |
| [23]            | -0.0029883| 0.0091357| -0.0067375|          |
| [24]            | -0.0042707| -0.0032498| -0.0020192|          |

3.4. Differential attacks

Differential attacks capitalize on differences in input processing within a cryptographic system. They target how the system reacts to minor alterations in plaintext or the key to infer critical details, such as the encryption key. To evaluate the algorithm’s performance against such attacks, metrics like the rate of pixel changes (NPCR), the unified average change intensity (UACI).

3.4.1. NPCR and UACI metrics analysis

These metrics can be given by (9) and (10).

\[ \text{NPCR} = \left( \frac{1}{3nm} \sum_{i,j=1}^{nm} |Df(i,j)| \right) \cdot 100 \tag{9} \]

\[ \text{UACI} = \left( \frac{1}{3nm} \sum_{i,j=1}^{nm} \frac{|Im_1(i,j) - Im_2(i,j)|}{255} \right) \cdot 100 \tag{10} \]

Where \( Df(i,j) = \begin{cases} 1 & \text{if } Im_1(i,j) \neq Im_2(i,j) \\ 0 & \text{if } Im_1(i,j) = Im_2(i,j) \end{cases} \)

Table 4 illustrates a comparison between our approach and some similar approaches. This comparison proves that our approach is significantly better than the other compared algorithms [21], [24]-[30]. This validates the resistance of our cryptosystem to differential attacks.

| Table 4: Comparison of the NPCR and UACI |
|-----------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
|                  |           | [21]     | [24]     | [28]     | [28]     | [29]     | [29]     | [30]     |
| Lena UACI (%)   | 33.55     | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    |
| Peppers UACI (%)| 33.53     | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    | 33.46    |
3.4.2. PSNR metric analysis

PSNR stands for mean squared error. \((n)\) denotes the number of rows in the original image, and \((m)\) represents the number of columns in the image. The PSNR is evaluated in decibels and is inversely proportional to the mean squared error. It is determined by (11).

\[
PSNR = 10 \times \log_{10}\left(\frac{2^{2n}-1}{MSE}\right) (dB)
\]  

- \(L=8\) denotes the bit depth of the particular image, \(MSE = \frac{1}{(nm)^2} \sum_{i=1}^{nm} (Im1(i,j) - Im2(i,j))^2\)
- \((Im1)\) and \((Im2)\) represent the original and encrypted images, respectively

Table 5 illustrates a comparison between our approach and some similar approaches. This comparison proves that our approach is significantly better than the other compared algorithms [23], [31]-[33]. This confirms the resilience of our cryptosystem against any differential attacks.

<table>
<thead>
<tr>
<th>Table 5. The PSNR (dB) between the original image, the encrypted image, and the decrypted image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>Lena</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Baboon</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

4. CONCLUSION

The obtained statistical and differential constants were evaluated according to international standards. To achieve this, we employed pseudo-random and reversible affine functions in the processes of confusion and diffusion. Additionally, two S-Boxes derived from chaotic maps were incorporated and framed by two specifically adapted genetic crossovers for the encryption of color images. This approach led to the development of a large-scale algorithm ensuring a uniform distribution of histograms for each encrypted image. Consequently, our cryptographic system demonstrates robustness against known attacks, as evidenced by comparisons with several similar algorithms.

REFERENCES


BIographies of authors

Hamid El Bourakkadi received the Master degree in Physics of Materials and Nanostructures from Sidi Mohammed Ben Abdellah University, Morocco, in 2012 and Master degree in Intelligent and Mobile Systems from Sidi Mohammed Ben Abdellah University, Morocco, in 2021, respectively. Currently, Ph.D. degrees in Mathematics and computer science in Mohammed First University, Oujda, Morocco. His research interests include computer science. He can be contacted at email: hamid.elbourakkadi.d23@ump.ac.ma.

Abdelhakim Chemlal received the Master degree in Computer Engineering with a Software Engineering specialization from National School of Applied Science in AL Houcimea, Morocco, in Currently, Ph.D. degrees in Mathematics and computer science in Mohammed First University, Oujda, Morocco. His research interests include computer science. He can be contacted at email: abdelhakim.chemlal.d23@ump.ac.ma.
Improved vigenere using affine functions surrounded by two genetic … (Hamid El Bourakkadi)

Hassan Tabti received the Master degree in Computer science Inforgaphy and Imaging from Sidi Mohammed Ben Abdellah University, Morocco, in 2014. Currently, Ph.D. degrees in Mathematics and computer science in Mohammed First University, Fez, Morocco. His research interests include computer science. He can be contacted at email: hassan.tabti1@usmba.ac.ma.

Mourad Kattass received a Master's degree in Robotics and Embedded Systems from the Faculty of Science and Technique, Abdelmalek ESSAADI University, Morocco, in 2020 and a Bachelor's degree in Computer Science, Electronic, Electrotechnical, and Automatic from Sidi Mohammed Ben Abdellah University, Morocco, in 2004, respectively. Currently, Ph.D. degrees in Mathematics and computer science in Mohammed First University, Oujda, Morocco. His research interests include computer science. He can be contacted at email: mourad.kattass@ump.ac.ma.

Abdellatif Jarjar received the Master degree in Fondamental Mathetimatics from Franche Compté Besonçon University, French, in 1987 and Laureate in Mathematics from High Normal School, Morocco, in 1988, respectively. Currently, Searcher in Mathematics and computer science in Mohammed First University, Oujda, Morocco. His research interests include computer science. He can be contacted at email: abdoujar@gmail.com.

Abdelhamid Benazzi Professor and searcher in computer science in Mohammed First University, Oujda, Morocco. His research interests include computer science. He can be contacted at email: a.benazzi @ump.ac.ma.