An image encryption algorithm with a novel chaotic coupled mapped lattice and chaotic image scrambling technique

Behrang Chaboki, Ali Shakiba
Department of Computer Science, Vali-e-Asr University of Rafsanjan, Iran

ABSTRACT

In this paper, we build a novel chaotic coupled lattice mapping with positive Lyapunov exponent, and introduce a novel chaotic image scrambling mechanism. Then, we propose a chaotic image encryption algorithm which uses the introduced chaotic coupled lattice mapping to apply permutation by iteratively applying the introduced chaotic image scrambling mechanism, and diffusing the pixel values. We use a sorting approach rather than quantizing the chaotic floating-point values to construct the diffusion matrix. We also study the security of the proposed algorithm concerning several security measures including brute-force attack, differential attack, key sensitivity, and statistical attacks. Moreover, the proposed algorithm is robust against data loss and noise attacks.

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

1. INTRODUCTION

In the current world, with the staggering speed of technology development in the era of digital computing and with the widespread application of the internet in our life, the application of digital color images become more and more inevitable. For instance, the application of digital images in medical imaging [6], social and personal life, military and other applications are almost clear to everyone. So, in some situations, such as medical images and military applications, the concept of privacy and security become one of the important challenges.

One strategy for solving this problem is to use some encryption techniques so that the image becomes unreadable for an unauthorized person. For approaching this problem, many encryption techniques have been proposed, such as compressive sensing [1], quantum theory [2], DNA coding [3], transform domains [4], matrix transforms [5] and chaotic systems [7–9]. In the last 20 years, the chaotic system encryption algorithms have been the attention of researchers due to its immense inherent aspects such as initial criteria sensitivity, unpredictability, and pseudorandomness. However, image encryption techniques that are solely based on chaotic systems, have been shown to be vulnerable against ciphertext-only and chosen plaintext attacks. Thus, the performance of a chaotic system and structure of encryption algorithm plays an important role in the resistance of an encryption scheme to the common attacks. By combining an appropriate chaotic-based system, in terms of initial sensitivity, unpredictability, and randomness with proper techniques of confusion and diffusion, a robust encryption algorithm will be acquired.
The rest of the paper is organized as follows: In Section 2, we first construct a novel chaotic coupled mapped lattice using two one-dimensional chaotic mappings and study its chaotic properties. Then, we generalize an image scrambling method and make it chaotic. These two primitives are combined to propose an image encryption algorithm. The security of the proposed method is studied with respect to several security measures in Section 3. Finally, a conclusion is drawn in Section 4.

2. METHOD

In this paper we use the logistic mapping in combination with the Chebyshev mapping of the first type, which is abbreviated as Chebyshev mapping. These mappings are combined with coupled mapped lattices, or CML for short, to construct a novel chaotic mapping with greater chaotic performance. The logistic mapping is a polynomial defined as

\[ x_{i+1} = L(\mu, x_i) = \mu x_i (1 - x_i) \]

where \( x_i \in [0, 1] \) for \( i = 0, 1, \ldots \) and \( \mu \in (0, 4] \). This mapping shows a chaotic behavior for \( \mu \in (3.5699456, 4] \) and this behavior improves as \( \mu \) gets closer to 4. The Chebyshev mapping is another chaotic mapping which is defined mathematically as

\[ T_n(x) = \cos(n \arccos(x)) \]

where \( x \in [-1, 1] \). Moreover, there is an equivalence recurrence relation used to define Chebyshev mappings as

\[ T_n(x) = 2xT_n(x) - T_{n-1}(x) \]

with initial conditions \( T_0(x) = T_1(x) = 1 \).

The CML originally uses the logistic mapping to generate sequences with the following relation

\[ y_{i+1} = (1 - \varepsilon)f[y_{i+1}] + \frac{\varepsilon}{2} \{ f[x_{i+1}] + f[x_{i-1}] \} \]

where \( \varepsilon \) is the coupling parameter and \( f[x] \) denotes the logistic map [10]. In this paper, we propose a novel chaotic map with excellent chaotic behavior using the CML with the logistic and the Chebyshev mappings. This novel mapping is defined as

\[ y_{i+1} = (1 - \varepsilon)L(\mu, T_{n+1}(x^{(i)})) + \frac{\varepsilon}{2} \{ L(\mu, T_{n+1}(x^{(i+1)})) + L(\mu, T_{n+1}(x^{(i-1)})) \} \]

where the sequences \( x^{(i)} \) are computed as follows:

- A random initial value \( x_0 \in (-1, 1) \) is generated
- A sequence of chaotic values \( s_i = T_\ell(s_{i-1}) \) are generated for \( i = 1, \ldots, m + m_0 \)
- We use the last \( m \) values of this sequence as the sequence \( x_0 \), i.e. \( x_0^{(i)} = s_{m+i} \) for \( i = 1, \ldots, m \)
- For \( n \geq 1 \), we set \( x_n^{(i)} = \frac{x_n^{(i)}}{1 + \max_i |y_n^{(i)}|} \), i.e. we normalize the sequence \( y_n \) to fall within the range \([-1, 1]\).

The proposed chaotic system has good chaotic properties, as it can be observed in Fig. 1 which illustrates its the Lyapunov exponent. Note that the Lyapunov exponent describes the rate of the convergence or divergence of trajectories and positive values of Lyapunov exponent show chaotic behavior [11].

![Figure 1. Bifurcation diagram of the proposed mapping with starting point \( x_0 = 1 \times 10^{-5} \) and \( n \in [0, 1 \times 10^5] \).](image-url)
Next, we describe our chaotic permutation algorithm which is a chaotic extension to an image scrambling procedure, which is introduced in [12]. First of all, the original image scrambling algorithm of [12] is as follows. The input to the original image scrambling algorithm is an image with a starting pixel, $IP$. As shown in Fig. 2 (a), suppose that the start position ($SP$) for scrambling of the image is the point $(5, 3)$, pixel with value 35. Based on this point, we divide the image into sub-blocks Fig. 2 (b). After this partitioning, those sub-blocks are converted into linear sequences by the following procedure. For the $SB_1$ and $SB_2$, we scan them down column-wise from left to right and right to left, respectively. The $SB_3$ and $SB_4$ are scanned upward column-wise from left to right and right to left, respectively. The $SB_5$ is scanned from left to right in a column-wise fashion and the $SB_6$ is scanned from top to bottom, in a row-wise fashion. This is illustrated in Fig. 2 (c). Then, all of the arrays are concatenated as $IP, SB_6, SB_5, SB_4, SB_3, SB_2$ and $SB_1$ to form a single array. Finally, the permuted image is re-constructed from this array in a row-wise fashion Fig. 2 (d).

To generalize this scrambling algorithm and make it chaotic, we make the following changes:
- As we iteratively apply this image scrambling technique for several times, we add an index value $i$ starting from zero.
- The rows and the columns of each of the submatrices $SB_1$ to $SB_8$ are permuted based on the chaotic vectors $R$ and $C$, respectively. To be precise, at each iteration $i$, the rows and columns of a sub-matrix $SB_j$ of size $r^{(j)} \times c^{(j)}$ are permuted with respect to the non-decreasing order of the numbers $R[i : i + r^{(j)}]$ and $R[i : i + c^{(j)}]$, respectively.
- Then, if we are in an even iteration, we convert $SB_1$ and $SB_6$ sub-matrices to vectors by appending their rows, and convert $SB_3$ and $SB_5$ sub-matrices to vectors by appending their columns. Moreover, in even iterations, we combine $SB_1$ through $SB_6$ and the starting pixel to obtain $IP$. In odd iterations, we convert $SB_1$ and $SB_6$ to vectors column-wise and $SB_3$ and $SB_5$ to vectors row-wise. Moreover, the $IP$ is constructed by concatenating $SB_1, SB_4, SB_6, SB_2, SB_3, SB_5, SB_3, SB_6$, and the starting index.

Now, we are ready to describe the proposed encryption algorithm. To be concrete, the variables used in the description of the proposed algorithm are as follows: - $\mu$ denotes the Coupling parameter in the CML
- $\ell$ is the Chebyshev mapping parameter to generate initial sequence $x_0$
- $s_0$ is the initial value used to generate initial sequence $x_0$
- $n_0$ is the number of initial iterations of the Chebyshev mapping in generating initial sequence to avoid transient effect
- $m$ is the length of sequences generated at each level
- $n_0$ is the number of initial iterations of the CML in generating sequences to avoid transient effect
- $n$ is the total number of sequences of length $m$ to generate by iterating the CML
- $r$ is the number of rows of the plain image
- $c$ is the number of columns of the plain image
- $K$ is a symmetric secret key used to encrypt the image
- $X$ denotes the plain image
- $Y$ denotes the encrypted image
- $h$ is the hash value of the plain image
- $R$ is the sequence used to permute rows of the plain image

---

Image encryption with a novel CCML and chaotic image scrambling technique (Behrang Chaboki)
- $C$ is the sequence used to permute the columns of the plain image
- $D$ is the sequence used to diffuse the pixels in the plain image.

The key $K$ and the hash value of the input image $h$ are given as inputs to the EXTRACTPARAMETERS algorithm to obtain the required encryption parameters as follows: $K' \leftarrow K + h \mod 2^{256}$.

- $\varepsilon \leftarrow \frac{k'}{2^{256}}$.
- Let $K' = (k'_{32}k'_{31} \ldots k'_{1})_2$ be the binary representation of the modified key in 32 Bytes, where $k'_i \in \{0,1\}^8$ for $i = 1, \ldots, 32$ and the plain (encrypted) image be of size $r \times c$.
- $n_0 \leftarrow \max \left( (k'_{31}k'_{29} \ldots k'_{1})_2 \mod 2^{128} \mod 4 \right),$
- $m \leftarrow \frac{4 \times r \times c}{n - 2} + 1,$
- $m_0 \leftarrow (k'_{31} \oplus k'_{29} \oplus \ldots \oplus k'_{17})_2 + (k'_{32} \oplus k'_{30} \oplus \ldots \oplus k'_{18})_2,
- $\ell \leftarrow (k'_{16} \oplus k'_{15} \oplus \ldots \oplus k'_{1})_2 + (k'_{32} \oplus k'_{31} \oplus \ldots \oplus k'_{17})_2,$
- $s_0 \leftarrow \frac{17}{2^{256}} - 1$, and $(10) \mu \leftarrow (k'_{\text{mod} 2^{24}} + 1) \times 0.43 + 3.57$.

The EXTRACTPARAMETERS algorithm is of constant time complexity, given that the key length is fixed. More precisely, it is linear in the key length in bits. We also use the GENERATESequences algorithm to generate chaotic sequences, which is described as follows: A chaotic matrix $SeqMat$ of size $n_0 + m$ by $m_0 + m$ chaotic values are generated by the proposed CML using the parameters of the Algorithm EXTRACTPARAMETERS.

The output of the algorithm is a $n \times m$ sub-matrix of $SeqMat$ by considering its last $n$ rows and its last $m$ columns, i.e. $SeqMat[-n :, -m :]$ in Python’s notation. The GENERATESequences algorithm requires $(n_0 + m) \times (m_0 + m)$ iterations of the proposed CML, e.g. is of time complexity $O(m \times n)$ considering evaluation of the proposed CML requires constant time. The ENCRYPTION algorithm is as follows: The hash value of the input image $X$ is computed by the SHA-256 algorithm and is denoted as $h$.

The encryption parameters $\varepsilon, \mu, n, n_0, m, m_0, \ell,$ and $s_0$ are extracted from the combination of the key $K$ and $h$ with the EXTRACTPARAMETERS algorithm.

The algorithm GENERATESequences is used to generate three chaotic sequences $R, C$ and $D$ of sizes $2 \times r, 2 \times c$ and $(n - 2) \times m$, respectively, as follows: (3-a) The first $2 \times r$ elements are sorted in a non-decreasing order and their corresponding sorted indices are assigned to the sequence $R$. (3-b) The next $2 \times c$ elements are sorted in a non-decreasing order and their corresponding sorted indices are assigned to the sequence $C$. (3-c) The next $r \times c$ elements are sorted in a non-decreasing order and their corresponding sorted indices are assigned to the matrix $D$, which is filled row-wise and is of shape $r \times c$. The sequences $R$ and $C$ are used to permute the input plain image according to the Step 4 and the sequence $D$ is used to diffus the input plain image according to the Step 5.

The sequences $R$ and $C$ are used to permute the input image using the chaotic extension of the image segmentation method of [12] by $X^{(a)} \leftarrow $ PERMUTEdIMAGE$(X^{(a-1)}, i, j)$, for $(R(i), C(i))$ where $R(i)$ and $C(i)$ denote the $i^{th}$ element of the sequences $R$ and $C$, respectively, and $a = 1, \ldots, \min r, c$. Let $X^{(m)}$ be the output of this step. - The sequence $D$ is used to diffuse the permuted image $X^{(m)}$ by $X^{D} \leftarrow X^{(m)} + D \mod 256$. pair $Y = (X^{D}, h)$ is the output of the ENCRYPTION.

The first and the second steps of the encryption algorithm can be considered of constant time complexity. In the third step of the encryption algorithm, we need to generate $m \times n$ chaotic elements from the CML sequence and then, sorting them which requires $O((m \times n) \log (m \times n) + m \times n)$ operations. In the fourth step, the input image is scrambled for $\min r, c$ steps, where each step requires $r \times c$ operations. Finally, the last step can be accomplished with $r \times c$ operations. Therefore, the time complexity of this algorithm is $O((m \times n) \log (m \times n) + r \times c \times \min(r, c) + r \times c + 1)$, e.g. it is polynomial in terms of the size of the input image.

Finally, an encrypted image can be decrypted by reversing the encryption process. More precisely, the DECRYPTION is as follows: Let the input of the algorithm be the pair $Y = (X^{D}, h)$ and the secret key $K$.

The encryption parameters $\varepsilon, \mu, n, n_0, m, m_0, \ell,$ and $s_0$ are extracted from the combination of the key $K$ and $h$ with the EXTRACTPARAMETERS algorithm.
The chaotic sequences $R$, $C$ and $D$ are generated with the same procedure as in the third step of the encryption algorithm. The diffusion is reversed by computing $Y^D = Y - D \mod 256$. The permutation is reversed by reversing each step of the chaotic image scrambling using sequences $R$ and $C$. It is easy to verify that the time complexity of the encryption algorithm and the decryption algorithm are the same, i.e., polynomial in the size of the image.

3. RESULTS AND DISCUSSION

An efficient encryption algorithm must demonstrate a proper security performance in encountering different security attack and could withstand them rigorously. To test the security of our encryption algorithm, we analyzing common factors, such as key space analysis, key and plaintext sensitivity, histogram analysis of encrypted image, entropy analysis, and correlation coefficient analysis.

The proposed algorithm is implemented in Python 3.6.7 and the results are obtained on an Intel Corei3-350 M Processor 2.26 GHz running Ubuntu 18.04 64-bit. The running time of the proposed algorithm is 1.44 seconds on average for plaintext images of size $512 \times 512$ using 256-bits key with this implementation. Our proposed algorithm is secure against the brute-force attack, since its key space is of size $2^{512}$.

Sensitivity against the encryption/decryption key is a requirement for a robust image encryption algorithm. There are three types of key sensitivity: (1) Decryption of an encrypted image with single bit perturbation in the legitimate key, (2) Decryption of an encrypted image with illegal keys, and (3) Encryption of the same image with single bit perturbation in the key. It is required that the result of each sensitivity analysis be different to each other as much as possible. The results of these tests for the proposed algorithm are reported in Tables 1-(a), 1-(b), and 1-(c), respectively. As it can be observed from these tables, the proposed algorithm provides acceptable sensitivity to the encryption key, as all these values are very close to their ideal values. Moreover, the encryption sensitivity of the proposed algorithm in the key is much higher than some recent chaotic image encryption algorithms, as it can be observed from Tables 1-(a) to 1-(c).

The main purpose of these tests is to ensure the diffusion property of an encryption algorithm, which means that the smallest change in plain image should have huge consequences in cipher image. The Unified Average Changing Intensity (UACI) and the Number of Pixels Change Rate (NPCR) are two of the most standard (or another synonym) tests that are used by researchers for testing the resistance of their encryption algorithm against the differential attacks (or chosen-plaintext attacks). The NPCR is used to calculate the percentage of difference between to image pixel numbers, on the other hand the UACI is used to measure the algorithm against the differential attacks (or chosen-plaintext attacks). The NPCR and UACI tests for two images $C_1$ and $C_2$ of size $M \times N$ are defined as the following equations:

$$\text{NPCR} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} |C_1(i,j) - C_2(i,j)|}{M \times N} \times 100\%,$$

and

$$\text{UACI} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} |D(i,j)|}{256 \times M \times N} \times 100\%,$$

where $D(i,j) = 1$ if $C_1(i,j) = C_2(i,j)$, otherwise $D(i,j) = 0$. In some cases, the NPCR and UACI cannot accurately detect the visual differences between the plain-image and cipher-image, so to compensate these criteria, we use BACI (Block Average Changing Intensity) test for analyzing our encryption algorithm. This test quantitatively evaluates the differences between our plain and cipher-images. Let $M$ and $P$ be two images of the size $r \times c \times 3$. Then, the value of their PSNR is calculated as

$$\text{PSNR}(M, P) = 20 \log \left( \frac{255}{\sqrt{\text{MSE}(M, P)}} \right)$$

where

$$\text{MSE}(M, P) = \frac{1}{\text{r} \times \text{c} \times 3} \sum_{i=1}^{r} \sum_{j=1}^{c} \sum_{k=1}^{3} (M(i, j, k) - P(i, j, k))^2.$$ 

Note that high numerical differences between the plain and the decrypted image results in lower values of PSNR.

In a differential attack, an attacker slightly changes the plain image and encrypts the original plain and the modified images with the same key. Then, he tries to trace the differences between the two encrypted images and use this knowledge to crack it. Our proposed algorithm provides acceptable sensitivity with respect to differential attack as is illustrated in Table 1-(d). Moreover, our proposed algorithm outperforms most of the recent image encryption algorithms, as it is shown in Table 1-(d).

In image processing and image encryption context, the histogram is a statistical feature of image that shows the frequency of each pixel intensity in grayscale or even color image. For grayscale images that are presented in this article, distribution of 256 different possible combinations are show graphically by image histogram. As depicted in the Figure 3, the uniform distribution of cipher image histogram shows that our algorithm resistance against statistical attacks.
Table 1. Sensitivity analysis where “—” denotes that the corresponding value is not available.

<table>
<thead>
<tr>
<th>Image</th>
<th>Sensitivity in the secret key</th>
<th>Sensitivity in illegal keys</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NPCR UACI BACI PSNR</td>
<td>NPCR UACI BACI PSNR</td>
</tr>
<tr>
<td>Pepper</td>
<td>99.61578 30.69388 0.23172 8.53623</td>
<td>99.60403 33.45612 0.26634 7.75197</td>
</tr>
<tr>
<td>Lena</td>
<td>99.60213 28.98183 0.21558 9.09798</td>
<td>99.61967 33.43398 0.26623 7.75487</td>
</tr>
<tr>
<td>Baboon</td>
<td>99.60728 27.95127 0.20694 9.48089</td>
<td>99.61891 33.37835 0.26645 7.76399</td>
</tr>
<tr>
<td>Ship</td>
<td>99.60918 28.44964 0.20764 9.29461</td>
<td>99.63074 33.46454 0.26656 7.75178</td>
</tr>
<tr>
<td>Ceremony</td>
<td>99.60678 32.66921 0.25568 7.95964</td>
<td>99.60785 33.47861 0.26727 7.74123</td>
</tr>
<tr>
<td>Cameraman</td>
<td>99.61529 28.85233 0.21541 9.15081</td>
<td>99.61967 33.41536 0.26648 7.75820</td>
</tr>
<tr>
<td>Girl</td>
<td>99.60777 32.39105 0.25241 8.03456</td>
<td>99.62425 33.43350 0.26648 7.75333</td>
</tr>
<tr>
<td>Home</td>
<td>99.60735 27.61322 0.19721 9.61956</td>
<td>99.60136 33.44234 0.26623 7.75573</td>
</tr>
<tr>
<td>Average</td>
<td>99.61022 29.70030 0.22282 8.89678</td>
<td>99.61581 33.43875 0.26648 7.75388</td>
</tr>
<tr>
<td>[14]</td>
<td>99.60860 28.69690 0.21382 —</td>
<td>63.62070 29.21160 0.34080 —</td>
</tr>
<tr>
<td>[16]</td>
<td>99.60960 28.63211 0.21321 —</td>
<td>99.60740 33.46030 0.26777 —</td>
</tr>
</tbody>
</table>

In a natural image, the correlation between adjacent pixels is usually high. Due to this high correlation between two adjacent pixels, the encryption algorithm could be vulnerable against statistical attacks. So, an encryption algorithm is as robust as possible if it could break down this relationship as much as possible. To measure the correlation of pixels in plain and cipher-images we randomly choose $3000$ pixels form original and corresponding cipher images in the vertical, horizontal and diagonal direction, and then compute the correlation coefficients. The results are given in Table 2-(a). As it can be observed from this data, our proposed algorithm outperforms four out of seven recent image encryption algorithms in all three directions, and outperforms the rest of them in at least one direction. Thus, it provides acceptable security in terms of correlation analysis.

The concept of information entropy is a solution for finding a degree of randomness and uncertainty in the image. As the information entropy of an image becomes closer to its ideal value, it suggests that the information is distributed randomly throughout the image. For grayscale images, the optimum value for information entropy is $8$, so as far as our analysis results become closer to this value, we can conclude that our encryption algorithm has a better uniform distribution of information. The entropy of a grayscale image $P$ is calculated by the following equations:

$$H(P) = - \sum_{i=0}^{255} p(i) \log_2 p(i),$$

where $p(i)$ is the fraction of pixels with color $i$ in the image $P$. As it can be verified in Table 2b, our proposed method provides acceptable security in terms of entropy analysis compared with several recent image encryption algorithms.

It is vital for a secure image encryption algorithm to be robust against data loss and noise attacks, since the presence of noise and data loss is quite usual in real world scenarios. The proposed algorithm satisfies this requirement as we have tested it by cropping $200 \times 200$ random square from an encrypted image and then decrypted it. Moreover, we have also added a Salt & Pepper noise of $0.1\%$, $1\%$ and $5\%$ to the encrypted image and decrypted it. As it can be visually observed in Figures 4 and 5, the proposed encryption scheme provides an acceptable robustness against noise and data loss attacks.
Figure 3. Histogram analysis for the proposed algorithm for a sample image.

Figure 4. Crop Attack $200 \times 200$. (a) Plaintext image, (b) Encrypted image, (c) Cropped encrypted image, (d) Decrypted image of the cropped encrypted image.

Figure 5. Noise attacks: (a) Plaintext image, (b) Encrypted image, (c) Salt and Pepper noise of 0.01 added to encrypted image, (d) Decryption of encrypted image with Salt and Pepper noise of 0.01 added, (e) Salt and Pepper noise of 0.1 added to encrypted image, (f) Decryption of encrypted image with Salt and Pepper noise of 0.1 added, (g) Salt and Pepper noise of 0.5 added to encrypted image, (h) Decryption of encrypted image with Salt and Pepper noise of 0.5 added, (i) Gaussian noise of mean 0 and variance 0.01 added to encrypted image, (j) Decryption of the encrypted image with Gaussian noise of mean 0 and variance of 0.01 added.

Image encryption with a novel CCML and chaotic image scrambling technique (Behrang Chaboki)
Table 2. Correlation and entropy analysis for plain and encrypted images with the proposed algorithm for all images.

(a) Correlation analysis

<table>
<thead>
<tr>
<th>Image</th>
<th>Correlation direction</th>
<th>Horizontal</th>
<th>Vertical</th>
<th>Diagonal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pepper</td>
<td>0.01403</td>
<td>0.00189</td>
<td>-0.00372</td>
<td></td>
</tr>
<tr>
<td>Lena</td>
<td>-0.03244</td>
<td>0.00820</td>
<td>0.04202</td>
<td></td>
</tr>
<tr>
<td>Baboon</td>
<td>0.00194</td>
<td>0.01289</td>
<td>-0.01126</td>
<td></td>
</tr>
<tr>
<td>Ship</td>
<td>0.00687</td>
<td>0.01787</td>
<td>0.04778</td>
<td></td>
</tr>
<tr>
<td>Cameraman</td>
<td>0.00559</td>
<td>0.00523</td>
<td>0.00272</td>
<td></td>
</tr>
<tr>
<td>Girl</td>
<td>-0.00811</td>
<td>0.01079</td>
<td>-0.00783</td>
<td></td>
</tr>
<tr>
<td>Home</td>
<td>-0.01582</td>
<td>-0.00224</td>
<td>0.05586</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>0.01320</td>
<td>0.00910</td>
<td>0.02259</td>
<td></td>
</tr>
<tr>
<td>[18]</td>
<td>0.02424</td>
<td>0.02608</td>
<td>0.02446</td>
<td></td>
</tr>
<tr>
<td>[20]</td>
<td>0.0493</td>
<td>0.0172</td>
<td>0.043375</td>
<td></td>
</tr>
<tr>
<td>[27]</td>
<td>-0.01055</td>
<td>0.0141</td>
<td>0.0081</td>
<td></td>
</tr>
<tr>
<td>[22]</td>
<td>0.039886</td>
<td>0.034475</td>
<td>0.001949</td>
<td></td>
</tr>
<tr>
<td>[23]</td>
<td>0.07700</td>
<td>-0.07236</td>
<td>-0.06153</td>
<td></td>
</tr>
<tr>
<td>[28]</td>
<td>0.0120</td>
<td>0.0917</td>
<td>0.1019</td>
<td></td>
</tr>
</tbody>
</table>

(b) Entropy for all images

<table>
<thead>
<tr>
<th>Image</th>
<th>Entropy Plain</th>
<th>Entropy Encrypted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pepper</td>
<td>7.59373</td>
<td>7.99939</td>
</tr>
<tr>
<td>Lena</td>
<td>7.44507</td>
<td>7.99934</td>
</tr>
<tr>
<td>Baboon</td>
<td>7.35815</td>
<td>7.99931</td>
</tr>
<tr>
<td>Ship</td>
<td>7.19137</td>
<td>7.99933</td>
</tr>
<tr>
<td>Cameraman</td>
<td>6.70385</td>
<td>7.99927</td>
</tr>
<tr>
<td>Girl</td>
<td>7.48451</td>
<td>7.99936</td>
</tr>
<tr>
<td>Home</td>
<td>7.20101</td>
<td>7.99929</td>
</tr>
<tr>
<td>Average</td>
<td>7.39129</td>
<td>7.99933</td>
</tr>
<tr>
<td>[19]</td>
<td>5.570216</td>
<td>7.99881</td>
</tr>
<tr>
<td>[20]</td>
<td>5.570216</td>
<td>7.99881</td>
</tr>
<tr>
<td>[21]</td>
<td>7.560453</td>
<td>7.989367</td>
</tr>
<tr>
<td>[22]</td>
<td>6.501775</td>
<td>7.98972</td>
</tr>
<tr>
<td>[23]</td>
<td>7.350433</td>
<td>7.989740</td>
</tr>
</tbody>
</table>

4. CONCLUSION

In this paper, we have constructed a novel chaotic coupled mapped lattice using the chaotic Logistic mapping and the Chebyshev mapping. We have also studied the chaotic behavior of the proposed chaotic mapping in terms of its Lyapunov exponent, which was greater than zero. We have also generalized an image scrambling method proposed in [12] by making it chaotic and iterative. Then, we have combined these two primitives to encrypt images by generating chaotic sequences from the novel proposed chaotic mapping and then, applying permutation using the proposed generalized chaotic image scrambling method. To construct the diffusion matrix, we have used a sorting approach rather than quantizing the chaotic floating-point values. We have also studied the security performance of the proposed encryption algorithm concerning brute-force attacks, key sensitivity attacks, differential attack, and statistical analysis. Moreover, the robustness of the proposed method is tested. The results of all these tests showed acceptable security in comparison with several recent image encryption algorithms.

REFERENCES


**BIOGRAPHIES OF AUTHORS**

**Behrang Chaboki** is a lecturer of Computer Science at the Vali-e-Asr University of Rafsanjan. His research interests include Algorithms and Computational Complexity. He holds a M.Sc. in Computer Science from Sharif University of Technology.
Ali Shakiba is an assistant professor of Computer Science at the Vali-e-Asr University of Rafsanjan. His research interest include Cryptography, Machine Learning and Chaotic Systems. He holds a Ph.D. in Computer Science from Yazd University.