Memristive Tabu learning neuron generated multi-wing attractor with FPGA implementation and application in encryption

Quanli Deng, Chunhua Wang, Yichuang Sun *Senior Member, IEEE*, Zekun Deng, Gang Yang

***Abstract*—Memristors, with their unique nonlinear characteristics, are highly suitable for construction novel neural models with rich dynamic behaviors. In this paper, a memristor with piecewise nonlinear state function is introduced into the tabu learning neuron model, resulting in a novel memristive tabu learning neuron model capable of generating a double-wing chaotic butterfly. By modulating the state function of the memristor, we can effectively and easily alter the number of wings of the chaotic butterfly. Equilibrium points analysis further elucidates the mechanism behind the generation of multi-wing chaos. Various numerical simulation techniques, including phase portraits, bifurcation diagrams, Lyapunov exponent spectra, and local attraction basins, are employed to illustrate the dynamical behaviors of the proposed model. Moreover, the newly constructed neuron model is validated using FPGA hardware, with the results aligning with numerical simulations, thereby offering a dependable foundation for a memristor digital circuit-based brain-like neuron model. Lastly, an image encryption application based on the multi-wing chaotic butterfly is developed to demonstrate the potential application of the model.**

***Index Terms*—multi-wing, memristor, tabu learning neuron, FPGA implementation, encryption**

I. Introduction

INVESTIGATING the dynamical behaviors of neural networks can guide us in exploring more appropriate control strategies to achieve neural dynamics in the artificial neural networks. The recurrent neural network proposed by J. J. Hopfield in 1984 [1], has received extensive attention not only because of its engineering applications in optimization problems [2] and content-address memory [3], but also because it can revel some dynamical behaviors of the human brain [4]–[6]. Aimed at solving non-convex optimization problems, Beyer and Ogier introduced tabu learning into the Hopfield neural network (HNN), which enables the state trajectory to climb out of local minima thus performing an efficient search of the energy surface [7].

Dynamical behaviors of tabu learning neurons (TLNs) have attracted attention, prompting a closer examination of the running trajectories of the neural network. Li *et al*. took the memory decay rate as a bifurcation parameter and studied the dynamical behaviors of tabu learning neurons, proving that a single TLN can transition between stable and unstable dynamics through the Hopf bifurcation [8]. Xiao and Cao studied the stability of a discrete-time tabu learning single neuron model, finding that Pitchfork, Flip, and Neimark-Sacker bifurcations occur when the bifurcation parameter exceeds a critical value [9]. Bao *et al*. studied the dynamical behaviors of a non-autonomous TLN by introducing an external input to the TLN, discovering complex neuron firing patterns in their non-autonomous TLN model [10]. Doubla *et al*. introduced and investigated a model of two-neuron tabu learning network based on a composite hyperbolic tangent function as the activation function, demonstrating the bistable property in their novel model [11]. Bao *et al*. presented a non-autonomous single TLN model based on a sinusoidal activation function, which can generate a class of multi-scroll chaotic attractors [12].

The memristor, considered as the fourth fundamental circuit component, has garnered attention due to its unique memory function. Its non-volatile and nonlinear properties make it a prime candidate in neural network applications [13]–[15]. In the realm of dynamical behaviors research, memristors are employed to simulate various nonlinear phenomena in neuron models function [16]–[18]. The memristive neuron models have led to the discovery of more abundant dynamic phenomena, promoting the rapid development of neurodynamics research. For example, Hou *et al*. introduced a memristor into a single TLN to reflect the self-adaption physical processing in biological neurons and found coexisting infinitely many nonchaotic attractors in the novel memristive tabu learning neuron (MTLN) [19]. Njitacke *et al*. introduced a memristor into the TLN and proposed a simple MTLN that can produce an infinite number of coexisted chaotic attractors [20]. Ding *et al*. substituted the external stimulus of a TLN with the memristive current and proposed a novel memristive TLN that can generate a multi-scroll chaotic attractor [21].

Neural systems with multi-scroll or multi-wing chaotic attractors exhibit complex topological structures and abundant dynamic characteristics, holding a significant position in both

Manuscript received Month xx, 2xxx; revised Month xx, xxxx; accepted Month x, xxxx. This work was supported in part by the National Natural Science Foundation of China under Grant 62271197, and in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515011910.

Quanli Deng, Chunhua Wang, Zekun Deng, and Gang Yang are with the College of Information Science and Engineering, Hunan University, Changsha, 410082, China (Corresponding author: Chunhua Wang, e-mail: wch1227164@hnu.edu.cn).

Chunhua Wang is also with the Greater Bay Area Institute for Innovation, Hunan University, Guangzhou, 511300, China.

Yichuang Sun is with the School of Engineering and Computer Science, University of Hertfordshire, Hatfield AL10 9AB, U.K.

engineering applications and the revelation of neural chaotic dynamic behaviors [22]–[24]. However, the current proposed MTLN models cannot generate multi-wing chaotic attractors. To address this gap in knowledge, this article proposes a novel memristive tabu learning neuron model capable of generating multi-wing chaotic attractors by controlling the state function of the memristor. Firstly, a novel memristor model with piecewise nonlinear state function is proposed and verified by numerical simulations. Then, the novel memristor is introduced into a single TLN model to construct the novel MTLN model. Furthermore, the dynamical behaviors of the novel MTLN are analyzed through various methods, including phase portraits, bifurcation diagrams, Lyapunov exponent spectra, and local attraction basins. The simulation results illustrate that the MTLN can generate butterfly-shape chaotic attractors. Additionally, the multi-wing butterfly can be easily generated by adjusting the state function of the memristor. Moreover, a field programmable gate array (FPGA)-based digital hardware implementation of the MTLN model is performed, demonstrating the correctness of numerical model. Finally, a multi-wing MTLN chaotic attractor-based encryption scheme is designed and tested. The numerical simulation results of the encryption system demonstrate that the proposed multi-wing based image encryption has good performance in key sensitivity, information entropy, and robustness in resisting attacks.

The main contributions of this work can be summarized in following aspects:

1) A novel multi-wing butterfly shape attractor generated by a memristive TLN is proposed for the first time.
2) The memristive TLN is implemented based on FPGA digital circuit, which may guide for the hardware implementation of tabu learning neural networks.
3) An image encryption system has been designed based on the multi-wing chaotic attractor, which possesses a large key space, extreme sensitivity to keys, and the ability to resist various attacks, thus effectively ensuring the security of image data.

The rest of the article is organized as follows. Section II describes the model of memristor and the model of the memristive TLN. In Section III, dynamical behaviors of the proposed memristive TLN are studied by the numerical simulation method from multiple perspectives. Section IV designs and implements the memristive TLN based on FPGA. Section V presents a chaotic image encryption scheme based on the multi-wing chaotic sequence, and its security performances are analyzed. Section VI concludes this paper and provides an outlook for future research.

II. Model description

A. Memristor model

A voltage controlled memristor can be described as

$$\left\{ \begin{array}{l} i = f(w)v \\ \frac{dw}{dt} = g(w, v) \end{array} \right. \tag{1}$$

where v and i denote the voltage and current, and w represents the internal state variable of the memrsitor device. The function $g(\cdot)$ defines the switching behavior of the memristor depending on the state variable w and the applied voltage v to the memristor [25].

TABLE I
Comparison of piece-wise memristors

Literature	memristor model	attractor type
Ref. [27]	sgn-based sawtooth function	multi-scroll
Ref. [28]	tanh-base step function	grid multi-scroll
Ref. [29]	sgn-based sawtooth function	multi-structure
this work	piece-wise quadratic function	multi-wing

In alignment with the concept of the general voltage-controlled memristor, we introduce a novel memristor model. The mathematical representation of this memristor model is formulated as

$$\left\{ \begin{array}{l} i = (w)v \\ \dot{w} = 1 - G(v) - 0.1w \end{array} \right. \tag{2}$$

where v, i and w denote the voltage, current, and internal state variable of the memristor, respectively. The state-dependent function $G(v)$ is given by

$$\mathrm{G}(v) = \left\{ \begin{array}{c} G_0 v^2, \quad N = 0 \\ G_0 v^2 - \sum_{n=1}^{N} G_n(\alpha + \beta(\mathrm{sgn}(v - E_n) \\ -\mathrm{sgn}(v + E_n)), \quad N > 0 \end{array} \right. \tag{3}$$

where α, β, G_0, G_n (where G_n=n+2), and E_n (where E_n=n+1) are positive parameters, and sgn$(\cdot)$ represents the sign function. The integer parameter N serves to modulate the number of wings. The piecewise nonlinear state function of the memristor is inspired by the method of constructing multi-wing chaotic attractors from the Lorenz-family chaotic systems. In Ref. [26], the goal of creating multi-wing chaotic attractors is accomplished by extending the unstable saddle-foci from the Lorenz-family chaotic systems. Motivated by prior research, we design the state function of the memristor as a nonlinear function with piecewise characteristics. This segmented approach, in turn, imparts greater complexity in dynamical behaviors to both neuron model and neural network model. Table I presents a selection of piece-wise linear memristor-based neural models and their associated types of chaotic attractors. An examination of the table reveals that the piece-wise memristor models are instrumental in the emergence of complex attractor structures within neural models.

To validate the proposed mathematical model of the memristor, we conducted tests on the pinched hysteresis loops of the memristor under periodic sinusoidal voltage excitation, given by v=Asin(ft). For the sake of generality, the parameters are chosen as G_0=1, α=1.5, β=0.75, N=5. Figs.1(a) and (b) illustrate the hysteresis loops influenced by voltage frequency and amplitude, respectively. The analysis reveals that in the v-i plane, the memristor's hysteresis loops pinched at the origin. The side lobe areas of the loops decrease as the voltage frequency increases, and the loop converge to a single-valued straight line as the frequency approaches infinity. These observations confirm the effectiveness of proposed memristor model [30].

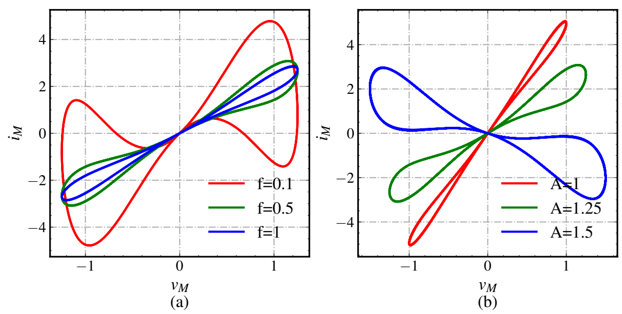

Fig. 1. Hysteresis loops of the memristor model with sinusoidal voltage source v=Asin(ft), (a) different frequencies at A=1V; (b) different amplitudes at f=0.5.

B. Memristive tabu learning neuron model

In the tabu learning neural network , the linear proximity function can be used to perform gradient descent on the energy function [7], thereby resulting in the state equation of the i-th neuron in the network being expressed as

$$\begin{cases} C_i\dot{u}_i = -\frac{1}{R_i}u_i + \sum_j T_{ij}V_j + J_i + I_i \\ \dot{J}_i = -cJ_i - dV_i \end{cases} \tag{4}$$

where C_i, R_i, and u_i represent the membrane capacitor, the membrane resistance, and the membrane potential of the i-th neuron in the network, respectively. T_{ij} is the correspondence connection matrix element for the j-th neuron to the i-th neuron.

Since (4) describes the state variations when the i-th neuron interacts with other neurons in the network. Li *et al.* simplified it during their study of the bifurcation behavior of individual neuron in the neural network, obtaining the single tabu learning neuron as

$$\begin{cases} \dot{x} = -ax + bf(x) + y + I \\ \dot{y} = -cy - df(x) \end{cases} \tag{5}$$

where x and y correspond to state variable u and J in (4), respectively, a, b, c and d are constant parameters, and I is the external input current. Recent studies, such as Ref. [10]–[12], have paid attention to the dynamic behavior of single tabu learning neuron under the influence of memristor. These studies focus on dynamics of individual neuron, laying the foundation for the research on the dynamics of memristive tabu learning neural networks.

Following the concept of studying of the dynamics of a single tabu learning neuron, we integrate the memristor into the state equation of TLN to study neuron's dynamical behavior. For the sake of simplifying the study, we have neglected the external input current I in the model. The resulting novel memristive TLN model is formulated as

$$\begin{cases} \dot{x_1} = -ax_1 + b\tanh(\mathrm{x_1}) + \mathrm{x_2} - \mathrm{kx_3x_1} \\ \dot{x_2} = -cx_2 - d\tanh(\mathrm{x_1}) \\ \dot{x_3} = \mathrm{G(x_1)} - 0.1\mathrm{x_3} - 1 \end{cases} \tag{6}$$

where x_1, x_2, x_3 denote the state variables of the tabu learning neuron and the memristor, respectively. The constants *a, b, c, d* and *k* are positive parameters. The hyperbolic tangent function tanh($\cdot$) serves as the activation function, while G($\cdot$) represents the nonlinear function in memristor. Compared to previous works, the key features of this model are that it can generate doubling chaotic attractor and the convenience of altering the number of wings of the chaotic attractor by modifying the state function of the memristor.

III. Dynamics of Memristive Tabu Learning Neuron Model

In this section, we uncover the intricate dynamics of the proposed MTLN through a combination of theoretical analysis and numerical simulations. The numerical simulations are performed using the MATLAB R2022b software and the ODE45 algorithm, with a fixed time step of 0.001, and time length of 500.

A. Equilibrium points and stability

The stability of equilibrium points is an important characteristic of nonlinear dynamical systems. The equilibrium points of the model (6), which is denoted by **P**=$(\hat{x_1}, \hat{x_2}, \hat{x_3})$, can be determined by the following equations

$$\begin{aligned} f_1(\hat{x_1}, \hat{x_3}) &= -a\hat{x_1} + b\tanh(\hat{\mathrm{x}_1}) - (\mathrm{d/c})\tanh(\hat{\mathrm{x}_1}) - \mathrm{k}\hat{\mathrm{x}_3}\hat{\mathrm{x}_1} \\ f_2(\hat{x_1}, \hat{x_3}) &= \mathrm{G}(\hat{\mathrm{x}_1}) - 0.1\hat{\mathrm{x}_3} - 1 \end{aligned} \tag{7}$$

where $\hat{x_2} = -(d/c)\tanh(\hat{\mathrm{x}_1})$. The value of $\hat{x_1}$ and $\hat{x_3}$ can be seen as the intersection of the curves drawn by (7). Without loss of generality, we set parameters as $a = 1.5, b = 2.5, c = 3.5, d = 17$ and $k = 3.5$ and choose four case of the parameter N in $G(\cdot)$ to plot the the solution of $\hat{x_1}$ and $\hat{x_3}$, as shown in Fig.2.

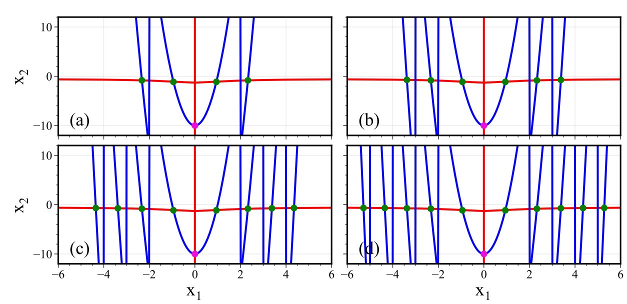

Fig. 2. Numerical simulated curve of (7) in which $f_1(\hat{x_1}, \hat{x_3})$ denoted by red color and $f_2(\hat{x_1}, \hat{x_3})$ denoted by blue color with parameters α=1.5, β=0.75, G_0=1 and (a) N=0, (b) N=1, (b) N=2, (b) N=3.

The Jacobian matrix of the system (6) at the equilibrium points can be calculated by

$$\begin{bmatrix} -a + b\mathrm{sech}^2(\hat{\mathrm{x}_1}) - k\hat{x_3} & 1 & -k\hat{x_1} \\ -d\mathrm{sech}^2(\hat{\mathrm{x}_1}) & -c & 0 \\ 2G_0(\hat{x_1}) & 0 & -0.1 \end{bmatrix}. \tag{8}$$

The equilibrium points and their stability can be numerically determined using MATLAB software, based on the positions identified in Fig.2. The stability of these points, as inferred from the eigenvalues of the Jacobian matrix, is visually represented in the same figure with different colors. Pink-colored stars correspond to the equilibrium points with one positive and two negative eigenvalues of the Jacobian matrix, indicating an unstable index-1 saddle. Green-colored dots represent points with one negative real eigenvalue and a pair of conjugate complex eigenvalues with positive real part, suggesting that these equilibrium points are unstable index-2 saddle-foci. In light of the Shil'nikov theorem [31], the presence of unstable index-2 saddle-foci in the system suggests the potential for chaotic attractors to emerge.

B. Double-wing chaotic attractor generated by MTLN

For the numerical integration of the model (6), we employed a set of randomly chosen initial conditions $x_1(0)$=0.1, $x_2(0)$=0.1, and $x_3(0)$=0.1. The parameters were set to a=1.5, b=2.5, c=3.5, $d = 17$ and $k = 3.5$. Additionally, the parameter N was set to zero in the function $G(\cdot)$. The resulting phase portraits and time domain waveforms of state variables are depicted in Fig.3. specifically, Figs.3(a) and (b) display the phase portraits of the system in $x_1 - x_2$ and $x_1 - x_3$ planes, respectively. Fig.3(c) presents the time domain waveforms of the state variables x_1 and x_3 within 500-second simulation period. As observed in the figures, the MTLN generates a double-wing butterfly-shaped chaotic attractor in the $x_1 - x_3$ plane.

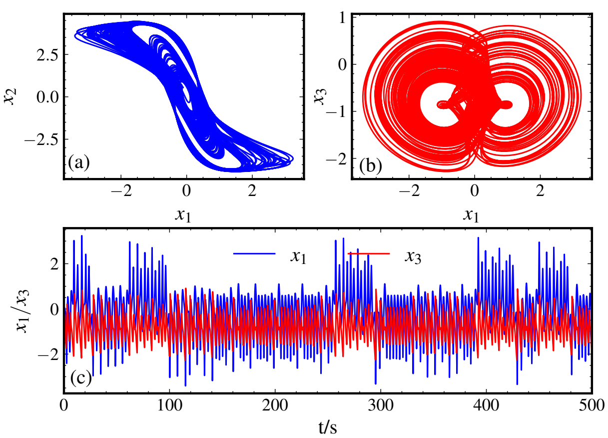

Fig. 3. Numerical simulation results of the model (6) for (a) phase portrait in $x_1 - x_2$ plane, (b) phase portrait in $x_1 - x_3$ plane, (c) time domain wave of the variable x_1 and x_3.

To delve deeper into the dynamics of the model (6), we maintained the parameters a=1.5, and b=2.5 constant while treating the parameter k, memory decay rate c, and learning rate d as adjustable control parameters. Our investigation focused on the Lyapunov exponent spectra and bifurcation diagrams to understand the dynamical behaviors. In a three-dimensional continuous-time system, the state can be determined by the signs of Lyapunov exponents (LEs). If the first LE is positive, the second on equals to zeros and the last one is negative, the system is in a chaotic state, indicating that the system has inherent instability. Furthermore, by examining the bifurcation diagram, one can gain insights into the dynamic evolution process of the system. This analysis enables a clear understanding of the system's long-term behavior under various parameter settings and the transition processes that occur as the system parameters change.

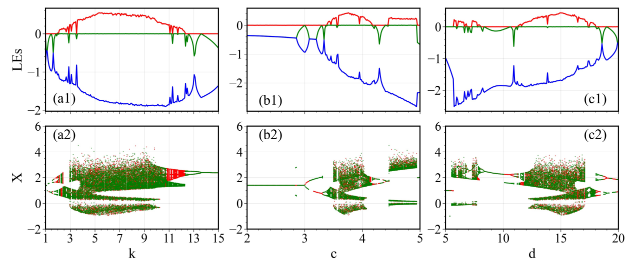

Fig. 4. Lyapunov exponent spectra and bifurcation diagrams (the blue and the red trajectories are with initial condition (0.1,0.1,0.1) and(0.1,-0.1,0.1), respectively) with parameters (a) fixing c=3.5,d=17 varying k in range [1,15], (b) fixing k=3.5,d=17 varying c in range [2,5], (c) fixing k=3.5,c=3.5 varying d in range [5,20].

Fig.4 presents the Lyapunov exponent spectra and the bifurcation diagrams of state variable x_1, with initial conditions (0.1,0.1,0.1), marked in red, and (0.1,-0.1,0.1), marked in green. In Fig.4(a), the influence of memristor on dynamics is examined by fixing the memory decay rate c=3.5, and the learning rate d=17, while varying k in region [1,15]. The chaotic region of the model is observed for $k \in [1.98, 12.36]$, where LE_1 is positive and LE_2 is zero. As depicted in Fig.4(a2), the MLTN demonstrates intricate bifurcation phenomena, including forward period-doubling bifurcation, reverse period-doubling bifurcation, as well as several periodic windows as k increments. An increase in the memory decay rate c leads to the observation of forward period-doubling bifurcation, crisis scenarios, periodic windows, and reverse period-doubling bifurcation, as depicted in Fig.4(b2). The chaotic regions corresponding to k=3.5, d=17 and $c \in [2, 5]$ are determined from Fig.4(b1), which are $c \in [3.36, 4.17]$ and $c \in [4.45, 4.95]$, respectively. When k=3.5, c=3.5, and the learning rate d is varied in the range [5,20], the Lyapunov exponent spectra and bifurcation diagram for x_1 are shown in Fig.4(c). The bifurcation diagram reverses reverse and forward period-doubling bifurcations, along with several periodic windows, as d increases. The numerical simulation results highlight the complex dynamical behaviors that emerge by changing the control parameters of the MTLN. Moreover, the bifurcation diagrams illustrate that different initial values lead to distinct bifurcation trajectories, indicating the multistability phenomenon of the MTLN.

To elucidate the impact of memristor on the dynamical evolution, we selected four representative values of k: 2, 3.5, 11 and 12.5, while keeping the memory decay rate c=3.5 and the learning rate d=17 constant. The phase portraits of the MTLN for these values are depicted in Fig.5. As the parameter k is incremented, the attractor of the system transitions from periodic state to a double-wing chaotic state, then to single-wing chaotic state and ultimately returns to a periodic state. This observed behavior is consistent with the dynamic analysis performed using the Lyapunov exponent spectra and bifurcation diagram with respect to parameter k.

As depicted in Fig.4, varying the initial conditions results in distinct bifurcation trajectories. The phase portraits of coexisting attractors are presented in Fig.5.Specifically, the left-side and right-side periodic attractors (in Fig.5(a)) as well

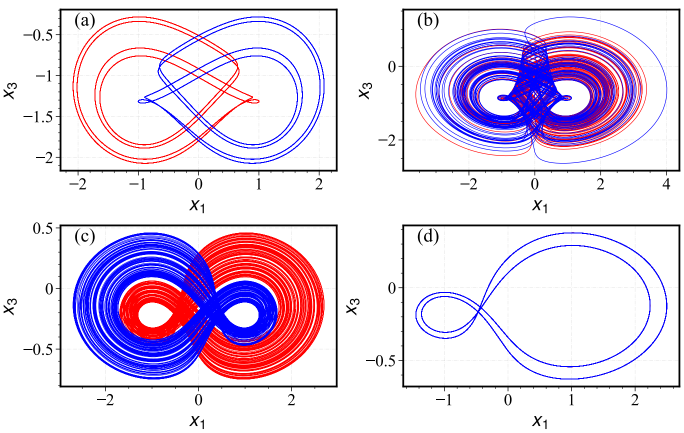

Fig. 5. Phase portraits of the model (6) with c=3.5, d=17 (a) k=2, (b) k=3.5, (c) k=11, (d) k=12.5, where red trajectories are from initial (0.1,0.1,0.1) and blue trajectories are from initial (0.1,-0.1,0.1), respectively.

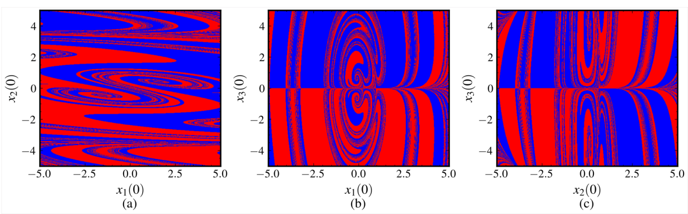

Fig. 6. Attraction basins of the coexisting attractors in (a) $x_1(0) - x_2(0)$ plane, (b) $x_1(0) - x_3(0)$ plane, (c) $x_2(0) - x_3(0)$ plane.

as the left-wing and right-wing chaotic attractors (in Fig.5(c)) correspond to initial conditions I_1=(0.1,0.1,0.1) and I_2=(0.1,-0.1,0.1), respectively. The blue curve represents the trajectory for initial condition I_1, while the red curve represents the trajectory for initial condition I_2. The basins of attraction for nonlinear system provide a visual representation of the dynamical distribution of the nonlinear system. By fixing the system control parameters at k=11, setting the initial value of x_3 to 0.1, and varying the initial values of x_1 and x_2 within the region [-5,5], we obtain the 2D attraction basin shown in Fig.6(a). Similarly, Figs.6(b) and (c) display the attraction basins of the MTLN in the $x_1(0) - x_3(0)$ and $x_2(0) - x_3(0)$ planes, respectively. In Fig.6, the attraction basins are color-coded to indicate the type of attractor: blue regions represent the attraction regions of the right-wing chaotic attractor, while red regions indicate the left-wing chaotic attractor. The figure reveals that the MTLN exhibits complex coexisting attraction basins when generating a single-wing chaotic attractor.

C. Multi-wing chaotic attractors generated by MTLN

In accordance with the memristor model in Section II, when the parameter N in state-dependent function $G(v)$ is positive, additional breaking points are introduced into the state function of the memristor. This can lead to the emergence of multi-wing attractors. Without loss of generality, we have chosen the parameter N in $G(v)$ to be 1, 2, 3, 4, 5, and 6 to show the dynamics of the multi-wing butterfly chaotic attractors. The phase portraits of the MTLN in the $x_1 - x_3$ plane for these values of N are displayed in Figs.7(a) to (f). The system parameters and initial values for these phase portraits are consistently set to a=1.5, b=2.5, c=3.5, d=17, k=3.5 and ($x_1(0)$,$x_2(0)$,$x_3(0)$)=(0.1,0.1,0.1). Observation from Fig.7 indicate that as the parameter N increases, the number of wings of the chaotic attractor transitions from 4-wing to 14-wing.

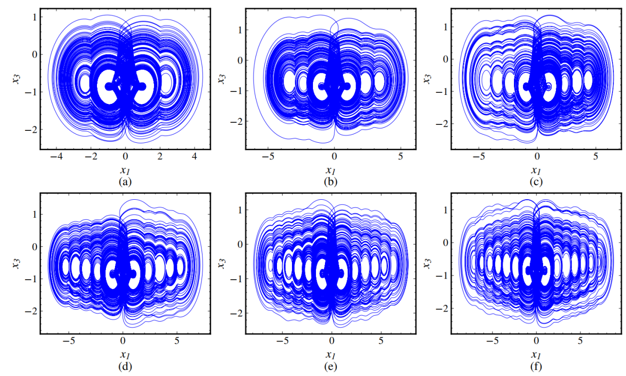

Fig. 7. Phase portraits of the model (6) in $x_1 - x_3$ plane with (a) 4-wing, (b) 6-wing, (c) 8-wing, (d) 10-wing, (e) 12-wing, (f) 14-wing.

To further investigate the dynamics of the multi-wing MTLN, we have plotted the bifurcation diagrams of the state variable X_1 for various control parameter values of k, under different cases of memristor parameter N=1, 2, 3, 4, 5, and 6. These bifurcation diagrams are presented in Fig.8. In contrast to the bifurcation diagrams shown in Fig.4(a), where the function $G(v)$ lacks breakpoints, it is evident that the chaotic regions in the bifurcation diagrams are expanded by the introduction of multi-wing forms. This analysis demonstrates that the complexity of the system's dynamics is enhanced by the presence of breakpoints in the state-dependent function of the memristor, leading to a richer variety of chaotic behaviors.

IV. Hardware implementation of the MTLN

A. Discretization of the MTLN

FPGA-based digital circuit implementations have gained widespread adoption in the design of chaotic neurons, surpassing analog circuit implementations. This preference is

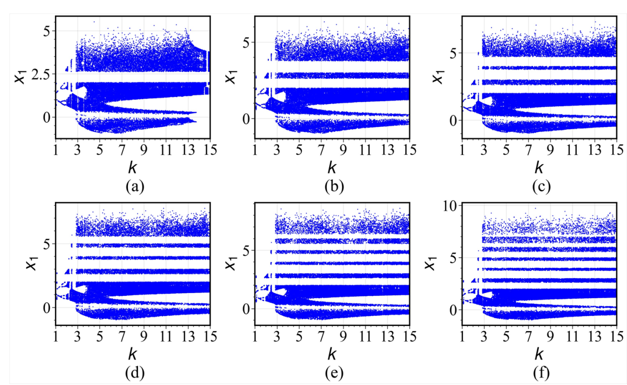

Fig. 8. Bifurcation diagrams of the model (6) with different number of attractor wings.

attributed to the FPGA's high calculation speed, high stability, and convenience of altering system parameters and initial values, as demonstrated in recent studies [32]–[34]. In this article, we implement the proposed MTLN using an FPGA platform. The neuron model presented in (6) can be transformed into a discrete-time system using the fourth-order Runge-Kutta (Rk4) method. For the three state variables, x_1, x_2, and x_3 in (6), we define x_n, y_n, and z_n as the sample values at the start of the *n*-th iteration. Similarly, x_{n+1}, y_{n+1}, and z_{n+1} are defined as the sample values at the beginning of the (*n*+1)-th iteration.

Firstly, three temporary variables x, y, and z are given as $x = x_n$, $y = y_n$, and $z = z_n$. Then we can get

$$\begin{aligned} K_{11} =& -ax_n + b\tanh(\mathrm{x_n}) + \mathrm{y_n} - \mathrm{kz_nx_n} \\ K_{21} =& -cy_n - d\tanh(\mathrm{x_n}) \\ K_{31} =& \mathrm{G(x_n)} - 0.1\mathrm{z_n} - 1 \end{aligned} \tag{9}$$

Secondly, the temporary variables are reassigned as $x = x_n + 0.5\Delta hK_{11}$, $y = y_n + 0.5\Delta hK_{21}$, and $z = z_n + 0.5\Delta hK_{31}$, where Δh is a sampled interval. Then one gets

$$\begin{aligned} K_{12} =& -a(x_n + 0.5\Delta hK_{11}) + b\tanh(\mathrm{x_n} + 0.5\Delta\mathrm{hK_{11}}) \\ &+ (0.5\Delta hK_{21}) - k(z_n + 0.5\Delta hK_{31})(x_n + 0.5\Delta hK_{11}) \\ K_{22} =& -c(0.5\Delta hK_{21}) - d\tanh(\mathrm{x_n} + 0.5\Delta\mathrm{hK_{11}}) \\ K_{32} =& \mathrm{G(x_n + 0.5\Delta hK_{11}) - 0.1(z_n + 0.5\Delta hK_{31}) - 1} \end{aligned} \tag{10}$$

Thirdly, reassigning the temporary variables as $x = x_n + 0.5\Delta hK_{12}$, $y = y_n + 0.5\Delta hK_{22}$, and $z = z_n + 0.5\Delta hK_{32}$, we can get

$$\begin{aligned} K_{13} =& -a(x_n + 0.5\Delta hK_{12}) + b\tanh(\mathrm{x_n} + 0.5\Delta\mathrm{hK_{12}}) \\ &+ (0.5\Delta hK_{22}) - k(z_n + 0.5\Delta hK_{32})(x_n + 0.5\Delta hK_{12}) \\ K_{23} =& -c(0.5\Delta hK_{22}) - d\tanh(\mathrm{x_n} + 0.5\Delta\mathrm{hK_{12}}) \\ K_{33} =& \mathrm{G(x_n + 0.5\Delta hK_{12}) - 0.1(z_n + 0.5\Delta hK_{32}) - 1} \end{aligned} \tag{11}$$

Finally, these three temporary variables are redefined as $x = x_n + \Delta hK_{13}$, $y = y_n + \Delta hK_{23}$, and $z = z_n + \Delta hK_{33}$, and we can obtain

$$\begin{aligned} K_{14} =& -a(x_n + \Delta hK_{13}) + b\tanh(\mathrm{x_n} + \Delta\mathrm{hK_{13}}) \\ &+ (\Delta hK_{23}) - k(z_n + \Delta hK_{33})(x_n + \Delta hK_{13}) \\ K_{24} =& -c(\Delta hK_{23}) - d\tanh(\mathrm{x_n} + \Delta\mathrm{hK_{13}}) \\ K_{34} =& \mathrm{G(x_n + \Delta hK_{13}) - 0.1(z_n + \Delta hK_{33}) - 1} \end{aligned} \tag{12}$$

With (9) – (12), the discrete-time model can be obtained as

$$\begin{cases} x_{n+1} = x_n + \Delta(K_{11} + 2K_{12} + 2K_{13} + K_{14})/6 \\ y_{n+1} = y_n + \Delta(K_{21} + 2K_{22} + 2K_{23} + K_{24})/6 \\ z_{n+1} = z_n + \Delta(K_{31} + 2K_{32} + 2K_{33} + K_{34})/6 \end{cases} \tag{13}$$

In an iterative process, x_n, y_n, and z_n provide data for the system, while x_{n+1}, y_{n+1}, and z_{n+1} provide data for the next iteration.

B. FPGA-based implementation

The hyperbolic tangent function, known for its smooth saturation characteristics, is used as the activation function in the MTLN. However, implementing this activation function in an FPGA-based neuron model presents challenges due to hardware resource limitations. To address this issue, Kwan *et al.* introduced a simple sigmoid-like second-order piecewise activation function that can be directly implemented in hardware and closely approximates the behavior of the hyperbolic tangent function [35]. In pursuit of enhanced hardware efficiency, we adopt this approximated tanh function in our FPGA implementation. The approximation is expressed as

$$\mathrm{tanh(x)} \approx \begin{cases} 1, & M < x \\ \mathrm{H_s(x)}, & \mathrm{-M \le x < M} \\ -1, & x < -M \end{cases} \tag{14}$$

$$\mathrm{H_s(x)} = \begin{cases} x(\mu - \theta x), & 0 \le x < M \\ x(\mu + \theta x), & -M < x < 0 \end{cases} \tag{15}$$

where μ=1 and θ=0.25 represent the slop and gain of $H_s(x)$, respectively, and M=2 determines the length of the middle area of the function.

We have developed an FPGA-based digital circuit for the RK4 algorithm-driven discrete-time system (6) using the Xilinx xc7z020clg400-1 platform, with a sampled interval of 0.0001. The Verilog Hardware Digital Language (Verilog HDL) was utilized to write the program code, and the variable values are outputted through a digital-to-analog converter (DAC) chip (AD9767). A 32-bit fixed-point decimal format, comprising 1 sign bit, 6 integer bits, and 25 decimal bits, is employed for precision. The hardware setup, including the Zynq FPGA, AD9767 DAC, analog oscilloscope, and the Vivado simulation platform is exhibited in Fig.9(a). Additionally, the program flow block diagram is provided in Fig.9(b). The MTLN model features five input signals and three output signals. The CLK and RST are 1-bit input signals used for synchronizing each module unit. To achieve the desired processing speed, a clock frequency of 50MHz was selected for the FPGA implementation. The initial values x_0, y_0, and z_0 are 32-bit fixed-point decimals. The three 32-bit output signals x_n, y_n, and z_n represent the state at the n-th iteration, which are fed back into the MTLN block for the $(n+1)$-th iteration and into the Data Transfer block for DAC output signal preparation. The Data Transfer unit performs truncation of bits [31:18] from the input 32-bit fixed-point decimals. According to the numerical simulation results presented in Fig.7, it can be determined that the minimum value of state variable x_1 is greater than -9.5, and the minimum value of state variable x_3 is greater than -2.5. Therefore, in order to ensure that the output data of the data transfer block is positive, offset values of 9.5 and 2.5 are respectively added to the variables x_n and z_n. Subsequently, the digital signals are converted to analog signals by the DAC and captured by an oscilloscope. The oscilloscope results are presented in Fig.10. These results confirm that the phase diagrams of FPGA-implemented MTLN are consistent with its numerical simulations, validating the FPGA implementation.

V. Application in Image Encryption

Modern cryptographic algorithms, including the Data Encryption Standard (DES), Advanced Encryption Standard (AES), and Rivest-Shamir-Adleman (RSA) possess advantages

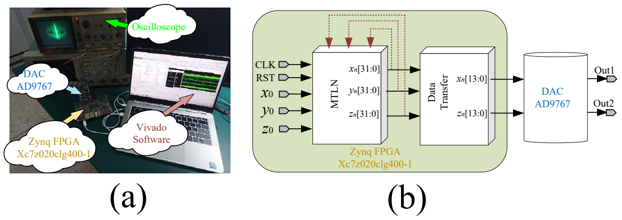

Fig. 9. FPGA-based circuit implementation for the MTLN, (a) hardware experimental prototype with the captured chaotic attractor, (b) flow block diagram of FPGA-based MTLN.

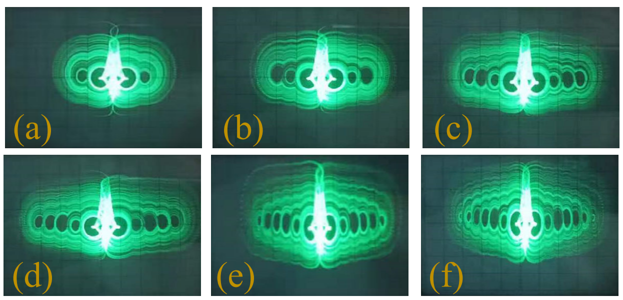

Fig. 10. FPGA-based implementation of multi-wing chaotic attractors in x_1-x_3 plane with (a) 4-wing, (b) 6-wing, (c) 8-wing, (d) 10-wing, (e) 12-wing, (f) 14-wing.

like ability to resistance to a wide range of attacks [36], high speed [37], and efficient hardware implementation [38]. However, these algorithms can encounter challenges when dealing with the large volume of image data and the intricate interdependencies among pixels. Chaotic systems, characterized by their extreme sensitivity and unpredictability, provide a compelling alternative for image encryption [39]–[41]. The adoption of chaos-based encryption methods has gained significant traction in the realm of secure communication and has been successfully implemented in practical applications, including optical communication [42]. Therefore, the exploration of chaos-based encryption techniques is of great importance for their application in secure communication.

A. Description of the cryptosystem

This paper introduces an image encryption scheme that leverages the proposed model (6) to demonstrate its potential application in secure communication. The encryption scheme's architecture is depicted in Fig.11. The encryption process begins with a gray-scale plaintext image P of dimensions M and N. For ease of manipulation, the plaintext image P is stretched into a one-dimensional vector containing M×N elements. The encryption and decryption procedures within the cryptosystem are outlined as follows:

The encryption operation can be decomposed into three main parts: pixel substitution-scrambling-substitution, where f_{D1} and f_{D2} represent the first and second pixel substitution operations, respectively, and f_S denotes the pixel scrambling operation. The pixel substitution operation refers to changing the pixel values of an image, while operation of pixel scrambling represents altering the positions of image pixels. In chaos-based image encryption systems, pixel substitution

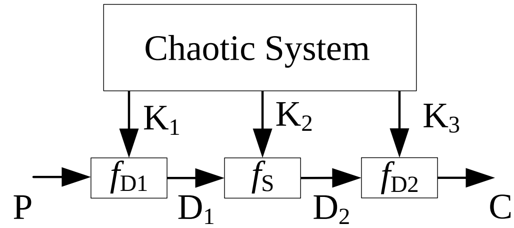

Fig. 11. Chaotic attractor based image encryption scheme.

and scrambling are typically combined to enhance security [43]. The chaotic sequences K_1, K_2 and K_3 are obtained by iterating the system (6) using the initial values set by the user. And they are applied to f_{D1}, f_S and f_{D2} respectively, resulting in intermediate output variables D_1, D_2 and the final encrypted result C.

$$K_1 = mod(floor((Y_1 + Y_2) \times 10^{12}), 256) \tag{16}$$

$$\begin{aligned} D_1(k) = P(k) \oplus mod(D_1(k-1) + P(k-1) \\ + K_1(k), 256) \oplus K_1(k-1) \end{aligned} \tag{17}$$

The chaotic sequence K_1 consists of M×N random numbers within the region [0,255], and its values can be obtained by (16), where Y_1 and Y_2 are the state variables of (6). The k-th element of the intermediate output D_1 in the first pixel substitution operation f_{D1} with K_1 can be calculated by (17), where $P(k)$ and $K_1(k)$ represent the k-th element in the original image P and the chaotic sequence K_1 and $P(k-1)$, $K_1(k-1)$ and $D_1(k-1)$ denote the previous step element of the original image P, chaotic sequence K_1 and intermediate output D_1, respectively.

$$[\sim, K_2] = sort(Y_1 + Y_2 + Y_3) \tag{18}$$

$$D_2(K_2(M \times N - k + 1)) = D_1(K_2(k)) \tag{19}$$

In the pixel scrambling operational, the sum of state variables are used to get the chaotic sequence K_2 through (18), where $sort(\cdot)$ denotes a function that sorts elements in ascending order. The single non-repetitive transformation of intermediate output D_1 is realized by (19), where D_1(k) denotes the k-th element in D_1.

$$K_1 = mod(floor((Y_1 + Y_3) \times 10^{12}), 256) \tag{20}$$

$$\begin{aligned} C(k) = D_2(k) \oplus mod(C(k-1) + K_3(k), 256) \\ \oplus mod(D_2(k-1) + K_3(k-1), 256) \end{aligned} \tag{21}$$

The final encrypted result C is obtained by applying another pixel substitution operation f_{D2} to the scrambled intermediate result D_2. The pixel substitution key K_3 for this operation is generated by (20), where Y_1 and Y_3 represent the first and the third state variables of (6), respectively. In this pixel substitution process, the value of the k-th element is determined by (21), where $D_2(k)$ and $K_3(k)$ represent the k-th element in D_2 and K_3, respectively and $D_2(k-1)$, $K_1(k-1)$ and $C(k-1)$ denote the previous step element of D_2, K_3 and the final output C, respectively.

The primary strengths of the encryption scheme designed in this paper reside in its innovative approach to secure generating current ciphertext. Specifically, it meticulously integrates the current plaintext, preceding plaintext, current key, prior key, and previous ciphertext through bitwise XOR and modulo operations. The strategic integration of the feedback mechanism significantly enhances the security of the encryption system [44]–[46].

The restoration of the original image from the encrypted data is achieved by inverting the encryption process. The MATLAB implementation of the above encryption and decryption algorithms have been uploaded to the github public repository and it can be downloaded by visiting https://github.com/quanliden/MultiwingEncryptionCode.git.

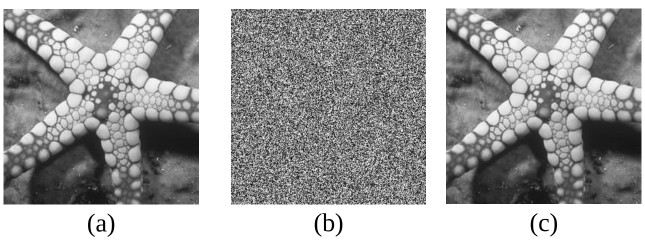

Fig. 12. Simulation results of the cryptosystem where (a) is the original image, (b) is the encrypted image (c) is the decrypted image.

Initialize the values of (6) to 0.1 for each state variables. For the sake of simplicity, all initial values within the pixel substitution process, namely $P(0)$, $D_1(0)$, $K_1(0)$, $D_2(0)$, $K_3(0)$ and $C(0)$ are uniformly set to 1. In practical applications, users have the flexibility to define these initial values. The numerical simulation outcomes of the cryptosystem are demonstrated in Fig.12. The results indicate that the encrypted image no longer contains any visually useful information, while the decrypted image is visually indistinguishable from the original image. This result reflects the effectiveness of the designed cryptosystem in terms of encryption and decryption from a visual perspective.

B. Security analysis

The fundamental components of a cryptosystem are the plaintext, ciphertext, encryption algorithm, decryption algorithm, and key. Adhering to Kerckhoff's principle, the encryption and decryption algorithms of a cryptosystem are made public, with the key being only confidential element. Given the cryptanalyst's knowledge of plaintext and ciphertext, typical attack methodologies can be categorized into four distinct types:

(1) Ciphertext-only attack: The adversary has access to some ciphertext but no corresponding plaintext or key information.

(2) Known-plaintext attack: The adversary is aware of certain plaintext-ciphertext pairs, which can be used to analyze the encryption process.

(3) Chosen-plaintext attack: The adversary is granted temporarily access to the encryption capabilities of the cryptosystem, allowing them to select specific plaintext and obtain the corresponding ciphertext.

(4) Chosen-ciphertext attack: The adversary gains temporary decryption authority within the cryptosystem, enabling them to choose ciphertext and obtain the corresponding plaintext.

In the context of ciphertext-only and known-plaintext attacks, two prevalent strategies are typically employed: brute force and statistical analysis [47]. As for defending against brute force attacks, selecting the initial value of the system (6) as the encryption key. Given the computational precision of a computer at 10^{-14}, the key space of the proposed encryption system is calculated to be 10^{42}, which is exceeds 2^{100}. This substantial key space renders the system highly resistant to brute force attacks [48]. Regarding the defense against statistical attacks, Fig.13 illustrate the statistical characteristics of both the plaintext, Starfish, and its decrypted counterpart. The figure demonstrates that the pixel histogram of the encrypted image is uniformly distributed, devoid of any discernible statistical patterns. Furthermore, the distribution of adjacent pixel pairs covers the entire area, effectively disrupting the original diagonal distribution characteristics. These observations confirm that the designed system is capable of effectively withstanding statistical attacks [49].

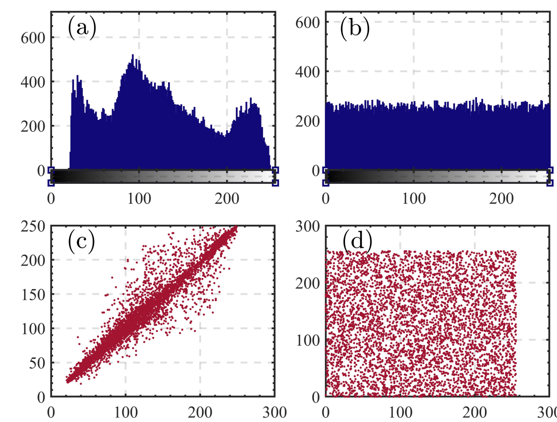

Fig. 13. Statistical analysis of the plaintext and the ciphertext where (a) and (b) are the histogram of the pixels; (c) and (d) are the correlation of adjacent pixels.

$$\begin{cases} D_1(k) = P(k) \oplus mod(D_1(k-1) + P(k-1) \\ \qquad + K_1(k), 256) \oplus K_1(k-1) \\ D_1^0(k) = P^0(k) \oplus mod(D_1^0(k-1) + P^0(k-1) \\ \qquad + K_1(k), 256) \oplus K_1(k-1) \end{cases} \tag{22}$$

$$\Delta D_1(k) \oplus \Delta P(k) = mod((\Delta D_1(k-1) + \Delta P(k-1) + K_1(k)) \oplus K_1(k), 256) \tag{23}$$

$$\begin{aligned} \Delta C(k) \oplus \Delta D_2(k) = {} & mod(C(k-1) + K_3(k), 256) \\ & \oplus mod(D_2(k-1) + K_3(k-1), 256) \\ & \oplus mod(C^0(k-1) + K_3(K), 256) \\ & \oplus mod(D_2^0(k-1) + K_3(k-1), 256) \end{aligned} \tag{24}$$

In scenarios involving plaintext-ciphertext pair attacks, such as chosen-plaintext and chosen-ciphertext attacks, let us delve into the methodology of differential cryptanalysis, a pivotal cryptographic analysis technique for assessing the security of encryption systems. Firstly, we consider the pixel substitution operation in the first round, characterized by operational relationship in (17). Define D_1^0 as an image composed entirely of zero elements. Let P^0 represent the plaintext image obtained after feeding D_1^0 into the first round of the inverse pixel substitution process of the decryption algorithm. To elucidate the relationship between the difference plaintext and the difference ciphertext, we need to rearrange the formula of (22). According to the computational relationship, we derive the relationship between the difference plaintext and the difference ciphertext, as shown in (23). Here ΔP and ΔD_1 denote the difference plaintext and ciphertext, respectively. The formula reveals that the influence of the chaotic random sequence K_1 persists in the form of the difference plaintext-ciphertext pair, thereby indicating that K_1 plays a protective role in this process. Similarly, in the subsequent pixel substitution process, we can get the relationship of the difference plaintext-ciphertext pair as (24). By observing the formula, it can be found that the influence of K_3 also cannot be eliminated. The above analysis based on the plaintext-ciphertext pair verified the keys in the designed encryption system has a protective effect on the encryption process, thereby confirming that the designed system has a certain level of security [50]–[52].

C. Numerical simulations

Key sensitivity analysis: Key sensitivity refers to the property where a minor alteration in the encryption key should prevent the decrypted image from revealing any discernible original image information. Fig.14 illustrates the simulation outcomes, showcasing the successful decryption using the original secret key and unsuccessful attempts with minuscule (10^{-15}) variations in different key values. The results confirm that even the slight change in the secret key makes the retrieval of the original image information from the ciphertext impossible.

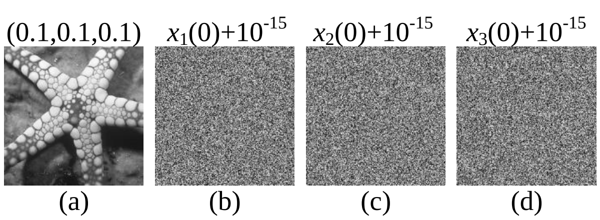

Fig. 14. Decrypted images with (a) correct secret keys, (b)-(d) incorrect secret keys with tiny variations in different initial values.

Histogram analysis: The distribution of pixel values throughout an image is a significant characteristic, and the histogram serves as a valuable tool for visualizing this distribution. In a robust encryption scheme, generating a flat histogram for the encryption image is crucial to defend against statistical attacks. The pixel histograms for the four original images and their corresponding encrypted counterparts are presented in Fig.13 (a) and (b). A comparison of the histograms of the encrypted images (Fig.13(b)) with that of the original images (Fig.13(a)) reveals that the proposed encryption scheme effectively disrupts the correlation within the original image, thereby offering strong defense against statistical attacks.

TABLE II
CORRELATION COEFFICIENTS OF ORIGINAL AND ENCRYPTED IMAGES

Direction	Plain image	Cipher image
Horizontal	0.9582	-0.0154
Vertical	0.9629	0.0084
Diagonal	0.9404	-0.0103

Correlation analysis: The correlation coefficient between adjacent pixels, as defined by (25), is a significant metric for assessing the robustness of the encrypted image. In a plaintext, adjacent pixels typically exhibit a high correlation coefficient, approaching 1 in all directions. Conversely, for an effective encryption scheme, the correlation coefficient of the encrypted image should approach zero, indicating a lack of correlation between adjacent pixels. This property is essential for ensuring that the encrypted image is resistant to pattern recognition and other forms of statistical analysis that could potentially compromise the security of the encrypted data.

$$\rho_{xy} = \frac{\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{\sum_{i=1}^{N}(x_i - E(x))^2}\sqrt{\sum_{i=1}^{N}(y_i - E(y))^2}} \tag{25}$$

The correlation coefficients for the Starfish image in various directions (horizontal, vertical, diagonal) are depicted in Fig13 (c) and (d). Table II provides a summary of these coefficients before and after encryption. The results clearly show that the correlation coefficients for the original image approach 1, whereas those for the encrypted images are nearly zero. This indicates that the encrypted images retain no discernible correlation information from the original image, effectively resisting any attempts to deduce the original content based on correlation analysis.

Differential attack analysis: Differential attack analysis involves manipulating one or more pixel values in the plaintext image to generate a new decrypted image. Subsequently, attackers analyze the differences in pixel values between the two encrypted images to identify patterns that could potentially undermine the encryption algorithm. Quantifying the impact of a single-pixel change in the plaintext image on the resulting ciphertext image is crucial. To this end, the number of pixel change rates (NPCR) and the unified average change intensity (UACI) serves as key metrics. Calculation of NPCR and UACI values can be carried as follows.

$$\begin{aligned} &\text{NCPR} = \textstyle\sum_{i,j} \frac{D(i,j)}{M \cdot N} \times 100\% \\ &\text{D}(i,j) = \begin{cases} 0, & \text{if } C_1(i,j) = C_2(i,j) \\ 1, & \text{if } C_1(i,j) \neq C_2(i,j) \end{cases} \\ &\text{UACI} = \frac{1}{M \cdot N} \textstyle\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100\% \end{aligned} \tag{26}$$

where M and N denote the image width and height of, respectively, C_1 and C_2 represent the cipher image before and after a single-pixel change in the plaintext image, respectively.

The NPCR and UACI values, detailed in Table III, reflect the results of altering a single pixel in the plaintext image

TABLE III
RESULTS OF NCPR AND UACI TEST FOR THE EXPERIMENTAL IMAGES

image	changed position	NPCR(%)	UACI(%)
plaintext	(37,69)	98.87	33.22
ciphertext	(95,21)	99.58	33.38

TABLE IV
COMPARISON OF ENCRYPTION/DECRYPTION SPEED WITH AES

	AES-128	AES-192	AES-256	This work
Encryption	15.71	16.04	16.42	0.53
Decryption	26.47	28.01	29.71	0.47

at random. The close alignment of the calculated NPCR and UACI values with these theoretical benchmarks, as observed in the cipher images produces by the encryption system, demonstrates a high level of resistance to different attacks.

D. Performance comparison

In the field of data encryption, the AES demonstrates robust security due to its intricate mechanisms, including byte substitution, row shifting, column mixing, and other multifaceted operations. However, when applying AES to image encryption, it also encounters several challenges. For instance, image data is characterized by its vast volume, high information redundancy, and strong correlation among adjacent pixels, requiring AES to process numerous data blocks during the encryption process. Despite the notable advantages of AES in the realm of data security, its encryption speed poses a challenge in image encryption tasks. Table IV compares the speed difference between the chaos-based image encryption method proposed in this work and the AES-based encryption method. For fairness, we utilized an open-source, manually coded AES encryption algorithm as specified in [53]. The experiment was conducted on a computer equipped with an AMD Ryzen 7-5800H CPU with a base clock speed of 3.2GHz, complemented by 32GB of RAM. The operating system was Windows 11, and software environment included MATLAB version 9.13 (R2022b). By examining the Table IV, it can be seen that chaos-based encryption offers superior speed performance compared to AES-based image encryption.

E. Analysis of non-ideal characteristics of memristor

Given the issue of device inconsistency inherent in memristors produced by the current manufacturing process, we aim to more realistically account for the limitations of physical memristors within the encryption system. To achieve this, we introduce noise into the state variable z to simulation the device inconsistency characteristics that memristors exhibit during their operational lifecycle. Within the system described by (6), the state variable z represents the resistance value of the memristor. Consequently, incorporating noise into this state variable enables us to mimic the non-ideal characteristics of physical memristors to a certain extent. The formulation for the simulated non-ideal characteristics in the memristor can be written as

$$\hat{z} = z + k(max(z) - min(z))rand(N) \tag{27}$$

where $\hat{z}$ is the resistance value affected by non-ideal phenomena, k is the variation strength from the ideal value of z and N is the total number of state variable z.

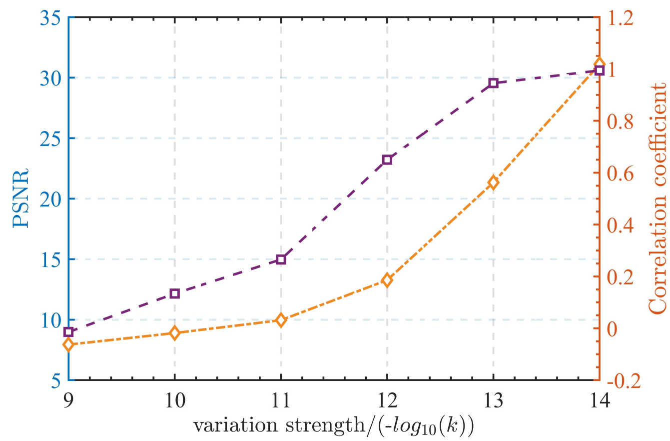

Fig. 15. Non-ideal characteristics of memristor effects on decrypted image measured by PSNR and correlation coefficient.

Taking the image of parrot as an example, Fig. reffig:devva shows the simulated results of the decrypted image effect caused by the mismatch degree of the memristor during the decryption process. The peak signal-to-noise ratio (PSNR) and the correlation coefficient between the original image and decrypted image are selected as indicators to measure the impact caused by the variation of memristor in the decryption process. For clarity, the reciprocal of the logarithm of k is used as the horizontal axis, and the larger the value of the horizontal axis, the less influence of the non-ideal factors on the memristor during the decryption. It can be observed from the figure that the non-ideal state of the memristor during the decryption process has a significant impact on the performance of the decrypted image. When k is approximately 10^{-14}, the PSNR value exceeds 30dB, indicating that the quality of the decrypted image obtained at this time is comparable to that of the original image. Through the analysis of this process, we can find that during the decryption process, when the resistance value of the memristor is affected by strong non-ideal characteristics such as device-to-device variability, it will severely affect the quality of the decrypted image. Therefore, improving the manufacturing process to fabricate memristors with high consistency is of great significance for the application and promotion of memristors.

VI. CONCLUSION AND OUTLOOK

This paper introduces, for the first time, a butterfly-shaped double-wing chaotic attractor generated by a memristive TLN. The number of chaotic butterfly wings can be effectively extended by simply manipulating the state function of the memristor. The rich dynamical properties of the proposed MTLN are verified through multiple numerical simulations, such as Lyapunov exponent spectra, bifurcation diagrams, and attraction basins. This lays the groundwork for revealing chaotic dynamical phenomena in neurons. The digital circuit design, implemented on FPGA, not only validates the proposed mathematical model but also offers a reliable circuit model reference for hardware research in brain-inspired computing using memristor-based digital circuits. Furthermore, the image encryption application proposed in this paper, which leverages the multi-wing MTLN, demonstrates excellent key sensitivity, good resistance to statistical attacks, and robustness against noise and data loss attacks through various analytical methods.

In the aspect of building neural systems with complex dynamical behaviors, we will devote to study neuron and neural network models that incorporate complex topology and intricate dynamics, in the future. In encryption applications, due to the extreme key sensitivity of the encryption system, there may be unpredictable impacts on the system caused by the variability of memristor devices. In our future works, we will conduct in-depth research on this issue from the perspective of using error correction codes, adaptive management of keys and other approaches.

REFERENCES

[1] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the national academy of sciences*, vol. 81, no. 10, pp. 3088–3092, 1984.

[2] Z. Uykan, "On the working principle of the Hopfield neural networks and its equivalence to the GADIA in optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3294–3304, 2019.

[3] R. McEliece, E. Posner, E. Rodemich, and S. Venkatesh, "The capacity of the Hopfield associative memory," *IEEE transactions on Information Theory*, vol. 33, no. 4, pp. 461–482, 1987.

[4] B. Bao, C. Chen, H. Bao, X. Zhang, Q. Xu, and M. Chen, "Dynamical effects of neuron activation gradient on Hopfield neural network: numerical analyses and hardware experiments," *International Journal of Bifurcation and Chaos*, vol. 29, no. 04, p. 1930010, 2019.

[5] C. Chen, F. Min, J. Cai, and H. Bao, "Memristor synapse-driven simplified Hopfield neural network: Hidden dynamics, attractor control, and circuit implementation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 5, pp.2308–2319, 2024.

[6] C. Wang, J. Liang and Q. Deng, "Dynamics of heterogeneous Hopfield neural network with adaptive activation function based on memristor," *Neural Networks*, vol. 178, p.106408, 2024.

[7] D. A. Beyer and R. G. Ogier, "Tabu learning: a neural network search method for solving nonconvex optimization problems," in *[Proceedings] 1991 IEEE International Joint Conference on Neural Networks*. IEEE, 1991, pp. 953–961.

[8] C. Li, G. Chen, X. Liao, and J. Yu, "Hopf bifurcation and chaos in tabu learning neuron models," *International Journal of Bifurcation and Chaos*, vol. 15, no. 08, pp. 2633–2642, 2005.

[9] M. Xiao and J. Cao, "Bifurcation analysis on a discrete-time tabu learning model," *Journal of computational and applied mathematics*, vol. 220, no. 1-2, pp. 725–738, 2008.

[10] B. Bao, L. Hou, Y. Zhu, H. Wu, and M. Chen, "Bifurcation analysis and circuit implementation for a tabu learning neuron model," *AEU-International Journal of Electronics and Communications*, vol. 121, p. 153235, 2020.

[11] I. S. Doubla, Z. T. Njitacke, S. Ekonde, N. Tsafack, J. D. D. Nkapkop, and J. Kengne, "Multistability and circuit implementation of tabu learning two-neuron model: application to secure biomedical images in IoMT," *Neural Computing and Applications*, vol. 33, pp. 14 945–14 973, 2021.

[12] H. Bao, R. Ding, B. Chen, Q. Xu, and B. Bao, "Two-dimensional non-autonomous neuron model with parameter-controlled multi-scroll chaotic attractors," *Chaos, Solitons & Fractals*, vol. 169, p. 113228, 2023.

[13] Q. Deng, C. Wang, J. Sun, Y. Sun, J. Jiang, H. Lin, and Z. Deng, "Nonvolatile CMOS memristor, reconfigurable array, and its application in power load forecasting," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 4, pp. 6130–6141, 2024.

[14] J. Lu, X. Xie, Y. Lu, Y. Wu, C. Li, and M. Ma, "Dynamical behaviors in discrete memristor-coupled small-world neuronal networks," *Chinese Physics B*, vol. 33, no. 4, 2024.

[15] Q. Deng, C. Wang, and Z. Deng, "Memristive circuit of quaternion multiplication and its application in aircraft attitude computation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. Early Access, 2024, doi=10.1109/TCSII.2024.3373017.

[16] H. Bao, M. Hua, J. Ma, M. Chen, and B. Bao, "Offset-control plane coexisting behaviors in two-memristor-based Hopfield neural network," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 10, pp. 10 526–10 535, 2023.

[17] F. Li, L. Bai, Z. Chen, and B. Bao, "Scroll-growth and scroll-control attractors in memristive bi-neuron Hopfield neural network," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 4, pp. 2354–2358, 2024.

[18] J. Li, C. Wang, and Q. Deng, "Symmetric multi-double-scroll attractors in Hopfield neural network under pulse controlled memristor", *Nonlinear Dynamics*, 2024, doi=https://doi.org/10.1007/s11071-024-09791-6.

[19] L. Hou, H. Bao, Q. Xu, M. Chen, and B. Bao, "Coexisting infinitely many nonchaotic attractors in a memristive weight-based tabu learning neuron," *International Journal of Bifurcation and Chaos*, vol. 31, no. 12, p. 2150189, 2021.

[20] Z. T. Njitacke, J. D. D. Nkapkop, V. F. Signing, N. Tsafack, M. E. Sone, and J. Awrejcewicz, "Novel extreme multistable tabu learning neuron: Circuit implementation and application to cryptography," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 8, pp. 8943–8952, 2023.

[21] R. Ding, H. Bao, N. Wang, H. Wu, and Q. Xu, "Generating multi-scroll chaotic attractor in a three-dimensional memristive neuron model," *Chinese Journal of Physics*, vol. 88, pp. 1053-1067, 2024.

[22] M. Ma, Y. Lu, "Synchronization in scale-free neural networks under electromagnetic radiation," *Chaos*, vol. 34, p. 033116, 2024.

[23] Q. Deng, C. Wang, and H. Lin, "Chaotic dynamical system of Hopfield neural network influenced by neuron activation threshold and its image encryption," *Nonlinear Dynamics*, vol. 112, pp. 6629–6646, 2024.

[24] H. Lin, C. Wang, and Y. Sun, "A universal variable extension method for designing multiscroll/wing chaotic systems," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 7, pp. 7806–7818, 2024.

[25] L. Chua, "Everything you wish to know about memristors but are afraid to ask," *Handbook of Memristor Networks*, pp. 89–157, 2019.

[26] S. Yu, W. K. S. Tang, J. Lv and G. Chen,"Design and implementation of multi-wing butterfly chaotic attractors via Lorenz-type systems,"*International Journal of Bifurcation and Chaos*, vol. 20, no. 1, pp.29–41, 2010.

[27] H. Bao, Z. Chen, M. Chen, Q. Xu, and B. Bao, "Memristive-cyclic Hopfield neural network spatial multi-scroll chaotic attractors and spatial initial-offset coexisting behaviors," *Nonlinear Dynamics*, vol. 111, pp. 22 535–22 550, 2023.

[28] Q. Lai, Z. Wan, and P. D. K. Kuate, "Generating grid multi-scroll attractors in memristive neural networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 3, pp. 1324–1336, 2023.

[29] D. Tang, C. Wang, H. Lin, and F. Yu, "Dynamics analysis and hardware implementation of multi-scroll hyperchaotic hidden attractors based on locally active memristive Hopfield neural network," *Nonlinear Dynamics*, vol. 112, no. 2, pp. 1511–1527, 2024.

[30] S. P. Adhikari, M. P. Sah, H. Kim, and L. O. Chua, "Three fingerprints of memristor," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 11, pp. 3008–3021, 2013.

[31] C. P. Silva, "Shil'nikov's theorem-a tutorial," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40, no. 10, pp. 675–682, 1993.

[32] F. Yu, C. Wu, Y. Lin, S. He, W. Yao, S. Cai and J. Jin, "Dynamic analysis and hardware implementation of multi-scroll Hopfield neural networks with three different memristor synapses," *Nonlinear Dynamics*, vol. 112, pp. 12393–12409, 2024.

[33] M. Ji'e, H. Peng, S. Duan, L. Wang, F. Zhang, and D. Yan, "Design and FPGA implementation of grid-scroll hamiltonian conservative chaotic flows with a line equilibrium," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 32, no. 4, pp. 658–668, 2024.

[34] F. Yu, X. Kong, W. Yao, J. Zhang, S. Cai, H. Lin, and J. Jin, "Dynamics analysis, synchronization and FPGA implementation of multiscroll Hopfield neural networks with non-polynomial memristor," *Chaos, Solitons & Fractals*, vol. 179, p. 114440, 2024.

[35] H. K. Kwan, “Simple sigmoid-like activation function suitable for digital hardware implementation,” *Electronics letters*, vol. 15, no. 28, pp. 1379–1380, 1992.

[36] A. Kaminsky, M. Kurdziel, and S. Radziszowski, “An overview of cryptanalysis research for the advanced encryption standard,” in *2010 - MILCOM 2010 MILITARY COMMUNICATIONS CONFERENCE*, 2010, pp. 1310–1316.

[37] J. Blocklove, S. Farris, M. Kurdziel, M. Łukowiak, and S. Radziszowski, “Hardware obfuscation of the 16-bit S-box in the MK-3 cipher,” in *2021 28th International Conference on Mixed Design of Integrated Circuits and System*, 2021, pp. 104–109.

[38] M. J. Foster, M. Łukowiak, and S. Radziszowski, “Flexible HLS-based implementation of the karatsuba multiplier targeting homomorphic encryption schemes,” in *2019 MIXDES - 26th International Conference ”Mixed Design of Integrated Circuits and Systems”*, 2019, pp. 215–220.

[39] C. Wang, D. Tang, H. Lin, F. Yu, and Y. Sun, “High-dimensional memristive neural network and its application in commercial data encryption communication,” *Expert Systems with Applications*, vol. 242, pp. 122513, 2024.

[40] Q. Deng, C. Wang, and H. Lin, “Memristive Hopfield neural network dynamics with heterogeneous activation functions and its application,” *Chaos, Solitons & Fractals*, vol. 178, p. 114387, 2024.

[41] X. Kong, F. Yu, W. Yao, S. Cai, J. Zhang, and H. Lin, “Memristor-induced hyperchaos, multiscroll and extreme multistability in fractional-order HNN: Image encryption and FPGA implementation,” *Neural Networks*, vol. 171, pp. 85–103, 2024.

[42] A. Argyris, D. Syvridis, L. Larger, V. Annovazzi-Lodi, P. Colet, I. Fischer, J. García-Ojalvo, C. Mirasso, L. Pesquera, and K. Shore, ”Chaos-based communications at high bit rates using commerical fibre-optic links”, *Nature*, vol. 438, pp. 343–346, 2005.

[43] Y. Zhang, Z. Hua, H. Bao, H. Huang and Y. Zhou, ”Generation of n-dimensional hyperchaotic maps using Gershgorin-type theorem and its application”, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no.10, pp.6516–6529, 2023.

[44] H. Wen, and S. Yu, ”Cryptanalysis of an image encryption cryptosystem based on binary bit planes extractionn and multiple chaotic maps” *The European Physical Journal Plus*, vol. 134, no. 337, 2019.

[45] R. Zhou, S. Yu, and Q. Wang, ”Security analysis of a chaotic encryption algorithm related to the sum of plaintext pixel value”, *Applied Physics B*, vol. 129, no. 92, 2023.

[46] J. Chen, L. Chen, and Y. Zhou, ”Cryptanalysis of image cipher with permutation-substitution network and chaos”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 6, pp. 2494–2508, 2021.

[47] X. Sun, Z. Chen, L. Wang and C. He, ”A lossless image compression and encryption algorithm combing JPEG-LS, neural network and hyperchaotic system”, *Nonlinear Dynamics*, vol. 111, pp. 15445–15475, 2023.

[48] G. Alvarez and S. Li, ”Some basic cryptanalysis requirements for chaos-based cryptosystems”, *International Journal of Bifurcation and Chaos*, vol. 16, no. 08, pp. 2129–2151, 2006.

[49] P. Liu, X. Wang, Y. Su, H. Liu and S. Unar, ”Globally coupled private image encryption algorithm based on infinite interval spatiotemporal chaotic system”, *IEEE Transactions on Circuit and Systems I: Regular Papers* vol. 70, no. 6, pp. 2511–2522, 2023.

[50] B. Chen, S. Yu, D. Li, and J. Lv, ”Cryptanalysis of some self-synchronous chaotic stream ciphers and their improve schemes”, *International Journal of Bifurcation and Chaos*, vol. 31, no. 08, pp. 2150142, 2021

[51] Q. Jiang, S. Yu, and Q. Wang, ”Cryptanalysis of an image encryption algorithm based on two-dimensional hyperchaotic map”, *Entropy*, vol. 25, no. 3, pp. 395, 2023.

[52] R. Zhou, S. Yu, ”On the divide-and-conquer attack of a plaintext related image chaotic encryption scheme”, *Nonlinear Dynamics*, vol. 112, pp. 11571–11594, 2024.

[53] David Hill (2024). Advanced Encryption Standard (AES)-128,192, 256. https://www.mathworks.com/matlabcentral/fileexchange/73412-advanced-encryption-standard-aes-128-192-256