applied sciences

MDPI

Article

Function Composition from Sine Function and Skew Tent Map and Its Application to Pseudorandom Number Generators

Leonardo Palacios-Luengas [1,†], Ricardo Marcelín-Jiménez [1], Enrique Rodriguez-Colina [1], Michael Pascoe-Chalke [1], Omar Jiménez-Ramírez [2] and Rubén Vázquez-Medina [3,*,†]

1 Department of Electrical Engineering, Autonomous Metropolitan University (UAM), Iztapalapa, Mexico City 09340, Mexico; lpl@xanum.uam.mx (L.P.-L.); rmarcelin@izt.uam.mx (R.M.-J.); erod@xanum.uam.mx (E.R.-C.); mpascoe@xanum.uam.mx (M.P.-C.)
2 Instituto Politécnico Nacional, ESIME Culhuacan, Ciudad de México 04430, Mexico; ojimenezr@ipn.mx
3 Instituto Politécnico Nacional, CICATA, Querétaro 76090, Mexico
* Correspondence: ruvazquez@ipn.mx
† These authors contributed equally to this work.

Featured Application: A pseudorandom number generator (PRNG) emulates to a truly random number generator in an interest interval and, its pseudorandomness depends on the size of the initial conditions space and the sensitivity to these conditions. A PRNG can be implemented through diverse strategies; but in cryptography applications, a PRNG must produce aperiodic number sequences with high linear complexity and a statistical distribution close to the uniform distribution. An approach to implement PRNGs is based on chaotic maps because they have inherent features, such as their highly sensitive dependence on initial conditions and the control parameters, their topological transitivity, ergodicity, aperiodicity and pseudorandomness properties. These features fully match with the practical implementation requirements of the PRNGs. Therefore, we propose a function composition based on skew tent map (STM) and the sine function that can be an effective alternative to implement PRNGs with high computational complexity that overcome pseudorandomness test suites.

Abstract: In cryptography, the pseudorandom number sequences must have random appearance to be used in secure information systems. The skew tent map (STM) is an attractive map to produce pseudorandom sequences due to its easy implementation and the absence of stability islands when it is in chaotic behavior. Using the STM and sine function, we propose and analyze a function composition to propose a pseudorandom number generator (PRNG). In the analysis of the function composition, we use the bifurcation diagram and the Lyapunov exponent to perform a behavioral comparison against the STM. We show that the proposed function composition is more sensitive to initial conditions than the STM, and then it is a better option than the STM for cryptography applications. For the proposed function we determine and avoid the chaos annulling traps. The proposed PRNG can be configured to generate pseudorandom numbers of 8, 16 or 32 bits and it can be implemented on microcontrollers with different architectures. We evaluate the pseudorandomness of the proposed PRNG using the NIST SP 800-22 and TestU01 suites. Additionally, to evaluate its quality, we apply tests such as correlation coefficient, key sensitivity, statistical and entropy analysis, key space, linear complexity, and speed. Finally, we performed a comparison with similar PRNGs that produce pseudorandom sequences considering numbers of 8 and 32 bits. The results show that the proposed PRNG maintains its security regardless of the selected configuration. The proposed PRNG has five important features: easy implementation, configurable to produce number with 8, 16 or 32 bits, high processing speed, high linear complexity, and wide key space. These features are necessary for cryptographic systems.

Keywords: pseudorandom number generator; function composition; nonlinear dynamics and chaos; robust chaotic map

Citation: Palacios-Luengas, L.; Marcelín-Jiménez, R.; Rodriguez-Colina, E.; Pascoe-Chalke, M.; Jiménez-Ramírez, O.; Vázquez-Medina, R. Function Composition from Sine Function and Skew Tent Map and Its Application to Pseudorandom Number Generators. *Appl. Sci.* **2021**, *11*, 5769. https://doi.org/10.3390/app11135769

Academic Editor: Luigi Fortuna

Received: 27 March 2021
Accepted: 7 June 2021
Published: 22 June 2021

1. Introduction

Several works related to the PRNG design have been proposed; for example, there are strategies that implement PRNGs using linear feedback shift registers (LFSR) [1–5], while other strategies are based on block cipher [6], stream cipher [7], quantum walks [8], cellular automata [9,10], chaotic oscillators and artificial neural networks (ANN) [11], or chaotic maps [12–15]. There are also PRNG design approaches that combine several of the above strategies [16]. Considering this context, we focus our research on PRNGs based on chaotic maps.

Chaotic maps are iterated functions that use an initial seed to produce non-linear sequences of numbers; these sequences when translated into binary sequences can generate random-looking and highly unpredictable numbers to be used in cryptography. Chaotic maps have high sensitivity to initial conditions when operating with their parameters inside specific domains, which can be determined. In these parameter domains the chaotic maps can operate as pseudorandom or aperiodic systems, but outside those parameter domains they can operate as periodic systems or their trajectories may also escape to infinity [17]. It may also happen that several chaotic repellers coexist in the chaotic system [18], and the trajectories move chaotically for a while before escaping and reaching another chaotic repeller [19,20]. Under these considerations, the behavior option can be selected from their control parameters [14]. In addition, when the chaotic maps are used in cryptographic applications, several drawbacks become evident, such as range discontinuity and non-uniform statistical distribution of the generated number sequences, as well as the small seed space [21]. Despite this, we cannot forget that there is a natural application relationship between chaos and cryptography. The main features of the chaotic systems, such as the sensitive dependence on initial conditions and control parameters, ergodicity, size of the parameter space, and mixing property, can be related to the confusion and diffusion conditions that must be applied to information to be protected by using cryptographic systems [22]. Therefore, many cryptographic systems and modules have been proposed based on chaotic systems [22–33]. In particular, chaotic maps have been successfully applied in the implementation of PRNGs [14,15,31,34–42].

In this way, the chaotic maps have inherent features that fully match with the practical implementation requirements of the PRNGs. The first proposal for a PRNG based on chaotic maps was developed in 1982 by Oishi and Inoue [43]. Later, Gonzalez and Pino in 1999 generalized the logistic map and designed a random function [44]. Stojanovski et al. in 2001 analyzed the application of a piecewise-linear chaotic map as PRNG [45,46]; and in the same year, Li et al. [47] performed an analysis suggesting that a couple, $g(f(x))$, of two piecewise-linear chaotic maps $f(x)$ and $g(x)$ has perfect cryptographic properties if it satisfies four requirements when used to build high-security stream ciphers. The requirements defined by Li et al. [47] for one-dimensional chaotic maps are: (R1) Piecewise-linear chaotic maps should be surjective maps on a same interval (a, b), (R2) Piecewise-linear chaotic maps should be ergodic on (a, b) with unique invariant density functions, (R3) Invariant density functions of the piecewise-linear chaotic maps should be equal to each other, and (R4) Chaotic orbit produced by one the piecewise-linear chaotic maps should be asymptotically independent to the chaotic orbit produced by the other map when the length of the chaotic orbits tends to be infinite.

After the work developed by Li et al. [47], many other researchers proposed PRNGs based on chaotic maps using different approaches [12–15,31,34–42,48]. In this extensive variety of proffers, several implementations were identified with security disadvantages attributed to one or more of the following features: non-uniform statistical distribution [49], digital degradation [50] and predictability [51] of the produced number sequences, as well as the small-sized seed space of the chaotic map [21]. In this context, other authors have proposed alternative solutions to counteract the exposed disadvantages. For example, in 2016, Wang et al. [52] compared cryptographically useful properties of piecewise-linear maps (ergodicity, Lyapunov exponent and bifurcation) to properties of logistic map and, in order to overcome the disadvantages of the logistic map used in designs of chaos-based

ciphers, they proposed a PRNG based on the piecewise-logistic map. In that proffer, Wang et al. claim that their PRNG achieves a trade-off between efficiency and security. However, in 2019, Lambić [53] analyzed the security of PRNG based on the piecewise-logistic map showing that it can be violated by using brute-force and known output sequence attacks. And then, also in 2019, Wang et al. [49] proposed a four-dimensional chaotic model based on a piecewise-logistic map with coupled parameters, but they just tried to overcome the fact that the statistical distribution of the piecewise-logistic map is non-uniform. Another example showing a solution to overcome the security disadvantages revealed for PRNGs based on chaotic maps is the work proposed by Zhou et al. [54] in 2016. Zhou et al. proposed a secret key generation algorithm based in operations in the YUV color space that combines two secret keys to produce the initial conditions required in the chaotic maps used in the encryption processes. In the attempt to strengthen their encryption system against differential attacks, Zhou et al. used a cubic map and a wavelet map to produce pseudorandom number sequences. Although Zhou et al. considered these maps to be highly sensitive to initial conditions, they did not perform a formal sensitivity analysis. Also, in 2021, Shi and Deng [55] when studying the dynamical degradation of the two-dimensional Barker map they found that this chaotic map can have valuable properties when it is used in a PRNG. Another example showing the application of strategies to overcome security disadvantages in PRNGs based on chaotic maps is the work proposed by Murillo-Escobar et al. in 2017 [36]. Since under the premise that low-dimensional chaotic systems may become more used than high-dimensional chaotic systems to produce the pseudorandom key stream used for encryption purposes, Murillo-Escobar et al. [36] proposed a PRNG based on the pseudorandomly enhanced logistic map, claiming that the produced number sequences have excellent statistical properties to cryptography applications. Although Murillo-Escobar et al. specified that the parameter domain for pseudorandomly enhanced logistic map is limited to (3.999, 4.0), they scaled and discretized the output of the chaotic map by applying *mod 1* to it when 1×10^6 is the scaling factor. With this scaling factor, they intended a uniform statistical distribution of the generated number sequences. Also, although Murillo-Escobar et al. [36] claim to avoid weak keys in their PRNG, we emphasize that they did not identify which conditions cause the chaotic map to produce weak keys in order to avoid them. A last example related to overcome the security disadvantages of the PRNGs based on chaotic maps is the work proposed by Chen et al. in 2019 [50]. Chen et al. [50] proposed a method to counteract the dynamical degradation of the digital sequences produced by using a chaotic system when it is implemented on low–precision devices; in that condition, all the produced sequences could be periodic sequences. In this way, the method proposed by Chen et al. [50] was based on a dynamical strategy to perturb a digital chaotic system by using pseudorandom sequences produced by a two–dimensional sine chaotic map with control parameters a and b. They specify $a = 1$ and $b = 5$ so that the map has a chaotic behavior, but they do not perform an analysis of the opportunities that exist to generate chaos, nor of the chaos annulling conditions in the chaotic system. Additionally, Chen et al. [50] showed two experiments in order to test effectiveness of their method to counteract the dynamical degradation of digital chaotic sequence. In the first experiment they selected the logistic map to represent the one–dimensional chaotic maps. In the second experiment, they selected the two-dimensional logistic cascade hyperchaotic map to represent the high-dimensional chaotic maps. In this way, Chen et al. [50] demonstrated the effectiveness of their method considering the linear complexity, correlation, and statistical distribution.

Therefore, although efforts are being made to overcome the security disadvantages of implementing PRNGs based on chaotic maps, there are still PRNGs based on chaotic maps that have security shortcomings. For example, some chaotic maps have stability islands within the parameter domains for chaotic behavior, adversely affecting the system security, other chaotic maps produce number sequences with non-uniform statistical distribution, and other chaotic maps only work by using a limited size of initial conditions space [15,53]. Therefore, to safely use a PRNG based on chaotic maps, we must carefully select the

initial conditions ensuring that the map will always produce pseudorandom sequences with uniform statistical distribution and it will operate into the parameter domains for chaotic behavior avoiding the annulling chaos conditions; and when the chaotic system is implemented electronically, the dynamic degradation of the digital sequences must be considered.

Focusing specifically on PRNGs based on a single chaotic map, the most commonly used systems to generate pseudorandom number sequences are one-dimensional (1-D) chaotic maps, and although they have security disadvantages when used in cryptography, they are commonly used due to their structural simplicity, discrete nature, reduced number of arithmetic operations, high performance processing, and relatively easy implementation in hardware and software. It is worth noting that the 1-D chaotic maps can be attacked using the non-linear prediction method based on phase space reconstruction. In fact, in 1994 Short [51] proposed a method that can attack almost all 1-D chaotic maps and, therefore, many authors of works related to chaos–based PRNGs tend to conclude that it is more appropriate to use high–dimensional (H-D) chaotic systems rather than low–dimensional (L-D) chaotic systems to build PRNGs. It should be also noted that Short indicated in [51] that the details of their nonlinear prediction method is in a work submitted to the *Int. J. Bifurcations and Chaos* since 1993, but it was not published. Instead of that work, there is another work published in 1997 by Short [56] that applies the non-linear dynamic prediction to extract, in the time domain, faithful representations of hidden message signals transmitted by chaotic communication systems. Short's experiments are based on two fundamental facts. The first fact is that two systems (transmitter and receiver) implemented to reproduce the dynamic of a chaotic system can be synchronized without transmitting information related to their initial state. The second fact was that the ability of the receiver to synchronize with the transmitter is not affected by the addition of a low-powered message on the chaotic carrier. This means that, once synchronization is achieved, the chaotic carrier can be removed to reveal the message.

In this way, considering that H-D chaotic maps are difficult to implement, 1-D chaotic maps have been the most used in different applications [14,15,57], but in order to avoid their security weaknesses the following issues must be considered: (i) existence in the chaotic map of chaos annulling conditions, which are not identified and therefore are not avoided, (ii) a high degradation rate of the dynamic behavior when digital maps are used as quantization functions to approximate the true chaotic maps, (iii) low complexity of the chaotic map, (iv) strong correlation between the data set and the number sequences produced by the chaotic map, and (v) non-uniform statistical distribution of the number sequences produced by the chaotic map.

Thus, PRNGs based on a single chaotic system are potentially insecure systems since the produced number sequences expose information related to the initial condition of the chaotic system. In such case, an intruder can be able to decrease the computational complexity to find that initial condition. However, in order to avoid this condition PRNGs based on a single chaotic system, the following approaches should be used: higher finite precision [47,58], methods reducing the dynamical degradation of digital sequences [50], cascading multiple chaotic systems [47,59–61], combining chaotic maps by using modular operations [62,63], and coupled chaotic systems [64–66]. In this way, it is more difficult to obtain information about the initial condition of the system, since the number sequences it produces will be determined by different conditions, configurations, and mixed chaotic orbits.

Under these considerations, we propose and analyze a function composition (FC) that couple the sine function and skew tent map (STM) to include three FCs as core in a PRNG. In this way, we also propose a PRNG that uses three modular operations to increase the precision in the scaling and discretizing procedures used to translate the real number sequences produced by FCs to binary number sequences, and it uses a modular operation to combine the pseudorandom binary sequences. Through this strategy we overcome the disadvantages of using a single chaotic system. To guarantee the effectiveness of the FC during the operation of the proposed PRNG, we avoid in each FC the chaos

annulling conditions; and in order to evaluate the proposed PRNG, the following tests have been considered: correlation coefficient, key sensitivity, entropy analysis, statistical analysis, linear complexity, key space analysis, pseudorandomness, and speed analysis. It is important to emphasize that in this work, we use the word ***key* of the PRNG** to identify what other authors call the ***seed*** or ***initial condition*** of the PRNG.

The rest of the paper is organized as follows. Section 2 shows the definition, the sensitivity analysis, and a basic sensitivity test of the FC. Section 3 provides design details of the proposed PRNG. Section 4 shows the results of performance tests applied to the number sequences produced by the proposed PRNG. Finally, Section 6 is devoted to conclusions.

2. The Proposed Function Composition

2.1. Definition

Garasym et al. [66] proposed a PRNG based on coupled one–dimensional chaotic maps. They claim that a robust PRNG can be designed by coupling the tent and logistic maps, and the number sequences produced by that PRNG can achieve excellent pseudorandom properties and uniform statistical distribution. So, they conclude that their PRNG is suitable for chaos-based cryptography applications. Garasym et al. [66] based their proposal on the idea of combining the characteristics of the tent and logistics maps to achieve a new map with improved properties, through the combination of various network topologies. They proposed it because both logistic and tent maps have never been used in cryptography as they have weak security. Then, based on the review of the network topologies of 1–D chaotic maps presented by Garasym et al. [66], we propose a function composition (FC) from sine function and skew tent map (STM).

Thus, we define STM using the linear functions $\sigma_1(\mu, \alpha, y) = \frac{y}{\mu}$ and $\sigma_2(\mu, \alpha, y) = \frac{\alpha - y}{\alpha - \mu}$ according to Equation (1).

$$\sigma(\mu, \alpha, y) = \begin{cases} \alpha\, \sigma_1(\mu, \alpha, y) & 0 < y \leq \mu \\ \alpha\, \sigma_2(\mu, \alpha, y) & \mu < y < \alpha \end{cases}, \tag{1}$$

The iterated version of the STM is given by Equation (2),

$$y_n = \sigma^n(\mu, \alpha, y_0) = \begin{cases} \alpha\, \sigma_1(\mu, \alpha, y_{n-1}) & 0 < y_{n-1} \leq \mu \\ \alpha\, \sigma_2(\mu, \alpha, y_{n-1}) & \mu < y_{n-1} < \alpha \end{cases}, \tag{2}$$

Then, when the function $g(x) = sin(\pi x)$ is applied in a conjugate form to $\sigma_1(\mu, \alpha, y)$ and $\sigma_2(\mu, \alpha, y)$, in such a way that $\tau_1(\cdot) = g \circ \sigma_1(\cdot) = g[\sigma_1(\mu, \alpha, y)]$ and $\tau_2(\cdot) = g \circ \sigma_2(\cdot) = g[\sigma_2(\mu, \alpha, y)]$, we define the FC according to Equation (3).

$$\tau(\mu, \alpha, x) = \begin{cases} \alpha\, sin\left[\pi \frac{x}{\mu}\right] & 0 < x \leq \mu \\ \alpha\, sin\left[\pi \frac{\alpha - x}{\alpha - \mu}\right] & \mu < x < \alpha \end{cases}, \tag{3}$$

The iterated version of the FC is given by Equation (4),

$$x_n = \tau^n(\mu, \alpha, x_0) = \begin{cases} \alpha\, sin\left[\pi \frac{x_{n-1}}{\mu}\right] & 0 < x_{n-1} \leq \mu \\ \alpha\, sin\left[\pi \frac{\alpha - x_{n-1}}{\alpha - \mu}\right] & \mu < x_{n-1} < \alpha \end{cases}, \tag{4}$$

In both cases, n = 0, 1, 2, ... represents the iteration step, y_0 and $x_0 \in (0, \alpha)$ are the initial conditions of the chaotic maps, y_n and x_n are the number produced by the iteration n of each chaotic map, $\mu \in (0, \alpha)$ and $\alpha \in \mathbb{R}^+$ are the control parameters of the chaotic map, and $\tau^n(\mu, \alpha, x_0)$ represents Equation (3) applied n times on x_0 using μ and α.

Figure 1 shows the behavior of the STM and the FC according to Equations (1) and (3), respectively. It worth noting that when $\mu = 0.5\alpha$ the STM is symmetric. In this case, each chaotic system is applied to the interval (0, α), $\alpha = 3.0$, and $\mu = 0.0, 0.25\alpha, 0.50\alpha, 0.75\alpha, \alpha$.

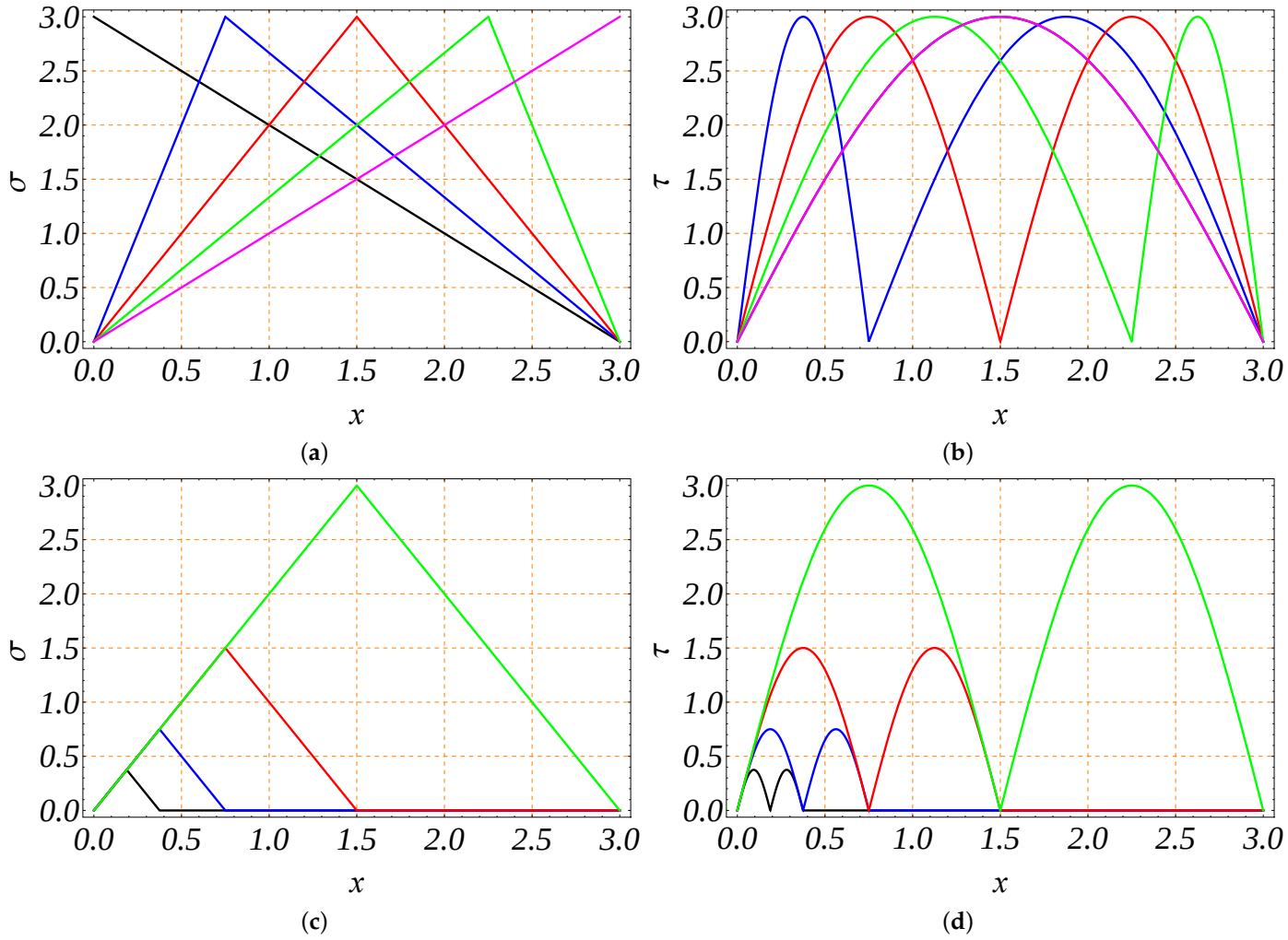

Figure 1. Chaotic maps are applied to $x \in (0, \alpha)$ with $\alpha = 3.0$. (**a**) STM and (**b**) FC with $\mu = 0$ (black), $\mu = 0.25\alpha$ (blue), $\mu = 0.50\alpha$ (red), $\mu = 0.75\alpha$ (green), and $\mu = \alpha$ (magenta). (**c**) STM and (**d**) FC with $\mu = 0.5\alpha$ and $\alpha = 0.375$ (black), $\alpha = 0.75$ (blue), $\alpha = 1.5$ (red), and $\alpha = 3.0$ (green).

2.2. Behavior Analysis

This section is aimed at showing the behavior analysis for the proposed chaotic maps. To such purpose, we identify the conditions for which the chaotic maps can generate periodic sequences, as well as those conditions for which they can generate aperiodic sequences. In order to offer this analysis of behavior (periodic or aperiodic), in a similar way to Palacios-Luengas et al. [14] and Pichardo-Méndez et al. [67], we identify the chaos annulling conditions in the proposed chaotic maps, and we calculate their bifurcation diagrams and Lyapunov exponents.

Firstly, to identify the chaos annulling conditions in the proposed chaotic maps, we must find their fixed points and their periodic orbit of order $m > 2$. For this intention, we assume that $x^{(1)}$ is a fixed point of the chaotic system $\xi(\cdot)$ when $\xi\left(\mu, \alpha, x^{(1)}\right) = x^{(1)}$, x^* is a preimage point when $x^{(1)} = \xi(\mu, \alpha, x^*)$, and $x^{(m)}$ is a condition that produces an periodic orbit of order $m > 2$ when $\xi^m\left(\mu, \alpha, x^{(m)}\right) = x^{(m)}$, considering that $\xi^m(\mu, \alpha, \hat{x})$ is m_{th} iteration of $\xi(\cdot)$ when $\hat{x}$ is its initial condition. As an example for the proposed chaotic maps, Table 1 shows the fixed points and some periodic orbits of order $m = 2, 3$, and 4. Note in Table 1 that the examples of conditions producing periodic orbit of order $m = 2$, 3 and 4 have be included by using only ten digits after the dot, but they are number with more significant digits. Additionally, note that Table 2 shows the preimage points.

Table 1. Fixed points and conditions that produce periodic orbits for the two chaotic maps.

STM	Conditions	FC	Conditions
$x^{(1)}=0$	$\alpha \neq \mu$	$x^{(1)}=0$	$0<\mu<\alpha$
$x^{(1)}=\frac{\alpha^2}{2\alpha-\mu}$	$\alpha>\mu$	$x^{(1)}=\mu$	$0<\mu<\alpha; \alpha<2\pi$
		$x^{(1)}=\alpha$	$0<\mu<\alpha; \alpha<2\pi$
$x^{(2)}=\frac{\alpha^2\mu}{\alpha^2+\mu\alpha-\mu^2}$	$0<\mu<\alpha; \alpha<2\pi$	$x^{(2)}=0.0265743377...$	$\mu=0.5\,;\alpha=3.0$
$x^{(2)}=\frac{\alpha^3}{\alpha^2+\mu\alpha-\mu^2}$	$0<\mu<\alpha; \alpha<2\pi$	$x^{(2)}=0.5071716543...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{2\alpha^2}{8\alpha^3+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.0396498406...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{2\alpha^2}{8\alpha^3-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.0400220305...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{4\alpha^3}{8\alpha^3+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.1121005595...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{4\alpha^3}{8\alpha^3-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.1298899309...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{2\alpha^2}{4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.2041531102...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{8\alpha^4-4\alpha^3+2\alpha^2}{8\alpha^3-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.2041531102...$	$\mu=0.5; \alpha=3.0$
$x^{(3)}=\frac{8\alpha^4}{8\alpha^3+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(3)}=0.4303378921...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{2\alpha^2}{16\alpha^4+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=0.0000750067...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{2\alpha^2}{16\alpha^4-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=0.1151873680...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{4\alpha^3}{16\alpha^4+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=0.3046575007...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{4\alpha^3}{16\alpha^4-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=0.4729887207...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{8\alpha^4-4\alpha^3+2\alpha^2}{16\alpha^4+8\alpha^3-12\alpha^2+6\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=0.6234955250...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{2\alpha^2}{4\alpha^2+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=1.3498296254...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{8\alpha^4}{16\alpha^4+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=1.5242944437...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{8\alpha^4}{16\alpha^4-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=1.7744661356...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{16\alpha^4-8\alpha^3+2\alpha^2}{16\alpha^4+8\alpha^3-12\alpha^2+6\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=1.9484472133...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{2\alpha^2}{4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.5475266813...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{16\alpha^5-8\alpha^4+4\alpha^3}{16\alpha^4+8\alpha^3-12\alpha^2+6\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.6286105510...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{4\alpha^3}{4\alpha^2+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.7855644262...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{16\alpha^5-4\alpha^3+2\alpha^2}{16\alpha^4+8\alpha^3-12\alpha^2+6\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.8315128133...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{16\alpha^5-4\alpha^3+2\alpha^2}{16\alpha^4-4\alpha^2+4\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.9105145253...$	$\mu=0.5; \alpha=3.0$
$x^{(4)}=\frac{16\alpha^5}{16\alpha^4+2\alpha-1}$	$0<\alpha<2\pi, \mu=0.5$	$x^{(4)}=2.9943872569...$	$\mu=0.5; \alpha=3.0$

Table 2. Some preimages of fixed points for the two chaotic maps considering $0<\mu<\alpha$ and $\alpha<2\pi$ and $k=1,2,3,....$

$x^p=\sigma^{-k}(\mu,\alpha,x^*)$	$x^p=\tau^{-k}(\mu,\alpha,x^*)$
$x\geq\alpha$ or $x\leq 0$	$\frac{\mu}{\pi}\arcsin\left(\frac{\mu}{\alpha}\right), k=1$
$\frac{\alpha\mu}{2\alpha-\mu}$	$\mu-\frac{\mu}{\pi}\arcsin\left(\frac{\mu}{\alpha}\right), k=1$
$\frac{\mu^2}{2\alpha-\mu}$	$\mu+\frac{\alpha-\mu}{\pi}\arcsin\left(\frac{\mu}{\alpha}\right), k=1$
$\frac{\mu^k}{\alpha^{k-2}(2\alpha-\mu)}$	$\alpha-\frac{\alpha-\mu}{\pi}\arcsin\left(\frac{\mu}{\alpha}\right), k=1$
$\frac{2\alpha^k-\alpha^{k-1}\mu-\alpha\mu^{k-1}+\mu^k}{\alpha^{k-2}(2\alpha-\mu)}$	$x=\frac{\mu}{2}, k=2$
$\frac{\mu^k(2\mu^2-2\alpha\mu-\mu^2)}{\alpha^k(2\alpha-\mu)}$	$\frac{\alpha+\mu}{2}, k=2$
$\frac{2\alpha^k-\alpha^{k-1}\mu-2\alpha^{k-4}\mu^{k-3}-4\alpha^{k-5}\mu^{k-2}-3\alpha\mu^{k-1}+\mu^k)}{\alpha^{k-2}(2\alpha-\mu)}$	$\frac{\mu}{\pi}\arcsin\left(\frac{2\alpha}{\mu}\right), k=2$
	$\mu-\frac{\mu}{\pi}\arcsin\left(\frac{2\alpha}{\mu}\right), k=2$
	$\mu+\frac{\alpha-\mu}{\pi}\arcsin\left(\frac{2\alpha}{\mu}\right), k=2$
	$\alpha-\frac{\mu-\alpha}{\pi}\arcsin\left(\frac{2\alpha}{\mu}\right), k=2$

On the other hand, in order to show the big picture of the statistical behavior of both chaotic systems, we show in Figure 2 the bifurcation diagrams for the STM, and in Figure 3

the bifurcation diagrams for the FC. Remember that, a bifurcation diagram illustrates the changes that occurred to the number sequences produced by a chaotic system considering different values of its control parameters. In Figures 2a and 3a, $\mu \in (0, 3.0)$ and $\alpha = 3.0$, meanwhile in Figures 2b and 3b $\mu = 0.5$ and $\alpha \in (0, 3.0)$, in Figures 2c and 3c $\mu = 1.5$ and $\alpha \in (0, 3.0)$, and in Figures 2d and 3d $\mu = 2.5$ and $\alpha \in (0, 3.0)$. In all cases, the initial conditions were randomly selected for each value of control parameter used when the chaotic system was being iterated.

According to Table 1, Figure 2 shows the fixed points (red lines) and some periodic orbits for the STM. For all cases, **A** corresponds to x=α, and **B** to $x = \frac{\alpha^2}{(2\alpha-\mu)}$. But in Figure 2b, **C** corresponds to $x = \frac{2\alpha^2}{(8\alpha^3-4\alpha^2+4\alpha-1)}$, and **D** to $x = \frac{2\alpha^2}{(16\alpha^4+2\alpha-1)}$; in Figure 2c, **C** corresponds to $x = \frac{8\alpha^4}{(8\alpha^3+18\alpha-27)}$, and **D** to $x = \frac{54\alpha^2}{(16\alpha^4+54\alpha-81)}$; and in Figure 2d, **C** corresponds to $x = \frac{50\alpha^2}{(8\alpha^3-20\alpha^2+100\alpha-125)}$, and **D** to $x = \frac{5.25968\times10^{14}\alpha^2}{(3.36619\times10^{13}\alpha^4+5.25968\times10^{14}\alpha-1.31492\times10^{15})}$.

On the other hand, Figure 3 shows the fixed points (red lines) to x=α and, additionally, Figure 3c indicates a stability island. The fixed points periodic orbits and stability islands must be identified and avoided when the chaotic system is applied in cryptosystems. Additionally, note that, the bifurcation diagrams in Figures 2a and 3a completely cover the plane μ *vs* $\tau(\mu, \alpha, x)$; but, the bifurcation diagram for the FC does not exhibit the annulling chaos conditions that are present on the STM, which are given by $x_0 = \mu$ or $x_0 = \alpha$, and $x_0 = 0$.

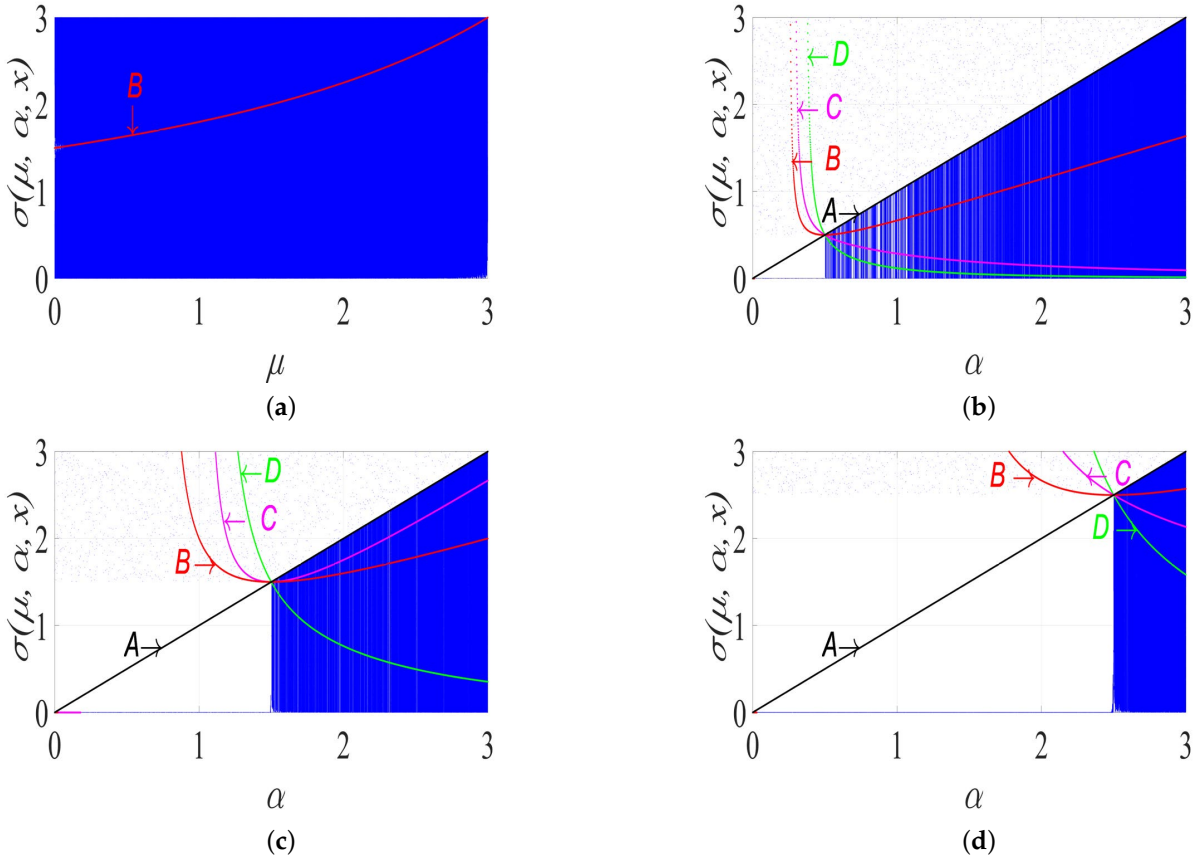

Figure 2. Bifurcation diagram of the STM: (**a**) $\mu \in (0, 3.0)$ and $\alpha = 3.0$, (**b**) $\alpha \in (0, 3.0)$ and $\mu = 0.5$, (**c**) $\alpha \in (0, 3.0)$ and $\mu = 1.5$, and (**d**) $\alpha \in (0, 3.0)$ and $\mu = 2.5$. Red and black lines show the fixed points, and green and magenta lines show the conditions for periodic orbits.

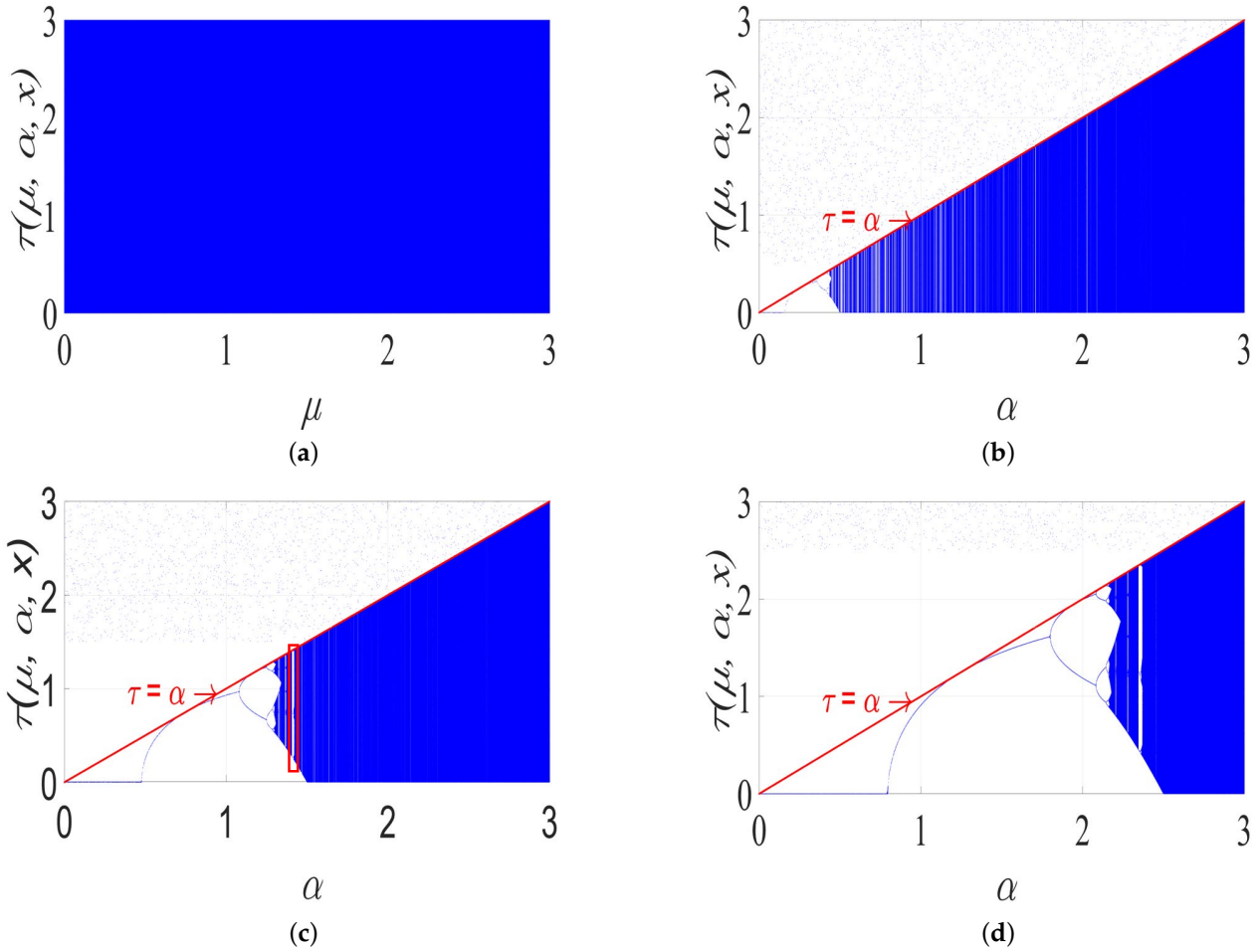

Figure 3. Bifurcation diagram of the FC: (**a**) $\mu \in (0, 3.0)$ and $\alpha = 3.0$, (**b**) $\alpha \in (0, 3.0)$ and $\mu = 0.5$, (**c**) $\alpha \in (0, 3.0)$ and $\mu = 1.5$, and (**d**) $\alpha \in (0, 3.0)$ and $\mu = 2.5$. Red lines show the fixed points of the chaotic system.

Now, in order to analyze the stability island of order 3 in the FC, we have generated Figure 4, which shows in detail the window for the stability island of order 3 identified at Figure 3. This window allows us to identify the auto-similarity property of the FC and, according to Sharkovski's Theorem [68,69], it ensures that the FC has an infinite number of stability islands. Therefore, in the FC the stability islands emerge according to Sharkovski sequence $2^n \times k$, with k = 3, 5, 7, 9, ... for n= 1, 2, 3, 4, ... [68,69], and they are inherited from the sine chaotic map. All stability islands must be considered and avoided. It is worth noting that the period doubling phenomenon observed in Figure 3c, boxed in red, appears again in Figure 4a. Figure 4b shows a zoom at region boxed in red at Figure 4a. Figure 4c,d have been included in order to show that the auto-similarity property, the doubling period phenomenon and the stability island of order 3 appear again. This is sufficiently clear evidence that the FC has an infinite number of stability islands, which emerge according to the Sharkovski sequence.

We highlight that, if μ is considered the main control parameter, both chaotic systems exhibit a chaotic behavior when $\mu \in (0, \alpha)$; meanwhile, if α is considered the main control parameter, they exhibit a chaotic behavior when $\alpha > \mu$, and this behavior is limited by the function $\tau = \alpha$. Additionally, the statistical distribution of the STM is closer to the uniform distribution than the statistical distribution of the FC, which is denser at ends. But, the FC does not have fixed points and stability islands at parameter domains of chaotic behavior when $\mu < \alpha$. In a similar way to Palacios-Luengas et al. [14] and Pichardo-Méndez et al. [67], the statistical distribution of the sequences produced by both chaotic maps can be estimated through the stationary statistical distribution by using the

Birkhoff's Ergodicity Theorem [70], and considering that the evolution of a set of initial condition must be studied when the chaotic system is applied to it.

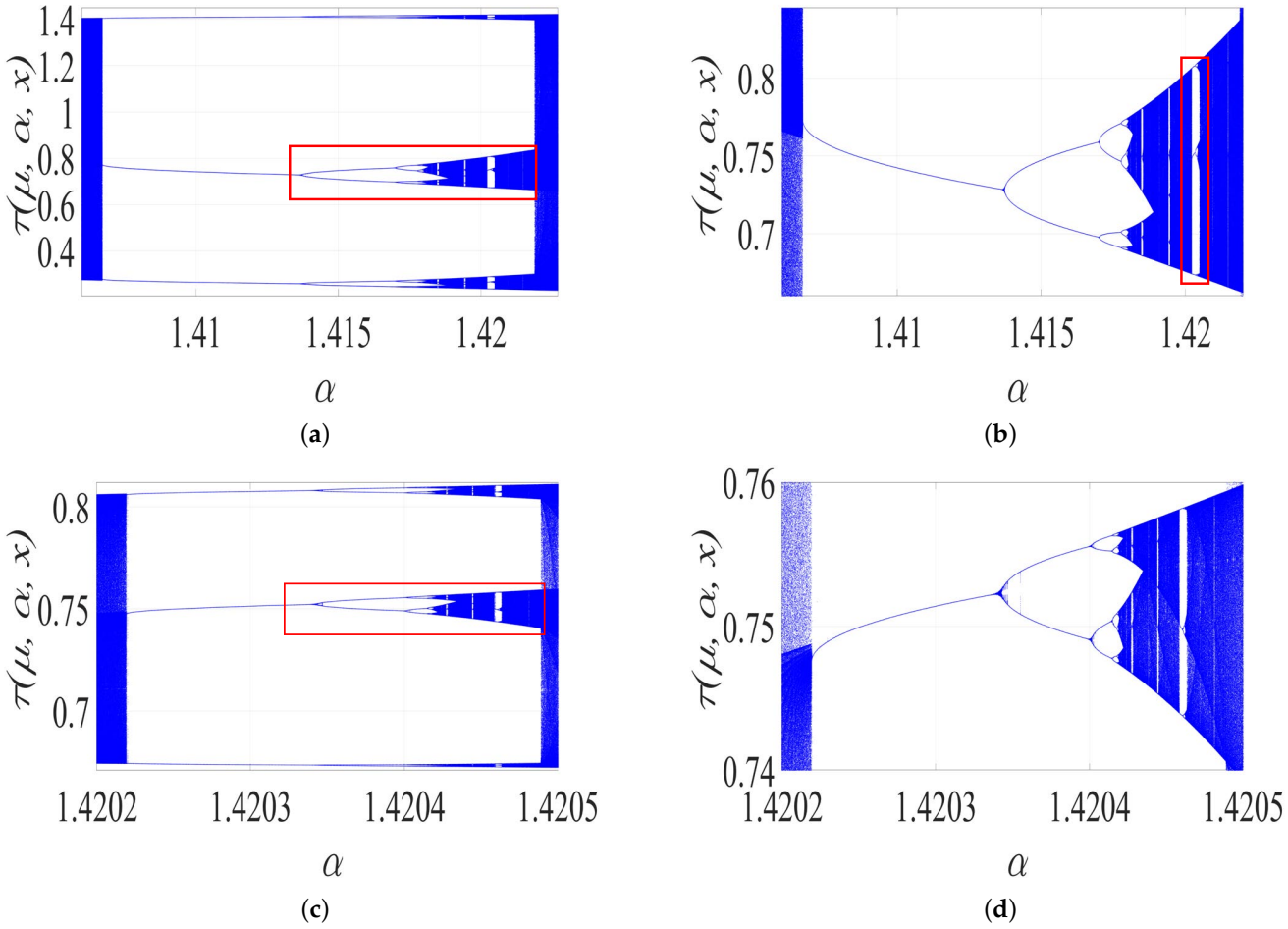

Figure 4. Bifurcation diagram of the FC when $\mu = 1.5$. (**a**) $\alpha \in (1.405, 1.425)$, it corresponds to a close-up on Figure 3c. (**b**) $\alpha \in (1.408, 1.422)$, it corresponds to a close-up on (**a**). (**c**) $\alpha \in (1.402, 1.4205)$, it corresponds to a close-up on (**b**). (**d**) $\alpha \in (1.402, 1.4205)$, it corresponds to a close-up on (**c**).

Now, as an example, in Figure 5 we show the trajectory diagrams of both chaotic systems when $\alpha = 3.0$ considering 10,000 iterations. For the STM, in Figure 5a, we use $\mu = 0.9$ and $x_0 = 1.9$, and, in Figure 5b, we show that a short trajectory reaches a fixed point when $\mu = 1.5$ and $x_0 = 0.0625$. On the other hand, for the FC, in Figure 5c, we show that a chaotic trajectory occurs when $\mu = 0.9$ and $x_0 = 1.9$, and, in Figure 5d, we show that a short trajectory reaches a fixed point when $\mu = 1.5$ and $x_0 = 1.460321868288294$.

The next step, in the behavior analysis of a chaotic system is to define whether the fixed points or the periodic orbits are stable (attractor) or unstable (repeller) points or orbits. Analyzing the stability of the fixed points, let $|\eta_n| = |\xi^n(\mu, \alpha, x_0 + \eta_0) - \xi^n(\mu, \alpha, x_0)|$ be the relative difference between the values of the position n in the number sequences produced by the chaotic system $\xi(\cdot)$, where $\xi(\cdot)$=$\sigma(\cdot)$ for the STM and $\xi(\cdot)$=$\tau(\cdot)$ for the FC. Using $x_0 + \eta_0$ and x_0 as initial conditions to produce two number sequences, let η_0 be some number arbitrarily small, and μ and α the control parameters of the chaotic systems. If $|\eta_{n+1}| < |\eta_n|$, then the selected control parameter will cause the chaotic map to converge $\forall\ n$, causing the produced number sequence to fall at some fixed point, which will be an attractor fixed point. Conversely, if $|\eta_{n+1}| > |\eta_n|$, then the selected control parameter will cause the chaotic map to diverge $\forall\ n$, causing the produced number sequence to move away from the fixed point, which will be a repeller fixed point. This derivative criterion for

repelling and attracting fixed points can be generalized to periodic orbits. For this purpose, we recommend reviewing Ref. [68].

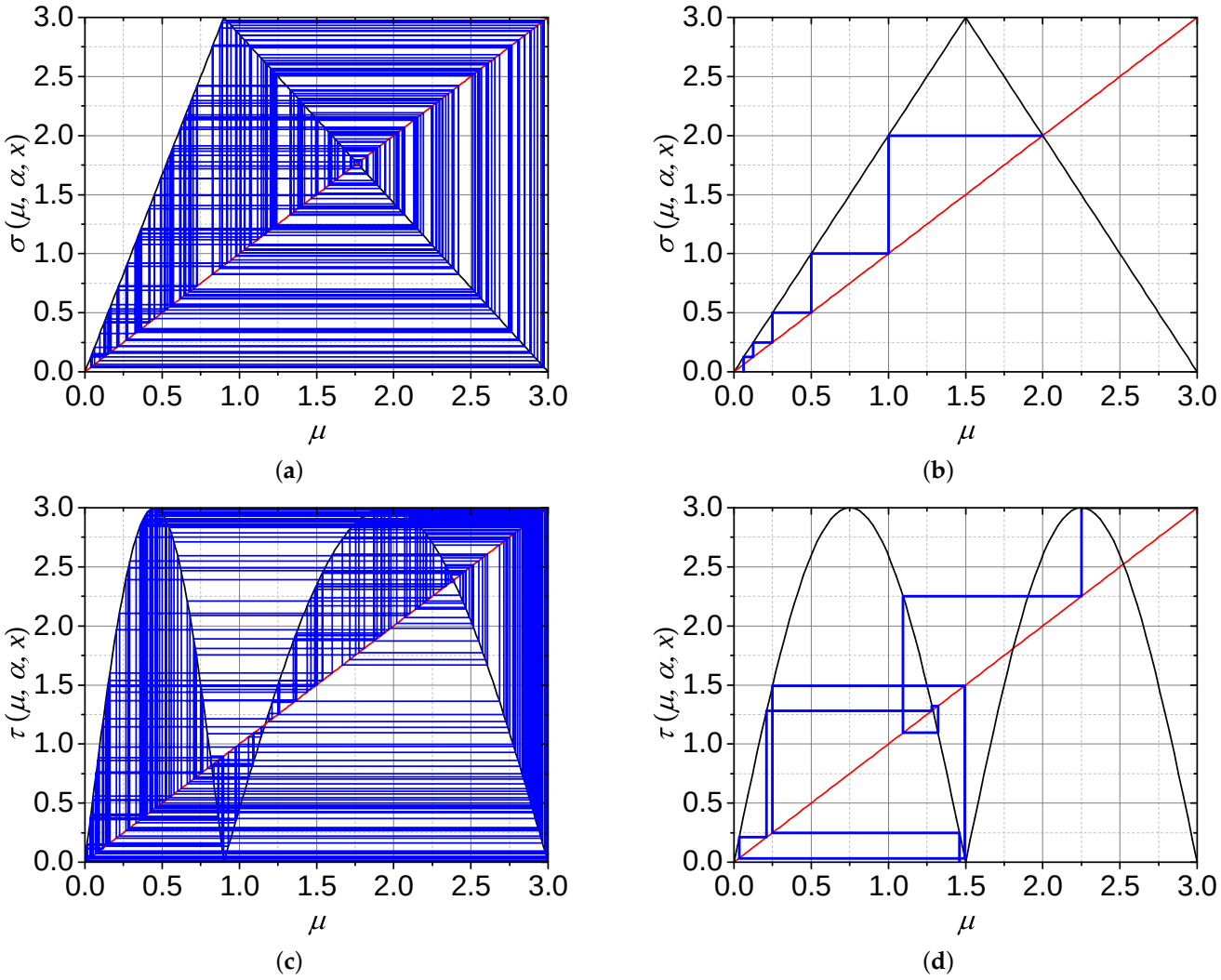

Figure 5. Trajectory diagrams. (**a**) STM when $\alpha = 3.0$, $\mu = 0.9$, and $x_0 = 1.9$ (chaotic trajectory), (**b**) STM when $\alpha = 3.0$, $\mu = 1.5$, and $x_0 = 0.0626$ (short trajectory reaches the fixed point $x^* = 2.0$), (**c**) FC when $\alpha = 3.0$, $\mu = 0.9$, and $x_0 = 1.9$ (chaotic trajectory), and (**d**) FC when $\alpha = 3.0$, $\mu = 1.5$, and $x_0 = 1.460321868288294$ (short trajectory reaches the fixed point $x^* = 3.0$).

Now, as an example of this concepts, we analyzed the cases when x_0 is a fixed point, in order to estimate the trap conditions. Thus, let x^* be a fixed point, and considering that $\tau^n(\mu, \alpha, x^*) = x^* \; \forall \; n$, the relative difference η_{n+1} can be written as,

$$\begin{aligned} |\eta_{n+1}| =& |\xi^n(\mu, \alpha, x^* + \eta_0) - \xi^n(\mu, \alpha, x^*)|, \\ =& |\xi^n(\mu, \alpha, x^* + \eta_0) - x^*|. \end{aligned} \tag{5}$$

By Taylor's series,

$$\begin{aligned} |\eta_{n+1}| &= \left| \xi^n(\mu,\alpha,x^*) - \eta_n \frac{d\xi(\mu,\alpha,x)}{dx}\bigg|_{x^*} - x^* \right|, \\ &= \left| \eta_n \frac{d\xi(\mu,\alpha,x)}{dx}\bigg|_{x^*} \right|, \\ &= |\eta_n \xi'(\mu,\alpha,x^*)|. \end{aligned} \tag{6}$$

Therefore, $\eta_{n+1} < \eta_n$ occurs when $\xi'(\mu,\alpha,x^*) < 1$ and x^* is a fixed attractor point, and $\eta_{n+1} > \eta_n$ occurs when $\xi'(\mu,\alpha,x^*) > 1$ and x^* is a repeller fixed point. Then, considering Table 1 for $\sigma(\mu,\alpha,\cdot)$, and according to Equation (7), $x^* = 0$ is a repeller point because $\sigma'(\mu,\alpha,x) > 1$ and $0 < \mu < \alpha$. We can verify this condition using Figure 5b, in which $\mu = 1.5$, $\alpha = 3.0$, and $x = 0.0625$, and the trajectory (number sequence) reaches the fixed point $x^* = 2.0$. In a similar way, according to Equation (7), $x^* = \frac{\alpha^2}{2\alpha-\mu}$ is a repeller point if $\sigma'(\mu,\alpha,x) > 1$, and this condition occurs when $1.0 < \alpha < 2.0$ and $2\alpha\text{-}\alpha^2 < \mu$, or when $2.0 < \alpha$. But it is an attractor point when $0.0 < \alpha < 1.0$, or when $1.0 < \alpha < 2.0$ and $0.0 < \mu < 2\alpha\text{-}\alpha^2$.

$$\sigma'(\mu,\alpha,x) = \begin{cases} \frac{\alpha}{\mu} & 0 < x \leq \mu \\ \frac{\alpha}{\mu-\alpha} & \mu < x < \alpha \end{cases}, \tag{7}$$

On the other hand, considering Table 1 for $\tau(\mu,\alpha,\cdot)$, and according to Equation (8), $x^* = 0.0$ is a repeller point when $0 < \alpha \leq \mu$ or when $\alpha < \mu + \pi\alpha cos\left(\frac{\pi(\mu-2\alpha)}{\alpha-\mu}\right)$, and it will be an attractor point when $\mu + \pi\alpha cos\left(\frac{\pi(\mu-2\alpha)}{\alpha-\mu}\right) < \alpha$. In a similar way, $x^* = \alpha$ will be a repeller point when $\pi\alpha cos\left(\frac{\pi\alpha}{\mu}\right) > \mu$, and it will be an attractor point when $\pi\alpha cos\left(\frac{\pi\alpha}{\mu}\right) < \mu$. Finally, $x^* = \mu$ is a repeller point because $\frac{\pi\alpha}{\mu}$ is always greater than 1.0.

$$\tau'(\mu,\alpha,x) = \begin{cases} \frac{\pi\alpha}{\mu} cos\left[\pi\frac{x}{\mu}\right] & 0 < x \leq \mu \\ \frac{\pi\alpha}{\alpha-\mu} cos\left[\pi\frac{x-2\alpha+\mu}{\alpha-\mu}\right] & \mu < x < \alpha \end{cases}, \tag{8}$$

In this way, a chaotic system will have a CAT condition when the sequences it produces reach an attractor fixed point. Therefore, it is very important to know and avoid fixed points in a chaotic system when it intends to be incorporated in cryptosystems.

In despite of the analysis performed so far, we must not forget that the mean value of the period of sequences generated by finite-state implementations of a chaotic map is influenced by the rounding error [71–73]. Therefore, and according to Li et al. [47] and Protopopescu et al. [58], it is highly recommended to use the highest precision available to represent real numbers and perform mathematical operations on devices and computers.

2.3. Sensitivity Analysis

A very effective way of determining the chaos annulling traps (CATs) in a chaotic system is by performing a sensitivity analysis. For this analysis, the Lyapunov exponent, λ, helps to detect a chaotic behavior in systems because it quantifies the separation rate of infinitesimally close trajectories in its phase space. In a similar way, to analyze the fixed points, λ is calculated considering that $|\eta_n| = \eta_0 e^{n\lambda}$. Note that if $\lambda > 0$, the two trajectories produced by the chaotic map will diverge when the separation of their initial conditions is arbitrary small. In this case, the map has a chaotic behavior, and in the case of $\lambda < 0$ the map will have a non-chaotic behavior. Although there are other approaches to calculate

the Lyapunov exponent such as using unstable periodic orbits [74], we decided to apply the following numerical approximation used by Palacios et al. [14].

$$\lambda \approx \frac{1}{n}\sum_{i=0}^{n-1} ln|\xi'(\mu,\alpha,x_i)|. \tag{9}$$

Note that, Equation (9) represents the maximum value of the velocity average, in exponential order, with which a first trajectory produced by a chaotic map moves away (or approaches) from other trajectories generated by the same map from an initial condition very close to the one used to produce the first trajectory. From Equation (9), and considering that for the STM, $\sigma'(\mu,\alpha,x_i)$ is defined by Equation (7), and for the FC, $\tau'(\mu,x_i)$ is defined by Equation (8), we have calculated λ for both chaotic maps and we showed in Figure 6 their behavior as a function of the control parameters, μ and α. Figure 6a shows λ_σ as a function of $\mu \in (0, \alpha)$ and $\alpha = 3.0$ for the STM. Note that $\lambda_\sigma > 0 \; \forall \; \mu$. In a similar way, Figure 6b shows λ_τ as a function of $\mu \in (0, \alpha)$ and $\alpha = 3.0$ for the FC. Also, note that $\lambda_\tau > 0 \; \forall \; \mu$.

Oppositely, Figure 6c shows λ_σ as a function of $\alpha \in (0, 3.0)$ and $\mu = 0.5$, 1.5 and 2.5 for the STM. Note that $\lambda_\sigma > 0$ when $\alpha > \mu$. In a similar way, Figure 6d shows λ_τ as a function of $\alpha \in (0, 3.0)$ and $\mu = 0.5$, 1.5, and 2.5 for the FC. Also, note that $\lambda_\tau > 0$ when $\alpha > \mu$. According to results showed in Figure 6d, we must emphasize that the FC exhibits islands of stability only when $\mu_{critical} < \alpha < \mu$ in the chaotic map; assuming that $\mu_{critical}$ is the value of μ when λ crosses zero for the first time. Thus, if $\alpha \in (0, 3.0)$ for the FC, it will always be true that $\lambda_\sigma > 0$ when $\alpha > \mu$ is satisfied avoiding that chaotic map generates chaos annulling conditions. On the other hand, if $\mu \in (0, 3.0)$ for the FC, it will always be true that $\lambda_\tau > 0$ when $\mu \in (0, \alpha)$. Therefore, when $\mu \in (0, \alpha)$, the FC will produce chaotic sequences.

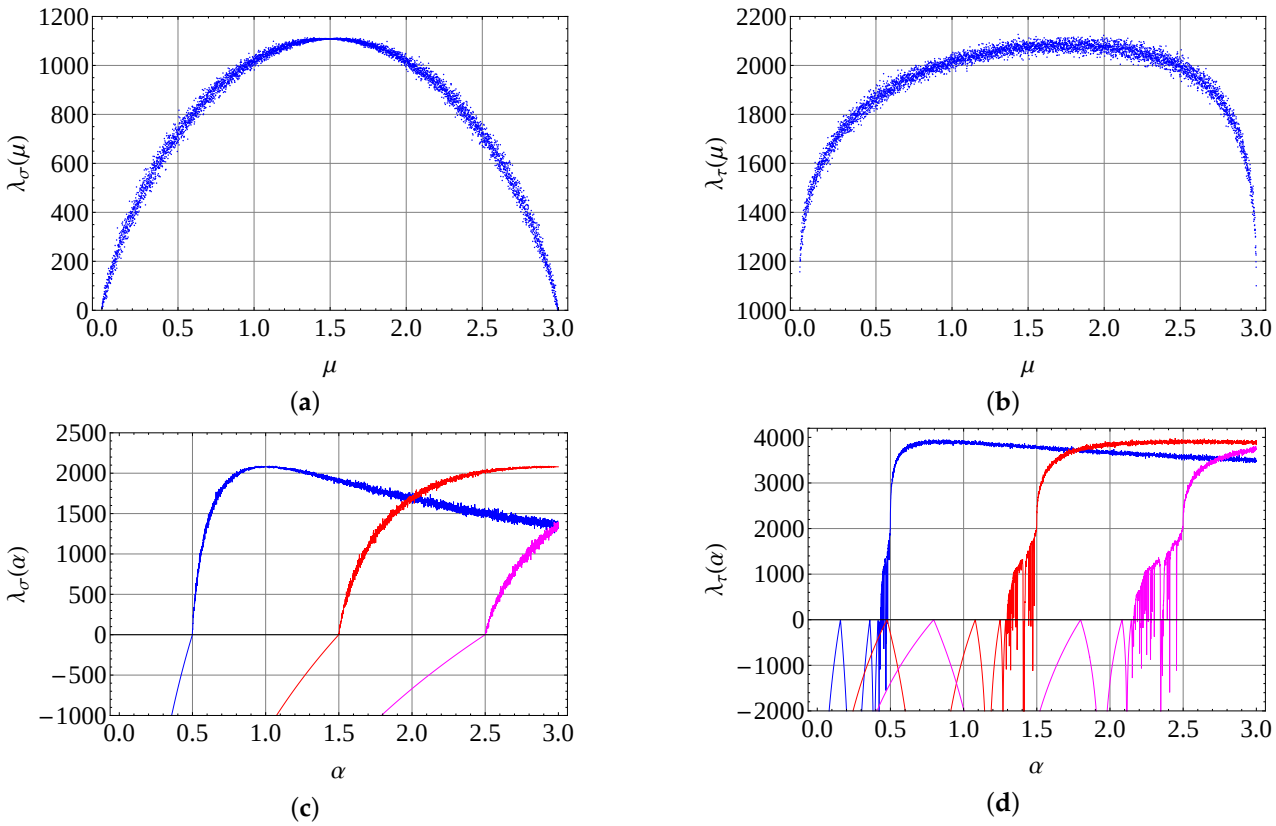

Figure 6. (**a**) λ_σ for the STM when $\alpha = 3.0$ and $\mu \in (0, \alpha)$, (**b**) λ_τ for the FC when $\alpha = 3.0$ and $\mu \in (0, \alpha)$, (**c**) λ_σ for the STM when $\alpha \in (0, 3.0)$ and $\mu = 0.3$ (blue line), $\mu = 1.5$ (red line), and $\mu = 2.5$ (magenta line), and (**d**) λ_τ for the FC when $\alpha \in (0, 3.0)$ and $\mu = 0.3$ (blue line), $\mu = 1.5$ (red line), and $\mu = 2.5$ (magenta line).

2.4. Sensitivity Test

In the first instance, the behavior of chaotic PRNGs can be predicted since they are deterministic systems and it is necessary to determine the conditions and limitations that allow such a prediction to be made. In order to address this, it is worth noting that chaotic PRNGs are implemented using dynamic systems with high dependence on initial conditions. Therefore, small variations in the initial conditions can imply, in the short term, great differences in the future behavior of the dynamic system. This feature limits the prediction of the system's behavior even in the short term. Thus, considering that the chaotic PRNGs are deterministic systems, their behavior can be completely determined if their initial conditions are known exactly. In this way, the sensitivity test helps to estimate how quickly the system's behavior changes when the initial condition changes by an arbitrarily small number; in this case the initial condition can vary by at least 1×10^{-15} and until 1×10^{-1}. In this sense, the main intention in designing a chaotic PRNG should be that the underlying chaotic map has the highest possible level of sensitivity, even for arbitrarily small initial conditions. This is true for the proposed chaotic map when compared to the STM.

In order to explain this condition, and considering the iterated functions expressed by Equations (2) and (4), both chaotic systems produce sequences whose behavior highly depends on initial condition x_0 and the control parameter μ. In both maps, if x_0 or μ are changed, the number sequences produced by them will also change. But the question that arises now is, which of two maps is more sensitive to initial conditions? A first answer to this question is given in Figure 7, which shows the temporal behavior for five number sequences produced by each chaotic map, considering that these sequences start with near initial conditions. That is, $x_0 = 0.5$ and $x_0^{'} = x_0 + \epsilon_0$, assuming that $\epsilon_0 = 1 \times 10^{-k}$ is an arbitrarily small number in $\mathbb{R}$, where $k = 1, 2, 3, 4, 5$. In both maps, we use $\alpha = 3.0$ and $\mu = 2.0$ as control parameters.

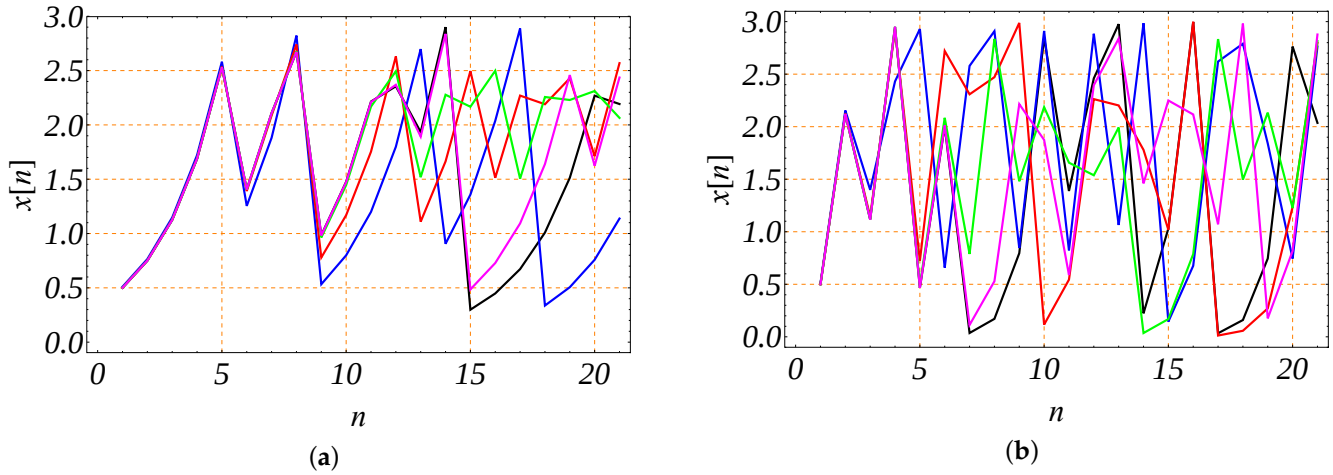

Figure 7. Sequences produced by both chaotic maps when $x_0 = 0.5$, $\alpha = 3.0$, and $\mu = 2$, with $\epsilon_0 = 0.0$ (black), $\epsilon_0 = 1 \times 10^{-2}$ (blue), $\epsilon_0 = 1 \times 10^{-3}$ (red), $\epsilon_0 = 1 \times 10^{-4}$ (green), and $\epsilon_0 = 1 \times 10^{-5}$ (magenta), (**a**) STM and (**b**) FC.

Note in Figure 7 that the chaotic sequences produced by using the STM are very close to each other until eighth iteration, and in later iterations they are notoriously separated. Instead, the sequences produced by using the FC are separated from second iteration. Note that this high sensitivity becomes more evident as k is decreased.

In order to obtain a sensitivity measure of a chaotic map to initial conditions, we define the tolerance level, N_{th}, that the chaotic map reaches when the initial condition changes from x_0 to $x_0' = x_0 + \epsilon_0$, considering a small threshold, δ, arbitrarily selected. Note that x_n' is the n-th element in the sequence from x_0', x_n is the n-th element in the sequence from x_0, and $\epsilon_0 = 1 \times 10^{-k}$ with $k \in [1, 15]$. In this case, N_{th} is the iteration number for which both sequences are separated by more than δ assuring that $\epsilon_{N_{th}} > \delta$, when δ is an arbitrarily

selected small number. Therefore, N_{th} is the tolerance level of the chaotic map as a function of k, where k is the smallness of ϵ_0. In summary, then, Figure 8a shows N_{th} versus k for the FC (blue lines) and the STM (red lines), considering that $k \in [1, 15]$, $x_0 = 0.1$, $\mu = 1.0$, $\alpha = 2.0$, and $n = 1000$ with $\delta = 1 \times 10^{-3}$ ("■") and $\delta = 1 \times 10^{-5}$ ("▲"). Note that the FC has a smaller tolerance level than the STM for changes in the initial conditions, because when using the same k, in both chaotic maps, N_{th} for the FC is smaller than N_{th} for the STM.

Now, using these concepts, we define the sensitivity level according to Equation (10). In Figure 8b, we show the behavior of $L(\delta, k)$ for both chaotic maps when $k \in [1, 15]$ with $\delta = 1 \times 10^{-3}$ and $\delta = 1 \times 10^{-5}$.

$$L(\delta, k) = \frac{1}{N_{th}(\delta, k)} \, . \tag{10}$$

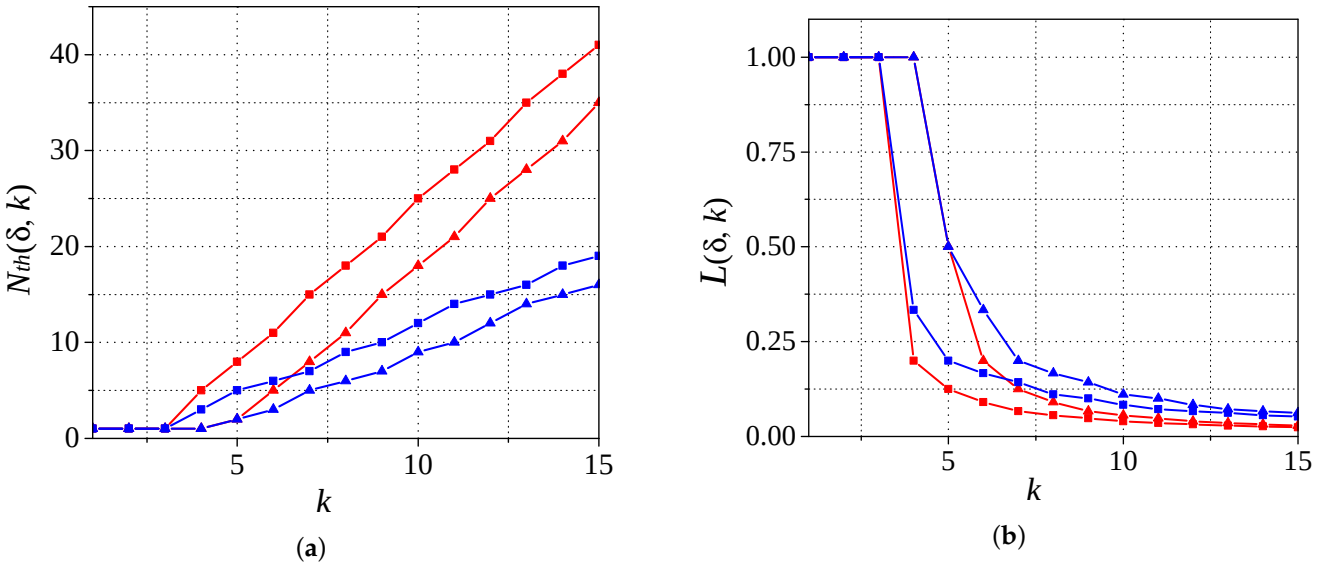

Figure 8. Sensitivity to initial conditions for the FC and the STM, (**a**) $N_{th}(\delta, k)$ and (**b**) $L(\delta, k)$ versus k, considering $k \in [1, 15]$, $x_0 = 0.1$, $\mu = 1.0$, $\alpha = 2.0$, and $n = 1000$ with $\delta = 1 \times 10^{-3}$ ("■") and $\delta = 1 \times 10^{-5}$ ("▲").

On the other hand, according to the work developed in 2019 by Liu and Feng [75], we apply the sensitivity test to both chaotic maps, and we calculate the sensitivity index, S_n, defined by Equation (11) when two sequences produced by the chaotic map have length n and their initial conditions are different by ϵ_0.

$$S(n, k) = \frac{1}{n} \sum_{i=1}^{n} \epsilon_i(k) \, . \tag{11}$$

In Figure 9a,b, we show the behavior of $S_{n,k}$ for the STM and the FC, respectively, when $k = 1$, 5, 10, and 15 and $n \in [1, 5000]$. Note that, in the long term, the FC is more sensitive to initial conditions that the STM, and after $n = 2500$ both maps tend to a constant value for $S(n, k)$. According to Liu and Feng [75], greater the value of $S_{n,k}$, the stronger the sensitivity.

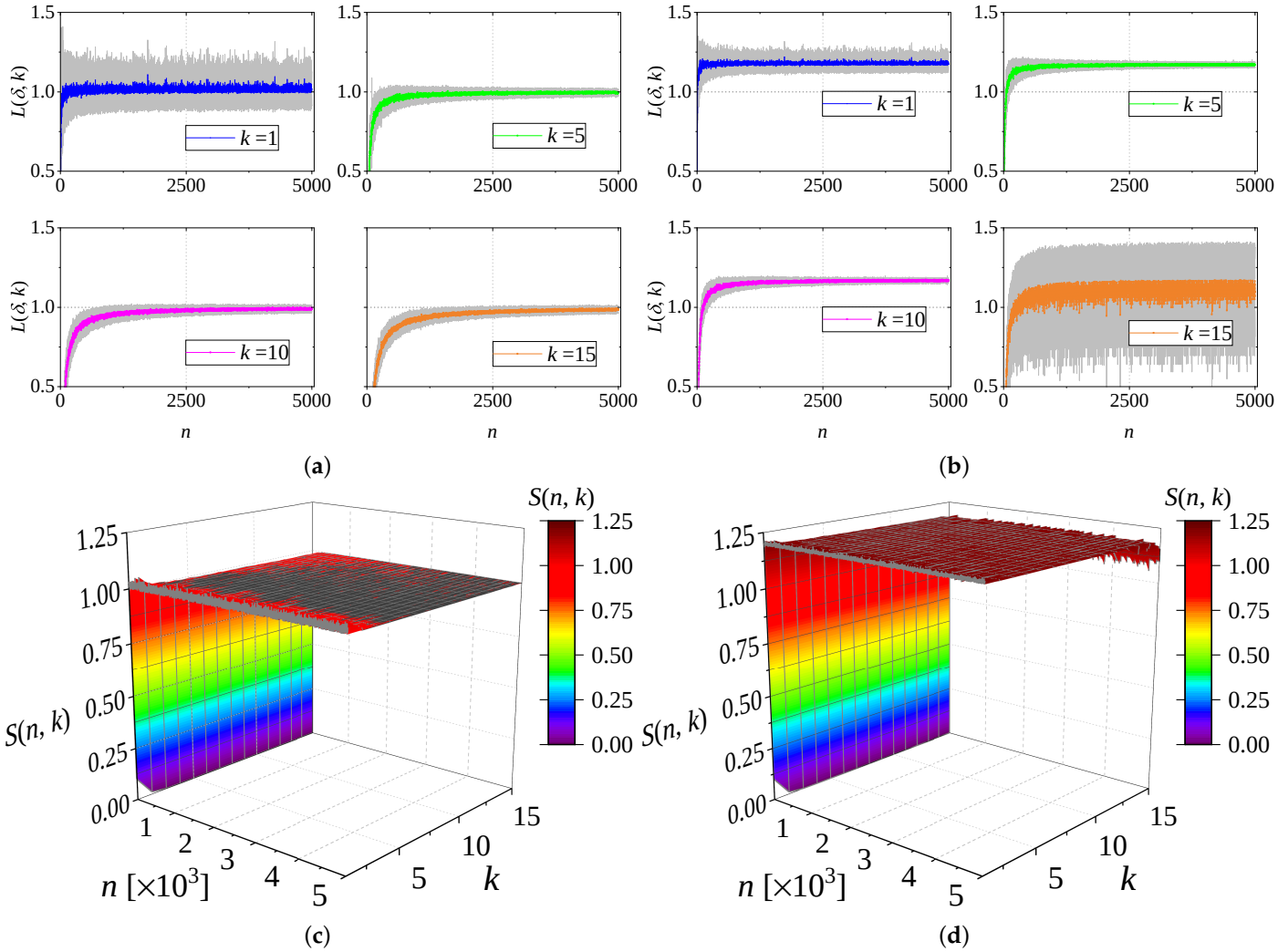

Figure 9. (**a**) Sensitivity level for the STM, considering 100 randomly selected initial conditions; (**b**) Sensitivity level for the FC, considering 100 randomly selected initial conditions; (**c**) Sensitivity index for the STM and (**d**) Sensitivity index for the FC. In all cases we consider $k \in [1, 15]$, $x_0 = 0.1$, $\mu = 1.0$, $\alpha = 2.0$, and $n \in [1, 5000]$.

2.5. Remarks

From the preliminary sensitivity tests, we highlight that the FC has a lower tolerance level and a higher sensitivity to the initial conditions than the STM. From the analysis of the chaos annulling conditions and the sensitivity analysis, we highlight that the FC has less chaos annulling conditions than the STM, and like the STM it does not have stability islands once it enters the chaotic behavior. Thus, the FC is an excellent option for the implementation of PRNGs.

3. The Proposed PRNG

Considering the approaches necessary to increase the complexity and to avoid the insecurity conditions of the PRNGs based on single chaotic maps, and assuming that the proposed PRNG is implemented in a computer, we use the highest finite precision [47,58], cascading chaotic maps [47,59–61], combining chaotic maps by using modular operation [62,63], and using a function composition from chaotic maps [64–66]. It is worth noting that, considering recent technological advances, it would be interesting to address the possibility that the PRNGs based on chaotic maps can be implemented in microfluidic lab-on-a-chip devices [76–79]. The microfluidic technology is characterized by its advantages of miniaturization, integration and automation. It has enabled the development

of universal computing based on two-phase microfluidics, and it is named *bubble logic* because the bubbles in a microfluidic device can carry process control information similar to what happens in a microprocessor, while performing chemical reactions [80–82].

Resuming the strategies mentioned to increase the complexity and to avoid the insecurity conditions of the PRNGs based on single chaotic maps, the cryptanalysis will be more difficult for the proposed PRNG, since the output sequences will be determined by many different mixed chaotic orbits. We emphasize that all mathematical operations included in the proposed PRNG have been performed considering double precision arithmetic and floating-point representation for the real numbers. In addition, we do not apply scaling or discretization processes to the functions used, rather we use them in their original form by using double precision arithmetic for the calculations. Thus, the final output of the proposed PRNG converted to 8-bit, 16-bit, and 32-bit integers, depending on the configuration used. It is worth noting that with a computer and any arithmetic, we can not produce chaos; the use of a computer leads to the degradation of the chaotic dynamics [83].

Thus, the proposed PRNG includes three chaotic maps produced by the function composition from the sine function and the skew tent map. It consists of three blocks: (i) **RCMb**- Block of the robust chaotic maps, which includes three FCs, each one using different values for μ, α and x_0. **RCMb** receives the key K of the PRNG as input, and it produces three output sequences; in this case, K is a word constituted by the concatenation of μ_i and α_i with $i = 1$, 2 and 3, and the values for the initial conditions x_0, y_0, and z_0; and each one of the three output sequences is a chaotic sequence of real numbers produced by each chaotic map. (ii) **Tb**- Block to translate real number sequences into integer number sequences, and it includes three functions with a single input and a single output. Finally, (iii) **MSb**- Block sum module 2^{bits}, which has three inputs and a single output that represents the output of the proposed PRNG, where *bits* can be 8, 16 or 32. As previously expressed, and considering Figure 10, we define the following steps to generate a pseudorandom number sequence with uniform distribution and good statistical properties.

1. Assuming that **RCMb** includes three FCs, from k, we produce three pseudorandom sequences of real numbers: $\hat{x} = \{\hat{x}_j\}$, $\hat{y} = \{\hat{y}_j\}$ and $\hat{z}$=$\{\hat{z}_j\}$, with $j = 1, 2, 3, ...$ Note in Figure 10 that K is constituted from the concatenation of μ_i and α_i with $i = 1$, 2 and 3, and the values for the initial conditions x_0, y_0, and z_0 required in the chaotic maps.
2. In **RCMb**, from $\hat{x}$, $\hat{y}$ and $\hat{z}$, three new pseudorandom sequences are produced and, for this, in each FC, the results of 1000 iterations are discarded to eliminate the transient values produced in the beginning by the chaotic maps. In this way, the final chaotic sequences are $x = \{x_0 = \hat{x}_0, x_i = \hat{x}_{i+30}\}$, $y = \{y_0 = \hat{y}_0, y_i = \hat{y}_{i+30}\}$ and $z = \{z_0 = \hat{z}_0, z_i = \hat{z}_{i+30}\}$, respectively, where $i = 1, 2, 3, ...$
3. Using **Tb**, the pseudorandom sequences $x = \{x_i\}$, $y = \{y_i\}$, and $z = \{z_i\}$ are translated from domain of real numbers to domain integer numbers of 8, 16 or 32 bits, producing $X = \{X_i\}$, $Y = \{Y_i\}$ and $Z = \{Z_i\}$, respectively. This action is performed by using $X_i = mod(x_i \cdot 10^u, 2^{bits})$, where X_i is i-*th* integer number of X, and it is produced from x_i, which is i-*th* real number of x; in this case, we considerate that $bits = 8, 16$ or 32 and u = 14.
4. By using $S_i = mod(X_i + Y_i + Z_i, 2^{bits})$, from X, Y, and Z, **MSb** produces the pseudorandom sequence, $S = \{S_i\}$, of integer numbers with 8, 16 or 32 bits.

   Note in step 3, that *bits* influences on the range for X_i; that is, $X_i \in (0, 2^{bits})$ and 10^u is a scaling factor that translates the real numbers $x_i \in (0, \alpha)$ to real numbers in $(0, 10^u)$. Therefore, considering that $\alpha \ll 10^u$, the *mod* function redistributes on the interval $(0, 2^{bits})$ the new sequence of numbers that had been rescaled from the sequence of numbers x_i to $(0, 10^u)$.

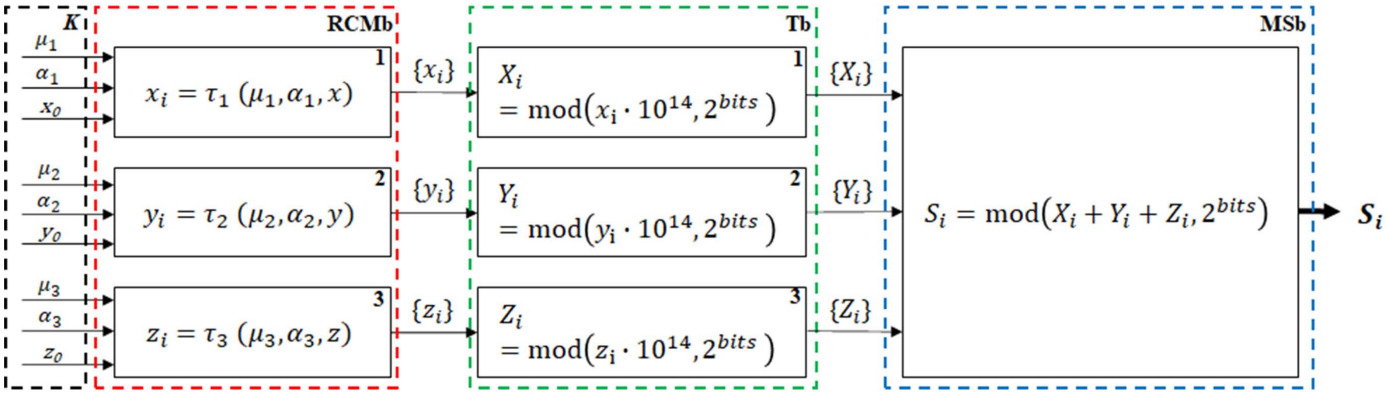

Figure 10. Block diagram of the proposed PRNG.

4. Performance Tests of the Proposed PRNG

In this section, we apply different tests for the pseudorandom sequences generated by the proposed PRNG. For each evaluation, we select a key set, $\hat{K} = \{K_1, K_2, ..., K_{1000}\}$, required in the PRNG. Then using $\hat{K}$, we generate the pseudorandom sequences S^t with $t = 1, 2, ..., 1000$, where each sequence consist of 8, 16 or 32-bit numbers. The tests carried out for the proposed PRNG are as follows: correlation coefficient, key sensitivity, entropy, statistical analysis, randomness, and linear complexity. In addition, the estimation of the keys space was made, and the execution speed was calculated. In all performance tests applied to the proposed PRNG, according to Sections 2.2 and 2.3, we have selected the parameters that avoid the annulling conditions of chaos and confirm that the Lyapunov exponent is positive. Also, Section 5 is included showing the results obtained with the proposed PRNG against similar algorithms based on chaotic maps.

4.1. Correlation Coefficient

We use the correlation coefficient, $r_{p,q}$, to determine the dependence degree and the statistical relationship between two pseudorandom sequences produced by the proposed PRNG. In this case, $r_{p,q}$ is defined as follows:

$$r_{p,q} = \frac{\sum_{i=1}^{n}(S_i^p - m_p)(S_i^q - m_q)}{\left(\sum_{i=1}^{n}(S_i^p - m_p)^2 \sum_{i=1}^{n}(S_i^q - m_q)^2\right)^{1/2}} \tag{12}$$

where S_i^p and S_i^q are the i-th element of the pseudorandom sequence S^p and S^q, respectively, m_p and m_q are the mean of S^p, and S^q, respectively, and p and $q = 1, 2, 3, ..., 1000$.

In this test, each S^t has a length of 1,000,000 6-bit numbers and for each one of them the values K_i, with $i = 1, 2, 3, ..., 1000$, were chosen pseudorandomly. Figure 11 shows the statistical distribution for the different correlation coefficients was obtained when $p \neq q$, and both can be 1, 2, 3, ..., 1000. According to numerical measure of r if two pseudorandom sequences are close to -1 or 1, then these sequences are very similar. Conversely, if the correlation is 0, then the sequences are not equal. Consequently, it is necessary that the values of r are too close to 0. Note that the values of correlation coefficient are distributed in $[-0.0045, 0.0034]$ with mean -5.4976×10^{-4}.

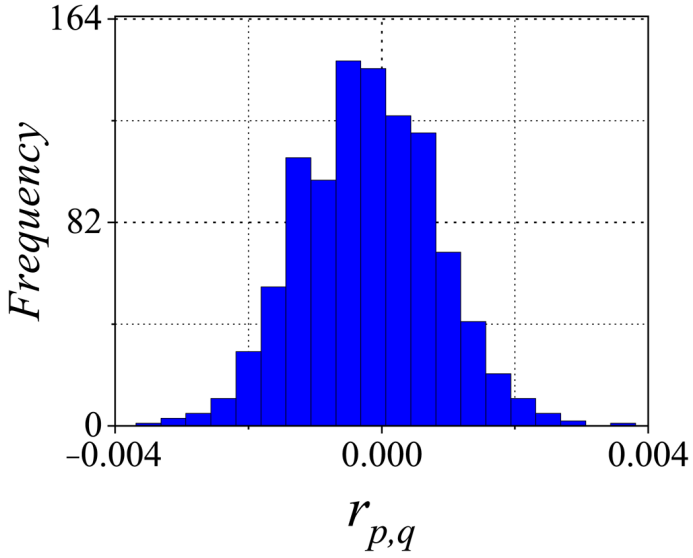

Figure 11. Statistical distribution for the different correlation coefficients.

4.2. Key Sensitivity

In order to evaluate the sensitivity of the proposed PRNG to small changes in the keys, we use the following metrics: the number of changing pixel rate (NPCR), the unified average changed intensity (UACI) [84] and the average absolute difference (AAD) [85]. For these tests, we select a set of 1000 keys, so that K_i and K_{i+1} ($i = 1, 2, 3, ..., 1000$) differ by a single bit between them. According to the structure of the proposed PRNG, we assume in the key for the proposed PRNG that only z_0 changes in the least significant bit (LSB), and this small change is quantified by η_0. Therefore, we select each initial condition considering that $z_0^{t+1} = z_0^t + \eta_0$, where $t = 1, 2, 3, ..., 1000$. Now, to calculate the NPCR, UACI, and ADD, we use Equations (13), (15) and (16), respectively, considering two cases. That is, the pseudorandom sequences are analyzed by reading 16-bits or 8-bits numbers.

In this way, NPCR is represented by Equation (13).

$$NPCR(p,q) = \frac{1}{n}\sum_{i=0}^{n} D_i(p,q), \tag{13}$$

where,

$$D(p,q) = \begin{cases} 0 \ if \ S_i^p = S_i^q \\ 1 \ if \ S_i^p \neq S_i^q \end{cases}, \tag{14}$$

UACI is represented by Equation (15).

$$UACI(p,q) = \frac{1}{n}\left[\sum_{i=0}^{n}\left(\frac{\left|S_i^p - S_i^q\right|}{2^{bits}-1}\right)\right], \tag{15}$$

where $bits = 8$ when the pseudorandom sequences are read in 8-bit numbers, and $bits = 16$ when they are read in 16-bit numbers.

And, AAD is represented by Equation (16).

$$AAD(p,q) = \frac{1}{n}\sum_{i=1}^{n}|S_i^p - S_i^q|, \tag{16}$$

Importantly, pseudorandom sequences of 1,000,000 numbers were generated for NPCR and UACI, and sequences of 2,000,000 numbers were generated for AAD. In all cases, the sequences are made up of 16-bit numbers. Figure 12 shows the statistical distribution for NPCR(p,q), UACI(p,q), and AAD(p,q). Figure 12a,c,e show $NPCR_{16}(p,q)$, $UACI_{16}(p,q)$,

$ADD_{16}(p,q)$ when the pseudorandom sequences are read in 16-bits numbers. On the other hand, Figure 12b,d,f show $NPCR_8(p,q)$, $UACI_8(p,q)$, $ADD_8(p,q)$ when the sequences are read in 8-bits numbers. Note that, when the pseudorandom sequences are observed as sequences with 16-bits numbers, the all calculated values for the NPCR(p,q) are 0.999977, for the UACI(p,q) the mean value is 0.33349, and for the AAD(p,q) the mean value is 2.18455×10^4. In a similar way, when the pseudorandom sequences are observed as sequences with 8-bits numbers, the calculated mean of the NPCR(p,q) is very close to 0.99608, for UACI(p,q) is close to 0.33460, and for AAD(p,q) is 85.3289, which is very close to ideal value reported by Wang et al. in 2016 [52].

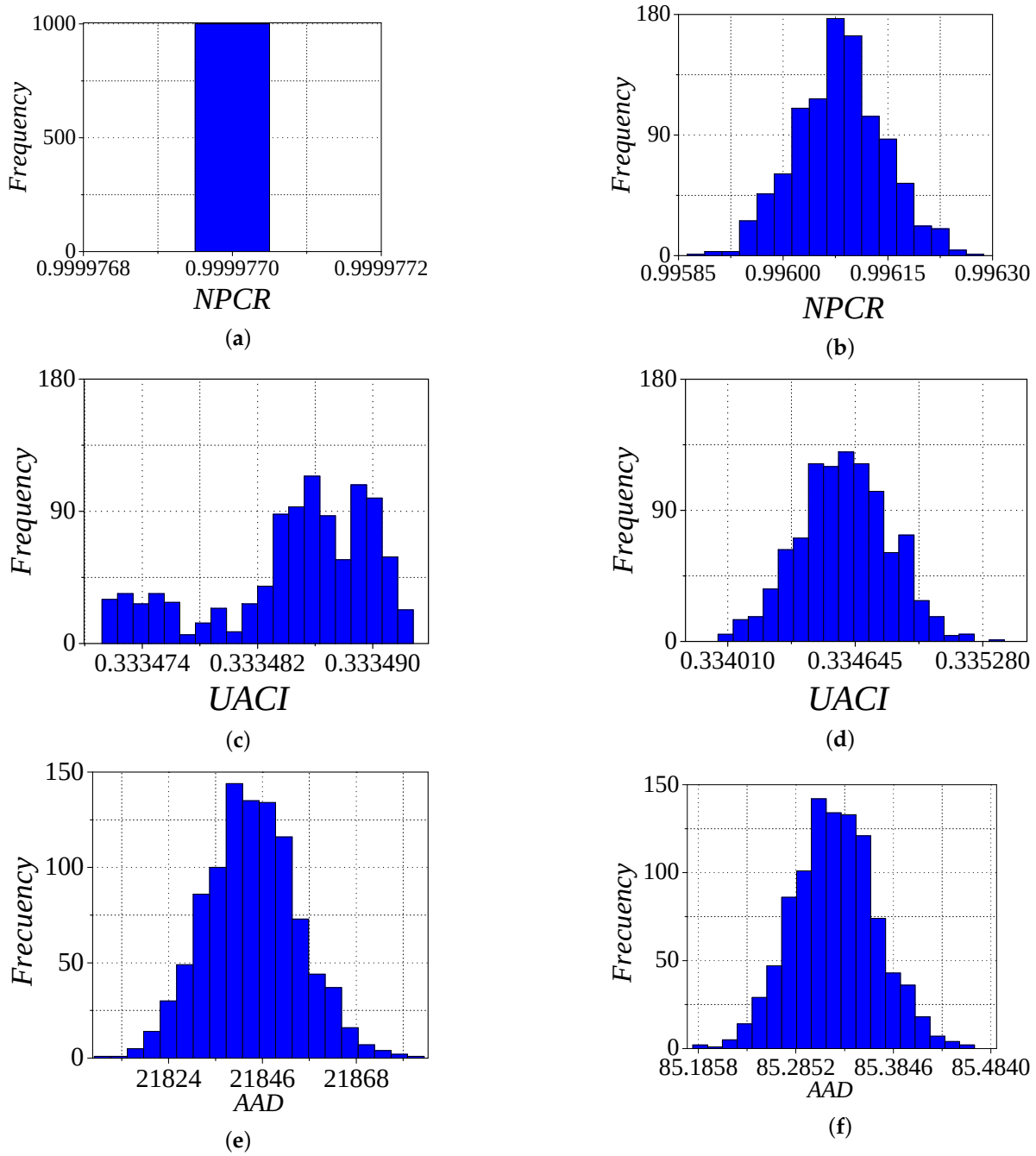

Figure 12. Statistical distribution for the sensitivity metrics estimated from 1000 pseudorandom sequences considering that the PRNG keys are very close to each other: (**a**) $NPCR_{16}(p,q)$, (**b**) $NPCR_8(p,q)$, (**c**) $UACI_{16}(p,q)$, (**d**) $UACI_8(p,q)$, (**e**) $AAD_{16}(p,q)$, and (**f**) $AAD_8(p,q)$.

4.3. Entropy Analysis

In order to measure the uncertainty degree in pseudorandom sequences generated by the proposed PRNG, we use the Shannon's entropy (H). For this test, we generate 1000 pseudorandom sequences of 1,000,000 with 16-bits numbers. In this case, we use the 8-bit entropy function. Thus, each pseudorandom sequence S^t was observed in 8-bit numbers and $H_8(S^t)$ can be calculated using Equation (17).

$$H(S^t) = \sum_{i=0}^{2^{bits}-1} p(S_i^t) log \frac{1}{p(S_i^t)}, \tag{17}$$

where $p(S_i^t)$ represents the probability estimated for each S_i^t, $t = 1, 2, ..., 1000$ and $bits = 8$.

When we calculated the statistical distribution of $H_8(S)$ for the pseudorandom sequences considering 8-bit numbers, the mean of $H_8(S^t)$ $\approx$7.99991 is very close to 8 as we expected; i.e., the proposed PRNG generates pseudorandom numbers of 8 bits approximately with equal distribution, which corresponds to a uniform statistical distribution.

4.4. Statistical Analysis and Randomness Testing

In order to evaluate the randomness of the sequences generated by the proposed PRNG, we consider two statistical test suites for the pseudorandomness evaluation of number sequences produced by the proposed PRNG: NIST SP 800-22 [86] and TestU01 [87]. For the NIST SP 800-22 suite and for each configuration of the proposed PRNG (8, 16 or 32-bit), we randomly select 2000 keys (initial conditions) to produce 2000 pseudorandom sequences generated with a size of $L = 1,000,000$ bits. Subsequently, to obtain the proportion of the sequences passing the test, we obtained the confidence interval defined as $\left(\hat{p} \pm 3\sqrt{\frac{\hat{p}(1-\hat{p})}{m}}\right) \cdot m$, where $m = 2000$, $\hat{p} = 1.0 - v$, and the significance level is $v = 0.01$. Therefore, the confidence interval for 2000 binary sequences must be in [1966, 1993]. It is worth noting that if $P_{value_T} \geq 0.0001$ then the sequences can be uniformly distributed. For those NIST sub-tests that consider more than one P_{value_T} an average value was obtained, which is shown in Table 3.

Table 3. Results of pseudorandomness tests applying the NIST SP 800-22 suite to 2000 binary sequences with a 1,000,000-bit length.

	8-Bit		16-Bit		32-Bit		
Statistical Test	**Proportion**	P_{value_T}	**Proportion**	P_{value_T}	**Proportion**	P_{value_T}	**Result**
Frequency	1983/2000	0.508172	1984/2000	0.279844	1979/2000	0.332188	Success
BlockFrequency	1978/2000	0.136499	1976/2000	0.474986	1984/2000	0.061453	Success
CumulativeSums *	1982/2000	0.785879	1981/2000	0.385543	1980/2000	0.401199	Success
Runs	1983/2000	0.872425	1979/2000	0.151616	1986/2000	0.119857	Success
LongestRun	1977/2000	0.307818	1983/2000	0.761719	1977/2000	0.709558	Success
Rank	1981/2000	0.541216	1976/2000	0.837781	1984/2000	0.536163	Success
FFT	1973/2000	0.239883	1981/2000	0.040768	1972/2000	0.020973	Success
NonOverlappingTemplate *	1979/2000	0.770499	1980/2000	0.985788	1985/2000	0.716696	Success
OverlappingTemplate	1973/2000	0.790621	1975/2000	0.714660	1982/2000	0.640243	Success
Universal	1981/2000	0.547298	1980/2000	0.060683	1974/2000	0.610070	Success
ApproximateEntropy	1974/2000	0.057875	1975/2000	0.496351	1976/2000	0.016431	Success
RandomExcursions *	1222/1242	0.640478	1228/1240	0.958260	1231/1240	0.887740	Success
RandomExcursionsVariant *	1973/2000	0.873987	1988/2000	0.961440	1978/2000	0.829047	Success
Serial *	1983/2000	0.357820	1978/2000	0.038565	1989/2000	0.899171	Success
LinearComplexity	1981/2000	0.059358	1981/2000	0.188090	1980/2000	0.279844	Success

* Average of multiple tests is considered.

On the other hand, the TestU01 test suite is a software library implemented in the ANSI C language and offering a collection of utilities for the empirical statistical testing of RNGs and PRNGs. TestU01 suite has three level of assessment: SmallCrush (15 tests), Crush

(144 tests) and BigCrush (160 tests). Besides, TestU01 includes the Alphabit, Rabbit and BlockAlphabit tests designed for testing bit generators implemented in hardware. Similar to the NIST tests, P_{value} is defined between [0.001, 0.9990] to pass the single tests. Then, we tested the proposed PRNG using the BigCrush (160 tests) level, Alphabit and Rabbit tests. Table 4 shows successful results when the proposed PRNG was configured for 8, 16 or 32 bits to generate 32-bit number sequences with size $L = 2^{26}$, 2^{28} and 2^{30}. It is worth noting that for the TestU01 suite a virtual computer with Ubuntu 64-bit operating system, 6 GB RAM and 3-core CPU was used. The total CPU time for testing the proposed PRNG were 58:08:56.82, 33:22:15.61 and 24:05:17.30 configured for 8, 16 and 32 bits, respectively.

Table 4. Results of pseudorandomness tests using TestU01 suite.

	PRNG Configuration					
	8-Bit		**16-Bit**		**32-Bit**	
Size	**Alphabit**	**Rabbit**	**Alphabit**	**Rabbit**	**Alphabit**	**Rabbit**
2^{26}	17/17	40/40	17/17	40/40	17/17	40/40
2^{28}	17/17	40/40	17/17	40/40	17/17	40/40
2^{30}	17/17	40/40	17/17	40/40	17/17	40/40
BigCrush	160/160		160/160		160/160	

4.5. Linear Complexity

The Berlekamp–Massey algorithm is used to estimate the linear complexity of a PRNG, and it is as tool to determine the shortest linear feedback shift register (LFSR) that produces a specific binary sequence [88]. Linear complexity in a PRNG is an important security condition when we want to know if such PRNG is suitable for cryptographic applications. A high linear complexity by itself does not guarantee any pseudorandomness property of the sequence under consideration, and therefore, it must also be known whether the sequences produced with the proposed PRNG pass the pseudo-randomness test suites of the NIST SP 800-22 and TestU01. Then we use the Berlekamp Massey algorithm to estimate the linear complexity, $L_c(S)$, of pseudorandom sequences, S^t, $t = 1, 2, 3, ..., W$, which are produced by the proposed PRNG and they are read in binary format. Basically, this test determines the minimum degree of the polynomial that produces, in a linear feedback shift register (LFSR), a sequence like S. Therefore, a PRNG with the highest possible linear complexity is desirable. For this test, we generate a set of $W = 1 \times 10^3$ pseudorandom sequences that in binary format have 2×10^5 bits. In this case, we use different initial conditions with slightly differences among them. Then, we compute the mean and the standard deviation of $L_c(S^t)$ for 1×10^3 sequences. Note in Figure 13 that $L_c(S)$ reaches a maximum level of 1×10^5, and the variation of the standard deviation values are small. Furthermore, note that the linear complexity test helps us to confirm that the approaches we implemented in the proposed PRNG have worked in order to increase linear complexity and avoid the problems presented by PRNGs based on single chaotic map.

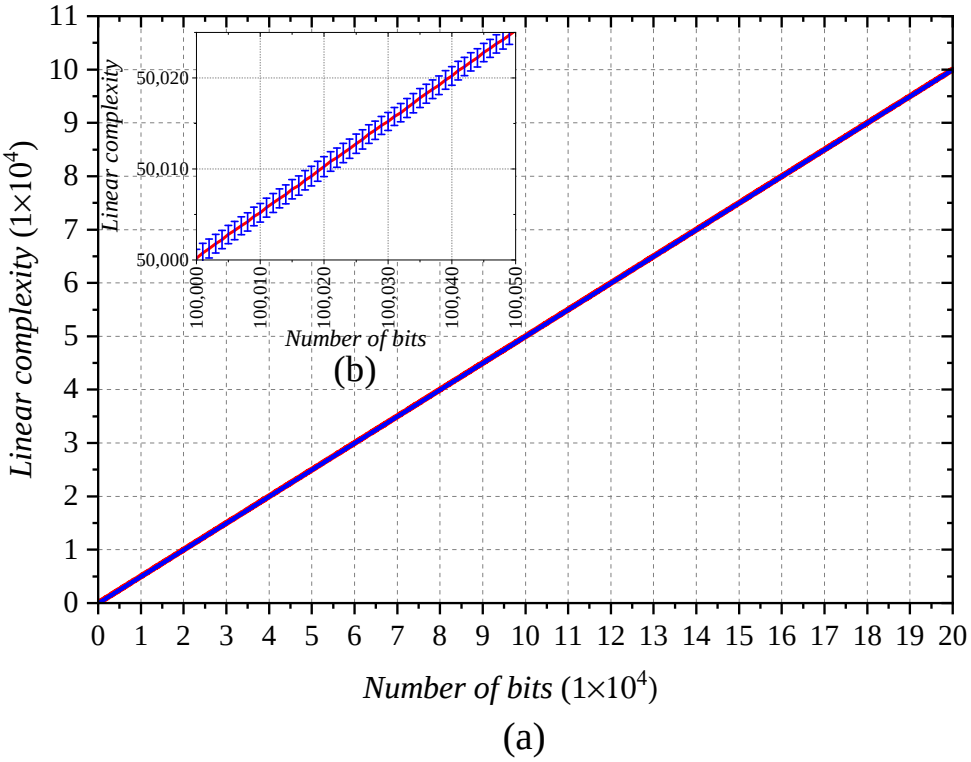

Figure 13. Linear complexity profile for 1000 sequences of pseudorandom numbers for the proposed PRNG. (**a**) considering sequences from 1 to 200,000 bits. (**b**) zoom of (**a**) considering sequences from 100,000 to 100,050 bits.

4.6. Key Space Analysis

The key space analysis is related with the security analysis in congruence with the Kerchoff's principle. This principle defines specifications related to the security analysis of a cryptographic module [89], and it says that a system for cryptography applications must be secure even if everything about the system is in the public domain, except the key. In this sense, we assume that the security of the proposed PRNG is associated with the size of the key space required to produce the numerical sequences. Assuming that the proposed PRNG is a cryptographic module of public knowledge, then its security is kept only in the key that is required to produce the pseudorandom sequences. Then, the proposed PRNG must have a key space as large as possible to be effective in a brute force attack. According to Figure 10, the key is constituted by μ_1, x_0, α_1, μ_2, y_0, α_2, μ_3, z_0, and α_3. Considering a standard format for floating point in double precision [90], the PRNG has 576 bits as key and then, the global key space is 2^{576} values, which satisfies the general requirement of resisting brute force attack. Now, to avoid the CATs conditions, we must select $\mu_i \leq \alpha_i$, $i = 1$, 2, and 3, and then, the key space is reduced to approximately 2^{384} values. In this calculation, we consider that each μ_i will be bounded to the least significant k_i bits, and consequently, each α_i will take values with the most significant $64 - k_i$ bits.

4.7. Speed Analysis

In order to show the performance of the proposed PRNG, we implemented it in electronic devices with 8, 16 and 32-bits architectures. Table 5 shows the clock cycles for the different configurations of the proposed PRNG according to the following criteria: when the proposed PRNG is set to 8-bit, the number obtained in each iteration is concatenated until to form a 32-bit number. Similarly, when the proposed PRNG is set to 16-bit, it must concatenate two 16-bit numbers to form a 32-bit number. Therefore, the results reported in Table 5 correspond to the clock cycles consumed by the proposed PRNG when it is configured for 8, 16 or 32 bits to generate 32-bit numbers.

Table 5. Clock cycles consumed by the proposed PRNG when the 8, 16, or 32-bit configurations are used.

Microcontroller Architecture	PRNG Configuration 8-Bit	16-Bit	32-Bit
8-bit	141,645	132,920	25,139
16-bit	67,699	65,290	12,319
32-bit	33,206	30,605	5308

Additionally, we implemented the proposed PRNG for the 8, 16 and 32-bit configurations by using a C language compiler (MinGW) on an Intel Core i7-4800MQ CPU-2.70GHz with 24G RAM. For each case, we obtained the time required to generate 1,000,000 pseudorandom numbers performing this process 2000 times, and then the average time was calculated. These results are reported in Table 6 and they allow consider that the proposed PRNG can be implemented in an electronic system with limited hardware.

Table 6. Execution time to generate 1,000,000 pseudorandom numbers by using the proposed PRNG considering the three configurations.

PRNG Configuration	Running Speed (MBytes/s)
8-bit	4.720276
16-bit	9.163551
32-bit	19.978822

5. Comparison Results

The efficiency of the proposed PRNG is compared with similar PRNGs based on chaotic maps. In this section, we focus on four tests: Correlation coefficient, Key sensitivity using correlation coefficient and variance ratio, key space and running speed. For this section three PRNGs were selected: (i) 32-bit PRNG proposed by Zhang et al. [91], (ii) 8-bit PRNG proposed by Huang et al. [40] and (iii) 8-bit PRNG proposed by Liu et al. [92]. We performed experiments on equal terms to the considered PRNGs for comparison with similar works. The comparison tests were developed using a C language compiler (MinGW) on an Intel Core i7-4800MQ CPU-2.70GHz and 24G RAM. Then, to determine the correlation coefficient we generate 6000 number sequences of 12000 numbers with different keys. The correlation coefficient obtained was within $[-0.032, 0.029]$, while for the PRNG reported by Zhang et al. the correlation coefficient was within $[-0.035, 0.035]$. Regarding the key sensitivity, four sets of keys with a single bit difference between them were defined, then four number sequences of 12,000 numbers were generated. Finally, we obtain the difference between the sequences by applying the correlation coefficient and calculate the average to obtain the value shown in the Table 7. Note that the key sensitivity obtained for the proposed PRNG is slightly lower than the key sensitivity reported by Zhang et al. [91]. It is worth noting that the key space of the proposed PRNG considers double precision for 64-bit numbers, which is considered a great advantage over the PRNG developed by Zhang et al. [91]. Regarding the speed running test, Zhang et al. use an Intel Core i7-10710U CPU and 16GB RAM. The algorithms were implemented in Visual Studio 2019 using C++, it can be observed that the PRNG proposed by Zhang et al. [91] has a high speed with respect to the proposed PRNG. However, the different architectures under which the tests were carried out could affect the measurements.

In the second part of this section, the tests were performed when the proposed PRNG is set to 8-bit and only three tests are considered: key sensitivity, key space and running speed. Considering that the proposed PRNG has a high sensitivity to key changes, we performed the key sensitivity test using two different sequences generated by using two keys: K_1 and K_2, where $|K_1 - K_2| = 1 \times 10^{-15}$. Then, we calculated the variance ratio (D) [40,92] between the two binary sequences with size $L = 1,000,000$ resulting $D = 49.9872\%$, which

is similar to results reported by Liu et al. [92] and Huang et al. [40]. On the other hand, the proposed PRNG has a key space larger than the PRNG proposed by Liu et al., but its key space is similar to the PRNG proposed by Huang et al. [40]. Finally, the running speed of the proposed PRNG is similar to running speed of the PRNG proposed by Zhang et al. [91]. It is woth noting that each author performs the tests with different equipment. For example, Liu et al. [92] used a computer with 3.3 GHz CPU and 4GB RAM, but they do not indicate the used programming language. Huang et al. [40] used a computer with 3.3 GHz CPU, 4GB RAM, and MATLAB 2014R. Note in Table 7 that the proposed PRNG has a competitive performance when it is configured for 8 and 32 bits, and when compared against the PRNGs proposed by Zhang et al. [91], Huang et al. [40], and Liu et al. [92]. Table 7 does not include information comparison for the 16-bit configuration because we do not find similar PRNGs with 16-bit configurations, which could be used in the comparison.

Table 7. Comparison analysis of the proposed PRNG against similar algorithms.

Test	Proposed PRNG	Zhang et al. [91]	Liu et al. [92]	Huang et al. [41]
Corr. Coeff. [b]	$[-0.032, 0.029]$	$[-0.035, 0.035]$	–	–
Key Sensit: Corr. Coeff [b]	0.009278	0.007724	–	–
Key Sensit: The variance ratio (D) [a]	49.9872%	–	49.9850%	49.9950%
Key space	2^{576}	2^{136}	2^{184}	2^{448}
Running speed (MB/s)	19.978822 [b]	65.867475 [b]	–	–
	4.720276 [a]	–	2.729887 [a]	2.688817 [a]

[a] 8-bit generator and [b] 32-bit generator.

6. Conclusions

This work contributes to the design of PRNGs based on chaotic maps. In this case, we introduce a function composition (FC), which couples the sine function and the skew tent map to produce pseudorandom number sequences. We analyze the behavior of the chaotic maps by using the bifurcation diagram and Lyapunov exponent, and identifying the chaos annulling conditions and stability islands. In the FC, the Lyapunov exponent is positive when the control μ is in $(0, \alpha)$ and then it can be used in the implementation of a PRNG. Using three FCs, the proposed PRNG has a large key space, it produces pseudorandom sequences with good statistical features and it has robust sensitivity to key changes. Ideally, the key space of the proposed PRNG is 2^{576}, and in a modest case it can be 2^{384}. Similarly, the strategy used to translate real numbers sequences into 8, 16 or 32-bit integer number sequences does not affect the behavior of the used chaotic maps. This does not exclude the possibility of having different behaviors due to precision errors in the representation of real numbers and arithmetic operations. Therefore, in this work we consider using the highest precision available when implemented on a computer or digital electonic device. In this regard, it would be interesting to research the possibility of implementing the proposed chaotic maps by using microfluidic-based processors and circuits. On the other hand, in this work, we prove that the proposed PRNG can produce uniformly distributed number sequences when the annulling chaos conditions are identified and avoided. Further, the number sequences generated by the proposed PRNG were evaluated by the following set of tests: correlation coefficient, key sensitivity, statistical analysis, entropy, linear complexity, and pseudorandomness. Additionally, we estimate the key space and the execution time when the proposed PRNG was programmed in C Language and electronically implemented on low-resources devices; notably, in all tests the proposed PRNG had a good performance. We especially emphasize that the proposed PRNG has a very high linear complexity when evaluated using the Berlekamp-Massey algorithm avoiding the problems presented by PRNGs based on a single chaotic map. Also, the proposed PRNG can be configured to generate pseudorandom 8, 16 or 32-bits numbers, so it can be implemented in microcontrollers of different architectures. Note that the proposed PRNG is two times faster than the algorithms proposed by Huang et al. and

Li et al., but is three times slower than the algorithm proposed by Zhang et al. when it is configured for 32 bits, since the algorithm proposed by Zhang et al. was computationally improved. In the key sensitivity test we considered two approaches: variance ration and correlation coefficient. Note that variance ratio is very close to 50%, which is similar to the results reported by Huang et al. and Li et al. Similarly, the correlation coefficient is very close to zero, which is similar to results reported by Zhang et al. Respecting to the pseudorandomness of the number sequences, we highlighted that the proposed PRNG configured for 8, 16 or 32 bits passes all tests of the NIST SP 800-22 suite considering 1×10^3 and 2×10^3 binary sequences, where each sequence has 1×10^6 numbers. For the TestU01 suite, we consider the BigCrush level, Alphabit and Rabbit tests. Note that the proposed PRNG configured for 8, 16 or 32 bits passes all tests. Consequently, based on the various tests performed the proposed PRNG generates pseudorandom sequences with good statistical properties when is configured for 8, 16 or 32 bits. It is important to mention that a strict security analysis to determine whether the proposed PRNG is cryptographically secure is not included in this work. This issue is not in the scope of this work. But the results obtained for linear complexity give a good indication that the proposed PRNG is secure. However, despite the analysis we present about key space and linear complexity, we recommend performing a strict cryptographic security analysis of the proposed PRNG before it can be used in cryptography and/or security applications. Note that the confirmation of compliance with the Shujun's requirements is not included in the scope of this work. This is because we do not propose the use of a single one-dimensional chaotic map, rather we propose a function composition, which couples the chaotic tent map and the sine function. Furthermore, we recommend that if the proposed PRNG is used in stream ciphers, the Shujun's requirements should be verified. Finally, we have to remark that it could be of interest to research chaotic maps that can be implemented in microfluidic-based processors and circuits.

Author Contributions: Conceptualization, L.P.-L. and R.V.-M.; methodology, R.M.-J. and R.V.-M.; software, L.P.-L.; validation, R.M.-J., E.R.-C. and M.P.-C.; formal analysis, L.P.-L. and R.V.-M.; investigation, L.P.-L. and O.J.-R.; resources, all authors; data curation, R.M.-J., E.R.-C. and M.P.-C.; writing—original draft preparation, L.P.-L. and R.V.-M.; writing—review and editing, R.V.-M., R.M.-J. and O.J.-R.; visualization, M.P.-C. and O.J.-R.; supervision, L.P.-L.; project administration, R.V.-M.; funding acquisition, R.V.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Council of Science and Technology (CONACyT), Autonomous Metropolitan University-Iztapalapa (L. Palacios-Luengas, visiting professor) and the Instituto Politécnico Nacional, México [Grants No. SIP-20210023 (R. Vázquez-Medina) and SIP-20210208 (O. Jiménez-Ramírez)].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that supports the findings of this study are available within the article.

Acknowledgments: The authors thank A. L. Quintanar-Reséndiz for the technical support in the implementation and realization of the experiments.

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this paper.

References

1. Ming, X.; Chen, Z.; Zhou, Z.K.; Zhang, B. An advanced spread spectrum architecture using pseudorandom modulation to improve EMI in class D amplifier. *Power Electron. IEEE Trans.* **2011**, *26*, 638–646. [CrossRef]
2. Meliá-Seguí, J.; Garcia-Alfaro, J.; Herrera-Joancomartí, J. J3Gen: A PRNG for low-cost passive RFID. *Sensors* **2013**, *13*, 3816–3830. [CrossRef]

3. Mandal, K.; Fan, X.; Gong, G. Design and implementation of warbler family of lightweight pseudorandom number generators for smart devices. *ACM Trans. Embed. Comput. Syst. TECS* **2016**, *15*, 1. [CrossRef]
4. Liao, Y.; Fan, X. Mathematical calculation of sequence length in LFSR-dithered MASH digital delta-sigma modulator with odd initial condition. *AEU Int. J. Electron. Commun.* **2017**, *80*, 114–126. [CrossRef]
5. Cotrina, G.; Peinado, A.; Ortiz, A. Gaussian pseudorandom number generator based on cyclic rotations of Linear Feedback Shift Registers. *Sensors* **2020**, *20*, 2103. [CrossRef]
6. Feng, L.; Xiaoxing, G. A new construction of pseudorandom number generator. *J. Netw.* **2014**, *9*, 2176–2183.
7. Payingat, J.; Pattathil, D.P. Pseudorandom bit sequence generator for stream cipher based on elliptic curves. *Math. Probl. Eng.* **2015**, *2015*, 257904. [CrossRef]
8. El-Latif, A.A.A.; El-Atty, B.A.; Venegas-Andraca, S.E. Controlled alternate quantum walk-based pseudo-random number generator and its application to quantum color image encryption. *Phys. A Stat. Mech. Appl.* **2020**, *547*, 123869. [CrossRef]
9. Spencer, J. Pseudorandom bit generators from enhanced cellular automata. *J. Cell. Autom.* **2015**, *10*, 295–317.
10. Bhattacharjee, K.; Das, S. Random number generation using decimal cellular automata. *Commun. Nonlinear Sci. Numer. Simul.* **2019**, *78*, 104878. [CrossRef]
11. Tuna, M. A novel secure chaos-based pseudo random number generator based on ANN-based chaotic and ring oscillator: Design and its FPGA implementation. *Analog. Integr. Circuits Signal Process.* **2020**, *105*, 167–181. [CrossRef]
12. Guo, W.; Zhao, J.; Ye, R. A chaos-based pseudorandom permutation and bilateral diffusion scheme for image encryption. *Int. J. Image Graph. Signal Process.* **2014**, *6*, 50.
13. Senouci, A.; Bouhedjeur, H.; Tourche, K.; Boukabou, A. FPGA based hardware and device-independent implementation of chaotic generators. *AEU Int. J. Electron. Commun.* **2017**, *82*, 211–220. [CrossRef]
14. Palacios-Luengas, L.; Pichardo-Méndez, J.L.; Díaz-Méndez, J.A.; Rodríguez-Santos, F.; Vázquez-Medina, R. PRNG Based on skew tent map. *Arab. J. Sci. Eng.* **2018**, *44*, 3817–3830. [CrossRef]
15. Irfan, M.; Ali, A.; Khan, M.A.; Ul Haq, M.E.; Shah, S.N.M.; Saboor, A.; Ahmad, W. Pseudorandom number generator (PRNG) design using hyper-chaotic modified robust logistic map (HC-MRLM). *Electronics* **2020**, *9*, 104. [CrossRef]
16. Alhadawi, H.S.; Zolkipli, M.F.; Ismail, S.M.; Lambić, D. Designing a pseudorandom bit generator based on LFSRs and a discrete chaotic map. *Cryptologia* **2019**, *43*, 190–211. [CrossRef]
17. Capeáns, R.; Sabuco, J.; Sanjuán, M.A.F. Parametric partial control of chaotic systems. *Nonlinear Dyn.* **2016**, *86*, 869–876. [CrossRef]
18. Pecora, L.M.; Carroll, T.L. Synchronization of chaotic systems. *Chaos Interdiscip. J. Nonlinear Sci.* **2015**, *25*, 097611. [CrossRef]
19. Csernák, G.; Gyebrószki, G.; Stépán, G. Multi-Baker map as a model of digital PD control. *Int. J. Bifurc. Chaos* **2016**, *26*, 1650023. [CrossRef]
20. Capeáns, R.; Sabuco, J.; Sanjuán, M.A.F.; Yorke, J.A. Partially controlling transient chaos in the Lorenz equations. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2017**, *375*, 20160211. [CrossRef]
21. Ferrer, J.; Ballesté, A.; Roca, J.; Virgili, U.R.; Gómez, A.; Arroyo, D.; Amigó, J.; Li, S.; Alvarez, G. *On the Inadequacy of Unimodal Maps for Cryptographic Applications*; URV: Tarragona, Spain, 2010.
22. Palacios-Luengas, L.; Delgado-Gutiérrez, G.; Díaz-Méndez, J.A.; Vázquez-Medina, R. Symmetric cryptosystem based on skew tent map. *Multimed. Tools Appl.* **2017**, *77*, 2739–2770. [CrossRef]
23. Teh, J.S.; Samsudin, A. A chaos-based authenticated cipher with associated data. *Secur. Commun. Netw.* **2017**, *2017*, 1–15. [CrossRef]
24. Yu, F.; Li, L.; Tang, Q.; Cai, S.; Song, Y.; Xu, Q. A survey on true random number generators based on chaos. *Discret. Dyn. Nat. Soc.* **2019**, *2019*, 1–10. [CrossRef]
25. Farah, M.A.B.; Farah, A.; Farah, T. An image encryption scheme based on a new hybrid chaotic map and optimized substitution box. *Nonlinear Dyn.* **2019**, *99*, 3041–3064. [CrossRef]
26. Liu, H.; Kadir, A.; Ma, C.; Xu, C. Constructing keyed hash algorithm using enhanced chaotic map with varying parameter. *Math. Probl. Eng.* **2020**, *2020*, 1–10. [CrossRef]
27. Kari, A.P.; Navin, A.H.; Bidgoli, A.M.; Mirnia, M. A new image encryption scheme based on hybrid chaotic maps. *Multimed. Tools Appl.* **2020**. [CrossRef]
28. Tutueva, A.V.; Karimov, A.I.; Moysis, L.; Volos, C.; Butusov, D.N. Construction of one-way hash functions with increased key space using adaptive chaotic maps. *Chaos Solitons Fractals* **2020**, *141*, 110344. [CrossRef]
29. Zhou, P.; Du, J.; Zhou, K.; Wei, S. 2D mixed pseudo-random coupling PS map lattice and its application in S-box generation. *Nonlinear Dyn.* **2021**. [CrossRef]
30. Midoun, M.A.; Wang, X.; Talhaoui, M.Z. A sensitive dynamic mutual encryption system based on a new 1D chaotic map. *Opt. Lasers Eng.* **2021**, *139*, 106485. [CrossRef]
31. Saber, M.; Eid, M.M. Low power pseudo-random number generator based on lemniscate chaotic map. *Int. J. Electr. Comput. Eng. IJECE* **2021**, *11*, 863. [CrossRef]
32. Hu, G.; Li, B. Coupling chaotic system based on unit transform and its applications in image encryption. *Signal Process.* **2021**, *178*, 107790. [CrossRef]
33. Mathivanan, P.; Balaji, G.A. QR code based color image stego-crypto technique using dynamic bit replacement and logistic map. *Optik* **2021**, *225*, 165838. [CrossRef]

34. Hu, H.; Liu, L.; Ding, N. Pseudorandom sequence generator based on the Chen chaotic system. *Comput. Phys. Commun.* **2013**, *184*, 765–768. [CrossRef]
35. García-Martínez, M.; Campos-Cantón, E. Pseudo-random bit generator based on multi-modal maps. *Nonlinear Dyn.* **2015**, *82*, 2119–2131. [CrossRef]
36. Murillo-Escobar, M.A.; Cruz-Hernández, C.; Cardoza-Avendaño, L.; Mendez-Ramírez, R. A novel pseudorandom number generator based on pseudorandomly enhanced logistic map. *Nonlinear Dyn.* **2017**, *87*, 407–425. [CrossRef]
37. Dastgheib, M.A.; Farhang, M. A digital pseudo-random number generator based on sawtooth chaotic map with a guaranteed enhanced period. *Nonlinear Dyn.* **2017**, *89*, 2957–2966. [CrossRef]
38. Sahari, M.L.; Boukemara, I. A pseudo-random numbers generator based on a novel 3D chaotic map with an application to color image encryption. *Nonlinear Dyn.* **2018**, *94*, 723–744. [CrossRef]
39. Garcia-Bosque, M.; Perez-Resa, A.; Sanchez-Azqueta, C.; Aldea, C.; Celma, S. Chaos-based bitwise dynamical pseudorandom number generator on FPGA. *IEEE Trans. Instrum. Meas.* **2019**, *68*, 291–293. [CrossRef]
40. Huang, X.; Liu, L.; Li, X.; Yu, M.; Wu, Z. A new two-dimensional mutual coupled logistic map and its application for pseudorandom number generator. *Math. Probl. Eng.* **2019**, *2019*, 1–10. [CrossRef]
41. Huang, X.; Liu, L.; Li, X.; Yu, M.; Wu, Z. A new pseudorandom bit generator based on mixing three–dimensional Chen chaotic system with a chaotic tactics. *Complexity* **2019**, *2019*, 1–9. [CrossRef]
42. Datcu, O.; Macovei, C.; Hobincu, R. Chaos based cryptographic pseudo-random number generator template with dynamic state change. *Appl. Sci.* **2020**, *10*, 451. [CrossRef]
43. OISHI, S.; INOUE, H. Pseudo-random number generators and chaos. *IEICE Trans.* **1982**, *E65*, 534–542.
44. González, J.A.; Pino, R. A random number generator based on unpredictable chaotic functions. *Comput. Phys. Commun.* **1999**, *120*, 109–114. [CrossRef]
45. Stojanovski, T.; Kocarev, L. Chaos-based random number generators-part I: Analysis [cryptography]. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2001**, *48*, 281–288. [CrossRef]
46. Stojanovski, T.; Pihl, J.; Kocarev, L. Chaos-based random number generators. part II: Practical realization. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2001**, *48*, 382–385. [CrossRef]
47. Li, S.; Mou, X.; Cai, Y. Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2001; pp. 316–329. [CrossRef]
48. Rezk, A.A.; Madian, A.H.; Radwan, A.G.; Soliman, A.M. Reconfigurable chaotic pseudo random number generator based on FPGA. *AEU Int. J. Electron. Commun.* **2019**, *98*, 174–180. [CrossRef]
49. Wang, Y.; Zhang, Z.; Wang, G.; Liu, D. A pseudorandom number generator based on a 4D piecewise logistic map with coupled parameters. *Int. J. Bifurc. Chaos* **2019**, *29*, 1950124. [CrossRef]
50. Chen, C.; Sun, K.; Peng, Y.; Alamodi, A.O.A. A novel control method to counteract the dynamical degradation of a digital chaotic sequence. *Eur. Phys. J. Plus* **2019**, *134*. [CrossRef]
51. Short, K.M. Steps toward unmasking secure communications. *Int. J. Bifurc. Chaos* **1994**, *4*, 959–977. [CrossRef]
52. Wang, Y.; Liu, Z.; Ma, J.; He, H. A pseudorandom number generator based on piecewise logistic map. *Nonlinear Dyn.* **2016**, *83*, 2373–2391. [CrossRef]
53. Lambić, D. Security analysis and improvement of the pseudo-random number generator based on piecewise logistic map. *J. Electron. Test.* **2019**, *35*, 519–527. [CrossRef]
54. Zhou, S.; Wei, Z.; Wang, B.; Zheng, X.; Zhou, C.; Zhang, Q. Encryption method based on a new secret key algorithm for color images. *AEU Int. J. Electron. Commun.* **2016**, *70*, 1–7. [CrossRef]
55. Shi, Y.; Deng, Y. Hybrid control of digital Baker map with application to pseudo-random number generator. *Entropy* **2021**, *23*, 578. [CrossRef]
56. Short, K.M. Signal extraction from chaotic communications. *Int. J. Bifurc. Chaos* **1997**, *7*, 1579–1597. [CrossRef]
57. Francois, M.; Grosges, T.; Barchiesi, D.; Erra, R. A new pseudo-random number generator based on two chaotic maps. *Informatica* **2013**, *24*, 181–197. [CrossRef]
58. Protopopescu, V.A.; Santoro, R.T.; Tolliver, J.S. *Fast and Secure Encryption-Decryption Method Based on Chaotic Dynamics*; Technical Report; Oak Ridge National Lab. (ORNL): Oak Ridge, TN, USA, 1995.
59. Alawida, M.; Samsudin, A.; Teh, J.S. Enhancing unimodal digital chaotic maps through hybridisation. *Nonlinear Dyn.* **2019**, *96*, 601–613. [CrossRef]
60. Alawida, M.; Samsudin, A.; Teh, J.S.; Alkhawaldeh, R.S. A new hybrid digital chaotic system with applications in image encryption. *Signal Process.* **2019**, *160*, 45–58. [CrossRef]
61. Zhou, Y.; Hua, Z.; Pun, C.M.; Chen, C.L.P. Cascade chaotic system with applications. *IEEE Trans. Cybern.* **2015**, *45*, 2001–2012. [CrossRef] [PubMed]
62. Hu, H.; Deng, Y.; Liu, L. Counteracting the dynamical degradation of digital chaos via hybrid control. *Commun. Nonlinear Sci. Numer. Simul.* **2014**, *19*, 1970–1984. [CrossRef]
63. Deng, Y.; Hu, H.; Xiong, N.; Xiong, W.; Liu, L. A general hybrid model for chaos robust synchronization and degradation reduction. *Inf. Sci.* **2015**, *305*, 146–164. [CrossRef]
64. Lu, H.; Wang, S.; Hu, G. Pseudo-random number generator based on coupled map lattices. *Int. J. Mod. Phys. B* **2004**, *18*, 2409–2414. [CrossRef]

65. Behnia, S.; Akhshani, A.; Mahmodi, H.; Akhavan, A. A novel algorithm for image encryption based on mixture of chaotic maps. *Chaos Solitons Fractals* **2008**, *35*, 408–419. [CrossRef]
66. Garasym, O.; Lozi, R.; Taralova, I. Robust PRNG based on homogeneously distributed chaotic dynamics. *J. Phys. Conf. Ser.* **2016**, *692*, 012011. [CrossRef]
67. Pichardo-Méndez, J.L.; Palacios-Luengas, L.; Martínez-González, R.F.; Jiménez-Ramírez, O.; Vázquez-Medina, R. LSB Pseudorandom algorithm for image steganography using skew tent map. *Arab. J. Sci. Eng.* **2019**, *45*, 3055–3074. [CrossRef]
68. Peitgen, H.; Jurgens, H.; Saupe, D. *Fractals for the Classroom: Part Two: Complex Systems And Mandelbrot Set*; Springer: New York, NY, USA, 1992.
69. Schroeder, M. *Fractals, Chaos, Power Laws: Minutes From an Infinite Paradise*; Dover Publication INC.: Mineola, NY, USA, 2009.
70. Lasota, A.; Mackey, M.C. *Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 97.
71. Grebogi, C.; Ott, E.; Yorke, J.A. Roundoff-induced periodicity and the correlation dimension of chaotic attractors. *Phys. Rev. A* **1988**, *38*, 3688–3692. [CrossRef]
72. Alawida, M.; Samsudin, A.; Teh, J.S.; Alshoura, W.H. Deterministic chaotic finite-state automata. *Nonlinear Dyn.* **2019**, *98*, 2403–2421. [CrossRef]
73. Fan, C.; Ding, Q. Analysing the dynamics of digital chaotic maps via a new period search algorithm. *Nonlinear Dyn.* **2019**, *97*, 831–841. [CrossRef]
74. Franzosi, R.; Poggi, P.; Cerruti-Sola, M. Lyapunov exponents from unstable periodic orbits. *Phys. Rev. E* **2005**, *71*. [CrossRef] [PubMed]
75. Liu, F.; Feng, Y. Dynamic multimapping composite chaotic sequence generator algorithm. *AEU Int. J. Electron. Commun.* **2019**, *107*, 231–238. [CrossRef]
76. Whitesides, G.M. The origins and the future of microfluidics. *Nature* **2006**, *442*, 368–373. [CrossRef]
77. Anandan, P.; Gagliano, S.; Bucolo, M. Computational models in microfluidic bubble logic. *Microfluid. Nanofluidics* **2014**, *18*, 305–321. [CrossRef]
78. Aryasomayajula, A.; Bayat, P.; Rezai, P.; Selvaganapathy, P.R. Microfluidic devices and their applications. In *Springer Handbook of Nanotechnology*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 487–536. [CrossRef]
79. Azizbeigi, K.; Pedram, M.Z.; Sanati-Nezhad, A. Microfluidic-based processors and circuits design. *Sci. Rep.* **2021**, *11*. [CrossRef] [PubMed]
80. Prakash, M.; Gershenfeld, N. Microfluidic bubble logic. *Science* **2007**, *315*, 832–835. [CrossRef] [PubMed]
81. Fuerstman, M.J.; Garstecki, P.; Whitesides, G.M. Coding/decoding and reversibility of droplet trains in microfluidic networks. *Science* **2007**, *315*, 828–832. [CrossRef]
82. Prakash, M.; Gershenfeld, N. Microfluidic Bubble Logic Devices. U.S. Patent 7918244 B2, 11 January 2007 .
83. Li, S.; Chen, G.; Mou, X. On the dynamical degradation of digital piecewise linear chaotic maps. *Int. J. Bifurc. Chaos* **2005**, *15*, 3119–3151. [CrossRef]
84. Li, Y.; Ge, G.; Xia, D. Chaotic hash function based on the dynamic S-Bx with variable parameters. *Nonlinear Dyn.* **2016**, *84*, 2387–2402. [CrossRef]
85. Teh, J.S.; Alawida, M.; Ho, J.J. Unkeyed hash function based on chaotic sponge construction and fixed-point arithmetic. *Nonlinear Dyn.* **2020**, *100*, 713–729. [CrossRef]
86. Rukhin, A.; Sota, J.; Nechvatal, J.; Smid, M.; Barker, E.; Leigh, S.; Levenson, M.; Vangel, M.; Banks, D.; Heckert, A.; et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, Revision 1a*; Technical Report; Information Technology Laboratory, National Institute of Standards and Technology, Department of Commerce: Gaithersburg, MD, USA, 2010. [CrossRef]
87. Sleem, L.; Couturier, R. TestU01 and Practrand: Tools for a randomness evaluation for famous multimedia ciphers. *Multimed. Tools Appl.* **2020**. [CrossRef]
88. Massey, J.L. Shift-register synthesis and BCH decoding. *IEEE Trans. Inform. Theory* **1967**, *13*, 21–27. [CrossRef]
89. van Tilborg, H.C.A.; Jajodia, S. Kerckhoffs' Law. In *Encyclopedia of Cryptography and Security*; Springer US: Boston, MA, USA, 2011; pp. 675. [CrossRef]
90. Hough, D. (Ed.) *754-2019—IEEE Standard for Floating-Point Arithmetic*; IEEE Computer Society, 754 WG Working Group for Floating Point Arithmetic; IEEE: Piscataway, NJ, USA, 2019.
91. Zhiqiang, Z.; Yong, W.; Leo, Y.Z.; Hong, Z. A novel chaotic map constructed by geometric operations and its application. *Nonlinear Dyn.* **2020**, *102*, 2843–2858. [CrossRef]
92. Lingfeng, L.; Bocheng, L.; Hanping, H.; Suoxia, M. Reducing the dynamical degradation by bi-coupling digital chaotic maps. *Int. J. Bifurc. Chaos* **2018**, *28*, 1850059. [CrossRef]