Automata-Derived Chaotic Image Encryption Scheme

Ioannis Kafetzis
Laboratory of Nonlinear Systems, Circuits & Complexity (LaNSCom), Department of Physics Aristotle University of Thessaloniki
Thessaloniki, Greece
kafetzis@physics.auth.gr

Lazaros Moysis
Laboratory of Nonlinear Systems, Circuits & Complexity (LaNSCom), Department of Physics Aristotle University of Thessaloniki
Thessaloniki, Greece
lmousis@physics.auth.gr

Christos Volos
Laboratory of Nonlinear Systems, Circuits & Complexity (LaNSCom), Department of Physics Aristotle University of Thessaloniki
Thessaloniki, Greece
volos@physics.auth.gr

Hector Nistazakis
Section of Electronic Physics and Systems Department of Physics, National and Kapodistrian University of Athens
Athens, Greece
enistaz@phys.uoa.gr

Jesus M. Munoz-Pacheco
Facultad de Ciencias de la Electronica Benemérita Universidad Autónoma de Puebla
Puebla, Mexico
jesusm.pacheco@correo.buap.mx

Ioannis Stouboulos
Laboratory of Nonlinear Systems, Circuits & Complexity (LaNSCom), Department of Physics Aristotle University of Thessaloniki
Thessaloniki, Greece
stouboulos@physics.auth.gr

***Abstract*—This work introduces an encryption scheme for gray-scale plain-text images, which is based on a chaotic map. Initially, the proposed chaotic map, which is a modification of the Renyi map, is introduced and is utilized in defining a Pseudo-Random Bit Generator. Subsequently, a finite automaton is introduced. This, in combination with the aforementioned PRBG defines the encryption strategy, that is, the order in which the rows and columns are encrypted. The proposed method is subjected to a number of statistical tests, to prove its resistance against common attacks.**

***Index Terms*—Chaos, Finite Automata, Image Encryption, Pseudo-Random Bit Generation**

I. Introduction

Chaos theory is a well established field which finds numerous applications in a wide scientific spectrum that includes physics, engineering, and computer science. Examples of such applications are secure communications, optimization, encryption and more [1]. Chaotic systems, most commonly low dimensional chaotic maps, are predominantly used as a deterministic source of entropy, with the added advantages of low computational cost and ease of implementation.

One of the most common applications of chaotic maps are Pseudo-Random Bit Generators, or PRBGs, [2], [3]. The prominent use of chaos based PRBGs is data encryption. New techniques for chaotic data encryption are constantly developed, with emphasis given on image encryption, see [4], [5] for an overview of recent results.

Also, in recent years, the use of automata in combination with chaotic maps for encryption is gaining attention [6]–[9]. Automata are discrete dynamical models, that can describe a sequence of transitions between states, and can be used to model interactions between discrete entities, like cells, machines, or discrete events [10], [11]. Automata are prominent in applications due to their close relation with logics, that can efficiently encode their behavior. In turn, this allows to use automata for developing coherent methods for validation, where the goal is to verify that a system or a piece of software is behaving according to the requirements of the design process [12]. Applications commonly utilize cellular automata in the definition of the encryption scheme, which are involved in either the diffusion process, that is, in the shuffling of the rows and columns [8] or the confusion process, where it contributes to altering the pixel values [9].

Motivated by the above, this work proposes an automaton driven chaos based image encryption technique. In our work, the automaton is used, in combination with a PRBG, to determine the order in which the operations of confusion and diffusion are performed. More explicitly, the permutation and encryption of the rows and columns of an image are intertwined, based on the order generated from the transitions of a finite state automaton, which is driven by a chaotic PRBG. This PRBG is designed through the values of a modified Renyi map [13], which uses an additional modulo operator for increased randomness [14]. A key advantage of this method is that the order in which the rows and columns are encrypted is random, since it is connected to the values of the PRBG. On the other hand, using finite automata to describe such a process guarantees that the method operates properly.

Finally, the performance of the proposed encryption method is tested using a collection of measures such as key space, histogram, entropy and correlation analysis. All the tests performed verify the security of the design against different types of potential adversarial attacks.

The rest of the work is structured as follows: In Section II the chaotic map used is defined and its behavior is presented through its bifurcation and Lyapunov exponent diagrams. Section III contains the definition of a PRBG using the proposed map. In Section IV the proposed map is utilized in order to create a permutation of a given set of integers, while Section V describes the main step used in the encryption method. Subsequently, Section VI presents the finite automaton and its role in determining the encryption scheme. In Section VII the complete encryption process is discussed, while in Section VIII the security of the proposed encryption scheme is verified. Finally, Section IX concludes the work.

II. A Polynomial-Renyi Chaotic Map

The discrete time chaotic system used in this work is a polynomial-Renyi map, described by

$$x_{k+1} = \mathrm{mod}(p \cdot x_k^3 + \mathrm{mod}(100 \cdot x_k, 1), 1) \qquad (1)$$

where $p > 0$ is a parameter that, for this study, is assumed to take values inside the interval $(0, 10)$. The bifurcation diagram

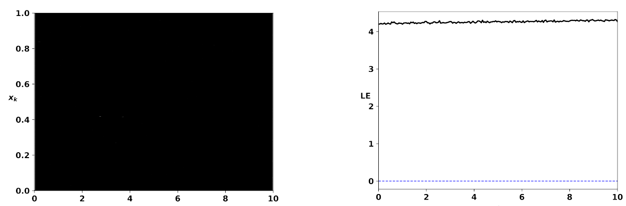

Fig. 1: Bifurcation diagram and Lyapunov exponent diagram of (1) with initial condition $x_0 = 0.6$.

for (1) with initial value 0.6 and Lyapunov exponent of the system are depicted in Fig. 1. The system is chaotic for all $p \in (0, 10)$, as indicated by the positive Lyapunov exponent, and is thus a robust chaotic map [15]. This robustness, combined with the low computational cost render (1) suitable for use in chaotic encryption schemes.

III. Pseudo Random Bit Generator

The values of the proposed chaotic map (1) are utilized in the definition of a PRBG. From any value x_k of the map (1), a random bit is produced as the boolean result of the comparison

$$b_k = \mathrm{mod}(1234 \cdot x, 2) < 1. \qquad (2)$$

The randomness of the proposed PRBG is verified through the NIST statistical suite [16] for 100 bitstreams, each consisting of 10^6 bits. The results of the tests are sucessful and are presented in Table I.

IV. Generation of Permutations

The PRBG defined in Section III is utilized in the definition of a method to generate permutations of a given set of integers. The end goal is, given an integer n, create a set, namely $\mathcal{N}$, that contains all the integers $1, \ldots, n$ in arbitrary order. This method is used to determine the order in which the rows and columns of the plain-text image are encrypted. Let n be a

TABLE I: Results of the NIST statistical suite test for the proposed PRBG for $x_0 = 0.6$ and $p = 5$.

Test	p-Value
Frequency	0.851383
BlockFreq.	0.637119
Cum.Sums	0.883171
Runs	0.319084
LongestRun	0.554420
Rank	0.437274
FFT	0.319084
NonOverl.Templ.	0.897763

Test	p-Value
Overl. Templ.	0.455937
Universal	0.171867
Approx. Entropy	0.350485
Rand.Excur.	0.016717
Rand.Excur.Var.	0.037566
Serial	0.494392
LinearComplexity	0.085587

given integer. Initially, the map (1) is iterated n times, to create a hash-map, of the form $(1, x_1)$, $(2, x_2)$, $\ldots$, (n, x_n). Subsequently the elements of hash-map are ordered in ascending order of chaotic map values. The permutation is obtained via the rearranged hash-keys.

For example, let $n = 5$, and consider (1) with $p = 5$ and $x_0 = 0.6$. The generated hash map and the result after reordering are shown on the left and right of Table II respectively. Thus, the permutation for $n = 5$ is $\mathcal{N} = \begin{bmatrix} 3, & 2, & 4, & 1, & 5 \end{bmatrix}$.

TABLE II: Example of obtaining a permutation via the values of the proposed map.

Index	System Value
1	0.6
2	0.07999999999999985
3	0.00255999999998479
4	0.25600008388455964
5	0.6838945509178695

Index	System Value
3	0.00255999999998479
2	0.07999999999999985
4	0.25600008388455964
1	0.6
5	0.6838945509178695

V. Row and Column Encryption

Suppose now that the plain-text is a gray-scale image $I \in [0, 255]^{m \times n}$. Using the values of the PRBG we can generate permutations of the sets $\{0, 1, \ldots, m-1\}$ and $\{0, 1, \ldots, n-1\}$ namely $\mathcal{R} = \{r_0, r_1, \ldots, r_{m-1}\}$, and $\mathcal{C} = \{c_0, c_1, \ldots, c_{n-1}\}$ that will be used to shuffle and encrypt the rows and columns of the image. The method for encrypting rows is discussed next. The method for columns is obtained as the dual of that of the rows.

Consider the index $r_k \in \mathbb{N}$. Then the k^{th} and ${r_k}^{\text{th}}$ rows of the matrix are involved in the current step. Initially, using the PRBG proposed in Section III, a random bit sequence of length $8 \cdot n$, namely ρ, is generated. Using ρ, the k^{th} row is encrypted by performing the element-wise XOR operation. Subsequently, the ${r_k}^{\text{th}}$ row is element-wise XORed with the result of the previous XOR. After changing the values in both k^{th} and ${r_k}^{\text{th}}$ row, the positions of the two rows are swapped, and the step is complete.

Decryption is achieved by performing the steps in reverse. Initially, the positions of the two vectors, either rows or columns, are swapped. Subsequently, performing the XOR operation between vectors decrypts the second one. Finally, performing the XOR operation between the remaining vector and the random bitstream recovers the first vector as well.

VI. Automaton Defined Encryption Scheme

In this section we present the automaton describing the encryption process. A graphical representation of the automaton is shown in Fig. 2, with a unique initial and final state, namely I and F, respectively.

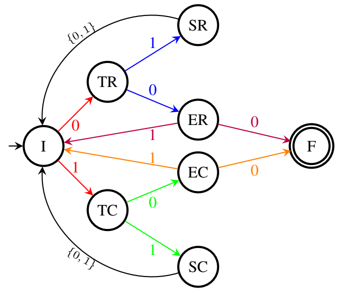

Fig. 2: Automaton describing encryption process.

The language of the automaton, that is, the set of words that drive the automaton from the initial state I to the final state F, is $\mathcal{L} = \left(0^2 1 \cup 01\{0,1\} \cup 1^2\{0,1\} \cup 101\right)^* \left(0^3 \cup 10^2\right)$ where $A = \{0, 1\}$ is the input alphabet and S^* denotes the concatenation of any number of elements of the set S. Observe that the operations and powers shown in $\mathcal{L}$ denote concatenation of characters and not integer multiplication. The above automaton is utilized in determining the order in which the rows and columns of the image are encrypted. Each state in Fig 2 is named according to the role it plays in the encryption scheme. Clearly I and F are the initial and final states. TR (resp. TC) denotes testing if all the rows (resp. columns) have been encrypted, SR (resp. SC) denotes swapping and encrypting rows (resp. columns). Finally ER (resp. EC) test if all of the columns (resp. rows) have been encrypted, given that all of the rows (reps. columns) have been encrypted. It is thus of uttermost importance to establish how the input letters 0 and 1 are determined for each state of the automaton. Assume that the goal is to encrypt an $m \times n$ gray-scale image. Before starting to iterate on the automaton, two counters namely i_r and i_c are set to zero. When the system is in the initial state I, then the letter for the next system transition comes from iterating the PRBG. The input letter at state TR (respectively TC) is 1 if $i_r < m$ (resp. $i_c < n$), and 0 otherwise. Furthermore, the index i_r (resp. i_c) is incremented by 1, each time the automaton reaches the state SR (resp. SC). If the automaton is at state SR or SC, then both 0 and 1 drive the system to the I state, hence without any loss the letter 1 is always given. For the state ER (resp. EC), the input letter is 1 if $i_c < n$ (resp. $i_r < m$) is non-empty and 0 otherwise. The state F is the final state and no transition from it is allowed.

Every word generated using the above process is recognized by the automaton. Initially, suppose that $i_r < m$ and $i_c < n$. Then starting from the initial condition I, all the possible inputs are $011, 111$ which both drive the automaton back into the initial state I. If $i_r \geq m$ and $i_c < n$ and the state of the automaton is $\mathcal{I}$, the possible inputs are 001, 111, which both drive the system back into I. Due to symmetry, the case where $i_r < m$ and $i_c \geq n$ drive the automaton into the initial state as well. Finally, suppose that the automaton is at its initial state, $i_r \geq m$ and $i_c \geq n$. Then both possible inputs 100 or 000, lead to the final state F. Hence, any word determined using the described input values is recognized by the automaton.

Fig. 3: Plain-text image and the ciphered imaged resulting from the proposed method.

VII. Image Encryption Scheme

Assume that the plain-text gray-scale image is represented as an $m \times n$ matrix. An initial condition x_0 and parameter p for the system (1) constitute the secret key of the method. These are also transmitted from the source to the receiver via a secure communication channel. Using these, a map as in (1) is defined. This map is initially used to define sets permutations of $\{1, \ldots, m\}$ and $\{1, \ldots, n\}$, namely $\mathcal{R}$ and $\mathcal{C}$ as in Sec. IV. The automaton, as in Fig. 2, is then utilized to define the order in which the rows and columns are encrypted. Subsequently, the automaton is used, with the input letters being determined as discussed in Sec. VI. We keep track of the order in which the states SR and SC appear. When the automaton reaches its final state, keep the first m and n occurrences of SR and SC, in the same order they appeared as automaton states in a list $\mathcal{O}$. Thus, $\mathcal{O}$ contains $m + n$ elements, indicating the order of encryption of the rows and columns. For each SR or SC in $\mathcal{O}$, the respective index in the sets $\mathcal{R}$ and $\mathcal{C}$ determines which rows or columns are involved in each encryption step. Finally, the encryption in each step is performed as discussed in Sec. VII. The method is complete when the encryption operation is performed for every element indicated by $\mathcal{O}$.

The decryption process follows similar steps. More explicitly, taking the same initial conditions into consideration, the sets $\mathcal{R}$ and $\mathcal{C}$ are recreated. Furthermore, the automaton allows for the recreation of the $\mathcal{O}$ set. Having the set $\mathcal{O}$, all of the sequences of random bits can be recreated in order. Finally, having computed all of the bit streams the image is decrypted by taking the elements of $\mathcal{O}$ in reverse order and performing the decryption process based on the row and column indices.

VIII. Method Application and Evaluation

The proposed method has been implemented using Python and is applied to the "peppers" image, which was downloaded from https://sipi.usc.edu/database/. The plain and cipher-text images are depicted in Fig. 3.

One of the most important characteristics for the encryption method is its key space, which has to have a value of more than 2^{100} [17] in order to guarantee that the proposed method is secure against brute-force attacks. System (1) requires a parameter p and an initial condition x_0. Thus, assuming 16-bit accuracy, a lower bound for the key space is $10^{(2 \cdot 16)} > 2^{100}$.

One of the most common types of attacks against image encryption schemes is the histogram attack. It is desired that the encrypted image's histogram is close to uniform, masking the existence of meaningful information. The proposed method achieves this, as can be verified through Fig. 4.

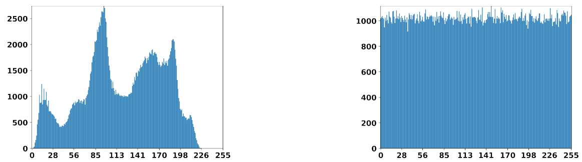

Fig. 4: Histogram of the plain-text and cipher images.

Another important security test is that of information entropy, which is used to determine the security of the method against entropy attacks. The entropy of an image is calculated as the sum $\sum_{i=0}^{2^8-1} p(i)\log_2(p(i))$ where $p(i)$ denotes the probability that i appears as a pixel value. Information entropy having a value of 8 indicates that the pixel values are random [18]. Calculating the information entropy for the example cipher image leads to a value of 7.9991995 which is close enough to 8, so that the pixel values are considered random.

Finally, when an image contains meaningful information, adjacent pixels tend to have similar values, thus leading to adjacent row and columns of the image having high correlation. It is desired that this is not transferred to the resulting image. In Fig. 5 the correlation of subsequent rows and columns for the plain-text and cipher-text images are presented. It can be seen that the scatter plot for the original image has high correlation values and also has a structure, none of which holds for the cipher-image.

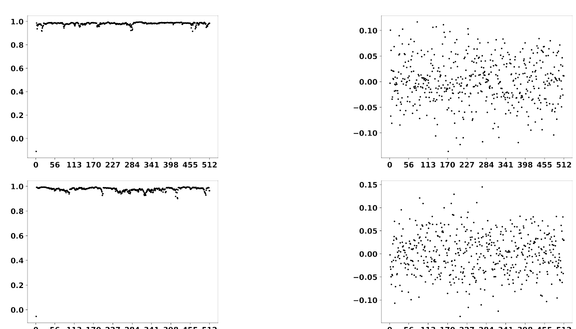

Fig. 5: Correlation of subsequent rows (above) and columns (below) for the plain-text and cipher-text images.

IX. Conclusions

In this work, a chaos based image encryption technique was developed, using a mixed row and column permutation, an encryption rule, generated using a chaotic PRBG, and a finite state automaton. The use of automata in this work allows the method to imbue randomness upon the confusion and diffusion precesses while guaranteeing that desired standards for the encryption scheme, such as shuffling every image row and column, are met. In the future, the proposed encryption method's resistance against more types of attacks shall be investigated. Furthermore, different types of automata such as fuzzy [19] will be considered.

X. Acknowledgments

The authors acknowledge funding from National and Kapodistrian University of Athens (NKUA), Special Account for Research Grants (SARG).

References

[1] G. Grassi, "Chaos in the real world: Recent applications to communications, computing, distributed sensing, robotic motion, bio-impedance modelling and encryption systems," *Symmetry*, vol. 13, no. 11, p. 2151, 2021.

[2] H. Zang, Y. Yuan, and X. Wei, "Research on pseudorandom number generator based on several new types of piecewise chaotic maps," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[3] R. B. Naik and U. Singh, "A review on applications of chaotic maps in pseudo-random number generators and encryption," *Annals of Data Science*, pp. 1–26, 2022.

[4] M. Kumar, A. Saxena, and S. S. Vuppala, "A survey on chaos based image encryption techniques," in *Multimedia security using chaotic maps: principles and methodologies*. Springer, 2020, pp. 1–26.

[5] K. M. Hosny, *Multimedia security using chaotic maps: principles and methodologies*. Springer Nature, 2020, vol. 884.

[6] X. Wang and D. Luan, "A novel image encryption algorithm using chaos and reversible cellular automata," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 11, pp. 3075–3085, 2013.

[7] A. Bakhshandeh and Z. Eslami, "An authenticated image encryption scheme based on chaotic maps and memory cellular automata," *Optics and Lasers in Engineering*, vol. 51, no. 6, pp. 665–673, 2013.

[8] B. Mondal, S. Singh, and P. Kumar, "A secure image encryption scheme based on cellular automata and chaotic skew tent map," *Journal of information security and applications*, vol. 45, pp. 117–130, 2019.

[9] P. K. Naskar, S. Bhattacharyya, D. Nandy, and A. Chaudhuri, "A robust image encryption scheme using chaotic tent map and cellular automata," *Nonlinear Dynamics*, vol. 100, no. 3, pp. 2877–2898, 2020.

[10] J. L. Schiff, *Cellular automata: a discrete view of the world*. John Wiley & Sons, 2011, vol. 45.

[11] S. Zhou, Z. Yu, E. S. A. Nasr, H. A. Mahmoud, E. M. Awwad, and N. Wu, "Homomorphic encryption of supervisory control systems using automata," *IEEE Access*, vol. 8, pp. 147 185–147 198, 2020.

[12] M. Pittou and G. Rahonis, "Architecture modelling of parametric component-based systems," in *International Conference on Coordination Languages and Models*. Springer, 2020, pp. 281–300.

[13] A. A. Alzaidi, M. Ahmad, M. N. Doja, E. Al Solami, and M. S. Beg, "A new 1d chaotic map and β-hill climbing for generating substitution-boxes," *IEEE Access*, vol. 6, pp. 55 405–55 418, 2018.

[14] Z. Hua, Y. Zhang, and Y. Zhou, "Two-dimensional modular chaotification system for improving chaos complexity," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1937–1949, 2020.

[15] Z. Hua and Y. Zhou, "Exponential chaotic model for generating robust chaos," *IEEE transactions on systems, man, and cybernetics: systems*, vol. 51, no. 6, pp. 3713–3724, 2019.

[16] L. Bassham, A. Rukhin, J. Soto, J. Nechvatal, M. Smid, S. Leigh, M. Levenson, M. Vangel, N. Heckert, and D. Banks, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," 2010-09-16 2010. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762

[17] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *International journal of bifurcation and chaos*, vol. 16, no. 08, pp. 2129–2151, 2006.

[18] J. Liang, Z. Shi, D. Li, and M. J. Wierman, "Information entropy, rough entropy and knowledge granulation in incomplete information systems," *International Journal of general systems*, vol. 35, no. 6, pp. 641–654, 2006.

[19] S. Zamani, M. Javanmard, N. Jafarzadeh, and M. Zamani, "A novel image encryption scheme based on hyper chaotic systems and fuzzy cellular automata," in *2014 22nd Iranian Conference on Electrical Engineering (ICEE)*. IEEE, 2014, pp. 1136–1141.