Article

Chaotic Path Planning for 3D Area Coverage Using a Pseudo-Random Bit Generator from a 1D Chaotic Map

Lazaros Moysis 1,*, Karthikeyan Rajagopal 2, Aleksandra V. Tutueva 3, Christos Volos 1, Beteley Teka 4 and Denis N. Butusov 3

1 Laboratory of Nonlinear Systems—Circuits & Complexity (LaNSCom), Physics Department, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece; volos@physics.auth.gr
2 Center for Nonlinear Systems, Chennai Institute of Technology, Chennai 600069, India; karthikeyan.rajagopal@citcollege.net
3 Youth Research Institute, Saint-Petersburg Electrotechnical University “LETI”, 197376 St Petersburg, Russia; avtutueva@etu.ru (A.V.T.); dnbutusov@etu.ru (D.N.B.)
4 Department of Electronics Engineering, Defence University College of Engineering, Bishoftu 1041, Ethiopia; beteley.teka@dec.edu.et
* Correspondence: lmousis@physics.auth.gr

Abstract: This work proposes a one-dimensional chaotic map with a simple structure and three parameters. The phase portraits, bifurcation diagrams, and Lyapunov exponent diagrams are first plotted to study the dynamical behavior of the map. It is seen that the map exhibits areas of constant chaos with respect to all parameters. This map is then applied to the problem of pseudo-random bit generation using a simple technique to generate four bits per iteration. It is shown that the algorithm passes all statistical NIST and ENT tests, as well as shows low correlation and an acceptable key space. The generated bitstream is applied to the problem of chaotic path planning, for an autonomous robot or generally an unmanned aerial vehicle (UAV) exploring a given 3D area. The aim is to ensure efficient area coverage, while also maintaining an unpredictable motion. Numerical simulations were performed to evaluate the performance of the path planning strategy, and it is shown that the coverage percentage converges exponentially to 100% as the number of iterations increases. The discrete motion is also adapted to a smooth one through the use of B-Spline curves.

Keywords: chaos; pseudo-random bit generator; path planning; chaotic mobile robot; UAV

1. Introduction

Chaotic systems have long been integrated into numerous scientific fields and have found various applications in engineering, sensors, data encryption, secure communications, robotics, economics and more (see, e.g., [1,2]).

Due to their practical feasibility, there is always a need for new chaotic maps that can be used in real-world applications. This can be performed either by considering modifications of existing maps or by constructing new maps altogether, usually having as few terms and parameters as possible. This is especially the case for one-dimensional chaotic maps since they are easier to implement due to their simplicity and low computational cost (see, e.g., [3–5]).

Motivated by the above, this work proposes a simple one-dimensional chaotic map with three parameters and only two terms, having a simple nonlinearity introduced by the absolute value function. From the dynamical analysis of the map through bifurcation diagrams and Lyapunov exponent diagrams, it is shown that the map has constant chaotic behavior for a range of parameter values. Furthermore, the proposed map is then applied to two engineering problems, the problem of pseudo-random bit generation and chaotic path planning for 3D area coverage.

The problem of chaotic pseudo-random bit generation (PRBG) refers to constructing an algorithm for generating a bit sequence, using a chaotic system [4,6–13]. Despite being
deterministic in nature (hence, the term pseudo-random), the resulting bitstream should have the statistical characteristics of a random sequence, which can be verified by a series of tests. Here, the proposed method uses a simple modulo operator to generate four bits for each iteration of the map. The resulting bitstream passes all statistical tests successfully and has good correlation characteristics as well.

Second, the application of chaotic path planning is considered [14,15]. In path planning, the aim is to design a navigation strategy for an autonomous agent, such as an unmanned ground vehicle (UGV) or an unmanned aerial vehicle (UAV), that explores a given area. The generated path should guarantee efficient area exploration and eventually achieve complete coverage of the terrain. In chaotic path planning, there is an additional aim, that of unpredictability of motion. An unpredictable motion can be advantageous in adversarial situations. Examples include patrolling missions, where an agent must inspect a space for possible intruders. If the agent moves unpredictably, it would be impossible for an intruder to accurately draw a breaching plan. Moreover, from the defender’s perspective, it would be challenging to efficiently trace an agent that moves unpredictably, putting an extra challenge in tracking problems. Similarly, chaotic motion would pose an extra challenge to the defender’s side in the design of camera networks or obstacle placement.

Thus, in all of the above situations, the agent should follow a trajectory that is impossible to predict and looks random, while also ensuring the complete coverage of the given area, as the number of steps increases. The best way to achieve this is to integrate a chaotic system into the path generation algorithm. There are many possible approaches to this problem (see, e.g., [16–20]). In this paper, we consider the application of the proposed bitstream to the generation of commands for the chaotic motion. This method was previously discussed by the authors of [19,21–24] for a robot exploring a 2D area moving in four or eight directions, by taking pairs or triples of bits to generate a motion command. Thus, here we expand this approach for a robot or UAV exploring a 3D area [25–33].

Numerical simulations were performed to evaluate the coverage that can be achieved by the proposed strategy with respect to the number of steps performed. It is shown that the coverage rate converges exponentially to complete coverage, as the number of steps increases. Thus, the proposed strategy gives satisfying results.

Notably, the simulations performed here only tested the path planning strategy itself, without taking into account the dynamics of a UAV that would implement the motion. The integration of the proposed path planning strategy with the model dynamics would surely pose a challenge. In addition, the energy and navigation limitations of a UAV would certainly give rise to errors between the theoretical and actual coverage performance. These are all subjects for future analysis. Implementations of chaotic path planning strategies have been performed previously [22,34–36], but all works consider the 2D case.

The rest of the paper is organized as follows. In Section 2, the proposed map is introduced and analyzed. In Section 3, the PRBG algorithm is presented and statistical tests are performed. In Section 4, the application to path planning is considered. Section 5 concludes the paper with a brief discussion on future research topics.

2. The Proposed Map

The 1D map proposed in this work is the following:

\[ x_i = f(x_{i-1}) = b|x_{i-1}|^r - a \]  

(1)

The map has three parameters and only two terms, so it has a very simple structure, similar to the known tent map [2] and the maps proposed in [37,38], which also utilize absolute nonlinearities, with only two terms. These maps are listed in Table 1. The proposed map differentiates from the given ones, in having additional parameters as well as in including the parametric exponentiation of the term \( |x_{i-1}|^r \). For example, the fourth map proposed in [37] can be considered a special case of (1) for \( a = 1, r = 1 \).
To understand how each parameter affects the behavior of the map, its phase portraits are first plotted. The phase portrait of a discrete map depicts the function \( x_i = f(x_{i-1}) \) with respect to \( x_{i-1} \). This can help identify how each parameter affects the shape of the mapping function \( f \). In Figure 1a, the curve is plotted for \( a = 1, b = 1.99 \), for different values of \( r \). It is seen that the parameter \( r \) in the exponent can control the curvature of \( f \). In Figure 1b, the curve is plotted for \( b = 1.99, r = 1 \), for different values of \( a \). The parameter \( a \) leaves the shape of \( f \) invariant, but translates it lower on the phase space. In Figure 1c, the curve is plotted for \( a = 1, r = 1 \), for different values of \( b \). As a scaling factor of the term \( b|x_{i-1}|^r \) in (1), the parameter \( b \) can affect the slope of \( f \). Note that in all figures the dashed line indicates the bisector, whose intersections with \( f \) indicate equilibrium points \( x^* = x_i = x_{i-1} \).

In addition, two different cobweb diagrams of the map (1) are plotted in Figure 2 for different parameter values. The cobweb diagram shows the successive iterations of the map and can visualize the chaotic behavior of the system, as indicated by the non-periodic succession of points \((x_{i-1}, x_i)\).

To further explore the dynamical behavior of the map, its bifurcation diagrams and corresponding Lyapunov exponent (LE) diagrams with respect to parameters \( r, a \), and \( b \) are plotted. The bifurcation diagram with respect to parameter \( r \), for \( a = 1, b = 1.99 \), is shown in Figure 3. One can see that the system exhibits chaos with a mapping on the interval \([-1, 1]\) for most values of the parameter \( r \) in the range of 1 to roughly 4.7, with some small periodic windows appearing. After 4.7, the system falls into periodic behavior. This behavior is verified by the diagram of Lyapunov exponent shown in Figure 4. The last was computed following the method in [39]. For both diagrams, the initial condition chosen in each iteration is \( x_0 = 0.1 \), which is also the case for all subsequent diagrams.

The bifurcation diagram with respect to parameter \( a \), for \( b = 1.99, r = 1 \), is shown in Figure 5. The system exhibits constant chaos, with a mapping on the interval \([-a,a]\). The constant chaotic behavior is verified by the diagram of its Lyapunov exponent shown in Figure 6.

Similarly, the bifurcation diagram with respect to parameter \( b \), for \( a = 1, r = 1 \), is shown in Figure 7. One can see that the system exhibits chaotic behavior roughly for \( b \in (1,2) \). This fact is verified by the diagram of Lyapunov exponent shown in Figure 8. It is worth noting that parameter \( a \) works as a scaling factor here, changing the mapping interval of the values \( x_i \), as can be seen in Figure 9, where multiple bifurcation diagrams with respect to \( b \), for different values of \( a \) are overlapped. The change of parameter \( a \), however, does not affect the Lyapunov exponent of the system.

To determine the values of nonlinearity parameters that are suitable to generate pseudo-random sequences using map (1), a two-dimensional LE diagram with respect to \( a \) and \( b \), for different values of the parameter \( r \), is plotted in Figure 10. Negative LE values correspond to white color, while positive LE values are marked black. In Figure 10c for \( r = 1 \), it can be seen that the system exhibits steady chaotic behavior with respect to both \( a \in (0.2,3) \) and \( b \in (1,2) \). For lower values \( r = 0.8,0.9 \) and higher values \( r = 1.1,1.2,1.3 \), regions of constant chaotic behavior still exist, but their range is smaller, as shown in Figure 10a,b,d–f. Based on these six graphs, in the rest of the paper, the parameter \( r \) is fixed as \( r = 1 \).

### Table 1. Chaotic maps with absolute value nonlinearity.

| Reference                     | Map                                      | Number of Parameters |
|-------------------------------|------------------------------------------|----------------------|
| Elaydi [2]                    | \( x_i = 1 - b|x_{i-1} - \frac{1}{b}| \) | 1 \((b = 2)\)        |
| San-Um and Ketthong [37]      | \( x_i = |1 - bx_{i-1}| \)               | 1                    |
| San-Um and Ketthong [37]      | \( x_i = | -1 + bx_{i-1}| \)             | 1                    |
| San-Um and Ketthong [37]      | \( x_i = 1 - |bx_{i-1}| \)              | 1                    |
| San-Um and Ketthong [37]      | \( x_i = -1 + b|x_{i-1}| \)             | 1                    |
| Fong-In et al. [38]           | \( x_i = |bx_{i-1} - a| \)               | 2                    |
Figure 1. Phase portrait depicting $x_i$ as a function of $x_{i-1}$ for different values of: (a) $r$, for $a = 1$, $b = 1.99$; (b) $a$, for $b = 1.99$, $r = 1$; (c) $b$, for $a = 1$ and $r = 1$.

Figure 2. Cobweb diagrams of the map (1) for $x_0 = 0.1$ and: (a) $r = 1$, $a = 3$, $b = 1.99$; (b) $r = 3$, $a = 1$, $b = 1.99$.

Figure 3. Bifurcation diagram with respect to parameter $r$, for $a = 1$, $b = 1.99$. 
Figure 4. LE diagram with respect to parameter $r$, for $a = 1, b = 1.99$.

Figure 5. Bifurcation diagram with respect to parameter $a$, for $b = 1.99, r = 1$.

Figure 6. LE diagram with respect to parameter $a$, for $b = 1.99, r = 1$. 
Figure 7. Bifurcation diagram with respect to parameter $b$, for $a = 1$, $r = 1$.

Figure 8. LE diagram with respect to parameter $b$, for $a = 1$, $r = 1$.

Figure 9. Bifurcation diagram with respect to parameter $b$ for different values of $a$, and $r = 1$. 
Figure 10. Two-dimensional LE diagram for the considered map, for: (a) $r = 0.8$; (b) $r = 0.9$; (c) $r = 1$; (d) $r = 1.1$; (e) $r = 1.2$; (f) $r = 1.3$.

3. Application to Pseudo-Random Bit Generation

3.1. The Proposed Generator

Based on the proposed map, a pseudo-random bit generator (PRBG) is proposed using a simple technique, which is outlined as follows:

Step 1. First, the initial value $x_0$ of the proposed map is chosen, along with parameters $b$ and $a$. These parameters constitute the secret keys of the algorithm. In addition, four bit sequences $b_{1,0}, b_{2,0}, b_{3,0}, b_{4,0}$ are initialized.

Step 2. At each iteration, the decimal parts of $\text{mod}(10^3|x_i|, 1)$, $\text{mod}(10^6|x_i|, 1)$, $\text{mod}(10^9|x_i|, 1)$, and $\text{mod}(10^{12}|x_i|, 1)$ are computed and compared to the threshold value of 0.5. Depending on the result, a ‘0’ or ‘1’ is produced and saved in $b_{1,i}$, $b_{2,i}$, $b_{3,i}$, and $b_{4,i}$, respectively.

Step 3. The bit sequences produced are combined into a single bitstream as $B_i = [b_{1,i}, b_{2,i}, b_{3,i}, b_{4,i}]$.

Using the above algorithm, four bits are produced in each iteration, which makes the algorithm faster than traditional algorithms that produce only one bit in each step. Gayathri
and Subashini [40] presented the results of comparing the encryption speed of different algorithms based on discrete chaotic maps. Most of the considered techniques involve generating sequences of pseudo-random bits and then summing them to the plaintext bits using the XOR operation. Following this idea, we implemented a similar encryption scheme based on the proposed generator. As one can see, the approach based on the proposed algorithm demonstrates the best performance, which indicates a high bit generation rate (Table 2).

Table 2. Results of performance comparison.

| Algorithm                  | Performance (B/ms) |
|---------------------------|--------------------|
| Proposed approach         | 2642               |
| Kanso et al. [41]         | 2023               |
| Zhou et al. [42]          | 407.1              |
| Tong et al. [43]          | 181.09548          |
| Xiaojun et al. [44]       | 121.7448027        |
| Fu et al. [45]            | 78                 |
| Chen et al. [46]          | 41                 |
| Wong et al. [47]          | 40.37              |
| Fu et al. [48]            | 11                 |

3.2. Statistical Testing

3.2.1. NIST Tests

Nowadays, the NIST statistical test suite is considered [49] as a proper tool to test the randomness of the produced bitstream. The NIST package consists of 15 tests, for which a $p$-value is returned. If the $p$-value is larger than a chosen significant value $\alpha$, the test is successful. Here, 40 bit sequences of $10^6$ bits are tested, generated for $x_0 = 0.1, a = 1, b = 1.99, \text{ and } r = 1$. The results are shown in Table 3, where it is seen that the bitstream passes all the tests.

Table 3. NIST statistical test results, with $\alpha = 0.01$.

| No. | Statistical Test           | $p$-Value   | Proportion |
|-----|----------------------------|-------------|------------|
| 1   | Frequency                  | 0.392456    | 40/40      |
| 2   | Block Frequency            | 0.875539    | 40/40      |
| 3   | Cumulative Sums            | 0.186566    | 40/40      |
| 4   | Runs                       | 0.484646    | 40/40      |
| 5   | Longest Run                | 0.689019    | 40/40      |
| 6   | Rank                       | 0.534146    | 40/40      |
| 7   | FFT                        | 0.213309    | 40/40      |
| 8   | Non-Overlapping Template   | 0.311542    | 39/40      |
| 9   | Overlapping Template       | 0.275709    | 39/40      |
| 10  | Universal                  | 0.213309    | 39/40      |
| 11  | Approximate Entropy        | 0.964295    | 40/40      |
| 12  | Random Excursions          | 0.637119    | 23/24      |
| 13  | Random Excursions Variant  | 0.162606    | 24/24      |
| 14  | Serial                     | 0.637119    | 40/40      |
| 15  | Linear Complexity          | 0.437274    | 40/40      |
3.2.2. ENT Tests

The ENT statistical test suite consists of six different randomness tests [50]. The test considers byte sequences, and, for each of the tests, the following result is expected for a random sequence:

- Entropy: The entropy of a random sequence should be close to 8.
- Optimum compression: This value should be close to zero.
- Chi square distribution rate: It should be between 10% and 90%.
- Arithmetic mean: It should be close to 127.5.
- Monte Carlo value for $\pi$: It should approximate $\pi$ with a small error.
- Serial correlation coefficient: It should be close to zero for an uncorrelated sequence.

Since the ENT suite is tested on bytes of data rather than bits, a set of $120 \times 10^6$ bits is generated for the same parameters noted above. Then, the sequential pairs of 8 bits are converted to bytes (integers within [0, 255]), leading to a byte stream of length $15 \times 10^6$. The test results are shown in Table 4, and it can be seen that they are successful.

### Table 4. ENT statistical test results.

| No. | Statistical Test                      | Result                           |
|-----|---------------------------------------|----------------------------------|
| 1   | Entropy                               | 7.999987                         |
| 2   | Optimum Compression                   | 0%                               |
| 3   | Chi-Square                            | 15.37%                           |
| 4   | Arithmetic mean                       | 127.4871                         |
| 5   | Monte Carlo value for $\pi$           | 3.1420352 (0.01% error)          |
| 6   | Serial Correlation Coefficient        | 0.000244                         |

3.2.3. Correlation

Moreover, Figure 11 shows the correlation plots for a bitstream of length $10^4$, generated with keys $x_0 = 0.1$, $b = 1.99$, $a = 1$, and $r = 1$. The auto-correlation measures the correlation of a signal with a time shifted (lagged) copy of itself, as a function of the chosen lag [51]. For a random sequence, the auto-correlation should be as close to zero for all time lags, with the exception of a peak appearing at lag zero, which is expected since, at zero lag, the sequence is similar to itself. This is verified in Figure 11a.

Similarly, the cross-correlation measures the similarity between two sequences as a function of their relative time shift. For unrelated sequences, the cross-correlation should be close to zero [6]. This can be verified in Figure 11b–d. In each case, only one key parameter is changed and the rest are fixed. In Figure 11b, the initial condition is chosen as $x_0 = 0.1$ and $\tilde{x}_0 = 0.1 + 10^{-16}$, with $b = 1.99, a = 1, r = 1$. In Figure 11c, the parameter $b$ is taken as $b = 1.99$ and $\tilde{b} = 1.99 - 10^{-15}$, with $x_0 = 0.1, a = 1$. In Figure 11d, the parameter $a$ is taken $a = 1$ and $\tilde{a} = 1 + 10^{-15}$, with $x_0 = 0.1, b = 1.99$.

The zero cross-correlation for two sequences generated from almost identical initial values verifies that the chaotic PRBG inherits the key sensitivity property from the chaotic map it uses as its source. Thus, any small change in the key values will lead to the generation of a totally different bitstream.

3.2.4. Key Space

Regarding the key space, the system has four key parameters, so $\text{Keys} = \{a, b, r, x_0\}$. Thus, for a 16-digit accuracy, the upper bound for the key space is $10^{4\times16} = 10^{64} = (10^3)^{21.3} \approx (2^{10})^{21.3} = 2^{213}$, which is higher than the required bound of $2^{100}$ to resist brute force attacks [52]. It must be noted, however, that, since the system is not chaotic for all parameter values, the acceptable key space is lower. An appropriate choice for the key values can be made using the results in Figure 10 as a guide. Fixing the value of parameter $r = 1$ limits the key space to $10^{3\times16} = 10^{48} \approx 2^{160}$, which is still higher than the required
bound. In the future, more robust chaotic maps will be considered, that is, chaotic maps that exhibit constant chaotic behavior with the absence of periodic windows and coexisting attractors in some neighborhood of the parameter space \[53,54\], to achieve an even higher key space.

![Figure 11](image)

**Figure 11.** (a) Auto-correlation and (b) cross-correlation for \(x_0 = 0.1\) and \(\hat{x}_0 = 0.1 + 10^{-16}\), with \(b = 1.99, a = 1\), (c) cross-correlation for \(b = 1.99\) and \(\hat{b} = 1.99 - 10^{-15}\), with \(x_0 = 0.1, a = 1\), (d) cross-correlation for \(a = 1\) and \(\hat{a} = 1 + 10^{-15}\), with \(x_0 = 0.1, b = 1.99\).

4. Application to Path Planning

4.1. The Proposed Chaotic Motion Generator

In this section, our aim is to apply the proposed PRBG to the problem of chaotic path planning. Most works that consider this problem in 2D surface motion using PRBGs take pairs or triples of bits for motion in four or eight discrete directions, respectively, and generate a series of chaotic motion commands for a robot exploring the area \[19,21–23\]. Here, we consider an adaptation of the above approaches for a UAV exploring a 3D area, using the following steps:

**Step 1.** First, the initial value \(x_0\) of the proposed map is chosen, along with parameters \(b\) and \(a\). These three parameters constitute the secret keys of the algorithm. Then, four-bit sequences \(b_{1,0}, b_{2,0}, b_{3,0},\) and \(b_{4,0}\) are initialized.

**Step 2.** At each iteration, four bits are generated, \(b_{1,i}, b_{2,i}, b_{3,i},\) and \(b_{4,i}\), based on the procedure described in the previous section. Then, the bits \([b_{2,i}, b_{3,i}, b_{4,i}]\) are combined to generate a motion command for the robot in the horizontal plane, using the rules found in Table 5.

**Step 3.** Based on the value of \(b_{1,i}\), the robot moves vertically up for \(b_{1,i} = 0\) or down for \(b_{1,i} = 1\).

Thus, the robot performs two consecutive moves per iteration, one horizontally and one vertically. In the above steps, when a motion command is to bring the UAV outside the given barriers, the motion is discarded and the robot waits for the next command. A similar approach was considered by Moysis et al. \[33\]. In that work, the values of a chaotic map were taken modulo 26, and at each iteration the UAV performed a motion among 26 discrete directions, signifying its adjacent cells. The algorithm we consider here differentiates from the previous one by introducing a statistically random bitstream to generate the motion, as well as by having the robot perform two motions per each iteration, changing its altitude as well.
Table 5. Bit pairs and motion commands on the horizontal plane.

| Bits  | Motion command |
|-------|----------------|
| 000   | up             |
| 100   | up-right       |
| 110   | right          |
| 101   | down-right     |
| 011   | down           |
| 111   | down-left      |
| 001   | left           |
| 010   | up-left        |

Figure 12 shows a simulation of the proposed technique for 10,000 iterations. The area considered is of size $50 \times 50 \times 50$ cells, so there are $50^3$ discrete cells for the robot to visit. The initial position is taken as $[1, 1, 1]$ and the map parameters are $b = 1.99, a = 1$. The coverage is computed by simply dividing the number of visited cells by the number of overall cells. A color-coded graph depicting the number of visits in each cell is shown in Figure 13. Similarly, a simulation for 60,000 iterations is shown in Figure 14, where it can be seen that, compared to Figure 13, the coverage is significantly increased to about 50%.

4.2. Coverage Performance

In order to formally study the relation between coverage and the number of iterations of the algorithm, the average of 50 simulations is considered with an increasing number of steps. In each simulation, the initial condition of the chaotic map and the UAV are chosen randomly. The iterations start from 20,000 steps and go up to 500,000, with a 20,000-step increment, and the results are shown in Figure 15. It can be observed that the coverage converges to 100% exponentially, as resulted from a curve fitting to the curve:

$$y(x) = Ae^{x/z} + y_0$$  \hspace{1cm} (2)

with $A = -99.11941, z = -85,329.59663, y_0 = 99.82191$, and reduced chi-square $R = 0.00664$. Thus, the algorithm has overall good coverage characteristics.

Figure 12. Simulation of motion for 10,000 iterations of the algorithm.
Figure 13. Color-coded simulation for the visits in each cell, for 10,000 iterations.

Figure 14. Color-coded simulation for the visits in each cell, for 60,000 iterations.

Figure 15. Coverage stats with respect to number of iterations.

4.3. Adjusting the Discrete Steps to Smooth Motion

One way to convert the discrete steps taken into smooth motion is through the use of B-Spline curves. B-Spline curves are extensively used in path planning, since
they are easy to generate using a set of points called control points. They also possess properties that make them attractive for path planning, namely affine invariance under linear transformations, strong convex hull property, the local control with respect to changes in the control points, and the smoothness of the curve. Hence, many works use B-Spline curves to design motion trajectories for ground vehicles and UAVs [33,55–57].

To generate a smooth curve, the discrete cells that the UAV visits are used as the control points for a B-Spline curve of \( n \)th degree. The closed form of the curve is

\[
x(t) = \sum_{i=0}^{p} N_n^i(t)P_i
\]

where \( p + 1 \) is the number of control points, which as mentioned is equal to the number of iterations of the algorithm; \( n \) is the degree of the curve; \( N_n^i(t) \) is the B-Spline polynomial basis; \( P_i \) is the visited cells that are used as control points. The curve is defined over the uniform knot sequence

\[
U = [t_0, ..., t_{p+n+1}] = \begin{bmatrix}
0, ..., 0, t_{n+1}, ..., t_p, 1, ..., 1
\end{bmatrix}_{n+1}
\]

An example of a 2D cubic \( (n = 3) \) B-Spline curve is shown in Figure 16. The curve consists of 15 control points, given by

\[
P_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, P_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, P_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, P_3 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, P_4 = \begin{pmatrix} 2 \\ 0 \end{pmatrix}, P_5 = \begin{pmatrix} 3 \\ 0 \end{pmatrix}, P_6 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, P_7 = \begin{pmatrix} 3 \\ 2 \end{pmatrix},
\]

\[
P_8 = \begin{pmatrix} 4 \\ 2 \end{pmatrix}, P_9 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, P_{10} = \begin{pmatrix} 4 \\ 0 \end{pmatrix}, P_{11} = \begin{pmatrix} 5 \\ 1 \end{pmatrix}, P_{12} = \begin{pmatrix} 6 \\ 1 \end{pmatrix}, P_{13} = \begin{pmatrix} 6 \\ 2 \end{pmatrix}, P_{14} = \begin{pmatrix} 7 \\ 1 \end{pmatrix}
\]

From its shape, one can observe that, while the curve interpolates the first and last control points, its shape approximates the polygon defined by the control points, and it also lies inside the convex hull defined by the control points, shown in light green.

![Figure 16. Example of a cubic B-Spline curve consisting of 15 control points.](image)

In Figure 17, a simulation is performed for 500 iterations of the algorithm, that is, 1000 steps along with the corresponding cubic B-Spline curve. The curve is generated after the simulation is performed, using the algorithms found in [38,59].

What must be mentioned is that the performed numerical simulations are just a simple test for the path planning technique, and the UAV navigation dynamics are not taken into account. Thus, a future challenge would be to integrate the proposed algorithm in an experimental real-world implementation.
5. Conclusions

In this study, a simple three-parameter one-dimensional chaotic map was proposed. The map exhibits chaotic behavior for a wide spectrum of parameter values. Based on the proposed map, a pseudo-random number generator was developed, being able to produce 4 bits at each iteration of the map. The proposed PRBG passes all NIST statistical tests and has good correlation characteristics. The PRBG was applied to the problem of chaotic path planning for a UAV exploring a 3D area. The simulation results show good coverage characteristics. Future aspects of this work will consider different path planning techniques, the combination of intelligent approaches to improve coverage, and an experimental implementation. In addition, the study of robust chaotic maps for relevant security applications is of interest.

Author Contributions: Conceptualization: L.M., K.R., A.V.T., Formal Analysis: L.M., A.V.T., Validation: K.R., C.V., D.N.B., Writing: L.M., A.V.T., Supervision: K.R., C.V., D.N.B., B.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: The authors are thankful to the anonymous reviewers for their valuable suggestions during the review process.

Conflicts of Interest: The authors declare they have no conflict of interest.

References

1. Strogatz, S.H. Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering; CRC Press: Boca Raton, FL, USA, 2018.
2. Elaydi, S.N. Discrete Chaos: With Applications in Science and Engineering; CRC Press: Boca Raton, FL, USA, 2007.
3. May, R.M. Simple mathematical models with very complicated dynamics. Nature 1976, 261, 459–467. [CrossRef] [PubMed]
4. de la Fraga, L.G.; Torres-Pérez, E.; Tlelo-Cuautle, E.; Mancillas-López, C. Hardware implementation of pseudo-random number generators based on chaotic maps. Nonlinear Dyn. 2017, 90, 1661–1670. [CrossRef]
5. Hua, Z.; Zhou, B.; Zhou, Y. Sine chaotification model for enhancing chaos and its hardware implementation. IEEE Trans. Ind. Electron. 2018, 66, 1273–1284. [CrossRef]
6. 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. [CrossRef]
7. Irfan, M.; Ali, A.; Khan, M.A.; Ebatisham-ul Haq, M.; Mehmood Shah, S.N.; Saboor, A.; Ahmad, W. Pseudorandom Number Generator (PRNG) Design Using Hyper-Chaotic Modified Robust Logistic Map (HC-MRLM). Electronics 2020, 9, 104. [CrossRef]
8. François, M.; Grosjes, T.; Barchiesi, D.; Erra, R. Pseudo-random number generator based on mixing of three chaotic maps. Commun. Nonlinear Sci. Numer. Simul. 2014, 19, 887–895. [CrossRef]
9. Alawida, M.; Samsudin, A.; Teh, J.S. Enhanced digital chaotic maps based on bit reversal with applications in random bit generators. Inf. Sci. 2020, 512, 1155–1169. [CrossRef]
10. Wang, L.; Cheng, H. Pseudo-Random Number Generator Based on Logistic Chaotic System. Entropy 2019, 21, 960. [CrossRef]
11. Wang, Y.; Liu, Z.; Ma, J.; He, H. A pseudorandom number generator based on piecewise logistic map. *Nonlinear Dyn.* 2016, 83, 2373–2391. [CrossRef]

12. Tutueva, A.V.; Nepomuceno, E.G.; Karimov, A.I.; Andreev, V.S.; Butusov, D.N. Adaptive chaotic maps and their application to pseudo-random numbers generation. *Chaos Solitons Fractals* 2020, 133, 109615. [CrossRef]

13. Muriel-Escobar, M.; Cruz-Hernández, C.; Cardoza-Avendaño, L.; Méndez-Ramírez, R. A novel pseudorandom number generator based on pseudorandomly enhanced logistic map. *Nonlinear Dyn.* 2017, 87, 407–425. [CrossRef]

14. Nakamura, Y.; Sekiguchi, A. The chaotic mobile robot. *IEEE Trans. Robot. Autom.* 2001, 17, 898–904. [CrossRef]

15. Martins-Filho, L.S.; Macau, E.E. Patrol mobile robots and chaotic trajectories. *Math. Probl. Eng.* 2007. [CrossRef]

16. Curicic, D.I.; Volosencu, C. A 2D chaotic path planning for mobile robots accomplishing boundary surveillance missions in adversarial conditions. *Commun. Nonlinear Sci. Numer. Simul.* 2014, 19, 3617–3627. [CrossRef]

17. Li, C.; Song, Y.; Wang, F.; Liang, Z.; Zhu, B. Chaotic path planner of autonomous mobile robots based on the standard map for surveillance missions. *Math. Probl. Eng.* 2015, 2015. [CrossRef]

18. Moysis, L.; Petavratzis, E.; Volos, C.; Nistazakis, H.; Stouboulos, I. A chaotic path planning generator based on logistic map and modulo tactics. *Robot. Auton. Syst.* 2020, 124, 103377. [CrossRef]

19. Volos, C.K.; Kyprianidis, I.M.; Stouboulos, I.N. A chaotic path planning generator for autonomous mobile robots. *Robot. Auton. Syst.* 2012, 60, 651–656. [CrossRef]

20. Nasr, S.; Meikki, H.; Bouallegue, K. A multi-scroll chaotic system for a higher coverage path planning of a mobile robot using flatness controller. *Chaos Solitons Fractals* 2019, 118, 366–375. [CrossRef]

21. Volos, C.K.; Kyprianidis, I.; Stouboulos, I.; Stavrinides, S.; Anagnostopoulos, A. An Autonomous Mobile Robot Guided by a Chaotic True Random Bits Generator. In *Chaos and Complex Systems*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 337–343.

22. Volos, C.; Kyprianidis, I.; Stouboulos, I. Experimental investigation on coverage performance of a chaotic autonomous mobile robot. *Robot. Auton. Syst.* 2013, 61, 1314–1322. [CrossRef]

23. Petavratzis, E.K.; Volos, C.K.; Moysis, L.; Stouboulos, I.N.; Nistazakis, H.E.; Tombras, G.S.; Valavanis, K.P. An Inverse Phenomenon Approach in a Chaotic Mobile Robot’s Path Planning Based on a Modified Logistic Map. *Technologies* 2019, 7, 84. [CrossRef]

24. Volos, C.K.; Prousalis, D.; Vaidyanathan, S.; Pham, V.T.; Munoz-Pacheco, J.; Tielo-Cuatle, E. Kinematic control of a robot by using a non-autonomous chaotic system. In *Advances and Applications in Nonlinear Control Systems*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 1–17.

25. Gohari, P.S.; Mohammadi, H.; Taghvaei, S. Using chaotic maps for 3D boundary surveillance by quadrotor robot. *Appl. Soft Comput.* 2019, 76, 68–77. [CrossRef]

26. Rosalie, M.; Danoy, G.; Chaumette, S.; Bouvry, P. Chaos-enhanced mobility models for multilevel swarms of UAVs. *Swarm Evol. Comput.* 2018, 41, 36–48. [CrossRef]

27. Tawfik, M.A.; Abdulkahab, E.N.; Swadi, S.M. Specific Chaotic System and its Implementation in Robotic Field. *Eng. Technol. J.* 2015, 33, 2231–2243.

28. Samuel, V.M.; Shehata, O.M.; Morgan, E.S.I. Chaos Generation for Multi-Robot 3D-Volume Coverage Maximization. In Proceedings of the 4th International Conference on Control, Mechatronics and Automation, Barcelona, Spain, 7–11 December 2016; ACM: New York, NY, USA, 2016; pp. 36–40.

29. Tharwat, A.; Elhoseny, M.; Hassanien, A.E.; Gabel, T.; Kumar, A. Intelligent Bézier curve-based path planning model using Chaotic Particle Swarm Optimization algorithm. *Clust. Comput.* 2019, 22, 4745–4766. [CrossRef]

30. Curicic, D.I.; Volosencu, C. Path planning algorithm based on Arnold cat map for surveillance UAVs. *Def. Sci. J.* 2015, 65, 483–488. [CrossRef]

31. Zhou, Z.; Duan, H.; Li, P.; Di, B. Chaotic differential evolution approach for 3D trajectory planning of unmanned aerial vehicle. In Proceedings of the 2013 10th IEEE International Conference on Control and Automation (ICCA), Hangzhou, China, 12–14 June 2013; pp. 368–372.

32. Mohanta, J.; Parhi, D.R.; Mohanty, S.; Keshari, A. A control scheme for navigation and obstacle avoidance of autonomous flying agent. *Arab. J. Sci. Eng.* 2018, 43, 1395–1407. [CrossRef]

33. Moysis, L.; Petavratzis, E.; Volos, C.; Nistazakis, H.; Stouboulos, I.; Valavanis, K. A Chaotic Path Planning Method for 3D Area Coverage Using Modified Logistic Map and a Modulo Tactic. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 1–4 September 2020; pp. 220–227.

34. He, H.; Cui, Y.; Lu, C.; Sun, G. Time Delay Chen System Analysis and Its Application. In Proceedings of the International Conference on Mechanical Design, Huzhou, China, 12–14 August 2019; Springer: Singapore, 2019; pp. 202–213.

35. Sambas, A.; Vaidyanathan, S.; Mamat, M.; Sanjaya, W.M.; Rahayu, D.S. A 3-D novel jerk chaotic system and its application in secure communication system and mobile robot navigation. In *Advances and Applications in Chaotic Systems*; Springer: Cham, Switzerland, 2016; pp. 283–310.

36. Vaidyanathan, S.; Sambas, A.; Mamat, M.; Sanjaya, W.M. A new three-dimensional chaotic system with a hidden attractor, circuit design and application in wireless mobile robot. *Arch. Control Sci.* 2017, 27, 541–554. [CrossRef]

37. San-Um, W.; Kethpong, P. The generalization of mathematically simple and robust chaotic maps with absolute value nonlinearity. In Proceedings of the TENCON 2014-2014 IEEE Region 10 Conference, Bangkok, Thailand, 22–25 October 2014; pp. 1–4.
38. Fong-In, S.; Kiattisin, S.; Leelasantitham, A.; San-Um, W. A partial encryption scheme using absolute-value chaotic map for secure electronic health records. In Proceedings of the 4th Joint International Conference on Information and Communication Technology, Electronic and Electrical Engineering (IJCTEE), Chiang Rai, Thailand, 5–8 March 2014; pp. 1–5.

39. Bovy, J. Lyapunov Exponents and Strange Attractors in Discrete and Continuous Dynamical Systems; Theoretical Physics Project; Technical Report; Katholieke Universiteit Leuven: Leuven, Belgium, 2004; Volume 9, pp. 1–19.

40. Gayathri, J.; Subashini, S. A survey on security and efficiency issues in chaotic image encryption. Int. J. Inf. Comput. Secur. 2016, 8, 347–381.

41. Kanso, A.; Ghebleh, M. A novel image encryption algorithm based on a 3D chaotic map. Commun. Nonlinear Sci. Numer. Simul. 2012, 17, 2943–2959. [CrossRef]

42. Zhou, Y.; Bao, L.; Chen, C.P. A new 1D chaotic system for image encryption. Signal Process. 2014, 97, 172–182. [CrossRef]

43. Tong, X.J. Design of an image encryption scheme based on a multiple chaotic map. Commun. Nonlinear Sci. Numer. Simul. 2013, 18, 1725–1733. [CrossRef]

44. Tong, X.; Cui, M. Image encryption scheme based on 3D baker with dynamical compound chaotic sequence cipher generator. Signal Process. 2009, 89, 480–491. [CrossRef]

45. Fu, C.; Chen, J.J.; Zou, H.; Meng, W.H.; Zhan, Y.F.; Yu, Y.W. A chaos-based digital image encryption scheme with an improved diffusion strategy. Opt. Express 2012, 20, 2363–2378. [CrossRef]

46. Chen, J.X.; Zhu, Z.L.; Yu, H. A fast chaos-based symmetric image cryptosystem with an improved diffusion scheme. Optik 2014, 125, 2472–2478. [CrossRef]

47. Wong, K.W.; Kwok, B.S.H.; Yuen, C.H. An efficient diffusion approach for chaos-based image encryption. Chaos Solitons Fractals 2009, 41, 2652–2663. [CrossRef]

48. Fu, C.; Meng, W.H.; Zhan, Y.F.; Zhu, Z.L.; Lau, F.C.; Chi, K.T.; Ma, H.F. An efficient and secure medical image protection scheme based on chaotic maps. Comput. Biol. Med. 2013, 43, 1000–1010. [CrossRef]

49. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications; Technical Report; Booz-Allen and Hamilton Inc. Mclean Va: Gaithersburg, MD, USA, 2001.

50. Walter, J. ENT: A Pseudo Random Number Sequence Test Program. 2008. Available online: https://www.fourmilab.ch/random/ (accessed on 22 July 2021).

51. Zhang, Z.; Wang, Y.; Zhang, L.Y.; Zhu, H. A novel chaotic map constructed by geometric operations and its application. Nonlinear Dyn. 2020, 102, 2843–2858. [CrossRef]

52. Alvarez, G.; Li, S. Some basic cryptographic requirements for chaos-based cryptosystems. Int. J. Bifurc. Chaos 2006, 16, 2129–2151. [CrossRef]

53. Zeraoulia, E. Robust Chaos and Its Applications; World Scientific: Singapore, 2012; Volume 79.

54. Hua, Z.; Zhou, Y. Exponential chaotic model for generating robust chaos. IEEE Trans. Syst. Man, Cybern. Syst. 2019, 51, 3713–3724. [CrossRef]

55. Foo, J.L.; Knutzon, J.; Kalivarapu, V.; Oliver, J.; Winer, E. Path planning of unmanned aerial vehicles using B-splines and particle swarm optimization. J. Aerosp. Comput. Inform. Commun. 2009, 6, 271–290. [CrossRef]

56. Koyuncu, E.; Inalhan, G. A probabilistic B-spline motion planning algorithm for unmanned helicopters flying in dense 3D environments. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 815–821.

57. Jung, D.; Tsiotras, P. On-line path generation for unmanned aerial vehicles using B-spline path templates. J. Guid. Control. Dyn. 2013, 36, 1642–1653. [CrossRef]

58. Moysis, L. Introduction to Computer Aided Geometric Design—A Student’s Companion with Matlab Examples; MathWorks, Inc.: Natick, MA, USA, 2018

59. Piegl, L.; Tiller, W. The NURBS Book; Springer: Berlin/Heidelberg, Germany: 2012.