ENGINEERING Artificial Intelligence
https://doi.org/10.2738/ENGAI.2026.0004

Original Research

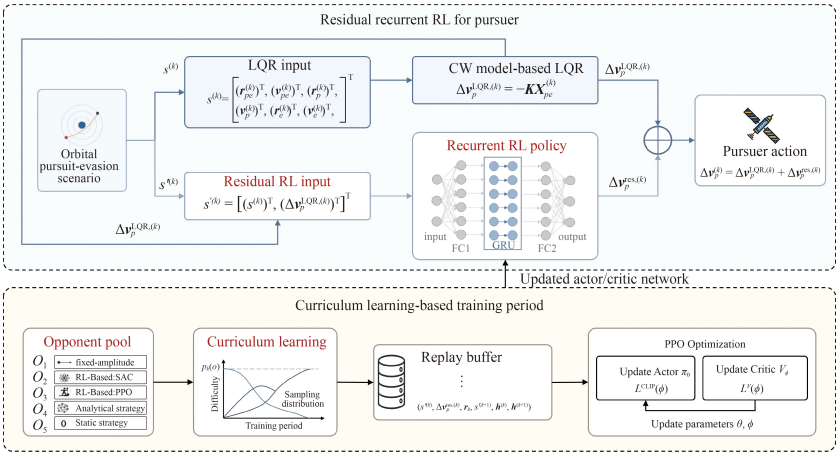
R^{3}LOOP: Residual Recurrent Reinforcement Learning for Cross-Opponent-Strategy Orbital Pursuit Decision

Wei Dong[1,2,3], **Jiajie Wang**[1], **Chen Chen**[1,2,3], **Fang Deng** (✉)[1,2,3,4], **Zhengtao Ding**[5]

1 School of AI, Beijing Institute of Technology, Beijing 100081, China
2 National Key Lab of Autonomous Intelligent Unmanned Systems, Beijing Institute of Technology, Beijing 100081, China
3 Beijing Key Laboratory of Lightweight Intelligent Systems, Beijing Institute of Technology, Beijing 100081, China
4 Shenzhen MSU-BIT University, Shenzhen 518172, China
5 Department of Electrical and Electronic Engineering, University of Manchester, Manchester M13 9PL, UK

© The Author(s) 2026. This article is published by Higher Education Press.

GRAPHICAL ABSTRACT



HIGHLIGHTS

- A cross-opponent orbital pursuit decision problem is studied by training a single pursuer policy against multiple evader strategies.
- An LQR-guided residual reinforcement learning structure learns corrections to nominal pursuit actions, reducing the search difficulty of pure reinforcement learning.
- A recurrent policy with multi-evader training uses historical interaction data to improve adaptability across different evader strategies.

ABSTRACT This paper proposes a residual recurrent reinforcement learning framework, i.e., R^{3}LOOP, for orbital pursuit decision-making with cross-opponent-strategy generalization. First, a linear quadratic regulator (LQR)-based nominal pursuit prior is constructed from the Clohessy–Wiltshire (CW) relative motion model to provide a baseline impulsive action for the pursuer. Then, a proximal policy optimization (PPO) policy is introduced to learn a residual correction to refine the LQR action, which reduces the search difficulty of pure reinforcement learning (RL) and improves the adaptability of the pursuer in adversarial interactions. Finally, to address the challenge of cross-opponent generalization, a gated recurrent unit (GRU)-based residual policy and an opponent-pool training mechanism are developed. Unlike fixed-opponent learning mechanisms, the proposed method aims to train a single pursuer policy against multiple evader strategies. This hybrid design combines the model-based pursuit prior with learning-based residual correction, enabling the learning policy to refine the nominal pursuit action without learning the policy from scratch. Numerical simulations are provided to show the effectiveness and advantages of the proposed framework.

KEYWORDS Orbital game; Recurrent reinforcement learning; Opponent pool; Cross-opponent-strategy generalization

Received 18 Jul 2026; accepted 8 Oct 2026

E-mail: dengfang@bit.edu.cn (Deng F.).

1 Introduction

Autonomous decision-making for non-cooperative spacecraft has become an important requirement for future space operations, such as on-orbit service, space situational awareness, active debris removal, and spacecraft safety management. With the increasing complexity of space missions, the interaction between the pursuer and the evader becomes more dynamic, adversarial, and uncertain. The orbital pursuit-evasion game is essentially a complex sequential decision-making problem. Designing a practical decision-making method with good adaptability and generalization is significant but challenging in this field [1–5].

For the orbital pursuit-evasion game, early studies mainly focused on classical game-theoretic and control methods [6–9]. The authors in Ref. [10] modeled the problem as a three-dimensional orbital pursuit-evasion game and systematically studied its numerical solution, which laid a foundation for later modeling and solution methods. In Ref. [11], the saddle-point solution of the orbital pursuit-evasion game under J_2-perturbed dynamics was investigated, and the classical game-theoretic framework was extended to perturbed orbital scenarios. The pursuit-evasion problem under incomplete information was studied in Ref. [12] and a switching-strategy method was proposed to improve the applicability in non-ideal information conditions. A parameter-optimized pursuit strategy was developed for orbital games in Ref. [13] where an effective pursuit decision framework was provided when the opponent information was unavailable or imperfect. In Ref. [14], J_2 perturbations and control constraints were introduced into the orbital pursuit-evasion formulation, making the model closer to practical engineering scenarios. On this basis, pursuit-evasion methods have gradually developed toward predictive control and hierarchical game frameworks. The authors in Ref. [15] combined predictive pursuit-evasion control with the model predictive control framework for approaching non-cooperative space targets. In Ref. [16], a model predictive Stackelberg solution was proposed for the orbital pursuit-evasion game to improve real-time decision-making capability. In Ref. [17], the impulsive orbital game was formulated in a Stackelberg equilibrium framework, which promoted the development of pursuit-evasion methods under hierarchical game structures. Notably, existing classical methods mostly rely on explicit and simplified orbital dynamics. The feasibility and adaptability under practical complex dynamics and unknown hostile behavior are challenging to ensure.

With the rapid development of deep learning and RL, intelligent decision-making methods have received increasing attention for orbital pursuit-evasion games [18–22]. In Ref. [23], deep RL was used to construct a near-optimal interception strategy, and the feasibility of learning-based closed-loop decision-making was verified in orbital pursuit-evasion scenarios. A deep-RL-based pursuit strategy was proposed in Ref. [24] for orbital interception against random evasion. The authors in Ref. [25] introduced a Q-learning method from the perspective of a nonzero-sum game to solve orbital pursuit-evasion control problems. A learning-based algorithm named PRD-MADDPG was developed in Ref. [26] for the orbital pursuit-evasion game with impulsive maneuvers. The authors in Ref. [27] extended RL-based decision-making to the orbital pursuit-evasion game in elliptical orbits. In Ref. [28], the multi-agent orbital pursuit-evasion problem under sparse rewards was investigated, and a hierarchical RL framework based on hindsight experience replay was proposed to improve training convergence and maneuver strategy learning. More recently, the two-player orbital pursuit-evasion problem under impulsive thrust control was studied in Ref. [29] and an impulse-like constraint was introduced to incorporate impulsive maneuvers into a differential-game framework. In Ref. [30], a continuous-time hierarchical RL framework was proposed for pursuit decision and its potential in long-horizon pursuit tasks was demonstrated. Although RL can learn a policy through interaction without an explicit model, training remains computationally expensive because of the large policy space and inefficient early exploration.

In recent years, researchers have increasingly combined model-based methods with RL to improve training efficiency [31]. Although this approach has received less attention in orbital pursuit-evasion games, it has shown promise in related fields such as robot control. For complex robot control tasks, the authors in Ref. [32] proposed residual policy learning, which learns a residual policy on top of an existing controller. This method improves learning efficiency and task performance by combining baseline control with learned correction. Similarly, in Ref. [33], conventional feedback control was directly combined with deep RL. The method used residual RL to compensate for contact effects, friction, and modeling errors and showed good performance in real robot control. Furthermore, the authors in Ref. [34] proposed a hybrid framework that first imitated the model predictive control and then used RL for fine-tuning. This work showed that a model-based controller can serve not only as a prior controller at deployment but also as an expert signal during training. In a recent work [35], a model-based whole-body controller was further integrated into an RL feedback policy to improve both accuracy and robustness. A similar hybrid design was

developed in [36] to address spacecraft proximity operations. Note that the above results are based on a specific scenario or a cooperative target. Existing hybrid methods generally refine a nominal controller for a specific task or fixed interaction setting. For the orbital pursuit-evasion game in adversarial scenarios, the design of an efficient and generalized intelligent decision-making method against an uncooperative target remains an open challenge.

Inspired by the above observations, this paper proposes R^{3}LOOP, an LQR-guided residual recurrent reinforcement learning framework for three-dimensional orbital pursuit-evasion. The main contributions of this paper are threefold:

(1) A cross-opponent orbital pursuit decision problem is studied. Different from fixed-opponent settings, this work trains a single pursuer policy against multiple evader strategies and evaluates its performance across different opponents.

(2) A prior-guided residual reinforcement learning structure is proposed. The LQR controller provides a model-based nominal pursuit action. The PPO policy learns a residual correction to refine this action. This design reduces the difficulty of pure RL search and improves the adaptability of the pursuer in adversarial interactions.

(3) A recurrent policy with multi-evader training is developed. The recurrent module uses historical interaction data to represent evader behavior. Multiple evader strategies are used during training. This improves the adaptability of one pursuer policy across different evader strategies.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation and mathematical model. Section 3 introduces the proposed decision-making framework for the orbital pursuit-evasion game. Section 4 provides numerical simulations and performance analysis. Finally, Section 5 concludes this paper.

2 Problem Formulation

This section introduces the mission scenario, dynamic model and constraints of the three-dimensional orbital pursuit-evasion problem.

2.1 Mission scenario and orbital dynamics

In this paper, a pursuit-evasion problem in the vicinity of the geostationary Earth orbit (GEO) is considered. Both the pursuer and the evader perform orbital control through impulsive maneuvers. To describe the pursuit-evasion process, a local-vertical local-horizontal (LVLH) coordinate frame is established. A notional reference spacecraft is assumed to move on a circular GEO, and the LVLH coordinate frame is attached to this reference spacecraft, whose center is taken as the origin. The x-axis is the radial direction and points from the Earth's center to the spacecraft, the y-axis is aligned with the velocity direction, and the z-axis completes a right-handed orthogonal coordinate system. In this frame, the states of the pursuer and evader are defined relative to the notional reference satellite. The configuration of the LVLH coordinate frame and the definition of the relative position vector are shown in Fig. 1.

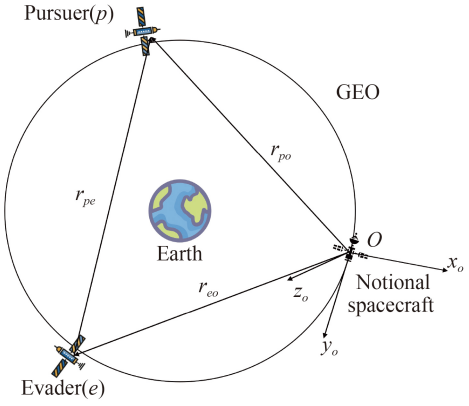


Fig. 1 The diagram of the relative position.

Similar to common practice [26,37], the relative distance between the pursuer and the evader is assumed to remain much smaller than the GEO orbital radius throughout the engagement. Therefore, the Clohessy–Wiltshire (CW) linearized model is adopted. For spacecraft $i \in \{p, e\}$, where p and e denote the pursuer and the evader, the relative motion equations are

$$\begin{aligned} &\ddot{x}_i - 2n\dot{y}_i - 3n^2 x_i = u_{x,i}, \\ &\ddot{y}_i + 2n\dot{x}_i = u_{y,i}, \\ &\ddot{z}_i + n^2 z_i = u_{z,i}, \end{aligned} \tag{1}$$

where x_i, y_i, and z_i are the position components of spacecraft i along the three axes of the LVLH frame, $u_{x,i}$, $u_{y,i}$, and $u_{z,i}$ are the corresponding continuous acceleration components, and n denotes the mean orbital angular velocity of the target spacecraft calculated by

$$n = \sqrt{\mu / R^3}, \tag{2}$$

where μ is the Earth's gravitational parameter and R is the GEO radius of the target spacecraft. For GEO, $R \approx 4.2164 \times 10^7$ m, yielding $n \approx 7.2921 \times 10^{-5}$ rad/s.

Define the state vector as

$$\boldsymbol{X}_i(t) = \begin{bmatrix} \boldsymbol{r}_i(t) \\ \boldsymbol{v}_i(t) \end{bmatrix} = \begin{bmatrix} x_i & y_i & z_i & \dot{x}_i & \dot{y}_i & \dot{z}_i \end{bmatrix}^{\mathrm{T}}, \tag{3}$$

where $\boldsymbol{X}_i(t)$, $\boldsymbol{r}_i(t)$, and $\boldsymbol{v}_i(t)$ denote the state, position, and

velocity of spacecraft i, respectively. Then, the CW dynamics can be written in the standard state-space form as

$$\dot{\boldsymbol{X}}_i(t) = \boldsymbol{A}\boldsymbol{X}_i(t) + \boldsymbol{B}\boldsymbol{u}_i(t), \tag{4}$$

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix}, \qquad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{0}_{3\times3} \\ \boldsymbol{I}_{3\times3} \end{bmatrix}, \tag{5}$$

with

$$\boldsymbol{A}_{21} = \begin{bmatrix} 3n^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -n^2 \end{bmatrix}, \quad \boldsymbol{A}_{22} = \begin{bmatrix} 0 & 2n & 0 \\ -2n & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{6}$$

and $\boldsymbol{u}_i(t) = [u_{x,i}(t), u_{y,i}(t), u_{z,i}(t)]^{\mathrm{T}}$ is the continuous control acceleration vector of spacecraft i.

Given the initial state $\boldsymbol{X}_i(t_0)$, the future state at any time t can be calculated by

$$\boldsymbol{X}_i(t) = \boldsymbol{\Phi}(t,t_0)\boldsymbol{X}_i(t_0) + \int_{t_0}^{t} \boldsymbol{\Phi}(t,\tau)\boldsymbol{B}\boldsymbol{u}_i(\tau)\mathrm{d}\tau \tag{7}$$

where $\boldsymbol{\Phi}(t,t_0)$ is the state transition matrix given by

$$\boldsymbol{\Phi}(t,t_0) = \begin{bmatrix} \boldsymbol{\Phi}_{rr}(t,t_0) & \boldsymbol{\Phi}_{rv}(t,t_0) \\ \boldsymbol{\Phi}_{vr}(t,t_0) & \boldsymbol{\Phi}_{vv}(t,t_0) \end{bmatrix}, \tag{8}$$

For the CW dynamics, the four submatrices of $\boldsymbol{\Phi}(t,t_0)$ are given by

$$\begin{aligned}
\boldsymbol{\Phi}_{rr}(t,t_0) &= \begin{bmatrix} 4-3\cos n(t-t_0) & 0 & 0 \\ 6(\sin n(t-t_0) - n(t-t_0)) & 1 & 0 \\ 0 & 0 & \cos n(t-t_0) \end{bmatrix}, \\
\boldsymbol{\Phi}_{rv}(t,t_0) &= \begin{bmatrix} \dfrac{\sin n(t-t_0)}{n} & \dfrac{2(1-\cos n(t-t_0))}{n} & 0 \\ \dfrac{2(\cos n(t-t_0)-1)}{n} & \dfrac{4\sin n(t-t_0) - 3n(t-t_0)}{n} & 0 \\ 0 & 0 & \dfrac{\sin n(t-t_0)}{n} \end{bmatrix}, \\
\boldsymbol{\Phi}_{vr}(t,t_0) &= \begin{bmatrix} 3n\sin n(t-t_0) & 0 & 0 \\ 6n(\cos n(t-t_0)-1) & 0 & 0 \\ 0 & 0 & -n\sin n(t-t_0) \end{bmatrix}, \\
\boldsymbol{\Phi}_{vv}(t,t_0) &= \begin{bmatrix} \cos n(t-t_0) & 2\sin n(t-t_0) & 0 \\ -2\sin n(t-t_0) & 4\cos n(t-t_0) - 3 & 0 \\ 0 & 0 & \cos n(t-t_0) \end{bmatrix}.
\end{aligned} \tag{9}$$

In this study, both pursuer and evader spacecraft use impulsive control to maneuver. When an impulsive velocity increment $\Delta\boldsymbol{v}_i$ is applied at time t_0, the post-impulse state is

$$\boldsymbol{X}_i(t_0^+) = \begin{bmatrix} \boldsymbol{r}_i(t_0) \\ \boldsymbol{v}_i(t_0) + \Delta\boldsymbol{v}_i \end{bmatrix}. \tag{10}$$

Then, the state at any later time t is

$$\boldsymbol{X}_i(t) = \boldsymbol{\Phi}(t,t_0)\boldsymbol{X}_i(t_0^+) = \boldsymbol{\Phi}(t,t_0)\boldsymbol{X}_i(t_0) + \boldsymbol{\Phi}_v(t,t_0)\Delta\boldsymbol{v}_i, \tag{11}$$

and the terminal state at the terminal time t_f is

$$\boldsymbol{X}_i(t_f) = \boldsymbol{\Phi}(t_f,t_0)\boldsymbol{X}_i(t_0) + \boldsymbol{\Phi}_v(t_f,t_0)\Delta\boldsymbol{v}_i, \tag{12}$$

where $\Delta\boldsymbol{v}_i$ denotes the impulsive velocity increment applied to spacecraft i at time t_0.

2.2 Constraints and termination conditions

Based on the above relative motion model, the orbital pursuit-evasion process can be formulated as a discrete-time sequential game using a fixed time step Δt. At the beginning of each step, the pursuer and the evader apply impulsive thrusts. Each episode contains at most N time steps, and the maximum pursuit time is

$$t_{\max} = N\Delta t. \tag{13}$$

The k-th decision instant is

$$t^{(k)} = k\Delta t. \tag{14}$$

After each impulse, the relative position and velocity of both agents propagate according to Eq. (11). The overall process of the pursuit-evasion game is shown in Fig. 2.

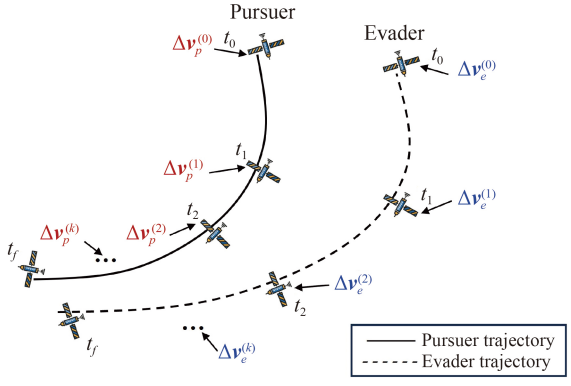


Fig. 2 The orbital pursuit-evasion process.

In the orbital pursuit-evasion game, both spacecraft are subject to the following constraints.

(1) Single-step per-axis impulsive magnitude constraint: at each time step k, the impulsive velocity increment of spacecraft i is denoted by $\Delta \boldsymbol{v}_i^{(k)}$. Define the per-axis impulse-bound vector as

$$\eta_i = \begin{bmatrix} \eta_{x,i} & \eta_{y,i} & \eta_{z,i} \end{bmatrix}^{\mathrm{T}}, \tag{15}$$

where $\eta_{q,i}$ is the impulse bound along axis $q \in \{x,y,z\}$ for spacecraft i, $i \in \{p,e\}$. The componentwise constraint is

$$|\Delta v_{q,i}^{(k)}| \leqslant \eta_{q,i}, \quad q \in \{x,y,z\}, i \in \{p,e\}. \tag{16}$$

(2) Cumulative velocity-increment constraint: the per-step maneuver cost and cumulative maneuver cost are defined as

$$\begin{aligned} c_i^{(k)} &= \|\Delta \boldsymbol{v}_i^{(k)}\|_1 = |\Delta v_{x,i}^{(k)}| + |\Delta v_{y,i}^{(k)}| + |\Delta v_{z,i}^{(k)}|. \\ C_i^{(k)} &= \sum_{\ell=0}^{k-1} c_i^{(\ell)}. \end{aligned} \tag{17}$$

Let $\bar{C}_i$ denote the maximum allowable cumulative maneuver cost. The constraint is

$$C_i^{(k)} \leqslant \bar{C}_i. \tag{18}$$

The cumulative maneuver cost $C_i^{(k)}$ is reported in m/s and is used as a maneuver-cost proxy rather than as a direct measure of propellant mass.

(3) Time constraint: the pursuit-evasion game is bounded by a finite time horizon as

$$t_f = k_f \Delta t \leqslant N \Delta t = t_{\max}. \tag{19}$$

Here, k_f denotes the number of decision intervals completed when the episode terminates, with $1 \leqslant k_f \leqslant N$.

In addition, the orbital pursuit-evasion game terminates if one of the following two conditions holds:

(1) Unsuccessful termination: the mission fails when the pursuer cannot capture the evader within the given constraints. This case includes:

(a) the maximum pursuit time is reached, that is

$$\begin{cases} t_f = t_{\max} \\ d^{(k_f)} > d_{\mathrm{cap}} \end{cases}. \tag{20}$$

(b) the pursuer violates its cumulative maneuver-cost budget, that is

$$C_p^{(k_f)} \geqslant \bar{C}_p. \tag{21}$$

(2) Successful termination: the mission succeeds when the pursuer captures the evader before the maximum time and without violating the cumulative maneuver-cost budget, that is

$$\begin{cases} d^{(k_f)} \leqslant d_{\mathrm{cap}} \\ t_f \leqslant t_{\max} \\ C_p^{(k_f)} \leqslant \bar{C}_p \end{cases}. \tag{22}$$

where $d^{(k)}$ denotes the relative distance between the pursuer and the evader, and d_{cap} denotes the preset capture threshold.

3 R^{3}LOOP for Orbital Pursuit Decision-Making

This section presents the proposed R^{3}LOOP framework for the three-dimensional orbital pursuit-evasion game. The flowchart of the R^{3}LOOP framework is shown in Fig. 3.

3.1 LQR-based baseline strategy

Note that a game-theoretic controller would require an explicit evader model or strategy assumption, which conflicts with the objective of adapting one pursuer to multiple evasion policies. Therefore, the LQR technique is used to derive an opponent-independent model-based prior, while opponent-dependent adaptation will be addressed by the residual policy in the next subsection.

For the nominal strategy design, the evader is assumed not to maneuver, i.e., $\Delta \boldsymbol{v}_e^{(k)} = \boldsymbol{0}$. Under this ideal condition, by Eq. (11), the discrete-time relative dynamics can be written as

$$\boldsymbol{X}_{pe}^{(k+1)} = \boldsymbol{A}_d \boldsymbol{X}_{pe}^{(k)} + \boldsymbol{B}_d \Delta \boldsymbol{v}_p^{(k)}, \tag{23}$$

where $\boldsymbol{X}_{pe}^{(k)} = \boldsymbol{X}_p^{(k)} - \boldsymbol{X}_e^{(k)}$ is the relative state at $t^{(k)}$, and $\Delta \boldsymbol{v}_p^{(k)}$ is the pursuer impulse. Note that the two terms on the right-hand side represent the free propagation of the relative state and the state increment induced by the pursuer impulse, respectively. The matrices $\boldsymbol{A}_d$ and $\boldsymbol{B}_d$ are constructed from the state-transition submatrices in Eq. (9) as

$$\begin{aligned} \boldsymbol{A}_d &= \begin{bmatrix} \boldsymbol{\Phi}_{rr}(t^{(k+1)}, t^{(k)}) & \boldsymbol{\Phi}_{rv}(t^{(k+1)}, t^{(k)}) \\ \boldsymbol{\Phi}_{vr}(t^{(k+1)}, t^{(k)}) & \boldsymbol{\Phi}_{vv}(t^{(k+1)}, t^{(k)}) \end{bmatrix} \\ \boldsymbol{B}_d &= \begin{bmatrix} \boldsymbol{\Phi}_{rv}(t^{(k+1)}, t^{(k)}) \\ \boldsymbol{\Phi}_{vv}(t^{(k+1)}, t^{(k)}) \end{bmatrix} \end{aligned}. \tag{24}$$

To obtain the baseline strategy, the quadratic performance index is defined as

$$J = \sum_{k=0}^{\infty} \left(\left(\boldsymbol{X}_{pe}^{(k)} \right)^{\mathrm{T}} \boldsymbol{Q} \boldsymbol{X}_{pe}^{(k)} + \left(\Delta \boldsymbol{v}_p^{(k)} \right)^{\mathrm{T}} \boldsymbol{R} \Delta \boldsymbol{v}_p^{(k)} \right), \tag{25}$$

where $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the state and control weighting matrices, respectively. The design objective is to minimize J subject to the discrete dynamics in Eq. (23).

By the LQR technique, the corresponding optimal value function is

$$V(\boldsymbol{X}_{pe}) = \boldsymbol{X}_{pe}^{\mathrm{T}} \boldsymbol{P} \boldsymbol{X}_{pe}, \tag{26}$$

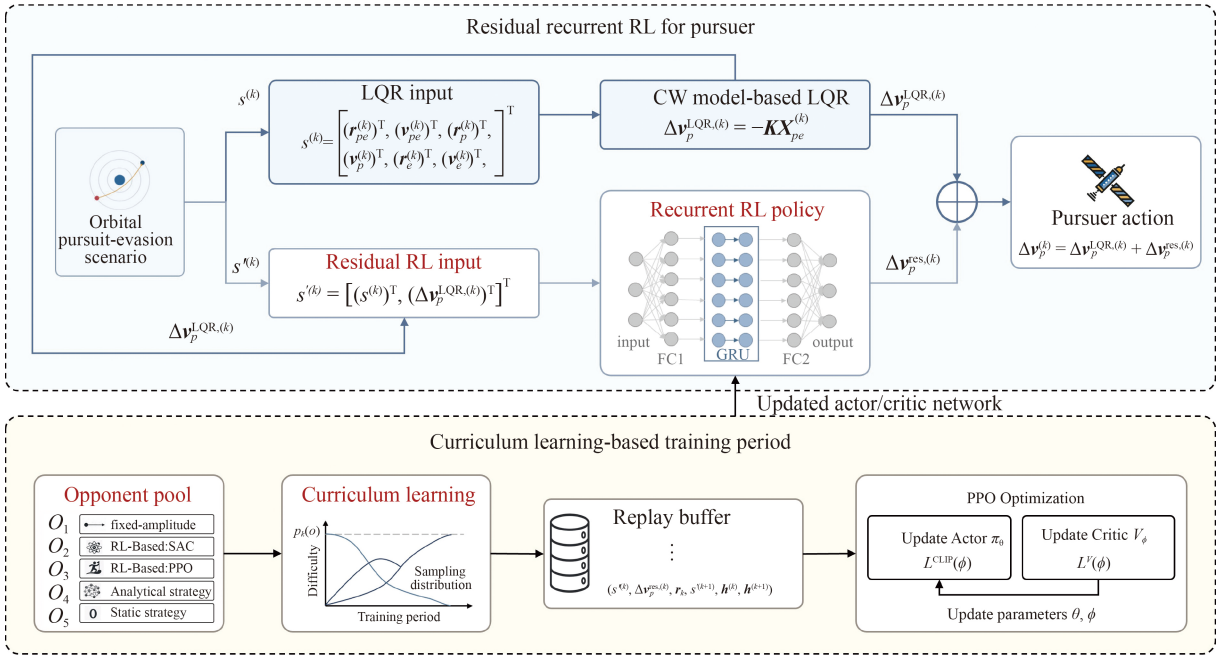


Fig. 3 Flowchart of the proposed R^{3}LOOP framework.

where $\boldsymbol{P}$ satisfies the following discrete algebraic Riccati equation

$$\boldsymbol{P}=\boldsymbol{Q}+\boldsymbol{A}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{A}_d-\boldsymbol{A}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{B}_d\left(\boldsymbol{R}+\boldsymbol{B}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{B}_d\right)^{-1}\boldsymbol{B}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{A}_d. \quad (27)$$

The resulting state-feedback law is

$$\Delta\boldsymbol{v}_p^{\mathrm{LQR},(k)}=-\boldsymbol{K}\boldsymbol{X}_{pe}^{(k)}. \quad (28)$$

where the feedback gain matrix is given by

$$\boldsymbol{K}=\left(\boldsymbol{R}+\boldsymbol{B}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{B}_d\right)^{-1}\boldsymbol{B}_d^{\mathrm{T}}\boldsymbol{P}\boldsymbol{A}_d. \quad (29)$$

The LQR baseline gives a model-based pursuit direction and reduces the search difficulty of the learning policy. In the actual implementation, the LQR command is further constrained by the single-step impulsive magnitude limit. Therefore, the LQR baseline should be understood as a nominal pursuit prior rather than a complete solution to the pursuit-evasion game.

3.2 Residual RL-based adaptive strategy

The nominal LQR designed in Subsection 3.1 is derived by assuming the evader remains non-maneuvering. However, during actual pursuit-evasion, the evader may actively perform unknown maneuvers. To handle such maneuvers, a PPO+LQR framework is introduced to correct the nominal LQR decision.

The environment state is explicitly defined as

$$\boldsymbol{s}^{(k)}=\begin{bmatrix}\boldsymbol{r}_{pe}^{(k)}\\ \boldsymbol{v}_{pe}^{(k)}\\ \boldsymbol{r}_{p}^{(k)}\\ \boldsymbol{v}_{p}^{(k)}\\ \boldsymbol{r}_{e}^{(k)}\\ \boldsymbol{v}_{e}^{(k)}\end{bmatrix}\in\mathbb{R}^{18}. \quad (30)$$

The LQR impulse is further appended to form the policy input as

$$\boldsymbol{s}'^{(k)}=\left[\left(\boldsymbol{s}^{(k)}\right)^{\mathrm{T}},\left(\Delta\boldsymbol{v}_p^{\mathrm{LQR},(k)}\right)^{\mathrm{T}}\right]^{\mathrm{T}}\in\mathbb{R}^{21}. \quad (31)$$

Before entering the network, the three position blocks, three velocity blocks, and LQR-impulse block are divided by fixed position, velocity, and impulse scales, respectively. Note that these scales should be appropriately chosen according to specific scenarios. Table 2 in Section 4 provides an exemplified case for the considered simulation settings.

The PPO algorithm adopts an actor-critic architecture and both the actor and the critic use standard feedforward neural networks. Based on the augmented input $\boldsymbol{s}'^{(k)}$, the actor outputs the mean of the residual action distribution, and the critic estimates the corresponding state value.

In the action representation, the policy outputs a residual correction. For the Gaussian policy, the actor outputs the action mean $\boldsymbol{\mu}^{(k)}$, and the policy distribution is defined as

$$\pi_\theta(\Delta\boldsymbol{v}_p^{\mathrm{res},(k)}\mid\boldsymbol{s}'^{(k)})=\mathcal{N}(\boldsymbol{\mu}^{(k)},\mathrm{diag}(\boldsymbol{\sigma}^2)). \quad (32)$$

The final control input applied to the pursuer is given

by

$$\Delta \boldsymbol{v}_p^{(k)} = \Delta \boldsymbol{v}_p^{\mathrm{LQR},(k)} + \Delta \boldsymbol{v}_p^{\mathrm{res},(k)}. \tag{33}$$

This structure keeps the LQR strategy and the PPO policy in the same impulsive control domain. The LQR strategy supplies the nominal pursuit action, while PPO learns an opponent-dependent residual correction.

During training, the policy is updated with the clipped PPO objective. Let the probability ratio between the new policy and the old policy be

$$\rho^{(k)}(\theta) = \frac{\pi_\theta(\Delta \boldsymbol{v}_p^{\mathrm{res},(k)} \mid \boldsymbol{s}'^{(k)})}{\pi_{\theta_{\mathrm{old}}}(\Delta \boldsymbol{v}_p^{\mathrm{res},(k)} \mid \boldsymbol{s}'^{(k)})}. \tag{34}$$

Then, the actor objective is

$$L^{\mathrm{CLIP}}(\theta) = \mathbb{E}_k\Big[\min\big(\rho^{(k)}(\theta)\hat{A}^{(k)},\ \mathrm{clip}(\rho^{(k)}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}^{(k)}\big)\Big]. \tag{35}$$

The advantage function is computed by generalized advantage estimation (GAE) as

$$\hat{A}^{(k)} = \sum_{l=0}^{N-k-1} (\gamma\lambda)^l \delta^{(k+l)}, \tag{36}$$

where $\hat{A}^{(k)}$ is the GAE advantage estimate at step k, and the temporal-difference error $\delta^{(k)}$ is

$$\delta^{(k)} = r^{(k)} + \gamma V_\phi(\boldsymbol{s}'^{(k+1)}) - V_\phi(\boldsymbol{s}'^{(k)}). \tag{37}$$

The critic is trained by minimizing the state value estimation error

$$L^V(\phi) = \big(V_\phi(\boldsymbol{s}'^{(k)}) - \hat{V}^{(k)}\big)^2, \tag{38}$$

where $\hat{V}^{(k)}$ denotes the value target at step k.

Based on the above design, PPO refines the LQR baseline action through residual correction, thereby improving the adaptability of the pursuer in dynamic adversarial interactions.

3.3 Residual recurrent RL-based cross-opponent strategy

Although the residual RL framework proposed in Subsection 3.2 can improve the baseline action, it mainly uses current-step information if a feedforward policy is adopted. This setting is not sufficient for cross-opponent tasks. Different opponents may show different maneuver patterns over time. Therefore, a residual recurrent policy is introduced to extract historical interaction information and improve generalization against multiple opponents.

The actor adopts a GRU-based network for temporal modeling. The same augmented state $\boldsymbol{s}'^{(k)}$ defined in Eq. (31) is used together with the GRU hidden state $\boldsymbol{h}^{(k)}$, which contains the historical interaction information available at decision step k. As such, the recurrent policy generates the residual correction using both the current orbital information and the accumulated interaction history. To this end, the input feature is first obtained by

$$\boldsymbol{f}^{(k)} = \sigma\big(\boldsymbol{W}_1 \boldsymbol{s}'^{(k)} + \boldsymbol{b}_1\big). \tag{39}$$

Then, the hidden state $\boldsymbol{h}^{(k+1)}$ is updated by the GRU according to

$$\boldsymbol{\xi}^{(k)} = \sigma\big(\boldsymbol{W}_\xi \boldsymbol{f}^{(k)} + \boldsymbol{U}_\xi \boldsymbol{h}^{(k)} + \boldsymbol{b}_\xi\big), \tag{40}$$

$$\boldsymbol{z}^{(k)} = \sigma\big(\boldsymbol{W}_z \boldsymbol{f}^{(k)} + \boldsymbol{U}_z \boldsymbol{h}^{(k)} + \boldsymbol{b}_z\big), \tag{41}$$

$$\tilde{\boldsymbol{h}}^{(k+1)} = \tanh\big(\boldsymbol{W}_h \boldsymbol{f}^{(k)} + \boldsymbol{U}_h\big(\boldsymbol{\xi}^{(k)} \odot \boldsymbol{h}^{(k)}\big) + \boldsymbol{b}_h\big), \tag{42}$$

$$\boldsymbol{h}^{(k+1)} = \big(\boldsymbol{1} - \boldsymbol{z}^{(k)}\big) \odot \boldsymbol{h}^{(k)} + \boldsymbol{z}^{(k)} \odot \tilde{\boldsymbol{h}}^{(k+1)}, \tag{43}$$

where $\boldsymbol{\xi}^{(k)}$ and $\boldsymbol{z}^{(k)}$ denote the reset gate and update gate, respectively, $\tilde{\boldsymbol{h}}^{(k+1)}$ denotes the candidate hidden state, and $\odot$ denotes the element-wise product.

Based on the updated hidden representation $\boldsymbol{h}^{(k+1)}$, an additional feature transformation is introduced as

$$\boldsymbol{g}^{(k)} = \sigma\big(\boldsymbol{W}_2 \boldsymbol{h}^{(k+1)} + \boldsymbol{b}_2\big). \tag{44}$$

The actor outputs the mean of the residual action distribution as

$$\boldsymbol{\mu}^{(k)} = \boldsymbol{W}_\mu \boldsymbol{g}^{(k)} + \boldsymbol{b}_\mu. \tag{45}$$

Compared with the feedforward policy, the recurrent actor can use historical interaction information through the hidden state. This design helps the pursuer adjust its residual correction according to different opponent behaviors.

The critic estimates the state value based on the augmented input, which can be written as

$$V_\phi\big(\boldsymbol{s}'^{(k)}\big) = f_\phi\big(\boldsymbol{s}'^{(k)}\big), \tag{46}$$

where $f_\phi(\cdot)$ denotes the value network. This value network is used to compute the PPO value loss and the advantage estimate.

Under the recurrent setting, the policy distribution is expressed as

$$\pi_\theta\big(\Delta \boldsymbol{v}_p^{\mathrm{res},(k)} \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\big) = \mathcal{N}\big(\boldsymbol{\mu}^{(k)}, \mathrm{diag}(\boldsymbol{\sigma}^2)\big). \tag{47}$$

Accordingly, the probability ratio in PPO becomes

$$\rho^{(k)}(\theta) = \frac{\pi_\theta\big(\Delta \boldsymbol{v}_p^{\mathrm{res},(k)} \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\big)}{\pi_{\theta_{\mathrm{old}}}\big(\Delta \boldsymbol{v}_p^{\mathrm{res},(k)} \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\big)}. \tag{48}$$

Remark 1: at the beginning of each episode, the GRU hidden state should be initialized before the first decision step and will be subsequently propagated throughout the episode. During PPO updates, transitions from the same episode are preserved in temporal order and divided into contiguous sequences with a maximum length of L. For each sequence, the hidden state stored at its first

transition is used as the initial hidden state.

To improve cross-opponent adaptability, an opponent-pool training mechanism is further introduced. The opponent pool is defined as

$$\mathcal{O} = \{o_1, o_2, \ldots, o_M\}. \tag{49}$$

In this work, the pool contains M different types of evaders, including learning-based evaders, model-based evaders, and non-maneuvering evaders. During each training episode, one opponent is sampled from the pool. This provides different state-transition distributions and prevents the pursuer from fitting only one fixed evader strategy.

Let the opponent sampling distribution associated with training episode e be

$$p_e(o), \quad o \in \mathcal{O}. \tag{50}$$

The opponent for each episode is sampled according to the stage-dependent distribution $p_e(o)$. The sampling distribution varies across different training stages. The training objective for episode e can be written as

$$\max_{\theta} \mathbb{E}_{o\sim p_e(o)}[G(\theta; o)]. \tag{51}$$

where $G(\theta; o)$ denotes the expected return of policy θ against opponent o.

Through the above residual recurrent policy and opponent-pool training, the proposed R^{3}LOOP framework combines the model-based LQR prior with learning-based residual adaptation. The LQR baseline provides the main pursuit direction, the PPO residual refines this action, and the recurrent hidden state encodes historical interaction information. As a result, one pursuer policy can better adapt to different evader strategies.

3.4 Training reward function design

In addition to the strategy structure, the reward function is also a key factor in the proposed framework. According to the objectives and constraints of the orbital pursuit-evasion task, the reward function $r^{(k)}$ is constructed by four parts: the distance variation term $r_d^{(k)}$, the time penalty term $r_\tau^{(k)}$, the terminal outcome term $r_f^{(k)}$, and the residual-impulse penalty term $r_{\Delta v}^{(k)}$. Therefore, the reward function can be written as

$$r^{(k)} = r_d^{(k)} + r_\tau^{(k)} + r_f^{(k)} + r_{\Delta v}^{(k)}. \tag{52}$$

The distance variation term $r_d^{(k)}$ is introduced to encourage the pursuer to continuously reduce the relative distance to the evader. Let $d^{(k)}$ denote the relative distance between the pursuer and the evader at decision step k. Then, this reward term is given by

$$r_d^{(k)} = \alpha(d^{(k)} - d^{(k+1)}), \tag{53}$$

where $\alpha > 0$ is the distance reward coefficient. When the relative distance decreases within the current time step, this term is positive. Otherwise, it becomes negative. In this way, the policy receives immediate feedback that is consistent with the objective of approaching the target.

The time penalty term $r_\tau^{(k)}$ is introduced to prevent ineffective delay during the pursuit-evasion process and to encourage the pursuer to complete the capture task as quickly as possible under the given constraints. This term is defined as

$$r_\tau^{(k)} = -\kappa_\tau, \tag{54}$$

where $\kappa_\tau > 0$ is the constant time-penalty coefficient. Since this term is negative at every time step, the accumulated penalty increases as the episode becomes longer. Therefore, this term drives the policy to accomplish the task in fewer steps.

The terminal outcome term $r_f^{(k)}$ is used to emphasize the final mission result and the importance of successful capture. A large positive reward is assigned when the pursuer satisfies the capture condition. A penalty is assigned when the episode ends in failure. This term is written as

$$r_f^{(k)} = \begin{cases} r_{\text{succ}}, & d^{(k+1)} \leqslant d_{\text{cap}} \\ -r_{\text{fail}}, & \text{mission failure} \\ 0, & \text{otherwise} \end{cases}, \tag{55}$$

where $r_{\text{succ}} > 0$ denotes the capture reward, $r_{\text{fail}} > 0$ denotes the failure penalty, and d_{cap} is the preset capture threshold. This term strengthens the final task objective, so that the policy is optimized not only for local gains during the process, but also for successful capture from a global perspective.

The residual-impulse penalty term $r_{\Delta v}^{(k)}$ is used to suppress excessive control input. It can also constrain excessive corrections to the baseline strategy and avoid unnecessary additional maneuvers. Define the residual maneuver cost as

$$\chi_p^{\text{res},(k)} = \|\Delta \boldsymbol{v}_p^{\text{res},(k)}\|_1. \tag{56}$$

The residual-impulse penalty is

$$r_{\Delta v}^{(k)} = -\beta \chi_p^{\text{res},(k)}, \tag{57}$$

where $\beta > 0$ is the residual-impulse penalty coefficient and $\|\cdot\|_1$ denotes the sum of the absolute values of the three components.

By combining the above terms together, the final reward at each step k is written as

$$r^{(k)} = \alpha(d^{(k)} - d^{(k+1)}) - \kappa_\tau + r_f^{(k)} - \beta \chi_p^{\text{res},(k)}. \tag{58}$$

This reward design jointly considers pursuit efficiency, mission duration, terminal outcome, and control cost. It provides a training signal that is consistent with the objective of the orbital pursuit-evasion task.

The overall training procedure of the proposed framework is summarized in Algorithm 1.

Algorithm 1 Training procedure of the pursuer R^{3}LOOP decision-making model.

Initialize the actor π_θ, critic V_ϕ, replay buffer $\mathcal{B}$, and opponent pool O
Set hyperparameters: episode number N_{ep}, maximum decision steps N, batch size B, sequence length L, and PPO update epochs K
for episode $e = 1,\ldots,N_{\text{ep}}$ **do**
  Reset the environment and obtain the initial state $\boldsymbol{s}^{(0)}$
  Sample an opponent $o \sim p_e(o)$ from the opponent pool
  Initialize the GRU hidden state $\boldsymbol{h}^{(0)} = \boldsymbol{0}$
  for $k = 0,\ldots,N-1$ **do**
    Compute the LQR baseline action $\Delta\boldsymbol{v}_p^{\text{LQR},(k)}$
    Form the augmented input

$$\boldsymbol{s}'^{(k)} = \left[\left(\boldsymbol{s}^{(k)}\right)^{\mathrm{T}}, \left(\Delta\boldsymbol{v}_p^{\text{LQR},(k)}\right)^{\mathrm{T}}\right]^{\mathrm{T}}$$

    Sample the residual action and update the hidden state

$$(\Delta\boldsymbol{v}_p^{\text{res},(k)}, \log\pi^{(k)}, \boldsymbol{h}^{(k+1)}) \sim \pi_\theta\left(\cdot \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\right)$$

    Generate the evader action $\Delta\boldsymbol{v}_e^{(k)}$ according to opponent o
    Compute the final pursuer action

$$\Delta\boldsymbol{v}_p^{(k)} = \text{sat}_{\eta_p}(\Delta\boldsymbol{v}_p^{\text{LQR},(k)} + \Delta\boldsymbol{v}_p^{\text{res},(k)})$$

    Execute $(\Delta\boldsymbol{v}_p^{(k)}, \Delta\boldsymbol{v}_e^{(k)})$, perform orbital propagation, and obtain $(\boldsymbol{s}^{(k+1)}, r^{(k)}, done)$
    Store the transition into $\mathcal{B}$
    Update $\boldsymbol{s}^{(k)} \leftarrow \boldsymbol{s}^{(k+1)}$
    Update $\boldsymbol{h}^{(k)} \leftarrow \boldsymbol{h}^{(k+1)}$
    if done **then**
      break
    end if
  end for
  if $|\mathcal{B}| \geqslant B$ **then**
    Compute values, returns, and advantages $\hat{A}^{(k)}$ by GAE
    Organize samples according to episode index and time order
    Split trajectories into sequence blocks of length L
    for epoch $j = 1,\ldots,K$ **do**
      Compute the policy ratio

$$\rho^{(k)}(\theta) = \frac{\pi_\theta\left(\Delta\boldsymbol{v}_p^{\text{res},(k)} \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\right)}{\pi_{\theta_{\text{old}}}\left(\Delta\boldsymbol{v}_p^{\text{res},(k)} \mid \boldsymbol{s}'^{(k)}, \boldsymbol{h}^{(k)}\right)}$$

      Update the actor by $L^{\text{CLIP}}(\theta)$
      Update the critic by the value loss $L^V(\phi)$
    end for
    Clear $\mathcal{B}$
  end if
end for
Output the trained pursuer R^{3}LOOP model

4 Simulation Results

This section presents numerical simulations of the proposed framework. The results are designed to validate the residual correction mechanism and the cross-opponent generalization capability under the recurrent policy.

4.1 Simulation settings

Two groups of simulations are conducted under the scenario described in Section 2. The simulations adopt the same scenario setting, dynamic model, and evaluation criteria. At the beginning of each episode, the initial position of the evader is fixed at the origin, while the initial position of the pursuer is randomly generated within a spherical shell centered at the origin with a radius of 30–35 km. The pursuer and the evader apply impulsive thrusts simultaneously at each control instant, and each impulse is treated as an instantaneous velocity increment directly added to the spacecraft velocity. After the impulse is applied, the system evolves freely within one discrete decision period according to the relative motion dynamics. The decision period $\Delta t = 600$ s is the interval between consecutive action updates. Within each decision period, the orbital state is propagated every $\Delta t_p = 30$ s, and the capture condition is checked at each propagation step; these steps do not introduce additional control decisions. In addition, both spacecraft are assumed to have full-state observability, that is, the relative position and relative velocity at the current control step are available at every decision instant.

During both training and testing, the reward function follows the unified form given in Section 3 and remains unchanged in all experiments. Unless otherwise specified, all subsequent experiments use the same parameter settings. The opponent pool contains LQR, PPO, soft actor-critic (SAC), and stationary evaders. The LQR evader applies a discrete-CW feedback impulse, while the stationary evader applies zero impulse. The independently trained PPO and SAC evaders receive the 18-dimensional state in Eq. (30) and output three-dimensional impulses; their parameters are fixed during pursuer training, and deterministic actions are used for evaluation. Each component of a maneuvering evader's impulse is constrained to [−1, 1] m/s. The detailed simulation and training parameters are listed in Tables 1 and 2.

The curriculum uses three training intervals: 0−20%, 20%−66%, and 66%−100%. For the ordered opponent set (PPO, SAC, LQR, stationary), the corresponding sampling probabilities are (0.35,0.35,0.20,0.10), (0.20, 0.25,0.50,0.05), and (0.20,0.35,0.40,0.05), respectively.

4.2 Residual RL effectiveness validation

This subsection isolates the effect of residual learning under single-opponent training. For each evader strategy, an independent PPO+LQR pursuer is trained against that

Table 1 Simulation environment parameters.

Parameter	Setting
Decision period	$\Delta t = 600$ s
State propagation period	$\Delta t_p = 30$ s
Maximum decision steps	$N = 30$
Maximum pursuit time	$t_{\max} = 18{,}000$ s
Capture threshold	$d_{\text{cap}} = 500$ m
Baseline impulse bound of pursuer	$\eta_p^{\text{LQR}} = [2, 2, 2]^{\text{T}}$ m/s
Residual impulse bound of pursuer	$\eta_p^{\text{res}} = [0.3, 0.3, 0.3]^{\text{T}}$ m/s
Final impulse bound of pursuer	$\eta_p = [2.3, 2.3, 2.3]^{\text{T}}$ m/s
Impulse bound of evader	$\eta_e = [1, 1, 1]^{\text{T}}$ m/s

evader and compared with the LQR and PPO baselines under identical simulation settings.

The simulations use the same settings as above. LQR is taken as the prior strategy. PPO is introduced to provide residual correction. Four representative evader policies are considered, including LQR, PPO, SAC, and a stationary policy. For each evader policy, a PPO+LQR pursuer model is trained separately. The trained model is then loaded for 500 Monte Carlo interception tests. The results are reported in Tables 3 and 4 and Figs. 4, 5 and 6.

Table 3 shows that the independent PPO policy can hardly complete the task under the considered setting. This is mainly because the terminal capture distance is small, while the CW update interval is relatively long. Although an intermediate capture check is performed every 30 s during propagation, the independent PPO policy with a traditional reward design still fails in most cases. In contrast, the independent LQR policy shows different performance against different evader policies. It achieves 100% capture against the stationary evader, but only 2.8% capture against the SAC evader. This indicates that the LQR baseline is effective in simple cases, but its performance can drop significantly under more adversarial evasion behaviors. After introducing the RL residual term, the PPO+LQR pursuer policy shows much stronger adaptability to different evasion behaviors. As shown in Table 3, it achieves 100% capture against the LQR, SAC, and stationary evaders, and 99.8% capture against the PPO evader. Table 5 and Fig. 5 further show the results under different initial distances. The independent PPO policy still has almost no successful captures, while the LQR policy is sensitive to the evader

Table 2 Training and controller parameters.

Parameter	Setting	Parameter	Setting
Discount factor	$\gamma = 0.99$	Batch size	$B = 2\,048$
GAE parameter	$\lambda = 0.95$	Mini-batch size	$B_{\text{mini}} = 256$
PPO clipping coefficient	$\epsilon = 0.1$	PPO update epochs	$K = 3$
Actor learning rate	$lr_a = 1\times10^{-3}$	Critic learning rate	$lr_c = 2\times10^{-3}$
Width of the first hidden layer	$h_1 = 256$	Width of the second hidden layer	$h_2 = 128$
GRU hidden-state dimension	$h_{\text{GRU}} = 256$	Sequence length	$L = 8$
Position normalization scale	2×10^4 m	Velocity normalization scale	100 m/s
LQR-impulse normalization scale	2 m/s	LQR control weighting matrix	$\boldsymbol{R} = 8\boldsymbol{I}_3$
Distance reward coefficient	$\alpha = 5\times10^{-4}$	Time penalty coefficient	$\kappa_\tau = 0.2$
Residual-impulse penalty coefficient	$\beta = 0.5$	Capture/failure rewards	$r_{\text{succ}} = 50$, $r_{\text{fail}} = 30$
LQR state weighting matrix	$\boldsymbol{Q} = \text{diag}(5\times10^{-6}, 5\times10^{-6}, 5\times10^{-6}, 10, 10, 10)$		

Table 3 Interception success rates of the pursuer policies against the evader policies over 500 episodes.

Pursuer policy	Evader policy			
	LQR	PPO	SAC	Stationary
LQR	56.2%	55.6%	2.8%	100%
PPO	0%	0%	0%	0.2%
PPO+LQR	100%	99.8%	100%	100%

Table 4 Average steps and cumulative maneuver cost under different evader policies over 500 episodes.

Evader policy	Pursuer policy	Steps (succ.)	$C_p^{(k_f)}$ (succ.) (m/s)	Steps (all)	$C_p^{(k_f)}$ (all) (m/s)
LQR	LQR	25.26	85.53	27.33	94.17
	PPO+LQR	18.30	70.96	18.30	70.96
PPO	LQR	12.07	33.64	20.03	37.00
	PPO+LQR	11.04	33.13	11.08	33.16
SAC	LQR	9.86	33.70	29.44	48.51
	PPO+LQR	16.83	41.01	16.83	41.01
Stationary	LQR	12.14	32.49	12.14	32.49
	PPO+LQR	11.73	32.02	11.73	32.02

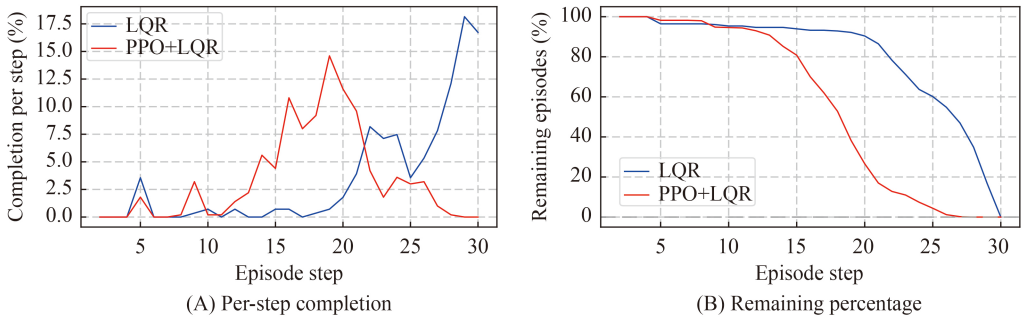


Fig. 4 Distribution of capture completion steps over test episodes.

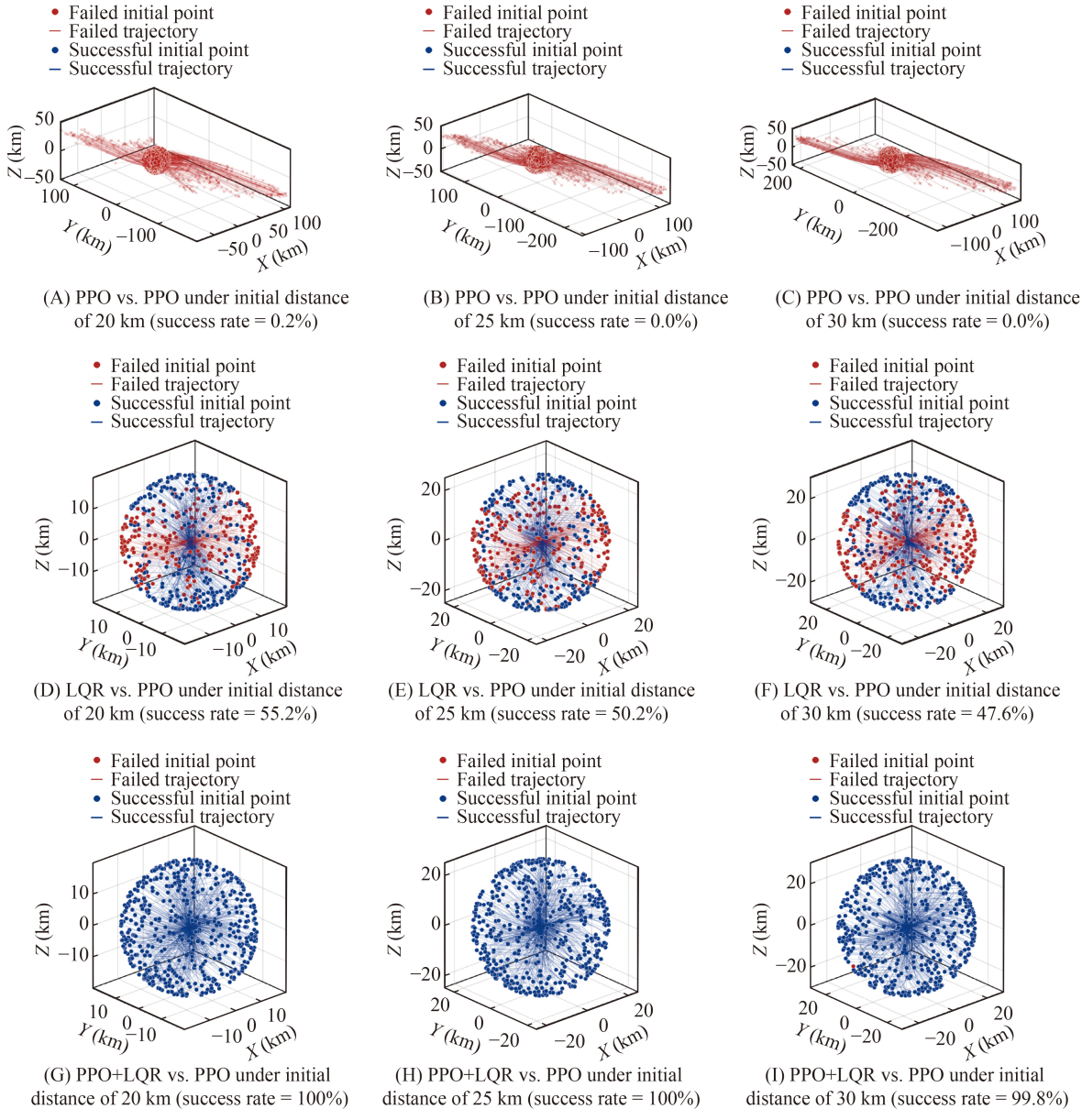


Fig. 5 Interception outcomes under different pursuer strategies and initial distances against the PPO evader.

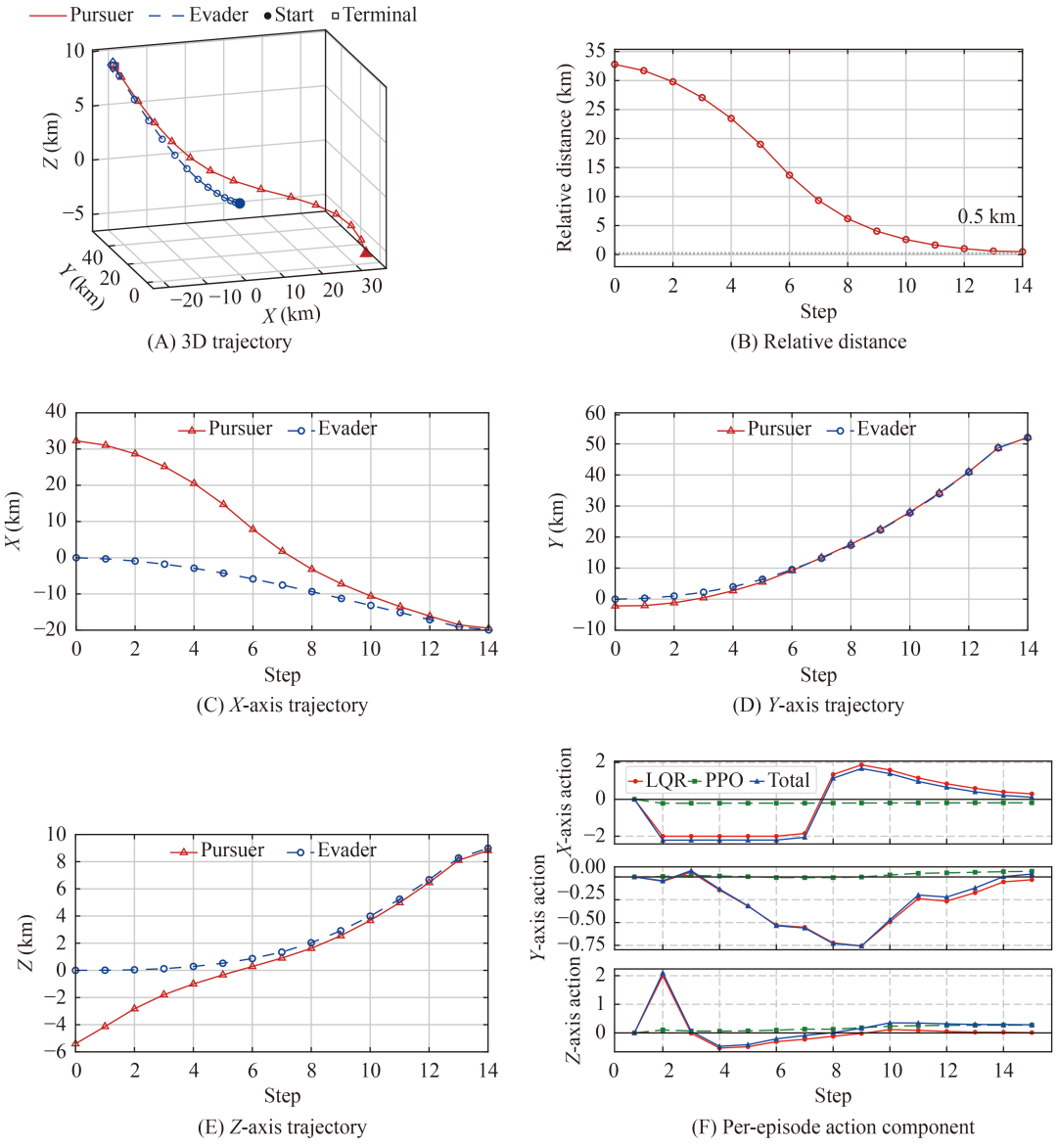


Fig. 6 Pursuer-evader motion visualization (pursuer with PPO+LQR vs. evader with PPO)

Table 5 Success rates under different initial distances and evader strategies.

Pursuer policy	Initial distance	Evader policy			
		PPO	LQR	SAC	Stationary
PPO	20 km	0.2%	0%	0%	0.2%
	25 km	0%	0%	0%	0%
	30 km	0%	0%	0%	0.2%
LQR	20 km	55.2%	95.8%	11.4%	100%
	25 km	50.2%	84.6%	6.8%	100%
	30 km	47.6%	71.6%	4.8%	100%
PPO+LQR	20 km	100%	100%	100%	100%
	25 km	100%	100%	100%	100%
	30 km	99.8%	100%	100%	100%

type and initial distance. In contrast, PPO+LQR maintains near-perfect success against the PPO evader, with 100% at 20 and 25 km and 99.8% at 30 km. These results verify that the residual correction can significantly

improve the capture capability of the pursuer.

Table 4 further shows that the PPO+LQR policy not only improves the mission success rate, but also outperforms the independent LQR policy in terms of average capture steps and cumulative maneuver cost in most cases. For example, against the LQR evader, the average successful capture steps are reduced from 25.26 to 18.30, and the average cumulative maneuver cost $C_p^{(k_f)}$ is reduced from 85.53 to 70.96 m/s. This result indicates that the PPO residual term does not improve performance by simply increasing the control magnitude. Instead, it refines the baseline LQR action while preserving the guidance provided by the prior strategy.

Fig. 4 presents the distribution of terminal episode steps when the evader adopts the LQR policy. The distributions show that the completion steps of the PPO+LQR policy are mainly concentrated before step 20, while the completion steps of the independent LQR policy are more scattered and extend to later steps. This concentration indicates that the PPO residual correction not only improves task completion, but also helps the pursuer complete the mission earlier. Fig. 5 further shows the initial-condition success maps under different initial distances. The pursuer model is trained with initial distances of 30–35 km, and is then tested under different initial-distance settings. The independent PPO policy almost fails to capture the evader under all tested distances, which indicates that pure RL is difficult to train from scratch in this long-horizon and sparse-capture task. The independent LQR policy achieves partial success, but its successful and failed initial points are mixed, and its performance is sensitive to the initial distance. In contrast, the PPO+LQR policy maintains near-perfect capture rates across the tested initial distances. This result shows that the residual correction improves the robustness of the LQR baseline and helps the pursuer generalize to different initial-distance conditions. Fig. 6 shows a representative pursuit-evasion episode, including the 3D trajectory, relative distance, position variations, and per-step control components. The residual impulse is bounded by ±0.3 m/s per axis, only 15% of the ±2.0 m/s LQR bound. Therefore, LQR supplies the principal pursuit impulse and PPO provides bounded local corrections. These results verify the effectiveness of the proposed residual RL strategy.

4.3 Residual recurrent RL effectiveness validation

This subsection evaluates cross-opponent generalization. A single PPO-GRU+LQR pursuer is jointly trained against all evaders in the opponent pool. For comparison, PPO+LQR and PPO-GRU+LQR use the same pool, stage-wise sampling schedule, and remaining training settings.

The simulation settings are the same as those in the previous subsection. This opponent pool contains the evader policies described above. After training, each pursuer is tested separately against all four evader policies for 500 Monte Carlo episodes per pairing. The results are given in Tables 6 and 7 and Figs. 7 and 8.

Table 6 and Fig. 7 show that the PPO-GRU+LQR policy can handle different evader policies well and maintains a high capture success rate in all cases. Compared with the standard PPO+LQR policy, PPO-GRU+LQR improves the success rate against the LQR evader from 72.4% to 91.0%, and improves the success rate against the SAC evader from 93.2% to 96.0%. Against the PPO evader and the stationary evader, both methods maintain high success rates. This result indicates that the GRU module improves the generalization performance under multiple opponent strategies.

Table 7 shows that, although the PPO-GRU+LQR

Table 6 Cross-opponent interception success rates of the pursuer policies against the evader policies over 500 episodes.

Pursuer policy	Evader policy			
	LQR	PPO	SAC	Stationary
PPO+LQR	72.4%	99.8%	93.2%	100%
PPO-GRU+LQR	91.0%	99.0%	96.0%	100%

Table 7 Average steps and cumulative maneuver cost under different evader policies over 500 episodes.

Evader policy	Pursuer policy	Steps (succ.)	$C_p^{(k_f)}$ (succ.) (m/s)	Steps (all)	$C_p^{(k_f)}$ (all) (m/s)
LQR	PPO+LQR	21.02	75.65	23.50	79.16
	PPO-GRU+LQR	22.78	80.08	23.43	81.14
PPO	PPO+LQR	13.86	34.64	13.89	34.64
	PPO-GRU+LQR	13.23	33.58	13.40	33.71
SAC	PPO+LQR	20.70	44.09	21.34	44.63
	PPO-GRU+LQR	20.46	43.60	20.84	43.57
Stationary	PPO+LQR	14.70	33.10	14.70	33.10
	PPO-GRU+LQR	12.87	32.41	12.87	32.41

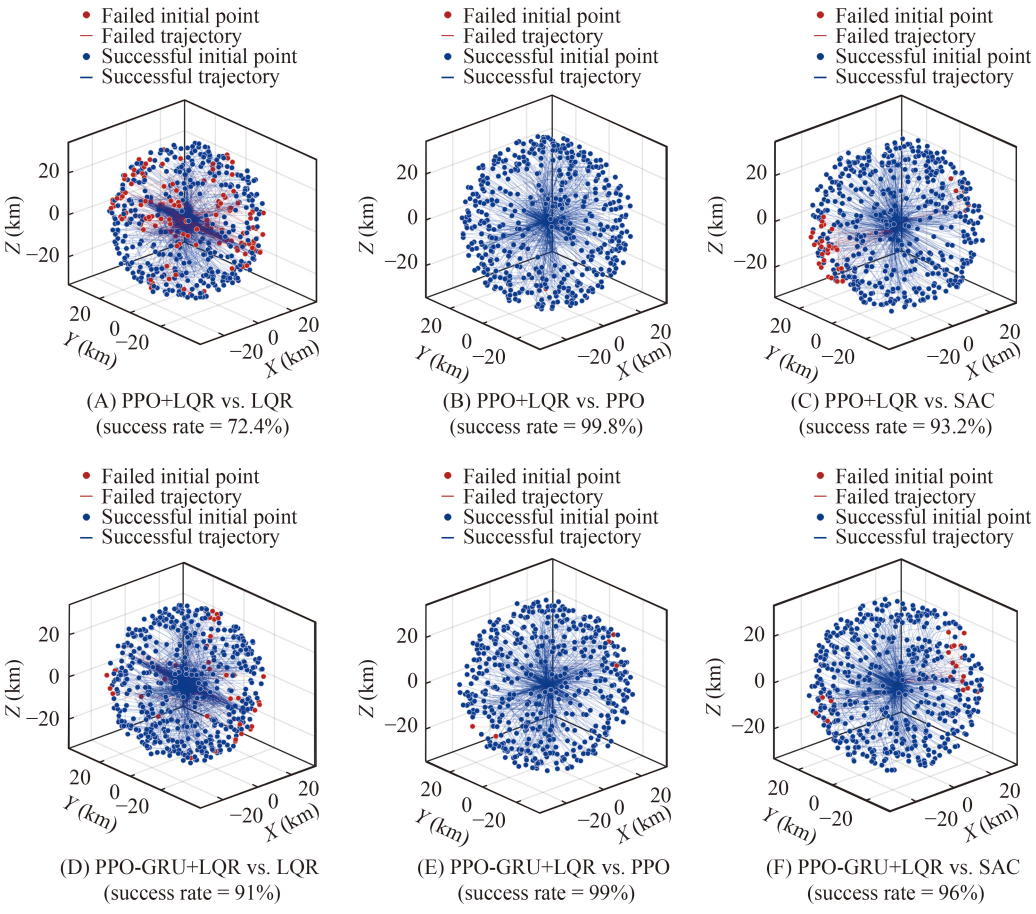


Fig. 7 Cross-opponent interception results for the PPO+LQR and PPO-GRU+LQR pursuer policies.

policy has better generalization ability, it is not always superior to the standard PPO+LQR policy in terms of average completion steps and cumulative maneuver cost. For example, against the LQR evader, PPO-GRU+LQR has a higher success rate, but its average successful steps and cumulative maneuver cost are slightly larger. Against the PPO, SAC, and stationary evaders, PPO-GRU+LQR achieves comparable or better average steps and cumulative maneuver cost. These results indicate that the main advantage brought by GRU lies in better robustness and stronger cross-opponent generalization, rather than absolute optimality in time or cumulative maneuver cost under every single scenario.

Fig. 8 shows a representative pursuit-evasion episode of PPO-GRU+LQR against the PPO evader, including the 3D trajectory, relative distance, position variations, and per-step control components. The action profiles show that, even after introducing the GRU structure, the PPO output still mainly acts as a correction to the baseline LQR action, rather than directly producing large maneuver commands. This indicates that the recurrent module mainly enhances temporal feature extraction, instead of changing the basic residual correction property of the proposed framework.

5 Conclusions

This paper proposes an R^{3}LOOP framework for the three-dimensional orbital pursuit-evasion game. First, an LQR-based model prior is introduced to provide a nominal pursuit baseline for the pursuer. Then, a residual recurrent RL policy is constructed to generate corrections to the prior action and improve adaptability against different evader maneuvers. Finally, an opponent-pool training mechanism is developed to support policy training under diverse evader strategies. Simulation results verify the effectiveness of the proposed framework and demonstrate its advantages in capture performance and cross-opponent generalization ability. Note that the proposed framework is not limited to the specific combination of LQR and PPO. Instead, it provides a general hybrid example in which the model-based baseline strategy and the learning-based residual

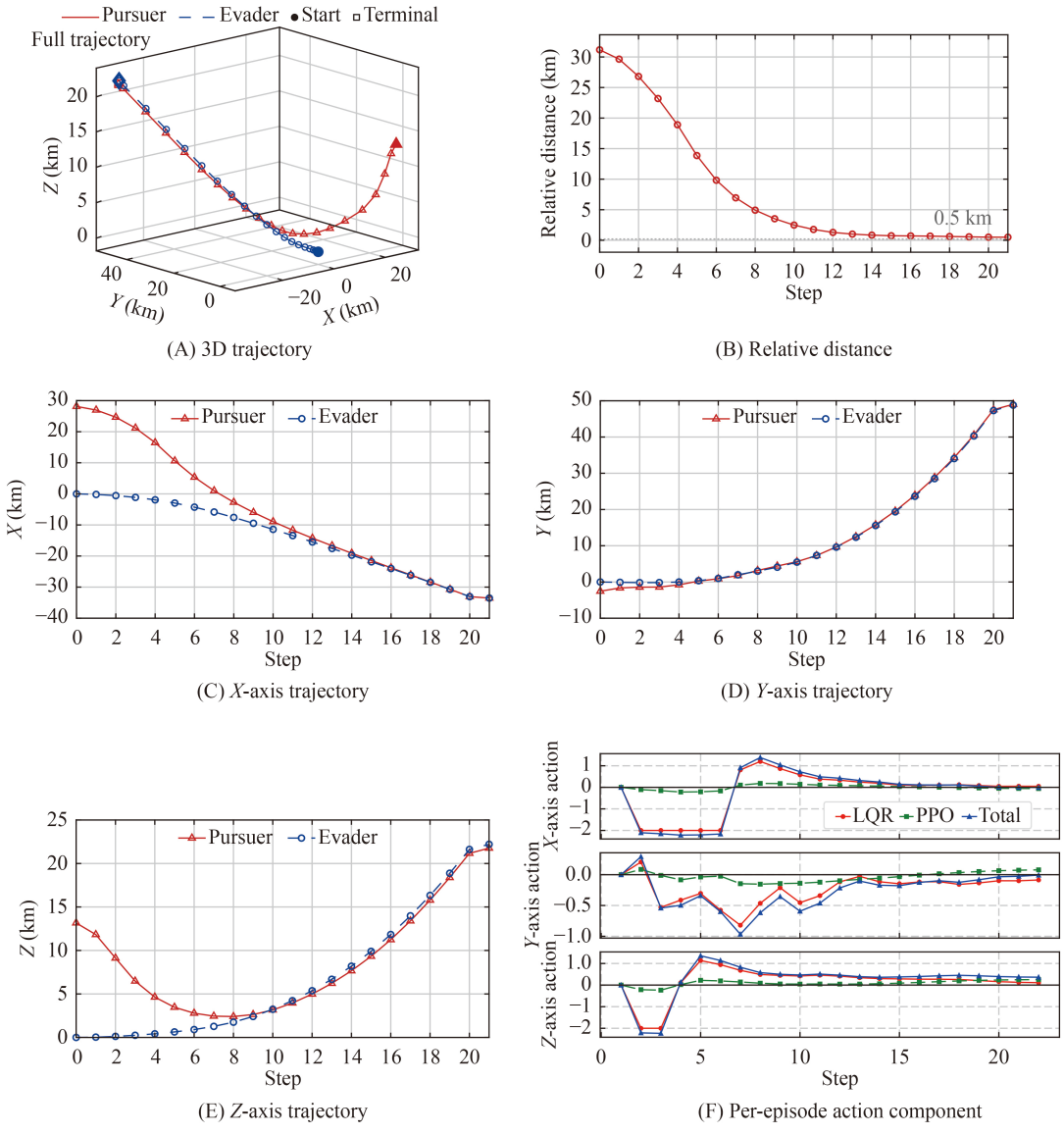


Fig. 8 Pursuer-evader motion visualization (pursuer with PPO-GRU+LQR vs. evader with PPO).

policy can be replaced by other suitable methods according to different task requirements. Future work will consider higher-fidelity orbital dynamics, imperfect observations, and more diverse opponent behaviors.

Data availability No datasets were generated or analyzed for this work.

Acknowledgments This work was supported in part by the National Science Foundation for Continuing Grant of Young Scientists Fund-Type A under Grant 62625301, the National Natural Science Foundation of China under Grant 62403052, the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under Grant JYB2025XDXM206, and the Young Scientists Foundation of CSAA (Guidance Navigation and Control, GNC) under Grant CSAA-YSF2025-GNC-20. The authors used GPT-5.6, online version as a language-editing tool to improve grammar, clarity, and readability. The AI tool was not used for data analysis, data generation, image processing, or scientific content creation. All outputs were carefully reviewed and verified by the authors, who take full responsibility for the final content.

Competing interests The authors declare no conflicts of interest.

References

1. Yang K, Shen A, Xu N W, et al. A review of reinforcement learning approaches for pursuit-evasion games. Chinese Journal of Aeronautics, 2026, 39(6): 103940
2. Cao X Y, Ning X, Liu S Y, et al. Spacecraft intelligent orbital game technology: a review. Chinese Journal of Aeronautics, 2025, 38(6): 103480
3. Dong W, Wang C Y, Wang J N, et al. *Spatial-temporal cooperative guidance for multiple flight vehicles*. Singapore:

Springer, 2025
4. Yi X, Wang C Y, Dong W, et al. Observation-shared cooperative look-angle tracking guidance for multiple missiles with partial seekers. IEEE Transactions on Aerospace and Electronic Systems, 2025, 61(5): 11586−11599
5. Di Mauro G, Lawn M, Bevilacqua R. Survey on guidance navigation and control requirements for spacecraft formation-flying missions. Journal of Guidance, Control, and Dynamics, 2018, 41(3): 581−602
6. Anderson G M, Grazier V W. Barrier in pursuit-evasion problems between two low-thrust orbital spacecraft. AIAA Journal, 1976, 14(2): 158−163
7. Jagat A, Sinclair A J. Nonlinear control for spacecraft pursuit-evasion game using the state-dependent Riccati equation method. IEEE Transactions on Aerospace and Electronic Systems, 2017, 53(6): 3032−3042
8. Han H Y, Dang Z H. Optimal delta-v-based strategies in orbital pursuit-evasion games. Advances in Space Research, 2023, 72(2): 243−256
9. Yang Q L, Luo Y Z, Guo X, et al. An approximate analytical method for solving the coplanar spacecraft pursuit-evasion game barrier. Acta Astronautica, 2026, 238: 717−730
10. Pontani M, Conway B A. Numerical solution of the three-dimensional orbital pursuit-evasion game. Journal of Guidance, Control, and Dynamics, 2009, 32(2): 474−487
11. Li Z Y, Zhu H, Yang Z, et al. Saddle point of orbital pursuit-evasion game under J2-perturbed dynamics. Journal of Guidance, Control, and Dynamics, 2020, 43(9): 1733−1739
12. Tang X, Ye D, Huang L, et al. Pursuit-evasion game switching strategies for spacecraft with incomplete-information. Aerospace Science and Technology, 2021, 119: 107112
13. Liu P X, Yang B, Li S, et al. Parameter-optimized pursuit strategy for orbital games with incomplete information. Journal of Guidance, Control, and Dynamics, 2025, 48(8): 1730−1742
14. Shi M M, Ye D, Sun Z W, et al. Spacecraft orbital pursuit-evasion games with J2 perturbations and direction-constrained thrust. Acta Astronautica, 2023, 202: 139−150
15. Chai Y, Luo J J, Wang M M, et al. Predictive pursuit-evasion game control method for approaching space non-cooperative target. IFAC-PapersOnLine, 2020, 53(2): 14882−14887
16. Liu Y C, Li C Y, Jiang J, et al. A model predictive Stackelberg solution to orbital pursuit-evasion game. Chinese Journal of Aeronautics, 2025, 38(2): 103198
17. Li Z Y, Luo Y Z. Orbital impulsive pursuit-evasion game formulation and Stackelberg equilibrium solutions. Journal of Spacecraft and Rockets, 2025, 62(2): 349−364
18. Shirobokov M, Trofimov S, Ovchinnikov M. Survey of machine learning techniques in spacecraft control design. Acta Astronautica, 2021, 186: 87−97
19. Federici L, Scorsoglio A, Zavoli A, et al. Meta-reinforcement learning for adaptive spacecraft guidance during finite-thrust rendezvous missions. Acta Astronautica, 2022, 201: 129−141
20. Li X, Wang X G. Online solution for orbital pursuit-evasion game via heterogeneous proximal policy optimization. IEEE Transactions on Aerospace and Electronic Systems, 2025, 61(5): 12044−12058
21. Gronauer S, Diepold K. Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review, 2022, 55(2): 895−943
22. Zhang J C, Wang Y D, Zheng W, et al. Spacecraft pursuit-evasion game with incomplete information: a pre-training approximate dynamic programming method. Acta Astronautica, 2026, 243: 251−262
23. Zhang J R, Zhang K P, Zhang Y, et al. Near-optimal interception strategy for orbital pursuit-evasion using deep reinforcement learning. Acta Astronautica, 2022, 198: 9−25
24. Jiang R, Ye D, Xiao Y, et al. Orbital interception pursuit strategy for random evasion using deep reinforcement learning. Space: Science & Technology, 2023, 3: 0086
25. Zheng Z X, Zhang P, Yuan J P. Nonzero-sum pursuit-evasion game control for spacecraft systems: a Q-learning method. IEEE Transactions on Aerospace and Electronic Systems, 2023, 59(4): 3971−3981
26. Zhao L R, Zhang Y L, Dang Z H. PRD-MADDPG: an efficient learning-based algorithm for orbital pursuit-evasion game with impulsive maneuvers. Advances in Space Research, 2023, 72(2): 211−230
27. Yu W Z, Liu C, Yue X K. Reinforcement learning-based decision-making for spacecraft pursuit-evasion game in elliptical orbits. Control Engineering Practice, 2024, 153: 106072
28. Wang H B, Zhang Y. Impulsive maneuver strategy for multi-agent orbital pursuit-evasion game under sparse rewards. Aerospace Science and Technology, 2024, 155: 109618
29. Wang H B, Zhang Y, Liu H, et al. Impulsive thrust strategy for orbital pursuit-evasion games based on impulse-like constraint. Chinese Journal of Aeronautics, 2025, 38(1): 103180
30. Wei L S, Ning X, Lian X B, et al. Continuous-time hierarchical reinforcement learning for satellite pursuit decision. Chinese Journal of Aeronautics, 2025, 38(12): 103662
31. Staessens T, Lefebvre T, Crevecoeur G. Adaptive control of a mechatronic system using constrained residual reinforcement learning. IEEE Transactions on Industrial Electronics, 2022, 69(10): 10447−10456
32. Silver T, Allen K, Tenenbaum J, et al. Residual policy learning. arXiv preprint: arXiv: 1812.06298, 2018
33. Johannink T, Bahl S, Nair A, et al. Residual reinforcement learning for robot control. In: Proceedings of 2019 International Conference on Robotics and Automation (ICRA). Montreal, Canada, 2019, 6023–6029
34. Youm D, Jung H, Kim H, et al. Imitating and finetuning model predictive control for robust and symmetric quadrupedal locomotion. IEEE Robotics and Automation Letters, 2023, 8(11): 7799−7806
35. Cheng J, Kang D, Fadini G, et al. Rambo: RL-augmented model-based whole-body control for loco-manipulation. IEEE Robotics and Automation Letters, 2025, 10(9): 9462−9469
36. Holt H, Armellin R. Reinforcement learning enhanced LQR and control Lyapunov functions for spacecraft proximity operations. IEEE Transactions on Robotics, 2025, 41: 5117−5129

37. Clohessy W H, Wiltshire R S. Terminal guidance system for satellite rendezvous. Journal of the Aerospace Sciences, 1960, 27(9): 653−658



Wei Dong received the B.Eng. degree in Flight Vehicle Design and Engineering and the Ph.D. degree in Aeronautical and Astronautical Science and Technology from the Beijing Institute of Technology, Beijing, China, in 2017 and 2023, respectively. He was a Visiting Researcher with the Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea, from 2021 to 2022, and a Postdoctoral Researcher with the Beijing Institute of Technology from 2023 to 2025. He currently serves as an Associate Professor at the School of AI, Beijing Institute of Technology, Beijing, China. His research interests include guidance and control of flight vehicles, cooperative control of multiagent systems, and aerospace intelligent systems.



Jiajie Wang received the B.Eng. degree in Aeronautical and Astronautical Engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2024. She is currently pursuing the Ph.D. degree in Intelligence Science and Technology with the School of AI, Beijing Institute of Technology, Beijing, China. Her research interests include orbital pursuit-evasion game and learning-based decision-making.



Chen Chen received the B.Eng. and Ph.D. degrees in Control Science and Engineering from the Beijing Institute of Technology, Beijing, China, in 2004 and 2009, respectively. She is currently a Professor with the School of AI, Beijing Institute of Technology, Beijing, China, and also with the Beijing Institute of Technology, Zhuhai, China. Her research interests include intelligent optimization and decision-making of complex systems.



Fang Deng received the B.Eng. and Ph.D. degrees in Control Science and Engineering from the Beijing Institute of Technology, Beijing, China, in 2004 and 2009, respectively. He is currently a Professor with the School of AI, Beijing Institute of Technology, Beijing, China, and also with the Shenzhen MSU-BIT University, Shenzhen, China. His research interests include swarm intelligence, autonomous sensor networks, and smart wearable devices. Prof. Deng serves as an Associate Editor for the *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Intelligent Vehicles*, *IEEE Transactions on Fuzzy Systems*, and *Chinese Journal of Aeronautics*. He is Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and the Institution of Engineering and Technology (IET).



Zhengtao Ding received the B.Eng. degree in Engineering from Tsinghua University, Beijing, China, in 1984, and the M.Sc. degree in Systems and Control in 1986 from the University of Manchester Institute of Science and Technology, Manchester, UK, where he received the Ph.D. degree in Control Systems, in 1989. After working as a Lecturer with Ngee Ann Polytechnic, Singapore, for ten years, in 2003, he joined the University of Manchester, where he is currently a Professor of Control Systems with the Department of Electrical and Electronic Engineering. He authored the book titled *Nonlinear and Adaptive Control Systems* (IET, 2013) and has coauthored over 300 research articles. His research interests include nonlinear and adaptive control theory and their applications, and more recently, network-based control, distributed optimization, and distributed machine learning, with applications to power systems and robotics. Dr. Ding has served as the Editor-in-Chief for *Encyclopedia of Systems and Control Engineering* (Elsevier, 2026), and an Associate Editor for *IEEE Transactions on Automatic Control*, *IEEE Control Systems Letters, Chinese Journal of Aeronautics*, and several other journals. He is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE).