A new 6D discrete system and its Arduino Uno implementation with LED visualization

[1]**M. I. Kopp**

July 14, 2022

[1] *Institute for Single Crystals, NAS Ukraine, Nauky Ave. 60, Kharkov 61001, Ukraine*

Abstract

In this paper, to demonstrate the chaotic behavior of a new 6D discrete system, the modern Matlab-Simulink software environment was used. We have obtained a discrete 6D chaotic system by the Euler method from a continuous 6D system of dynamic equations. In the Matlab-Simulink environment, models for continuous and discrete 6D systems of equations were created, and the results were identical. To demonstrate the synchronization of two unidirectional connected continuous and discrete 6D systems, Simulink models were proposed. A discrete 6D system was applied for the chaotic masking of a narrow-band harmonic signal and image encryption. For visualizing and practical realization of the new chaotic discrete system we used Arduino Uno board and six light-emitting diodes (LEDs). The programming code and connecting technique are also shown. The program code was debugged into the free Arduino simulation environment such as Tinkercad. Compiling hex. file in the program software Arduino IDE allows us to simulate a 6D chaotic system using the Arduino Uno microcontroller in the Proteus 8 environment.

Keywords: Euler method, chaotic behavior, synchronization, secure communication, Tinkercad, Arduino UNO, Proteus simulation

1 Introduction

Dynamic chaos research continues to develop rapidly due to its wide application in various fields of science and technology, such as telecommunication systems [1]-[2], cryptography [3]-[4], neural network systems [5], and other applications. Recently, the design of new dynamic chaos electronic generators has been significantly increased. Many circuit solutions are known for the classical equations of dynamic chaos [6]-[8], and also for new equations of dynamic chaos. Paper [9] announces a new easy to implement 3D chaotic system with five quadratic nonlinearities and three positive parameters, which has a larger bandwidth compared to at least 50 other systems that have been described in this paper.

Analytical solutions to the equations characterizing the system's chaotic behavior are impossible. As a result, numerical simulation in the MATLAB, Python, or Simulink environments

is actively used. Deterministic nonlinear systems generate chaotic signals, but they are unpredictable over long time intervals and are extremely sensitive to initial conditions. Because of the random nature of the signals created by dynamic systems, these systems may be used to build a pseudo-random number generator (PRNG).

Chaotic systems are classified into two types: discrete and continuous. The first type is represented by maps, whereas the second by nonlinear differential equations. In [10]-[12], new discrete-time chaotic systems based on the logistic map were proposed. Modified logistic mappings have been used in secure communication systems and image encryption. The discretization of differential equations technique is used to implement continuous chaotic systems in digital devices. Paper [13] examines several discretization techniques for well-known chaotic systems such as the Lorenz system, the Duffing oscillator, and the Nose-Hoover system. The results in [13] did not provide a definitive answer to the question of which discretization algorithm is the best, but it was mentioned that the Euler and Heun methods should be used first for discretization due to their simplicity and ease of synthesis when compared to the more accurate RK4 algorithm. This strategy was effectively applied in article [14]. The Lorentz system was solved by the Euler method, which was programmed into the PIC18F2520 microcontroller. As noted in [14], instead of PIC18F2520, one can use other popular microcontrollers such as Arduino.

Several papers [15]-[19] have recently been published on the employment of the Arduino microcontroller for the study of various chaotic systems. The experiment utilizing Arduino in [15] demonstrated the applicability of using discrete chaotic oscillators in today's digital technology in the hands of almost anybody. In [16] was presented a communication system based on chaotic logistic maps and an experimental realization using Arduino Uno microcontroller boards, which perform the encryption and decryption algorithms in the transmitter and receiver, respectively. The results of the experiment demonstrated the viability of employing Arduino microprocessors for the requested purpose. In [17], two-dimensional (2D) Lozi, Tinkerbell, and Barnsley Fern discrete Chaotic Maps are implemented based on the microcontroller Arduino UNO. [18] presents the implementation and experimental testing of a microcontroller-based system that solves Lorenz equations in real time and generates chaotic waveforms. The Arduino Mega 2560 R3 microcontroller was used to implement the proposed design. Using digital-to-analog converters, the microcontroller sends the chaotic signals to the circuit's outputs [18]. As shown in [19], the Arduino Mega 2560 R3 microcontroller-based system can demonstrate chaotic behavior memristive circuits and reproduce the Matlab and Proteus simulations well. The Arduino UNO board, three light-emitting diodes (LEDs), and three resistors for each coordinate were used in the article [20] for practical realization and simple visualization of the chaotic Lorenz system. In another work [21], for visualizing and practical realization of the new modified nonlinear logistic map, the Arduino Uno board was used along with ten light-emitting diodes (LEDs).

In this work, the discretization method is applied to the equations of six-dimensional (6D) chaotic dynamics. These equations were obtained in a study of convection in a nonuniformly rotating electrically conductive fluid in a constant vertical magnetic field [22]. In work [23], a scheme for chaotic modulation secure communication based on the chaotic synchronization of a new 6D chaotic system was suggested. As shown in [23], the proposed chaotic masking and signal decoding scheme is implemented through the analog electronic circuit, which is characterized by its high accuracy and good robustness.

The purpose of this study is to construct computer models of a discrete 6D chaotic dynamical system using the Matlab-Simulink environment and practical realization of the new discrete 6D chaotic system with the help of the Arduino Uno board.

2 Discretization of continuous 6D chaotic system

Discretization issues for chaotic systems with continuous-time are important for the implementation of TRNGs in microcontrollers and FPGAs. Many continuous-time systems that are typically constructed and synthesized using analog components must be approximated by digital equivalents due to rapid technological improvements. Secure communication systems based on chaos are among the most promising possibilities for converting from analog to digital implementations. To attain this purpose, discretization is necessary, which is accomplished by employing mathematical techniques to approximate the dynamics of continuous-time systems with simpler difference equations. For our investigation, we will use the continuous six-dimensional (6D) chaotic system from the paper [22]:

$$\begin{cases} \dot{x} = -x + ry - au - bv \\ \dot{y} = c(-y + x - xz) \\ \dot{z} = c(-gz + xy) \\ \dot{u} = -u + cx \\ \dot{v} = -v + hx + kw \\ \dot{w} = -w - cv - lu \end{cases} \tag{1}$$

where the dot on the symbol indicates the differentiation with respect to time, $r = 290, a = 2, b = 0.1, c = 0.1, g = 2.67, h = 8.21, k = 2, l = 24.65$ are real constants. The initial conditions are given as $x(0) = y(0) = z(0) = u(0) = v(0) = w(0) = 1$. As can be seen, the nonlinear system of Eq. (1) has the dimension of a 6D phase space and as a result, has a huge variety of multiple behavior modes. Based on the area of different parameter changes, the system (1) is more than likely to realize all possible chaotic transitions. In [22] completed a dynamic analysis of Eq. (1), demonstrating the existence of periodic, quasi-periodic, and chaotic regimes depending on the parameter r values.

The approximation degree of the exact system solutions differs amongst numerical simulation methods for differential equations. The Euler and Runge-Kutta techniques are the most popular. The exact solutions of chaotic systems models cannot be obtained using analytical methods due to the sensitivity to arbitrarily slight changes in the initial conditions, so we will use numerical methods. Numerical approaches consist of presenting a set of differential equations as a recurrent dependency and calculating sequence points on the trajectory in discrete moments with timestep T_s [13]:

$$\dot{S} \cong \frac{S[n+1] - S[n]}{T_s} \tag{2}$$

Euler's technique has the most computer complexity, as the ratio between neighboring points

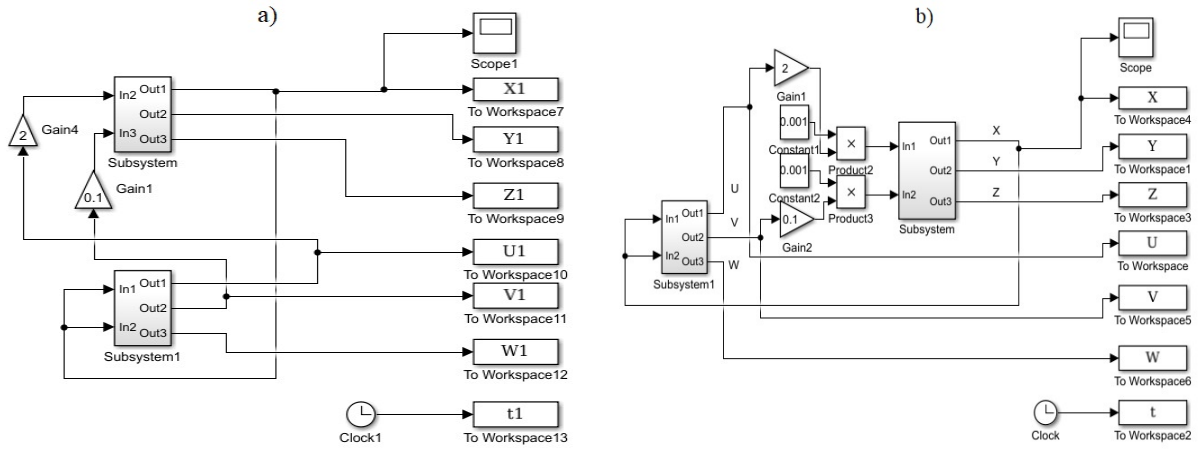

Figure 1: Models in Simulink for a 6D continuous (a) and discrete (b) chaotic system.

of the system trajectories (1) according to (2) is represented by the following equations:

$$\begin{cases} x[n+1] = (1 - T_s)x[n] + rT_s y[n] - aT_s v[n] - bT_s u[n] \\ y[n+1] = (1 - cT_s)y[n] + cT_s x[n] - cT_s x[n]z[n] \\ z[n+1] = (1 - gcT_s)z[n] + cT_s x[n]y[n] \\ u[n+1] = (1 - T_s)u[n] + cT_s x[n] \\ v[n+1] = (1 - T_s)v[n] + hT_s w[n] + kT_s x[n] \\ w[n+1] = (1 - T_s)w[n] - cT_s v[n] + lT_s u[n] \end{cases} \tag{3}$$

Figs. 1a and 1b show Simulink block diagrams for the cases of the continuous system (1) and discrete system (3). T_s was assigned to a constant value ($T_s = 0.001$), and the starting conditions for the six variables were set internally, within the integrators $x(0) = y(0) = z(0) = u(0) = v(0) = w(0) = 1$, respectively. Fig. 2 shows subsystems for Simulink models of continuous and discrete 6D chaotic systems. The results of a 300-second simulation of two systems (1) and (3) at $T_s = 0.001$s are displayed in Fig. 3. According to Fig. 3, the discretized system (3) retains the chaotic behavior of the original continuous system (1).

3 Synchronization of two coupled 6D chaotic systems

The main issue in telecommunication systems for secure information transfer is the synchronization of chaotic generators on the transmitting and receiving devices. Synchronization, from a physical point of view, means that the trajectories of one system, for example, transmitting $\mathbf{D}(x, y, z, u, v, w)$, converge to another receiving system $\mathbf{R}(x_r, y_r, z_r, u_r, v_r, w_r)$ in time $t \to \infty$:

$$\lim_{t\to\infty} \mathbf{R} = \mathbf{D}.$$

Consider a pair of 6D chaotic continuous systems consisting of the drive (or master) and response (or slave) systems. The master system corresponds to the system (1) and is comparable to the Simulink model (Fig. 1a). The slave system varies from the master system because it

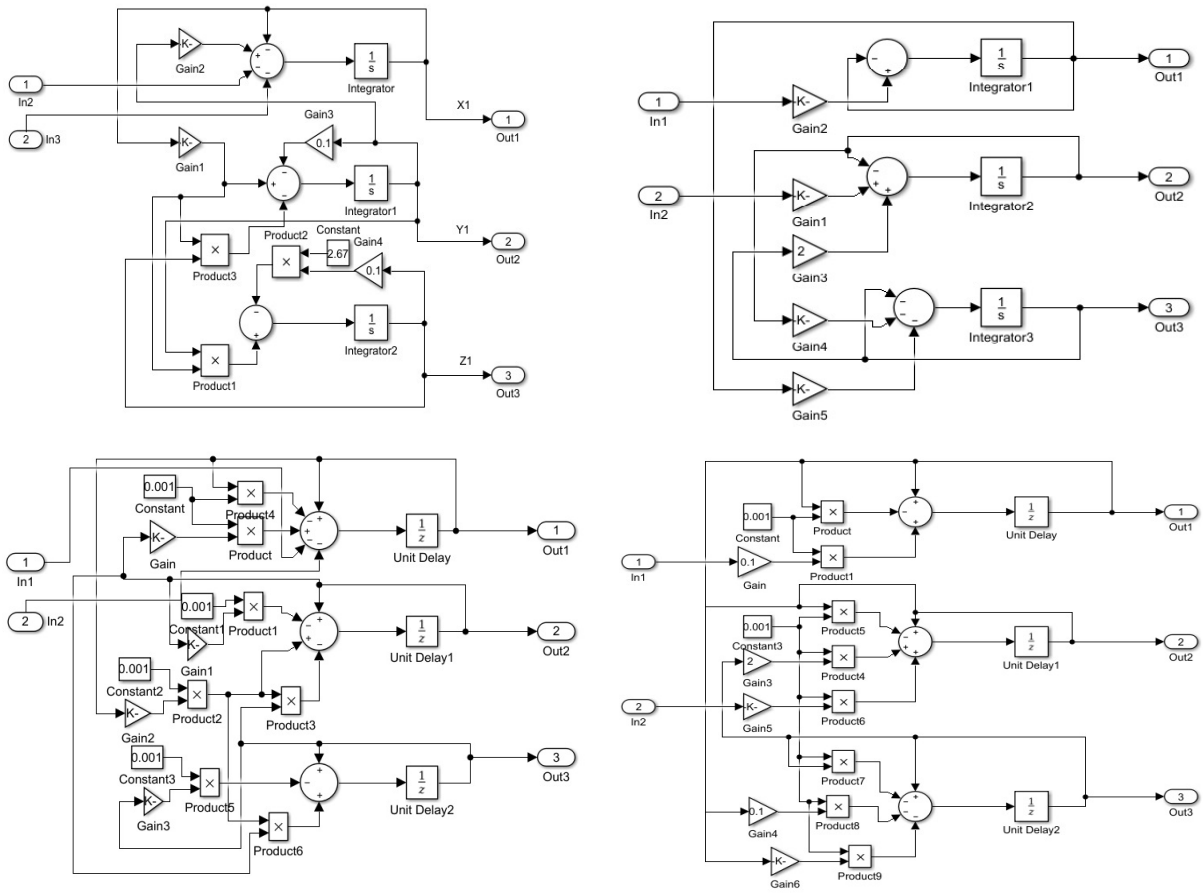

Figure 2: Subsystems for continuous (top) and discrete (bottom) Simulink models.

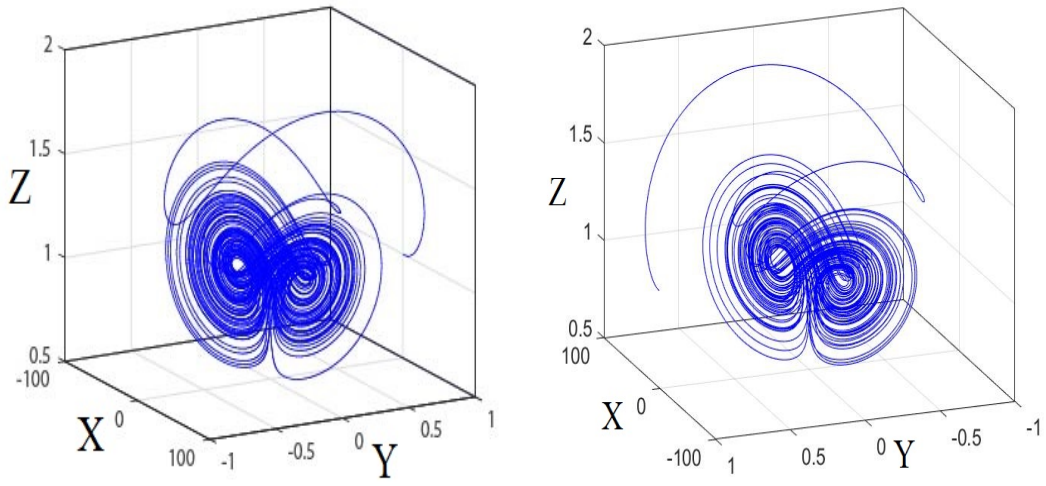

Figure 3: On left – strange attractor of continuous system (1); on right – strange attractors of discrete system (3).

has an input signal for synchronization. The following system of differential equations can represent a pair of single-direction connected system

$$\begin{cases} \dot{x}_1 = -x_1 + r_1 y_1 - a_1 u_1 - b_1 v_1 \\ \dot{y}_1 = c_1(-y_1 + x_1 - x_1 z_1) \\ \dot{z}_1 = c_1(-g_1 z_1 + x_1 y_1) \\ \dot{u}_1 = -u_1 + c_1 x_1 \\ \dot{v}_1 = -v_1 + h_1 x_1 + k_1 w_1 \\ \dot{w}_1 = -w_1 - c_1 v_1 - l_1 u_1 \\ \dot{x}_2 = -x_2 + r_1 y_1 - a_2 u_2 - b_2 v_2 \\ \dot{y}_2 = c_2(-y_1 + x_2 - x_2 z_2) \\ \dot{z}_2 = c_2(-g_2 z_2 + x_2 y_1) \\ \dot{u}_2 = -u_2 + c_2 x_2 \\ \dot{v}_2 = -v_2 + h_2 x_2 + k_2 w_2 \\ \dot{w}_2 = -w_2 - c_2 v_2 - l_2 u_2 \end{cases} \tag{4}$$

where $x_1, y_1, z_1, u_1, v_1, w_1$ and $x_2, y_2, z_2, u_2, v_2, w_2$ are coordinates of the master and slave systems, respectively. $a_1, b_1, c_1, r_1, g_1, h_1, k_1, l_1$ and $a_2, b_2, c_2, r_2, g_2, h_2, k_2, l_2$ are parameters of the master and slave systems, respectively. As can be seen from the system (4), the synchronization mode is carried out under the condition: $y_1 = y_2$.

The model of a pair of single-directionally coupled 6D chaotic continuous systems built in the Simulink system is shown in Fig. 4 (system parameters are identical). The transient synchronization process in the example of coordinates $x_1(t)$ and $x_2(t)$ is shown in Fig. 5a. Fig. 5b shows the convergence of the synchronization error $\zeta_1 = x_2(t) - x_1(t)$ to zero exponentially with time.

Similarly, consider a pair of 6D chaotic discrete systems consisting of a drive (or master) and a response (or slave) system. The master system relates to the system (3) and is equivalent to the Simulink model (Fig. 1b). The slave system differs from the master system in that it has an input signal for synchronization. The differential equation system shown below can represent a pair of single-direction coupled systems:

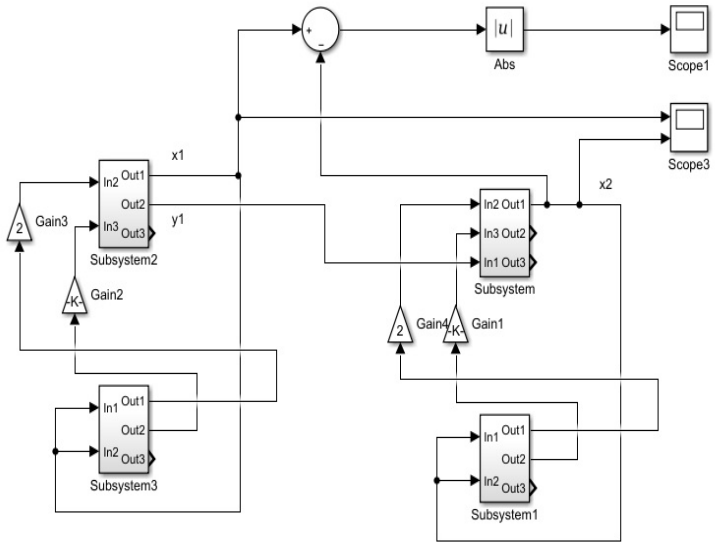

Figure 4: Simulink model of a pair of single-directionally connected 6D continuous systems.

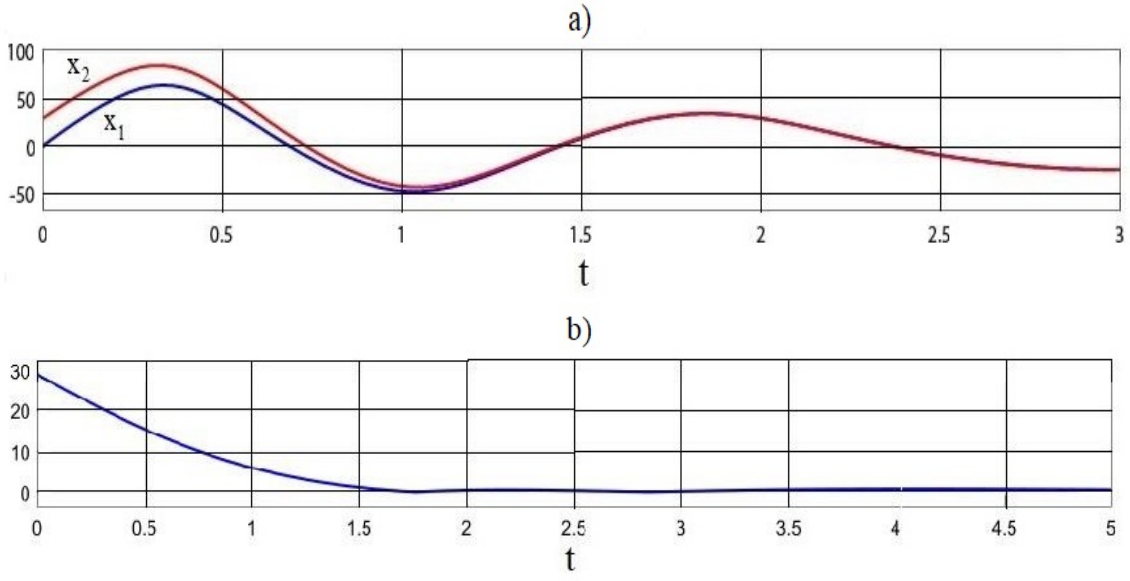

Figure 5: The synchronization process of coupled 6D continuous systems: a) timing diagrams of output signals x_1 and x_2 of coupled systems; b) time-history of the synchronization error.

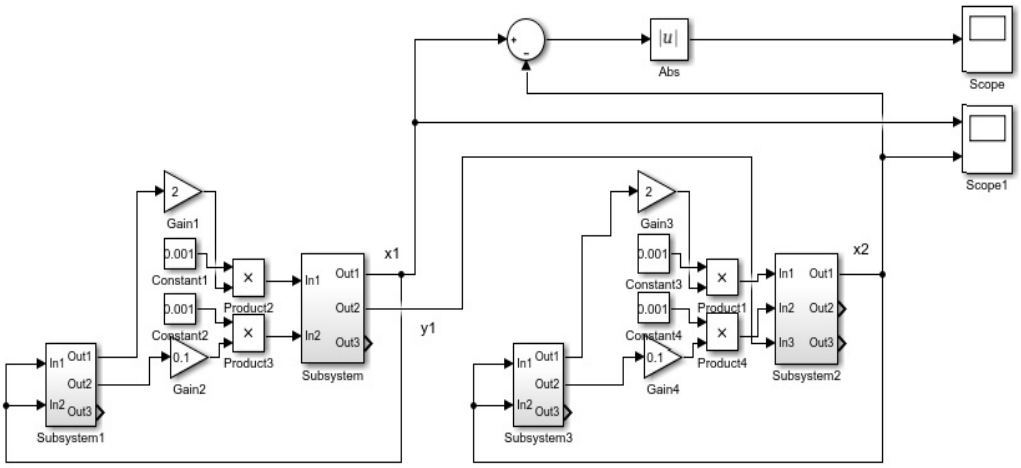

Figure 6: Simulink model of a pair of single-directionally connected 6D discrete systems.

$$
\begin{cases}
x_1[n+1] = (1-T_s)x_1[n] + r_1T_sy_1[n] - a_1T_sv_1[n] - b_1T_su_1[n] \\
y_1[n+1] = (1-c_1T_s)y_1[n] + c_1T_sx_1[n] - c_1T_sx_1[n]z_1[n] \\
z_1[n+1] = (1-g_1c_1T_s)z_1[n] + c_1T_sx_1[n]y_1[n] \\
u_1[n+1] = (1-T_s)u_1[n] + c_1T_sx_1[n] \\
v_1[n+1] = (1-T_s)v_1[n] + h_1T_sw_1[n] + k_1T_sx_1[n] \\
w_1[n+1] = (1-T_s)w_1[n] - c_1T_sv_1[n] + l_1T_su_1[n] \\
x_2[n+1] = (1-T_s)x_2[n] + r_2T_sy_1[n] - a_2T_sv_2[n] - b_2T_su_2[n] \\
y_2[n+1] = (1-c_2T_s)y_1[n] + c_2T_sx_2[n] - c_2T_sx_2[n]z_2[n] \\
z_2[n+1] = (1-g_2c_2T_s)z_2[n] + c_2T_sx_2[n]y_1[n] \\
u_2[n+1] = (1-T_s)u_2[n] + c_2T_sx_2[n] \\
v_2[n+1] = (1-T_s)v_2[n] + h_2T_sw_2[n] + k_2T_sx_2[n] \\
w_2[n+1] = (1-T_s)w_2[n] - c_2T_sv_2[n] + l_1T_su_2[n]
\end{cases}
\tag{5}
$$

We are also observed from the system (5) that the synchronization mode is carried out under the condition: $y_1 = y_2$

Fig. 6 depicts the Simulink model of a pair of single-directionally connected 6D chaotic discrete systems when the system parameters are identical. Figure 7a depicts the transient synchronization process using the coordinates $x_1(t)$ and $x_2(t)$. Figure 7b depicts the exponential convergence of the synchronization error $\zeta_1 = x_2(t) - x_1(t)$ to zero.

Thus, synchronization of continuous and discrete 6D chaotic systems is successfully implemented even under different initial conditions between the master ($x_1(0) = 1$) and slave ($x_2(0) = 30$) systems. As can be seen from the simulation results, the synchronization processes for discrete and continuous systems are identical.

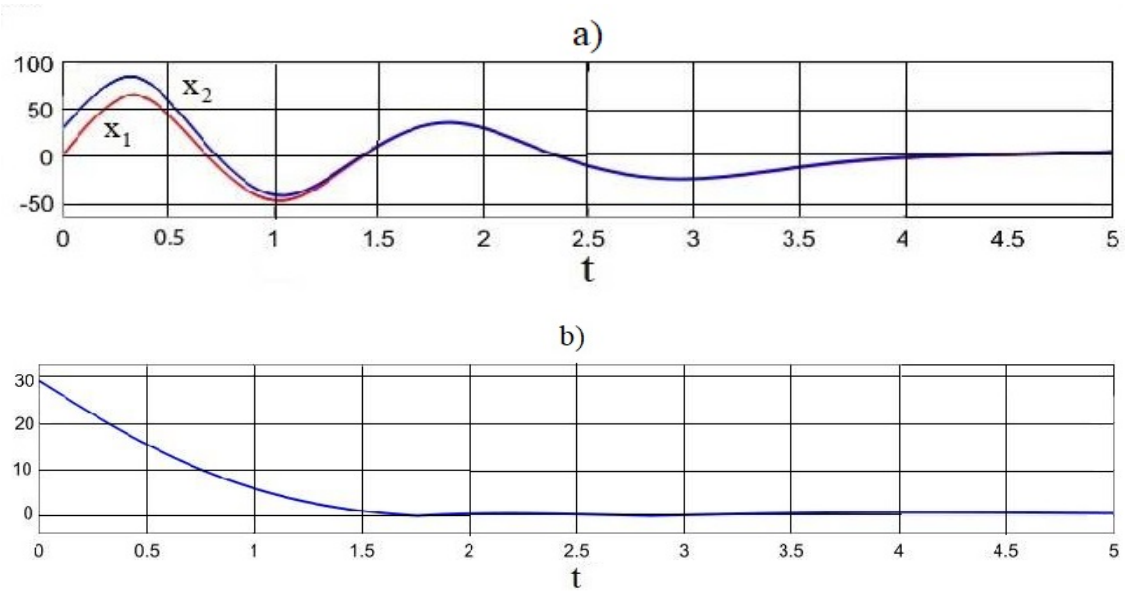

Figure 7: The synchronization process of coupled 6D discrete systems: a) timing diagrams of output signals x_1 and x_2 of coupled systems; b) time-history of the synchronization error.

4 Chaotic masking of narrowband signals by a 6D discrete system

The method of transmitting information based on chaotic masking is the following steps:

1. the information signal $s(t)$ is mixed with the chaotic signal $x1(t)$ of the generator of the drive (or master) system and transmitted over the communication channel;

2. after setting the synchronization mode, the output signal $x2(t)$ of the slave system of the receiver and the chaotic component $x1(t)$ of the transmitted signal are identical;

3. demodulation is carried out by subtracting the chaotic signal $x2(t)$ of the slave system generator from the received signal.

The model of a transmission system with chaotic masking developed in the Simulink environment is shown in Fig. 8. It is a modification of the synchronization system shown in Fig. 6 with an additional Sine Wave block, which is the source of the harmonic information signal $s(t)$, and signal addition/subtraction blocks. The $y(t)$ coordinate is used to synchronize the systems (Subsystem-Subsystem1) and (Subsystem2-Subsystem3) depicted in Figure 8. The synchronized systems' parameters are assumed as identical.

It should be noted that the masking chaotic signal should exceed the information signal in terms of spectral power and working frequency band width [3]. Thus, chaotic masking may be considered as an additional level of security in information transmission in telecommunication networks. A harmonic signal can be calculated analytically as follows:

$$s(t) = S_0 \sin(\omega_0 t + \varphi_0), \tag{6}$$

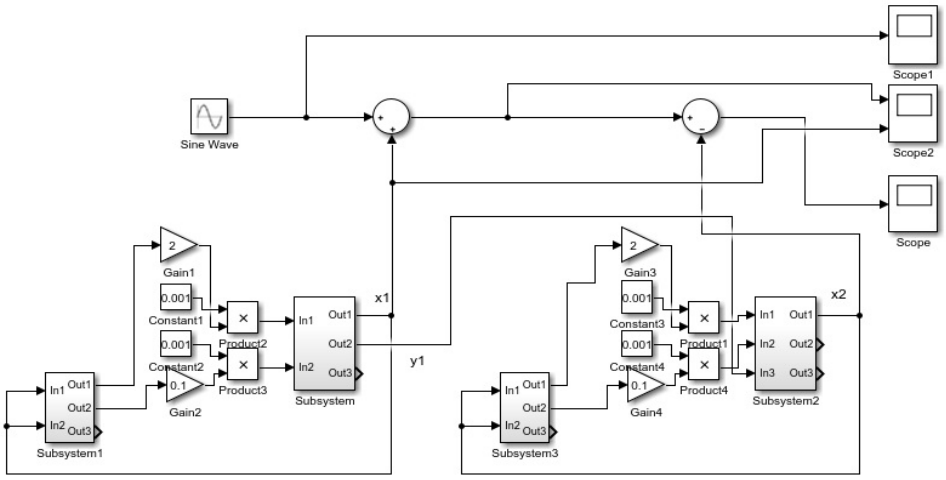

Figure 8: Information transfer model using chaotic masking in Simulink.

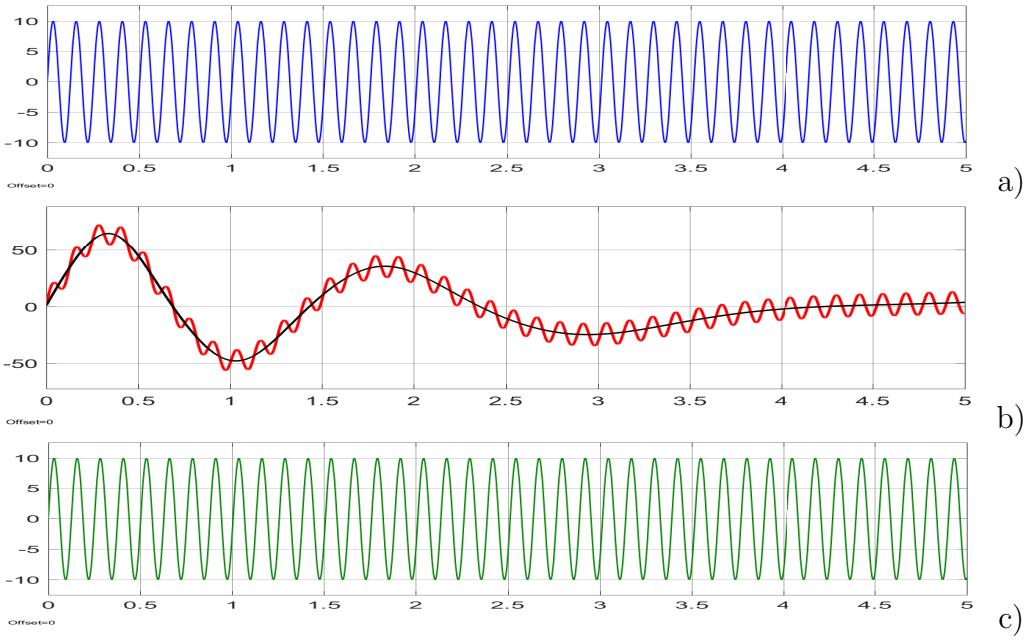

Figure 9: In the temporal domain, chaotic masking of a harmonic signal: a) signal at the input of Subsystem-Subsystem1 (transmitter);b) the chaotic signal is depicted in black, while the mixed signal is depicted in red; c) signal after demodulation of the received signal by Subsystem2-Subsystem3.

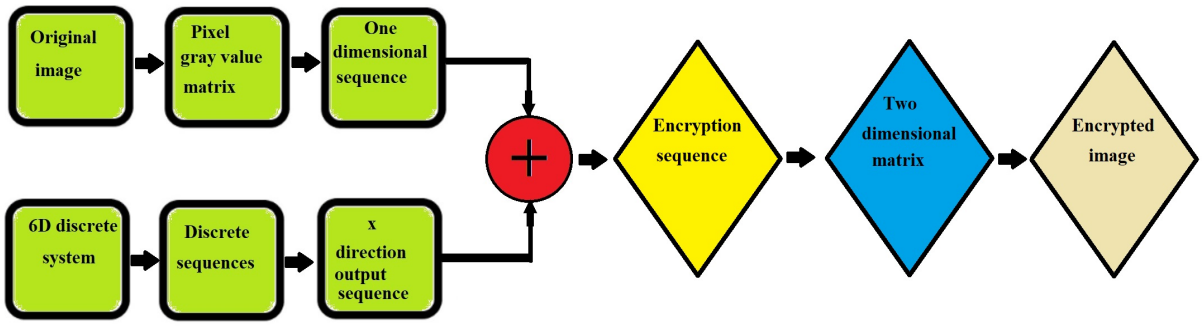

Figure 10: The encryption algorithm.

where S_0, ω_0, φ_0 amplitude, angular frequency, and initial phase of the signal. Fig. 9 depicts the results of a chaotic masking simulation on the example of a harmonic signal. Results Fig. 9 depicts the possibilities of developing effective encryption algorithms and systems for secure real-time information transmission based on chaotic synchronization using a discrete 6D systems.

5 Image encryption by a 6D discrete system

We will use a well-known chaos-based picture encryption algorithm in this section [24],[25]. We are encrypting the picture with x direction output as an example, consider a grayscale 227×303 pixels kitten image. A digital image can be transformed into a two-dimensional matrix, with each element representing a gray pixel value in the image. The matrix is converted into a one-dimensional array once it has been processed. Then the chaotic sequence of the discrete 6D system in the x direction is mixed with the array obtained after converting the two-dimensional matrix into a one-dimensional sequence. As a result, the image is encrypted. The process of encrypting a image is depicted in Fig. 10.

We may encrypt the image using the image encryption procedure. Figure 11 depicts the program's results after encrypting and decrypting a raster image in .jpeg format with a size of 227×303 pixels. We can see immediately from the image encryption that we were unable to obtain any information from the original image, therefore we can conclude that the 6D discrete system encryption algorithm can achieve the effect of image encryption. We also use the MATLAB code for the image decryption methods below. The inverse operation of encryption is decryption. The results are depicted in Fig. 11c.

We can observe from the decrypted images that the decrypted image and the original image are almost identical, indicating good encryption effects.

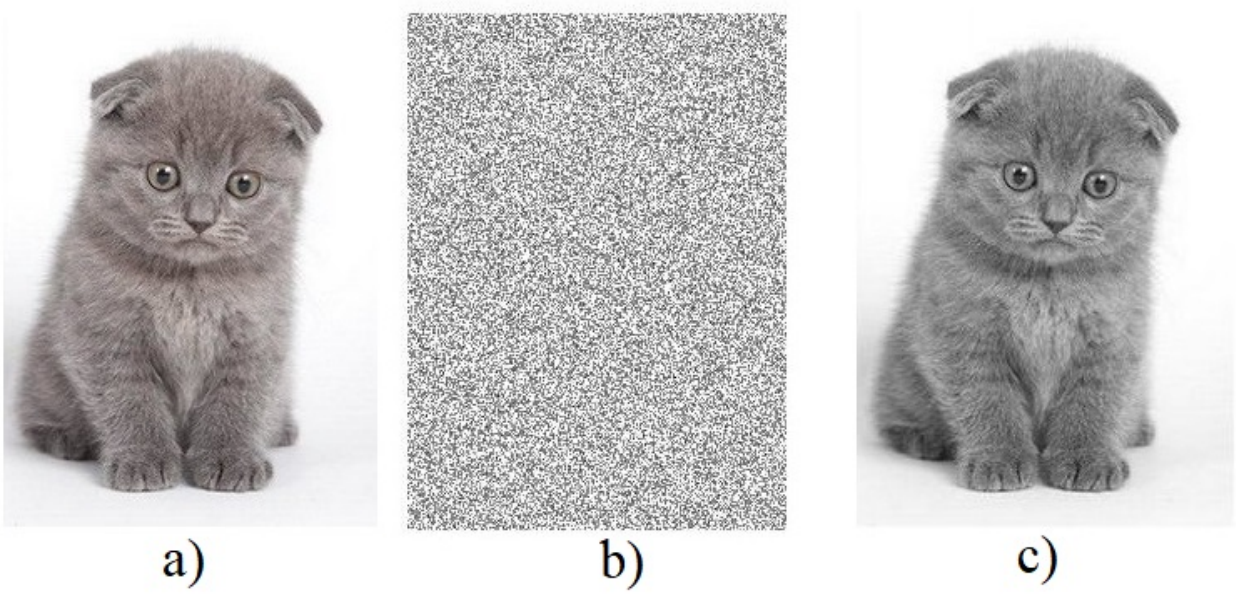

Figure 11: The results of the encryption program: a) original .jpeg image; b) .jpeg image after encryption; c) .jpeg image after decryption.

6 Approbation of the program code for Arduino Uno in the free Tinkercad environment

The Tinkercad system from Autodesk is one of the most popular online services for emulating Arduino's work. Tinkercad is an Arduino emulator that simulates the functioning of electrical circuits and controllers while also implementing almost all of the basic functions of the Arduino IDE, from the editing environment and compiler to the port monitor and connecting libraries. Using the Tinkercad Circuits Arduino, you can draw electronic circuits from a fairly extensive library of virtual instruments and devices. In real-time, you can observe the behavior of the circuit, and check and debug its performance. If you add an Arduino virtual board to such a simulator, you can simulate the behavior of the circuit in your own Arduino projects. In the Tinkercad environment, it is possible to load known sketches, as well as debug sketches for the model being created. As a result of the simulation, the circuit with connected elements works the same way as with the included real board. So, we don't need to buy any components or even the Arduino IDE software, and we don't have to worry about the Arduino Uno board being broken down.

To demonstrate a 6D chaos generator, we need to go to www.tinkercad.com and build our circuit [26]. A simple way to visualize the chaotic behavior of a 6D dynamic system is to build an electrical circuit with six blinking LEDs based on Arduino UNO. Fig. 12 shows an Arduino wiring scheme with LEDs on a breadboard for simulating a 6D chaos generator. The circuit consists of the following components: Arduino Uno board, six LEDs of different colors, and six resistors with a nominal value of 220Ω. We connected an ensemble of six light-emitting diodes to the microcontroller's digital pins (D6-D11) using six 220Ω resistors. To begin performing the simulation, we must program the Arduino UNO board within Tinkercad environment. Using

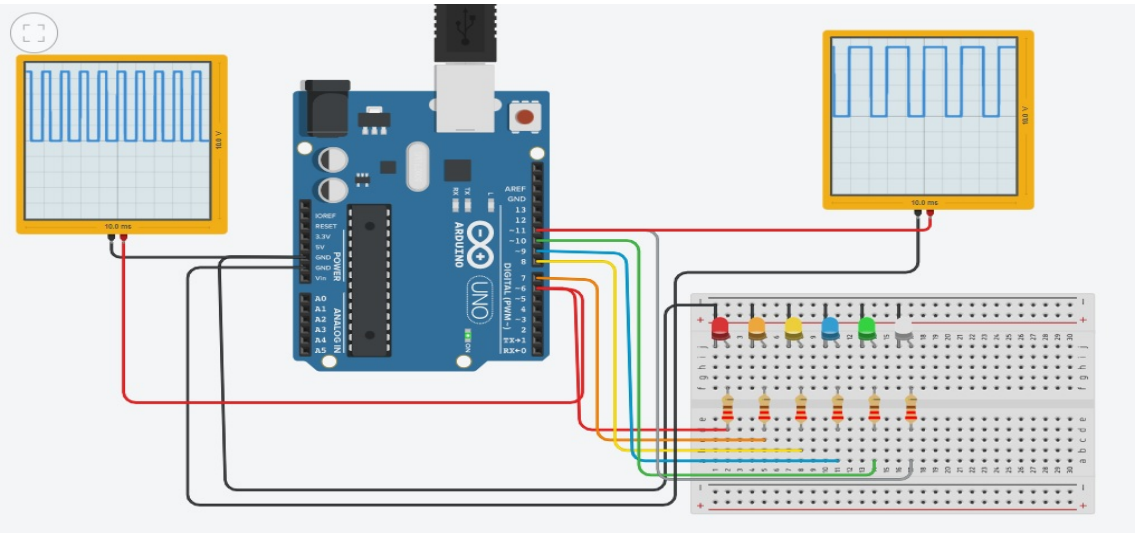

Figure 12: This diagram help us in seeing the connections between the various components. Tinkercad model: 6D chaos.

a language similar to C++, we wrote and debugged the program code, the form the form of which is shown in the Appendix.

As a result of simulating this code, we see pins D6-D11 of the Arduino UNO are utilized as digital outputs to switch on or off the corresponding connected LED. The programmed activation and des-activation times were 10ms. Fig. 12 displays the experiment results using oscilloscopes in Tinkercad environment. On the left side of Fig. 12, the oscilloscope gives the signal output from pin 6 of the Arduino UNO, and on the right side of Fig. 12 output from pin 11.

Thus, the Arduino UNO can be used as a device for generating digital chaotic sequences.

7 Proteus simulations of a chaotic circuit based on a microcontroller Arduino

In this section, the implementation of the 6D chaos generator project (see Fig. 13) in Proteus 8 environment is shown. The circuit is made of an Arduino UNO R3 microcontroller, six LEDs of different colors, and six resistors with a nominal value of 220Ω. In the program code, state variables of the 6D dynamic system x, y, z, u, v, and w are solved numerically using Euler method. We compile the program code presented above in the software Arduino IDE and create a hex.file. This file is to program the Arduino Uno R3 microcontroller in the Proteus 8 environment. The results of the programmed microcontroller are shown in Fig. 14.

8 Experimental realization of discrete 6D chaotic system

For the practical implementation of the 6D discrete system (3), the Arduino UNO was used. The Arduino UNO controller is based on the ATmega328. The platform has a 16 MHz crystal

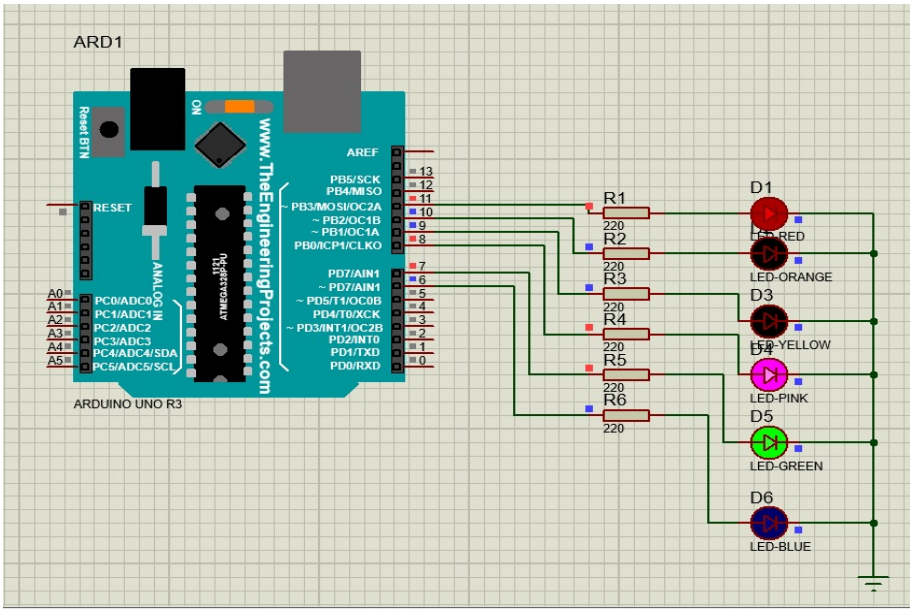

Figure 13: Circuit schematic of the microcontroller-based circuit of 6D chaotic generator depicted in Proteus.

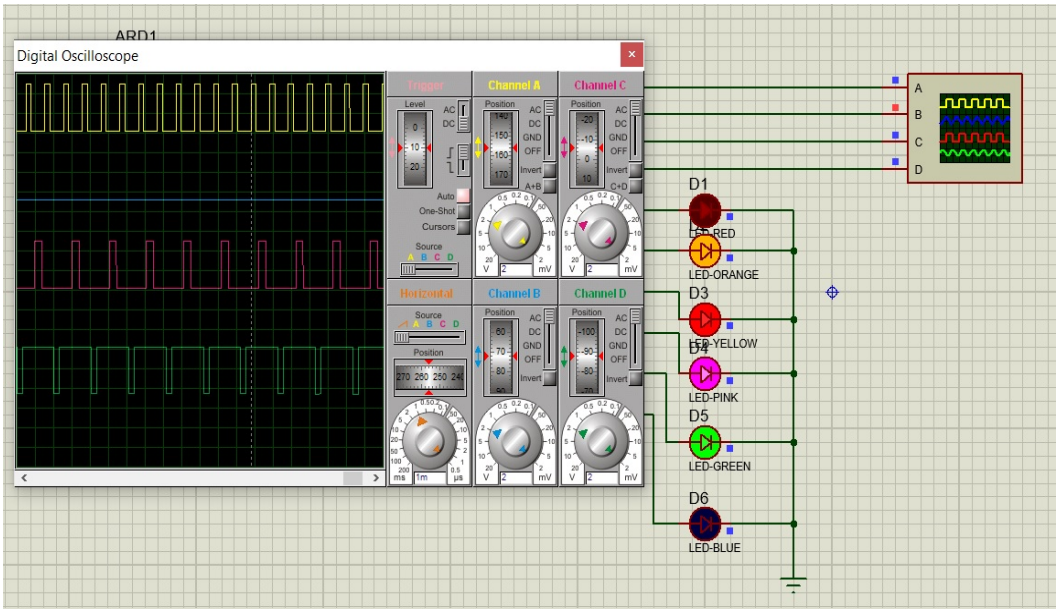

Figure 14: Oscillograms of output signals received from the microcontroller Arduino UNO in Proteus.

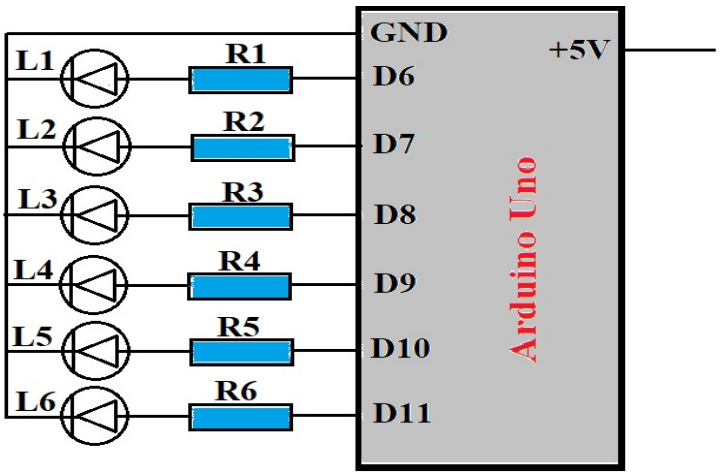

Figure 15: Structure of connections for experimental demonstration of chaotic behavior.

oscillator, 14 digital inputs/outputs (6 of which can be used as PWM outputs), 6 analog inputs, an ICSP connector, a USB connector, a power connector, and a reset button [27]. The Arduino UNO board may be powered in three ways: via the external power connector, via USB, and by the Vin connector, which is connected to one of the combs on the side. The platform has an onboard stabilizer, which allows us to automatically pick the power source while also stabilizing the current to a stable 5 Volts, which is necessary for the controller to operate. External power can be supplied either directly from the computer's USB port or through the power or USB connection from any DC power source.

A connection diagram for an experimental example of chaotic behavior that implements a chaotic 6D discrete system is shown in Fig. 15. Using six 220 ohm resistors, we connected a matrix of six LEDs (L1-L6) to the microcontroller's digital pins (D6-D11) (R1-R6). Fig. 16 depicts an experimental implementation of a 6D discrete system demonstrating chaotic behavior. On the left in Fig. 16, is the breadboard, Arduino UNO, six colored LEDs, and resistors. On the right in Fig. 16, shows the operation of the Arduino UNO board after flashing with the code given in the Appendix. Fig. 16 shows that the brightness of the LEDs depends on the value of the output voltage in each channel.

9 Conclusions

In this work, a computer simulation of a new discrete 6D chaotic system from the article was carried out. In this regard, the following conclusions can be drawn:

1. A Simulink model of a discrete 6D system was developed to describe the chaotic behavior.

2. A model has been developed in the Simulink environment to study the synchronization of unidirectionally connected 6D discrete systems.

3. A Simulink model of chaotic masking of a harmonic signal was built based on the principle of synchronization of 6D discrete systems.

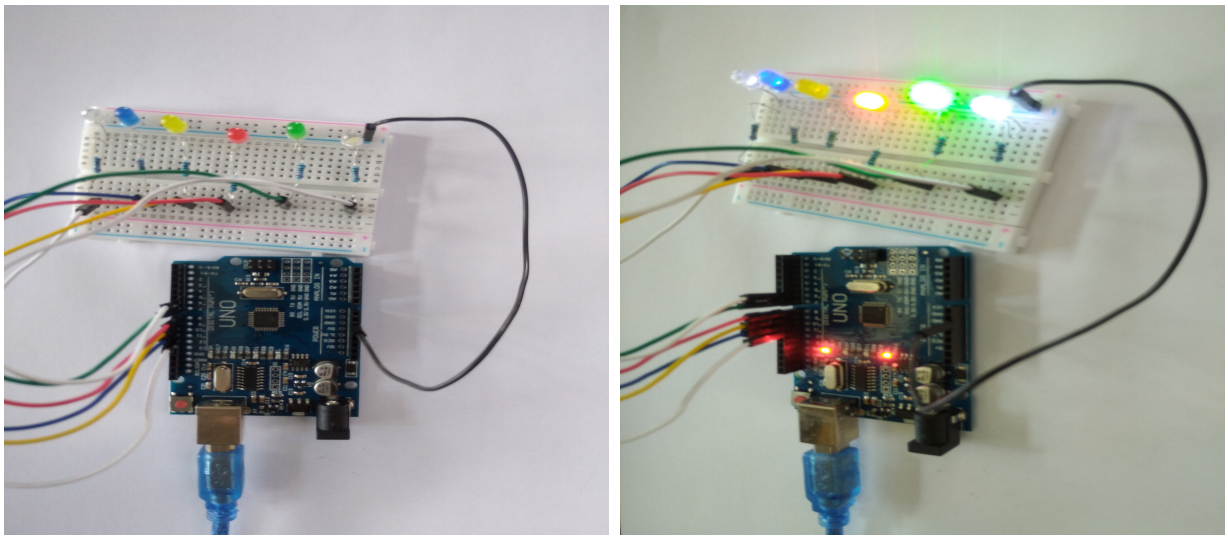

Figure 16: Photo of the experiment's realization using an Arduino UNO board.

4. An image encryption algorithm is realized on the basis of a discrete 6D system.

5. An experiment employing Arduino demonstrated the effectiveness of using discrete chaotic generators in today's digital technologies.

Thus, the simulation results confirm that the proposed Simulink model of the new discrete chaos generator can implement the encryption and decryption of signals and images.

References

[1] F. Yu, et al., Secure Communication Scheme Based on a New 5D Multistable Four-Wing Memristive Hyperchaotic System with Disturbance Inputs, Complexity, vol. 2020, pp. 1-16, January 2020.

[2] A. Ouannas, et al., A Novel Secure Communications Scheme Based on Chaotic Modulation, Recursive Encryption and Chaotic Masking, Alexandria Engineering Journal, vol. 60, no. 1, pp. 1873-1884, December 2020.

[3] S. Wang, et al., An Image Encryption Algorithm Based on a Hidden Attractor Chaos System and the Knuth-Durstenfeld Algorithm, Optics and Lasers in Engineering, vol. 128, Article no. 105995, May 2020.

[4] A. Sambas, et al., A 3D Multi-Stable System with a Peanut-Shaped Equilibrium Curve: Circuit Design, FPGA Realization, and an Application to Image Encryption, IEEE Access, vol. 8, pp. 137116-137132, July 2020.

[5] S. Vaidyanathan, et al., A New Biological Snap Oscillator: Its Modelling, Analysis, Simulations and Circuit Design, International Journal of Simulation and Process Modelling, vol. 13, no. 5, pp. 419-432, January 2018.

[6] K. M. Cuomo, et al., Circuit Implementation of Synchronized Chaos with Applications to Communications, Physical Review Letters, vol. 71, no. 1, pp. 65–68, August 1993.

[7] Q. H. Alsafasfeh, et al., A New Chaotic Behavior from Lorenz and Rossler Systems and Its Electronic Circuit Implementation, Circuits and Systems, vol. 2, no. 2, pp. 101-105, April 2011.

[8] Y. Y. Hou, et al., Rikitake Dynamo System, Its Circuit Simulation and Chaotic Synchronization via Quasi-Sliding Mode Control, TELKOMNIKA Telecommunication, Computing, Electronics, and Control, vol. 19, no. 4, pp. 1428-1438, August 2021.

[9] Khaled Benkouider et al. Dynamics, Control and Secure Transmission Electronic Circuit Implementation of a New 3D Chaotic System in Comparison With 50 Reported Systems, IEEE Access, vol. 9, pp. 152150-152168, 2021. https://doi.org/10.1109/ACCESS.2021.3126655

[10] Ebrahim, Seham Muawadh Ali, Hybrid Chaotic Method for Medical Images Ciphering (2020). International Journal of Network Security & Its Applications (IJNSA) Vol.12, No.6, November 2020, Available at SSRN: https://ssrn.com/abstract=3741050

[11] Salman, F. A., & Salman, K. A., Enhanced Image Encryption Using Two Chaotic Maps. Journal of ICT Research and Applications, vol. 14, no. 2, pp. 134-148, 2020. https://doi.org/10.5614/itbj.ict.res.appl.2020.14.2.3

[12] Wang, J. and Liu, L., A Novel Chaos-Based Image Encryption Using Magic Square Scrambling and Octree Diffusing. Mathematics, vol. 10, pp. 457, 2022. https://doi.org/10.3390/math10030457

[13] A. A. Zaher, On the Discretization of Continuous-Time Chaotic Systems for Digital Implementations, IOP Conf. Series: Journal of Physics: Conf. Series 1141, 012116, 2018. https://doi.org/10.1088/1742-6596/1141/1/012116

[14] James Calusdian and Xiaoping Yun, Microcontroller Generates Chaotic Lorenz Signals, Electronic Design, 2014. http://electronicdesign.com/analog/microcontroller-generates-chaotic-lorenz-signals

[15] Leonardo Acho, A discrete-time chaotic oscillator based on the logistic map: A secure communication scheme and a simple experiment using Arduino, Journal of the Franklin Institute, http://dx.doi.org/10.1016/j.jfranklin.2015.03.028

[16] Mauricio Zapateiro De la Hoz, Leonardo Acho, and Yolanda Vidal, An Experimental Realization of a Chaos-Based Secure Communication Using Arduino Microcontrollers, Hindawi Publishing Corporation, Scientific World Journal, Volume 2015, Article ID 123080, 10 pages. http://dx.doi.org/10.1155/2015/123080

[17] Serdar Cicek, Microcontroller-based Random Number Generator Implementation by Using Discrete Chaotic Maps, Sakarya University Journal of Science, vol. 24, no.5, pp. 832-844, 2020. https://doi.org/10.16984/saufenbilder.727449

[18] S. C. Yenera, C. Barbaros, R. Mutlu and E. Karakulak, Implementation of Microcontroller-Based Memristive Chaotic Circuit, Acta Physica Polonica A, vol. 132, pp. 1058-1061, 2017. https://doi.org/10.12693/APhysPolA.132.1058

[19] Suayb Cagri YENER, Resat MUTLU, Ertugrul KARAKULAK, Implementation of a Microcontroller-Based Chaotic Circuit of Lorenz Equations, Balkan Journal of Electrical and Computer Engineering, 2020. https://doi.org/10.17694/bajece.624645

[20] Volodymyr Rusyn, Aceng Sambas, Maria S. Papadopoulou, Chaotic Lorenz system: analysis of the main information properties, circuit realization and LED visualization using Arduino, Fifteenth International Conference on Correlation Optics, edited by Oleg V. Angelsky, Proc. of SPIE Vol. 12126, 121260R, 2021. https://doi.org/10.1117/12.2615516

[21] Volodymyr Rusyn, Christos H. Skiadas, Aceng Sambas, Analysis, computer modelling and LED visualization of the new modified nonlinear logistic map, Proc. SPIE 12126, Fifteenth International Conference on Correlation Optics, 1212604 (20 December 2021). https://doi.org/10.1117/12.2614694

[22] M. I. Kopp, A. V. Tur, V. V. Yanovsky, Magnetic Convection in a Nonuniformly Rotating Electroconducting Medium, Journal of Experimental and Theoretical Physics, 2018, **127**(6) 1173-1196. https://doi.org/10.1134/S106377611812018X

[23] Michael Kopp and Andrii Kopp, A New 6D Chaotic Generator: Computer Modelling and Circuit Design, International Journal of Engineering and Technology Innovation, 2022.(In press). https://doi.org/10.46604/ijeti.2022.9601

[24] Qi Zhang, Yuchao Guo, Wangshu Li and Qun Ding, Image Encryption Method Based on Discrete Lorenz Chaotic Sequences, Journal of Information Hiding and Multimedia Signal Processing, 2016, **7**(3) 576-586.

[25] Marius Iulian Mihailescu and Stefania Loredana Nita, Cryptography and Cryptanalysis in MATLAB: Creating and Programming Advanced Algorithms, 2021, (Bucharest, Romania, Apress Berkeley, CA). https://doi.org/10.1007/978-1-4842-7334-0

[26] 6D chaos. https://www.tinkercad.com/things/5xPEpVY271U-6dchaos

[27] Arduino UNO board pinout. https://www.circuito.io/blog/arduino-uno-pinout/

10 Appendix

```
//6D chaotic generator
const int LEDx=6;
const int LEDy=7;
const int LEDz=8;
const int LEDu=9;
const int LEDv=10;
const int LEDw=11;
```

```
//dynamic variables
float x=1;
float y=1;
float z=1;
float u=1;
float v=1;
float w=1;
float newx;
float newy;
float newz;
float newu;
float newv;
float neww;
//system parameters
const float r=290;
const float a=2;
const float b=0.1;
const float c=0.1;
const float g=2.67;
const float h=8.21;
const float k=2;
const float l=24.65;
float brightness_x;
float brightness_y;
float brightness_z;
float brightness_u;
float brightness_v;
float brightness_w;
const float dt=0.01;
float maxX=x+1;
float minX=x-1;
float maxY=y+1;
float minY=y-1;
float maxZ=z+1;
float minZ=z-1;
float maxU=u+1;
float minU=u-1;
float maxV=v+1;
float minV=v-1;
float maxW=w+1;
float minW=w-1;
void setup() {
pinMode(LEDx,OUTPUT);
pinMode(LEDy,OUTPUT);
pinMode(LEDz,OUTPUT);
```

```
pinMode(LEDu,OUTPUT);
pinMode(LEDv,OUTPUT);
pinMode(LEDw,OUTPUT);
  // put your setup code here, to run once:
}

void loop() {
maxX=max(maxX,x);
minX=min(minX,x);
maxY=max(maxY,y);
minY=min(minY,y);
maxZ=max(maxZ,z);
minZ=min(minZ,z);
maxU=max(maxU,u);
minU=min(minU,u);
maxV=max(maxV,v);
minV=min(minV,v);
maxW=max(maxW,w);
minW=min(minW,w);
// Euler
newx=x+(-x+r*y-a*u-b*v)*dt;
newy=y+(c*(-y+x-x*z))*dt;
newz=z+(c*(-g*z+x*y))*dt;
newu=u+(-u+c*x)*dt;
newv=v+(-v+h*x+k*w)*dt;
neww=w+(-w-l*u-c*v)*dt;
x=newx;
y=newy;
z=newz;
u=newu;
v=newv;
w=neww;
//setting the intensity of LEDs
brightness_x=255*(x-minX)/(maxX-minX);
brightness_y=255*(y-minY)/(maxY-minY);
brightness_z=255*(z-minZ)/(maxZ-minZ);
brightness_u=255*(u-minU)/(maxU-minU);
brightness_v=255*(v-minV)/(maxV-minV);
brightness_w=255*(w-minW)/(maxW-minW);
analogWrite(LEDx,brightness_x);
analogWrite(LEDy,brightness_y);
analogWrite(LEDz,brightness_z);
analogWrite(LEDu,brightness_u);
analogWrite(LEDv,brightness_v);
analogWrite(LEDw,brightness_w);
```

```
delay(10);
 }
```