Nonlinear Dyn
https://doi.org/10.1007/s11071-025-11021-6

RESEARCH

Unmanned aerial-ground vehicle finite-time docking control via pursuit-evasion games

Junkai Tan · Shuangsi Xue · Qingshu Guan · Tiansen Niu · Hui Cao · Badong Chen

Received: 29 December 2024 / Accepted: 17 February 2025

Abstract Cooperation between unmanned autonomous systems has attracted increasing attention in recent years, particularly the challenging problem of unmanned aerial vehicle (UAV) and unmanned ground vehicle (UGV) docking in complex environments with dynamic vehicle interactions. This paper proposes a novel finite-time reinforcement learning control scheme for UAV–UGV docking based on a pursuit-evasion game framework. A pursuit-evasion game formulation is developed where the evader vehicle navigates through complex environments while being pursued by a pursuer vehicle required to track and dock with it. The docking performance is optimized through achieving Nash equilibrium of the pursuit-evasion game. The proposed finite-time reinforcement learning algorithm transforms the value function to finite-time space and employs Actor-Critic neural networks to approximate the value function and optimal controller. A finite-time concurrent learning law is utilized to update the neural network weights, ensuring both the pursuit-evasion game equilibrium and learning process converge within finite time. Lyapunov stability analysis proves the finite-time convergence properties of the algorithm. Experimental validation on an aerial-ground vehicle system demonstrates the effectiveness of the proposed approach in achieving optimal pursuit-evasion performance while maintaining safe landing capability.

Keywords Unmanned aerial vehicle · Unmanned ground vehicle · Pursuit-evasion game · Finite-time reinforcement learning · Docking control

1 Introduction

The cooperation between unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) has garnered increasing attention in recent years due to its potential to enhance system performance [1,2], operational flexibility [3], and system robustness [4]. Among various challenges in UAV–UGV cooperation, the docking control problem [5] remains one of the most critical yet difficult tasks, where vehicles must navigate through complex environments to establish stable connections for payload transfer or battery charging [6]. The complexity arises from multiple factors: dynamic interactions between vehicles [7,8], environmental uncertainties affecting UAV [9,10], uneven terrain impacting UGV [11], and obstacles in the environment [12]. These challenges necessitate sophisti-

J. Tan · S. Xue (✉) · Q. Guan · T. Niu · H. Cao
The Shaanxi Key Laboratory of Smart Grid, The State Key Laboratory of Electrical Insulation and Power Equipment, and School of Electrical Engineering, Xi'an Jiaotong University, Xi'an 710049, China
e-mail: xssxjtu@xjtu.edu.cn

S. Xue · B. Chen
The National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center for Visual Information and Applications, and the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, China

Published online: 15 March 2025

cated control strategies to ensure successful docking operations. While several interaction frameworks have been proposed to address these challenges, including robust control [13], hierarchical control [14], cooperative game theory [15], and reinforcement learning [16]. However, existing approaches often require long convergence times and lack finite-time guarantees, limiting their applicability to time-critical applications like UAV–UGV docking [7,17].

The pursuit-evasion game (PEG) framework offers a rigorous mathematical foundation for analyzing such dynamic multi-agent interactions [18,19]. In this formulation, two agents with competing objectives interact: an evader that navigates through the environment while attempting to evade, and a pursuer that aims to intercept and capture the evader. The PEG framework has demonstrated success across various domains including security systems [20], robotics [21], and differential game theory [15]. For the UAV–UGV docking problem, PEG provides an elegant solution where the UAV acts as the evader navigating through complex environments while being pursued by the UGV pursuer for tracking and docking [22,23]. Compared to traditional control strategies, PEG offers a more robust and efficient approach to optimizing docking performance [1,2]. This game-theoretic approach enables systematic analysis and optimization of vehicle interactions through Nash equilibrium solutions [18,24]. However, existing PEG-based approaches mainly focus on homogeneous multi-agent systems with simple interaction dynamics [25], and lack practical implementation strategies for complex UAV–UGV docking scenarios [26,27].

Adaptive dynamic programming and reinforcement learning (ADP&RL) has emerged as a powerful tool for learning optimal control policies in complex and uncertain environments [28,29]. Traditional ADP&RL methods have achieved remarkable success in robotics, autonomous driving, and UAV control [30,31], particularly through model-free approaches that don't require explicit system dynamics [32,33]. However, these methods typically require infinite time horizons to theoretically guarantee optimality [34,35], and lack explicit convergence time bounds needed for safety-critical operations [8,22]. These limitations have motivated the development of finite-time reinforcement learning (FT-RL) algorithms [36,37]. The key innovation of FT-RL lies in transforming the value function into finite-time space while maintaining the learning capabilities of traditional RL. This transformation enables critical advantages in finite-time convergence [38], and provable stability properties [39,40]. This makes FT-RL well-suited for time-critical applications like UAV–UGV docking. Recent advances in FT-RL have demonstrated promising results for multi-agent systems [41,42] and game scenarios [9,19]. However, most existing FT-RL approaches focuses on simpler interaction scenarios without considering the full complexity of aerial-ground vehicle coupling, Also, regular current learning architectures may not efficiently capture the unique dynamics of pursuit-evasion games, and finite-time convergence guarantees are often limited to specific system classes rather than general game-theoretic frameworks [43,44].

To overcome these limitations, this paper proposes a novel finite-time reinforcement learning with pursuit-evasion game equilibrium guidance (FT-RL-PEG) scheme. The key innovation lies in reformulating the optimal control problem to ensure both the pursuit-evasion game equilibrium and learning process converge within an analytically bounded time horizon compared to regular ADP&RL algorithms [20,29,31]. The framework employs an actor-critic architecture where the critic network approximates the finite-time value function while the actor network learns the optimal control policy [8,28,36]. A carefully designed finite-time concurrent learning law updates neural network weights with provable convergence properties [22,41,42]. Through rigorous Lyapunov stability analysis, we establish theoretical guarantees on the finite-time convergence of both value function approximation and optimal control policy [1,39,45]. This enables achieving optimal performance within predefined time bounds—a critical capability for practical UAV–UGV docking applications requiring rapid and reliable convergence [2,43,46]. Motivated by the above challenges posed by UAV–UGV docking, this paper makes the following contributions:

(1) A novel PEG framework is developed for unmanned aerial-ground vehicle docking, where the evader vehicle navigates through complex environments while being pursued by a pursuer vehicle required to track and dock with it. The docking performance is optimized through achieving Nash equilibrium of the PEG, which is more robust and efficient than traditional docking control strategies [1,2,8,22].

(2) A novel FT-RL framework is proposed to integrate PEG with finite-time learning. Our work provides a unified approach that overcomes previous theoretical gap between game theory and FT-RL [29,37]. By combining the stability guarantees of FT-RL with the competitive equilibrium properties of PEG, our framework ensures finite-time convergence with provable stability guarantees for real-time equilibrium solving and learning, overcoming the infinite-horizon limitations of traditional methods [8,22].
(3) Comprehensive numerical simulations and experimental validation on an real-world aerial-ground vehicle system demonstrate the effectiveness of the proposed approach in achieving optimal pursuit-evasion performance while maintaining safe landing capability, while Lyapunov stability analysis proves the finite-time convergence properties of the algorithm [1,39,45].

The rest of this paper is organized as follows: Section 2 presents system dynamics and finite-time optimal control problem. Section 3 introduces the proposed FT-RL-PEG algorithm. Sections 4 and 5 provide simulation examples and hardware experiments to verify the effectiveness of FT-RL-PEG scheme. Section 6 concludes the paper.

Notation: In this paper, $\mathbb{R}$ denotes the set of real numbers, $\mathbb{R}^n$ denotes the n-dimensional Euclidean space, $\mathbb{R}^{n\times m}$ denotes the set of $n \times m$ real matrices, $\mathbb{N}$ denotes the set of natural numbers, $\dagger$ denotes the Moore–Penrose pseudo-inverse, $\|\cdot\|$ denotes the Euclidean norm, $\mathrm{sign}(\cdot)$ denotes the sign function, $\mathrm{sig}^{\alpha}(\cdot) = |\cdot|^{\alpha}\,\mathrm{sgn}(\cdot)$ denotes the fractional power signum function, $\tanh(\cdot)$ denotes the hyperbolic tangent function,

2 Preliminaries

2.1 System description

The interconnected pursuit-evasion system is composed of evader and pursuer agents, where the evader agent navigates through complex environments while being pursued by the pursuer agent, and the pursuer agents track and capture the evader agents. Consider single evader and single pursuer with the following nonlinear affine input dynamics:

$$\begin{cases} \dot{x}_e(t) = f_e(x_e(t)) + g_e(x_e(t))\mathscr{U}_e(t) \\ \dot{x}_p(t) = f_p(x_p(t)) + g_p(x_p(t))\mathscr{U}_p(t) \end{cases} \tag{1}$$

where $x_e \in \mathbb{R}^{n_e}$, $x_p \in \mathbb{R}^{n_p}$ denote the states of the evader and the pursuer, respectively, $\mathscr{U}_e \in \mathbb{R}^{m_e}$, $\mathscr{U}_p \in \mathbb{R}^{m_p}$ denote the control inputs, $f_e : \mathbb{R}^{n_e} \to \mathbb{R}^{n_e}$, $f_p : \mathbb{R}^{n_p} \to \mathbb{R}^{n_p}$ denote the drift dynamics, and $g_e : \mathbb{R}^{n_e} \to \mathbb{R}^{n_e\times m_e}$, $g_p : \mathbb{R}^{n_p} \to \mathbb{R}^{n_p\times m_p}$ denote the control effectiveness matrices. The evader and the pursuer are connected by a communication channel, which transmits the real-time state information of the evader to the pursuer. Assume that the communication channel is ideal, i.e., the evader's state information can be transmitted to the pursuer without any delay or loss. For the pursuer, to track the evader agent, The tracking error is defined as $x = x_p - x_e$. The tracking error dynamics is derived as:

$$\begin{aligned} \dot{x}(t) =& \dot{x}_p(t) - \dot{x}_e(t) \\ =& f_p(x_p) - f_e(x_e) + g_p(x_p)\mathscr{U}_p(t) - g_e(x_e)\mathscr{U}_e(t) \\ =& f(x) + g(x)\mathscr{U}_p + k(x)\mathscr{U}_e \end{aligned} \tag{2}$$

where $f(x) = f_p(x_p) - f_e(x_e)$, $g(x) = g_p(x_p)$, and $k(x) = -g_e(x_e)$ represent the drift dynamics and control effectiveness matrices of the error system.

2.2 Optimal control problem

First, we define the following cost function:

$$\mathscr{V}_i(x, \mathscr{U}_p, \mathscr{U}_e) = \int_0^{\infty} r_i\big(x(\tau), \mathscr{U}_p(\tau), \mathscr{U}_e(\tau)\big)\,\mathrm{d}\tau, \quad i = p, e \tag{3}$$

where the control inputs $\mathscr{U}_i(t)$ are constrained within symmetric bounds μ_i, i.e., $-\mu_i \le \mathscr{U}_i(t) \le \mu_i$. The instantaneous reward function $r_i(x, \mathscr{U}_p, \mathscr{U}_e)$ for the i-th player is defined as:

$$\begin{aligned} r_p(x, \mathscr{U}_p, \mathscr{U}_e) =& -r_e(x, \mathscr{U}_p, \mathscr{U}_e) \\ =& \left(x^{\top}\omega x\right)^{\alpha} + \Lambda_p(\mathscr{U}_p) - \Lambda_e(\mathscr{U}_e) \end{aligned} \tag{4}$$

where $\omega = Q_i \in \mathbb{R}^{n\times n}$ is a positive definite weight matrix, $0 < \alpha < 1$ is a fractional order, and $\Lambda_i(\mathscr{U}_i)$ is

the penalty function for the control input, which penalizes large control signals and helps maintain stability [44]:

$$\Lambda_i(\mathscr{U}_i) = 2\mu_i R_i \int_0^{\mathscr{U}_i} \tanh^{-1}\left(\frac{\gamma}{\mu_i}\right) \mathrm{d}\gamma, \quad i = p, e \tag{5}$$

where $R_i \in \mathbb{R}^{m_i \times m_i}$ is a positive definite weight matrix that penalizes control effort, and γ is the integration variable. This penalty function grows rapidly as control inputs approach the saturation bounds $\pm\mu_i$, effectively preventing control saturation. To facilitate the subsequent analysis and controller design for the dynamics (2), we make the following assumption and definition for the pursuit-evasion game of a two-player system.

Assumption 2.1 (**Lipschitz Continuity and Boundedness** [16,43]) For the pursuit-evasion system dynamics (2), the following conditions hold:

(1) For any tight set $x \in \chi \subset \mathbb{R}^n$:

- The drift dynamics $f(x)$ is Lipschitz continuous with $f(0) = 0$
- The control effectiveness matrices $g_i(x)$ are Lipschitz continuous and bounded: $\|g_i(x)\| \leq G_{Hi}$

(2) The cost matrices satisfy uniform boundedness conditions:

- State cost: $0 \leq \underline{\lambda}_{Q_i} \leq \lambda_{\min}(Q_i) \leq \lambda_{\max}(Q_i) \leq \bar{\lambda}_{Q_i}$
- Control cost: $0 \leq \underline{\lambda}_{R_{ij}} \leq \lambda_{\min}(R_{ij}) \leq \lambda_{\max}(R_{ij}) \leq \bar{\lambda}_{R_{ij}}$

where $\{\underline{\lambda}_{Q_i}, \underline{\lambda}_{R_{ij}}, \bar{\lambda}_{Q_i}, \bar{\lambda}_{R_{ij}}\}$ are positive constants.

Definition 1 (**Nash Equilibrium** [18,24]) Consider the pursuit-evasion game (PEG) system (2) with control inputs $\{\mathscr{U}_p, \mathscr{U}_e\}$. A Nash equilibrium is achieved if:

$$\begin{cases} \mathscr{V}_p^*(x) = \mathscr{V}_p(x, \mathscr{U}_p^*, \mathscr{U}_e^*) \leq \mathscr{V}_p(x, \mathscr{U}_p, \mathscr{U}_e^*), & \forall \mathscr{U}_p \in \Omega_U \\ \mathscr{V}_e^*(x) = \mathscr{V}_e(x, \mathscr{U}_p^*, \mathscr{U}_e^*) \leq \mathscr{V}_e(x, \mathscr{U}_p^*, \mathscr{U}_e), & \forall \mathscr{U}_e \in \Omega_U \end{cases} \tag{6}$$

where $\mathscr{V}_i^*(x)$ denotes the optimal value function for the i-th player at Nash equilibrium $\{\mathscr{V}_p^*, \mathscr{V}_e^*\}$.

The optimal value function $\mathscr{V}_i^*(x)$ satisfies:

$$\mathscr{V}_i^*(x) = \int_t^\infty r_i\big(x(\tau), \mathscr{U}_p^*(\tau), \mathscr{U}_e^*(\tau)\big)\mathrm{d}\tau, \quad i = p, e, \tag{7}$$

where $\Omega_U \subset \mathbb{R}^m$ denotes the admissible control input set. The corresponding Hamilton function is defined as:

$$\begin{aligned} H_i(x, \mathscr{U}_p, \mathscr{U}_e, \nabla\mathscr{V}_i^*) &= r_i(x, \mathscr{U}_p, \mathscr{U}_e) \\ &+ \nabla\mathscr{V}_i^{*\top}(f + g\mathscr{U}_p + k\mathscr{U}_e) \end{aligned} \tag{8}$$

where $\nabla\mathscr{V}_i^* = \frac{\partial\mathscr{V}^*}{\partial x}$ represents the optimal value function gradient. By solving the extreme condition of (14) and (8), the Nash equilibrium control input is derived as:

$$\begin{cases} \mathscr{U}_p^*(x) = \underset{\mathscr{U}_p \in \Omega_U}{\operatorname{argmin}} \; H_p = -\mu_p \tanh\left(\dfrac{R_p^{-1} g^\top}{2\mu_p} \nabla\mathscr{V}_p^{*\top}\right) \\ \mathscr{U}_e^*(x) = \underset{\mathscr{U}_e \in \Omega_U}{\operatorname{argmin}} \; H_e = \mu_e \tanh\left(\dfrac{R_e^{-1} k^\top}{2\mu_e} \nabla\mathscr{V}_e^{*\top}\right) \end{cases} \tag{9}$$

Combining the Nash equilibrium control input (9) with the optimal value function (14), the Hamilton–Jacobi–Issac (HJI) equation is derived as:

$$\begin{cases} 0 = \underbrace{\nabla\mathscr{V}_p^{*\top}(f + g\mathscr{U}_p + k\mathscr{U}_e)}_{\text{Dynamics term}} + \underbrace{\Lambda_p(\mathscr{U}_p) - \Lambda_e(\mathscr{U}_e)}_{\text{Input penalty}} + \underbrace{|x|_\omega^\alpha}_{\text{State cost}} \\ 0 = -\underbrace{\nabla\mathscr{V}_e^{*\top}(f + g\mathscr{U}_p + k\mathscr{U}_e)}_{\text{Dynamics term}} + \underbrace{\Lambda_p(\mathscr{U}_p) - \Lambda_e(\mathscr{U}_e)}_{\text{Input penalty}} - \underbrace{|x|_\omega^\alpha}_{\text{State cost}} \end{cases} \tag{10}$$

The optimal controller defined in Eq. (9) is designed to stabilize the system states at the optimal equilibrium point, as defined in Eq. (14). However, it should be noted that the performance of the optimal controller is not guaranteed. The next subsection will introduce the FT optimal controller, which is capable of achieving FT stabilization control of the system, as defined in Eq. (2).

2.3 Transformed finite-time value function

To achieve finite-time stabilization of system states, we first transform the value function into finite-time (FT) stable space. We begin by presenting two key definitions that characterize the FT stability properties.

Definition 2 (**Finite-Time Stability**) The system state $x(t)$ of (2) is finite-time stable with respect to equilibrium point x^* if there exists a settling time $T[x(0)] \in (0, \infty)$ such that:

$$\mathscr{V}\{x(t_f), x^*\} \leq \delta, \quad \forall \delta > 0, \quad \forall t_f \geq T[x(0)]$$

where $\mathscr{V}\{x(t), x^*\} : \Omega_n \rightarrow [0, L_V]$ is a smooth distance function.

Definition 3 (**Finite-Time Value Function**) A finite-time value function $\Xi_i(x, x^*) \geq 0$ satisfies:

(1) $\nabla \Xi_i(x^*, x^*) = 0$ at equilibrium
(2) $\nabla^2 \Xi_i \geq 0$ for all $x \in \Omega_n$

where $\Omega_n = \{x \in \mathbb{R}^n \mid \mathscr{V}(x, x^*) \leq L_V\}$ defines the state space neighborhood.

To derive the FT optimal controller, we transform the asymptotic value function $\mathscr{V}_i(x)$ from (7) into FT space using:

$$\mathscr{V}_i\{x, x^*\} = \int_x^{x^*} \text{sig}^{\frac{\alpha}{2}}(\nabla \Xi_i(\zeta, x^*)) \mathrm{d}\zeta, \quad i = p, e \tag{11}$$

where $\text{sig}^{\frac{\alpha}{2}}(\cdot) = |\cdot|^{\frac{\alpha}{2}} \text{sgn}(\cdot)$ is the fractional power signum function, $\alpha \in (0, 1)$ is the fractional order parameter, $\text{sgn}(\cdot)$ is the sign function, and $\nabla \Xi_i(x, x^*)$ is the gradient of the FT value function.

This transformation ensures finite-time convergence properties while maintaining smoothness of the value function. With the transformed value function (11) in the FT stable space, we derive the corresponding transformed FT Hamiltonian function:

$$\begin{aligned} &\mathscr{H}_i\left(x, x^*, \mathscr{U}_p, \mathscr{U}_e, \nabla \Xi_i\right) \\ &\quad = \text{sig}^{\frac{\alpha}{2}}(\nabla \Xi_i)^\top (f + g\mathscr{U}_p + k\mathscr{U}_e) \\ &\qquad + r_i(x, \mathscr{U}_p, \mathscr{U}_e), \ i = p, e \end{aligned} \tag{12}$$

The FT optimal controller can be obtained by solving the extremum condition of the transformed Hamiltonian function (12):

$$\begin{cases} \mathscr{U}_p^*(x) = -\mu_p \tanh\left(\dfrac{R_p^{-1} g^\top}{2\mu_p} \text{sig}^{\frac{\alpha}{2}}(\nabla \Xi_p^{*\top})\right) \\ \mathscr{U}_e^*(x) = \mu_e \tanh\left(\dfrac{R_e^{-1} k^\top}{2\mu_e} \text{sig}^{\frac{\alpha}{2}}(\nabla \Xi_e^{*\top})\right) \end{cases} \tag{13}$$

where $\mathscr{U}_p^*(x)$ represents the optimal pursuit control and $\mathscr{U}_e^*(x)$ denotes the optimal evasion control. The optimal controller (13) is designed to stabilize the system states at the optimal equilibrium point with finite-time convergence guarantees through the transformed value function (11). To implement this controller in practice, we need to approximate both the finite-time value function and optimal control policy. This motivates the development of our FT-RL-PEG algorithm presented in the next section.

3 Main result: FT-RL-PEG algorithm

3.1 Approximation of finite-time value function

To approximate the FT optimal value function and optimal controller with identified system dynamics, actor and critic neural networks (NN) are employed to approximate the value function and the optimal controller, respectively. First, the following critic NN is designed to approximate the value function in the FT stable space:

$$\mathscr{V}_i^*(x) = W_{ci}^{*\top} \psi_i(x) + \epsilon_i^*(x), \ i = p, e \tag{14}$$

where W_{ci}^* denotes the optimal weights of the critic NN, $\psi_i(x) \in \mathbb{R}^N$ represents the basis function of the critic NN, and $\epsilon_i^*(x)$ indicates the approximation error. The critic NN is estimated in practice as:

$$\hat{\mathscr{V}}_i(x) = \hat{W}_{ci}^\top \psi_i(x) + \epsilon_i(x), \ i = p, e \tag{15}$$

where $\hat{W}_{ci}$ represents the estimated weights of the critic NN. By substituting (15) into the Hamiltonian function (12), the Hamilton–Jacobi–Issac (HJI) equation with optimal approximation becomes:

$$\begin{cases} 0 = (W_{cp}^{*\top} \nabla \psi_p + \nabla \epsilon_p^*)(f + g\mathscr{U}_p + k\mathscr{U}_e) \\ \qquad + |x|_\omega^\alpha + \Lambda_p(\mathscr{U}_p) - \Lambda_e(\mathscr{U}_e) \\ 0 = -(W_{ce}^{*\top} \nabla \psi_e + \nabla \epsilon_e^*)(f + g\mathscr{U}_p + k\mathscr{U}_e) \\ \qquad - |x|_\omega^\alpha + \Lambda_p(\mathscr{U}_p) - \Lambda_e(\mathscr{U}_e) \end{cases} \tag{16}$$

where $\Lambda_{ik}(\mathscr{U}_k)$ denotes the penalty of the k-th control input to i-th player. To estimate the critic NN weights, we design an actor NN to approximate the optimal controller:

$$\hat{\Xi}_i(x) = \hat{W}_{ai}^\top \phi_i(x), \ i = p, e \tag{17}$$

where $\phi_i(x)$ represents the basis function of the actor NN, and $\hat{W}_{ai}$ denotes the actor NN weights. The actor NN satisfies the transformed value function derivative

condition: $\nabla^2 \Xi_i = \hat{W}_{ai}^\top \nabla^2 \phi_i(x) \geq 0, \forall x \in \Omega_n$, as stated in Definition 3. The Hamilton function (12) can then be expressed as the HJI equation residual error $\delta_{\mathscr{H}}$:

$$\begin{aligned}\delta_{\mathscr{H}_p} =& \left\{(\hat{W}_{cp} - W_{cp}^*)^\top \nabla \psi_p - \nabla \epsilon_p^*\right\} \\ & \times (f + g\mathscr{U}_p + k\mathscr{U}_e) \qquad (18)\\ \delta_{\mathscr{H}_e} =& -\left\{(\hat{W}_{ce} - W_{ce}^*)^\top \nabla \psi_e + \nabla \epsilon_e^*\right\} \\ & \times (f + g\mathscr{U}_p + k\mathscr{U}_e) \qquad (19)\end{aligned}$$

To achieve the FT stability and convergence of the critic NN (15) and the actor NN (17), and inspired by [47,48] a squared loss function is defined with a historical stack of the state and residual error:

$$\begin{aligned}E_i =& \left\{\mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i})\right\}^\top \mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i}) \\ &+ \frac{1}{M}\sum_{k=1}^{M}\left\{\mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i}^k)\right\}^\top \mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i}^k),\ i = p, e\end{aligned} \qquad (20)$$

where $\Delta_{\mathscr{H}_i}^k$ is the integral of the k-th historical residual error (16) of the i-th player. Compared with regular concurrent learning laws [31,49], a finite-time concurrent learning law is employed to ensure finite-time convergence of the critic NN:

$$\begin{aligned}\dot{\hat{W}}_{ci} =& -k_{i,c1}\frac{\psi_i\,\mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i})}{(\psi_i^\top\psi_i + 1)^2} \\ & - \frac{k_{i,c2}}{M}\sum_{k=1}^{M}\frac{\psi_i^k\,\mathrm{sig}^\alpha(\Delta_{\mathscr{H}_i}^k)}{(\psi_i^{k\top}\psi_i^k + 1)^2},\ i = p, e\end{aligned} \qquad (21)$$

where $k_{i,c1}, k_{i,c2}$ are positive learning rates, $\alpha \in (0, 1)$ enables finite-time convergence, and $\{\psi_i^k, \Delta_{\mathscr{H}_i}^k\}_{k=1}^M$ contains historical data for concurrent learning. To ensure the convergence of the critic NN weights $\hat{W}_{ci}$, the persistent excitation condition (PE) is assumed in Assumption 3.1. To acquire the actor NN weights $\hat{W}_{ai}$ of the FT optimal controller, the transformation (11) is employed to map the value function to the FT convergence space. The weights of the actor NN are estimated using:

$$\hat{W}_{ai} = \left\{\int_{\Omega_n}\nabla\phi_i\nabla\phi_i^\top \mathrm{d}x\right\}^\dagger \left\{\int_{\Omega_n}\nabla\phi_i\,\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\hat{\mathscr{V}}_i\right)\mathrm{d}x\right\} \qquad (22)$$

where $i = p, e, \dagger$ denotes the Moore–Penrose pseudo-inverse, and $\int_{\Omega_n}\cdot\,\mathrm{d}x$ represents the Lebesgue integral [37].

Remark 1 [Lebesgue Integral in Finite-Time Value Function]

The Lebesgue integral is utilized for two essential purposes in our framework: (1) It enables rigorous transformation of the asymptotic value function $\mathscr{V}_i(x)$ into finite-time stable space, ensuring measure-theoretic completeness critical for establishing finite-time convergence properties. (2) It provides the mathematical foundation for mapping the value function into the finite-time convergence space, which is necessary for accurate estimation of actor NN weights $\hat{W}_{ai}$ within well-defined finite-time stable regions. This choice of integral formulation is made for maintaining theoretical guarantees while enabling practical implementation.

By substituting the actor NN weights (22) into the optimal controller (13), we obtain:

$$\begin{cases}\hat{\mathscr{U}}_p^*(x) = -\mu_p \tanh\left(\dfrac{R_p^{-1}g^\top}{2\mu_p}\,\mathrm{sig}^{\frac{\alpha}{2}}(\nabla\phi_p^\top\hat{W}_{ap})\right)\\ \hat{\mathscr{U}}_e^*(x) = \mu_e \tanh\left(\dfrac{R_e^{-1}k^\top}{2\mu_e}\,\mathrm{sig}^{\frac{\alpha}{2}}(\nabla\phi_e^\top\hat{W}_{ae})\right)\end{cases} \qquad (23)$$

For the actor NNs, to maintain bounded weights during updates while ensuring finite-time convergence, and inspired by literatures [28,44], a finite-time gradient projection law is employed:

$$\dot{\hat{W}}_{ai} = \mathrm{Proj}\left\{-k_{ai}F_i\,\mathrm{sig}^\alpha\left(\hat{W}_{ai} - \hat{W}_{ci}\right)\right\},\quad i = p, e \qquad (24)$$

where $k_{ai} > 0$ are learning rates, $F_i \in \mathbb{R}^{n_{\varphi_i}\times n_{\varphi_i}}$ are positive definite matrices, and $\mathrm{Proj}(\cdot)$ is a projection operator ensuring the weights remain within specified bounds [28].

Remark 2 [Finite-Time Update of Actor and Critic Networks in FT-RL-PEG]

The proposed FT-RL-PEG scheme builds upon adaptive dynamic programming and reinforcement learning (ADP&RL) frameworks [47,48], while making several key improvements. Unlike traditional ADP methods that only guarantee asymptotic convergence

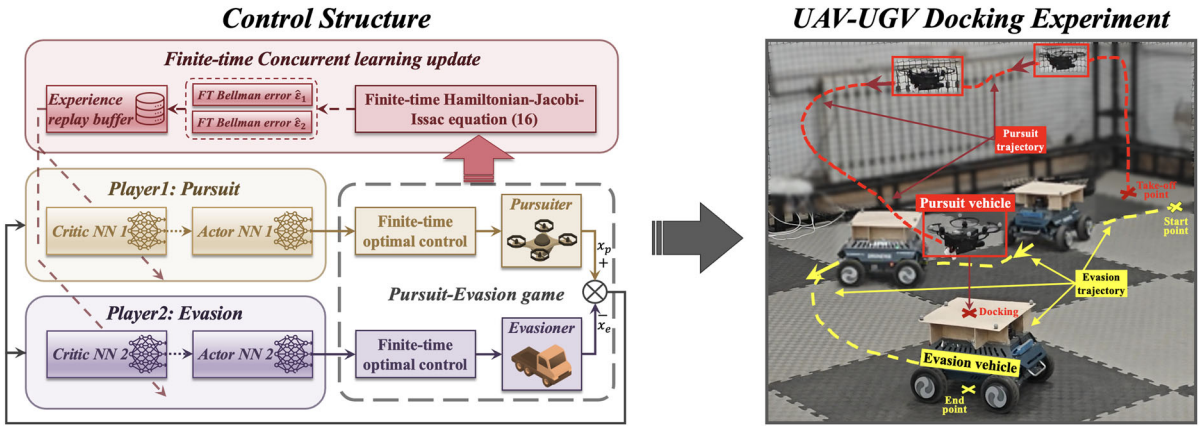


Fig. 1 Structure of the FT-RL-PEG scheme and its corresponding docking control process

[31,49], our finite-time concurrent learning approach in (21) maintains data efficiency while providing guaranteed convergence times through the $\text{sig}^{\alpha}(\cdot)$ terms. Furthermore, the projection operator in (24) ensures actor weights remain bounded during updates while maintaining finite-time convergence properties, addressing stability issues in traditional ADP implementations [28,44]. These improvements enable reliable real-time implementation while maintaining theoretical guarantees on convergence and stability.

After completing the online learning process of the FT-RL-PEG scheme, the overall structure is illustrated in Fig. 1, where the right side depicts the docking control process of the UAV–UGV system, and the left side shows the FT-RL-PEG scheme, which consists of four main components:

1. The pursuit-evasion game (PEG) dynamics (2), represented by the grey dashed box, which models the interaction between the UAV and UGV.
2. The finite-time actor-critic neural network for the pursuit player (FT-AC-NN-P), shown in the yellow box, which learns optimal pursuit strategies with finite-time convergence guarantees.
3. The finite-time actor-critic neural network for the evasion player (FT-AC-NN-E), depicted in the purple box, which learns optimal evasion policies within finite time.
4. The finite-time concurrent learning update laws (21) and (24), highlighted in the red box, which ensure finite-time adaptation of both neural networks.

The following assumptions are made to ensure the convergence of the FT-RL-PEG algorithm.

Assumption 3.1 [Persistent Excitation Condition [37, 50]] The historical data collected online for weight updates satisfies the following conditions:

$$\begin{cases} \int_t^{t+T} \frac{\psi_i(\tau)\psi_i(\tau)^\top}{\left(\psi_i^\top \psi_i+1\right)^2} \mathrm{d}\tau \geq \vartheta_{1i} I_{\mathscr{L}}, \\ \frac{1}{M}\sum_{k=1}^{M} \int_t^{t+T} \frac{\psi_i^k(\tau)\psi_i^k(\tau)^\top}{\left(\psi_i^{k\top} \psi_i^{k+1}\right)^2} \mathrm{d}\tau \geq \vartheta_{2i} I_{\mathscr{L}}, \\ \int_t^{t+T} \psi_i^{\dagger}(\tau)\, \text{sig}^{\alpha}\left(\psi_i(\tau)\right)^\top \mathrm{d}\tau \geq \vartheta_{3i} I_{\mathscr{L}}, \\ \frac{1}{M}\sum_{k=1}^{M} \int_t^{t+T} \psi_i^k(\tau)^{\dagger}\, \text{sig}^{\alpha}\left(\psi_i^k(\tau)\right)^\top \mathrm{d}\tau \geq \vartheta_{4i} I_{\mathscr{L}}, \end{cases}$$

where $i \in \{1, \ldots, N\}$, and at least one constant ϑ_i $(i = 1, 2, 3, 4)$ is strictly positive.

Assumption 3.2 (Boundedness of NN Weights and Basis Functions) For all system states $x \in \Omega_n$, there exist positive constants $W_H, \psi_H, \phi_H, \psi_{DH}, \phi_{DH}, \epsilon_H$ and ϵ_{DH} such that:

1. The neural network weights are bounded:
$$\|\hat{W}_{ci}\| \leq W_H, \quad \|\hat{W}_{ai}\| \leq W_H$$
2. The basis functions and their gradients satisfy:
$$\|\psi_i(x)\| \leq \psi_H, \quad \|\phi_i(x)\| \leq \phi_H$$
$$\|\nabla\psi_i(x)\| \leq \psi_{DH}, \quad \|\nabla\phi_i(x)\| \leq \phi_{DH}$$
3. The approximation errors are bounded:
$$\|\epsilon_i(x)\| \leq \epsilon_H, \quad \|\nabla\epsilon_i(x)\| \leq \epsilon_{DH}$$

The detailed implementation procedure of the FT-RL-PEG framework is presented in Algorithm 1. This algorithm describes the initialization, parameter settings, and iterative learning process to achieve optimal pursuit-evasion strategies.

Algorithm 1 FT-RL-PEG Algorithm for UAV–UGV Docking Control

Initialize:

- Actor-Critic NN weights: $\{\hat{W}_{ci}, \hat{W}_{ai}\}_{i=p,e}$ randomly with small values
- Historical data buffers: Empty state and control trajectories
- System states: Initial positions $\{x_e(0), x_p(0)\}$

Parameters:

- Learning rates: $k_{i,c1}, k_{i,c2}, k_{ai}$ for NN updates
- Weight matrices: F_i, R_i, Q_i for control design
- Control bounds: μ_i for input saturation
- Thresholds: T_{end} for convergence, ϵ for error

while $\|x_p - x_e\| > T_{\text{end}}$ **and** $t < T_{\max}$ **do**

// State measurement and processing

Measure current states: $\{x_e(t), x_p(t)\}$

Calculate tracking error: $x(t) = x_p(t) - x_e(t)$

Process obstacle information: $x_o(t)$ if present

// Optimal control computation

Update basis functions: $\{\phi_i(x), \psi_i(x)\}_{i=p,e}$ from (32)

Calculate pursuit control: $\hat{u}_p(t)$ from (23)

Calculate evasion control: $\hat{u}_e(t)$ from (23)

// Learning error calculation

Compute current Hamiltonian residuals:

- Pursuit error: $\delta_p(t)$ from (19)
- Evasion error: $\delta_e(t)$ from (18)

Update historical data: $\{\delta_i^k(t), \psi_i^k(t)\}_{k=1}^M$

// Neural network adaptation

Update critic weights via (21):

- $\dot{\hat{W}}_{ci} = -k_{i,c1} \frac{\psi_i \operatorname{sig}^\alpha(\Delta_{\mathscr{H}_i})}{(\psi_i^\top \psi_i + 1)^2} - \frac{k_{i,c2}}{M} \sum_{k=1}^M \frac{\psi_i^k \operatorname{sig}^\alpha(\Delta_{\mathscr{H}_i}^k)}{(\psi_i^{k\top} \psi_i^k + 1)^2}$

Update actor weights via (24):

- $\dot{\hat{W}}_{ai} = \text{Proj}\{-k_{ai} F_i \operatorname{sig}^\alpha(\hat{W}_{ai} - \hat{W}_{ci})\}$

Apply controls: $\{\hat{u}_p(t), \hat{u}_e(t)\}$ to vehicles

Update system states and time: $t = t + \Delta t$

end while

return Optimal control policies $\{\hat{u}_p^*, \hat{u}_e^*\}$

Remark 3 (Extension to Multi-Agent Scenarios) For multi-agent implementations, the FT-RL-PEG framework adopts a hierarchical structure where the main pursuit-evasion game decomposes into interconnected sub-games. This extension requires several key components: (1) a hierarchical control architecture combining localized decision-making with system-wide optimization, (2) adaptive coalition strategies enabling coordinated pursuit behaviors, (3) enhanced Nash equilibrium formulations suitable for multiple interacting agents, (4) robust communication protocols supporting distributed implementation, and (5) comprehensive collision avoidance mechanisms for safe multi-vehicle operation. Compared with traditional leader-follower paradigms that rely on fixed hierarchical structures [51,52], our game-theoretic approach enables dynamic competitive interactions between pursuers and evaders. This leads to emergent equilibrium behaviors that better reflect real-world pursuit-evasion scenarios. The framework maintains theoretical convergence guarantees while addressing practical challenges in multi-agent coordination and stability.

Remark 4 (Comparison with DRL Methods) While deep reinforcement learning (DRL) methods like PPO, SAC and DroQ have shown impressive performance in various control tasks, our approach, as a derivation of classical adaptive control method—Adaptive Dynamic Programming and Reinforcement Learning (ADP&RL) [47,48], differs fundamentally in its theoretical foundations and guarantees. Unlike traditional ADP&RL methods that provide only asymptotic convergence and handle constraints through reward shaping, our approach achieves provable finite-time convergence through Lyapunov functions and incorporates game-theoretic Nash equilibrium concepts. This theoretically grounded method enables both rapid learning and rigorous equilibrium guarantees while maintaining interpretability through Lyapunov analysis rather than relying on black-box deep neural networks, making it particularly suitable for safety-critical and performance-demanded UAV–UGV docking applications.

3.2 Finite-time stability analysis

In this subsection, we present the finite-time (FT) stability analysis for the system states and neural network parameters. We begin by analyzing the FT convergence properties of the optimal controller (13) with the following lemma.

Lemma 3.3 *(Finite-Time Stability of Optimal Controller* [40]*) Consider the nonlinear system* (2) *under the optimal control input* (13)*. The system states* $x(t)$ *converge to the equilibrium point in finite time.*

Define the Lyapunov candidate function:

$$\mathscr{L}_{V_i} = \frac{2\left|\nabla \Xi_i^*\right|^{\frac{\alpha}{2}+1}}{\alpha + 2} \tag{25}$$

The time derivative of $\mathscr{L}_{V_i}$ satisfies:

$$\begin{aligned}\dot{\mathscr{L}}_{V_i} &\leq -\frac{n\underline{\lambda}_{G_i}\underline{\lambda}_{K_i}}{4}\left|\nabla \Xi_i^*\right|^{\alpha} \\ &\leq -\frac{n\underline{\lambda}_{G_i}\underline{\lambda}_{K_i}}{4}\left(1+\frac{\alpha}{2}\right)^{\frac{2\alpha}{\alpha+2}} \mathscr{L}_{V_i}^{\frac{2\alpha}{\alpha+2}}\end{aligned} \tag{26}$$

where $\underline{\lambda}_{K_i} = \min_{1\leq j\leq n}\left\{\left|\nabla^2_{x_j}\Xi_i^*(x_j)\right|\right\}$, *and* $\underline{\lambda}_{G_i}$ *denotes the minimal eigenvalue of* $g_i R_{ii}^{-1} g_i^{\top}$.

Thus, the system converges to the equilibrium within the finite settling time:

$$T_{\mathscr{U}_i}[x(0)] = \frac{(\alpha+2)\left\{\mathscr{L}_{V_i}[x(0)]\right\}^{\frac{2-\alpha}{\alpha+2}}}{c_{\mathscr{U}_i}(2-\alpha)} \tag{27}$$

where $c_{\mathscr{U}_i} = \left(\frac{n\underline{\lambda}_{G_i}\underline{\lambda}_{K_i}}{4}\right)\left(1+\frac{\alpha}{2}\right)^{\frac{2\alpha}{\alpha+2}}$.

Lemma 3.4 *(Practical Finite-Time Stability* [37,45,46]*) Consider the system (2) with control input (13). Let $\mathscr{V}$ be a positive definite function on $\Omega_n \setminus \{x_0\}$ with $\mathscr{V}(0) = 0$. If, for all $x \in \Omega_n \setminus \{x_0\}$, the following inequality holds:*

$$\dot{\mathscr{V}} \leq -\gamma \mathscr{V}^{\alpha} + \delta_{\Gamma} \tag{28}$$

where $\gamma > 0$, $\alpha \in (0,1)$, and $\delta_{\Gamma} \in (0,\infty)$, then there exists a constant $\Gamma \in (0,1)$ such that:

1. *The system states converge to a region bounded by*

$$\mathscr{V} \leq \left(\frac{\delta_{\Gamma}}{(1-\Gamma)\gamma}\right)^{1/\alpha} \tag{29}$$

2. *The convergence time is bounded by*

$$T_{\Gamma}[x(0)] = \frac{\mathscr{V}(0)^{1-\alpha}}{\gamma\Gamma(1-\alpha)} \tag{30}$$

To analyze the stability properties of the proposed framework, we present the following key theoretical results concerning the finite-time convergence of the neural network weights and system states.

Theorem 3.5 *(Finite-Time Convergence of Neural Networks and System States) Under Assumptions 3.1 and 3.2, for the system (2) with the learning algorithm defined by Eqs. (21), (24), and (20), the following results hold:*

1. *The critic neural network weights $\hat{W}_{ci}$ converge to their optimal values W_{ci}^* in finite time.*
2. *The transformed value functions $\mathscr{V}_i$, as defined in (11), achieve finite-time convergence.*
3. *The Stackelberg equilibrium is attained in finite time.*

Proof See Appendix 1. □

The next theorem establishes finite-time convergence guarantees for the actor networks and closed-loop system.

Theorem 3.6 *[Finite-Time Stability of Actor Networks and System States] For system (2) under the proposed FT-RL-PEG learning framework, the following convergence properties hold:*

1. *The approximate pursuit policy $\hat{u}_p$ from (23) converge to optimal policies u_p^* in finite time*
2. *The approximate evasion policy $\hat{u}_e$ from (23) converge to optimal policies u_e^* in finite time*
3. *The closed-loop system states x achieve finite-time stability under the approximate optimal control*

Proof The detailed proof is provided in Appendix 1. □

4 Numerical simulations

In this section, two numerical simulation examples are conducted to verify the effectiveness of the proposed FT-RL-PEG scheme. The first example is performed using a representative nonlinear pursuit-evasion game scenario. The second example is conducted in the scenario of a pursuit aerial vehicle docking with a evasion ground vehicle.

4.1 Example 1: Nonlinear pursuit-evasion system

4.1.1 Simulation setup

In this example, a representative nonlinear pursuit-evasion system is designed to demonstrate docking control in 2D space. The dynamic models of both the pursuer and evader in system (1) are selected as nonlinear affine systems with parameters from [12,28]:

$$f_i(x_i) = \begin{bmatrix} -x_{i,1} + x_{i,2} \\ -\frac{1}{2}x_{i,1} - \frac{1}{2}x_{i,2}\left(1-\left(\cos\left(2x_{i,1}\right)+2\right)^2\right) \end{bmatrix},$$

$$g_i(x_i) = \begin{bmatrix} \sin\left(2x_{i,1}\right)+2 & 0 \\ 0 & \cos\left(2x_{i,1}\right)+2 \end{bmatrix},\ i = p, e.$$

Table 1 Parameters of the FT-RL-PEG scheme

Example	Initial conditions	Controller parameters	Learning parameters	Weights update
Example 1:	$x_e(0) = [2, 3]^T$	$R_e = I_2,\ Q_e = I_3$	$\hat{W}_{ce} = \text{rand}(3, 1) + 0.1$	$k_{e,c1} = 0.5,\ k_{e,c2} = 0.1$
Nonlinear		$\mu_e = 0.5,\ \alpha = 0.8$	$\hat{W}_{ae} = \text{rand}(3, 1) + 0.1$	$k_{e,a} = 1,\ F_e = I_3$
Pursuit-Evasion	$x_p(0) = [2.5, 2.5]^T$	$R_p = I_2,\ Q_p = 20I_2$	$\hat{W}_{ce} = \text{rand}(3, 1) + 0.5$	$k_{p,c1} = 0.5,\ k_{p,c2} = 0.1$
System		$\mu_p = 0.5, \alpha = 0.8$	$\hat{W}_{ae} = \text{rand}(3, 1) + 0.5$	$k_{p,a} = 1,\ F_p = I_6$
Example 2:	$x_e(0) = [2, 3]^T$	$R_e = 100I_2,\ Q_e = I_2$	$\hat{W}_{ce} = \text{rand}(3, 1) + 0.1$	$k_{e,c1} = 0.5,\ k_{e,c2} = 0.1$
UAV–UGV		$\mu_e = 0.75,\ \alpha = 0.8$	$\hat{W}_{ae} = \text{rand}(3, 1) + 0.1$	$k_{e,a} = 1,\ F_e = I_3$
Docking	$x_p(0) = [3, 2, 1]^T$	$R_p = 50I_3,\ Q_p = 20I_3$	$\hat{W}_{ce} = \text{rand}(7, 1) + 0.5$	$k_{p,c1} = 0.5,\ k_{p,c2} = 0.1$
System		$\mu_p = 1,\ \alpha = 0.8$	$\hat{W}_{ae} = \text{rand}(7, 1) + 0.5$	$k_{p,a} = 1,\ F_p = I_6$

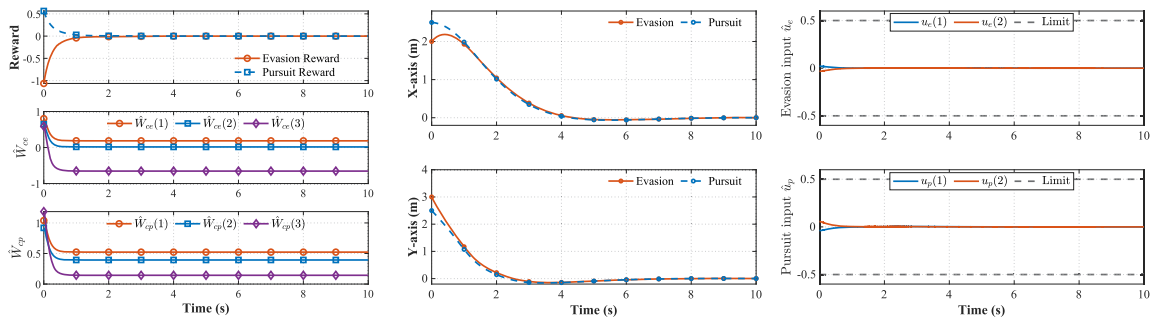


Fig. 2 Example 1: **a** Revolution of rewards and NNs weights. **b** Position of pursuiter-evasioner. **c** Control inputs to pursuiter-evasioner

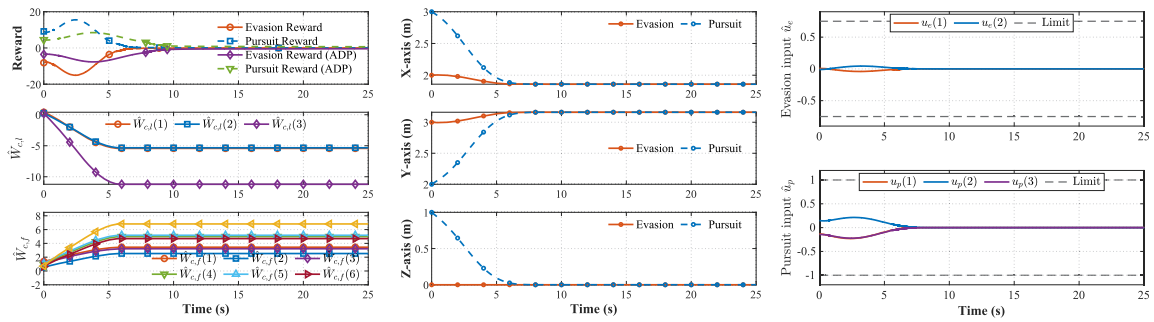


Fig. 3 Example 2: **a** Revolution of rewards and NNs weights. **b** Position of pursuiter-evasioner. **c** Control inputs to pursuiter-evasioner

For implementing the FT-RL-PEG scheme, actor-critic neural networks are employed with the following specifications:

- Network Architecture: Both pursuer and evader use 3-dimensional networks ($n_{\varphi_e} = n_{\varphi_p} = 3$)
- Initial Weights: Randomly initialized with small offsets
  - Evader: $\hat{W}_{e,c} = \hat{W}_{e,a} = \text{rand}(3, 1) + 0.5$
  - Pursuer: $\hat{W}_{p,c} = \hat{W}_{p,a} = \text{rand}(3, 1) + 0.5$
- Basis Functions: Fractional power form

$$\varphi_i = \frac{1}{\alpha + 1}\left[x_1^{\alpha+1}, x_2^{\alpha+1}, x_3^{\alpha+1}\right]^\top \tag{31}$$

where x_i denotes the i-th state component

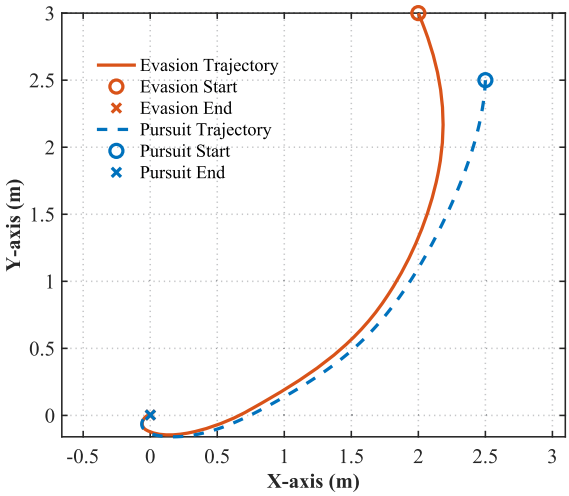


Fig. 4 Example 1: Trajectory of the pursuit-evasion in 2D space

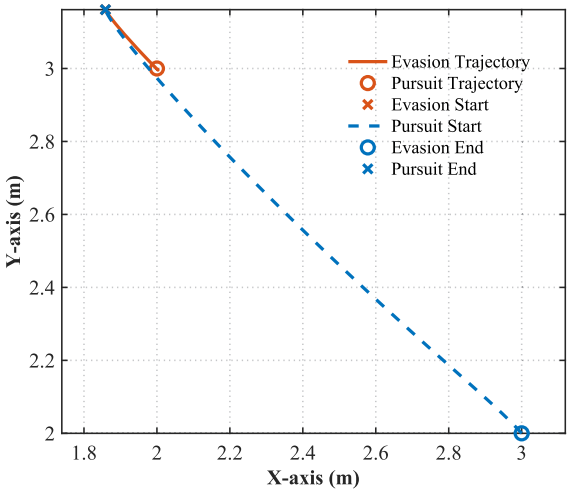


Fig. 5 Example 2: Trajectory of the pursuit-evasion in 2D space

The neural network weights are updated online using the concurrent learning law (21) and gradient projection law (24). Detailed control parameters are provided in Table 1. The numerical simulations are implemented in MATLAB R2023b Simulink with the following specifications:

- Hardware: Intel Core i3-12100F (3.3GHz), 24GB RAM
- Solver: Fourth-order Runge–Kutta method
- Time Step: 0.001s fixed
- Simulation Duration: 10 s

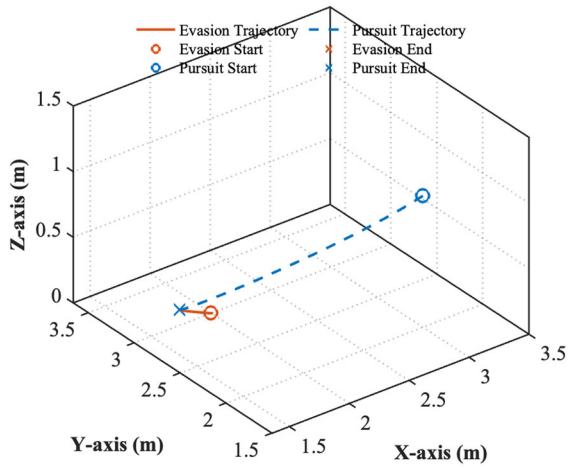


Fig. 6 Example 2: Trajectory of the pursuit-evasion in 3D space

4.1.2 Simulation result

The results of Example 1 are presented in Figs. 2 and 4. The performance analysis is shown in Fig. 2:

- Figure 2a demonstrates the convergence of rewards and actor-critic neural network weights, validating finite-time learning convergence
- Figure 2b displays the position states of both vehicles, showing effective tracking performance with bounded tracking errors
- Figure 2c illustrates that all control inputs remain within their prescribed saturation bounds while achieving optimal pursuit-evasion strategies

The trajectory visualization in Fig. 3 and 4 shows the pursuit-evasion paths in 2D space. The results validate that the proposed FT-RL-PEG framework successfully achieves optimal docking control with guaranteed finite-time convergence.

4.2 Example 2: UAV–UGV docking system

In this example, we design a pursuit-evasion system to accomplish an aerial-ground vehicle docking task. The UGV (evader) operates in the X-Y plane while the UAV (pursuer) maneuvers in X-Y-Z 3D space.

4.2.1 Simulation setup

The dynamic models for both vehicles are given by:

$$f_e = \begin{bmatrix} -0.1x_{e,1} + 0.05x_{e,2} \\ -0.05x_{e,1} - 0.1x_{e,2} \end{bmatrix},$$
$$g_e = I_{2\times 2},\ x_e \in \mathbb{R}^2,\ u_e \in \mathbb{R}^2$$
$$f_p = \begin{bmatrix} -0.1x_{p,1} + 0.05x_{p,2} \\ -0.05x_{p,1} - 0.1x_{p,2} \\ -0.1x_{p,3} \end{bmatrix},$$
$$g_p = I_{3\times 3},\ x_p \in \mathbb{R}^3,\ u_p \in \mathbb{R}^3$$

which represents a damped kinematic model that accounts for air resistance and ground friction effects. This model extends standard position-velocity kinematics [12,28] by incorporating realistic damping terms.

Remark 5 (Model Selection Rationale)
The simplified UAV and UGV models balance tractability with practical relevance. For the UGV in 2D space, the damped kinematic model f_e includes friction terms, while the input matrix $g_e = I_{2\times 2}$ enables planar motion control. For the UAV in 3D space, the model f_p adds aerial damping, with input matrix $g_p = I_{3\times 3}$ enabling spatial control. These models incorporate key dynamics while enabling efficient controller design. Additionally, the models for simulation example 2 were carefully chosen to validate the algorithm's performance before hardware implementation, while maintaining practical relevance. The experimental results in the later section confirm that this simplified yet representative modeling approach achieves reliable real-world performance, striking an effective balance between theoretical analysis and practical implementation.

The evader UGV aims to navigate efficiently while the pursuer UAV must track and ultimately dock with the UGV. For the evader UGV, we employ finite-time neural networks identical to Example 1. For the pursuer UAV, we design enhanced neural network basis functions:

$$\varphi_i = \frac{1}{\alpha+1}\left[x_1^{\alpha+1}, x_2^{\alpha+1}, x_3^{\alpha+1}, (x_1x_2)^{\alpha+1}, (x_1x_3)^{\alpha+1}, (x_2x_3)^{\alpha+1}, (x_1x_2x_3)^{\alpha+1}\right]^\top \quad (32)$$

where x_i represents the i-th element of the state vector x. This expanded basis set captures both polynomial and trigonometric state interactions necessary for precise 3D tracking control. The inclusion of sinusoidal terms improves the network's ability to approximate periodic motion patterns typical in aerial vehicle maneuvers.

Table 2 Hardware platform specifications

Component	Specifications
UGV (Evader)	4-wheel drive, RK3566 CPU 4GB RAM, PX4 Autopilot
UAV (Pursuer)	X150 quadcopter, RK3566 CPU 4GB RAM, PX4 Autopilot
Motion capture	OptiTrack system with 8 cameras 120 Hz sampling frequency
Control computer	Intel i7-12700 (3.60 GHz) 32GB RAM, 30 Hz control rate
Graphics station	Intel i7-8650u (1.90 GHz) 16GB RAM, Intel UHD 620

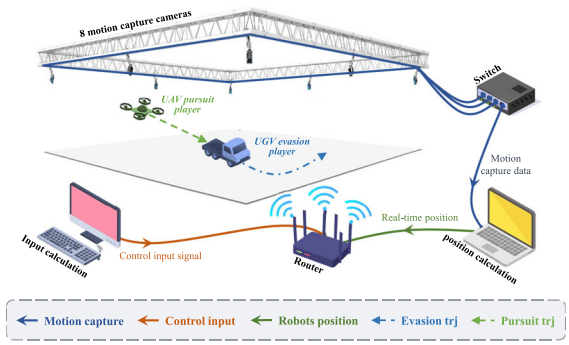


Fig. 7 Sketch of the setup for the hardware experiments

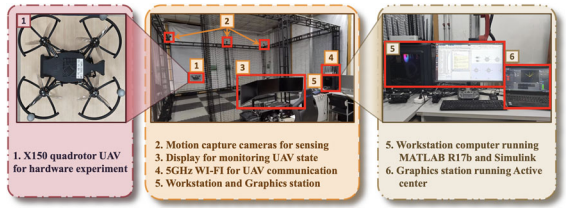


Fig. 8 Detailed hardware platform for the experiments

Table 3 Experiment parameters for hardware platform

Parameter	Notation	Value
Control frequency	f	30 Hz
Fixed time step	Δt	1/30 s
Gravitational acceleration	g	9.8 m/s^2
Vehicle mass	m_e	15.0 kg
Quadcopter mass	m_p	310 g
Vehicle input saturation	μ_e	0.75 m/s
Quadcopter input saturation	μ_p	1.0 m/s

4.2.2 Simulation result

The results of example 2 are presented in Fig. 3 and Figs. 5 and 6. The performance analysis is shown in Fig. 3:

- Figure 3a demonstrates the convergence of rewards and actor-critic neural network weights, validating finite-time learning convergence
- Figure 3b displays the position states of both vehicles, showing effective tracking performance with bounded tracking errors
- Figure 3c illustrates that all control inputs remain within their prescribed saturation bounds while achieving optimal pursuit-evasion strategies

In Fig. 3a, the baseline adaptive dynamic programming (ADP) method derived from [31,48] is included for comparison, It shows that the ADP method exhibits slower convergence compared to the proposed FT-RL-PEG scheme, in which the FT-RL-PEG scheme converges to the optimal solution within 8 s, while the ADP method does not converge to a similar range within the simulation duration of 25 s. The 2D and 3D trajectories of the pursuit-evasion system are shown in Figs. 5 and 6, respectively. The results validate that the proposed FT-RL-PEG framework successfully accomplishes the aerial-ground vehicle docking task, with the pursuer UAV effectively tracking and landing on the evader UGV.

These results confirm that the FT-RL-PEG framework successfully coordinates the UAV–UGV system to achieve optimal pursuit-evasion performance with guaranteed finite-time convergence properties.

5 Hardware experiments

In this section, hardware experiments are conducted to validate the effectiveness of the proposed FT-RL-PEG scheme.

5.1 Experiment setup

The experimental setup consists of a hardware platform for validating the FT-RL-PEG framework and a motion capture system for state measurement. The detailed hardware specifications are provided in Table 2, Figs. 7 and 8. The experiment scenarios are designed to validate:

- The evader UGV's ability to evade the pursuer UAV.
- The pursuer UAV's capability to accurately track and land on the moving UGV
- Real-time performance of the FT-RL-PEG algorithm on physical hardware

Three experimental cases are conducted:

1. **Case 1:** Basic pursuit-evasion scenario to validate convergence and stability
2. **Case 2:** Complex pursuit-evasion scenario with large initial offsets.
3. **Case 3:** Unsafe pursuit-evasion scenario with complex maneuvers and obstacle avoidance

For the third case, the pursuer UAV must navigate around a stationary obstacle while tracking and landing on the moving UGV. The reward function is modified to penalize collisions and reward obstacle avoidance in the following form:

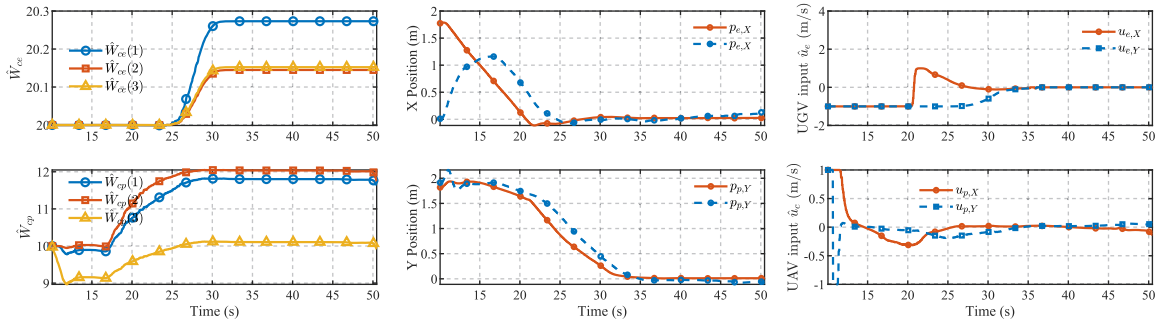
$$r = -\left(x^\top \omega x\right)^\alpha - \Lambda_p(\mathscr{U}_p) + \Lambda_e(\mathscr{U}_e) + \mathscr{B}(x) \quad (33)$$

where $\mathscr{B}(x)$ is a continuous function that penalizes collisions with the obstacle, which is modeled as a sphere with a fixed radius and position in the 3D space from literature [28]. The detailed experimental parameters are provided in Table 3.

5.2 Experiment results

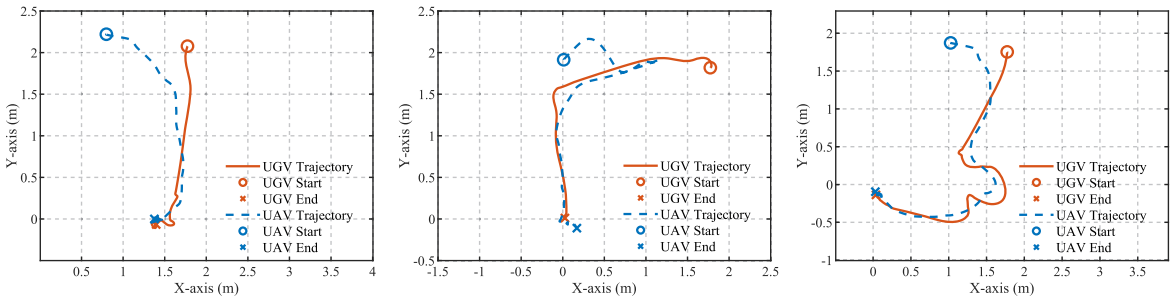
Case 1—Standard 2D pursuit-evasion:

The experimental results validate the effectiveness of the FT-RL-PEG framework. Key findings include: First, Fig. 9a demonstrates rapid convergence of neural

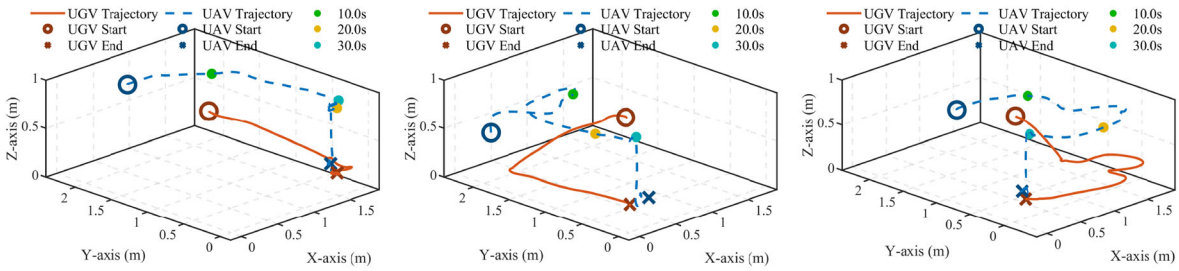


(a) Revolution of actor-critic weights (b) Positions of the UGV-UAV (c) Control inputs of the UGV-UAV

Fig. 9 Case 1: **a** Evolution of actor-critic neural network weights. **b** Position states of both vehicles. **c** Control inputs for pursuit-evasion



(a) Standard 2D pursuit-evasion (b) Large initial offsets case (c) Obstacle avoidance case



(d) Standard 3D tracking and landing (e) Complex 3D maneuvers (f) 3D obstacle avoidance

Fig. 10 2D and 3D trajectories for three experimental cases demonstrating the FT-RL-PEG framework

network weights, with both pursuer and evader networks reaching steady state by $t = 30$ s. The negligible variations in critic parameters $\hat{W}_{ci}$ thereafter confirm successful finite-time learning convergence, even with modeling uncertainties. Second, Fig. 9b shows the position trajectories converging to the center point with minimal tracking errors, validating that the optimal pursuit-evasion strategy effectively guides both vehicles to the desired docking location. Third, Fig. 9c verifies that all control inputs remain within their saturation bounds μ_i from (5), ensuring safe operation throughout. Finally, Fig. 10a and d confirm the framework's capability in handling complex maneuvers. These results validate the finite-time convergence and effectiveness of the FT-RL-PEG scheme under standard conditions .

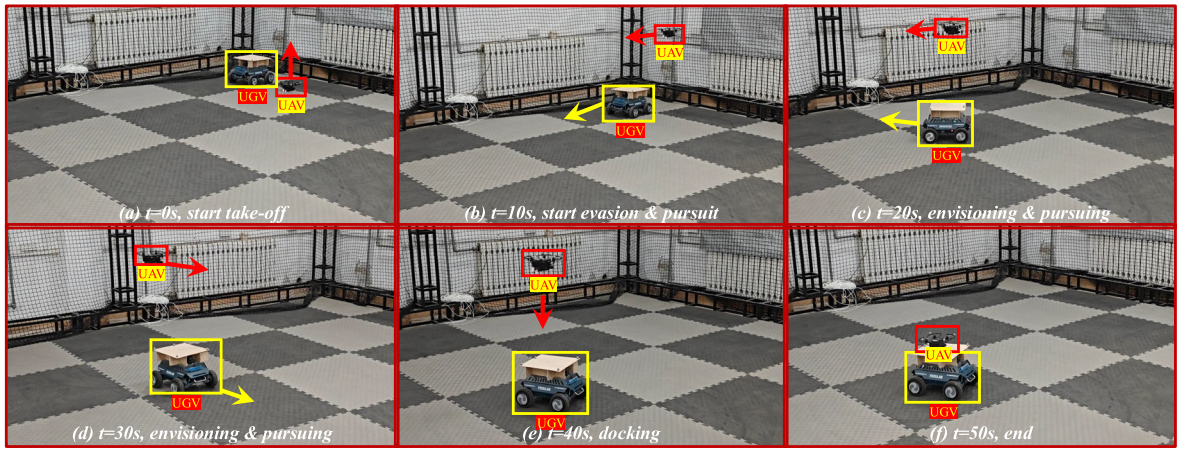


Fig. 11 Experiment: snapshots of the pursuit-evasion system of UAV–UGV docking process of case 3

Case 2 & 3—Large initial offsets & bstacle avoidance:

To demonstrate the FT-RL-PEG framework's robustness in handling complex docking scenarios, case 2 & 3 are conducted, in which case 2 is set up with larger initial position offsets (up to 2 metrics) to test robustness to spatial deviations, while case 3 introduces a spherical obstacle that both vehicles must avoid during the docking maneuver. Figure 10a–c show the 2D trajectories for:

- Standard tracking scenario validating baseline performance with minimal tracking errors
- Testing with large initial offsets (2x baseline) demonstrates robust convergence despite significant spatial deviations
- Obstacle avoidance case demonstrating safe navigation while avoiding a spherical obstacle during pursuit and docking

The 3D trajectories in Fig. 10d–f confirm similar performance in complex spatial environments. Figure 11 captures key snapshots of the docking process for case 3, highlighting the precise coordination between vehicles during landing. These results validate that the FT-RL-PEG scheme effectively enables autonomous aerial-ground docking while maintaining safety constraints.

6 Conclusion

This paper has developed a finite-time reinforcement learning with pursuit-evasion game (FT-RL-PEG) approach for unmanned aerial-ground vehicle docking control. The FT-RL-PEG framework combines finite-time learning with pursuit-evasion games to optimize docking performance. A novel actor-critic neural network architecture was designed for approximating optimal policies, utilizing finite-time concurrent learning laws for online weight adaptation. Rigorous stability analysis established theoretical guarantees on finite-time convergence properties. Comprehensive experimental validation demonstrated the framework's effectiveness in coordinating aerial-ground vehicle systems. Results confirm that FT-RL-PEG enables optimal path planning and tracking while maintaining safe docking capabilities.

Key limitations of the current approach encompass:

1. **Platform Scalability:** Framework currently addresses single pursuer-evader pairs rather than multi-vehicle scenarios.
2. **Disturbance Handling:** Performance sensitivity to environmental factors including wind effects and sensor uncertainties.
3. **Resource Efficiency:** Computational overhead requiring further optimization for resource-constrained systems.
4. **System Modeling:** Reliance on simplified vehicle dynamics and static obstacle representations.
5. **State Estimation:** Dependence on high-quality state measurements for control implementation.

Important directions for future research include extension to multi-vehicle scenarios and enhanced robustness.

Funding This research is supported by National Natural Science Foundation of China (No. U21A20485), International (Regional) Cooperation and Exchange (ICE) Projects of the National Natural Science Foundation of China (NSFC) (No. 62311540022), and China Postdoctoral Science Foundation (Project ID: 2024M762602).

Data availibility The datasets generated during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Appendix: Proof of Theorem 3.5

Proof To analyze the finite-time convergence and stability properties, we introduce a composite Lyapunov candidate function incorporating both critic network weight estimation errors and value function approximation errors:

$$\begin{aligned} &V^i(t,x,\{\hat{\mathscr{V}}_i\}_{i=1}^N,\{\hat{W}_{ci}\}_{i=1}^N) \\ &=\frac{1}{\alpha+1}\sum_{i=1}^{N}\left(|\hat{\mathscr{V}}_i-\mathscr{V}_i^*|^{\alpha+1}+|\hat{W}_{ci}-W_{ci}^*|^{\alpha+1}\right) \\ &=\sum_{i=1}^{N}\left(V_1^i+V_2^i\right)=\sum_{i=1}^{N}V^i \end{aligned} \tag{34}$$

where $\alpha\in(0,1)$ is the fractional power enabling finite-time convergence, and:

- Value function error: $V_1^i=\frac{|\hat{\mathscr{V}}_i-\mathscr{V}_i^*|^{\alpha+1}}{\alpha+1}$
- Weight estimation error: $V_2^i=\frac{|\hat{W}_{ci}-W_{ci}^*|^{\alpha+1}}{\alpha+1}$

Here $\hat{\mathscr{V}}_i$ and $\mathscr{V}_i^*$ denote the approximate and optimal value functions, while $\hat{W}_{ci}$ and W_{ci}^* represent the estimated and optimal critic network weights.

Taking the time derivative of (34) along the concurrent learning law (21):

$$\begin{aligned} \dot{V}&=\sum_{i=1}^{N}\left\{\mathrm{sig}^{\alpha}(\hat{\mathscr{V}}_i-\mathscr{V}_i^*)^{\top}\dot{\hat{\mathscr{V}}}_i+\mathrm{sig}^{\alpha}(\hat{W}_{ci}-W_{ci}^*)^{\top}\dot{\hat{W}}_{ci}\right\} \\ &=\sum_{i=1}^{N}\left\{\mathrm{sig}^{\alpha}(\mathscr{V}_i^*-\hat{\mathscr{V}}_i)^{\top}\psi_i^{\top}+\mathrm{sig}^{\alpha}(W_{ci}^*-\hat{W}_{ci})^{\top}\right\} \\ &\quad\times\left\{\frac{\alpha_1\psi_i\,\mathrm{sig}^{\alpha}(\Delta_{\mathscr{H}_i})}{(\psi_i^{\top}\psi_i+1)^2}+\frac{\alpha_2}{M}\sum_{k=1}^{M}\frac{\psi_i^k\,\mathrm{sig}^{\alpha}(\Delta_{\mathscr{H}_i}^k)}{(\psi_i^{k\top}\psi_i^k+1)^2}\right\} \end{aligned} \tag{35}$$

The Hamiltonian approximation error over $[t,t+T]$ is:

$$\begin{aligned} \Delta_{\mathscr{H}_i}&=\int_t^{t+T}\hat{\mathscr{H}}_i\left(\nabla\Xi_i,\mathscr{U}_p,\mathscr{U}_e,x\right)\mathrm{d}\tau \\ &=\int_t^{t+T}\left\{\hat{W}_{ci}^{\top}\nabla\psi_i\left(f+g\hat{\mathscr{U}}_p+k\hat{\mathscr{U}}_e\right)\right. \\ &\quad\left.+|x|_{\omega}^{\alpha}+\Lambda_i-\Lambda_k\right\}\mathrm{d}\tau \\ &\quad-\int_t^{t+T}\left\{\left(W_{ci}^{*\top}\nabla\psi_i+\nabla\epsilon_i^*\right)\left(f+g\mathscr{U}_p+k\mathscr{U}_e\right)\right. \\ &\quad\left.+|x|_{\omega}^{\alpha}+\Lambda_i-\Lambda_k\right\}\mathrm{d}\tau \\ &=\int_t^{t+T}\left(\hat{W}_{ci}-W_{ci}^*\right)^{\top}\nabla\psi_i\left(f+g\hat{u}+g\omega\right)\mathrm{d}\tau \\ &\quad-\int_t^{t+T}\nabla\epsilon_i^*\mathrm{d}\tau=\hat{\mathscr{V}}_i-\mathscr{V}_i^* \end{aligned} \tag{36}$$

Using the integral form of the Hamiltonian function (36) and Assumption 3.1, we can analyze the first term $\mathrm{sig}^{\alpha}(\hat{\mathscr{V}}_i-\mathscr{V}_i^*)^{\top}\dot{\hat{\mathscr{V}}}_i$ in (35). Its integral over time interval $[t,t+T]$ can be derived as:

$$\begin{aligned} &\int_t^{t+T}\mathrm{sig}^{\alpha}\left(\hat{\mathscr{V}}_i-\mathscr{V}_i^*\right)^{\top}\dot{\hat{\mathscr{V}}}_i\mathrm{d}\tau \\ &=-\int_t^{t+T}\mathrm{sig}^{\alpha}\left(\hat{\mathscr{V}}_i-\mathscr{V}_i^*\right)^{\top}\psi_i^{\top}\left\{\frac{\alpha_1\psi_i\,\mathrm{sig}^{\alpha}(\Delta_{\mathscr{H}_i})}{\left(\psi_i^{\top}\psi_i+1\right)^2}\right. \\ &\quad\left.+\frac{\alpha_2}{M}\sum_{k=1}^{M}\frac{\psi_i^k\,\mathrm{sig}^{\alpha}(\Delta_{\mathscr{H}_i}^k)}{\left(\psi_i^{k\top}\psi_i^k+1\right)^2}\right\}\mathrm{d}\tau \\ &\le-(\vartheta_{1i}+\vartheta_{2i})\,\mathrm{sig}^{\alpha}\left(\hat{\mathscr{V}}_i-\mathscr{V}_i^*\right)^{\top}I_{\mathscr{L}}\,\mathrm{sig}^{\alpha}\left(\hat{\mathscr{V}}_i-\mathscr{V}_i^*\right) \\ &\le-\vartheta_{5i}\left|\hat{\mathscr{V}}_i-\mathscr{V}_i^*\right|^{2\alpha} \end{aligned} \tag{37}$$

where $\vartheta_{5i}=\alpha_1\vartheta_{1i}+\alpha_2\vartheta_{2i}$ represents the composite learning rate. Therefore, the first term in (35) satisfies: $\mathrm{sig}^{\alpha}(\hat{\mathscr{V}}_i-\mathscr{V}_i^*)^{\top}\dot{\hat{\mathscr{V}}}_i\le-\vartheta_{4i}|\hat{\mathscr{V}}_i-\mathscr{V}_i^*|^{2\alpha}$, establishing a negative definite upper bound.

Similarly, for the second term $\text{sig}^{\alpha}(\hat{W}_{ci}-W_{ci}^{*})^{\top}\dot{\hat{W}}_{ci}$, we perform the following analysis. Integrating over $[t, t+T]$ yields:

$$\begin{aligned}\int_{t}^{t+T} &\text{sig}^{\alpha}\left(\hat{W}_{ci}-W_{ci}^{*}\right)^{\top}\dot{\hat{W}}_{ci}\mathrm{d}\tau \\ &=-\int_{t}^{t+T}\text{sig}^{\alpha}\left(\hat{W}_{ci}-W_{ci}^{*}\right)^{\top}\left\{\frac{\alpha_1\psi_i\,\text{sig}^{\alpha}(\Delta_{\mathscr{H}_i})}{\left(\psi_i^{\top}\psi_i+1\right)^2}\right. \\ &\quad\left.+\frac{\alpha_2}{M}\sum_{k=1}^{M}\frac{\psi_i^{k}\,\text{sig}^{\alpha}(\Delta_{\mathscr{H}_i}^{k})}{\left(\psi_i^{k\top}\psi_i^{k}+1\right)^2}\right\}\mathrm{d}\tau \end{aligned} \tag{38}$$

Applying Assumption 3.1 and the Cauchy-Schwarz inequality, we can bound this integral:

$$\begin{aligned}(38) \le &-\text{sig}^{\alpha}\left(\hat{W}_{ci}-W_{ci}^{*}\right)^{\top}\left\{\alpha_1\int_{t}^{t+T}\psi_i^{\dagger}\,\text{sig}^{\alpha}(\psi_i)^{\top}\mathrm{d}\tau\right. \\ &\left.+\frac{\alpha_2}{M}\sum_{k=1}^{M}\int_{t}^{t+T}\psi_i^{k\dagger}\,\text{sig}^{\alpha}\left(\psi_i^{k}\right)^{\top}\mathrm{d}\tau\right\}\text{sig}^{\alpha}\left(\hat{W}_{ci}-W_{ci}^{*}\right) \\ \le &-\vartheta_{6i}\left|\hat{W}_{ci}-W_{ci}^{*}\right|^{2\alpha}\end{aligned} \tag{39}$$

where $\vartheta_{6i}=\alpha_1\vartheta_{3i}+\alpha_2\vartheta_{4i}$ represents the composite learning rate for the critic weights. Combining (37) and (39), we obtain an upper bound for the integral of the Lyapunov function derivative (35):

$$\int_{t}^{t+T}\dot{V}^{i}\mathrm{d}\tau\le-\left\{\vartheta_{4i}\left|\hat{\mathscr{V}}_i-\mathscr{V}_i^{*}\right|^{2\alpha}+\vartheta_{6i}\left|\hat{W}_{ci}-W_{ci}^{*}\right|^{2\alpha}\right\} \tag{40}$$

This inequality establishes negative definiteness of the Lyapunov derivative integral, ensuring convergence of both the value function and critic weights. Based on inequality (40) and Lyapunov function (34), we can derive:

$$\dot{V}^{i}\le-\vartheta_{Vi}(V^{i})^{\frac{2\alpha}{\alpha+1}} \tag{41}$$

where the composite convergence rate ϑ_{Vi} is defined as:

$$\begin{aligned}\vartheta_{Vi}=\min\Big\{&\vartheta_{4i}(1+\alpha)^{\frac{2\alpha}{\alpha+1}}/T, \\ &\vartheta_{6i}(1+\alpha)^{\frac{2\alpha}{\alpha+1}}/T\Big\}\end{aligned} \tag{42}$$

The above inequality is obtained through the following steps:

1. Normalizing the learning rates by the time interval T:
   - $\vartheta_{4i}^{'}=\vartheta_{4i}/T$ for value function
   - $\vartheta_{6i}^{'}=\vartheta_{6i}/T$ for weights
2. Applying Jensen's inequality to each term:

$$\left|\hat{\mathscr{V}}_i-\mathscr{V}_i^{*}\right|^{2\alpha}\ge(1+\alpha)^{\frac{2\alpha}{\alpha+1}}(V_1^{i})^{\frac{2\alpha}{\alpha+1}}$$

$$\left|\hat{W}_{ci}-W_{ci}^{*}\right|^{2\alpha}\ge(1+\alpha)^{\frac{2\alpha}{\alpha+1}}(V_2^{i})^{\frac{2\alpha}{\alpha+1}}$$

3. Combining terms using the minimum convergence rate

Therefore, according to Lemma 3.4, there exists a finite settling time $T_{\hat{W}_{ci}}(V^{i})$ such that:

(1) The weights $\hat{W}_{ci}$ of the critic NN converge to optimal values W_{ci}^{*} within finite time:

$$\|\hat{W}_{ci}(t)-W_{ci}^{*}\|\le\epsilon,\quad\forall t\ge T_{\hat{W}_{ci}}(V^{i}) \tag{43}$$

for any small $\epsilon>0$.

(2) The value function approximation error is bounded by:

$$|V^{i}(t)|\le\delta_i,\quad\forall t\ge T_{\hat{W}_{ci}}(V^{i}) \tag{44}$$

for any arbitrary $\delta_i>0$.

The finite convergence time is explicitly given by:

$$T_{\hat{W}_{ci}}(V^{i})=\frac{(\alpha+1)\{V^{i}(x,x_0)\}^{\frac{1-2\alpha}{1+\alpha}}}{\vartheta_{Vi}(1-\alpha)} \tag{45}$$

Based on the above analysis, we can conclude:

(1) The critic neural networks achieve finite-time convergence to optimal weights within time $T_{\hat{W}_{ci}}(V^{i})$.
(2) The approximated value functions converge to optimal values within finite time with bounded error.
(3) The Nash equilibrium of the pursuit-evasion game is achieved within finite time:

$$T_{\hat{\mathscr{V}}}(V)=\max_{i=1,\ldots,N}\{T_{\hat{W}_{ci}}(V^{i})\} \tag{46}$$

This completes the proof of finite-time convergence for both the neural networks and Nash equilibrium. □

Appendix: Proof of Theorem 3.6

Proof First we establish a relationship between the optimal actor weights W_{ai}^{*} and optimal critic weights W_{ci}^{*} through Eq. (22):

$$W_{ai}^{*}=\left\{\int_{\Omega_n}\nabla\phi_i\nabla\phi_i^{\top}\mathrm{d}x\right\}^{\dagger}\left\{\int_{\Omega_n}\nabla\phi_i\,\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}W_{ci}^{*}\right)\mathrm{d}x\right\} \tag{47}$$

Let $\underline{\lambda}_{\phi_i}$ and $\bar{\lambda}_{\phi_i}$ denote the minimum and maximum eigenvalues of $\int_{\Omega_n}\nabla\phi_i\nabla\phi_i^{\top}\mathrm{d}x$ respectively. To analyze convergence, consider the difference between estimated actor weights $\hat{W}_{ai}$ from (22) and optimal weights W_{ai}^{*}:

$$\begin{aligned}&\|\hat{W}_{ai}-W_{ai}^{*}\|_2^2\\&=\left\|\left\{\int_{\Omega_n}\nabla\phi_i\nabla\phi_i^{\top}\mathrm{d}x\right\}^{\dagger}\times\int_{\Omega_n}\nabla\phi_i\right.\\&\quad\left.\left\{\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}\hat{W}_{ci}\right)-\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}W_{ci}^{*}\right)\right\}\mathrm{d}x\right\|_2^2\end{aligned} \tag{48}$$

By applying Cauchy-Schwarz inequality and properties of matrix norms:

$$\begin{aligned}\|\hat{W}_{ai}-W_{ai}^{*}\|_2^2&\le\frac{1}{\underline{\lambda}_{\phi_i}^2}\left\|\int_{\Omega_n}\nabla\phi_i\left\{\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}\hat{W}_{ci}\right)\right.\right.\\&\quad\left.\left.-\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}W_{ci}^{*}\right)\right\}\mathrm{d}x\right\|_2^2\\&\le\frac{\bar{\lambda}_{\phi_i}}{\underline{\lambda}_{\phi_i}^2}\int_{\Omega_n}\left\|\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}\hat{W}_{ci}\right)\right.\\&\quad\left.-\mathrm{sig}^{\frac{2}{\alpha}}\left(\nabla\psi_i^{\top}W_{ci}^{*}\right)\right\|_2^2\mathrm{d}x\\&\le\frac{\bar{\lambda}_{\phi_i}\bar{\lambda}_{\psi_i}^2}{\underline{\lambda}_{\phi_i}^2}\|\hat{W}_{ci}-W_{ci}^{*}\|_2^{\frac{4}{\alpha}}\end{aligned} \tag{49}$$

where $\bar{\lambda}_{\psi_i}$ denotes the maximum eigenvalue of $\int_{\Omega_n}\nabla\psi_i\nabla\psi_i^{\top}\mathrm{d}x$.

From Theorem 3.5, we know that the critic NN weights $\hat{W}_{ci}$ converge to the optimal weights W_{ci}^{*} in finite time, satisfying:

$$\|\hat{W}_{ci}-W_{ci}^{*}\|_2^{1+\alpha}\le(1+\alpha)\delta_i,\quad\forall\delta_i>0 \tag{50}$$

for all $t>T_{\hat{W}_{ci}}\left(V^i(x,0)\right)$, where $T_{\hat{W}_{ci}}\left(V^i(x,0)\right)$ is the convergence time given by (45).

Substituting (50) into (49) yields:

$$\|\hat{W}_{ai}-W_{ai}^{*}\|_2^2\le\frac{\bar{\lambda}_{\phi_i}\bar{\lambda}_{\psi_i}^2}{\underline{\lambda}_{\phi_i}^2}\{(1+\alpha)\delta_i\}^{\frac{4}{(1+\alpha)\alpha}},\quad\forall\delta_i>0 \tag{51}$$

This inequality establishes that the actor NN weights $\hat{W}_{ai}$ converge to the optimal weights W_{ai}^{*} in finite time as well. Then we analyze the convergence of the approximate optimal control input $\hat{\mathscr{U}}$. According to Lemma 3.3, the actor weights $\hat{W}_{ai}$ converge to optimal weights W_{ai}^{*} with settling time:

$$T_{u_i}[x(0)]=\frac{(\alpha+2)\left\{\mathscr{L}_{V_i}[x(0)]\right\}^{\frac{2-\alpha}{\alpha+2}}}{c_{L_i}(2-\alpha)} \tag{52}$$

where the Lyapunov function is:

$$\mathscr{L}_{V_i}(x,x_0)=\frac{2}{\alpha+2}\left|\nabla\phi_i^{\top}\hat{W}_{ai}(0)\right|^{\frac{\alpha}{2}+1} \tag{53}$$

Therefore, combining Theorem 3.5 with the above analysis, we can conclude:

(1) The approximate optimal control $\hat{\mathscr{U}}_i$ converges to optimal controls $\mathscr{U}_i^{*}$ within finite time:

$$T_{\hat{\mathscr{U}}_i}=\max\{T_{u_i}[x(0)],T_{\hat{W}_{ci}}(V^i(x,0))\} \tag{54}$$

(2) The actor NN weights $\hat{W}_{ai}$ achieve finite-time convergence to optimal weights W_{ai}^{*} with explicit settling time bounds. Next, we analyze the finite-time convergence of the system states x. With the actor-critic NNs approximating the optimal weights, the optimal control $\mathscr{U}_i^{*}$ can be expressed as:

$$\mathscr{U}_i^{*}=-\mu_i\tanh\left(\frac{1}{2\mu_i}R_i^{-1}g_i^{\top}\nabla\psi_i^{\top}W_{ci}^{*}\right) \tag{55}$$

The control error between optimal and approximate control inputs satisfies:

$$\|\mathscr{U}_i^{*}-\hat{\mathscr{U}}_i\|^2\le\Sigma_i\|\tilde{W}_{ci}\|^2+\Pi_{\mathscr{U}_i} \tag{56}$$

where Σ_i is bounded by the neural network approximation parameters $\varphi_H,\varphi_{DH},\sigma_H$ and σ_{DH}, $\Pi_{\mathscr{U}_i}$ represents the approximation error bound, and $\tilde{W}_{ci}=\hat{W}_{ci}-W_{ci}^{*}$ denotes the weight estimation error.

For the closed-loop system stability analysis, we derive the time derivative of the Lyapunov function $\mathscr{L}_V$ from (26):

$$\begin{aligned}\dot{\mathscr{L}}_{V_i}&=\mathrm{sig}^{\frac{\alpha}{2}}\left(\nabla\Xi_i^{*}\right)^{\top}\nabla^2\Xi_i^{*}\left\{f+g\hat{u}+g\omega^{*}+\sum_{i=1}^{N}g_i\left(\hat{\mathscr{U}}_i-\mathscr{U}_i^{*}\right)\right\}\\&\le n\alpha\left\{\mathrm{sig}^{\frac{\alpha}{2}}(\nabla\Xi^{*})^{T}\left\{f+g\hat{u}+g\omega^{*}+\sum_{i=1}^{N}g_i\left(\hat{\mathscr{U}}_i-\mathscr{U}_i^{*}\right)\right\}\right\}\\&\le n\alpha\left\{-|x|_{\omega}^{\alpha}-\mathscr{B}(x,x_o)-\sum_{k=1}^{N}\Lambda_{ik}(\mathscr{U}_k)\right.\\&\quad\left.+\left(\hat{\mathscr{U}}_i-\mathscr{U}_i^{*}\right)^{\top}R_{ii}\left(\hat{\mathscr{U}}-\mathscr{U}_i^{*}\right)\right\}\\&\le-c_{\mathscr{U}_i}\mathscr{L}_{V_i}^{\frac{2\alpha}{\alpha+2}}+\bar{\Pi}_{\mathscr{U}_i}\end{aligned} \tag{57}$$

where $\bar{\Pi}_{\mathscr{U}_i} = n\alpha_{\max}\bar{\lambda}_{R_i}\Pi_{\mathscr{U}_i}$ denotes the bounded perturbation term arising from approximation errors. This inequality establishes the practical finite-time stability of the closed-loop system states. By Lemma 3.4, with $0 < \gamma < 1$ being the contraction rate, the Lyapunov function will be bounded by:

$$\hat{\mathscr{L}}_{V_i} \leq \left\{ \frac{\bar{\Pi}_{\mathscr{U}_i}}{(1-\gamma)c_{\mathscr{U}_i}} \right\}^{\frac{\alpha+2}{2\alpha}} \tag{58}$$

where $c_{\mathscr{U}_i}$ is the coefficient from Lemma 3.3.

The finite-time convergence of the closed-loop system states is guaranteed with settling time:

$$T_x[x(0)] = \frac{\mathscr{L}_{V_i}[x(0)]^{1-\alpha}}{c_{\mathscr{U}_i}\gamma(1-\alpha)} \tag{59}$$

Therefore, combining the results from Theorem 3.5 and the above analysis, we can conclude:

1. The approximate optimal control inputs $\hat{\mathscr{U}}_i$ converge to the optimal controls $\mathscr{U}_i^*$ within finite time $T_{\hat{\mathscr{U}}_i}$
2. The closed-loop system states x achieve practical finite-time stability with explicit settling time $T_x[x(0)]$
3. The tracking error is ultimately bounded by a function of the approximation errors $\Pi_{\mathscr{U}_i}$

This completes the proof of finite-time convergence for both the optimal control approximation and closed-loop system stability.

□

References

1. Li, K., Li, Y.: Adaptive predefined-time optimal tracking control for underactuated autonomous underwater vehicles. IEEE/CAA J. Autom. Sinica **10**(4), 1083–1085 (2023). https://doi.org/10.1109/JAS.2023.123153
2. Yi, X., Liu, H., Wang, Y., Duan, H., Valavanis, K.P.: Safe reinforcement learning-based visual servoing control for quadrotors tracking unknown ground vehicles. IEEE Trans. Intell. Vehic. 1–11 (2024). https://doi.org/10.1109/TIV.2024.3464094
3. Cao, H., Shen, J., Liu, C., Zhu, B., Zhao, S.: Motion planning for aerial pick-and-place with geometric feasibility constraints. IEEE Trans. Autom. Sci. Eng. (2024). https://doi.org/10.1109/TASE.2024.3382296
4. Zheng, Z., Li, J., Guan, Z., Zuo, Z.: Constrained moving path following control for UAV with robust control barrier function. IEEE/CAA J. Autom. Sinica **10**(7), 1557–1570 (2023). https://doi.org/10.1109/JAS.2023.123573
5. Weiss, A., Baldwin, M., Erwin, R.S., Kolmanovsky, I.: Model predictive control for spacecraft rendezvous and docking: strategies for handling constraints and case studies. IEEE Trans. Control Syst. Technol. **23**(4), 1638–1647 (2015). https://doi.org/10.1109/TCST.2014.2379639
6. Xian, B., Wang, S., Yang, S.: Nonlinear adaptive control for an unmanned aerial payload transportation system: theory and experimental validation. Nonlinear Dyn. **98**(3), 1745–1760 (2019). https://doi.org/10.1007/s11071-019-05283-0
7. McEwen, R.S., Hobson, B.W., McBride, L., Bellingham, J.G.: Docking control system for a 54-cm-diameter (21-in) AUV. IEEE J. Oceanic Eng. **33**(4), 550–562 (2008). https://doi.org/10.1109/JOE.2008.2005348
8. Liu, Y., Wang, H., Fan, J.: Novel docking controller for autonomous aerial refueling with probe direct control and learning-based preview method. Aerosp. Sci. Technol. **94**, 105403 (2019). https://doi.org/10.1016/j.ast.2019.105403
9. Liu, K., Wang, R., Zheng, S., Dong, S., Sun, G.: Fixed-time disturbance observer-based robust fault-tolerant tracking control for uncertain quadrotor UAV subject to input delay. Nonlinear Dyn. **107**(3), 2363–2390 (2022). https://doi.org/10.1007/s11071-021-07080-0
10. Derrouaoui, S.H., Bouzid, Y., Belmouhoub, A., Guiatni, M.: Improved robust control of a new morphing quadrotor UAV subject to aerial configuration change. Unmanned Syst. (2023). https://doi.org/10.1142/S2301385025500116
11. Borowczyk, A., Nguyen, D.T., Nguyen, A.P.V., Nguyen, D.Q., Saussié, D., Le Ny, J.: Autonomous landing of a quadcopter on a high-speed ground vehicle. J. Guid. Control. Dyn. **40**(9), 2378–2385 (2017). https://doi.org/10.2514/1.G002703
12. Deptula, P., Chen, H.Y., Licitra, R.A., Rosenfeld, J.A., Dixon, W.E.: Approximate optimal motion planning to avoid unknown moving avoidance regions. IEEE Trans. Rob. **36**(2), 414–430 (2020). https://doi.org/10.1109/TRO.2019.2955321
13. Derrouaoui, S.H., Bouzid, Y., Doula, A., Boufroua, M.A., Belmouhoub, A., Guiatni, M., Hamissi, A.: Trajectory tracking control of a morphing UAV using radial basis function artificial neural network based fast terminal sliding mode: Theory and experimental. Aerosp. Sci. Technol. **155**, 109719 (2024). https://doi.org/10.1016/j.ast.2024.109719
14. Zhu, J., Zhu, J., Wang, Z., Guo, S., Xu, C.: Hierarchical decision and control for continuous multitarget problem: policy evaluation with action delay. IEEE Trans. Neural Netw. Learn. Syst. **30**(2), 464–473 (2019). https://doi.org/10.1109/TNNLS.2018.2844466
15. Ren, L., Jin, Y., Niu, Z., Wang, G., Yao, W., Zhang, X.: Optimal strategies for large-scale pursuers against one evader: a mean field game-based hierarchical control approach. Syst. Control Lett. **183**, 105697 (2024). https://doi.org/10.1016/j.sysconle.2023.105697
16. Li, M., Qin, J., Li, J., Liu, Q., Shi, Y., Kang, Y.: Game-based approximate optimal motion planning for safe human-swarm interaction. IEEE Trans. Cybern. (2023). https://doi.org/10.1109/TCYB.2023.3340659
17. Cai, H., Hu, G.: Distributed tracking control of an interconnected leader-follower multiagent system. IEEE Trans. Autom. Control **62**(7), 3494–3501 (2017). https://doi.org/10.1109/TAC.2017.2660298

18. Xue, L., Ye, J., Wu, Y., Liu, J., Wunsch, D.C.: Prescribed-time nash equilibrium seeking for pursuit-evasion game. IEEE/CAA J. Autom. Sinica **11**(6), 1518–1520 (2024). https://doi.org/10.1109/JAS.2023.124077
19. Zhang, Z.X., Zhang, K., Xie, X.P., Sun, J.Y.: Fixed-time zero-sum pursuit-evasion game control of multi-satellite via adaptive dynamic programming. IEEE Trans. Aerosp. Electron. Syst. (2024). https://doi.org/10.1109/TAES.2024.3351810
20. Kokolakis, N.M.T., Vamvoudakis, K.G.: Safety-aware pursuit-evasion games in unknown environments using gaussian processes and finite-time convergent reinforcement learning. IEEE Trans. Neural Netw. Learn. Syst. (2022). https://doi.org/10.1109/TNNLS.2022.3203977
21. Zhang, Y., Zhang, P., Wang, X., Song, F., Li, C., Hao, J.: An open loop Stackelberg solution to optimal strategy for UAV pursuit-evasion game. Aerosp. Sci. Technol. **129**, 107840 (2022). https://doi.org/10.1016/j.ast.2022.107840
22. Xie, T., Li, Y., Jiang, Y., Pang, S., Xu, X.: Three-dimensional mobile docking control method of an under-actuated autonomous underwater vehicle. Ocean Eng. **265**, 112634 (2022). https://doi.org/10.1016/j.oceaneng.2022.112634
23. Lu, S., Yang, H.: Reinforcement learning based reciprocal decision-making in multi-player pursuit-evasion differential games. Int. J. Robust Nonlinear Control (2024). https://doi.org/10.1002/rnc.7736
24. Tan, J., Xue, S., Cao, H., Li, H.: Nash equilibrium solution based on safety-guarding reinforcement learning in nonzero-sum game. In: 2023 International Conference on Advanced Robotics and Mechatronics (ICARM), pp. 630–635 (2023). https://doi.org/10.1109/ICARM58088.2023.10218910
25. Vamvoudakis, K.G., Fotiadis, F., Kanellopoulos, A., Kokolakis, N.M.T.: Nonequilibrium dynamical games: a control systems perspective. Annu. Rev. Control. **53**, 6–18 (2022). https://doi.org/10.1016/j.arcontrol.2022.03.006
26. Zhang, K., Zhang, Z.X., Xie, X.P., Rubio, J.d.J.: An unknown multiplayer nonzero-sum game: prescribed-time dynamic event-triggered control via adaptive dynamic programming. IEEE Trans. Autom. Sci. Eng. pp. 1–12 (2024). https://doi.org/10.1109/TASE.2024.3484412
27. Wang, P., Yu, C., Lv, M., Cao, J.: Adaptive fixed-time optimal formation control for uncertain nonlinear multiagent systems using reinforcement learning. IEEE Trans. Netw. Sci. Eng. **11**(2), 1729–1743 (2024). https://doi.org/10.1109/TNSE.2023.3330266
28. Wang, K., Mu, C., Ni, Z., Liu, D.: Safe reinforcement learning and adaptive optimal control with applications to obstacle avoidance problem. IEEE Trans. Autom. Sci. Eng. (2023). https://doi.org/10.1109/TASE.2023.3299275
29. Kamalapurkar, R., Rosenfeld, J.A., Dixon, W.E.: Efficient model-based reinforcement learning for approximate online optimal control. Automatica **74**, 247–258 (2016). https://doi.org/10.1016/j.automatica.2016.08.004
30. Ramírez, J., Yu, W., Perrusquía, A.: Model-free reinforcement learning from expert demonstrations: a survey. Artif. Intell. Rev. **55**(4), 3213–3241 (2022). https://doi.org/10.1007/s10462-021-10085-1
31. Tan, J., Xue, S., Guo, Z., Li, H., Cao, H., Chen, B.: Data-driven optimal shared control of unmanned aerial vehicles. Neurocomputing **622**, 129428 (2025). https://doi.org/10.1016/j.neucom.2025.129428
32. Dao, P.N., Nguyen, V.Q., Duc, H.A.N.: Nonlinear RISE based integral reinforcement learning algorithms for perturbed bilateral teleoperators with variable time delay. Neurocomputing **605**, 128355 (2024). https://doi.org/10.1016/j.neucom.2024.128355
33. Liu, H., Weng, P., Tian, X., Mai, Q.: Distributed adaptive fixed-time formation control for UAV-USV heterogeneous multi-agent systems. Ocean Eng. **267**, 113240 (2023). https://doi.org/10.1016/j.oceaneng.2022.113240
34. Tong, K., Li, M., Qin, J., Ma, Q., Zhang, J., Liu, Q.: Differential game-based control for nonlinear human-robot interaction system with unknown desired trajectory. IEEE Trans. Cybern. (2024). https://doi.org/10.1109/TCYB.2024.3402353
35. Tan, J., Xue, S., Cao, H., Li, H.: Safe Human-machine cooperative game with level-k rationality modeled human impact. In: 2023 IEEE International Conference on Development and Learning (ICDL), pp. 188–193 (2023). https://doi.org/10.1109/ICDL55364.2023.10364413
36. Yang, W., Cui, G., Ma, Q., Ma, J., Guo, S.: Finite-time adaptive optimal tracking control for a QUAV. Nonlinear Dyn. **111**(11), 10063–10076 (2023). https://doi.org/10.1007/s11071-023-08349-2
37. Zhang, L., Chen, Y.: Finite-time adaptive dynamic programming for affine-form nonlinear systems. IEEE Trans. Neural Netw. Learn. Syst. (2023). https://doi.org/10.1109/TNNLS.2023.3337387
38. Zhang, Z., Zhang, K., Xie, X., Stojanovic, V.: ADP-based prescribed-time control for nonlinear time-varying delay systems with uncertain parameters. IEEE Trans. Autom. Sci. Eng. (2024). https://doi.org/10.1109/TASE.2024.3389020
39. Huang, D., Huang, T., Qin, N., Li, Y., Yang, Y.: Finite-time control for a UAV system based on finite-time disturbance observer. Aerosp. Sci. Technol. **129**, 107825 (2022). https://doi.org/10.1016/j.ast.2022.107825
40. Haddad, W.M., L'Afflitto, A.: Finite-time stabilization and optimal feedback control. IEEE Trans. Autom. Control **61**(4), 1069–1074 (2016). https://doi.org/10.1109/TAC.2015.2454891
41. Zhang, L., Chen, Y.: Distributed finite-time ADP-based optimal control for nonlinear multiagent systems. IEEE Trans. Circuits Syst. II Express Briefs **70**(12), 4534–4538 (2023). https://doi.org/10.1109/TCSII.2023.3291399
42. Zhang, L., Chen, Y.: Distributed finite-time ADP-based optimal secure control for complex interconnected systems under topology attacks. IEEE Trans. Syst. Man Cybern. Syst. **54**(5), 2872–2883 (2024). https://doi.org/10.1109/TSMC.2024.3351909
43. Tan, J., Xue, S., Li, H., Cao, H., Li, D.: Safe stabilization control for interconnected virtual-real systems via model-based reinforcement learning. In: 2024 14th Asian Control Conference (ASCC), pp. 605–610 (2024)
44. Tan, J., Xue, S., Cao, H., Ge, S.S.: Human–AI interactive optimized shared control. J. Autom. Intell. p. S2949855425000024 (2025). https://doi.org/10.1016/j.jai.2025.01.001
45. Li, D., Ge, S., He, W., Ma, G., Xie, L.: Multilayer formation control of multi-agent systems. Automatica **109**, 108558 (2019). https://doi.org/10.1016/j.automatica.2019.108558

46. Li, D., Ge, S., Lee, T.: Simultaneous arrival to origin convergence: sliding-mode control through the norm-normalized sign function. IEEE Trans. Autom. Control **PP**, 1–1 (2021). https://doi.org/10.1109/TAC.2021.3069816
47. Lewis, F.L., Syrmos, V.L., Vrabie, D.L.: Optimal Control, 3rd edn. Wiley, Hoboken (2012)
48. Lewis, F.L., Vrabie, D.: Reinforcement learning and adaptive dynamic programming for feedback control. IEEE Circuits Syst. Mag. **9**(3), 32–50 (2009). https://doi.org/10.1109/MCAS.2009.933854
49. Mu, C., Wang, K., Xu, X., Sun, C.: Safe adaptive dynamic programming for multiplayer systems with static and moving no-entry regions. IEEE Trans. Artif. Intell. (2023). https://doi.org/10.1109/TAI.2023.3325780
50. Cohen, M.H., Belta, C.: Safe exploration in model-based reinforcement learning using control barrier functions. Automatica **147**, 110684 (2023). https://doi.org/10.1016/j.automatica.2022.110684
51. Nguyen, T.L., Mai, X.S., Dao, P.N.: A robust distributed model predictive control strategy for leader-follower formation control of multiple perturbed wheeled mobile robotics. Eur. J. Control. **81**, 101160 (2025). https://doi.org/10.1016/j.ejcon.2024.101160
52. Nguyen, K., Dang, V.T., Pham, D.D., Dao, P.N.: Formation control scheme with reinforcement learning strategy for a group of multiple surface vehicles. Int. J. Robust Nonlinear Control **34**(3), 2252–2279 (2024). https://doi.org/10.1002/rnc.7083