IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Online Planning for Multi-UAV Pursuit-Evasion in Unknown Environments Using Deep Reinforcement Learning

Jiayu Chen, Chao Yu, Guosheng Li, Wenhao Tang, Shilong Ji, Xinyi Yang, Botian Xu, Huazhong Yang, *Fellow, IEEE*, and Yu Wang, *Fellow, IEEE*

***Abstract*—Multi-UAV pursuit-evasion, where pursuers aim to capture evaders, poses a key challenge for UAV swarm intelligence. Multi-agent reinforcement learning (MARL) has demonstrated potential in modeling cooperative behaviors, but most RL-based approaches remain constrained to simplified simulations with limited dynamics or fixed scenarios. Previous attempts to deploy RL policy to real-world pursuit-evasion are largely restricted to two-dimensional scenarios, such as ground vehicles or UAVs at fixed altitudes. In this paper, we propose a novel MARL-based algorithm that learns online planning for multi-UAV pursuit-evasion in unknown environments (OPEN). OPEN introduces an evader prediction-enhanced network to tackle partial observability in cooperative policy learning. Additionally, OPEN proposes an adaptive environment generator within MARL training, enabling higher exploration efficiency and better policy generalization across diverse scenarios. Simulations show our method significantly outperforms all baselines in challenging scenarios, generalizing to unseen scenarios with a 100% capture rate. Finally, after integrating calibrated dynamics models of UAVs into training, we derive a feasible policy via a two-stage reward refinement and deploy the policy on real quadrotors in a zero-shot manner. To our knowledge, this is the first work to derive and deploy an RL-based policy using collective thrust and body rates control commands for multi-UAV pursuit-evasion in unknown environments. The open-source code and videos are available at https://sites.google.com/view/pursuit-evasion-rl.**

***Index Terms*—Cooperating robots, multi-robot systems, reinforcement learning.**

Received 4 March 2025; accepted 16 June 2025. Date of publication 26 June 2025; date of current version 4 July 2025. This article was recommended for publication by Associate Editor G. Pereira and Editor M. A. Hsieh upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62406159 and Grant 62325405, in part by the Postdoctoral Fellowship Program of CPSF under Grant GZC20240830 and Grant 2024M761676, and in part by China Postdoctoral Science Special Foundation under Grant 2024T170496. *(Jiayu Chen and Chao Yu contributed equally to this work.) (Corresponding authors: Chao Yu; Yu Wang.)*

Jiayu Chen, Guosheng Li, Shilong Ji, Xinyi Yang, Huazhong Yang, and Yu Wang are with Tsinghua University, Beijing 100084, China (e-mail: jia768167535@gmail.com; yu-wang@tsinghua.edu.cn).

Chao Yu is with Tsinghua University, Beijing 100084, China, and also with Beijing Zhongguancun Academy, Beijing 100094, China (e-mail: zoeyuchao@gmail.com).

Wenhao Tang is with Tsinghua Shenzhen International Graduate School, Shenzhen 518055, China.

Botian Xu is with Beijing Zhongguancun Academy, Beijing 100094, China.

Digital Object Identifier 10.1109/LRA.2025.3583620

I. Introduction

MULTI-UAV pursuit-evasion, where teams of pursuers attempt to capture evaders while the evaders employ evasive strategies to avoid capture, is a key application of UAV swarms. Multi-UAV pursuit-evasion has important applications in both military and civilian contexts, including UAV defense systems [1], adversarial drone engagements [2], and search-and-rescue [3].

Traditional approaches to solving pursuit-evasion games, such as game theory [4], control theory [5], [6], [7], [8], [9], [10], and heuristic methods [11], [12], [13], face significant limitations when applied to complex real-world scenarios. These methods require accurate and reasonable knowledge of models and initial conditions [14], which may struggle to handle the nonlinear dynamics and high-dimensional environments typically encountered in practical applications. As a result, researchers have increasingly turned to artificial intelligence (AI) techniques, with reinforcement learning (RL) emerging as a leading solution [15], [16], [17], [18], [19]. RL enables UAVs to iteratively learn pursuit and evasion strategies by interacting with simulated environments. By leveraging neural networks, RL can model complex, cooperative behaviors and discover strategies that are difficult to encode using explicit rules.

However, despite these advancements, most RL-based methods for pursuit-evasion focus on simplified tasks in simulation [20], [21], [22], [23], [24]. These methods frequently model agents, both pursuers and evaders, as point masses or with limited kinematic properties, developing only high-level strategies. Such abstractions overlook the kinematic and dynamic constraints of real-world systems, limiting the effectiveness of these strategies outside simulation environments. Moreover, many RL approaches are tailored to fixed, predefined scenarios, reducing their ability to generalize to diverse and unknown scenarios [15], [20], [25]. Although recent work has explored deploying RL-based pursuit-evasion strategies in real-world settings [9], [25], [26], [27], these efforts have primarily been restricted to two-dimensional tasks, such as ground vehicles or UAVs constrained to fixed altitudes, without addressing the complexities of real three-dimensional environments.

This paper aims to learn an RL policy for multi-UAV pursuit-evasion, perform online planning in unknown environments, and deploy it on real UAVs. The problems are:

- *Joint optimization of planning and control:* UAV actions must be coordinated to capture the evader under partial

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

observation, while avoiding environmental obstacles, preventing collisions, and adhering to dynamics model and physical constraints for safe and feasible flight.
- *Large exploration space:* The 3D nature of UAVs, combined with varying scenarios, significantly expands the state space, resulting in a large number of samples required to find viable strategies using RL.
- *Policy generalization:* RL strategies that are optimized for specific scenarios often fail to generalize to new environments.
- *Sim-to-real transfer:* The sim-to-real gap, a common issue in RL, is particularly pronounced in multi-UAV pursuit-evasion tasks due to the physical constraints of UAVs and the need for agile, precise maneuvers.

We address these challenges by integrating calibrated dynamics models of UAVs into training policy for multi-UAV pursuit-evasion tasks. The RL-based policy generates collective thrust and body rates (CTBR) control commands, balancing flexibility, cooperative decision-making, and sim-to-real transfer. The overall algorithm is named OPEN (Online planning for multi-UAV pursuit-evasion in unknown environments). OPEN proposes an attention-based, evader-prediction-enhanced network that integrates predictive information about the evader's movements into the RL policy inputs, improving the ability to cooperatively capture the evader with partial observation. Additionally, we introduce an adaptive environment generator to MARL training, which automatically generates diverse and appropriately challenging curricula. This boosts sample efficiency and enhances policy generalization to unseen scenarios. Finally, we employ a two-stage reward refinement process to regularize policy output and ensure stability of behavior during real-world deployment.

We evaluate performance in four test scenarios, with results showing that OPEN outperforms all baselines with a clear margin and maintains a 100% capture rate in unseen scenarios, demonstrating strong generalization. Ablation studies show our approach improves sample efficiency by over 50% in the training tasks. Our method also demonstrates robustness to the evader speed, maintaining a high capture success rate even when confronted with high-speed evaders that are never encountered during training. We also deploy our policy on real Crazyflie quadrotors in a zero-shot manner, achieving strategies in the real world that closely mirror those observed in simulation. To the best of our knowledge, this is the first RL policy with CTBR commands in pursuit-evasion tasks and can be deployed directly on real quadrotors in unknown environments. Our contributions can be summarized as follows:

- We propose OPEN, a novel MARL-based algorithm for online planning in multi-UAV pursuit-evasion tasks in unknown environments. It uses collective thrust and body rates control commands as policy outputs, ensuring flexibility and enabling zero-shot transfer to real-world.
- We introduce an attention-based evader-prediction network that integrates predicted evader trajectories into policy inputs to guide cooperative strategies under partial observations, alongside an adaptive environment generator that generates challenging curricula to boost sample efficiency and generalization to unseen scenarios. We integrate a calibrated UAV dynamics model into training and employ a two-stage reward refinement process to ensure RL policy stability for real-world deployment.
- Simulation experiments demonstrate that our method achieves near 100% capture rates across four test scenarios, outperforming all baseline approaches. Furthermore, we successfully deploy the multi-UAV pursuit policy on real Crazyflie quadrotors in a zero-shot manner.

II. Related Work

In pursuit-evasion games, one or more pursuers aim to capture one or more evaders, while the evaders actively avoid capture, distinguishing this from search strategies where the target is passive [28]. Traditional research in this area can be broadly categorized into game-theoretic, control-theoretic, and heuristic approaches [14]. Game-theoretic methods simplify the problem into mathematical models, primarily focusing on differential games [4]. Control-theoretic approaches model the nonlinear dynamics of the pursuit, optimizing strategies using the Hamilton-Jacobi-Isaacs equation [5], [6], [7]. Heuristic methods design forces to guide pursuers—such as attraction to the evader and repulsion from teammates and obstacles—and often utilize optimization techniques like particle swarm optimization (PSO) [10] and artificial potential fields (APF) [13]. However, these traditional methods have limitations in real-world applications due to the gap between their simple system modeling and real UAVs, and the difficulty of handling complex cooperative strategies.

Reinforcement learning (RL) has emerged as a powerful data-driven approach for solving multi-agent coordination problems in pursuit-evasion games [20], [21], [22], [23], [24], [25], [29]. Several studies have applied RL to enhance traditional methods [25] or address issues such as varying agent numbers [21], efficient communication [20], credit assignment [19] or perception uncertainty [29]. However, most RL research remains limited to simplified simulation environments, where agents are modeled as particles or constrained by limited kinematics, reducing the effectiveness of these strategies in real-world scenarios. While recent work has explored deploying RL-based strategies in pursuit-evasion tasks [9], [25], [26], [27], these efforts have largely been confined to two-dimensional environments, such as ground vehicles, surface vehicles or UAVs at fixed altitudes.

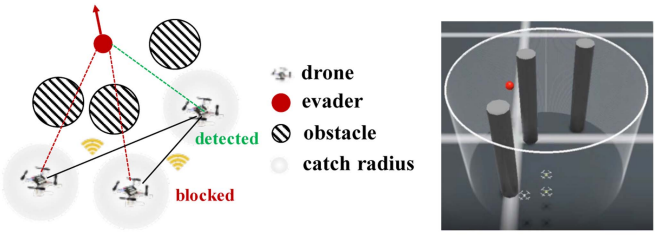
In this paper, we consider the UAV dynamics model and physical constraints, introducing an evader-prediction enhanced network and an adaptive environment generator to address the challenges of large exploration space and policy generalization to unseen scenarios. The learned RL policy is capable of zero-shot deployment to real quadrotors.

III. Preliminary

A. Pursuit-Evasion Problem

As shown in Fig. 1(a), the multi-UAV pursuit-evasion problem involves N UAVs chasing a faster evader in an obstacle-filled environment. The UAVs aim to capture the evader quickly while avoiding obstacles. The evader is captured if it comes within the capture radius of a UAV. Detection is possible only if no obstacles block the line of sight. When detected, the UAV can pass information to its teammates. The non-strategic evader, controlled using a potential field method [27], experiences repulsive forces from UAVs, obstacles, and arena boundaries, with the magnitude of the force inversely proportional to the distance. The evader moves at a constant speed of v_e, with its direction determined by the resultant of these repulsive forces.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.



(a) Partially observable scenario (b) Sim. environment

Fig. 1. Setup of multi-UAV pursuit-evasion.

B. Problem Formulation

We formulate the multi-UAV pursuit-evasion task as decentralized, partially observable Markov decision processes (Dec-POMDPs), $M =< \mathcal{N}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{S}_0, \mathcal{P}, \mathcal{R}, \gamma >$ with the state space $\mathcal{S}$, the joint action space $\mathcal{A}$, the observation space $\mathcal{O}$, the space of initial states $\mathcal{S}_0$, the transition probability $\mathcal{P}$, reward function $\mathcal{R}$ and the discount factor γ. $\mathcal{N} \equiv \{1, \ldots, N\}$ is a set of N UAVs. $o_i = \mathcal{O}(s; i)$ denotes the observation for the UAV i under state $s \in \mathcal{S}$.

The goal of our work is to construct a policy that performs well across diverse scenarios. The task space $\mathcal{T}$ defines a series of Dec-POMDPs with similar properties. We use a parameter space $\mathcal{W}$ to represent the inter-task variation of $\mathcal{T}$. We generate the corresponding initial states by a task parameter $w \in \mathcal{W}$. In our setup, w comprises the initial positions of the UAVs and the evader, as well as the number and positions of obstacles. We consider homogeneous UAVs and learn a parameter-sharing policy π_θ parameterized by θ to output action $a_i \sim \pi_\theta(\cdot|o_i)$ for UAV i. The final objective $J(\theta)$ is to maximize the expected accumulative reward for any task parameter $w \in \mathcal{W}$, i.e., $J(\theta) = \mathbb{E}_{w\sim\mathcal{W}}[\sum_t \gamma^t R(s^t, \boldsymbol{a}^t; w) \mid w]$, where $\boldsymbol{a}^t$ is joint actions at timestep t.

C. Quadrotor Model

In this paper, we use quadrotors, one of the most widely used UAV, as our physical platform. The quadrotor is assumed to be a 6 °-of-freedom rigid body of mass m and diagonal moment of inertia matrix $\boldsymbol{I} = diag(I_x, I_y, I_z)$. The dynamics of the quadrotor are modeled by the differential equation: $\dot{\boldsymbol{x}} = [\dot{\boldsymbol{p}}^T, \dot{\boldsymbol{q}}^T, \dot{\boldsymbol{v}}^T, \dot{\boldsymbol{\omega}}^T]$, where the quadrotor state $\boldsymbol{x} \in \mathbb{R}^{13}$ consists the position $\boldsymbol{p}$, the orientation $\boldsymbol{q}$ in quaternions, the linear velocity $\boldsymbol{v}$ and the angular velocity $\boldsymbol{\omega}$. The acceleration of the quadrotor is described as

$$\dot{\boldsymbol{v}} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ \sum_j \boldsymbol{f}_j / m \end{bmatrix}, \tag{1}$$

in the world frame with gravity g. $\boldsymbol{f}_j$ is the force generated by the j-th rotor. R is the rotation matrix from the body frame to the world frame.

The angular acceleration calculated by Euler's rotation equation in the body frame is $\dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega} \times (\boldsymbol{I}\boldsymbol{\omega}))$. The torques $\boldsymbol{\tau}$ acting in the body frame are determined by $\boldsymbol{\tau} = \sum_j \boldsymbol{\tau}_j + \boldsymbol{r}_{pos,j} \times \boldsymbol{f}_j$, where $\boldsymbol{\tau}_j$ is the torques generated by the j-th rotor, $\boldsymbol{r}_{pos,j}$ is the position of the rotor j expressed in the body frame. We model the rotational speeds of the four motors Ω_j as first-order system with time constant T_m where the commanded motor speeds Ω_{cmd} are the input, i.e., $\dot{\Omega}_j = T_m(\Omega_{cmd,j} - \Omega_j)$. The force and torque produced by the j-th rotor are modeled as follows: $\boldsymbol{f}_j = k_f {\Omega_j}^2, \boldsymbol{\tau}_j = k_m {\Omega_j}^2$.

IV. Methodology

Our method's pipeline is illustrated in Fig. 2. To bridge the sim-to-real gap, we calibrate quadrotor dynamics via system identification and integrate them into a GPU-parallel simulator [30]. We use collective thrust and body rates (CTBR) as control commands and train RL policy to output it via a SOTA MARL algorithm, MAPPO [31]. To model the evader's future trajectory and guide cooperative strategies under partial observability, we design an *Evader Prediction-Enhanced Network* that leverages an attention-based architecture to capture the interrelations within observations and a trajectory prediction network to forecast the evader's movement. To further enhance policy generalization to different scenarios, we propose an *Adaptive Environment Generator* to automatically generate appropriately challenging curricula, enabling efficient exploration of the entire task space. Finally, with the two-stage reward refinement, the learned policy is applied directly to real quadrotors without any real data fine-tuning. The following sections detail the MARL setup, the *Evader Prediction-Enhanced Network*, the *Adaptive Environment Generator*, and the two-stage reward refinement.

A. Multi-Agent Reinforcement Learning Setup

Observation Space: The observation $\mathbf{o}_i$ for quadrotor i is composed of three components: the self-information $\mathbf{o}_{self}$, states of other quadrotors $\mathbf{o}_{other}$ and information about obstacles $\mathbf{o}_{ob}$. $\mathbf{o}_{self}$ consists of its orientation in quaternions, the linear velocity, the real and predicted future relative position to the evader. If the evader is not detected by any of the quadrotors, we mask the real relative position with a fixed marker value which is -5 in our setup. $\mathbf{o}_{other}$ contains the relative positions to other quadrotors. $\mathbf{o}_{ob}$ denotes the relative positions to the k-nearest obstacles, where k is smaller than the maximum number of obstacles, representing the quadrotor's inability to sense global information. In our setup, $k = 3$.

Action Space: To achieve agile control and facilitate highly cooperative decision-making, we employ collective thrust and body rates (CTBR) commands as the policy actions. These CTBR commands are subsequently translated into precise motor inputs by a low-level PID controller. Specifically, the action for quadrotor i is defined as $a_i = (F, \omega_{\text{roll}}, \omega_{\text{pitch}}, \omega_{\text{yaw}})$, where $F \in [0, 1]$ represents the normalized collective thrust, and $\omega \in [-\pi, \pi]\, rad/s$ denotes the body rates along the roll, pitch, and yaw axes, respectively.

Reward Function: The team-based reward function promotes cooperative evader capture and obstacle avoidance, comprising four components: capture reward, distance reward, collision penalty, and smoothness reward. The capture reward incentivizes teamwork, granting all quadrotors a +6 bonus if the evader enters any quadrotor's capture radius. The distance reward, with a coefficient of -0.1, penalizes quadrotors based on their distance to the evader, encouraging closer proximity. A collision penalty of -10 is imposed if quadrotors breach a safety threshold near obstacles, ensuring safe capture. The smoothness reward regularizes policy outputs to ensure reliable behavior during deployment, as detailed in Section IV-D.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

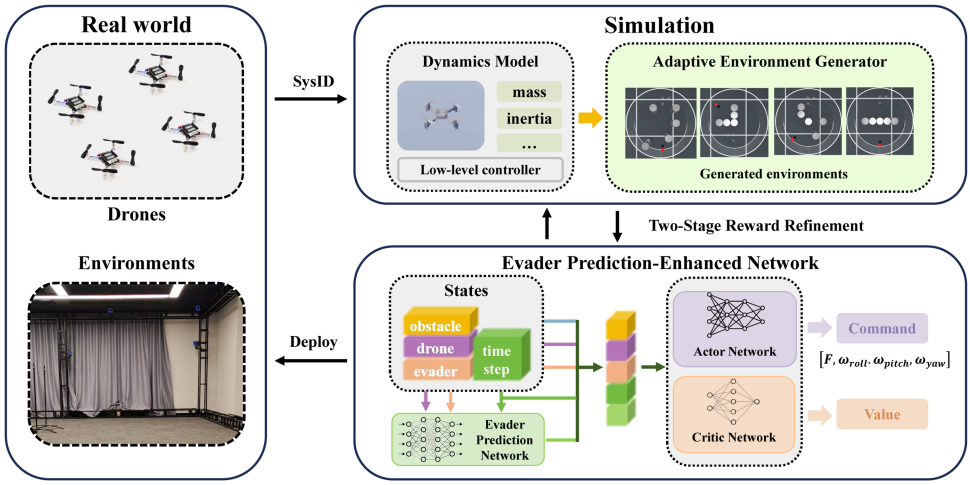


Fig. 2. Pipeline of our method. We begin by calibrating the parameters of the quadrotor dynamics model through system identification. Next, we introduce an *adaptive environment generator* to automatically generate tasks for policy training and employ an *evader prediction-enhanced network* for cooperative capture under partial observations. Finally, with two-stage reward refinement, the learned policy is directly transferred to real quadrotors in a zero-shot manner.

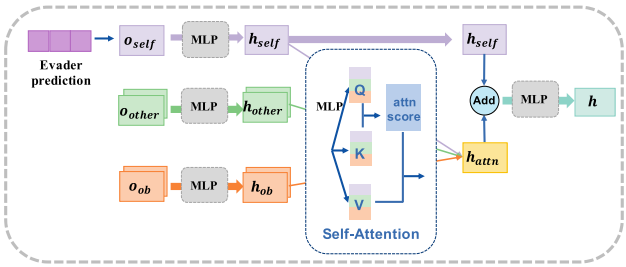


Fig. 3. Attention-based observation encoder.

Algorithm 1: Training With Adaptive Environment Generator.

Input: θ, p, $\mathcal{Q}_{active}$, σ_{min}, σ_{max};
Output: final policy π_θ;
$\mathcal{Q}_{active} \leftarrow \{\}$;
repeat
  // Local expansion and global exploration.
  $W_{env} \leftarrow$ sample tasks from $\mathcal{Q}_{active}$ with probability p and obtain expanded tasks w by **Expand**, else $w \sim \text{Unif}(\mathcal{W})$ with probability $1-p$;
  // Train the policy π_θ.
  Train and evaluate π_θ with W_{env} via MARL;
  // select task parameters from W_{env}
  $W_{new} \leftarrow$ **Selection**$(W_{env}, \sigma_{min}, \sigma_{max})$;
  Add W_{new} to the active archive $\mathcal{Q}_{active}$;
until π_θ *converges*;

B. Evader Prediction-Enhanced Network

In the evader prediction-enhanced network, we build an *Evader Prediction Network* that predicts the evader's future trajectories based on historical observations, facilitating the learning of cooperative pursuit strategies under partial observation. The predicted trajectories, concatenated with the raw observations, are then fed into the *Actor and Critic Networks*. These networks utilize an attention-based observation encoder to better capture the relationships between different entities.

1) Evader Prediction Network: We use an LSTM to predict the evader's trajectory for the next K timesteps. The input consists of n-step historical data, including the positions of all quadrotors, the positions and velocities of the evader, and the current timestep. If the evader is blocked by an obstacle and undetected by any quadrotor, we use a marker value to replace the evader's positions and velocities. To train the network, we collect data over $n+K$ timesteps during the rollout phase, using the first n timesteps as inputs X and the subsequent $n+1$ to $n+K$ timesteps as labels Y. The evader prediction network P_ϕ is trained via supervised learning, and the loss is defined as $L_\phi = \mathbb{E}(Y - P_\phi(X))^2$.

2) Actor and Critic Networks: The actor and critic networks are based on the attention-based observation encoder, as illustrated in Fig. 3. The raw observation of quadrotor i consists of three components: $\mathbf{o}_{self}$, $\mathbf{o}_{other}$, and $\mathbf{o}_{ob}$. The predicted trajectory of the evader is concatenated into $\mathbf{o}_{self}$. Each component is separately encoded using distinct MLPs, producing embeddings of 128 dimensions each. A multi-head self-attention module is employed to capture the relationships between these embeddings, resulting in features $\mathbf{h}_{attn}$. To emphasize self-information, the self-embeddings $\mathbf{h}_{self}$ are added to $\mathbf{h}_{attn}$ and passed through an MLP to generate the final feature $\mathbf{h}$. In the actor network, actions are parameterized using a Gaussian distribution based on $\mathbf{h}$. For the critic network, $\mathbf{h}$ is input into an MLP to produce a scalar value representing the estimated state value.

C. Adaptive Environment Generator

Our goal is to derive an RL policy that generalizes effectively to unknown environments. While domain randomization [32] is a common technique for improving generalization, its reliance on randomizing task parameters often requires a large number of samples to achieve reliable results. This approach becomes sample-inefficient and struggles with complex or rare

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

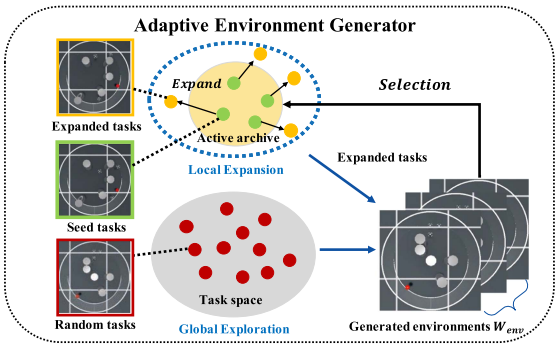


Fig. 4. Design of the *adaptive environment generator*.

corner cases, where underrepresented environmental configurations lead to suboptimal performance in unseen scenarios (see Section V-D). In contrast, our adaptive environment generator efficiently explores the task space by focusing on corner cases and rare scenarios, integrating this dynamic approach into MARL training, as detailed in Algorithm 1.

We leverage two inductive biases: (1) different obstacle configurations require distinct pursuit strategies, and (2) under the same obstacle setup, the initial positions of UAVs and the evader significantly affect capture difficulty. Based on these insights, we decompose the task space into two modules: *Local Expansion*, which explores UAV and evader positions under fixed obstacles, and *Global Exploration*, which investigates diverse obstacle configurations to generate simulation environments (Fig. 4).

The *Local Expansion* module enhances the quadrotors' ability to capture the evader from any initial position within a fixed obstacle setup. It maintains an active archive $\mathcal{Q}_{active}$ of task parameters w for unsolved environments. Tasks are expanded from seed tasks sampled from $\mathcal{Q}_{active}$ by adding noise $\epsilon \in [-\delta, \delta]$ to UAV and evader positions while keeping obstacles unchanged. The archive is updated by evaluating the policy in generated environments W_{env} and retaining tasks with success rates $c \in [\sigma_{min}, \sigma_{max}]$, where $\sigma_{min} = 0.5$ and $\sigma_{max} = 0.9$. This process gradually increases task complexity, enabling automatic curriculum generation.

The *Global Exploration* module explores unseen obstacle configurations by randomly sampling task parameters from the entire parameter space $\mathcal{W}$, including the number and positions of obstacles, as well as the initial positions of the UAVs and the evader. To balance focused and diverse task generation, we combine fast *Local Expansion* (selected with a probability of $p = 0.7$) and slow *Global Exploration* (selected with a probability of $1 - p = 0.3$).

D. Two-Stage Reward Refinement

We use the smoothness reward $e^{-||a^t - a^{t-1}||_2}$, where $\boldsymbol{a}_t$ represents the policy's CTBR output, to balance task completion and action smoothness [33]. To address the challenge of maintaining a feasible policy for real-world deployment while ensuring task success, we employ a two-stage reward refinement process. In the first stage, we exclude the smoothness reward and focus on the other three rewards. In the second stage, we reintroduce the smoothness reward and continue training from the first-stage checkpoint. This ensures high task success while improving action smoothness, with the smoothness reward coefficient set to 2.0.

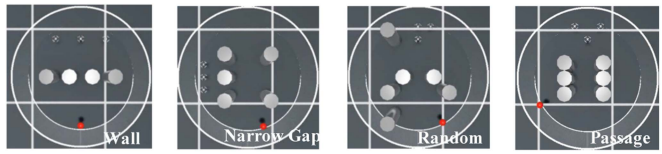


Fig. 5. Illustration of four test scenarios.

V. Experiment

A. Experiment Setting

We use OmniDrones [30], a high-speed UAV simulator, to construct the task. The environment (Fig. 1(b)) is a 0.9-radius, 1.2-height arena with 4–5 randomly placed cylindrical obstacles (radius 0.1, height 1.2). Collisions are avoided by maintaining a minimum distance of 0.07 between UAVs and obstacles. The setup includes 3 quadrotors (max speed 1.0) pursuing an evader (speed 1.3), with a capture radius of 0.3 and a maximum episode length of 800 steps.

B. Evaluation

Evaluation Scenarios: We design four test scenarios (Fig. 5) to comprehensively evaluate our method. The within-distribution scenarios, *Wall* and *Narrow Gap*, challenge pursuit strategies: *Wall* restricts access to lateral paths, while *Narrow Gap* allows the evader to exploit a speed advantage along the perimeter. For out-of-distribution (OOD) evaluation, *Random* tests autonomous exploration by hiding the evader initially, and *Passage* requires strategic interception across three escape routes.

Evaluation Metrics: We use three statistical metrics to evaluate pursuit strategies: Capture Rate, Capture Step, and Collision Rate. Each experiment is averaged over 300 testing episodes (100 per seed). All numerical values are reported as mean (standard deviation).

- *Capture Rate:* Ratio of successful episodes where the evader is captured before the maximum episode length. Higher values indicate better performance.
- *Capture Step:* Average timestep of first capture. If not captured, the maximum episode length is recorded. Lower values indicate quicker capture.
- *Collision Rate:* Average number of collisions between quadrotors and obstacles or other quadrotors per episode length. Lower values indicate safer pursuit.

C. Baselines

We challenge our method with three heuristic methods (Angelani, Janosov, and APF) and two RL-based methods (DACOOP and MAPPO).

- *Angelani* [11]: Pursuers are attracted to the nearest particles of the opposing group, i.e., the evader.
- *Janosov* [12]: Janosov designs a greedy chasing strategy and collision avoidance system that accounts for inertia, time delay, and noise.
- *APF* [13]: APF guides pursuers to a target position by combining attractive, repulsive, and inter-individual forces.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I
RESULTS OF ALL METHODS IN TEST SCENARIOS

Scenarios	Metrics	Angelani	Janosov	APF	DACOOP	MAPPO	OPEN
Wall	*Cap. Rate*↑	0.008(0.003)	0.009(0.002)	0.010(0.005)	0.205(0.007)	0.817(0.095)	**0.987(0.019)**
	Cap. Step↓	798.0(000.8)	797.0(000.8)	797.0(001.4)	744.0(009.1)	569.1(046.9)	**480.4(052.4)**
	Coll. Rate↓	0.010(0.000)	0.010(0.000)	0.010(0.000)	0.040(0.008)	0.010(0.076)	**0.000(0.000)**
Narrow Gap	*Cap. Rate*↑	0.213(0.012)	0.219(0.012)	0.253(0.012)	0.447(0.017)	0.917(0.034)	**1.000(0.000)**
	Cap. Step↓	755.0(004.5)	757.7(004.2)	743.0(002.2)	553.0(002.2)	**484.0(041.2)**	526.0(032.9)
	Coll. Rate↓	0.094(0.001)	0.077(0.001)	0.085(0.000)	0.039(0.002)	0.039(0.032)	**0.017(0.024)**
Random	*Cap. Rate*↑	0.277(0.002)	0.145(0.019)	0.314(0.018)	0.563(0.016)	0.783(0.272)	**1.000(0.000)**
	Cap. Step↓	639.0(001.6)	716.0(012.0)	620.0(014.8)	488.67(008.7)	493.8(183.6)	**329.9(060.6)**
	Coll. Rate↓	0.015(0.001)	0.018(0.002)	0.022(0.002)	0.020(0.000)	0.168(0.160)	**0.011(0.008)**
Passage	*Cap. Rate*↑	0.733(0.010)	0.009(0.003)	0.229(0.027)	0.818(0.027)	0.617(0.275)	**1.000(0.000)**
	Cap. Step↓	552.7(005.0)	799.3(000.5)	750.3(007.4)	533.7(008.8)	541.0(144.9)	**329.6(076.9)**
	Coll. Rate↓	**0.000(0.000)**	0.001(0.000)	0.001(0.000)	0.001(0.000)	0.013(0.008)	0.006(0.008)

Our approach significantly outperforms all baselines in test scenarios.

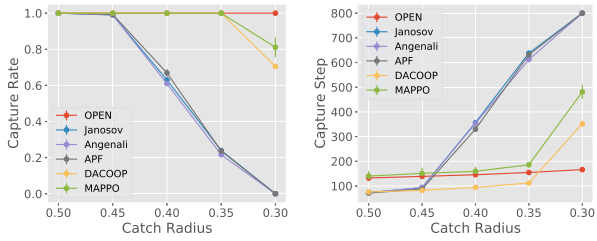
- *DACOOP* [25]: DACOOP employs RL to adjust the primary hyperparameters of APF, enabling effective adaptation to diverse scenarios. For a fair comparison, we replace D3QN [34] in the original paper with a SOTA MARL algorithm, MAPPO.
- *MAPPO* [31]: MAPPO is a SOTA multi-agent reinforcement learning algorithms for general cooperative problems. We regard MAPPO as a strong baseline for policy training, particularly in tasks where no additional specialized techniques are employed.

For heuristic methods, we perform a grid search on hyperparameters in the training scenarios. For DACOOP, we carefully tune the hyperparameters and use the best results. Baselines, treating the quadrotor as a point mass, outputs velocity commands executed by a velocity controller.

D. Main Results in Simulation

Quantitative Results: Table I presents performance for the four test scenarios, focusing on corner cases. In *Wall*, our method achieves over 98% capture rate, with the lowest collision rate and fewest capture steps. In *Narrow Gap*, it maintains a 100% capture rate and the lowest collision rate, despite slightly higher capture steps than MAPPO. These within-distribution results highlight our method's effectiveness in corner cases and efficient agent cooperation. In *Random* and *Passage*, our method achieves 100% capture rates, outperforming baselines (78.3% and 81.8%), and requires the fewest capture steps. While the collision rate in *Passage* is slightly higher (0.6%), the results demonstrate strong generalization to unseen scenarios.

Given the poor baseline performance, we further test the algorithms in a simpler, obstacle-free scenario. As shown in Fig. 6, baselines exhibit a sharp decline in capture rate as the capture radius decreases, underscoring the task's difficulty. A smaller radius demands more agile UAV behavior, requiring rapid formation adjustments to block all evader escape routes. However, heuristic algorithms and DACOOP, which rely on force-based adjustments for obstacle avoidance and pursuit, inherently limit UAV agility. When the capture radius is small (e.g., 0.3), MAPPO struggles to learn an optimal capture strategy, with the success rate converging to approximately 80%. In contrast,



(a) Capture rate. (b) Capture timestep.

Fig. 6. Results of all methods in a obstacle-free scenario.

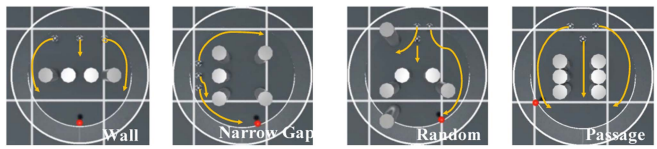
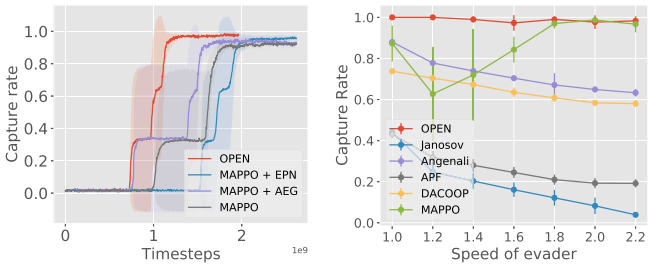


Fig. 7. Demonstration of the pursuit strategies.

our method maintains high capture rates with only a slight increase in capture steps as the radius shrinks, demonstrating the superior cooperative capture capability of pure RL strategies.

Behavior Analysis: As shown in Fig. 7, we identify four emergent behaviors in test scenarios, highlighting our strategy's cooperative pursuit capabilities (see supplementary video). In *Wall*, our method employs a double-sided surround strategy: one quadrotor monitors while two flank the evader, unlike baselines hindered by frontal obstacles. In *Narrow Gap*, our method intercepts the evader by taking shortcuts, unlike baselines that only follow. In *Random*, guided by predicted trajectories, our method quickly locates the evader behind obstacles, while baselines fail initially. In *Passage*, our method divides quadrotors to block all escape routes, whereas baselines greedily approach, leaving routes open.



(a) Training curves. (b) Policy Gen. across v_e.

Fig. 8. Ablation studies. (a) Performance of OPEN and baselines on training tasks. (b) Policy generalization across evader speeds in the *passage* scenario.

TABLE II
PREDICTION ERRORS OF OUR METHOD IN FOUR TEST SCENARIOS

Scenarios	*Wall*	*Narrow Gap*	*Random*	*Passage*
Pred. Error	0.054(0.013)	0.084(0.007)	0.109(0.014)	0.081(0.036)

E. Ablation Studies

We conduct ablation studies on the core modules of our method. The baseline, denoted as "MAPPO", represents the naive implementation without the *Evader Prediction-Enhanced Network* (EPN) and *Adaptive Environment Generator* (AEG) modules. The variant "MAPPO + EPN" incorporates the *Evader Prediction-Enhanced Network* module into the MAPPO framework, while "MAPPO + AEG" combines MAPPO with the *Adaptive Environment Generator* module.

1) Performance on Training Tasks: As shown in Fig. 8(a), our method demonstrates the highest capture performance and improves sample efficiency by more than 50% compared to MAPPO. "MAPPO" achieves only a 90% capture rate with over 2.0 billion samples, highlighting the challenging nature of deriving an RL policy that considers UAV dynamics and performs well across diverse scenarios. And as shown in Table I, MAPPO's average performance on the training tasks is suboptimal, leading to significantly worse performance in corner cases such as the *Wall* and *Narrow Gap* scenarios.

For "MAPPO + EPN", the training curves show that while the EPN module slightly increases training data requirements, it improves final performance over MAPPO. This is due to the additional computational overhead required to train the *Evader Prediction-Enhanced Network*. Once the network converges and accurately predicts the evader's future position, "MAPPO + EPN" achieves superior performance. We also report the average prediction error (Pred. Error) of the evader's next position within an episode for the four test scenarios. Even in the *Random* scenario, where the evader is initially invisible, the prediction error is comparable to the UAV size and much smaller than the capture radius, validating the predictions. For "MAPPO + AEG", the training curves show that the AEG module significantly improves training efficiency and slightly enhances final performance compared to MAPPO.

2) Performance on Out-of-Distribution Scenarios: We further evaluate our method and its variants in out-of-distribution scenarios. As shown in Table III, our method achieves the highest capture rate and the lowest collision rate in the *Random* scenario, with capture timesteps comparable to"MAPPO + EPN". In the *Passage* scenario, our method consistently outperforms all variants. The EPN module provides UAVs with the capability to predict the evader's strategy, allowing them to plan captures based on the evader's future trajectory. This significantly reduces the impact of OOD scenarios on policy performance. Additionally, the AEG module enhances the policy's ability to address corner cases, promoting better generalization across diverse environments. By integrating these two modules, our method surpasses all baseline variants.

TABLE III
COMPARISON OF OPEN AND VARIANTS IN OOD SCENARIOS

Methods	Metric	OPEN	MAPPO + EPN	MAPPO + AEG	MAPPO
Random	*Cap. Rate*↑	**1.000(0.000)**	**1.000(0.000)**	0.967(0.025)	0.783(0.272)
	Cap. Step↓	329.9(060.6)	**301.0(028.7)**	477.1(044.8)	493.8(183.6)
	Coll. Rate↑	**0.011(0.008)**	0.012(0.009)	0.107(0.110)	0.168(0.160)
Passage	*Cap. Rate*↑	**1.000(0.000)**	0.886(0.120)	0.727(0.193)	0.617(0.275)
	Cap. Step↓	**329.6(076.9)**	427.6(123.0)	602.2(041.1)	541.0(144.9)
	Coll. Rate↑	**0.006(0.008)**	0.013(0.010)	0.216(0.292)	0.013(0.008)

3) Policy Generalization Across Evader Speeds v_e: As shown in Fig. 8(b), we evaluate the sensitivity of OPEN and baselines to varying evader speeds in the *Passage* scenario. Unlike obstacle-free settings, *Passage* allows the evader to exploit obstacle occlusion and speed advantages. As speed increases, heuristic baselines and DACOOP show significant capture rate drops due to poor early interception. As the evader's speed increases, MAPPO's capture rate first drops then rises, as it learns a three-way interception strategy similar to OPEN. At higher speeds, the evader is more likely to be caught in its blind spot during turns, lacking time to adjust direction. In contrast, our method maintains robust performance across speeds, achieving high capture rates for faster evaders without retraining, despite being trained only for $v = 1.3$.

F. Real-World Experiments

We deploy the policy on three real CrazyFlie 2.1 quadrotors, each with a maximum speed of 1.0 m/s. A motion capture system provides the quadrotors' states, while a virtual evader's true states are input into the actor and evader prediction network upon detection. The actor and evader prediction network run on a local computer, sending CTBR control commands at 100 Hz to the quadrotors via radio. We align the initial conditions of the simulation with real-world to ensure fair comparison. All real experiments are repeated 5 times and recorded as the mean and standard deviation.

We compare the two-stage reward refinement (*Two-stage*) with its variants. We define the first stage of *Two-stage* as *w/o Smoothness*, training only with task rewards. So the simulation results of *w/o Smoothness* can be considered as *oracle*. *One-stage* denotes training with both smoothness and task rewards. As shown in Table IV, the results demonstrate that *w/o Smoothness* fails completely in real-world experiments due to overly aggressive pursuit strategies which lead to unstable behaviors and UAV crashes. *One-stage* causes severe exploration challenges in MARL training, resulting in no policy improvement during training and yielding only trivial stationary strategies. In contrast, only the *Two-stage* achieves successful real-world deployment while preserving capture performance approximately consistent with simulation results.

TABLE IV
CAPTURE STEPS OF ALL METHODS IN FOUR TEST SCENARIOS

Methods	Envs.	*Wall*	*Narrow Gap*	*Random*	*Passage*
w/o Smoothness (*oracle*)	Sim.	258.6(0.00)	478.0(0.00)	361.9(0.00)	362.0(0.00)
w/o Smoothness	Real.	fail	fail	fail	fail
One-stage	Real.	/	/	/	/
Two-stage	Real.	320.2(26.8)	441.7(82.1)	385.2(29.4)	377.5(49.2)

"/" notes that the one-stage fails to generate effective pursuit strategies in the simulation and we do not conduct experiments in the real-world.

VI. CONCLUSION

We propose a feasible RL policy for online planning in unknown environments, enabling real-world deployment on Crazyflie quadrotors for multi-UAV pursuit-evasion. To promote generalization, we introduce an Adaptive Environment Generator for automatic curriculum generation and a Prediction-Enhanced Network for cooperative capture. Our method outperforms baselines in simulation and generalizes well to unseen scenarios. A two-stage reward refinement ensures physical feasibility and enables zero-shot real-world transfer.

Our method still has some limitations. The policy supports varying drone numbers but still needs retraining for performance. Moreover, future work will explore vision-based methods for unstructured environments.

REFERENCES

[1] V. Turetsky and J. Shinar, "Missile guidance laws based on pursuit–evasion game formulations," *Automatica*, vol. 39, no. 4, pp. 607–618, 2003.

[2] J. M. Eklund, J. Sprinkle, and S. S. Sastry, "Switched and symmetric pursuit/evasion games using online model predictive control with application to autonomous aircraft," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 3, pp. 604–620, May 2012.

[3] D. W. Oyler, P. T. Kabamba, and A. R. Girard, "Pursuit–evasion games in the presence of obstacles," *Automatica*, vol. 65, pp. 1–11, 2016.

[4] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry, "Probabilistic pursuit-evasion games: Theory, implementation, and experimental evaluation," *IEEE Trans. Robot. Automat.*, vol. 18, no. 5, pp. 662–669, Oct. 2002.

[5] D. Ye, M. Shi, and Z. Sun, "Satellite proximate pursuit-evasion game with different thrust configurations," *Aerosp. Sci. Technol.*, vol. 99, 2020, Art. no. 105715.

[6] X. Fang, C. Wang, L. Xie, and J. Chen, "Cooperative pursuit with multi-pursuer and one faster free-moving evader," *IEEE Trans. Cybern.*, vol. 52, no. 3, pp. 1405–1414, Mar. 2022.

[7] B. Tian, P. Li, H. Lu, Q. Zong, and L. He, "Distributed pursuit of an evader with collision and obstacle avoidance," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 13512–13520, Dec. 2022.

[8] H. Huang, W. Zhang, J. Ding, D. M. Stipanović, and C. J. Tomlin, "Guaranteed decentralized pursuit-evasion in the plane with multiple pursuers," in *Proc. 2011 50th IEEE Conf. Decis. Control Eur. Control Conf.*, 2011, pp. 4835–4840.

[9] A. Pierson, Z. Wang, and M. Schwager, "Intercepting rogue robots: An algorithm for capturing multiple evaders with multiple pursuers," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 530–537, Apr. 2017.

[10] R. Palm and A. Bouguerra, "Particle swarm optimization of potential fields for obstacle avoidance," in *Proc. Recent Adv. Robot. Mechatron.*, 2013, pp. 117–123.

[11] L. Angelani, "Collective predation and escape strategies," *Phys. Rev. Lett.*, vol. 109, no. 11, 2012, Art. no. 118104.

[12] M. Janosov, C. Virágh, G. Vásárhelyi, and T. Vicsek, "Group chasing tactics: How to catch a faster prey," *New J. Phys.*, vol. 19, no. 5, 2017, Art. no. 053003.

[13] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. ICRA*, 1991, vol. 2, no. 1991, pp. 1398–1404.

[14] Z. Mu, J. Pan, Z. Zhou, J. Yu, and L. Cao, "A survey of the pursuit–evasion problem in Swarm intelligence," *Front. Inf. Technol. Electron. Eng.*, vol. 24, no. 8, pp. 1093–1116, 2023.

[15] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Proc. Auton. Agents Multiagent Syst.*, AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8–12, 2017, Revised Sel. Papers 16, 2017, pp. 66–83.

[16] S. F. Desouky and H. M. Schwartz, "Q (λ)-learning adaptive fuzzy logic controllers for pursuit–evasion differential games," *Int. J. Adaptive Control Signal Process.*, vol. 25, no. 10, pp. 910–927, 2011.

[17] M. D. Awheda and H. M. Schwartz, "The residual gradient FACL algorithm for differential games," in *Proc. 2015 IEEE 28th Can. Conf. Elect. Comput. Eng.*, 2015, pp. 1006–1011.

[18] D. Luo, Z. Fan, Z. Yang, and Y. Xu, "Multi-UAV cooperative maneuver decision-making for pursuit-evasion using improved MADRL," *Defence Technol.*, vol. 35, pp. 187–197, 2024.

[19] W. Gan, X. Qu, D. Song, and P. Yao, "Multi-USV cooperative chasing strategy based on obstacles assistance and deep reinforcement learning," *IEEE Trans. Automat. Sci. Eng.*, vol. 21, no. 4, pp. 5895–5910, Oct. 2024.

[20] Y. Wang, L. Dong, and C. Sun, "Cooperative control for multi-player pursuit-evasion games with reinforcement learning," *Neurocomputing*, vol. 412, pp. 101–114, 2020.

[21] L. Xu, B. Hu, Z. Guan, X. Cheng, T. Li, and J. Xiao, "Multi-agent deep reinforcement learning for pursuit-evasion game scalability," in *Proc. 2019 Chin. Intell. Syst. Conf., Vol. I 15th*, 2020, pp. 658–669.

[22] M. Hüttenrauch, S. Adrian, and G. Neumann, "Deep reinforcement learning for Swarm systems," *J. Mach. Learn. Res.*, vol. 20, no. 54, pp. 1–31, 2019.

[23] N. -M. T. Kokolakis and K. G. Vamvoudakis, "Safety-aware pursuit-evasion games in unknown environments using Gaussian processes and finite-time convergent reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 3, pp. 3130–3143, Mar. 2024.

[24] D. Liu, Q. Zong, X. Zhang, R. Zhang, L. Dou, and B. Tian, "Game of drones: Intelligent online decision making of multi-UAV confrontation," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 8, no. 2, pp. 2086–2100, Apr. 2024.

[25] Z. Zhang, X. Wang, Q. Zhang, and T. Hu, "Multi-robot cooperative pursuit via potential field-enhanced reinforcement learning," in *Proc. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 8808–8814.

[26] R. Zhang, Q. Zong, X. Zhang, L. Dou, and B. Tian, "Game of drones: Multi-UAV pursuit-evasion game with online motion planning by deep reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7900–7909, Oct. 2023.

[27] C. D. Souza, R. Newbury, A. Cosgun, P. Castillo, B. Vidolov, and D. Kulić, "Decentralized multi-agent pursuit using deep reinforcement learning," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4552–4559, Jul. 2021.

[28] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion in a polygonal environment," *IEEE Trans. Robot.*, vol. 21, no. 5, pp. 875–884, Oct. 2005.

[29] Y. Cheng, J. Zha, R. Yang, Z. Sun, S. Xu, and X. Chen, "Multi-agent target pursuit using perception uncertainty-aware reinforcement learning," in *Proc. 30th Annu. Int. Conf. Mobile Comput. Netw.*, 2024, pp. 1992–1997.

[30] B. Xu, F. Gao, C. Yu, R. Zhang, Y. Wu, and Y. Wang, "Omnidrones: An efficient and flexible platform for reinforcement learning in drone control," 2023, *arXiv:2309.12825*.

[31] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of PPO in cooperative, multi-agent games," 2021, *arXiv:2103.01955*.

[32] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. 2017 IEEE/RSJ Int. Conf. Intell. Robots Syst.* 2017, pp. 23–30.

[33] J. Chen et al., "What matters in learning a zero-shot sim-to-real RL policy for quadrotor control? A comprehensive study," 2024, *arXiv:2412.11764*.

[34] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.