Design of Multi-Scroll Memristive Hopfield Neural Network and Application to Synchronous Coverage Search

Qianqian Zhang(张倩倩)[1] , Qingyu Shi(石擎宇)[1,*] Xia Huang(黄霞)[1] ,
and Zhen Wang(王震)[2]
[1]College of Electrical Engineering and Automation,
Shandong University of Science and Technology, Qingdao 266590, China
[2]College of Mathematics and Systems Science,
Shandong University of Science and Technology, Qingdao 266590, China

September 17, 2026

Abstract

This paper proposes a multi-scroll memristive Hopfield neural network (M-HNN) and applies it to air-ground collaborative synchronization search tasks. To be specific, a hyperbolic-tangent memristive synapse is first constructed to generate a chaotic multi-scroll M-HNN with a controllable number of scroll attractors, and the memristor flux is then employed as the driving signal for mobile robot motion control. By exploiting the ergodic property of the designed multi-scroll M-HNN, the search capability of the chaotic mobile robot is enhanced. When the air-ground collaborative search task is further considered, the problem can be reformulated as a synchronization tracking control problem for a chaotic mobile robot guided by a chaotic unmanned aerial vehicle (UAV). To address this problem, a critic-only integral reinforcement learning (IRL)-based synchronization strategy is developed to minimize the synchronization tracking error. Finally, the dynamics of the multi-scroll M-HNN are analyzed, and the effectiveness and superiority of the proposed synchronous coverage search algorithm are validated.

Keywords: Memristive Hopfield neural network, multi-scroll chaos, memristor, drive-response synchronization, mobile robot, integral reinforcement learning

PACS: 05.45.-a, 05.45.Gg, 05.45.Xt

1. Introduction

In coverage search and inspection tasks, especially in air-ground collaborative search scenarios, autonomous agents are often required to achieve uniform and efficient spatial coverage in unknown or weakly structured environments [1], [2]. To this end, chaotic dynamical systems have been widely employed as motion-driving mechanisms within path planning and coverage control frameworks [3], [4], [5]. Under conditions of limited computational precision, classical low-dimensional dynamical models or conventional neural network-based chaotic

*Corresponding author. E-mail: shiqingyu08@163.com

generators often exhibit periodic or quasi-periodic behaviors over long-term evolution, thereby limiting trajectory diversity and exploration efficiency. In contrast, nonlinear dynamical driving mechanisms with higher complexity can generate non-periodic, persistently exciting trajectories, thereby enhancing global coverage performance [6]. Therefore, developing nonlinear models with higher dynamical complexity and stronger ergodic properties is crucial for improving the coverage performance of autonomous agents.

In the study of nonlinear dynamics in neural networks, chaotic behavior has attracted considerable attention due to its inherent non-periodicity and ergodic properties. Building on the study of chaotic behavior in neural networks, researchers have explored the application of chaotic systems in robot path planning and coverage search tasks [7], [8]. These studies demonstrate that classical chaotic systems can provide a foundation for generating complex motion patterns; however their attractor structures are relatively limited and often insufficient to support highly unpredictable and diverse trajectories. As a result, multi-scroll chaotic attractors have gradually become a research focus [9], [10], [11].

Memristors, owing to their intrinsic memory effects and strong nonlinearity, provide an effective means for constructing multi-scroll chaotic systems, and M-HNNs have emerged as an important framework for generating such multi-scroll chaotic dynamics [12], [13], [14]. Several studies have demonstrated the realization of multi-scroll attractors using M-HNNs. For example, pulse-controlled memristors embedded in Hopfield neural networks can generate symmetric multi-double-scroll attractors [15], and both multi-scroll attractors and coexisting attractors that split into centrosymmetric chaotic attractors can be induced by memristive synaptic coupling in cyclic neural networks [16]. Beyond M-HNN-based realizations, multi-scroll chaotic attractors have also been achieved in fractional-order systems through nonlinear switching mechanisms with circuit validation [17]. More broadly, chaotic dynamics in memristive neurons and M-HNNs have been extensively investigated, providing a general dynamical background for the development of multi-scroll M-HNNs [18].

Nevertheless, existing methods for constructing multi-scroll attractors in M-HNNs often rely on complex nonlinear functions or parameter-coupling mechanisms, leading to limited controllability of scroll structures and limited ergodic performance. To address these issues, this paper proposes a novel memristive synapse and constructs a structurally simple yet dynamically tunable multi-scroll memristor-based neural network, thereby providing persistently exciting and ergodic nonlinear driving signals for coverage search tasks.

In coverage search and inspection scenarios, multiple mobile robots are typically required to operate within a cooperative framework to improve task efficiency and system robustness. To achieve this, the drive-response architecture has been widely adopted due to its simplicity and scalability. When air-ground cooperative coverage search tasks are further considered, the problem can be equivalently reformulated as a synchronization tracking control problem between an UAV and a chaotic mobile robot. In this formulation, the ground chaotic mobile robot is regarded as the drive system that generates complex and ergodic trajectories for environment exploration, whereas the UAV acts as the response system, synchronizing with and tracking the ground robot to achieve cooperative coverage and information complementarity. From a control perspective, the air-ground cooperative coverage task is therefore cast as an optimal synchronization tracking problem of drive-response chaotic systems.

The rich and random-like chaotic dynamics generated by the multi-scroll M-HNN enable the ground mobile robot to achieve high workspace coverage during environmental exploration. However, chaotic synchronization is

highly sensitive to both initial-condition mismatches and deviations during the synchronization process [19], [20], which makes synchronization between the UAV and the ground mobile robot more challenging. Traditional synchronization methods for chaotic systems, such as adaptive sliding mode control [21] and adaptive active or optimal control [22], have been widely investigated, demonstrating robustness to system uncertainties but often requiring complex design. In addition to these classical approaches, optimization-based strategies have been explored for chaotic synchronization. For example, a particle swarm optimization-based approach has been proposed for optimal synchronization of chaotic systems [23], and a PID controller tuned via particle swarm optimization has also been developed for chaotic synchronization [24]. Although these approaches are theoretically well established, they often involve complex computations or implementation challenges when applied to high-dimensional or strongly nonlinear chaotic systems.

In recent years, reinforcement learning (RL) and its continuous-time extensions have shown significant potential in nonlinear optimal control [25]. These methods approximate solutions to the continuous-time Hamilton-Jacobi-Bellman (HJB) equation through online policy iteration, enabling continuous improvement of control performance [26], [27], [28]. IRL has been applied to adaptive neural network control for continuous-time nonlinear MIMO systems, demonstrating adaptability and suitability for online continuous-time optimization [29]. Moreover, off-policy IRL methods have been developed for continuous-time multi-agent nonzero-sum games, providing a flexible framework for learning optimal policies in multi-agent systems [30]. Building on these developments, this paper proposes a critic-only IRL optimal synchronization control strategy [31] for the UAV to minimize the impact of synchronization errors within a performance optimization framework.

Based on the above motivation, this paper proposes an air-ground collaborative coverage search framework that enhances search coverage and improves air-ground coordination performance. The main contributions of this paper are summarized as follows: (1) A structurally simple and readily extendable memristor model is designed to construct a controllable multi-scroll M-HNN, providing nonlinear driving signals with high dynamical complexity and strong ergodicity for coverage search tasks. (2) To address synchronization tracking error induced by the sensitivity of chaotic systems to initial conditions, an optimal synchronization tracking control strategy based on critic-only IRL is proposed, achieving high-precision drive-response synchronization for mobile robot systems while optimizing synchronization tracking performance online.

The remainder of this paper is organized as follows. Section II introduces the memristor model and constructs the proposed multi-scroll M-HNN, which is then integrated with the kinematic model of mobile robots. Section III presents the design of the synchronization controller, along with a convergence analysis to theoretically validate the proposed control strategy. Section IV provides simulation results to validate the multi-scroll dynamical properties and synchronization tracking performance. Finally, Section V concludes the paper and discusses potential directions for future research.

2. Model Construction

2.1. Memristor Model

Based on the flux-charge relationship of memristors [32], the piecewise memristor model in [15] is adopted to construct the multi-scroll M-HNN, which is expressed as follows:

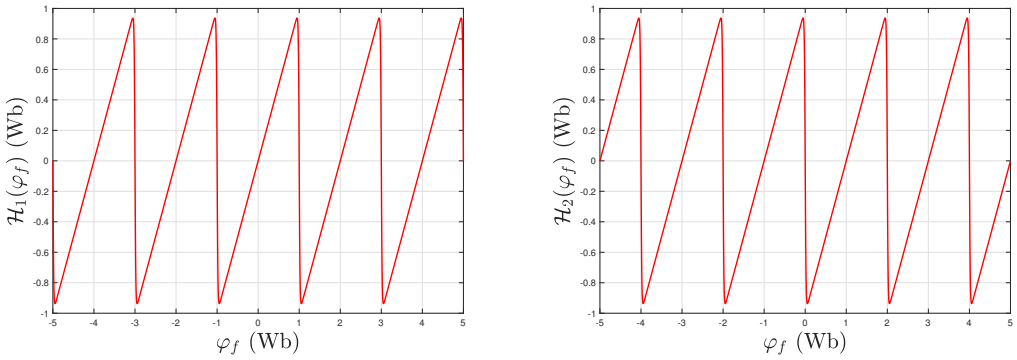
$$\begin{cases} i_m = w(\varphi_f)\nu, \\ w(\varphi_f) = a + b\sin(\varphi_f), \\ \frac{d\varphi_f}{dt} = c_f\nu - d\mathcal{H}(\varphi_f), \end{cases} \tag{1}$$

where a, b, c_f and d are positive parameters, ν, i_m, $\varphi_f (f = 1, 2)$, and $w(\varphi_f)$ denote the memristor's voltage, current, internal magnetic flux, and the flux-controlled memductance, respectively, the core term $\mathcal{H}(\varphi_f)$ is constructed as a sum of hyperbolic tangent functions to regulate the number of scrolls and is specifically designed as

$$\mathcal{H}(\varphi_f) = \begin{cases} \mathcal{H}_1(\varphi_f) = \varphi_f - m\left[\sum_{i=0}^{N}\tanh(n(\varphi_f - (2i+1)m)) + \sum_{i=0}^{N}\tanh(n(\varphi_f + (2i+1)m))\right], \\ \mathcal{H}_2(\varphi_f) = \varphi_f - m\left[\sum_{i=0}^{N}\tanh(n(\varphi_f - 2im)) + \sum_{i=0}^{N}\tanh(n(\varphi_f + 2im)) - \tanh(n\varphi_f)\right], \end{cases} \tag{2}$$

where N is a nonnegative integer and $m > 0$, $n > 0$ are design parameters.

When $N = 1$, $m = 1$, $n = 50$, the graphs of $\mathcal{H}_1(\varphi_f)$ and $\mathcal{H}_2(\varphi_f)$ are shown in Fig. 1.

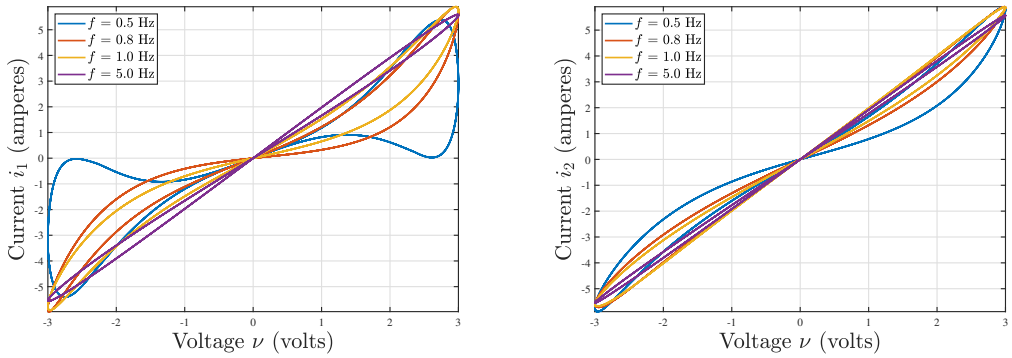


(a) (b)

Fig. 1. Profiles of (a) $\mathcal{H}_1(\varphi_f)$ and (b) $\mathcal{H}_2(\varphi_f)$.

For a given nonnegative integer N, selecting $\mathcal{H}_1(\varphi_f)$ yields a multi-scroll attractor with $4N + 6$ scrolls, whereas selecting $\mathcal{H}_2(\varphi_f)$ yields a multi-scroll attractor with $4N + 4$ scrolls.

To analyze the hysteresis characteristics of the constructed memristors, the parameters are set as $a = 1.01$, $b = 1$, $N = 1$, $m = 1$, and $n = 50$. The dynamic response of the memristor to external stimuli can be characterized by its hysteresis loop, which intuitively reflects the evolution of memductance under sinusoidal voltage excitation with varying frequency. A sinusoidal voltage source $\nu = A\sin(2\pi ft)$ is applied as the excitation signal to control the fingerprints of the memristor model, where the amplitude is fixed at $A = 3$ V, and the excitation frequency is selected as $f = 0.5, 0.8, 1,$ and 5 Hz. As shown in Fig. 2(a), for the sum of hyperbolic tangent functions $\mathcal{H}(\varphi_f) = \mathcal{H}_1(\varphi_f)$, the hysteresis loop continuously contracts with increasing excitation frequency. Similarly, the sum of hyperbolic tangent functions $\mathcal{H}(\varphi_f) = \mathcal{H}_2(\varphi_f)$ in Fig. 2(b) exhibits the same trend.



(a) (b)

Fig. 2. Memristor current-voltage characteristics at different frequencies: (a) $\mathcal{H}(\varphi_f) = \mathcal{H}_1(\varphi_f)$ and (b) $\mathcal{H}(\varphi_f) = \mathcal{H}_2(\varphi_f)$.

2.2. Memristive Hopfield Neural Network

For a multi-scroll M-HNN composed of ζ coupled neurons, the state equation for the i-th neuron is given by

$$C_i \frac{dx_i}{dt} = -\frac{x_i}{R_i} + \sum_{j=1}^{\zeta} \mathcal{W}_{ij} \tanh(x_j) + I_i, \tag{3}$$

where C_i, x_i, and R_i represent the membrane capacitance, membrane voltage, and membrane resistance of the i-th neuron, respectively. The nonlinear function $\tanh(\cdot)$ serves as the neuron activation function, while W_{ij} denotes the synaptic weight from neuron j to neuron i. The external input current, I_i, is typically set to zero.

By replacing the connection weights in the neural network with memristive synapses, a multi-scroll M-HNN is constructed. The multi-scroll M-HNN proposed in this section is illustrated in Fig. 3, and its synaptic weight matrix $\mathcal{W}$ is defined as follows:

$$\mathcal{W} = \begin{bmatrix} \mathcal{W}_{11} & \mathcal{W}_{12} & \mathcal{W}_{13} \\ \mathcal{W}_{21} & \mathcal{W}_{22} & \mathcal{W}_{23} \\ \mathcal{W}_{31} & \mathcal{W}_{32} & \mathcal{W}_{33} \end{bmatrix} = \begin{bmatrix} 2.09 & -1.16 & k_1 w(\varphi_2) \\ 2.1 & 1.5 & k_2 w(\varphi_1) \\ -4.77 & 0.2 & -0.7 \end{bmatrix}, \tag{4}$$

where the memductance function $w(\varphi_f)$, $f = 1, 2$, represents the flux-controlled memductance of the f-th memristive synapse and is measured in siemens (S), with φ_f denoting the corresponding magnetic flux. The coefficients k_1 and k_2 are dimensionless scaling factors that regulate the contribution of the memductive synapses. For numerical simulations, the state variables, time, and model parameters are nondimensionalized, with the same symbols retained for notational simplicity. The resulting dimensionless dynamics are given by

$$\begin{cases} \dot{x}_1 = -x_1 + 2.09 \tanh(x_1) - 1.16 \tanh(x_2) + k_1 w(\varphi_2) \tanh(x_3), \\ \dot{x}_2 = -x_2 + 2.1 \tanh(x_1) + 1.5 \tanh(x_2) + k_2 w(\varphi_1) \tanh(x_3), \\ \dot{x}_3 = -x_3 - 4.77 \tanh(x_1) + 0.2 \tanh(x_2) - 0.7 \tanh(x_3), \\ \dot{\varphi}_1 = 3.3 \tanh(x_2) - 0.9\mathcal{H}(\varphi_1), \\ \dot{\varphi}_2 = 1.5 \tanh(x_1) - 0.9\mathcal{H}(\varphi_2), \end{cases} \tag{5}$$

The equilibrium points of system (5) are determined by setting all time derivatives to zero, i.e., $\dot{x}_1 = \dot{x}_2 = \dot{x}_3 = \dot{\varphi}_1 = \dot{\varphi}_2 = 0$. This procedure transforms the original dynamical system into a set of algebraic equations

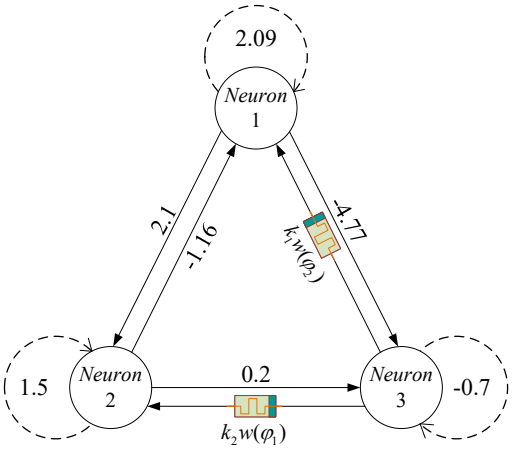


Fig. 3. Network topology of the multi-scroll M-HNN.

with respect to the variables $x_1, x_2, x_3, \varphi_1, \varphi_2$. The resulting equilibrium state equations are expressed as follows:

$$\begin{cases} 0 = -x_1 + 2.09\tanh(x_1) - 1.16\tanh(x_2) + k_1 w(\varphi_2)\tanh(x_3), \\ 0 = -x_2 + 2.1\tanh(x_1) + 1.5\tanh(x_2) + k_2 w(\varphi_1)\tanh(x_3), \\ 0 = -x_3 - 4.77\tanh(x_1) + 0.2\tanh(x_2) - 0.7\tanh(x_3), \\ 0 = 3.3\tanh(x_2) - 0.9\mathcal{H}(\varphi_1), \\ 0 = 1.5\tanh(x_1) - 0.9\mathcal{H}(\varphi_2). \end{cases} \tag{6}$$

Since directly solving equation (6) is computationally challenging, an equivalent reformulation is introduced to determine all equilibrium points of the system [33], [34]. To be specific, equation (6) can be rewritten as

$$\begin{cases} x_1 = \tanh^{-1}\left(\dfrac{0.9}{1.5}\mathcal{H}(\varphi_2)\right), \\ x_2 = \tanh^{-1}\left(\dfrac{0.9}{3.3}\mathcal{H}(\varphi_1)\right), \\ x_3 = \tanh^{-1}\left(\dfrac{1}{k_1 w(\varphi_2)}(x_1 - 2.09\tanh(x_1) + 1.16\tanh(x_2))\right), \\ l_1 = -x_2 + 2.1\tanh(x_1) + 1.5\tanh(x_2) + k_2 w(\varphi_1)\tanh(x_3), \\ l_2 = -x_3 - 4.77\tanh(x_1) + 0.2\tanh(x_2) - 0.7\tanh(x_3). \end{cases} \tag{7}$$

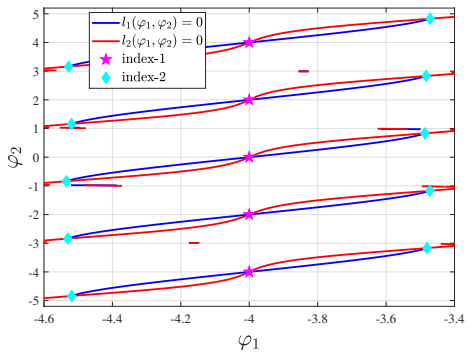


Fig. 4. l_1, l_2 and their intersection points.

Table 1. Equilibrium points, eigenvalues, and equilibrium types.

Equilibrium Points	Eigenvalues	Equilibrium Types
$(0.0000, 0.0000, 0.0000, -4.0000, -4.0000)$	$0.8843,\ -0.4972 \pm 1.5391i,\ -0.9000,\ -0.9000$	index-1
$(0.0000, 0.0000, 0.0000, -4.0000, -2.0000)$	$0.8863,\ -0.4981 \pm 1.5309i,\ -0.9000,\ -0.9000$	index-1
$(0.0000, 0.0000, 0.0000, -4.0000, 0.0000)$	$0.8852,\ -0.4976 \pm 1.5354i,\ -0.9000,\ -0.9000$	index-1
$(0.0000, 0.0000, 0.0000, -4.0000, 2.0000)$	$0.8842,\ -0.4971 \pm 1.5398i,\ -0.9000,\ -0.9000$	index-1
$(0.0000, 0.0000, 0.0000, -4.0000, 4.0000)$	$0.8861,\ -0.4980 \pm 1.5317i,\ -0.9000,\ -0.9000$	index-1
$(-0.5550, -0.1424, 1.7204, -4.5186, -4.8404)$	$0.4103 \pm 1.2686i,\ -0.8694,\ -0.9010,\ -0.9058$	index-2
$(0.5482, 0.1427, -1.6982, -3.4803, -3.1680)$	$0.4207 \pm 1.2618i,\ -0.8929 \pm 0.0394i,\ -0.9044$	index-2
$(-0.5535, -0.1456, 1.7151, -4.5302, -2.8386)$	$0.4093 \pm 1.2687i,\ -0.8615,\ -0.8978,\ -0.9148$	index-2
$(0.5471, 0.1450, -1.6940, -3.4720, -1.1694)$	$0.4195 \pm 1.2623i,\ -0.8951 \pm 0.0317i,\ -0.8973$	index-2
$(-0.5530, -0.1467, 1.7132, -4.5342, -0.8380)$	$0.4105 \pm 1.2671i,\ -0.8743,\ -0.9011 \pm 0.0057i$	index-2
$(0.5491, 0.1409, -1.7013, -3.4866, 0.8331)$	$0.4195 \pm 1.2634i,\ -0.8947,\ -0.8966 \pm 0.0297i$	index-2
$(-0.5549, -0.1426, 1.7201, -4.5193, 1.1597)$	$0.4105 \pm 1.2683i,\ -0.8717,\ -0.9024 \pm 0.0037i$	index-2
$(0.5486, 0.1420, -1.6994, -3.4828, 2.8324)$	$0.4207 \pm 1.2619i,\ -0.8930 \pm 0.0391i,\ -0.9042$	index-2
$(-0.5538, -0.1449, 1.7162, -4.5277, 3.1610)$	$0.4093 \pm 1.2689i,\ -0.8612,\ -0.8977,\ -0.9152$	index-2
$(0.5470, 0.1452, -1.6937, -3.4714, 4.8305)$	$0.4197 \pm 1.2621i,\ -0.8944 \pm 0.0334i,\ -0.8993$	index-2

To identify the intersections, the implicit functions l_1 and l_2 are represented by the blue and red curves, respectively, as shown in Fig. 4. It can be clearly observed that there are 15 intersection points in Fig. 4, corresponding to 15 equilibrium points. The Jacobian matrix of system (5) is given by

$$J = \begin{bmatrix} -1 + 2.09h_1 & -1.16h_2 & k_1 w_2 h_3 & 0 & k_1 w_2' \tanh(x_3) \\ 2.1h_1 & -1 + 1.5h_2 & k_2 w_1 h_3 & k_2 w_1' \tanh(x_3) & 0 \\ -4.77h_1 & 0.2h_2 & -1 - 0.7h_3 & 0 & 0 \\ 0 & 3.3h_2 & 0 & -0.9\mathcal{H}'(\varphi_1) & 0 \\ 1.5h_1 & 0 & 0 & 0 & -0.9\mathcal{H}'(\varphi_2) \end{bmatrix}, \tag{8}$$

where $h_1 = \operatorname{sech}^2(x_1)$, $h_2 = \operatorname{sech}^2(x_2)$, $h_3 = \operatorname{sech}^2(x_3)$, $w_f = w(\varphi_f)(f = 1, 2)$, $w_f' = b\cos(\varphi_f)$, and $\mathcal{H}'(\varphi_f) = d\mathcal{H}(\varphi_f)/d\varphi_f$.

By evaluating the Jacobian matrix at each equilibrium point, the corresponding eigenvalues can be obtained to characterize its local stability and equilibrium type. To be specific, the 15 equilibrium points shown in Fig. 4 are substituted into the Jacobian matrix (8), and the resulting eigenvalues are listed in Table 1. Based on the signs of the real parts of these eigenvalues and the distribution of complex-conjugate eigenvalues, 10 of the 15 equilibrium points are identified as index-2 saddle-focus equilibrium points.

The multi-scroll M-HNN system exhibits multi-scroll chaotic behavior under suitable parameter settings. For example, when $k_1 = 0.35$, $k_2 = 1.18$, $m = 1$ (using $\mathcal{H}_1(\varphi_2)$ and $N = 1$) and initial conditions $(0.1, 0, 0, 0, 0)$, numerical simulation results show that the system can generate a 10-scroll chaotic attractor as shown in Fig. 5.

From a dynamical perspective, the constructed hyperbolic-tangent summation terms introduce multiple nonlinear transition regions within different intervals of the magnetic-flux state φ_2, giving rise to multiple index-2 saddle-focus equilibrium points. The system trajectories exhibit local spiral motion around these equilibrium points, with each index-2 saddle-focus equilibrium points corresponding to one scroll of the attractor. Further investigation reveals that the number of scrolls in the attractor is primarily determined by the activation function $\mathcal{H}(\varphi_2)$ and the parameters N and m. With other parameters fixed (e.g., $n = 50$) and under the same initial

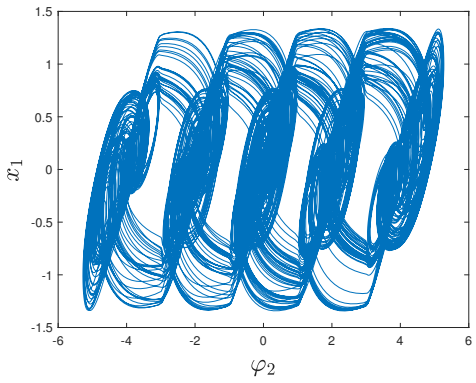


Fig. 5. Attractor with 10-scroll generated by $\mathcal{H}_1(\varphi_2)$ with $N = 1$.

conditions, the following rules can be summarized:

When $m = 1$: If the activation function is $\mathcal{H}_1(\varphi_2)$, the system generates a $4N + 6$ scroll chaotic attractor; if the activation function is $\mathcal{H}_2(\varphi_2)$, the system generates a $4N + 4$ scroll chaotic attractor.

When $m = 0$: The system invariably produces only a 2-scroll chaotic attractor, regardless of the form of $\mathcal{H}(\varphi_2)$.

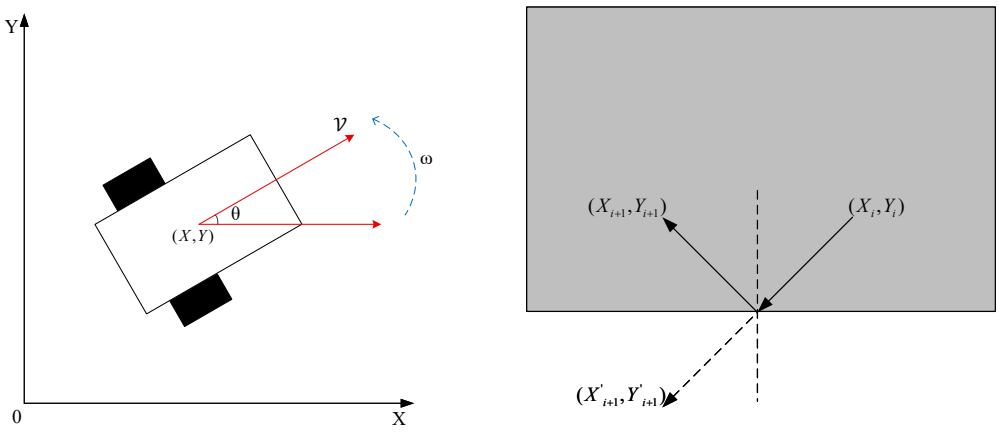
A detailed analysis of the system's dynamics, including phase portraits, bifurcation diagrams, Lyapunov exponent spectra, and Poincaré sections, is presented in Section 4.

2.3. Chaotic Mobile Robot System Construction

Our prior work has revealed the existence of multi-scroll chaotic attractors within the proposed multi-scroll M-HNN model. Building upon this result, this section further incorporates the chaotic dynamics to drive the ground mobile robot for air-ground collaborative coverage search. Chaotic systems exhibit extreme sensitivity to initial conditions, a property that generates complex and non-repeatable dynamic behavior. When this chaotic dynamic characteristic is incorporated into robotic controllers, the robot can produce highly irregular and unpredictable motion trajectories. The paths produced by this method exhibit excellent ergodicity and strong randomness, enabling efficient area coverage. The structural schematic of the two-wheeled differential-drive robot is shown in Fig. 6(a). Precise motion is achieved by controlling the speeds of its two drive wheels. The robot's pose is defined by its planar coordinates (X, Y) and orientation θ, and its kinematic model is presented as follows:

$$\begin{pmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathcal{V} \\ \omega \end{pmatrix}, \tag{9}$$

where (X, Y) represent the robot's position in the Cartesian coordinate system, while $\mathcal{V}$ and ω denote its linear and angular velocities, respectively. By coupling the dynamics of the multi-scroll M-HNN with the kinematic model of the mobile robot, a chaotic mobile robot system is constructed. The complete system dynamics are expressed as follows:



(a) (b)

Fig. 6. Mobile path planning diagram: (a) schematic of mobile robot's motion on a plane, (b) presents the diagram of the mirror reflection principle.

$$\begin{cases} \dot{x}_1 = -x_1 + 2.09\tanh(x_1) - 1.16\tanh(x_2) + 0.35w(\varphi_2)\tanh(x_3), \\ \dot{x}_2 = -x_2 + 2.1\tanh(x_1) + 1.5\tanh(x_2) + 1.18w(\varphi_1)\tanh(x_3), \\ \dot{x}_3 = -x_3 - 4.77\tanh(x_1) + 0.2\tanh(x_2) - 0.7\tanh(x_3), \\ \dot{\varphi}_1 = 3.3\tanh(x_2) - 0.9\mathcal{H}(\varphi_1), \\ \dot{\varphi}_2 = 1.5\tanh(x_1) - 0.9\mathcal{H}(\varphi_2), \\ \dot{X} = \mathcal{V}\cos(\theta), \\ \dot{Y} = \mathcal{V}\sin(\theta), \\ \dot{\theta} = \omega. \end{cases} \tag{10}$$

where $\omega = 0.05\varphi_2$. The magnetic flux φ_2 is selected as the driving signal for the ground robot to perform coverage search and to provide synchronization signals for the UAV. A key requirement in the robot operation is that its trajectory must remain within the predefined workspace at all times. To enforce this boundary constraint, a mirror mapping method is adopted. The basic idea is to automatically adjust the trajectory of the robot whenever the next position predicted by the dynamic model of the robot falls outside the prescribed workspace. To be specific, should the robot's trajectory cross the left or right boundaries (i.e., $X < X_{\min}$ or $X > X_{\max}$), its horizontal coordinate is mirrored with respect to the corresponding boundary line. The position is updated as follows:

$$X = \begin{cases} 2X_{\min} - X, & \text{if } X < X_{\min}, \\ 2X_{\max} - X, & \text{if } X > X_{\max}. \end{cases} \tag{11}$$

This reflection is symmetric about a vertical axis, which transforms the heading angle θ according to:

$$\theta = \pi - \theta. \tag{12}$$

Notably, the alternative expression $\theta = -\pi - \theta$ is mathematically equivalent after normalization to the interval $[-\pi, \pi]$. Similarly, if the trajectory attempts to cross the top or bottom boundaries (i.e., $Y < Y_{\min}$ or

$Y > Y_{\max}$), the vertical coordinate is mirrored:

$$Y = \begin{cases} 2Y_{\min} - Y, & \text{if } Y < Y_{\min}, \\ 2Y_{\max} - Y, & \text{if } Y > Y_{\max}. \end{cases} \tag{13}$$

The corresponding reflection about a horizontal axis updates the heading angle as:

$$\theta = -\theta. \tag{14}$$

This approach not only provides a simple boundary-handling mechanism but also enhances operational efficiency by maintaining continuous motion.

To evaluate the effectiveness of the proposed strategy, the coverage rate (C) is used as an efficiency metric, representing the surface area covered by the robot during its operation [35]. It is expressed as:

$$C = \frac{1}{\mathcal{M}} \sum_{i=1}^{\mathcal{M}} I(i), \tag{15}$$

where C denotes the coverage rate, $\mathcal{M}$ denotes the total number of cells, while $I(i)$ is an indicator function that determines whether the i-th cell in the discretized workspace has been covered. If the robot trajectory enters cell i at least once, then $I(i) = 1$; otherwise, $I(i) = 0$. The coverage rate is therefore defined as the ratio of the number of covered cells to the total number of cells, serving as a key metric for efficiency.

3. Synchronization Tracking Controller Design

When further considering the air-ground collaborative search task, this problem can be transformed into the synchronization tracking control of a chaotic mobile robot by a UAV. To ensure consistent cooperative behavior between the drive system and the response system under complex nonlinear dynamics, a performance index incorporating the synchronization error and control effort is introduced to formulate the synchronization tracking task as an optimal control problem. To clearly illustrate the complete implementation procedure of the proposed air-ground collaborative coverage framework, from multi-scroll chaotic signal generation and ground-robot motion to UAV synchronization control and coverage evaluation, Fig. 7 presents the overall control flow of the proposed framework.

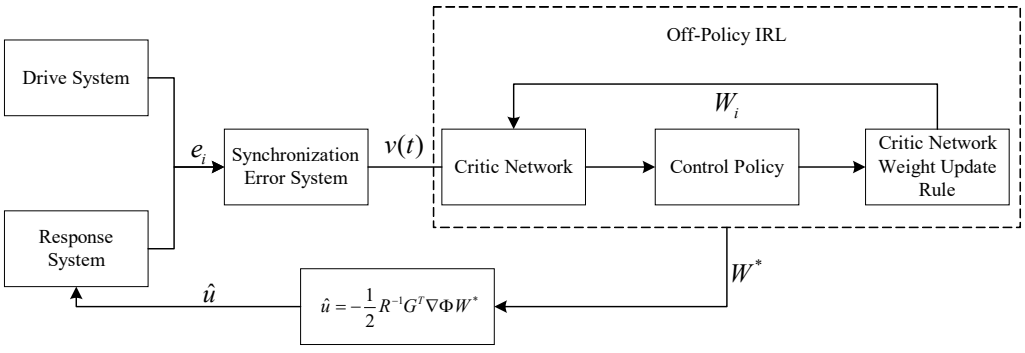


Fig. 7. Overall Flowchart.

3.1. Synchronization Tracking Scheme

To transform the collaborative synchronization requirement into a tractable control problem, we first establish the drive-response synchronization scheme. The dynamic model (10) of the chaotic mobile robot is selected as the drive system, and the response system is constructed by augmenting control inputs as follows:

$$\begin{cases} \dot{y}_1 = -y_1 + 2.09\tanh(y_1) - 1.16\tanh(y_2) + 0.35w(\varphi_{s2})\tanh(y_3) + u_1, \\ \dot{y}_2 = -y_2 + 2.1\tanh(y_1) + 1.5\tanh(y_2) + 1.18w(\varphi_{s1})\tanh(y_3) + u_2, \\ \dot{y}_3 = -y_3 - 4.77\tanh(y_1) + 0.2\tanh(y_2) - 0.7\tanh(y_3) + u_3, \\ \dot{\varphi}_{s1} = 3.3\tanh(y_2) - 0.9\mathcal{H}(\varphi_{s1}) + u_4, \\ \dot{\varphi}_{s2} = 1.5\tanh(y_1) - 0.9\mathcal{H}(\varphi_{s2}) + u_5, \\ \dot{X}_s = \mathcal{V}\cos(\theta_s) + u_6, \\ \dot{Y}_s = \mathcal{V}\sin(\theta_s) + u_7, \\ \dot{\theta}_s = \omega_s + u_8, \end{cases} \tag{16}$$

where $\omega_s = 0.05\varphi_{s2}$. The variables X_s, Y_s, and θ_s denote the position coordinates and heading angle of the response system, respectively. The subscript s is used throughout this section to distinguish the response-system states from the corresponding drive-system states. Define the synchronization tracking error as $e_i = y_i - x_i$ $(i = 1, 2, 3)$, $e_4 = \varphi_{s1} - \varphi_1$, $e_5 = \varphi_{s2} - \varphi_2$, $e_6 = X_s - X$, $e_7 = Y_s - Y$, $e_8 = \theta_s - \theta$.

The resulting dynamic behavior of the synchronization tracking error system is as follows:

$$\begin{cases} \dot{e}_1 = -e_1 + 2.09\tanh(x_1 + e_1) - 1.16\tanh(x_2 + e_2) + 0.35w(\varphi_2 + e_5)\tanh(x_3 + e_3) \\ \qquad - 2.09\tanh(x_1) + 1.16\tanh(x_2) - 0.35w(\varphi_2)\tanh(x_3) + u_1, \\ \dot{e}_2 = -e_2 + 2.1\tanh(x_1 + e_1) + 1.5\tanh(x_2 + e_2) + 1.18w(\varphi_1 + e_4)\tanh(x_3 + e_3) \\ \qquad - 2.1\tanh(x_1) - 1.5\tanh(x_2) - 1.18w(\varphi_1)\tanh(x_3) + u_2, \\ \dot{e}_3 = -e_3 - 4.77\tanh(x_1 + e_1) + 0.2\tanh(x_2 + e_2) - 0.7\tanh(x_3 + e_3) \\ \qquad + 4.77\tanh(x_1) - 0.2\tanh(x_2) + 0.7\tanh(x_3) + u_3, \\ \dot{e}_4 = 3.3\tanh(x_2 + e_2) - 0.9\mathcal{H}(\varphi_1 + e_4) - 3.3\tanh(x_2) + 0.9\mathcal{H}(\varphi_1) + u_4, \\ \dot{e}_5 = 1.5\tanh(x_1 + e_1) - 0.9\mathcal{H}(\varphi_2 + e_5) - 1.5\tanh(x_1) + 0.9\mathcal{H}(\varphi_2) + u_5, \\ \dot{e}_6 = \mathcal{V}\cos(\theta + e_8) - \mathcal{V}\cos(\theta) + u_6, \\ \dot{e}_7 = \mathcal{V}\sin(\theta + e_8) - \mathcal{V}\sin(\theta) + u_7, \\ \dot{e}_8 = 0.05e_5 + u_8. \end{cases} \tag{17}$$

For convenience, we define the synchronization tracking error vector $v = [e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8]^T$, the control input $u = [u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8]^T$. Here, u_6 and u_7 denote the auxiliary velocity compensation commands along the inertial X- and Y-directions for the UAV response system, respectively. In practical implementation, they can be mapped to the desired horizontal velocity commands of the UAV autopilot. Moreover, u_8 represents the yaw-rate compensation command. Thus, the dynamic characteristics of the synchronization tracking error system (17) can be uniformly described in the following form:

$$\dot{v}(t) = F(v(t)) + Gu(t), \tag{18}$$

where the nonlinear functions $F(v(t))$ and $G(v(t))$ are expressed as

$$F(v(t)) = \begin{bmatrix} -e_1 + 2.09\tanh(x_1+e_1) - 1.16\tanh(x_2+e_2) + 0.35w(\varphi_2+e_5)\tanh(x_3 \\ \qquad + e_3) - 2.09\tanh(x_1) + 1.16\tanh(x_2) - 0.35w(\varphi_2)\tanh(x_3) \\ -e_2 + 2.1\tanh(x_1+e_1) + 1.5\tanh(x_2+e_2) + 1.18w(\varphi_1+e_4)\tanh(x_3 \\ \qquad + e_3) - 2.1\tanh(x_1) - 1.5\tanh(x_2) - 1.18w(\varphi_1)\tanh(x_3) \\ -e_3 - 4.77\tanh(x_1+e_1) + 0.2\tanh(x_2+e_2) - 0.7\tanh(x_3+e_3) \\ \qquad + 4.77\tanh(x_1) - 0.2\tanh(x_2) + 0.7\tanh(x_3) \\ 3.3\tanh(x_2+e_2) - 0.9\mathcal{H}(\varphi_1+e_4) - 3.3\tanh(x_2) + 0.9\mathcal{H}(\varphi_1) \\ 1.5\tanh(x_1+e_1) - 0.9\mathcal{H}(\varphi_2+e_5) - 1.5\tanh(x_1) + 0.9\mathcal{H}(\varphi_2) \\ \mathcal{V}\cos(\theta+e_8) - \mathcal{V}\cos(\theta) \\ \mathcal{V}\sin(\theta+e_8) - \mathcal{V}\sin(\theta) \\ 0.05e_5 \end{bmatrix},$$

$$G = I_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Base on the synchronization error system (18), consider the following value function:

$$V(t) = \int_t^\infty e^{-\alpha(\tau-t)} r(v(\tau), u(\tau)) d\tau, \tag{19}$$

where $\alpha > 0$ is a discount factor, and the exponential term $e^{-\alpha(\tau-t)}$ serves as a decay factor that attenuates future costs and promotes convergence of the value function. $r(v(\tau), u(\tau)) = v(\tau)^\top Q v(\tau) + u(\tau)^\top R u(\tau)$ is the running cost. This function incorporates the costs on the state and the control input. The matrices Q and R are positive definite. Define the HJB equation as follows

$$H(u, \nabla V) = v^T Q v + u^T R u - \alpha V + \nabla V^T (F + Gu) = 0, \tag{20}$$

where $\nabla V = \partial V / \partial v$. Applying the stationarity condition $\partial H(u, \nabla V)/\partial u = 0$ gives the optimal control input as

$$u^*(v(t)) = -\frac{1}{2} R^{-1} G^T \nabla V^*. \tag{21}$$

By substituting the optimal solution for the control u^*, given by (21), back into the HJB equation (20):

$$v^T Q v - \frac{1}{4} (\nabla V^*)^T G R^{-1} G^T \nabla V^* - \alpha V^* + (\nabla V^*)^T F = 0. \tag{22}$$

It is evident from equation (21) that the determination of the optimal control policy u^* depends on the optimal value function V^*. This indicates that the core of this problem reduces to solving the HJB equations given by (22). However, (22) is a highly nonlinear partial differential equation (PDE), for which analytical solutions are generally intractable. To address this challenge, this paper employs an IRL approach to approximately solve the HJB equation, thereby circumventing the difficulty of direct analytical derivation.

Algorithm 1 Model-Based RL Algorithm

1: **Initialization:** Start with an initial admissible stabilizing control policy $u_0(x)$.

2: **for** $i = 0, 1, 2, \ldots$ until convergence **do**

3: **1) Policy Evaluation:** For a given control policy u_i, find the value function V_i by solving the following Bellman equation:

$$v^T Q v + u_i^T R u_i - \alpha V_i + \nabla V_i^T (F + G u_i) = 0. \tag{23}$$

4: **2) Policy Improvement:** Update the control policy using the obtained value function V_i:

$$u_{i+1}(v(t)) = -\frac{1}{2} R^{-1} G^T \nabla V_i. \tag{24}$$

5: **end for**

3.2. Off-Policy IRL for Synchronization Tracking

To obtain the optimal control policy given in (21), it is necessary to solve the HJB equations presented in (22). However, this set of equations consists of nonlinear PDEs, for which an analytical solution is generally intractable. Policy iteration (PI) is among the most widely used methods to overcome this difficulty [36], [37]. We adopt policy iteration as the underlying framework and then derive an off-policy IRL implementation that allows the optimal policy to be learned from data without explicitly solving the HJB PDE. The PI method approximates the solution of the HJB equations through an iterative procedure, such as Algorithm 1, and its convergence is theoretically guaranteed. That is, as the number of iterations approaches infinity ($i \to \infty$), the value function $V_i(v)$ and the control policy $u_i(v)$ converge to the optimal solutions $V_i^*(v)$ and $u_i^*(v)$, respectively. This convergence is established because the iteration process is essentially a Newton's iteration sequence that converges to the unique solution of the HJB equation (22).

A major limitation of conventional PI is that it typically requires explicit knowledge and evaluation of system dynamics within each policy evaluation step. To relax this requirement, this study employs an off-policy IRL framework, which enables the policy to be updated directly from system data without explicitly computing the system equations in closed form. In this sense, the synchronization controller is obtained while still operating within the established system dynamics. As a first step, the dynamics in equation (18) are rewritten as follows:

$$\dot{v} = F + G u_i + G(u - u_i), \tag{25}$$

where the policy $u_i = [u_{i,1}, \ldots, u_{i,8}]^T \in \mathbb{R}^8$. Differentiating $V_i(v)$ along the system dynamics (25) and using the optimal control from (24) gives

$$\begin{aligned} \frac{dV_i(v)}{dt} &= \nabla V_i^T (F + G u_i) + \nabla V_i^T G (u - u_i) \\ &= \alpha V_i - v^T Q v - u_i^T R u_i - 2 u_{i+1}^T R (u - u_i). \end{aligned} \tag{26}$$

Multiplying both sides of (26) by $e^{-\alpha(\tau-t)}$ and subsequently integrating with respect to τ yields the following off-policy IRL Bellman equation:

$$\begin{aligned} e^{-\alpha T}V_i(v(t+T)) - V_i(v(t)) = &\int_t^{t+T} e^{-\alpha(\tau-t)}(-v^TQv - u_i^TRu_i)d\tau \\ &+ \int_t^{t+T} e^{-\alpha(\tau-t)}(-2u_{i+1}^TR(u-u_i))d\tau. \end{aligned} \tag{27}$$

It can be seen from equation (27) that the solution can be obtained without explicitly using the system dynamics $F(v)$. Therefore, we replace equation (23) in Algorithm 1 with equation (27). To justify this substitution, we will subsequently prove the equivalence between equation (23) and equation (27).

Theorem 1 Let $V_i(v) \in C^1(\Omega)$, where $C^1(\Omega)$ denotes a function space on Ω with continuous first-order derivatives. Assume $V_i(v) \geq 0$, $V_i(0) = 0$. Then $V_i(v)$ is the solution of (27) if and only if it is the solution of (23).

Proof. Dividing both sides of the off-policy IRL Bellman equation (27) by T, and taking the limit as $T \to 0$, yields

$$\begin{aligned} &\lim_{T\to 0} \frac{e^{-\alpha T}V_i(v(t+T)) - V_i(v(t))}{T} \\ &+ \lim_{T\to 0} \int_t^{t+T} \frac{e^{-\alpha(\tau-t)}(v^TQv + u_i^TRu_i)}{T} d\tau \\ &+ \lim_{T\to 0} \int_t^{t+T} \frac{e^{-\alpha(\tau-t)}2u_{i+1}^TR(u-u_i)}{T} d\tau \\ &= 0. \end{aligned} \tag{28}$$

Applying L'Hopital's rule to the first, second, and third terms of (28) respectively, we get

$$\begin{aligned} \lim_{T\to 0} \frac{e^{-\alpha T}V_i(v(t+T)) - V_i(v(t))}{T} &= \lim_{T\to 0}(-\alpha e^{-\alpha T}V_i(v(t+T)) + e^{-\alpha T}\dot{V}_i(v(t+T))) \\ &= -\alpha V_i + \nabla V_i^T(F + Gu_i + G(u-u_i)), \end{aligned} \tag{29}$$

$$\begin{aligned} \lim_{T\to 0} \int_t^{t+T} \frac{e^{-\alpha(\tau-t)}(v^TQv + u_i^TRu_i)}{T} d\tau &= \lim_{T\to 0} e^{-\alpha T}(v^TQv + u_i^TRu_i) \\ &= v^TQv + u_i^TRu_i, \end{aligned} \tag{30}$$

$$\begin{aligned} \lim_{T\to 0} \int_t^{t+T} \frac{e^{-\alpha(\tau-t)}2u_{i+1}^TR(u-u_i)}{T} d\tau &= \lim_{T\to 0} e^{-\alpha T}2u_{i+1}^TR(u-u_i) \\ &= 2u_{i+1}^TR(u-u_i). \end{aligned} \tag{31}$$

Substituting (29)-(31) in (28) yields

$$-\alpha V_i + \nabla V_i^T(F + Gu_i + G(u-u_i)) + v^TQv + u_i^TRu_i + 2u_{i+1}^TR(u-u_i) = 0. \tag{32}$$

The Bellman equation (23) is obtained by substituting the updated policy u_{i+1} from equation (24) into equation (32). The proof is thus complete.

To approximate the unknown function $V_i(v)$ in Equation (27) using system data, we employ a critic-only neural network structure. According to the policy improvement equation (24), the control policy is directly

determined by the gradient of the value function. Therefore, approximating $V_i(v)$ alone is sufficient to obtain the control policy, without requiring an additional actor network. The feasibility of this method is guaranteed by the universal approximation theorem. Accordingly, the output of the critic neural network is given by

$$\hat{V}_i(v) = W_i^T \Phi(v), \qquad i = 0, 1, 2, \dots \tag{33}$$

where $\Phi(v) = [\Phi_1, \dots, \Phi_l]^T \in \mathbb{R}^l$ and $W_i \in \mathbb{R}^l$ is the weight vector of the critic neural network. From (24) and (33), the control policies are given by

$$\hat{u}_i(v) = -\frac{1}{2} R^{-1} G^T \nabla \Phi^T(v) W_i, \qquad i = 0, 1, 2, \dots \tag{34}$$

where $\nabla \Phi(v) = [\partial \Phi_1 / \partial v \dots \partial \Phi_l / \partial v]^T$ is the Jacobian matrix of $\Phi(v)$. Then, substituting (33)-(34) in (27) yields the integral Bellman error (IBE)

$$\begin{aligned}
\varepsilon_i(t) &= e^{-\alpha T} \hat{V}_i(v(t+T)) - \hat{V}_i(v(t)) - \int_t^{t+T} e^{-\alpha(\tau - t)} (-v^T Q v - u_i^T R u_i) d\tau \\
&- \int_t^{t+T} e^{-\alpha(\tau - t)} (-2 u_{i+1}^T R (u - u_i)) d\tau \\
&= W_i^T (e^{-\alpha T} \Phi(v(t+T)) - \Phi(v(t))) + \int_t^{t+T} e^{-\alpha(\tau - t)} v^T Q v d\tau \\
&+ \frac{1}{4} \int_t^{t+T} e^{-\alpha(\tau - t)} W_i^T \nabla \Phi G R^{-1} G^T \nabla \Phi^T W_i d\tau - \int_t^{t+T} e^{-\alpha(\tau - t)} W_{i+1}^T \nabla \Phi G u d\tau \\
&- \frac{1}{2} \int_t^{t+T} e^{-\alpha(\tau - t)} W_{i+1}^T \nabla \Phi G R^{-1} G^T \nabla \Phi^T W_i d\tau.
\end{aligned} \tag{35}$$

For notation simplicity, define

$$\begin{aligned}
h_{\nabla \Phi}(v(t)) &= e^{-\alpha T} \Phi(v(t+T)) - \Phi(v(t)), \\
h_{G\Phi}(v(t)) &= \int_t^{t+T} e^{-\alpha(\tau - t)} \nabla \Phi G R^{-1} G^T \nabla \Phi^T d\tau, \\
h_{Gu}(v(t)) &= \int_t^{t+T} e^{-\alpha(\tau - t)} \nabla \Phi G u d\tau, \\
\eta(v(t)) &= \int_t^{t+T} e^{-\alpha(\tau - t)} v^T Q v d\tau.
\end{aligned}$$

Then, expression (35) can be rewritten as

$$\begin{aligned}
\varepsilon_i(t) &= W_i^T h_{\nabla \Phi}(v(t)) + \frac{1}{4} W_i^T h_{G\Phi}(v(t)) W_i - \frac{1}{2} W_{i+1}^T h_{G\Phi}(v(t)) W_i - W_{i+1}^T h_{Gu}(v(t)) + \eta(v(t)) \\
&= W_{i+1}^T h(v(t)) - z_i(t),
\end{aligned} \tag{36}$$

where

$$h(v(t)) = -\frac{1}{2} h_{G\Phi}(v(t)) W_i - h_{Gu}(v(t)), \tag{37}$$

$$z_i(t) = -W_i^T h_{\nabla \Phi}(v(t)) - \frac{1}{4} W_i^T h_{G\Phi}(v(t)) W_i - \eta(v(t)). \tag{38}$$

To determine the weight vector W_{i+1} for the critic network at iteration $i+1$, we seek to minimize the Bellman error from (36) in a least-squares sense. This is achieved using system data collected at $\mathcal{N}$ distinct time instants $\{t_k\}_{k=1}^{\mathcal{N}}$, where $\mathcal{N} \geq l$ (l being the number of elements in W).

To be specific, for each sample point t_k, the Bellman error equation (36) can be rearranged into a linear regression form with respect to the unknown weight vector W_{i+1}:

$$W_{i+1}^T h(t_k) \approx z_i(t_k), \tag{39}$$

where $h(t_k)$ and $z_i(t_k)$ are defined in (37) and (38), respectively, and can be computed from the collected data and the current weight vector W_i.

To compute W_{i+1}, the equations corresponding to all $\mathcal{N}$ samples are stacked into a single matrix equation. Accordingly, the regressor matrix H and the target vector Z_i are defined as follows:

$$H = [h(t_1), h(t_2), \ldots, h(t_{\mathcal{N}})], \tag{40}$$

$$Z_i = [z_i(t_1), z_i(t_2), \ldots, z_i(t_{\mathcal{N}})]. \tag{41}$$

Using these definitions, the system of $\mathcal{N}$ regression equations can be compactly written as:

$$W_{i+1}^T H \approx Z_i. \tag{42}$$

The problem of determining the optimal W_{i+1} is thereby cast as a standard least-squares optimization, which aims to minimize the sum of squared residuals $\|W_{i+1}^T H - Z_i\|_F^2$. The corresponding solution is obtained from the classical normal equation, provided that the matrix HH^T is invertible:

$$W_{i+1} = (HH^T)^{-1} H Z_i^T. \tag{43}$$

Based on the weight update law (43), an offline iterative learning procedure is outlined in Algorithm 2. The algorithm first executes the data measurement phase (Step 1) to collect and preprocess system operation data. It then proceeds to the offline learning phase (Steps 2-4), where Step 2 initializes the initial weight vector and iteration index. Steps 3 and 4 form an iterative loop that repeatedly updates the weights until convergence criteria are met, thereby approximating the ideal weight vector. □

Algorithm 2 Off-Policy RL for Optimal Synchronization Tracking Control

1: Collect the system data (v, u), and then compute $h_{\nabla\Phi}(v)$, $h_{G\Phi}(v)$, $h_{Gu}(v)$, $\eta(v)$;
2: Select initial critic neural network weight vector W_0, let the iteration index $i = 0$;
3: Compute H and Z_i, and update W_{i+1} according to (43);
4: **if** $\|W_{i+1} - W_i\|^2 \leq \epsilon$ (where ϵ is a small positive number) **then**
5: Stop the iteration and use W_{i+1} to obtain the control policy (34);
6: **else**
7: Set $i = i + 1$ and return to Step 3.
8: **end if**

Remark 1. *In learning algorithms based on policy or value function updates, the PE condition must be satisfied to ensure the invertibility of the relevant matrix HH^T. Specifically, the PE condition requires that there exist positive constants ρ and T_0 such that*

$$\int_t^{t+T_0} H(\tau) H^\top(\tau)\, d\tau \geq \rho I, \quad \forall t \geq 0, \tag{44}$$

which ensures that the information matrix is positive definite. In practical applications, this condition is typically satisfied by injecting random noise or sinusoidal signals of varying frequencies into the control input, or by increasing the sample size to enhance data richness.

Algorithm 3 Overall procedure of the proposed air-ground collaborative coverage framework

Require: Parameters of the multi-scroll M-HNN, ground robot and UAV parameters, task-region information, IRL design parameters, and initial system states.

Ensure: Converged critic weights W^* and synchronization control input.

1: Initialize the system states and control parameters of the ground robot and UAV;
2: Construct the sum of hyperbolic tangent functions $\mathcal{H}(\varphi_2)$ according to (2);
3: Construct the multi-scroll M-HNN according to (5), and generate the chaotic driving signal φ_2;
4: Drive the ground robot using φ_2 to obtain the chaotic reference trajectory;
5: Construct the synchronization error vector v according to (17);
6: Perform offline IRL training according to Algorithm 2 to obtain the converged weights W^*;
7: **while** the coverage task is not completed **do**
8: Calculate the synchronization control policy

$$\hat{u}(v) = -\frac{1}{2}R^{-1}G^T\nabla\Phi^T(v)W^*; \tag{45}$$

9: Apply the control policy $\hat{u}$ to the UAV response system and update the system state;
10: Update the synchronization error vector v;
11: If the system state leaves the prescribed region, apply the boundary-reflection rule according to (11)-(14);
12: **end while**
13: Evaluate the coverage rate according to (15).

The implementation process of the proposed air-ground collaborative coverage framework is illustrated in Algorithm 3. The procedure begins with the construction of a multi-scroll M-HNN to generate driving signals for the ground robot. Subsequently, a critic-only IRL synchronization tracking control strategy is developed to synchronize the planar projection of the UAV motion with the motion of the ground mobile robot. Through the coordinated design of chaotic driving signals and the synchronization tracking control strategy, the proposed framework enhances both coverage search capability and air-ground coordination performance.

3.3. Convergence Analysis

For the proposed neural-network-based off-policy RL algorithm, this section analyzes the convergence of the critic-network weights and proves the uniform ultimate boundedness (UUB) of the synchronization error.

Lemma 1. *[38] For $i = 0, 1, 2, \ldots$, suppose that V_{i+1} is the solution of (27), and let the critic neural network activation functions $\Phi_l(v)$, $l = 1, 2, \ldots, L$, be chosen such that they form a complete basis on Ω as $L \to \infty$. Under these conditions, both V_{i+1} and ∇V_{i+1} can be uniformly approximated, i.e.,*

$$\sup_{v\in\Omega}\left|\widehat{V}_{i+1}(v) - V_{i+1}(v)\right| \to 0, \tag{46}$$

$$\sup_{v\in\Omega}\left|\nabla\widehat{V}_{i+1}(v) - \nabla V_{i+1}(v)\right| \to 0, \tag{47}$$

$$\sup_{v\in\Omega}|\widehat{u}_{i+1}(v) - u_{i+1}(v)| \to 0. \tag{48}$$

Theorem 2 Based on Lemma 1, for any $\beta > 0$, there exist constants $i_0 > 0$ and $L_0 > 0$ such that when the

iteration index $i \geq i_0$ and the number of neurons $L \geq L_0$, the following inequalities hold:

$$\sup_{v\in\Omega}\left|\widehat{V}_i(v) - V^*(v)\right| < \beta, \tag{49}$$

$$\sup_{v\in\Omega}\|\widehat{u}_i(v) - u^*(v)\| < \beta. \tag{50}$$

Proof. First, we analyze the error between the network output and the true optimal value function. By applying the triangle inequality, we can decompose this error into two distinct components:

$$\sup_{v\in\Omega}\left|\widehat{V}_i(v) - V^*(v)\right| \leq \sup_{v\in\Omega}\left|\widehat{V}_i(v) - V_i(v)\right| + \sup_{v\in\Omega}\left|V_i(v) - V^*(v)\right|. \tag{51}$$

According to Lemma 1, for any fixed iteration index i, as the number of neurons $L \to \infty$, the approximation error of the neural network converges to zero:

$$\lim_{L\to\infty}\sup_{v\in\Omega}\left|\widehat{V}_i(v) - V_i(v)\right| = 0. \tag{52}$$

This implies that for any given number $\beta > 0$, we can always find a sufficiently large L_0 such that for all $L \geq L_0$,

$$\sup_{v\in\Omega}\left|\widehat{V}_i(v) - V_i(v)\right| < \frac{\beta}{2}. \tag{53}$$

According to the convergence theory of policy iteration algorithms, as the iteration index $i \to \infty$, the sequence $\{V_i\}$ converges to the optimal value function V^*,

$$\lim_{i\to\infty}\sup_{v\in\Omega}|V_i(v) - V^*(v)| = 0. \tag{54}$$

Similarly, for the given β above, we can find a sufficiently large iteration index i_0 such that for all $i \geq i_0$,

$$\sup_{v\in\Omega}|V_i(v) - V^*(v)| < \frac{\beta}{2}. \tag{55}$$

When $i \geq i_0$ and $L \geq L_0$, substituting (53) and (55) into (51) yields:

$$\sup_{v\in\Omega}\left|\hat{V}_i(v) - V^*(v)\right| < \beta. \tag{56}$$

This proves that the approximate value function $\hat{V}_i$ converges uniformly to the optimal value function V^*.

Next, the control error can be obtained from (21) and (34):

$$\begin{aligned} e_{ui} &= u_i^* - \hat{u}_i \\ &= -\frac{1}{2}R^{-1}G^T\nabla V^* + \frac{1}{2}R^{-1}G^T\nabla\Phi^T W_i \\ &= \frac{1}{2}R^{-1}G^T(\nabla\hat{V}_i - \nabla V^*). \end{aligned} \tag{57}$$

Similar to the value-function convergence argument, we can demonstrate that the gradient error also converges uniformly. This implies that for any given $\beta' > 0$, when i and L are sufficiently large, we have:

$$\sup_{v\in\Omega}\left|\nabla\hat{V}_i(v) - \nabla V^*(v)\right| < \beta'. \tag{58}$$

Taking norms on both sides of (57) gives

$$\begin{aligned} \|e_{ui}(v)\| &= \left\|\frac{1}{2}R^{-1}G^T(v)(\nabla\hat{V}_i(v) - \nabla V^*(v))\right\| \\ &\leq \left\|\frac{1}{2}R^{-1}G^T(v)\right\| \cdot \left\|\nabla\hat{V}_i(v) - \nabla V^*(v)\right\|. \end{aligned} \tag{59}$$

Since $v \in \Omega$ and $G(v)$ is continuous on Ω, the matrix norm $\|\frac{1}{2}R^{-1}G^T(v)\|$ is bounded on Ω. We define this upper bound as $b_k = \sup_{v\in\Omega} \left\|\frac{1}{2}R^{-1}G^T(v)\right\|$, taking the supremum over Ω in (59) and using (58) yields

$$\begin{aligned}\sup_{v\in\Omega}\|e_{ui}(v)\| &\leq \sup_{v\in\Omega}\left(\left\|\frac{1}{2}R^{-1}G^T(v)\right\|\cdot\left\|\nabla\hat{V}_i(v)-\nabla V^*(v)\right\|\right)\\ &\leq \left(\sup_{v\in\Omega}\left\|\frac{1}{2}R^{-1}G^T(v)\right\|\right)\cdot\left(\sup_{v\in\Omega}\left\|\nabla\hat{V}_i(v)-\nabla V^*(v)\right\|\right)\\ &= b_k\cdot\beta'.\end{aligned} \tag{60}$$

Let $\beta' = \beta/b_k$. Substituting into (60) yields:

$$\sup_{v\in\Omega}\|\hat{u}_i(v)-u^*(v)\| < \beta. \tag{61}$$

This completes the proof. □

Remark 2. *According to the conclusion of Theorem 2, we can see that the weight estimation error of the critic neural network ultimately converges to a bounded neighborhood determined by the number of neurons. This implies that as long as the number of neurons l is sufficiently large, our estimated value function $\hat{V}_i(v)$ can infinitely approach the optimal value function $V^*(v)$. Consequently, the control policy $\hat{u}_i(v)$ generated by (34) will also converge to the optimal policy $u^*(v)$, thereby ensuring the stability and near-optimal performance of the closed-loop system.*

Theorem 3 When applying the policy $\hat{u}_i(v)$ to the nonlinear system (18), the synchronization tracking error $v(t)$ of the closed-loop system is UUB.

Proof. To prove the UUB of the closed-loop system, we select the optimal value function $V^*(v)$ as the Lyapunov function candidate.

The time derivative of $V^*(v)$ along the trajectories of the system (18) under the learned control policy $\hat{u}_i(v)$ is given by:

$$\dot{V}^* = (\nabla V^*)^T(F(v)+G(v)\hat{u}_i(v)). \tag{62}$$

The optimal value function $V^*(v)$ and the optimal control policy $u^*(v)$ satisfy the HJB equation:

$$(\nabla V^*)^T(F(v)+G(v)u^*(v)) + v^TQv + u^*(v)^TRu^*(v) = 0. \tag{63}$$

Rearranging (63) yields

$$(\nabla V^*)^TF(v) = -(\nabla V^*)^TG(v)u^*(v) - v^TQv - u^*(v)^TRu^*(v). \tag{64}$$

Substituting (64) into (62) gives

$$\begin{aligned}\dot{V}^* &= -(\nabla V^*)^TG(v)u^*(v) - v^TQv - u^*(v)^TRu^*(v) + (\nabla V^*)^TG(v)\hat{u}_i(v)\\ &= -v^TQv - u^*(v)^TRu^*(v) + (\nabla V^*)^TG(v)(\hat{u}_i(v)-u^*(v)).\end{aligned} \tag{65}$$

From the optimal control law, we know that $u^*(v) = -\frac{1}{2}R^{-1}G(v)^T\nabla V^*(v)$, which implies that $(\nabla V^*)^TG(v) = -2u^*(v)^TR$. Substituting this into (65) yields

$$\begin{aligned}\dot{V}^* &= -v^TQv - u^*(v)^TRu^*(v) + (-2u^*(v)^TR)(\hat{u}_i(v)-u^*(v))\\ &= -v^TQv + u^*(v)^TRu^*(v) - 2u^*(v)^TR\hat{u}_i(v)\\ &= -v^TQv - \hat{u}_i(v)^TR\hat{u}_i(v) + (u^*(v)-\hat{u}_i(v))^TR(u^*(v)-\hat{u}_i(v)).\end{aligned} \tag{66}$$

Furthermore, according to Theorem 2, the control policy approximation error is bounded by a small positive constant β, i.e., $\|u^*(v) - \hat{u}_i(v)\| \leq \beta$. Combining these facts, we derive the following inequality:

$$\begin{aligned} \dot{V}^* &\leq -v^T Q v + (u^*(v) - \hat{u}_i(v))^T R (u^*(v) - \hat{u}_i(v)) \\ &\leq -\lambda_{\min}(Q)\|v\|^2 + \lambda_{\max}(R)\beta^2. \end{aligned} \tag{67}$$

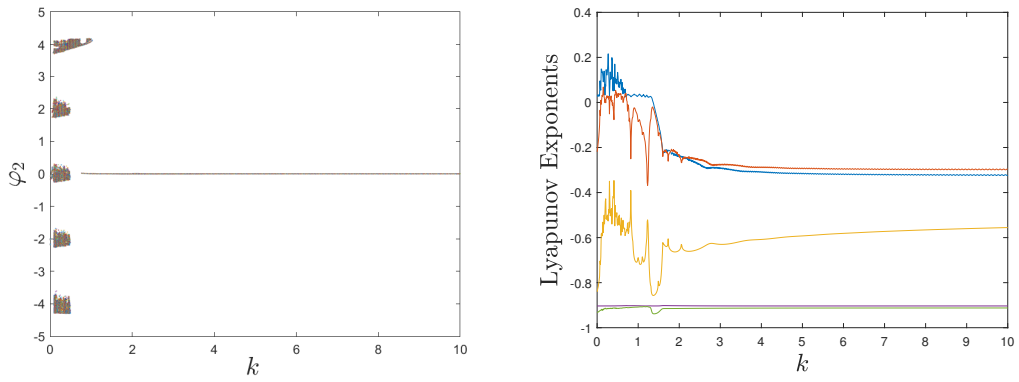
This expression shows that $\dot{V}^*(v) < 0$ as long as $\|v\|^2 > \frac{\lambda_{\max}(R)\beta^2}{\lambda_{\min}(Q)}$.

This completes the proof. □

4. Simulations

4.1. Numerical Analysis of Multi-Scroll Chaotic Dynamics

To rigorously validate its chaotic properties, Fig. 8 presents the corresponding bifurcation diagram and Lyapunov exponent spectrum of the system shown in Fig. 5. The bifurcation parameter is $k_1 = k$, while $k_2 = 1.18$ is fixed. The presence of positive Lyapunov exponents confirms the chaotic behavior of the system. These results were obtained by simulating the system over the time interval [0, 6000] with a step size of 0.1 and initial conditions (0.1, 0, 0, 0, 0). Furthermore, different bifurcation diagrams can be obtained by varying N and the core term $\mathcal{H}_i(\varphi_2)$. As shown in Fig. 9, the bifurcation structures exhibit significant variations under different parameter configurations, demonstrating that both N and $\mathcal{H}_i(\varphi_2)$ play important roles in shaping the dynamical behavior of the system. Collectively, these figures confirm that the multi-scroll M-HNN system indeed generates complex multi-scroll chaotic attractors.



(a) (b)

Fig. 8. Numerical simulation results: (a) bifurcation diagram and (b) Lyapunov exponents.

The 10-scroll attractor is not an isolated case. Further investigations reveal that the number of scrolls can be precisely controlled through the form of the activation function $\mathcal{H}(\varphi_2)$ and the parameters N. Under fixed conditions for other parameters (e.g., $k_1 = 0.35, k_2 = 1.18, n = 50$), this relationship has been systematically summarized in Table 2.

To visually validate the rules summarized in Table 2, Fig. 10 presents a gallery of typical multi-scroll attractors generated under various parameter combinations. Each subfigure demonstrates that the number of scrolls can be controlled in accordance with the tabular rules.

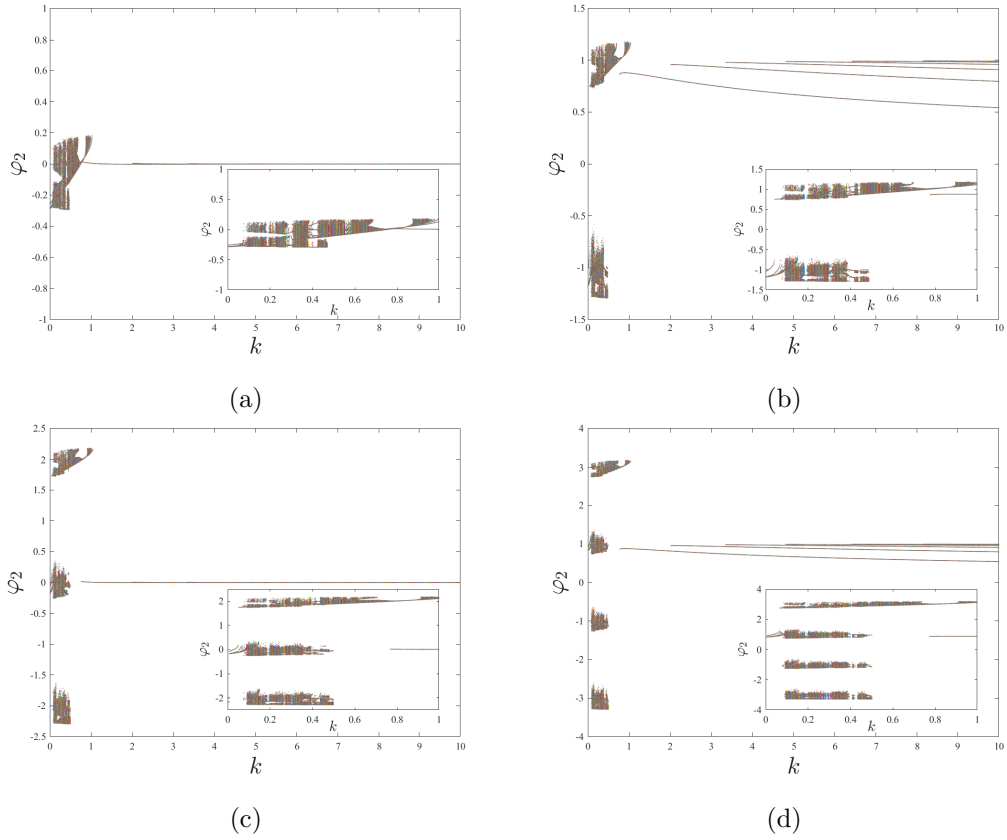


Fig. 9. Bifurcation diagrams from different forms: (a) $\mathcal{H}_1(\varphi_2)$ with $m = 0$; (b) $\mathcal{H}_2(\varphi_2)$ with $N = 0$; (c) $\mathcal{H}_1(\varphi_2)$ with $N = 0$; (d) $\mathcal{H}_2(\varphi_2)$ with $N = 1$.

Fig. 11(a) and (b) display the Poincaré map of the 8-scroll and 10-scroll attractor, respectively, in the φ_2-x_1 plane. Fig. 11(a) clearly reveals eight distinct clusters of points, confirming the existence of a 8-scroll structure. Similarly, the ten distinct clusters in Fig. 11(b) validate the 10-scroll attractor. These Poincaré maps provide conclusive evidence that the system can predictably generate multiple multi-scroll chaotic attractors according to established rules.

In summary, the above dynamical analyses demonstrate that the proposed multi-scroll M-HNN can generate chaotic driving signals with controllable scroll numbers, complex aperiodic behavior, and rich transitions across multiple scroll regions, thereby providing a dynamically rich excitation for the subsequent coverage search of the ground mobile robot.

4.2. Coverage Performance: Chaotic Driving vs. Random Walk

For the numerical simulation, we define a square workspace of $100\,\mathrm{m} \times 100\,\mathrm{m}$, resulting in a total of $\mathcal{M} = 10,000$ cells. The position of the robot is mapped to the corresponding cell according to its (X, Y) coordinates. The operational parameters are configured as follows: the robot's initial position is set to the origin, $(X(0), Y(0)) = (0, 0)$, its linear velocity is kept constant at $\mathcal{V} = 1\,\mathrm{m/s}$, and all parameters of the multi-scroll M-HNN model are fixed throughout the simulation. Fig. 12(a) and (b) illustrate the robot's trajectory generated by our proposed chaotic strategy over a total of 80,000 discrete simulation time steps. In this approach, the robot's orientation at each step is driven by the state of the memristor's magnetic flux φ_f. For comparison,

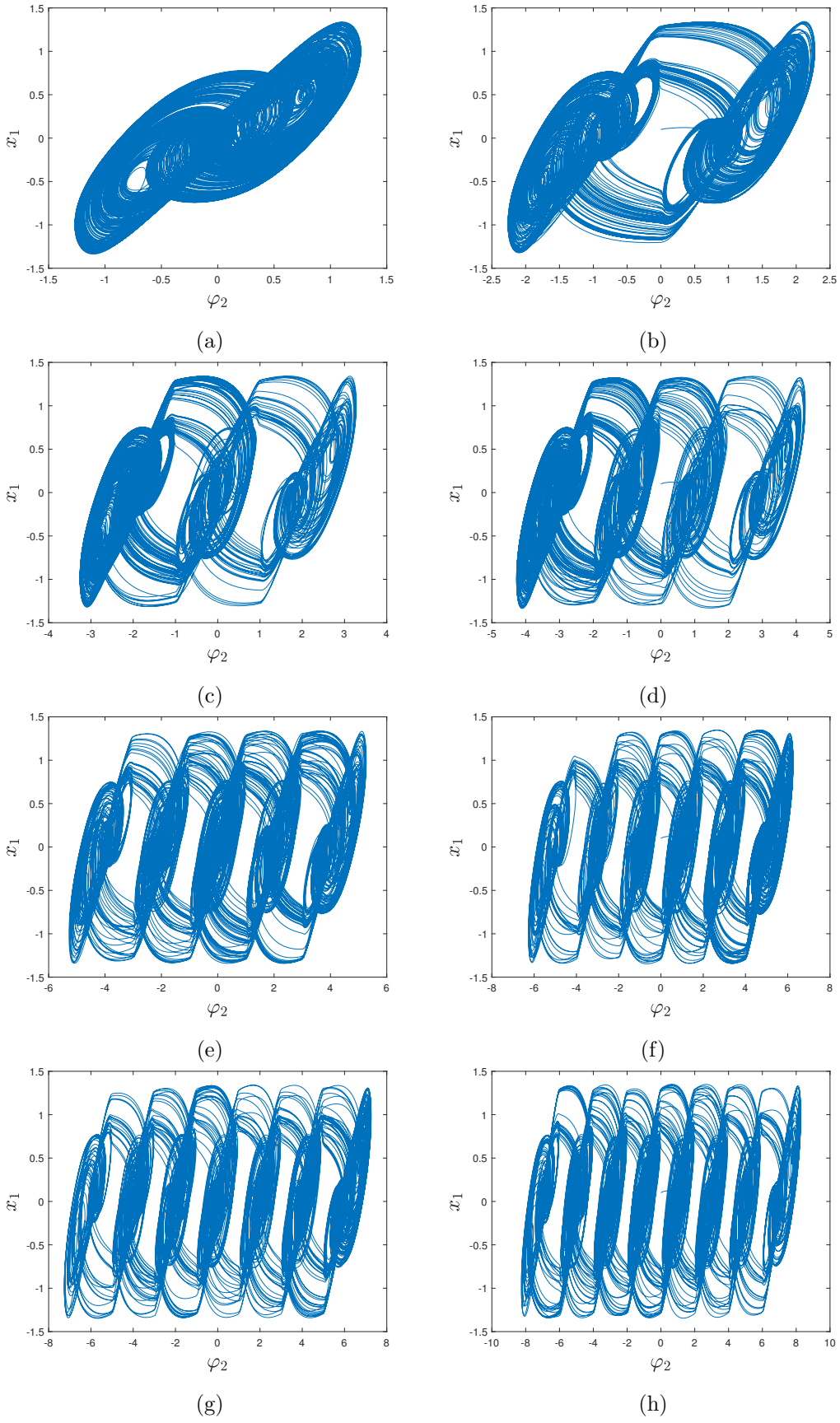
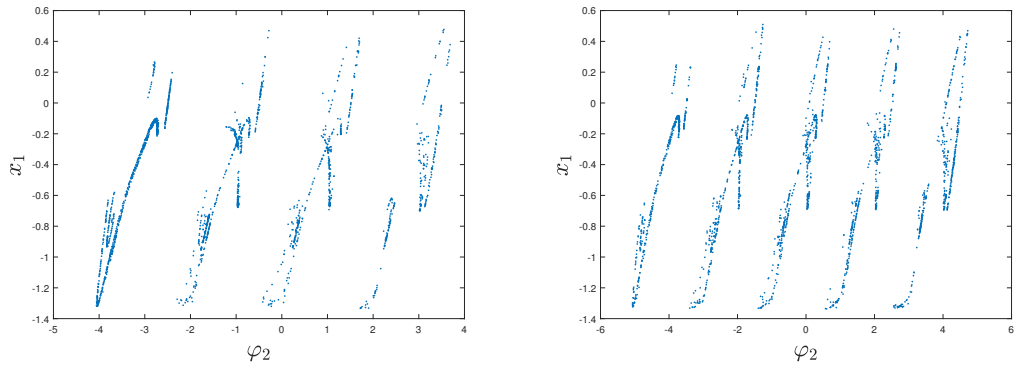


Fig. 10. Chaotic phase orbits from different forms of $\mathcal{H}(\varphi_2)$: (a) using $\mathcal{H}_1(\varphi_2)$ with $m = 0$, (b) using $\mathcal{H}_2(\varphi_2)$ with $N = 0$, (c) using $\mathcal{H}_1(\varphi_2)$ with $N = 0$, (d) using $\mathcal{H}_2(\varphi_2)$ with $N = 1$, (e) using $\mathcal{H}_1(\varphi_2)$ with $N = 1$, (f) using $\mathcal{H}_2(\varphi_2)$ with $N = 2$, (g) using $\mathcal{H}_1(\varphi_2)$ with $N = 2$, and (h) using $\mathcal{H}_2(\varphi_2)$ with $N = 3$.

Table 2. Scroll count for different forms of $\mathcal{H}(\varphi_2)$ and parameter settings (N, m).

m	Form of $\mathcal{H}(\varphi_2)$	N	Number of scrolls
0	$\mathcal{H}_1(\varphi_2)$ or $\mathcal{H}_2(\varphi_2)$	$0-N$	2
1	$\mathcal{H}_1(\varphi_2)$	0	6
		1	10
		2	14
		⋮	⋮
		N	$4N+6$
1	$\mathcal{H}_2(\varphi_2)$	0	4
		1	8
		2	12
		⋮	⋮
		N	$4N+4$



(a) (b)

Fig. 11. Poincaré map in $\varphi_2 - x_1$ plane: (a) result for the attractor with 8-scroll and (b) result for the attractor with 10-scroll.

30 independent random-walk trials are conducted under the same simulation conditions, with the heading angle is determined by a random variable sampled from a uniform distribution. Fig. 12(c) presents the trajectory from the 13th random-walk trial as a representative example, yielding a final coverage rate of 62.28%. A comparative analysis of these trajectories shows that, under the same simulation horizon, the chaotic robot, driven by the multi-scroll M-HNN, explores the workspace more efficiently than its random-walk counterpart, achieving more extensive and faster coverage.

Furthermore, Fig. 13 presents the simulation results for the coverage rate under different chaotic mapping strategies and a traditional random walk. As the benchmark, the performance of the random-walk method is characterized by the mean coverage rate and standard deviation obtained from the 30 independent trials. The results clearly demonstrate that after 80,000 iterations, all paths generated by the chaotic maps achieve a coverage rate exceeding 90%, whereas the random-walk method remains substantially lower. This demonstrates the superior coverage performance of the proposed chaotic driving strategies over the traditional random-walk method. The high unpredictability of the paths generated by these chaotic strategies promotes effective exploration of the workspace. Furthermore, under both chaotic mapping schemes, the ground mobile robot exhibits irregular motion induced by the chaotic dynamics and achieves relatively high workspace coverage.

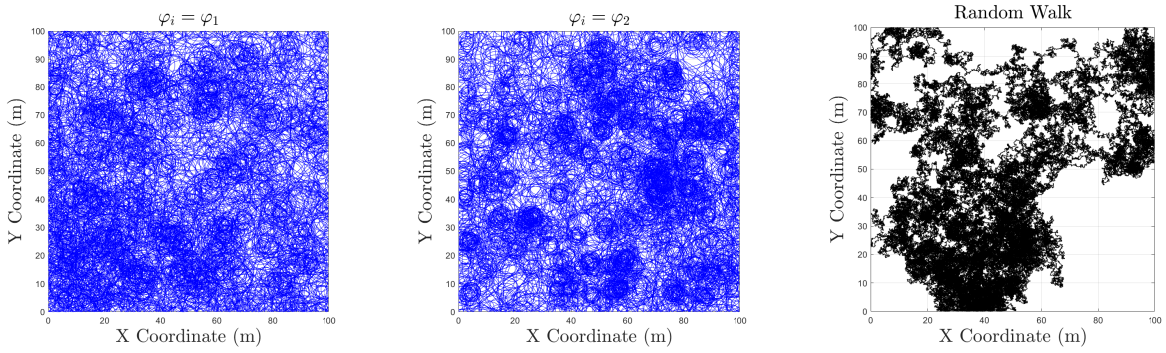


Fig. 12. Robot trajectories of system (10) using (a) φ_1 as the orientation, (b) φ_2 as the orientation and (c) a random-walk baseline.

These results indicate that the proposed chaotic mapping strategy is not dependent on a specific flux state variable. Therefore, robot trajectory driven using φ_2 is selected as the representative result for the subsequent analysis.

In summary, our proposed model not only achieves a high coverage rate faster but also demonstrates higher coverage rate compared to the random walk method.

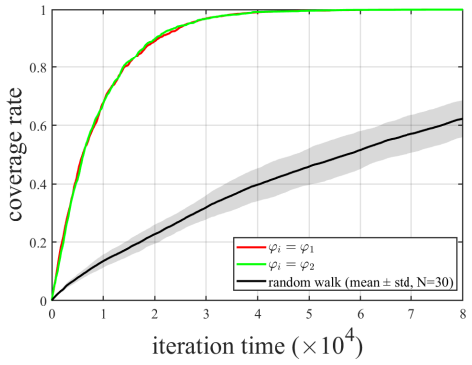


Fig. 13. Coverage rate comparison between trajectories generated by chaotic systems and random walk.

4.3. Synchronization Tracking Control Simulation

To validate the effectiveness and robustness of the proposed control algorithm, the learning and evaluation processes are conducted on the response system. The multi-scroll M-HNN system is considered with $m = 1$ (using $\mathcal{H}_1(\varphi_2)$ and $N = 1$) and initial conditions $(0.1, 0, 0, 0, 0)$. The weighting matrices for the cost function are set to $Q = \text{diag}([50, 50, 50, 50, 50, 200, 200, 100])$ and $R = \text{diag}(\mathbf{1}_{8\times1}) \times 0.1$. The critic network approximates the value function as $V(v) = W^T\Phi(v)$ by employing a quadratic basis function vector $\Phi(v)$, where W is a 36-dimensional weight vector. To accelerate the convergence, the initial weight vector W_0 is obtained by solving the Linear Quadratic Regulator (LQR) problem for the corresponding linearized system. During the policy iteration learning process, the learning rate and regularization parameter are set to $\delta = 0.0005$ and $\mu = 80$, respectively. The critic neural network basis-function vector is selected as

$$\begin{aligned}\Phi(v) = [&v_1^2, \sqrt{2}v_1v_2, \sqrt{2}v_1v_3, \sqrt{2}v_1v_4, \sqrt{2}v_1v_5, \sqrt{2}v_1v_6,\\ &\sqrt{2}v_1v_7, \sqrt{2}v_1v_8, v_2^2, \sqrt{2}v_2v_3, \sqrt{2}v_2v_4, \sqrt{2}v_2v_5,\\ &\sqrt{2}v_2v_6, \sqrt{2}v_2v_7, \sqrt{2}v_2v_8, v_3^2, \sqrt{2}v_3v_4, \sqrt{2}v_3v_5,\\ &\sqrt{2}v_3v_6, \sqrt{2}v_3v_7, \sqrt{2}v_3v_8, v_4^2, \sqrt{2}v_4v_5, \sqrt{2}v_4v_6,\\ &\sqrt{2}v_4v_7, \sqrt{2}v_4v_8, v_5^2, \sqrt{2}v_5v_6, \sqrt{2}v_5v_7, \sqrt{2}v_5v_8,\\ &v_6^2, \sqrt{2}v_6v_7, \sqrt{2}v_6v_8, v_7^2, \sqrt{2}v_7v_8, v_8^2]^T.\end{aligned} \tag{68}$$

The convergence process of the critic network's weight vector is illustrated in Figs. 14-16, where $W(\iota)$ represents the ι-th element of W. After 8000 iterations, the learned optimal weight vector W^* is given by

$$\begin{aligned}W^* = [&0.0167, 0.0005, -0.0034, 0.0004, 0.0015, 0.0000,\\ &-0.0000, 0.0000, 0.0147, 0.0022, 0.0035, -0.0002,\\ &0.0000, 0.0000, -0.0000, 0.0100, 0.0001, 0.0002,\\ &0.0000, -0.0000, 0.0000, 0.0111, -0.0000, -0.0000,\\ &0.0000, -0.0000, 0.0114, 0.0000, 0.0000, 0.0001,\\ &0.0818, 0.0002, -0.0000, 0.0815, 0.0021, 0.0184]^T.\end{aligned} \tag{69}$$

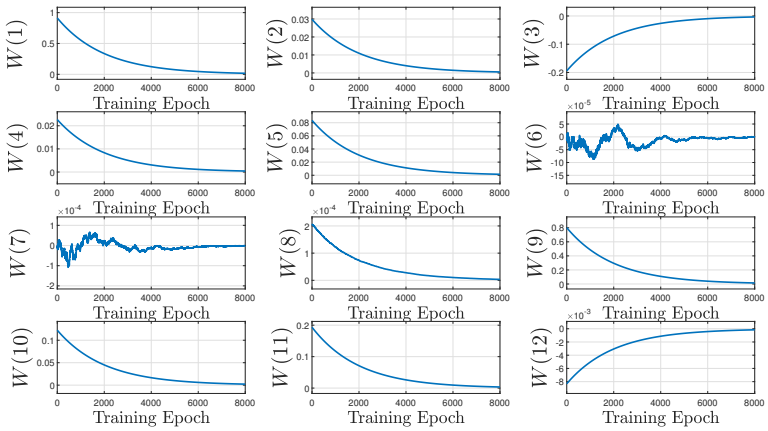


Fig. 14. $W(1)$ to $W(12)$.

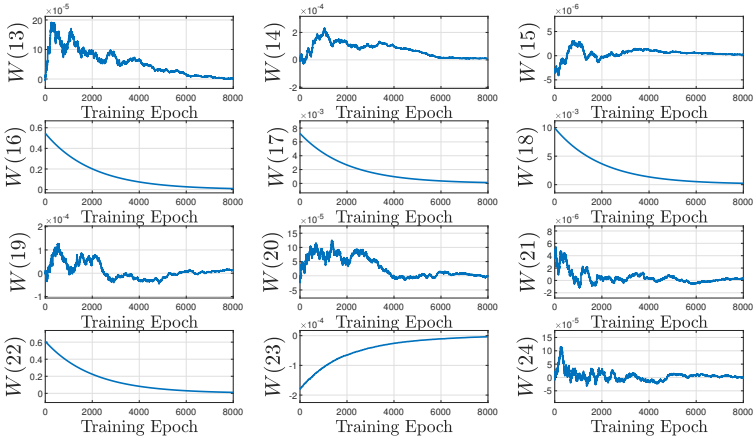


Fig. 15. $W(13)$ to $W(24)$.

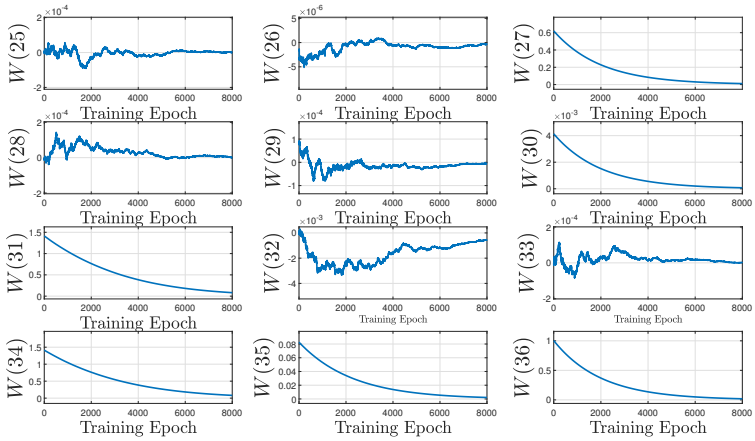


Fig. 16. $W(25)$ to $W(36)$.

To evaluate the performance of the control policy corresponding to this optimal weight, it is deployed in the online synchronization task with a total simulation time of 16,000 seconds. Fig. 17 depicts the synchronization tracking performance of the drive and response systems. Figs. 18 and 19 show the control input signals and the synchronization tracking errors, respectively. As observed from the simulation results, the trajectory of the response system rapidly converges to that of the drive system. This result indicates that the learned controller can quickly stabilize the synchronization tracking error to a neighborhood of zero, thereby validating the effectiveness and feasibility of the proposed control strategy.

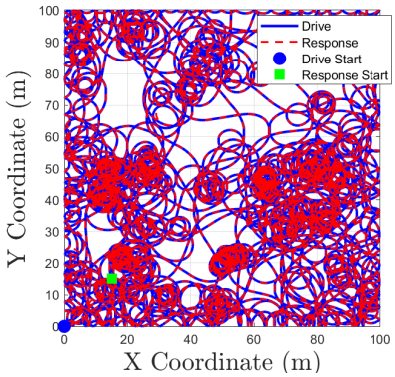


Fig. 17. State trajectories of the drive and response systems.

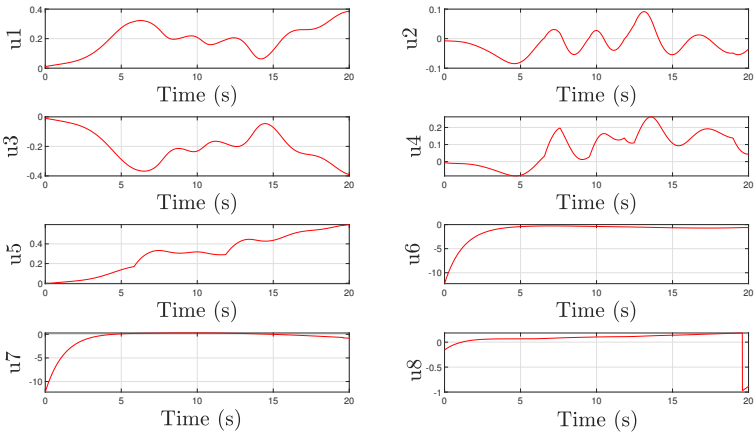


Fig. 18. Control input signals.

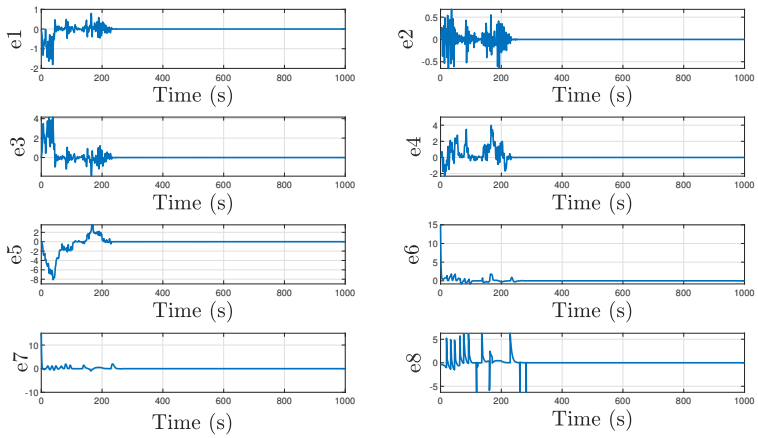


Fig. 19. Synchronization Tracking Error.

5. Conclusion

This paper investigates a chaotic-signal-driven synchronization control framework for air-ground collaborative coverage search tasks. A multi-scroll M-HNN is designed as the chaotic motion source for the ground search robot, and a critic-only IRL synchronization strategy is developed to synchronize the planar projection of the UAV motion with the chaotic search trajectory of the ground robot. The proposed hyperbolic-tangent memristive synapse enables the designed chaotic multi-scroll M-HNN to generate a controllable number of scroll attractors. Dynamical analysis shows that the designed multi-scroll M-HNN provides an effective chaotic source with stronger ergodicity for robotic search motion generation. Furthermore, when the air-ground collaborative search task is considered, a critic-only IRL synchronization controller is proposed. The results demonstrate improved synchronization accuracy and convergence performance. Future work will focus on experimental validation on physical robotic platforms and on extending the proposed framework to networked multi-agent systems.

Data availability statement

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 62573274, and the Shandong Provincial Natural Science Foundation ZR2024MF001.

References

[1] Ramesh M, Imeson F, Fidan B and Smith S L 2024 *IEEE Trans. Robot.* **40** 4190–4206

[2] Tang J, Mao Z and Ma H 2025 *IEEE Trans. Robot.* **41** 3348–3367

[3] Yu D X, Xu H, Chen C P L, Bai W and Wang Z 2021 *IEEE Trans. Ind. Electron.* **69** 5333–5341
[4] Schwager M, Rus D and Slotine J J 2009 *Int. J. Robot. Res.* **28** 357–375
[5] Zhang L, Zhang Z, Siegwart R and Chung J J 2022 *IEEE Trans. Robot.* **39** 147–164
[6] Müller M A and Domínguez-García A D 2012 *Automatica* **48** 1372–1379
[7] Volos C K, Kyprianidis I M and Stouboulos I N 2012 *Robot. Auton. Syst.* **60** 651–656
[8] Nasr S, Mekki H and Bouallegue K 2019 *Chaos Solitons Fractals* **118** 366–375
[9] Lü J, Han F, Yu X and Chen G 2004 *Automatica* **40** 1677–1687
[10] Zhang L, Li Z and Peng Y 2024 *Chaos Solitons Fractals* **185** 115109
[11] Yan D, Ji M and Wang L, Duan S and Du X 2022 *Nonlinear Dyn.* **107** 3919–3944
[12] Wu J, Wang L, Chen G and Duan S 2016 *Chaos Solitons Fractals* **92** 20–29
[13] Joksas D, Freitas P, Chai Z, Ng W H, Buckwell M, Li C and Mehonic A 2020 *Nat. Commun.* **11** 4273
[14] Liu X, Sun K, Wang H and He S 2023 *Chaos Solitons Fractals* **174** 113791
[15] Li J, Wang C and Deng Q 2024 *Nonlinear Dyn.* **112** 14463–14477
[16] Lai Q and Chen Y 2023 *Chaos* **33** 083119
[17] Chen L, Pan W, Wu R, Wang K and He Y 2016 *Chaos Solitons Fractals* **85** 22–31
[18] Lin H, Wang C, Deng Q, Xu C, Deng Z and Zhou C 2021 *Nonlinear Dyn.* **106** 959–973
[19] Zhang X, Yang X and Huang C, Cao J, Liu H 2024 *Inf. Sci.* **661** 120148
[20] Wang Y W and Guan Z H 2006 *Chaos Solitons Fractals* **27** 97–101
[21] Li W L and Chang K M 2009 *Chaos Solitons Fractals* **39** 2086–2092
[22] Yaghoubi Z, Elmi M and Adeli M 2025 *IET Contr. Theory Appl.* **19** 450–461
[23] Kim J and Jin M 2016 *Nonlinear Dyn.* **86** 2003–2015
[24] Chang W D 2009 *Chaos Solitons Fractals* **39** 910–917
[25] Pecora L M and Carroll T L 2015 *Chaos* **25** 097611
[26] Lee J Y, Park J B and Choi Y H 2014 *IEEE Trans. Neural Netw. Learn. Syst.* **26** 916–932
[27] Li H, Liu D and Wang D 2014 *IEEE Trans. Autom. Sci. Eng.* **11** 706–714
[28] Modares H and Lewis F L 2014 *Automatica* **50** 1780–1792
[29] Guo X, Yan W and Cui R 2019 *IEEE Trans. Syst. Man Cybern. Syst.* **50** 4068–4077
[30] Song R, Lewis F L and Wei Q 2016 *IEEE Trans. Neural Netw. Learn. Syst.* **28** 704–713
[31] Chen J, Cao J, Cheng Z and Jiang S 2023 *IEEE Trans. Mob. Comput.* **23** 3605–3619
[32] Zhang S, Zheng J, Wang X, Zeng Z and He S 2020 *Nonlinear Dyn.* **102** 2821–2841
[33] Li Z, Jin J, Zhang D and Chen C 2025 *Integration, the VLSI Journal* **105** 102500
[34] He S, Xiao J, Chen Q and Wang H 2026 *Chin. Phys. B* **35** 060502
[35] Zhu L, Cheng J, Zhang H, Zhang W and Liu Y 2024 *IEEE Trans. Intell. Transp. Syst.* **25** 5022–5033
[36] Zhang D and Zhao D 2019 *IEEE Trans. Cybern.* **49** 2874–2884
[37] Song R, Yang G and Lewis F L 2024 *IEEE Trans. Neural Netw. Learn. Syst.* **35** 2793–2802
[38] Abu-Khalaf M and Lewis F L 2005 *Automatica* **41** 779–791