

Pseudo-random bit generator based on multi-modal maps

M. García-Martínez · E. Campos-Cantón

Received: 4 September 2014 / Accepted: 24 July 2015 / Published online: 9 August 2015
© Springer Science+Business Media Dordrecht 2015

Abstract In this work we present a pseudo-random Bit Generator via unidimensional multi-modal discrete dynamical systems called k -modal maps. These multi-modal maps are based on the logistic map and are useful to yield pseudo-random sequences with longer period, i.e., in order to attend the problem of periodicity. In addition the pseudo-random sequences generated via multi-modal maps are evaluated with the statistical suite of test from NIST and satisfactory results are obtained when they are used as key stream. Furthermore, we show the impact of using these sequences in a stream cipher resulting in a better encryption quality correlated with the number of modals of the chaotic map. Finally, a statistical security analysis applied to cipher images is given. The proposed algorithm to encrypt is able to resist the chosen-plaintext attack and differential attack because the same set of encryption keys generates a different cipher image every time it is used.

Keywords Chaotic behavior · Lyapunov exponent · Bifurcation parameter · Bifurcation diagram · Pseudo-random generator · NIST · Stream cipher · Key stream · k -modal maps

1 Introduction

In these days information security has become an important issue due to the fact that every day the Internet has more and more activity such as online banking, e-commerce, e-mail. Considering that the internet is an open channel, the information circulating on it can be intercepted by other users, so in order to solve this drawback we make use of cryptography and an scheme of confidentiality.

A variety of ciphers have been proposed with the intention of providing confidentiality and keep secretly the information, but these ciphers have been proposed for text like DES [1, 2], IDEA [3], AES [4], RSA [5]. These encryption schemes do not seem to be ideal for image applications, due to some intrinsic features of images such as bulk data capacity, high pixel correlation and high redundancy.

On the other hand, dynamical systems have been a very active area of research and specifically those with chaotic behavior; since their inception they have been developed in many areas such as communication systems [6], neural networks [7], switching systems [8], synchronization [9] and cryptography [10]. There is a close relationship between chaos and cryp-

M. García-Martínez (✉) · E. Campos-Cantón
División de Matemáticas Aplicadas, Instituto Potosino de Investigación Científica y Tecnológica, Camino a la Presa San José 2055, 78216 San Luis Potosí, S.L.P., México
e-mail: moises.garcia@ipicyt.edu.mx

E. Campos-Cantón
e-mail: eric.campos@ipicyt.edu.mx

M. García-Martínez
Colegio de la Frontera Sur CHETUMAL, Av. del Centenario Km. 5.5, 77900 Chetumal, Q. Roo, México

tography, for example in [11] authors made a comparison between the properties of these areas and show that the ergodicity, sensitivity to initial conditions and the control parameter, mixing property, deterministic dynamics and complex structure are analogous to confusion, sensitivity to key, diffusion, deterministic pseudo-randomness and algorithm complexity, respectively.

Most of the image ciphers are in the framework of symmetric key, and they are based on blocks [12]. Stream ciphers may provide better security under the concept of perfect security; however, this kind of ciphers has various critical requirements and one of them is the randomness of the key stream.

There have been proposals of generators based on unimodal chaotic maps, for example in [13, 14] proposed a pseudo-random bit generator by using only one logistic map, but in [15] the authors pointed out that bit streams generated through only one chaotic system are potentially insecure due to the output may leak some information about the chaotic system. In order to overcome the aforementioned vicissitude, they proposed a pseudo-random bit generator based on a couple of piecewise linear chaotic maps, which are iterated independently and the bit streams are generated by comparing the outputs of these chaotic maps. A pseudo-random bit generator is proposed in [16] which employs the logistic map as a perturbation and a piecewise linear chaotic map as the main generator. Other works such as [17–19] used two chaotic maps and combined to obtain a complex sequence of bits. In this paper we show how the use of only a multi-modal chaotic map may replace two unimodal chaotic maps in the generation of pseudo-random sequences.

In this work we propose a new architecture for a pseudo-random bit generator based on k -modal chaotic maps where we can yield a complex sequence of bits with one multi-modal map. The sequences are analyzed by the statistical suite of tests named NIST. This paper is organized as follows. In the next Section we introduce the k -modal maps. In Sect. 3 a pseudo-random bit generator is presented via k -modal maps. In Sect. 4 the statistical suite of tests of randomness proposed by NIST are used to prove safety of the sequences and the results of the tests are shown. In Sect. 5 we use a stream cipher and applied statistical security test like entropy, pixel correlation, encryption quality. Furthermore, we show that the proposed cryptosystem can resist the most

common attacks such as chosen-plaintext attack and differential attack. Finally in Sect. 6 we give conclusions about the pseudo-random bit generator via multi-modal chaotic maps.

2 Multi-modal maps

A discrete-time dynamical system is used to construct a pseudo-random bit generator and is given as follows:

$$x_{n+1} = f(x_n), \quad n = 0, 1, 2, \dots, N,$$

where $x_n \in \mathfrak{R}$ and x_0 is the initial condition, such dynamical system is usually referred to as **map**, as it is fully determined by its right-hand side. To ensure boundedness of trajectories, the study is usually restricted to maps that are mapping a compact interval into itself and without loss of generality we may consider only the compact interval $[0, 1]$. The simplest maps are the so-called unimodal maps like the logistic and tent maps, while their generalization, the so-called multi-modal or k -modal maps may present even more rich dynamical behaviors, [20, 21].

To be more specific, we denote $\mathcal{I} := [0, 1]$ and recall, that the **critical point** c of the continuous piecewise smooth map $f(x) : \mathcal{I} \mapsto \mathcal{I}$ is $c \in \mathcal{I}$ where f is differentiable and $f'(c) = 0$. The critical point c occurs for $f'(c) = 0$ or $f'(c)$ does not exist. But continuous smooth maps always present $f'(c) = 0$.

Definition 1 [20] The map $f : \mathcal{I} \mapsto \mathcal{I}$ is called a k -modal mapping, if it is continuous on \mathcal{I} and it has k critical points denoted by $c_0, c_1 \dots c_{k-1}$ in \mathcal{I} . Moreover, there exist intervals $\mathcal{I}_i, i = 0, \dots, k-1, \cup_{i=1}^k \mathcal{I}_{i-1} = \mathcal{I}$, such that $\forall i = 0, \dots, k-1$ it holds $c_i \in \mathcal{I}_i$ and $f_\beta(c_i) > f_\beta(x), \forall x \in \mathcal{I}_i$ and $x \neq c_i$, where β is a parameter. The case $k = 1$ is simply referred as to the so-called **unimodal** map.

The above definition does not constraint a function to have only k critical points. However, it only considered those that are local maximum on a subinterval. This means that the number k defines the maximal numbers of modals in a family \mathcal{F} and the interval $\mathcal{I} = [a, b]$ is divided between k subintervals $\mathcal{I}_0 = [d_1, d_2), \mathcal{I}_1 = [d_2, d_3), \dots, \mathcal{I}_{k-2} = [d_{k-2}, d_{k-1}), \mathcal{I}_{k-1} = [d_{k-1}, d_k]$ so the system f_β is a piecewise function by k unimodal maps.

The parameterized family \mathcal{F} of maps $f_\beta(x)$ is defined by the following piecewise function

$$f_\beta(x) = \beta(d_{r+1} - x)(x - d_r), \quad x \in \mathcal{I}_r \tag{1}$$

where $d_r = r/k$, ($r = 0, 1, 2, \dots, k - 1$), k is the number of modals, $\beta = \beta(k, \gamma)$ is the bifurcation parameter, $\gamma = 1/k$ is the carrying on capacity. To obtain the maximum value of β with k modals there is a direct relationship, $\beta_{\max} = (4)(k)/(\gamma)$ for more detail information see ref. [20].

Let's make an example of k -modal map with $k = 3$, to construct the function of the monoparametric family we use the equation (1) and we will have 3 subintervals \mathcal{I}_r , ($r = 0, 1, 2$), so the function f_β which is given by the equation (2) is expressed as follows

$$f_\beta(x) = \beta \begin{cases} (1/3 - x)(x), & \text{for } x \in [0, 1/3]; \\ (2/3 - x)(x - 1/3), & \text{for } x \in [1/3, 2/3]; \\ (1 - x)(x - 2/3), & \text{for } x \in [2/3, 1]; \end{cases} \tag{2}$$

where $\beta \in [0, 36]$ is the bifurcation parameter, depending on the value of β the system may be unimodal, bi-modal or tri-modal.

Figure 1 shows a bifurcation diagram of the tri-modal map $f_\beta(x)$. In order to demonstrate whether the system is chaotic, there are several ways, but the most common for discrete dynamical systems are based on the definition of Devaney [22] and Lyapunov exponent [23, 24]. In this paper we show that the system is chaotic

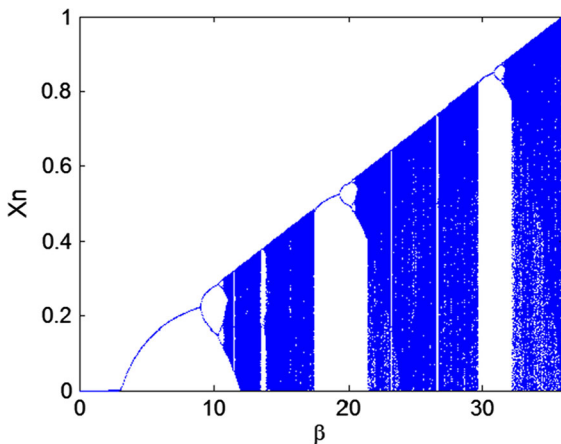


Fig. 1 Bifurcation diagram of the tri-modal map given by Eq. 2

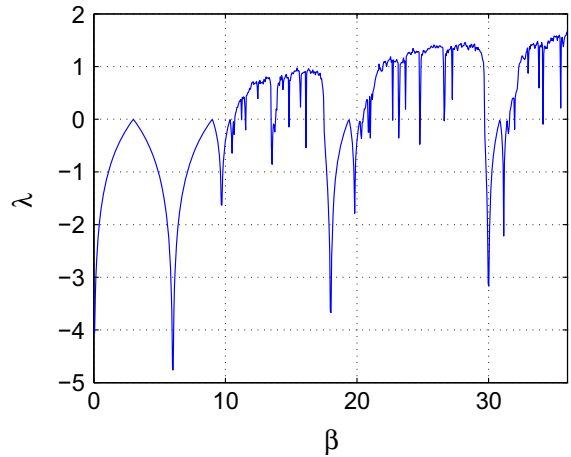


Fig. 2 Lyapunov exponent of the tri-modal map

by means of Lyapunov exponent which is denoted by λ as is shown in Fig. 2. When $\lambda \leq 0$ the system's orbit converges to a fixed point or in a periodic orbit and when $\lambda > 0$ the system's orbit behaves chaotically.

3 Proposed pseudo-random bit generator

First we introduce some basic concepts of random and pseudo-random number generators which are given in [5].

Definition 2 A True Random Number Generator (TRNG) is characterized by the fact that its output cannot be reproduced, so this type of generator is based on physical processes like semiconductor noise. In cryptography, a TRNG is often needed for generating session keys but not for stream ciphers.

Definition 3 A pseudo-random number generator (PRNG) produces sequences which are computed from an initial seed value, note that a PRNG is not random in a true sense because it can be computed by an algorithm, thus it is completely deterministic. A common requirement of a PRNG is that it possesses good statistical properties, i.e., its output that resembles to a sequence of true random numbers.

Definition 4 A cryptographically secure pseudo-random number generator (CSPRNG) is a special type of PRNG which possesses the following additional property: a CSPRNG is a PRNG which is unpredictable. This means that given n consecutive bits of

the key stream, there is no polynomial time algorithm that can predict the next bit s_{n+1} with better than 50 % chance of success. Another property of CSPRNG is that given the above sequence, it is computationally unfeasible to compute any preceding bits s_{n-1}, s_{n-2} .

In this paper we propose a CSPRNG using one k -modal map and a combination of its k -time series. The algorithm for the construction of the generator is given for any value of k and an example is also given for the particular case of tri-modal map. The algorithm to produce a CSPRNG is as follows:

Step 1 Set the value of $k \in \mathbb{N}^+$.

Step 2 Compute the values of β_j , for $j = 1, \dots, k$, by means of the following equations.

$$\beta_1 = (4)(k); \tag{3}$$

$$\beta_j = j * \beta_1; \text{ for } j \geq 2. \tag{4}$$

Taking these values of β_j we are avoiding periodic windows and guarantee chaotic orbits.

Step 3 Take the values of β_j and split the space into $2 * j$ regions $\delta_1^j, \dots, \delta_{2*j}^j$ which are determined by values $\kappa_1^j, \dots, \kappa_{(2*j)-1}^j$. Iterating the system $x_n^j = f_{\beta_j}(x_n)$ and depending on which region evolves, i.e., δ_1^1, δ_2^1 represent the value of 0 or 1, this generates a binary sequence ζ^j . Note that the generation of the number of 0's is approximately equal to the number of 1's in all generated sequences with a tolerance of 1%.

Step 4 A k number of chaotic time series are generated by $x_n^j = f_{\beta_j}(x_n)$ and each one produces a binary sequence ζ^j . These sequences are mixed and the final sequence Z is gotten as follows:

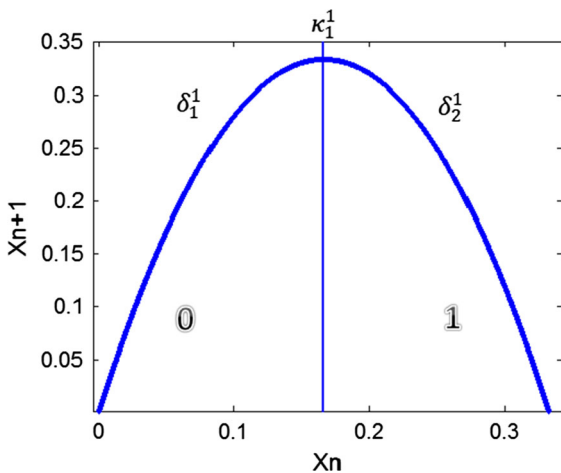


Fig. 3 Plot of the phase space of the tri-modal map with $\beta = \beta_1$

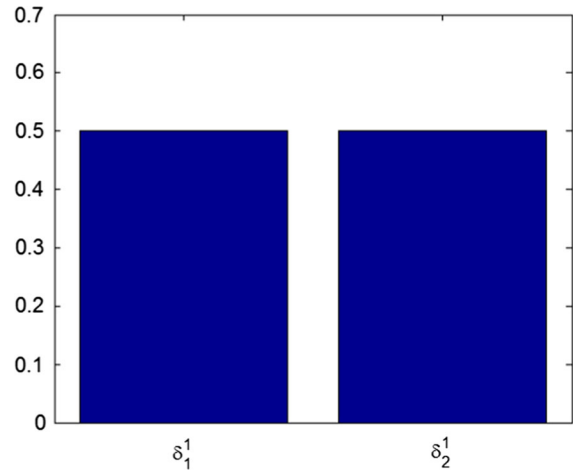


Fig. 4 Distribution of the 2 regions from the 3-modal map with $\beta = \beta_1$

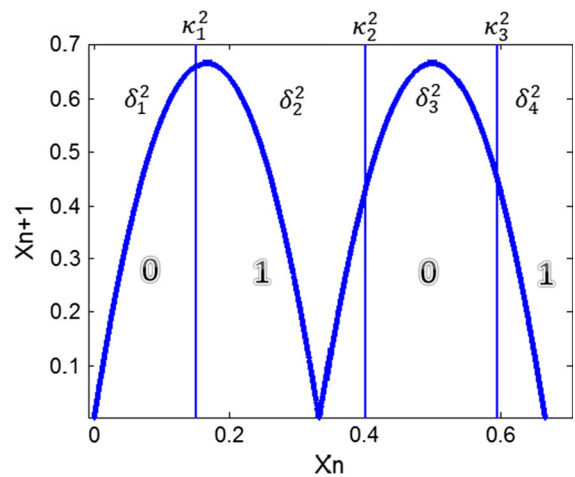


Fig. 5 Plot of the phase space of the tri-modal map with $\beta = \beta_2$

$$Z = \zeta^1 \oplus \zeta^2 \oplus \dots \oplus \zeta^k. \tag{5}$$

In order to clarify the mechanism to generate CSPRNG, we exemplify the process for $k = 3$ and iterate the system 1,000,000 times. So the first step is given by $k = 3$. According to the second step, $\beta_1 = 12, \beta_2 = 24, \beta_3 = 36$. The third step is to take β_1 for obtaining the sequence ζ^1 , therefore the space is divided in two regions δ_1^1 and δ_2^1 split by $\kappa_1^1 = 1/6$. These regions represent a 0 and a 1, respectively, as is shown in Fig. 3. The histogram of the two regions is shown in Fig. 4. In the same step three, we need to compute ζ^2 and ζ^3 , so carry on with β_2 , we have a bi-modal map with four regions $\delta_1^2, \dots, \delta_4^2$ separated by $\kappa_1^2 = 0.1521, \kappa_2^2 = 0.40174, \kappa_3^2 = 0.5954$. In Fig. 5 is

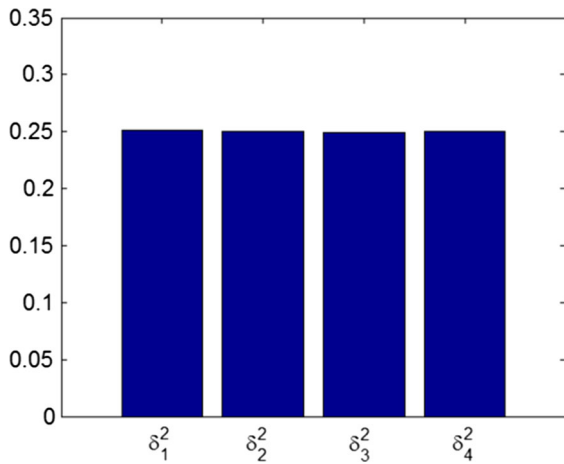


Fig. 6 Distribution of the tri-modal map with $\beta = \beta_2$

shown the bi-modal map for β_2 and the regions δ 's. Fig. 6 shows the histogram of the binary sequence ζ^2 . Now for the last case β_3 splits the space in six regions $\delta_1^3, \dots, \delta_6^3$ which are determined by the values $\kappa_1^3 = 0.1465, \kappa_2^3 = 0.396, \kappa_3^3 = 0.639, \kappa_4^3 = 0.8335, \kappa_5^3 = 0.9572$.

Finally, the sequences are mixed by $Z = \zeta^1 \oplus \zeta^2 \oplus \zeta^3$, where \oplus is the operation XOR. We have computed some values that κ can take for different values of $k = 1, \dots, 4$ and they are shown in Table 1.

Table 1 Summary of values κ

Map	$\beta = \beta_1$	$\beta = \beta_2$	$\beta = \beta_3$	$\beta = \beta_4$
$k = 1$	$\kappa_1^1 = 1/2$	–	–	–
$k = 2$	$\kappa_1^1 = 1/4$	$\kappa_1^2 = 0.2273$ $\kappa_2^2 = 0.6020$ $\kappa_3^2 = 0.8931$	–	–
$k = 3$	$\kappa_1^1 = 1/6$	$\kappa_1^2 = 0.1521$ $\kappa_2^2 = 0.4017$ $\kappa_3^2 = 0.5954$	$\kappa_1^3 = 0.1465$ $\kappa_2^3 = 0.3960$ $\kappa_3^3 = 0.6390$ $\kappa_4^3 = 0.8335$ $\kappa_5^3 = 0.9572$	–
$k = 4$	$\kappa_1^1 = 1/8$	$\kappa_1^2 = 0.1140$ $\kappa_2^2 = 0.3012$ $\kappa_3^2 = 0.4465$	$\kappa_1^3 = 0.1105$ $\kappa_2^3 = 0.2970$ $\kappa_3^3 = 0.4790$ $\kappa_4^3 = 0.6250$ $\kappa_5^3 = 0.7180$	$\kappa_1^4 = 0.1070$ $\kappa_2^4 = 0.2911$ $\kappa_3^4 = 0.4829$ $\kappa_4^4 = 0.6576$ $\kappa_5^4 = 0.8034$ $\kappa_6^4 = 0.9114$ $\kappa_7^4 = 0.9776$

For security reasons and in order to increase the key space, we use the high sensitivity to initial conditions, then we iterate the algorithm 200 times without considering its output bits.

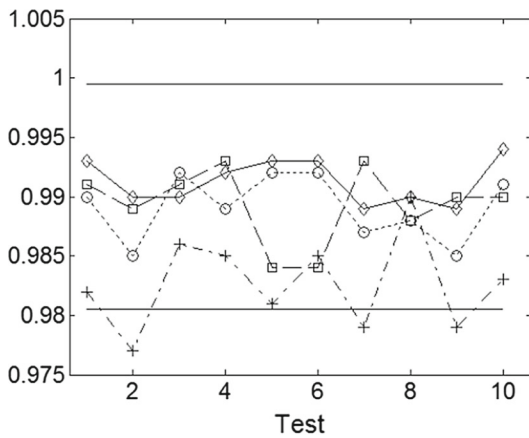
Thus, the correlation is checked between the produced sequences $x_n^j = f_{\beta_j}(x_n)$ by the proposed algorithm (step 3) with nearby keys and also it is verified the sensitivity to initial conditions. The correlation coefficient C_{xy} for each pair of sequences $x = [x_1, \dots, x_N]$ and $y = [y_1, \dots, y_N]$ are computed as follows [19]:

$$C_{xy} = \frac{\sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})}{\left[\sum_{i=1}^N (x_i - \bar{x})^2\right]^{1/2} \left[\sum_{i=1}^N (y_i - \bar{y})^2\right]^{1/2}} \quad (6)$$

where $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$ and $\bar{y} = \frac{1}{N} \sum_{i=1}^N y_i$ are the mean values of x and y , respectively. The coefficients C_{xy} are computed for each pair of produced sequences with nearby initial conditions, and this has been done for every value of j . Remember when the parameter k increases, then the number of sequences and the number of initial conditions also increase, for example, in the case of bi-modal map $k = 2$. There are two sequences x_n^1 and x_n^2 with initial conditions x_{01} and x_{02} for evaluate C_{xy} , we need other two sequences x_n^1 and x_n^2 with small different initial conditions $x_{01} + \delta$ and $x_{02} + \delta$. The corresponding data are listed in Table 2

Table 2 Correlation coefficients of pseudo-random sequences

Unimodal	$x_n^1 \rightarrow x_{01} = 0.24879652$	$x_n^{r1} \rightarrow x_{01} + \delta = 0.248796520000001$	$C_{x^1x^{r1}} = 0.0015$
Bimodal	$x_n^1 \rightarrow x_{01} = 0.19685672$	$x_n^{r1} \rightarrow x_{01} + \delta = 0.196856720000001$	$C_{x^1x^{r1}} = 0.0018$
	$x_n^2 \rightarrow x_{02} = 0.86241957$	$x_n^{r2} \rightarrow x_{02} + \delta = 0.862419570000001$	$C_{x^2x^{r2}} = -0.0022$
Trimodal	$x_n^1 \rightarrow x_{01} = 0.38461795$	$x_n^{r1} \rightarrow x_{01} + \delta = 0.384617950000001$	$C_{x^1x^{r1}} = -0.0004$
	$x_n^2 \rightarrow x_{02} = 0.73519846$	$x_n^{r2} \rightarrow x_{02} + \delta = 0.735198460000001$	$C_{x^2x^{r2}} = -0.0013$
	$x_n^3 \rightarrow x_{03} = 0.48617852$	$x_n^{r3} \rightarrow x_{03} + \delta = 0.486178520000001$	$C_{x^3x^{r3}} = -0.0016$
Quadmodal	$x_n^1 \rightarrow x_{01} = 0.26487146$	$x_n^{r1} \rightarrow x_{01} + \delta = 0.264871460000001$	$C_{x^1x^{r1}} = -0.0006$
	$x_n^2 \rightarrow x_{02} = 0.52975314$	$x_n^{r2} \rightarrow x_{02} + \delta = 0.529753140000001$	$C_{x^2x^{r2}} = 0.00001$
	$x_n^3 \rightarrow x_{03} = 0.93457812$	$x_n^{r3} \rightarrow x_{03} + \delta = 0.934578120000001$	$C_{x^3x^{r3}} = -0.0004$
	$x_n^4 \rightarrow x_{04} = 0.63789514$	$x_n^{r4} \rightarrow x_{04} + \delta = 0.637895140000001$	$C_{x^4x^{r4}} = 0.0008$



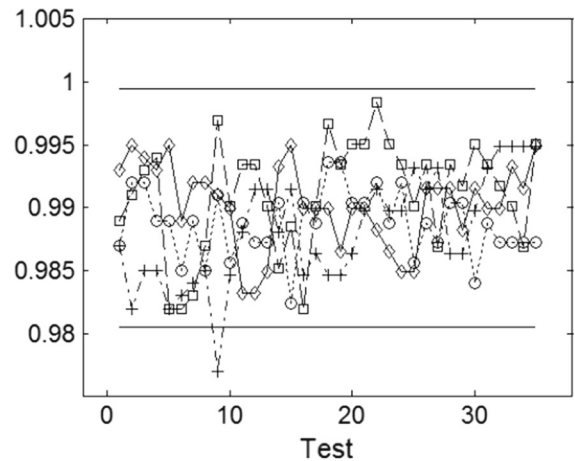
- 1) Frequency (Monobit) Test
- 2) Frequency Test within a Block (Block=128)
- 3) Runs Test
- 4) Test for the Longest Run
- 5) Binary Matrix
- 6) Discrete Fourier Transform (Spectral Test)
- 7) Overlapping Template Matching Test (Block=9)
- 8) Maurer's Universal Statistical Test
- 9) Approximate Entropy
- 10) Linear Complexity

Fig. 7 Part 1 of the results of the suite of statistical tests

where it is possible to see that there is no correlation between the generated sequences as a consequence that the chaotic map is very sensitive to very small changes in all initial conditions.

4 The statistical suite test

In order to characterize the proposed generator and demonstrate that it is safe for use in cryptography, it



- 1) - 2) Serial Test (Block=16)
- 3) - 4) Cumulative Sums
- 5) - 9) Non-overlapping Template Matching Test (Block=9)
- 10) - 17) Random Excursions Test
- 18) - 35) Random Excursions Variant Test

Fig. 8 Part 2 of the results of the suite of statistical tests

must be analyzed with a variety of statistical tests. These statistical tests determine whether the generated sequences possess specific characteristics similar to truly random sequences. To achieve this goal there are several options available for analyzing the randomness of the pseudo-random bit generators, for example the suite developed by Beker and Piper [25], the Gustafson's suite [26] or the DIEHARD suite [27]. However, the most used and standard test is defined by the NIST [28] that contains a sufficient number of independent statistical tests and detects any deviation from the randomness.

A random process means lack of pattern or predictability in events, and it is a sequence of random variables describing a process whose outcomes do not follow a deterministic pattern, but follow an evolution described by probability distributions. So a randomness is a probabilistic property; that is, a theoretical reference distribution of this statistic is determined by mathematical methods, in addition the results of statistical testing must be interpreted with some care and caution to avoid incorrect conclusions about a specific generator. First we need to define a significance level σ . Typically, it is chosen in the range [0.001, 0.01], by default $\sigma = 0.01$ and indicates that one would expect one sequence of 100 sequences to be rejected by the test if the sequence is random.

The NIST has adopted two ways to interpret results. In this paper we used the examination of the proportion of sequences that pass a statistical test. For this we need the confidence interval defined as:

$$(1 - \sigma) \pm 3\sqrt{\frac{1 - (1 - \sigma)}{m}}, \tag{7}$$

where $\sigma = 0.01$ and m is the sample size of sequences that in our case is 1,000 sequences and each has 1,000,000 of elements. If the proportion falls outside

of this interval, then there is evidence that the data is non-random.

In Figs. 7 and 8, we show the confidence interval of sequences generated with $k = 1, \dots, 4$. The dash-dot line with marker plus “+” represents the result for uni-modal map or logistic map, and we can see that there are some results outside of the confidence interval then we cannot use this sequence for cryptography. On the other hand, the dot line with marker empty circle “o” corresponds to the bi-modal map, the dash line with marker square “□” corresponds to the tri-modal map, and the solid line with marker diamond “◇” is the quad-modal map. This information is summarized in Tables 3 and 4.

It is clear that the maps with $k \geq 2$ lie inside the confidence interval; hence, these sequences are cryptographically secure according to the suite of tests proposed by NIST.

4.1 Differential attack

This technique of cryptanalysis is mainly applicable on block ciphers and was introduced by Biham and Shamir [29]. The basic idea is to analyze the effect of a small difference in two inputs and the difference of corresponding two cipher outputs. In this case we applied

Table 3 Part 1 of results from statistical suite of tests

Test name	Unimodal portion passing	Result	Bi-modal portion passing	Result	Tri-modal portion passing	Result	Quad-modal portion passing	Result
Frequency test	0.9820	Success	0.9900	Success	0.9910	Success	0.9930	Success
Frequency test within a block	0.9770	Fail	0.9850	Success	0.9890	Success	0.9900	Success
Runs test	0.9860	Success	0.9920	Success	0.9910	Success	0.9900	Success
Test for the longest run	0.9850	Success	0.9890	Success	0.9930	Success	0.9920	Success
Binary matrix	0.9810	Success	0.9920	Success	0.9840	Success	0.9930	Success
Discrete Fourier transform	0.9850	Success	0.9920	Success	0.9840	Success	0.9930	Success
Overlapping template matching test	0.9790	Fail	0.9870	Success	0.9930	Success	0.9890	Success
Maurer’s universal statistical test	0.9900	Success	0.9880	Success	0.9880	Success	0.9900	Success
Approximate entropy	0.9790	Fail	0.9850	Success	0.9900	Success	0.9890	Success
Linear complexity	0.9830	Success	0.9910	Success	0.9900	Success	0.9940	Success

Table 4 Part 2 of results from statistical suite of tests

Test name	Unimodal portion passing	Result	Bi-modal portion passing	Result	Tri-modal portion passing	Result	Quad- modal portion passing	Result
Serial test 1	0.9870	Success	0.9870	Success	0.9890	Success	0.9930	Success
Serial test 2	0.9820	Success	0.9920	Success	0.9910	Success	0.9950	Success
Cumulative sums test								Success
a) Forward	0.9850	Success	0.9920	Success	0.9930	Success	0.9940	Success
b) Backward	0.9850	Success	0.9890	Success	0.9940	Success	0.9930	Success
Non-overlapping template matching test								
a)	0.9820	Success	0.9890	Success	0.9820	Success	0.9950	Success
b)	0.9830	Success	0.9850	Success	0.9820	Success	0.9890	Success
c)	0.9840	Success	0.9890	Success	0.9830	Success	0.9920	Success
d)	0.9850	Success	0.9850	Success	0.9870	Success	0.9920	Success
e)	0.9770	Fail	0.9910	Success	0.9970	Success	0.9910	Success
Random excursions test								
a) -4	0.9850	Success	0.9860	Success	0.9900	Success	0.9900	Success
b) -3	0.9880	Success	0.9890	Success	0.9930	Success	0.9830	Success
c) -2	0.9920	Success	0.9870	Success	0.9930	Success	0.9830	Success
d) -1	0.9920	Success	0.9870	Success	0.9900	Success	0.9850	Success
e) 1	0.9890	Success	0.9900	Success	0.9850	Success	0.9930	Success
f) 2	0.9920	Success	0.9820	Success	0.9880	Success	0.9950	Success
g) 3	0.9850	Success	0.9900	Success	0.9820	Success	0.9900	Success
h) 4	0.9860	Success	0.9890	Success	0.9900	Success	0.9900	Success
Random excursions variant test								
a) -9	0.9850	Success	0.9930	Success	0.9960	Success	0.9900	Success
b) -8	0.9850	Success	0.9930	Success	0.9930	Success	0.9880	Success
c) -7	0.9860	Success	0.9900	Success	0.9950	Success	0.9900	Success
d) -6	0.9900	Success	0.9900	Success	0.9950	Success	0.9900	Success
e) -5	0.9910	Success	0.9920	Success	0.9980	Success	0.9880	Success
f) -4	0.9900	Success	0.9890	Success	0.9950	Success	0.9870	Success
g) -3	0.9900	Success	0.9920	Success	0.9930	Success	0.9850	Success
h) -2	0.9930	Success	0.9860	Success	0.9900	Success	0.9850	Success
i) -1	0.9920	Success	0.9890	Success	0.9930	Success	0.9910	Success
i) 1	0.9930	Success	0.9870	Success	0.9870	Success	0.9910	Success
k) 2	0.9860	Success	0.9900	Success	0.9930	Success	0.9910	Success
l) 3	0.9860	Success	0.9900	Success	0.9920	Success	0.9880	Success
m) 4	0.9900	Success	0.9840	Success	0.9950	Success	0.9910	Success
n) 5	0.9930	Success	0.9890	Success	0.9930	Success	0.9900	Success
o) 6	0.9950	Success	0.9870	Success	0.9910	Success	0.9900	Success
p) 7	0.9950	Success	0.9870	Success	0.9900	Success	0.9930	Success
q) 8	0.9950	Success	0.9870	Success	0.9870	Success	0.9920	Success
r) 9	0.9950	Success	0.9870	Success	0.9950	Success	0.9950	Success

the same idea to pseudo-random sequences and instead of having two inputs with plaintext, there are two initial conditions and an output with the corresponding sequences, the difference can be used as:

1. Sum of absolute difference (SAD) which is applicable to sequences $x_n^j \in \mathfrak{N}$ and is defined by $SAD = |(x_n^j - x_n'^j)|$ where the sequences x_n^j and $x_n'^j$ are obtained with initial conditions x_{01} and $x_{01} + \delta$, respectively.
2. XOR difference is applicable to sequences $Z, Z' \in \{0, 1\}$ and is defined by $D = Z \oplus Z'$ where the sequences Z and Z' are obtained with k initial conditions.

In previous section we analyzed the sequences $x_n^j \in \mathfrak{N}$ with nearby initial conditions, now the goal in this section is to analyze the $Z, Z' \in \{0, 1\}$ sequences (step 4 of the proposed algorithm) with nearby initial conditions through the XOR Difference. Recall that for obtaining the sequence Z we need k initial conditions, for example if $k = 3$ we need three initial conditions x_{01}, x_{02}, x_{03} to obtain the sequence Z , and three initial conditions $x'_{01}, x'_{02}, x'_{03}$ to obtain the sequence Z' and then apply the XOR Difference. In the worst case of nearby initial conditions we have $x_{01} = x_{02} = x_{03} = x'_{01} = x'_{02}, x'_{03} = x_{03} + \delta$, with $\delta \neq 0$ which means that we have the same initial conditions to generate x_n^1, x_n^2 and small difference in x_n^3 for Z' . When we applied the XOR Difference to Z and Z' , we obtain that approximately 50% of the elements in the sequences are different. Even if we have the same initial condition $x_{01} = x_{02}$ the series $x_n^1 \neq x_n^2$ because the $f_{\beta_j}(x_n)$ is different in each case. We can conclude that there is no correlation between the binary sequences Z and Z' , so the chaotic map is very sensitive in x_n^j and Z .

5 Chaos based gray-scale image cryptosystem

In this section we show a stream cipher algorithm for cipher gray-scale images and the impact of using multi-modal maps, i.e., the quality of the cipher image depends on the pseudo-random bit generator employed. At the end of the last section, we can infer that a multi-modal map generates a sequence cryptographically secure, but in order to determine which sequence is better, we need a statistical security analysis applied to cipher images instead of key stream.

We use the following algorithm for cipher the gray-scale image:

- First we consider an image of $N \times M$ pixels, secondly we add a column of random values $R(i, 1) \in \{0, 1, \dots, 255\}$ with $i = 1, \dots, N$, so the size of the image increases to $N \times (M + 1)$. This image will be called augmented image (\bar{P}) with a total of $(N)(M + 1)$ pixels.
- Compute pixel by pixel as follows:

$$\begin{cases} C_1 = \bar{P}_1 \oplus Z_1 \oplus IV \\ C_i = \bar{P}_i \oplus Z_i \oplus C_{i-1} \end{cases} \tag{8}$$

where C_i is the i th pixel of the cipher image, \bar{P}_i is the i th pixel of the augmented image, note that the value of $\bar{P}_1 = R(1, 1)$, Z_i is the key stream according to the proposed algorithm given in Sect. 3, $IV \in \{0, 255\}$ is an initialization vector of 8 bits, it is used only once and \oplus is the operation XOR bit by bit. We generate different key streams from multi-modal maps with $k = 1, 2, 3, 4$.

For decrypt the image we need the same key stream for this we need the values: k , the initial conditions $x_{01}, x_{02}, \dots, x_{0k}$ and the initialization vector (IV) as follows:

$$\begin{cases} \bar{P}_1 = C_1 \oplus Z_1 \oplus IV \\ \bar{P}_i = C_i \oplus Z_i \oplus C_{i-1} \end{cases} \tag{9}$$

Note that the decrypted image is the augmented image then has one extra column so when the process of decrypt is completed we need to remove this column to get original size of the image $(N)(M)$. In Fig. 9 are some examples of encrypt and decrypt images.

Some statistical analysis has been performed on the proposed image encryption scheme, including the correlation of pixels, entropy and quality encryption, as demonstrated in the following subsections.

5.1 Key space analysis

A good encryption algorithm must have a large key space enough to render brute-force attacks unfeasible. For the proposed image encryption algorithm, the key is given by the initial conditions x_{01}, \dots, x_{0k} , where each initial condition has a double precision. According to the IEEE floating-point standard [30], the computational precision of the 64-bit double-precision number is about 10^{15} ; thereby, the total space depends on the

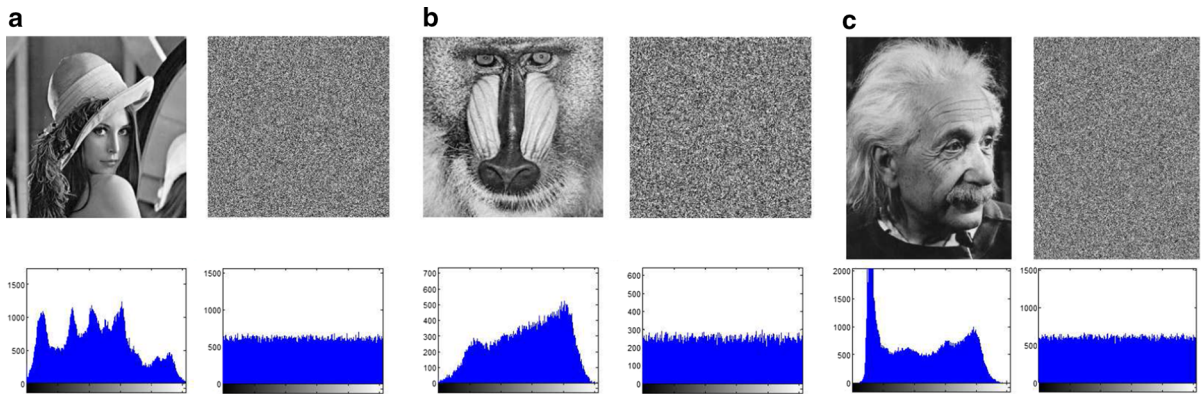


Fig. 9 Different plain images and encrypted-images with the proposed algorithm. **a** Lenna with its histogram for plain and encrypted image, **b** Baboon with its histogram for plain and encrypted image, **c** Einstein with its histogram for plain and encrypted image

number of initial conditions, for $k = 3$ the key space is:

$$\simeq 1 \times 10^{47} \simeq 2^{159}.$$

As the number k increases so does the key space but without loss of generality with $k = 3$, we assure that the key space is enough to resist brute-force attacks.

5.2 Correlation pixels

In general for any original image P , each pair of adjacent pixels is highly correlated in horizontal, vertical or diagonal direction; however, a cryptosystem should produce changes in all pixels and low correlation in adjacent pixels. To quantify and compare the correlations of adjacent pixels in the plain and cipher image, we use the correlation coefficient r_{xy} defined by

$$r_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}}, \tag{10}$$

$$E(x) = \frac{1}{\eta} \sum_{i=1}^{\eta} x_i, \tag{11}$$

$$D(x) = \frac{1}{\eta} \sum_{i=1}^{\eta} (x_i - E(x))^2, \tag{12}$$

$$cov(x, y) = \frac{1}{\eta} \sum_{i=1}^{\eta} (x_i - E(x))(y_i - E(y)), \tag{13}$$

where x and y denote two adjacent pixels and η is the total number of duplets (x, y) in this case $\eta = 2,000$. In Table 5 we show the results of three different images and it is possible to see that the adjacent pixels of

the original plain image (P) are highly correlated in any direction unlike the cipher image (C), the adjacent pixels are low correlated. This demonstrate the performance of the stream cipher algorithm.

5.3 Information entropy

As it is known, the entropy is a statistical measure of randomness that can be used to characterize the texture of the image and can be defined by [31]:

$$H(s) = - \sum_{i=0}^{2^n-1} Pr(s_i) \log_2 Pr(s_i), \tag{14}$$

where $n=8$ is the number of bits to represent a symbol $s_i \in s$ and $Pr(s_i)$ represents the probability of the symbol s_i then the entropy is expressed in bits, for a cipher gray-scale image with 256 levels, the entropy should ideally be $H(s) = 8$.

The entropies for plain images and cipher images using various k -modal maps are calculated and listed in Table 6. According to the Table 6, it is possible to see that when the number of modals is increased so does the entropy of each encrypted image.

5.4 Encryption quality

The encryption creates large changes in the value of pixels, these pixels should be completely different from the original image, these changes are irregular and more changes in the value of pixels show more effectiveness

Table 5 Correlation coefficients of two adjacent pixels

Image	Plain image	$k = 1$	$k = 2$	$k = 3$	$k = 4$
Lenna					
Vertical	0.9852	-0.0280	-0.0051	0.0226	-0.0050
Horizontal	0.9658	0.0252	-0.0113	0.0022	-0.0041
Diagonal	0.9506	0.0176	-0.0157	-0.0081	-0.0077
Baboon					
Vertical	0.9852	-0.0280	-0.0051	0.0226	-0.0050
Horizontal	0.9658	0.0252	-0.0113	0.0022	-0.0041
Diagonal	0.9506	0.0176	-0.0157	-0.0081	-0.0077
Einstein					
Vertical	0.9817	0.0091	-0.0245	-0.0164	-0.0001
Horizontal	0.9797	0.0165	-0.0095	0.0085	0.0050
Diagonal	0.9647	0.0136	0.0124	0.0038	0.0035

Table 6 Entropy for plain image and cipher image with $k = 1, 2, 3, 4$

Entropy	Plain image	$k = 1$	$k = 2$	$k = 3$	$k = 4$
Lenna	7.8059	7.9988	7.9988	7.9989	7.9989
Baboon	7.7091	7.9970	7.9970	7.9971	7.9973
Einstein	7.4913	7.9989	7.9989	7.9989	7.9989

Table 7 Encryption quality for cipher images with $k = 1, 2, 3, 4$

Encryption quality	Unimodal	Bimodal	Trimodal	Quamodal
Lenna	267.6016	268.2734	268.6094	269.1953
Baboon	116.2344	117.5859	117.9062	128.9297
Einstein	309.7734	311.2969	312.2500	312.6406

of encryption algorithm and thus better quality. The encryption quality represents the average number of changes to each gray level according to [32] and can be expressed as:

$$Q = \frac{\sum_{L=0}^{255} |H_L(C) - H_L(\bar{P})|}{256}, \tag{15}$$

where L is the gray levels of the images, $H_L(C)$ and $H_L(\bar{P})$ are the numbers of repetition from each gray value in the original and the encrypted image, respectively. The results of this test are shown in Table 7, where it is possible to see that if the k number of a multi-modal map increases then the encryption quality of the cipher image increases too.

5.5 Chosen-plaintext attack

One common weakness in many ciphers is when the algorithm encrypts an original image twice using the same set of keys the cipher images are always the same, this provides the opportunity to break the algorithm using the chosen-plaintext attack. In this kind of attack somewhat the attacker has temporary access to the encryption algorithm and he can choose the image to encrypt and obtains its corresponding cipher image then trying to find the equivalent key stream image.

Seeking to solve this problem we add one column of random values (noise) in the original image, with this each time, we encrypt the image, these values will be different, even if we use the same set of keys so we obtain a different cipher image each time. We encrypt an image twice using the same set of keys, and the results are shown in Fig. 10: (a) Plain image of Lenna, (b) encrypted image in the first time C_1 , (c) the encrypted image second time with the same set of keys C_2 , (d) the pixel difference $|C_1 - C_2|$ and the histogram for each image, respectively. This ensures that the proposed algorithm is able to resist the chosen-plaintext attack.

5.6 Differential attack

In the differential attack the opponent can usually perform a slight change just by modifying a pixel of the original image in order to observe the changes on the corresponding cipher image, trying to find a relation-

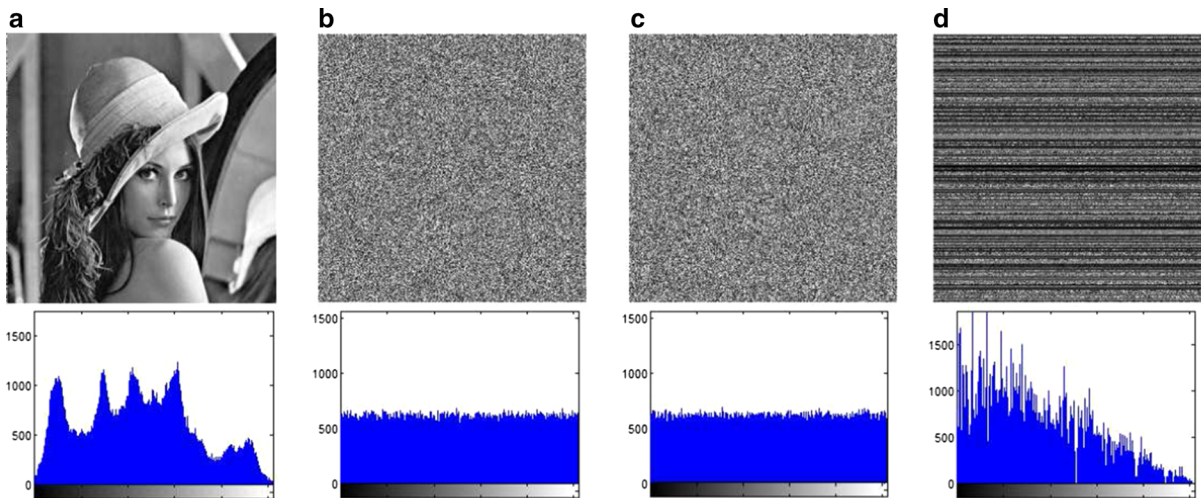


Fig. 10 **a** Plain Image of Lenna, **b** Encrypted Image of Lenna for first time C_1 , **c** Encrypted Image of Lenna for second time C_2 , **d** the pixel difference $|C_1 - C_2|$ and its histogram for each image, respectively

ship between the plain image and the cipher image. If one minor change in the plain image produces a significant change in the cipher image then this attack becomes very inefficient and practically useless. As we can see in the previous section each time we encrypt the image the result is different then without modifying the original image the result between the cipher images is different with this the differential attack is unfeasible but we are able to measure how different are the images when we encrypt twice. This can be measured by two criteria, the number of pixel change rate (NPCR) and the unified average changing intensity (UACI).

The NPCR measures the number of different pixels between two cipher images.

$$NPCR = \frac{\sum_{i,j} \delta_a(i, j)}{\nu} \times 100 \% \tag{16}$$

$$\delta_a(i, j) = \begin{cases} 0 & \text{if } C_1(i, j) = C_2(i, j); \\ 1 & \text{if } C_1(i, j) \neq C_2(i, j). \end{cases} \tag{17}$$

where ν is the total number of pixels in the image, C_1 and C_2 are the cipher images encrypted twice with the same set of keys.

The second criterion, UACI can be defined as:

$$UACI = \frac{1}{\nu} \left[\sum_{i,j} \frac{|C_1(i, j) - C_2(i, j)|}{2^8 - 1} \right] \tag{18}$$

Results of these tests are shown in Table 8. These results show that the algorithm produces complete different

Table 8 NPCR and UACI from an image encryption twice

	Proposed scheme
NPCR	99
UACI	34.8353

cipher images even using the same set of keys. Thus the algorithm can resist differential attack and chosen-plaintext attack.

6 Conclusion

In this work we present a Cryptographically Secure Pseudo-random Bit Generator based on discrete dynamical system of one-dimensional and multi-modal or k -modal map. A key stream was constructed by means of the combination of k sequences and were evaluated with the statistical suite of test from the NIST, satisfactory results were obtained which show that these sequences possess statistical properties like truly random sequences and show that the k -modal map is highly sensitive to initial conditions. Furthermore, we use the Pseudo-random Bit Generator like a key stream and show the impact of using multi-modal maps which are reflected in the encryption quality. Finally, a pre-processing in image was proposed by which the encryption process becomes probabilistic. In other words, in this approach it is achieved that an encrypted image

twice under the same set of keys, the encrypted image is different each time, while the process of decryption remains deterministic. However if the encryption function is modified as follows: $C_i = \bar{P}_i \oplus Z_i \oplus C_{i-1} \oplus C_{i-1} = \bar{P}_i \oplus Z_i$, if we consider the assumption that all values \bar{P}_i are zero, it is possible to retrieve the image with the exception of the $\bar{P}(i, 1)$ column, in summary this requires extra computation and the assumption that $\bar{P}_i = 0$ for all values of i . We are currently working to build a probabilistic encryption scheme in which this point would be infeasible.

Acknowledgments M. García-Martínez is doctoral fellow of the CONACYT in the Graduate Program on control and dynamical systems at DMAp-IPICYT. E. Campos-Cantón acknowledges the CONACYT financial support through project No. 181002.

References

- National Bureau of Standards: Data Encryption standard, Federal Information Processing Standards Publication 46. US Government Printing Office, Washington DC (1977)
- National Bureau of Standards: Data encryption standard modes of operation, Federal Information Processing Standards Publication 81. US Government Printing Office, Washington DC (1980)
- Menezes, A.J., Van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
- Knudsen, L.R., Robshaw, M.J.B.: The Block Cipher Companion. Springer, Berlin (2011)
- Paar, C., Pelzl, J.: Understanding Cryptography. Springer, Berlin (2010)
- Mengue, A.D., Essimbi, B.Z.: C: Secure communication using chaotic synchronization in mutually coupled semiconductor lasers. *Nonlinear Dyn.* **70**(2), 1241–1253 (2012)
- Li, X., Rakkiyappan, R.: Impulsive controller design for exponential synchronization of chaotic neural networks with mixed delays. *Commun. Nonlinear Sci. Numer. Simulat.* **18**(6), 1515–1523 (2013)
- Chen, Y., Fei, S., Zhang, K.: Stabilization of impulsive switched linear systems with saturated control input. *Nonlinear Dyn.* **69**(6), 793–804 (2012)
- Ontañón-García, L.J., Campos-Cantón, E., Femat, R., Campos-Cantón, I., Bonilla-Marín, M.: Multivalued synchronization by Poincaré coupling. *Commun. Nonlinear Sci. Numer. Simulat.* **18**(10), 2761–2768 (2013)
- Kanso, A., Ghebleh, M.: A novel image encryption algorithm based on a 3D chaotic map. *Commun. Nonlinear Sci. Numer. Simulat.* **17**(7), 2943–2959 (2012)
- Alvarez, G., Li, S.: Some cryptographic requirements for Chaos-based cryptosystems. *Int. J. Bifurc. Chaos* **16**(8), 2129–2151 (2006)
- Behnia, S., Akhshani, A., Akhavan, A., Mahmodi, H.: Application of tripled chaotic maps in cryptography. *Chaos Solitons Fractals* **40**(1), 505–519 (2009)
- Andreucot, M.: Logistic map as a random number generator. *Int. J. Modern Phys. B* **12**(9), 921–930 (1998)
- Xing-Yuan, Wang, Xie, Yi-Xin: A design of pseudo-random bit generator based on single chaotic system. *Int. J. Modern Phys. C* **23**(3), 1250024 (2012)
- Shujun, Li, Xuanqin, Mou, Yuanlong, Cai: Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography. *Prog. cryptol.—INDOCRYPT* **2247**, 316–329 (2001)
- Wang, X.Y., Yang, L.: Design of pseudo-random bit generator based on chaotic maps. *Int. J. Modern Phys. B* **26**(32), 1250208 (2012)
- Kanso, A., Smaoui, N.: Logistic chaotic maps for binary numbers generations. *Chaos Solitons Fractals* **40**(5), 2557–2568 (2009)
- García-Martínez, M., Campos-Cantón, E.: Pseudo-random bit generator based on lag time series. *Int. J. Modern Phys. C* **25**(4), 1350105 (2014)
- Franois, M., Grosge, T., Barchiesi, D., Erra, R.: Pseudo-random number generator based on mixing of three chaotic maps. *Commun. Nonlinear Sci. Numer. Simulat.* **19**(4), 887–895 (2014)
- Campos-Cantón, E., Femat, R., Pisarchik, A.N.: A family of multimodal dynamic maps. *Commun. Nonlinear Sci. Numer. Simulat.* **16**(9), 3457–3462 (2011)
- García-Martínez, M., Campos-Cantón, I., Campos-Cantón, E., Celikovský, S.: Difference map and its electronic circuit realization. *Nonlinear Dyn.* **74**(3), 819–830 (2013)
- Devaney, R.: An Introduction to Chaotic Dynamical Systems. Westview Press, Boulder (2003)
- Li, C., Chen, G.: Estimating the Lyapunov exponents of discrete systems. *Chaos* **14**(2), 343–346 (2004)
- Yang, C., Wu, C.Q., Zhang, P.: Estimation of Lyapunov exponents from a time series for n-dimensional state space using nonlinear mapping. *Nonlinear Dyn.* **69**(4), 1496–1507 (2012)
- Beker, H., Piper, F.: Cipher Systems: The Protection of Communications. Wiley, New York (1982)
- Gustafson, H., Dawson, E., Nielsen, L., Caelli, W.: A computer package for measuring the strength of encryption algorithms. *Comput. Secur.* **13**(8), 687–697 (1994)
- Marsaglia, G.: DIEHARD Statistical Tests: <http://www.stat.fsu.edu/pub/diehard/>
- A. Rukhin et al: A Statistical test suite for random and pseudo-random number generators for cryptographic applications, pp. 800–822. NIST special publication (2010)
- Biham, E., Shamir, A.: Differential cryptanalysis of the data encryption standard. Springer, Newyork (1993)
- IEEE Computer Society: IEEE Standard Binary Floating-Point Arithmetic, ANSI/IEEE std (1985)
- Shannon, C.: Communication theory of secrecy systems. *Syst. Tech. J.* **28**, 623 (1948)
- Ahmed, H.E.D.H., Kalash, H.M., Farag Allah, O.S.: Encryption quality analysis of the RC5 block cipher algorithm for digital images. *Opt. Eng.* **45**(10), 107003 (2006)