Peer-reviewed version available at *Sensors* **2019**, *20*, 83; doi:10.3390/s20010083

Article

A Multidimensional Hyperjerk Oscillator: Dynamics Analysis, Analogue and Digital Implementation, and its Application as a Cryptosystem

Tsafack Nestor [1,2], Nkapkop Jean De Dieu [3,4], Kengne Jacques [1], Effa Joseph Yves [3], Abdullah M. Iliyasu[5,6,7,*], Ahmed A. Abd El-Latif [8,9,10]

1 Research Unit of Laboratory of Automation and Applied Computer (LAIA), Electrical Engineering Department of IUT-FV, University of Dschang, P.O. Box 134, Bandjoun, Cameroon

2 Research Unit of Laboratory of Condensed Matter, Electronics and Signal Processing (URMACETS) Department of Physics, Faculty of Sciences, University of Dschang, P.O. Box 67, Dschang, Cameroon

3 Department of Physics, University of Ngaoundéré, P.O. Box 454, Ngaoundéré, Cameroon

4 Department of Communications, Technical University of Cluj-Napoca, 26-28 Baritiu Street, 400027 Cluj-Napoca, Romania

5 Electrical Engineering Department, College of Engineering, Prince Sattam Bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia

6 School of Computing, Tokyo Institute of Technology, Yokohama 226-8502, Japan

7 School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China

8 Mathematics and Computer Science Department, Faculty of Science, Menoufia University, P.O. Box 32511, Shebin El-Koom, Egypt

9 School of Information Technology and Computer Science, Nile University, Egypt;

10 School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150080, China

* Correspondence: (A.M.I.) a.iliyasu@psau.edu.sa; abdul-m-elias@ieee.org; +966-115-888-8259

Abstract: A lightweight image encryption algorithm based on chaos induction via a 5-dimensional hyperjerk oscillator (5DHO) is presented. First, the dynamics of our 5DHO network is investigated and shown to exhibit up to five coexisting hidden attractors in the state space that depend exclusively on the system's initial values. Further, a simple implementation of the circuit was used to validate its ability to exhibit chaotic dynamical properties. Second, an Arduino UNO platform is used to confirm the usability of our oscillator in digital implementation of the system. Finally, an efficient image encryption application is executed using the proposed chaotic networks based on the use of permutation-substitution sequences. The superior qualities of the proposed strategy are traced to the dynamic set of keys used in the substitution process which heralds the generation of the final ciphered image. Based on the results obtained from the entropy analysis (7.9976), NPCR values (99.62), UACI tests (33.69) and encryption execution time for 512x512 images (0.1179 sec), the proposed algorithm is adjudged to be fast and robust to differential and statistical attacks relative to similar approaches.

Keywords: Hyperjerk oscillator; Arduino board; multiple coexisting attractors; information security; image encryption

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

1. Introduction

A lot of development in internet and multimedia technology have been witnessed over the past decade. This has facilitated seamless exchange and transfer of confidential information. The confidentiality, authentication and integrity (CIA) triad is widely cited as the cornerstone of information security. Among others, it provides the effective copyright protection and confidentiality needed in business, entertainment, healthcare, military, etc. communication. Encryption of sensitive data is one of the most important information security strategies and is necessary for confidentiality. Available encryption and decryption algorithms used to forestall malicious attacks from unauthorised parties include DES, 3-DES, AES, IDEA, RSA, etc. [1-3]. However, due to data capacity resource demands and high correlation among pixels in image files, these standard algorithms withered in providing efficient protection for images [4]. Moreover, although widely used in cryptanalysis, computer science and electrical engineering, pseudo random number generators (PRNG) are less effective in cryptography. As a solution, chaos-based protocols have continued to gain traction in mitigating image security issues [5-11].

Many studies have focused on ergodicity, deterministic dynamics, unpredictable behaviors, non-linear transformation, sensitivity dependence, etc. of the system. Research efforts have explored the use of striking periodic attractors, chaotic attractors or hyperchaotic attractors, antimonotonicity, period doubling, hysteresis, coexisting bifurcators, etc. in investigating the dynamic behaviours of systems and their possible applications [12-17]. Interestingly, some of these characteristics have been found useful in image encryption [18, 6]. In [19], Shuqin and collaborators presented a novel encryption algorithm based on chaos and SHA-256 whose experimental results show that it was efficient and reliable. This was further enhanced in [12], wherein Quing et al. proposed an S-box design algorithm based on a new compound chaotic system. In their effort, in [20], Biham et al. demonstrated the exploitation of the weakness inherent to piecewise linearity of the tent map and its limitation to 75 random bits to violate the intensity of the system using a pair of known and chosen plain text attacks. Similarly, in [21], Baptista suggested the use of a chaotic attractor, plaintext and logistic map for image encryption.

Arduino is one of the most widely used open-source computer hardware and software products used in education, craftsmanship, etc. It is an electronic card based on a microcontroller that defines a variety of development board packages including Arduino Uno. The Arduino Uno is based on the ATmega328 microcontroller [22]. It consists of 14 digital input/output pins where six of them are configured as Pulse Width Modulation (PWM) outputs. Furthermore, it is provided with six analog inputs, a 16-MHz crystal oscillator, a USB connection, a power jack and an in-circuit serial programming header. The Arduino Uno can be powered via USB connection or with an external power supply (i.e., a 9-V battery). The power source is selected automatically. The board can operate on an external supply of 6 to 20 volts. The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library). Each of the 14 digital pins on the Uno can be used as an input or output, using *pinMode()*, *digitalWrite()*, and *digitalRead()* functions, which operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20–50 kilohms. In addition, some pins have specialized functions: Pins 0 and 1 may be used to receive (RX) and transmit (TX) TTL serial data; Pins 2 and 3 can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value; finally, Pins 3, 5, 6, 9, 10, and 11 provide 8-bit PWM output with the *analogWrite()* function. The Uno has six analog inputs, labeled A0 through A5, each of which provides 10 bits of resolution (1,024 different values). By default, they measure from ground to 5 volts, although is it possible to change the upper end of their range using the AREF pin and the *analogReference()* function [23].

Due largely to its relatively cheap pricing and utility, a large community has arisen around the Arduino idea and through it many hardware and hundreds of free scripts for different projects are easily available. In [22], Mauricio et al., presented a communication system based on chaotic logistic maps and an experimental realization of it using Arduino board. Therein, the input message was

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

moderated using a Delta modulator and encrypted using a logistic map. The key signal is also encrypted using the same logistic map but with different initial conditions. On the receiver side, the binary-coded message is decrypted using the encrypted key signal that is sent through a communication channel. In [23], Leonardo presented a modified discrete-time chaotic system obtained from the standard logistic map model and its implementation using the Arduino-UNO board was demonstrated. Similarly, Van et al. introduced a new chaotic map, which can be considered as a system with hidden attractors [24]. Feasibility of the approach was illustrated via realisation of the map using an open-source electronic platform.

The important gains emanating from the highlighted above provide the foundation of our study whose contributions are enunciated in the sequel.

1.1 *Our contributions*

In this study, we present a multidimensional oscillator (5DHO) network for use as a chaos generator and cubic nonlinearity to the network in [25]. This choice utilises semiconductor diodes rather than analogue multipliers. Specifically, we utilise a network of diode operational amplifiers and resistors to derive a piecewise linear (PWL) approximation of the cubic and quadratic functions needed for chaotic non-linearity [26]. Therefore, low cost, convenient circuits whose output is the square or cube of their input are used to realise 5-D hyperjerk characteristics. Additionally, an Arduino UNO board is used to establish the dynamics of our oscillator and its usability in digital technologies.

Finally, a lightweight encryption algorithm is designed based on permutation-substitution boxes and the sequences of the 5DHO. Our strategy offers a dynamic set of keys for use in generating the ciphered image.

The remainder of paper is structured as follows: Section 2 introduces the dynamics of the proposed multidimensional hyperjerk oscillator. Analogue and digital implementations of the proposed network are presented in Section 3. Following that, Section 4 presents our proposed encryption and decryption procedures as well as their performance analysis are also reported.

2. Dynamics of the proposed Multidimensional Hyperjerk Oscillator

2.1. Mathematical formulation of proposed 5-D hyperjerk oscillator network

The mathematical model of the proposed 5-D hyperjerk system is formalised in the set of differential equations in (1).

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = bx_5 \\ \dot{x}_5 = -a_0x_5 - a_1x_3 - a_2x_2 - a_3x_1 - y \end{cases} \tag{1}$$

where $y = a_4x_4(x_4 - l_1)(x_4 - l_2) = a_4l_1l_2x_4 - a_4(l_1 + l_2)x_4^2 + a_4x_4^3$ is the nonlinear function containing both cubic and quadratic nonlinearities. In the present study, these nonlinearities are implemented without any analog multiplier. $x_i (i = 1, 2, 3, 4, 5)$ are state variables and $a_0 \in [0.8; 2]$, $a_1 \in [2.7; 6]$, $a_2 \in [1; 5]$, $a_3 \in [0.1; 1.5]$, $a_4 \in [0.1; 1.5]$, $l_1 \in [0; 3]$, $l_2 \in [0; 3]$ are positively valued constants.

The fourth order Runge-Kutta algorithm will be used with a trifling integration step to analyze the behaviour of the 5-D hyperjerk system in (1) through Hopf bifurcation diagrams, Lyapunov exponents and phase space trajectories. Meanwhile, the transient is canceled out during the integration and an analogue electronic circuit will be used to illustrate the practical realisation of our chaotic network. Finally, the electronic DSP shield (ADSP-BF533) will be used as a source to induce

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

chaos required to build our robust image cryptosystem. This latter process is illustrated using VisualDSP++ environment.

2.2. Fixed point and stability

Since the unique equilibrium (origin) point ($O(0,0,0,0,0)$) of the proposed model in (1) is the solution of the nonlinear system: $\dot{x}_1 = \dot{x}_2 = \dot{x}_3 = \dot{x}_4 = \dot{x}_5 = 0$, then the stability of the equilibrium can be described by the following characteristic equation:

$$\lambda^5 + a_0\lambda^4 + a_4 b l_1 l_2 \lambda^3 + a_1 b\lambda^2 + a_2 b\lambda + a_3 b = 0 \quad (2)$$

As the real parts of the correlated eigen values are always negatively valued; hence, the equilibrium is stable for the entire region of system parameters. For instance, if we set $b=3$; $a_0=1.5$; $a_1=3$; $a_2=2$; $a_3=1$; $a_4=1$; $l_1=1$; $l_2=2.6$ then its eigen values can be calculated as:

$$\begin{aligned} &\lambda_1 = -0.1194 + 2.5677i;\ \lambda_2 = -0.1194 - 2.5677i;\ \lambda_3 = -0.8425 + 0.0000i \\ &\lambda_4 = -0.2094 + 0.7036i;\ \lambda_5 = -0.2094 - 0.7036i \end{aligned} \quad (3)$$

It is a general conclusion that since the equilibrium point is always stable, it can be predicted that a point attractor coexists with a strange attractor [25]. This is further clarified later in Figure 6.

2.3. Bifurcations and multistability

The bifurcations of an oscillator with respect to parameter a_2 can be investigated when parameters values are assigned as $(a_0, a_1, a_3, a_4, b, l_1, l_2)=(1.5, 2.6, 1, 0.6, 3, 1, 2.6)$. Figure 1a shows that the oscillator studied in our study exhibits the reverse period doubling paths to chaos with the primary value $(6,0,0,0,0)$. It is patent that the oscillator experiences antimonotonicity behaviour. Lyapunov spectrum [27-29] is also used to attest the chaotic dynamics of the system (Figure 1b). Figure 2 provides four views of the 5-D Hyperjerk chaotic attractor where the stable equilibrium point is shown as a red dot.

Coexisting bifurcation diagrams are used to illustrate the phenomenon of multistability in the system (1) is the (see Figure 3 and Figure 5). We note that these graphs are the plots of the local maximums of the variable x_1 against parameter a_2.

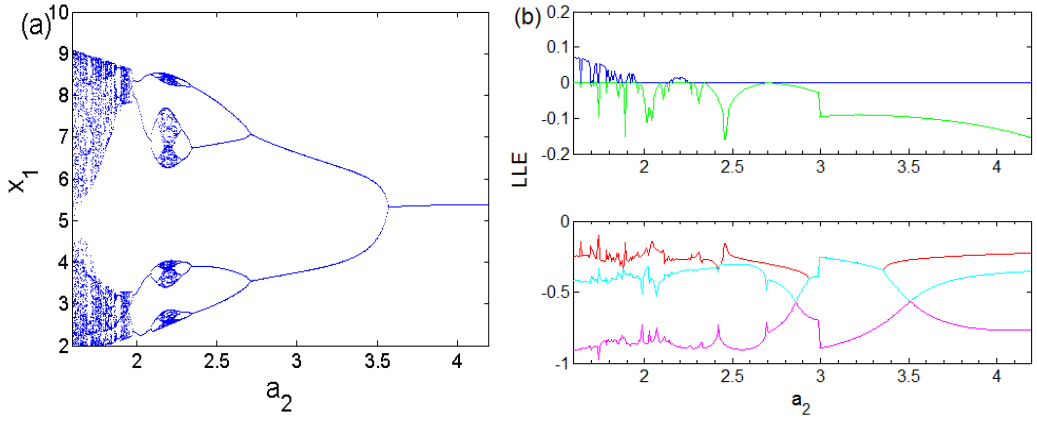


Figure 1: Dynamics of the 5-D oscillator for conditions $(a_0, a_1, a_3, a_4, b, l_1, l_2)=(1.5, 2.6, 1, 0.6, 3, 1, 2.6)$

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

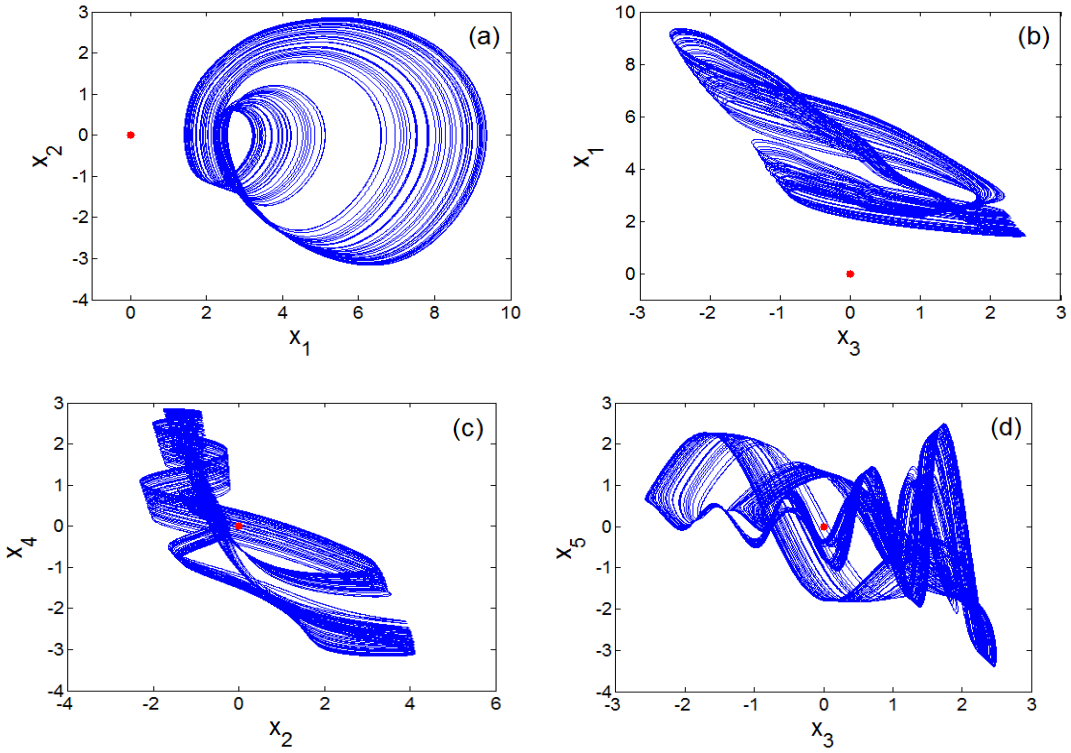


Figure 2: Four views of the 5D Hyperjerk attractor with stable equilibrium point shown as a red dot.

Table 1: Strategies used to obtain coexisting bifurcations of Figure 7 and Figure 10

Figure number	Graph colour	Parameter range	Sweeping Direction	Initial state (x1(0), x2(0), x3(0), x4(0), x5(0))
7	Green	$2 \le a_2 \le 4$	Downward	(4,0,0,0,0,0)
	Red	$2 \le a_2 \le 4$	Downward	(4.4,0,0,0,0,0)
	Blue	$3.212 \le a_2 \le 4$	Upward	(5.2,0,0,0,0,0)
	Black	$2 \le a_2 \le 4$	Downward	(0.4,0,0,0,0,0)
10	Green	$3.405 \le a_2 \le 3.454$	Downward	(4,0,0,0,0,0)
	Red	$3.405 \le a_2 \le 3.454$	Downward	(4.4,0,0,0,0,0)
	Blue	$3.405 \le a_2 \le 3.454$	Upward	(5.2,0,0,0,0,0)
	Black	$3.405 \le a_2 \le 3.454$	Downward	(0.4,0,0,0,0,0)
	Cyan	$3.405 \le a_2 \le 3.454$	Downward	(1.2,0,0,0,0)

Table 2: Numerical initial conditions for multistability analysis for selected parameters

$(a_0, a_1, a_3, a_4, b, l_1, l_2)=(1.5,3,1,1,3,1,2.6)$

Figure number	Type of coexistence	Control Parameter (a_2)	Numerical initial conditions
8	One cycle and a chaotic attractor with fixed point	2.9	(4,0,0,0,0,0), (6,0,0,0,0,0)
11	Three different limit cycles and a chaotic attractor with fixed point	3.454	(a) (0.4,0,0,0,0) , (1.2,0,0,0,0) ; (b) (5.2,0,0,0,0) ; (c) (4,0,0,0,0); (d) (4.4,0,0,0,0)

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

While details of strategies used are presented in Table 1, it is evident from these graphics that up to four attractors can coexist (Figures 4 and 6). Table 2 provides initial solution and system parameters in each case. Offset boosting is another striking behaviour observed in the system presented in (1). For illustration, (1) is rewritten by replacing the state x_1 with x_1+k as presented in (4).

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = bx_5 \\ \dot{x}_5 = -a_0x_5 - a_1x_3 - a_2x_2 - a_3(x_1 + k) - a_4x_4(x_4 - l_1)(x_4 - l_2) \end{cases} \tag{4}$$

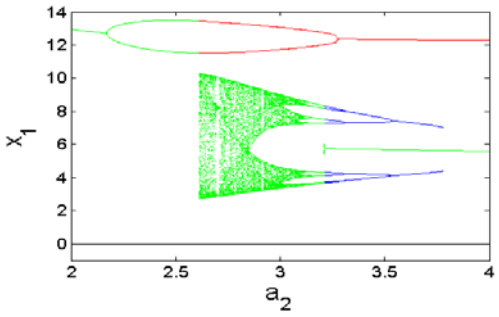


Figure 3: Dynamics of the 5-D Hyperjerk oscillator illustrating multistability.

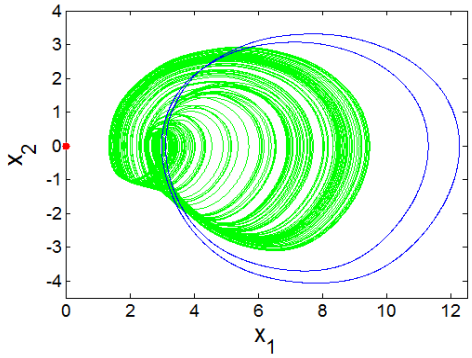


Figure 4: Evidence of strange attractor coexisting with period-2 limit cycle.

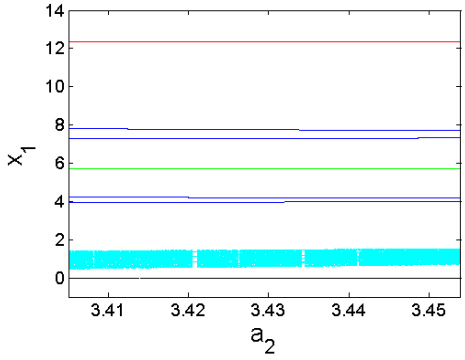


Figure 5: Blow-up of bifurcation plot in Figure 3 for the range $3.405 \leq a_2 \leq 3.455$.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

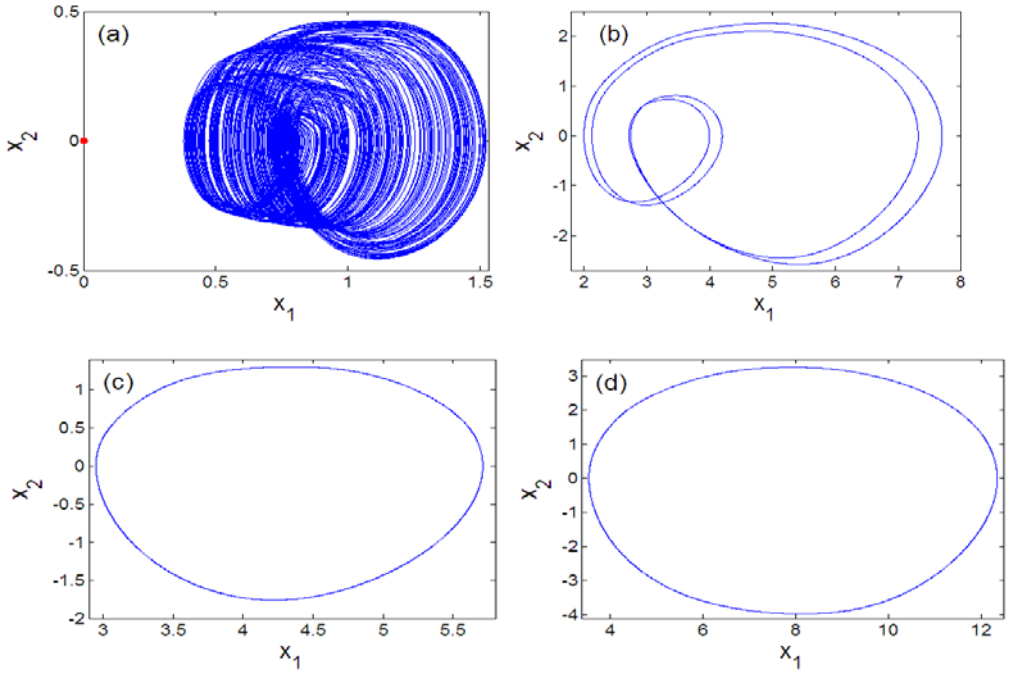


Figure 6: Illustration of coexistence of strange atrractor with limit cycles.

When switching parameter *k*, chaotic signal x_1 can be transferred from a bipolar signal to a unipolar signal as illustrated in Figure 7.

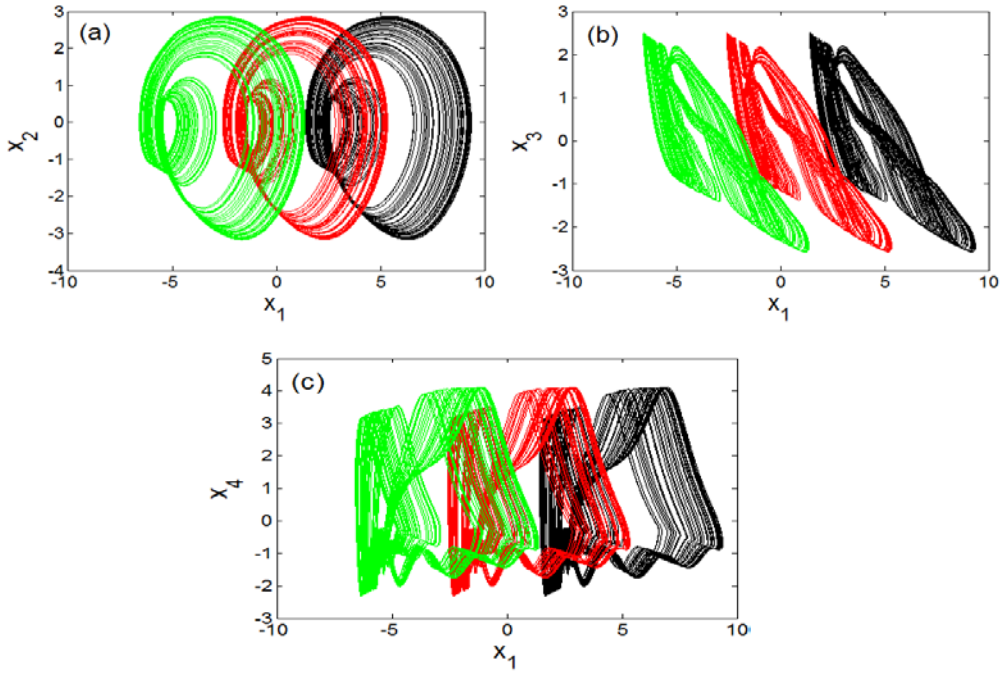


Figure 7: Offset-boosting of chaotic attractor for varying control parameter (*k*) values.

3. Experimental Analysis of proposed oscillator

3.1. Analogue simulation results on the designed circuit using Spice

This section highlights the intricacies of designing and simulating our proposed 5-D hyperjerk oscillator on an analogue computer. We start by recalling that this system exhibits both cubic and quadratic polynomials. The circuit in Figure 10 produces an output that is the square of its input, while the one in Figure 11 implements a PWL approximation of a circuit whose output is the cube of

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

its input. These circuits are convenient for low-cost analogue realisation of our proposed network that is depicted in Figure 9. Here, state variables x_i (i=1….5) of the system in (1) are associated with the voltages v_i (i=1….5) across the capacitors C_i (i=1….5) respectively. By linking the state variable x_i (i=1…5) with the voltages v_i (i=1…5) across the capacitors C_i (i=1…5), we derive circuit equations in the form presented in (5).

$$\begin{cases} C_1\dfrac{d\upsilon_1}{dt}=\dfrac{\upsilon_2}{R} \\ C_2\dfrac{d\upsilon_2}{dt}=\dfrac{\upsilon_3}{R} \\ C_3\dfrac{d\upsilon_3}{dt}=\dfrac{\upsilon_4}{R} \\ C_4\dfrac{d\upsilon_4}{dt}=\dfrac{\upsilon_5}{R_b} \\ C_5\dfrac{d\upsilon_4}{dt}=-\dfrac{\upsilon_5}{R_{a0}}-\dfrac{\upsilon_3}{R_{a1}}-\dfrac{\upsilon_2}{R_{a2}}-\dfrac{\upsilon_1}{R_{a3}}-\dfrac{\upsilon_4}{R_1}-\dfrac{\upsilon_4^2}{R_2}+\dfrac{\upsilon_4^3}{R_3} \end{cases} \tag{5}$$

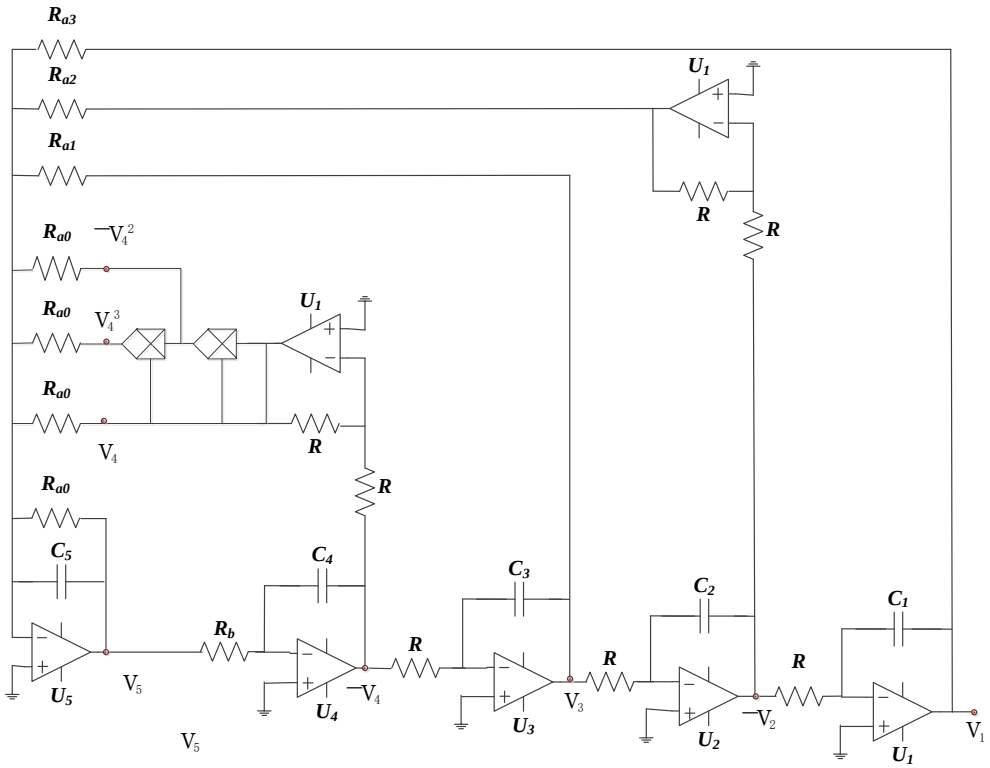


Figure 9: Circuit design for implementation of proposed 5-D hyperjerk system

Similarly, by rescaling time and other variables: $t_e = tRC;\ \upsilon_i = x_i n V_T\ (i=1,2,3)$ and subject to adjustments in parameter values in (6), the system in (5) can be seen to be identical to the one in (1).

$$a_0=\frac{R}{R_{a_0}};\ a_1=\frac{R}{R_{a_1}};\ a_2=\frac{R}{R_{a_2}};\ a_3=\frac{R}{R_{a_3}};\ a_4 l_1 l_2=\frac{R}{R_1};\ a_4(l_1+l_2)=\frac{R}{R_2};\ a_4=\frac{R}{R_3}; \tag{6}$$

Pspice simulation is used to validate the theoretical expectations of the circuit in Figure 9 in terms of coexistence of hidden attractors. R_{a_2} is used as the main control resistor and the rest of circuit components are fixed as mentioned in Table 3. The synergy between the theoretical results (i.e. in Figure 2 and Figure 4) and the Pspice simulation results (in Figure 12 and Figure 13) shows

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

the feasibility of the suggested chaotic system with hidden attractors based on the stated electronic components.

Table 3: Component values used in circuit simulation analysis.

Components	Property	Rating
R	Resistance	10 KΩ
R_{a0}	Resistance	6.66 KΩ
R_{a1}	Resistance	3.33 KΩ
R_{a2}	Resistance	3.5 KΩ
R_{a3}	Resistance	10 KΩ
R_b	Resistance	3.33 KΩ
R_1	Resistance	3.85 KΩ
R_2	Resistance	277.77 KΩ
R_3	Resistance	0.1 KΩ
C_i (i=1,…5)	Capacitance	$10\eta F$
U_i (i=1,…5)	Operational Amplifier	TL084

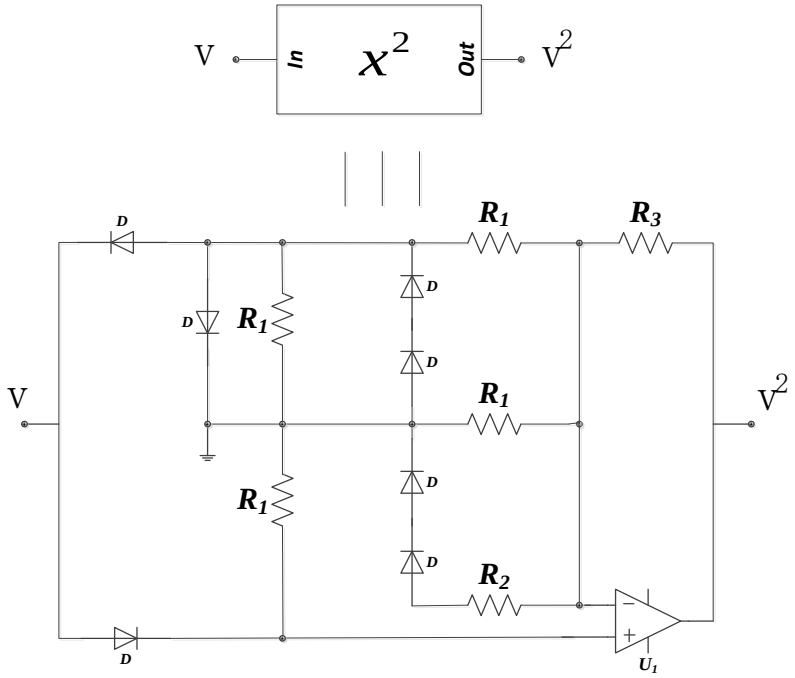


Figure 10: Design of square nonlinearity for circuit values $R_1 = 10k\Omega;\quad R_2 = 4k\Omega;\quad R_3 = 30k\Omega.$.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

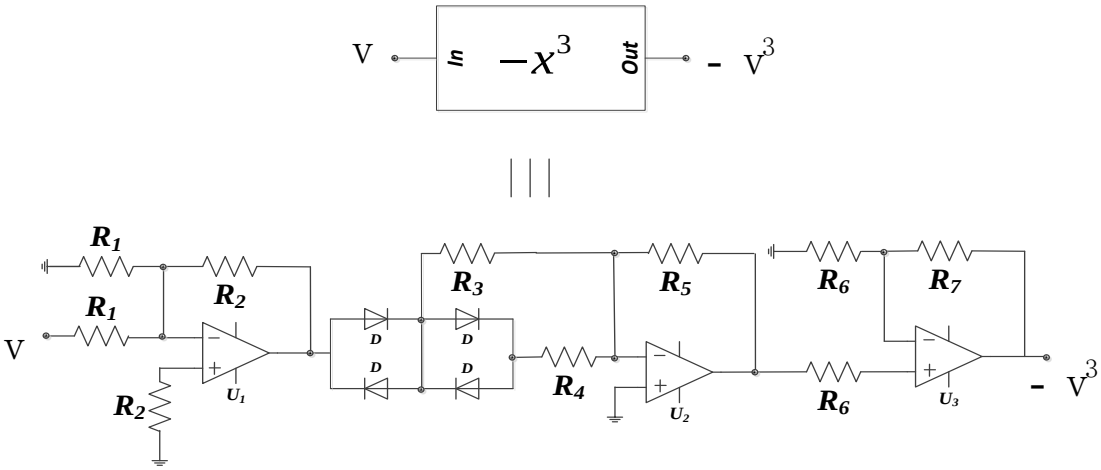


Figure 11: Design of the cube function for circuit values $R_1 = 200k\Omega;\quad R_2 = 100k\Omega;$ $R_3 = 12k\Omega;\quad R_4 = 2k\Omega;\quad R_5 = 15k\Omega;\quad R_6 = 10k\Omega.$.

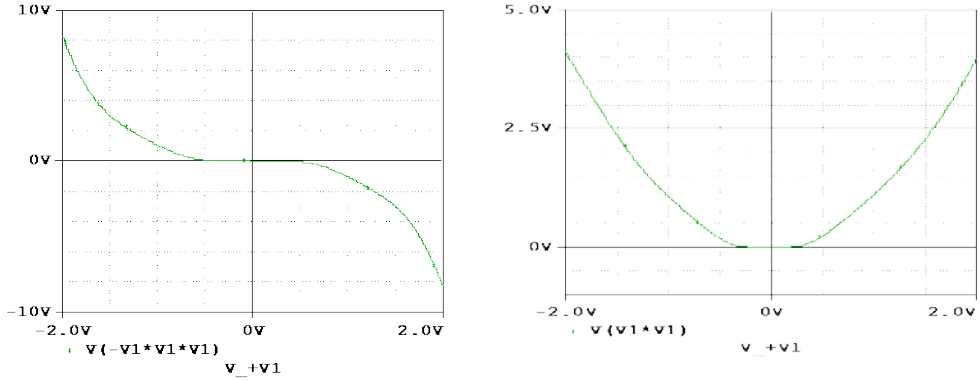


Figure 12: Transfer functions for the cube square functions as obtained Pspice simulation of the circuits in Figure 10 and Figure 11

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

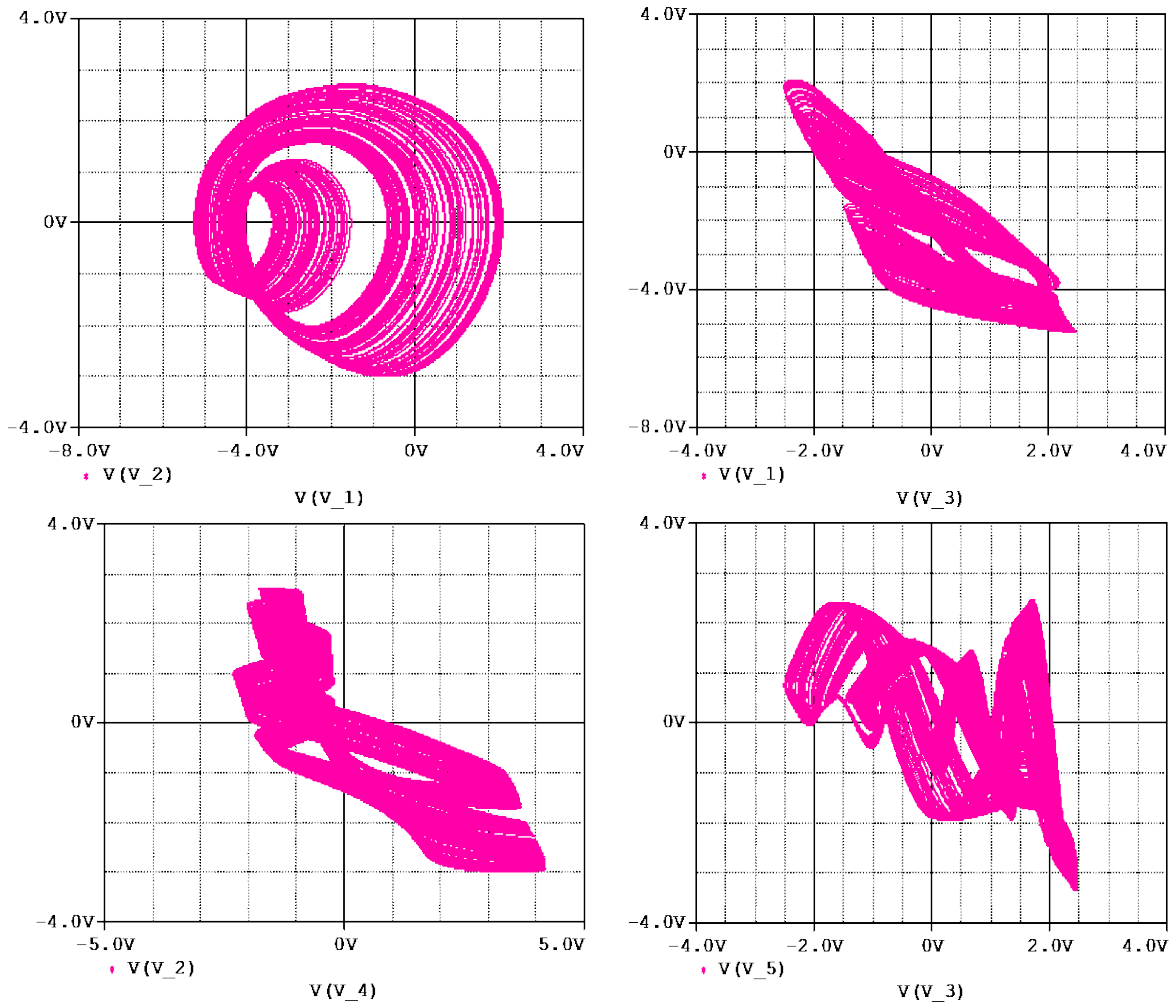


Figure 12: Phase plots for of the proposed 5D Hyperjerk system as observed via Pspice simulation of the network for component values listed in Table 3 and initial conditions set at (1, 1, 1, 1, 1).

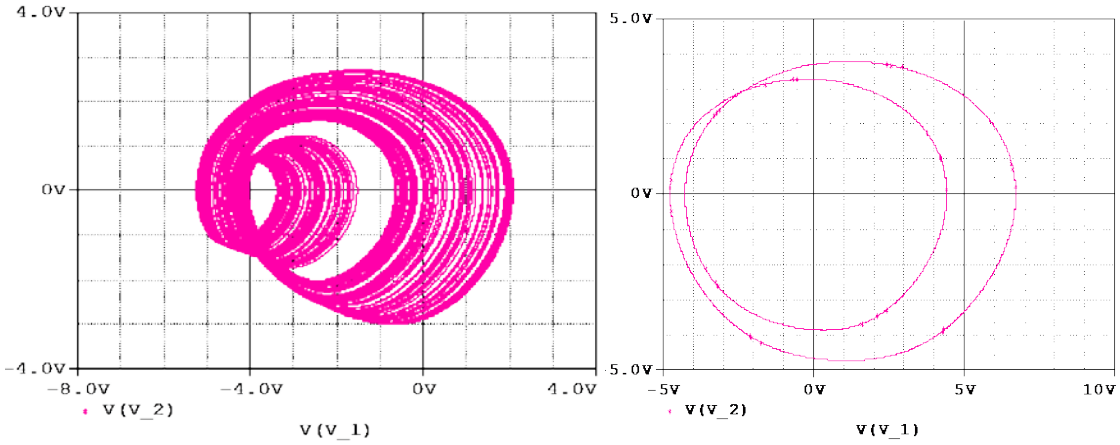


Figure 13: Evidence of presence of coexistence of hidden chaotic attractor with a hidden limit cycle as observed via Pspice simulation of the network for component values listed in Table 3 and initial conditions are set at (1, 1, 1, 1, 1) and (10, 0, 0, 0, 0).

3.2. Arduino based realisation of proposed oscillator

While Field Programmable Gate Arrays (FPGA) are the popular option for providing configurable circuits practical implementation of embedded systems using chaos [44], recently some microcontroller-based chaotic systems have been considered due to their equally flexible and cheap

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

pricing for different programming applications [47-50]. In this study, we use an Arduino UNO board platform to compute and visualise the solutions (for example, using an oscilloscope) of our chaos generator. The Arduino board used in our study is presented in Figure 14 and further details pertaining to its implementation are outlined in the sequel.

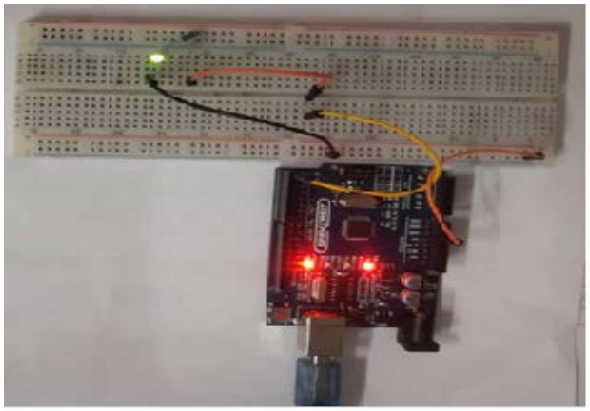


Figure 14: Chaos generation via experimental operation using Arduino Uno board.

Step 1: Set pins 1 and 2 as outputs. The solutions of our chaotic oscillator will be written here.
Step 2: Define the discrete chaotic oscillator, its parameters and initial conditions under an infinite loop.
Step 3: Write the solutions of the discrete chaotic oscillator on Arduino pins. Pin 1 is activated when x2>0.5 and pin 2 is activated when x1>1.

The above algorithm is executed using the open-source platform Arduino 1.8.9 and the experimental result (in Figure 15) is recovered via traces on an oscilloscope connected at pin 2 with scales set at X = 2 V/div and Y = 500 ms/div.

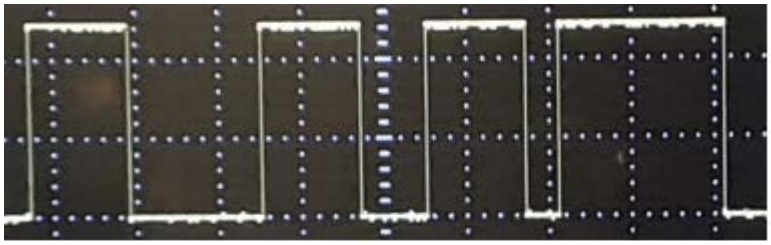


Figure 15: Experimental result at pin 5 of Arduino Uno

Meanwhile, by using "*millis()*"function on the Arduino platform, the experimental time can be printed. Here, a readout of 0.064ms for a 16MHz crystal oscillator (the frequency of the quartz mounted on the card) using only 10% of the Arduino memory is obtained.

4. Application of Proposed Network as a Cryptosystem

4.1. Chaos-based image encryption using proposed 5-D hyperjerk oscillator network

Employing a permutation-substitution procedure, we propose the use of our 5-D hyperjerk chaos generator for image encryption, which requires adjustments to our 5-D hyperjerk chaos generator as presented in (7):

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

$$\begin{cases} \dot{x}_1 = x_2 \bmod 1 \\ \dot{x}_2 = x_3 \bmod 1 \\ \dot{x}_3 = x_4 \bmod 1 \\ \dot{x}_4 = bx_5 \bmod 1 \\ \dot{x}_5 = \left(-a_0 x_5 - a_1 x_3 - a_2 x_2 - a_3 x_1 - a_4 x_4 (x_4 - l_1)(x_4 - l_2)\right) \bmod 1 \end{cases} \quad (7)$$

This proposed scheme is outlined in Fig. 16 and its execution is realised via the following steps where we use a plain image (P) and key parameters (x_1, x_2, x_3, x_4, x_5, a_0, a_1, a_2, a_3, a_4, b, l_1, l_2) for iterating 5-D hyperjerk chaos generator as input and the cipher-image (c) as output.

Step 1: Iterate the 5-D hyperjerk chaos generator for h*w times, where h*w is the size of the plain image *P*, which produces output is five sequences X1, X2, X3, X4, and X5 as output.

Step 2: Construct a permutation sequence of length h using the first sequence X1, which has h distinct elements from 1 to h as follows:

- Order the elements of first h elements and discard the first 10 elements in ascending order.
  Eh= order (X1(11 : h+10))
- Obtain the index of each element of the sequence Eh as a sequence X1(11 : h+10).
  Ph=index (Eh in X1(11 : h+10))

Step 3: Construct a permutation sequence of length w using the second sequence X2 with w distinct elements from 1 to w.

- Order the elements of first w elements and discard the first 10 elements in ascending order.
  Ew= order (X2(11 : w+10))
- Obtain the index of each element of the sequence Ew as a sequence X2(11 : h+10).
  Pw=index (Ew in X2(11 : w+10))

Step 4: Construct the substitution sequence of length 256 using the third and fourth sequences X3 and X4, which have 256 distinct elements in the range 0 to 255.

- Y=X3(11 : 266) + X4(11 : 266)
- Order the elements of Y sequence in ascending order.
  Ey= order(Y)
- Obtain the index of each element of the sequence Ey as a sequence Y.
  Sb=index (Ey in Y)

Step 5: Using the fifth sequence X5, construct the key matrix K with size h×w.

$$K = fix\left(X\,5\times10^{12}\right) \bmod 256$$

Step 6: Permute the plain image *P* using the permutation sequences Ph and Pw (which originate from Step 2 and Step 3, respectively), each targeting the rows and columns.

```
for i=1 to h
    for j=1 to w
        Per(i,j)=P(Ph(i),Pw(j));
    end
end
```

Step 7: Substitute the permutated image 'Per' (in Step 6) using Sb substitution sequence (in Step 4).

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083


```
Sub=zeros(a,b);
for i=1 to h
      for j=1 to w
        Sub (i,j)=Sb(Per(i,j)+1);
       end
end
```

Step 8: Perform bitwise XOR operation on substituted image 'Sub' (in Step 7) using key matrix K (in Step 5). C=bitxor(Sub,K)

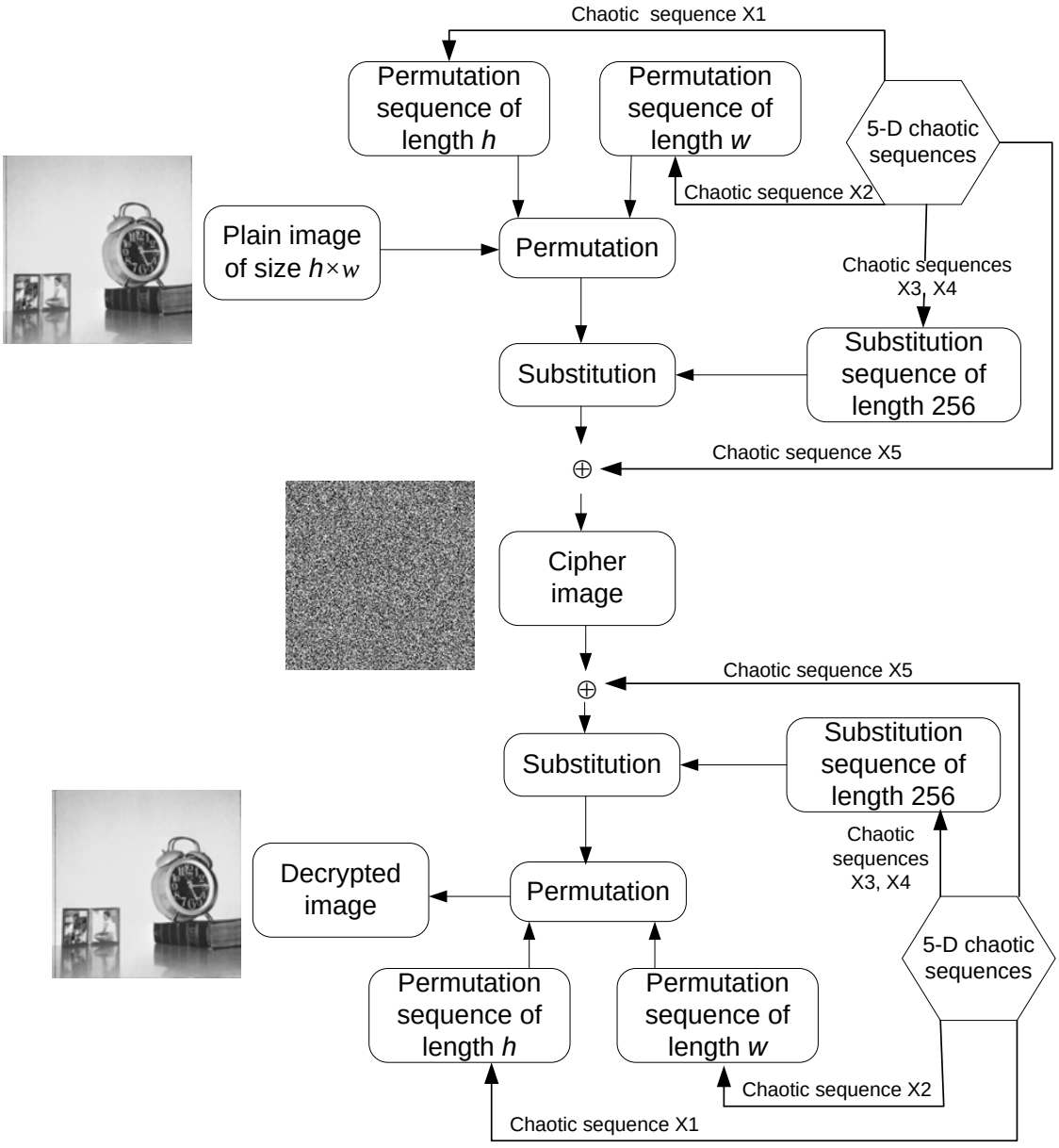


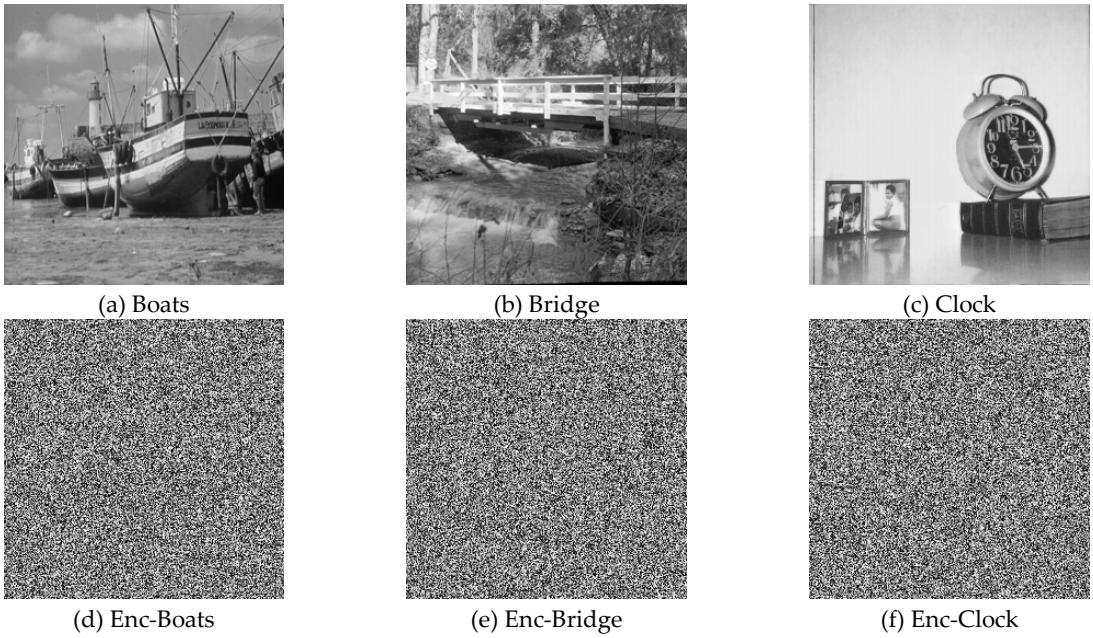
Figure 16: Outline of proposed permutation-substitution image encryption scheme based on sequences X1, X2, X3, X4, X5 of the 5-D hyperjerk chaotic generator.

4.2. Performance tests

To test the performances of our encryption scheme, we set system parameters and initial values within the window of coexisting attractors as $(a_0, a_1, a_2, a_3, a_4, b, l_1, l_2)$=

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

$(1.5, 3, 3.454, 1, 1, 3, 1, 2.6)$ respectively $(x_1, x_2, x_3, x_4, x_5) = (0.7752, 0.6733, 0.9534, 0.8735, 0.8736)$. Further, we simulated implementation of the proposed scheme using 256×256 sized versions of the Boats, Bridge, and Clock greyscale images in Fig. 17(a)-(c) on an Intel ® core™ i5-2450M and 6 GB RAM workstation with a preinstalled MATLAB R2016b software. As seen from the outcome in Figure 17(d)-(f), the encrypted images are visually imperceptible. However, the simple visual inspection remains insufficient to judge the quality of a good encryption scheme. It is well known that many encryption schemes have been successfully violated using simple statistical and differential analysis and attacks. Therefore, we establish the robustness of our proposed technique via these simple, yet important tests as presented in the remainder of this subsection.



(a) Boats (b) Bridge (c) Clock

(d) Enc-Boats (e) Enc-Bridge (f) Enc-Clock

Figure 17: Visual test results showing the plain image and its cipher version

4.2.1. Statistical tests

4.2.1.1. Correlation of adjacent pixels

In sensitivity analysis of encryption keys, quantitative analyses are undertaken using the correlation coefficient [51] metric. In such analysis, the neighbouring pixels of a plain image should be highly correlated with correlation coefficient close to unity (i.e. 1) in each direction. Furthermore, an ideal encryption scheme must produce cipher image with no correlation between neighbouring pixels (i.e. correlation coefficient should be close to 0 in each direction). For this purpose, correlation coefficient is computed using (8).

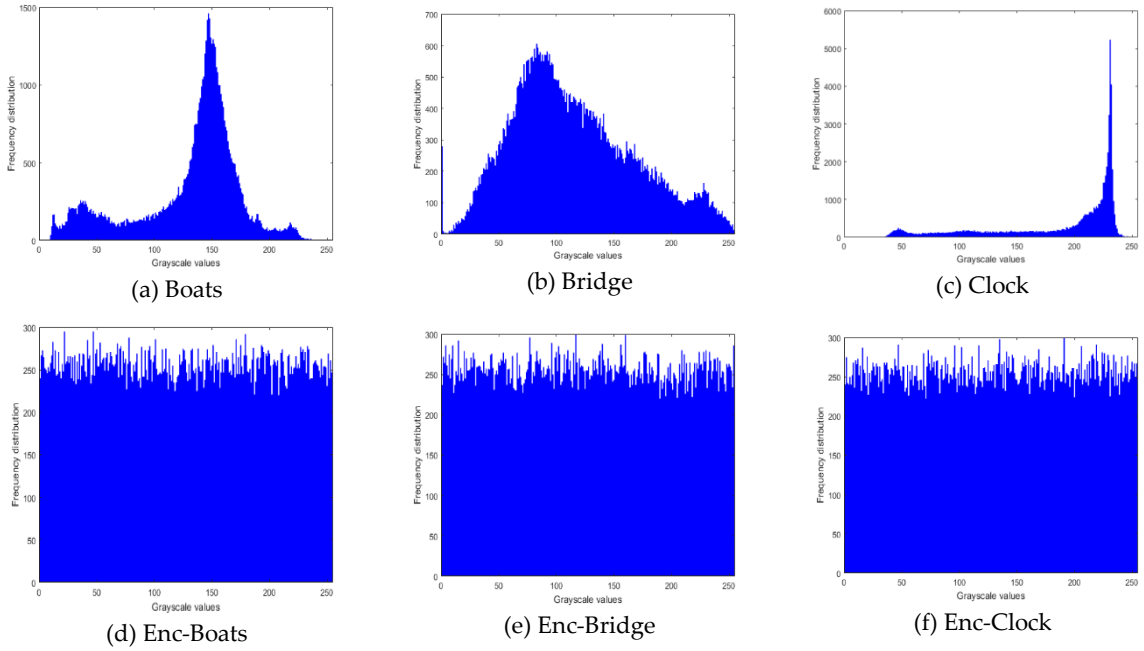
$$r_{xy} = \frac{\sum_{i=1}^{M}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{M}(x_i - \bar{x})^2 \sum_{i=1}^{M}(y_i - \bar{y})^2}} \tag{8}$$

where a pixel p is defined by $p(x_i, y_i)$ and L is the total number of pixels in the cipher image. Table 4 provides the correlation coefficients for the plain and encrypted versions of images in Fig. 17 and from this table it is apparent that the input and encrypted images are highly correlated since correlation coefficient of the encrypted images are very close to 0 in each direction. Consequently, we conclude that the proposed encryption algorithm produces efficiently correlated ciphered images.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

Table 4: Correlation coefficients for the plain image and the related encrypted version

Image	Correlation coefficients					
	Plain image			Cipher version		
Direction	Diagonal	Horizontal	Vertical	Diagonal	Horizontal	Vertical
[10]	0.9466	0.9839	0.9526	−0.0474	−0.033	0.0068
[11]	0.9116	0.9282	0.9644	−0.0319	0.0245	0.0295
[53]	0.8888	0.9567	0.9239	-0.00012	0.0006	-0.0052
Proposed method						
Boats	0.9452	0.9266	0.8855	-0.0007	0.0007	-0.0015
Bridge	0.9203	0.9403	0.8866	-0.0027	0.0008	-0.0010
Clock	0.9767	0.9578	0.9426	-0.0001	0.0007	-0.0023



(a) Boats (b) Bridge (c) Clock

(d) Enc-Boats (e) Enc-Bridge (f) Enc-Clock

Figure 18: Histograms of original and encrypted images in Fig. 17.

4.2.1.2. Histogram tests

An image histogram is the representation of each pixel with respect to its intensity value [55]. This analysis is very useful in deciding the statistical strength of an encryption algorithm. As a representation of incomprehensible information, the histogram of a cipher image is uniformly distributed, while the non-uniform nature of a pristine un-enciphered image depicts the details therein. Figure 18 presents the histograms of the plain and ciphered images used in our experiment, outcomes of which further establish the performance of our proposed scheme in resisting statistical manipulations to the content of the encrypted image.

4.2.1.3. Information entropy

Another statistical metric that is widely used to assess the capability of a cipher scheme to resist statistical attacks is the information entropy. The distribution (entropy) of each pixel x_i with the probability $p(x_i)$ in a given image can be defined as:

$$E(X) = -\sum_{i=1}^{2^L-1} p(x_i)\log_2\left(p(x_i)\right) \tag{9}$$

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

Given that a greyscale image has $256 = 2^8$ possible values then the ideal entropy value should be close to 8. Table 5 provides information entropy values for the encrypted images in comparison with values obtained via previous studies as indicated in the table.

Table 5: Assessment of information entropy for encrypted image

Encryption algorithm	Entropy
[10] Greyscale flower image	7.9969
[11] Cameraman image	7.9455
Proposed method	
Boats	7.99763
Bridge	7.99738
Clock	7.99746

4.2.2. Differential test: NPCR and UACI

In addition to performing well in the statistical tests reported above, a well design encryption algorithm should be very sensitive to slight changes in the composition of the plain image [51-54]. This sensitivity can be evaluated by computing the Number of Pixels Change Rate (NPCR) and the Unified Average Changing Intensity (UACI) which are defined in (16) and (17) respectively.

$$NPCR = \frac{\sum_{i;j} D(i,j)}{w \times h} \times 100\% \quad , D(i,j) = \begin{cases} 0 & if \quad IC_1(i,j) = IC_2(i,j) \\ 1 & if \quad IC_1(i,j) \neq IC_2(i,j) \end{cases} \tag{16}$$

$$UACI = \frac{100}{w \times h} \sum_1^w \sum_1^h \frac{\left| IC_1(i,j) - IC_2(i,j) \right|}{255} \tag{17}$$

where IC_1 and IC_2 are two encrypted images obtained from plain images different in just one pixel, w and h are the dimensions of the images. For an image to be uniformly distributed, the minimum expected values of NPCR and UACI should satisfy (18) and (19) respectively.

$$NPCR_{\max} = \left(1 - 2^8\right) \times 100 = 99.609375\% \tag{18}$$

$$UACI_{\max} = \frac{\sum_{j=1}^{2^8-1} j(j+1}{2^8\left(2^8 - 1\right)} \times 100 = 33.46354\% \tag{19}$$

The results in Table 6 validate the sensitivity and ability of images obtained via proposed scheme to withstand differential attacks aimed at violating their integrity.

Table 6: comparative analysis of UACI and NPCR values with respect to encrypted image

Encryption algorithm	NPCR (%)	UACI (%)
[10] Gray flower image	99.15	33.21
[11] Cameraman image	99.34	33.61
Proposed method		
Boats	99.62	33.69
Bridge	99.60	33.24
Clock	99.64	35.26

4.2.3. Key sensitivity test

An efficient and robust encryption algorithm must show sensitivity to even the slightest changes in the composition of its secret key [51-54]. This is especially important in resisting brute force attacks. To evaluate the key sensitivity of our proposed scheme, the encrypted image is

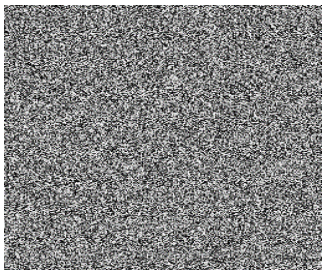
Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

decrypted using four slightly different test keys. The results presented in Figure 19 the impact of slight modifications to key parameters in yielding erroneous outcomes, i.e. ensuring the encrypted image is inaccessible unless with the exact key parameters.

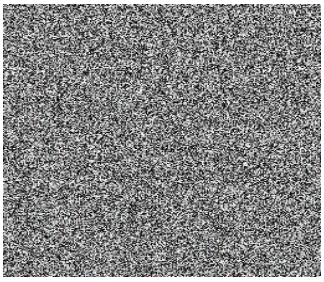
4.2.4. Time and complexity analysis

The speed of an algorithm depends on some important factors such as the specifications and structure of the CPU, the size of memory, the size of image, the software used, etc. To assess our algorithm with those in [32, 41-43], we first attuned it with those in [32, 41-43] using 512 × 512 sized images. Second, we simulate the execution under the same environment: a laptop with Intel coreTM i5-2450M 6 GB RAM and a preinstalled MATLAB R2016b software. Finally, the encryption time is measured without need to consider the time needed to solve chaotic system equations. Consequently, only temporal constraints arising from diffusion and confusion procedures of each algorithm are assessed. Table 7 provides the time analysis for our scheme based on the specs outlined in comparison with results from similar techniques as reported.

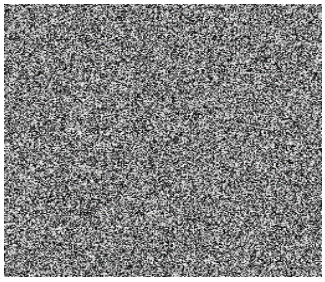
(a) Decryption with correct key



(b) Decryption with $x_1 = 0.775200000000001$



(c) Decryption with $x_2 = 0.673300000000001$



(d) Decryption with $a_0 = 1.500000000000001$

Figure 19: Correct decoding of Boats image and decoding by a slightly changed keys

Table 7: Encryption time (in seconds) where N.R. = Not reported implies that the parameter was not reported in the cited study.

Encryption algorithm	image size					
	32x32	64x64	128x128	256x256	512x512	1024x1024
[32]	N.R	0.0045	0.0163	0.0629	0.2673	1.2157
[41]	N.R	N.R	N.R	0.0460	0.2300	0.9530
[42]	N.R	N.R	N.R	0.0790	0.2454	N.R
[43]	N.R	N.R	N.R	N.R	0.2141	N.R
Proposed method	0.0089	0.0117	0.0247	0.08204	0.1179	1.3441

In addition to encryption time tests, we also undertook a complexity analysis [54] as outlined here. For uniformity and level playing ground, this analysis is done in terms of CPU operations required to execute the different methods. Therefore, each step of of our proposed method as well as those to be used in the comparison are used to estimate the complexity cost.

- Step 1: (5*h*w) steps are required to iterate the chaotic map

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

- Step 2: (h^2) steps each are required to retrieve h elements and obtain the index
- Step 3: (w^2) steps each are required to retrieve w elements and obtain the index
- Step 4: (256*256) steps each are required to retrieve 256 elements, and obtain the index of h elements
- Step 5: (h*w) steps are each required for the multiplication mod operations
- Step 6: (h*w) steps are required for the permutation operation
- Step 7 : (h*w) steps are required for substitution operation
- Step 8: (h*w) steps are required for number of exclusive-XOR operations in the final step

Therefore, the complexity of the algorithm proposed to execute the encryption procedure is $O(max(h^2, w^2, h*w)$ which is an improvement over the complexity reported in [53].

4.2.5. NIST test

To establish the effectiveness of the presented encryption mechanism, we assessed the randomness property of the resulting fifth sequence (for example) as stipulated via NIST SP 800-22 tests, which are considered as the industry standard. These tests consist of 15 examinations that are performed on the fifth generated sequence with 106 bits length and as presented in Table 8 the generated sequence passed tests administered.

Table 8: NIST SP 800-22 tests results

Test-Name	P-Value	Result
Frequency	0.890240	Passed
Block-frequency	0.563092	Passed
DFT	0.378341	Passed
Rank	0.236565	Passed
Runs	0.089504	Passed
Longest runs of ones	0.172795	Passed
Overlapping templates	0.320178	Passed
No overlapping templates	0.465065	Passed
Universal	0.518372	Passed
Approximate entropy	0.844091	Passed
Linear complexity	0.042035	Passed
Cumulative sums (forward)	0.793995	Passed
Cumulative sums (reverse)	0.899532	Passed
Serial test 1	0.179396	Passed
Serial test 2	0.662233	Passed
Random excursions x=1	0.207249	Passed
Random excursions variant x=1	0.042985	Passed

4.2.6. Key space analysis

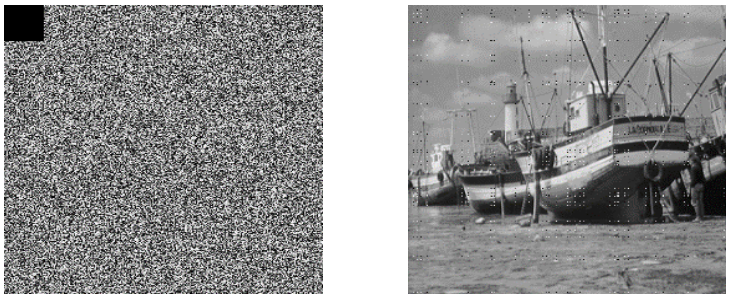
A well-designed image encryption approach should have a sufficiently large key-space which is known as the several keys that can be used in brute-force attacks. For our proposed technique, the image encryption algorithm utilises the key parameters ($a,b,y,x_1,x_2,x_3,x_4,x_5$) to generate the encryption key K. For context, suppose that the calculation precision (floating point operations) for each key is 10^{16}, then the total key space of whole system is 10^{128}, which is within thresholds expected from state-of-the-art encryption algorithms.

4.2.7. Impact of noise on the transmission of cipher images

In providing a thorough assessment of our proposed technique, it is important to evaluate the effect of noise on the cipher image during the transmission. For this, we consider a black cut out that is obtained by modifying 1024 pixels (32×32) in the encrypted image and setting their values to zero. We then execute the decryption procedure on the noisy encrypted image as presented in Figure

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

20. From this result we observe that despite the noise, the original Boats image can be recovered with high visual fidelity (see Figure 20 (b)). Therefore, we can infer the utility of our proposed scheme for public transmission.



(a) (b)

Figure 20: Effect of noise on the cipher Boats image: (a) noise infected cipher image (b) successful decrypted image

5. Concluding remarks

This study has proposed the design and implementation of a 5-D hyperjerk oscillator as chaos generator. Extensive numerical analysis employed showed that the proposed oscillator network comprising of diodes and resistors can produce nonlinearities (both square and cubic) required to generate chaos. Moreover, the proposed design is cheaper than standard analogue multipliers. Additionally, the network exhibited the capacity to experience coexistence of hidden attractors in the phase space. Furthermore, we exploited the efficient cost-effective design of our network to explore deployment of its lightweight version as an image cryptosystem. Based on this application and outcomes of standard security analysis validated the performance and utility of our proposed image encryption scheme were validated. Additionally, an Arduino UNO set up was utilised to implement the network and experimental results showed that our proposed chaos generator could have useful applications in emerging paradigms for information and communication security. For future and ongoing work, our study is being improved in the following directions. First, we note that despite their relative objectivity, statistical tests do not cover all aspects of cryptanalytical attacks. Therefore, following necessary refinements, we plan on integrating differential trails over the encryption process. These are reputed to be more powerful then permutation-only and substitution paradigms. Additionally, in ongoing work, we are exploring the use of Cobweb diagrams (representation of phase plot for digital systems) for chaos generation based on the Arduino platform. Insights from this and other improvements to this study will be used to improved image complexity analysis, develop faster and more robust encryption strategies for colour images aimed primarily at securing medical images for applications in telemedicine.

Author Contributions: Conceptualisation, T.N., N.J.D.D., E.J.Y., and A.A.A; methodology, K.J. , A.A.A and E.J.Y.; software, T.N., A.A.A, and N.J.D.D.; validation, K.J., A.A.A., and A.M.I.; formal analysis, T.N., E.J.Y., A.A.A, and N.J.D.D.; investigation, T.N. and A.M.I.; writing—original draft preparation, T.N. , A.A.A and K.J.; writing—review and editing, T.N., A. A. A. and A.M.I.; visualisation, A.M.I. and A.A.A.; supervision, K.J. and A.M.I.; project administration K.J, A.A.A. and A.M.I.

Funding: This study is sponsored by the Prince Sattam Bin Abdulaziz University, Saudi Arabia via the Deanship for Scientific Research funding for the Advanced Computational Intelligence & Intelligent Systems Engineering (ACIISE) Research Group Project Number 2019/01/9862.

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

References

1. FIPS PUB, Data Encryption Standard (DES), NIST, Publication 46, 1999.
2. Diffie, W.; Hellman M. New direction in cryptograph. IEEE Trans. on inf. theory 1976, 22(6), 644-654.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

3. Abid Z.; Wang W. Countermeasures for hardware fault tattack in multiprime RSA cryptosystems. Int. Jour. of Net. S. 2008, 6, 190-200.
4. Li, S.; Chen, G.; Cheung, A.; Bhargava, B.; Lo, K.T. On the design of perceptual MPEG-video encryption algorithms. In: IEEE Transactions on Circuits and Systems for Video Technology (2007), 214–223.
5. Nkapkop J.D.D.; Effa J.Y.; Borda M.; Terebes R. A Novel Fast and Secure ChaosBased Algorithm for Image Encryption, Innov. Sec. Sol. for Inf. Tech. and Com. 2015, 9522, 87-101.
6. Tsafack N.; Kengne J.; Abd-El-Atty B.; Iliyasu A.M.; Hirota K.; Abd EL-Latif A.A. Design and implementation of a simple dynamical 4-D chaotic circuit with its cryptographic applications. Inf. Sc. 2019, to appear.
7. El-Latif A.A.A.; Li L.; Wang N.; Han Q.; Niu X. A new approach to chaotic image encryption based on quantum chaotic system, exploiting color spaces. Signal Processing 2013, 93 (11), 2986–3000.
8. Li L.; Abd-El-Atty B.; El-Latif A. A. A.; Ghoneim A. Quantum color image encryption based on multiple discrete chaotic systems. FedCSIS, IEEE 2017, 555–559.
9. Luo X.; Zhou R.; Liu J.; Cao Y.; Ding X. A parallel image encryption algorithm based on the piecewise linear chaotic map and hyper-chaotic map. NoDy 2018 93 (3) 1165–1181.
10. Khan JS, Ahmad J. Chaos based efficient selective image encryption. Multidimensional Systems and Signal Processing. 2019 Apr 1;30(2):943-61.
11. Ahmad J, Khan MA, Ahmed F, Khan JS. A novel image encryption scheme based on orthogonal matrix, skew tent map, and XOR operation. Neural Computing and Applications. 2018 Dec 1;30(12):3847-57.
12. Quing L.; Congxu Z.; Guojun W. A novel S-Box design Algorithm based on new compound chaotic system. Entropy, 2019, 21, 1004, doi.org/10.3390/21101004.
13. Folifack V. R., Kengne J. Coexistence of hidden attractors, 2-torus and 3-torus in a new simple 4-D chaotic system with hyperbolic cosine nonlinearity. IJDC 2018, 6(4), 1421–1428.
14. Njitacke Z.T.; Kengne J. Nonlinear Dynamics of Three-Neurons-Based Hopfield Neural Networks (HNNs): Remerging Feigenbaum Trees, Coexisting Bifurcations and Multiple Attractors. J. of Cir., Syst., and Comp. 2019, 28.
15. Tsafack N.; Kengne J.; A particular class of simple chaotic circuits: multistability analysis. Lap LAMBERT Academic Publishing 2019; ISBN: 978-613-9-46143-1.
16. Kengne J.; Jafari S.; Njitacke Z.T.; Yousefi K.M.; Cheukem A. Dynamic analysis and electronic circuit implementation of a novel 3D autonomous system without linear terms. Com. in No. Sc. and Num. Sim. 2017, doi: 10.1016/j.cnsns.2017.04.017.
17. Kengne J.; Njikam S.M.; Folifack V. R. A plethora of coexisting strange attractors in a simple jerk system with hyperbolic tangent nonlinearity, Chaos, Solitons and Fractals 2018, 106, 201–213.
18. Wang B.; Xie Y.; Zhou C.; Zhou S.; Zheng X. Evaluating the permutation and diffusion operations used in image encryption based on chaotic map. Optik 2016, 127, 3541–3545.
19. Shuqin Z.; Congxu Z.; Wenhong W. A new image encryption algorithm based on chaos and secure Hash SHA-256. Entropy, 2018, 20, 716, doi: 10.339/e20090716.
20. Benrhouma, O., Hermassi, H., El-Latif, A. A. A., & Belghith, S. Cryptanalysis of a video encryption method based on mixing and permutation operations in the DCT domain. Signal, Image and Video Processing 2015, 9(6), 1281-1286.
21. Belazi, A., Khan, M., El-Latif, A. A. A., & Belghith, S. (2017). Efficient cryptosystem approaches: S-boxes and permutation–substitution-based encryption. Nonlinear Dynamics, 87(1), 337-361.
22. De la Hoz M.Z.; Leonardo A.; Yolanda V. An experimental realization of a chaos-based secure communication using arduino microcontrollers. The Scientific World Journal 2015, 10 pages, doi.org/10.1155/2015/123080.
23. Siva J.; Thenmozhi K.; John B.B.R.; Amirtharajan R. Lightweight chaotic image encryption algorithm for real-time embedded system: Implementation and analysis on 32-bit microcontroller, Micropro. and Microsys. 2017, doi: 10.1016/j.micpro.2017.10.013
24. Tanougast C.; Dandache A.; Azzaz M.; Sadoudi S. Hardware Design of Embedded Systems for Security Applications. INTECH 2012, DOI: 10.5772/38649
25. Tsafack N.; Kengne J. A Novel Autonomous 5-D Hyperjerk RC Circuit with Hyperbolic Sine Function. The Scientific World Journal 2018, 17 pages, doi.org/10.1155/2018/1260325.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

26. Fortuna L.; Rizzo A.; Xibilia M. G. Modeling complex dynamics via extended PWL-based CNNS, IJBC 2003, 13(11), 3273-3286.
27. Li C., Sprott J.C. Coexisting hidden attractors in a 4-D simplified Lorenz system. IJBC 2014, 24.
28. Dudkowski D.; Prasad A.; Kapitaniak T., Perpetual points and hidden attractors in dynamical systems. Phys. Lett. A 2015, 379, 2591–2596.
29. Prasad A. Existence of perpetual points in nonlinear dynamical systems and its applications. IJBC 2015, 25.
30. Khan J.; Ahmad J. Chaos based efficient selective image encryption. Multidim Syst Sign Process. 2019, 30: 943, doi:10.1007/ s11045-018-0589-x
31. Ahmad J.; Ali M.; Ahmed F..; Khan J. A novel image encryption scheme based on orthogonal matrix, skew tent map and XOR operation. Comput & Applic. 2018, 30: 3847, doi: 10.1007/s00521-017-2970-3
32. Zhijuan D. and Shaojun Z. A digital image encryption algorithm based on chaotic mapping. Journal of Algorithms & Computational Technology 2019, 13, 1–11, doi: 10.1177/1748302619853470
33. Chong F.; Chen J.J.; Zou H.; Meng W.H.; Zhan Y.F. A chaos-based digital image encryption with an improved permutation strategy. Optic. Express 2012, 20, 2363–2378.
34. Hua Z.; Zhou Y.; Huang H. Cosine-transform-based chaotic system for image encryption, Inf. Sc. 2019, 480, 403–419.
35. Wang X.; Feng L.; Zhao H. Fast image encryption algorithm based on parallel computing system, Inf. Sc. 2019, 486 340–358.
36. Ravichandran D.; Praveenkumar P.; Rayappan J. B. B.; Amirtharajan R. Dna chaos blend to secure medical privacy, IEEE transactions on nanobioscience 2017, 16 (8), 850–858.
37. Lv X.; Liao X.; Yang B. A novel scheme for simultaneous image compression and encryption based on wavelet packet transform and multi-chaotic systems, Multimedia Tools and Applications 2018 77 (21), 28633–28663.
38. Xu M.; Tian Z.; A novel image cipher based on 3d bit matrix and latin cubes, Inf. Sc. 2019, 478 1–14.
39. Abd-El-Atty B.; El-Latif A. A. A.; Venegas-Andraca S. E. An encryption protocol for NEQR images based on one-particle quantum walks on a circle. Quantum Information Processing 2019, 18 (9), 272.
40. Deb S.; Biswas B.; Bhuyan B. Secure image encryption scheme using high efficiency word-oriented feedback shift register over finite field. Multimedia Tools and Applications 2019, 1–25.
41. Behnis S.; Akhshani A.; Ahadpour S.; Mahnodi H.; Akhavan A. A fast-chaotic encryption scheme based on piecewise nonlinear chaotic maps. Phys. Lett. A 2007, 366(4–5), 391–396
42. Wang X.Y.; Zhao J.F.; Liu H.J. A new image encryption algorithm based on chaos. Opt. Commun. 2012 285(5), 562–566
43. Gao T.G.; Chen Z.Q. A new image encryption algorithm based on hyper-chaos. Phys. Lett. A 2008, 372(4), 394–400
44. Azzaz M.S.; Tanougast C.; Sadoudi S. A new auto-switched chaotic system and its FPGA implementation. Commun Nonl Sci Numer Simul 2013, 18(7):1792-1804. doi.org/10.1016/j.cnsns.2012.11.025
45. Murillo-Escobar M.; Cruz-Hernández C.; Abúndiz-Pérez F. A robust embedded biometric authentication system based on fingerprint and chaotic encryption. Exp. Syst. Appl. 2015, 42(21), 8198-8211, doi.org/10.1016/j.eswa.2015.06.035.
46. Jasio L.D. Programming 32-Bit Microcontrollers in C: Exploring the PIC32. Newnes, Burlington, USA, 2008.
47. Rodrigo M.; Adrian A.; Cesar C.; Fausto A.; Rigoberto M. Chaotic digital cryptosystem using serial peripheral interface protocol and its dsPIC implementation. Frontiers of information technology and electronic engineering 2018, doi: 10.1631/FITEE.1601346.
48. Veronique G.; Pierre P.; Daniel F.; Taha A. Chaos based cryptosystem on DSP'', Chaos solitons and Fractals 2009, 42, 2135-2144.
49. Siddiqui R.A.; Grosvenor R.I.; Prickett P.W. dsPIC-based advanced data acquisition system for monitoring, control and security applications. 12th Int Bhurban Conf on Applied Sciences and Technology, 2015, 293-298. doi.org/10.1109/IBCAST.2015.7058519
50. Uriz A.J.; Agüero P.D.; Moreira J.C. Flexible pseudorandom number generator for tinnitus treatment implemented on a dsPIC. IEEE Latin Am Trans 2016, 14(1):72-77. doi.org/10.1109/TLA.2016.7430063.

Peer-reviewed version available at Sensors 2019, 20, 83; doi:10.3390/s20010083

51. Zhu S, Zhu C. Image encryption algorithm with an avalanche effect based on a six-dimensional discrete chaotic system. Multimedia Tools and Applications. 2018 Nov 1;77(21):29119-42.
52. Zhu C, Wang G, Sun K. Improved cryptanalysis and enhancements of an image encryption scheme using combined 1D chaotic maps. Entropy. 2018 Nov;20(11):843.
53. Zhu C, Wang G, Sun K. Cryptanalysis and improvement on an image encryption algorithm design using a novel chaos-based S-box. Symmetry. 2018 Sep;10(9):399.
54. Zhu S, Wang G, Zhu C. A secure and fast image encryption scheme based on double chaotic s-boxes. Entropy. 2019 Aug;21(8):790.
55. Zhu C, Wang G, Sun K. Improved Cryptanalysis and Enhancements of an Image Encryption Scheme Using Combined 1D Chaotic Maps, Entropy 2018, 20, 843; doi:10.3390/e20110843