Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Multi-Agent Deep Reinforcement Learning for Cooperative Path Planning of UAV Swarms

Pingping Qu
Shenyang Aerospace University
Huan Liu
Shenyang Aerospace University
Song Xu
Shenyang Aerospace University
Tengli Yu
Shenyang Aerospace University
Yunhao Chen
cyh_619@126.com

Yunnan Minzu University
Ershen Wang
Shenyang Aerospace University
La Na
Shenyang Aerospace University

Research Article

Keywords: UAV swarm, path planning, multi-agent reinforcement learning, partially observable Markov decision process, reward shaping

Posted Date: October 16th, 2025

DOI: https://doi.org/10.21203/rs.3.rs-6508231/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Multi-Agent Deep Reinforcement Learning for Cooperative Path Planning of UAV Swarms

Pingping Qu[1,2], Huan Liu[1], Song Xu[1], Tengli Yu[3], Yunhao Chen[4*], Ershen Wang[1*], La Na[1]

[1]School of Electronic and Information Engineering, Shenyang Aerospace University, Shenyang, 110136, Liaoning, China.
[2]State key Laboratory of Extreme Environment Optoelectronic Dynamic Testing Technology and Instrument , North University of China, Taiyuan, 030051,Shanxi, China.
[3]College of Aeronautics and Astronautics, Shenyang Aerospace University, Shenyang, 110136, Liaoning, China.
[4]Yunnan Key Laboratory of Unmanned Autonomous Systems, Yunnan Minzu University, Kunming, 650504, Yunnan, China.

*Corresponding author(s). E-mail(s): cyh_619@126.com; wes2016@sau.edu.cn;

Abstract

Collaborative path planning for UAV swarms in dynamic uncertain environments faces dual challenges of partial observability and cooperation mechanism design. The decentralized decision-making nature of multi-agent reinforcement learning (MARL) establishes a novel theoretical framework for autonomous coordination of heterogeneous UAV swarms under partially observable conditions. This paper proposes a reciprocity reward-enhanced multi-agent deep reinforcement learning method (PMI-MADDPG) that optimizes UAV cooperative decision-making through a centralized training with decentralized execution framework. By constructing a partially observable Markov decision process (POMDP) model, we design continuous action spaces considering UAV kinematic constraints, and quantify inter-agent state dependencies using pointwise mutual information. A novel cooperative coefficient estimation network is introduced to dynamically balance individual rewards and swarm-level objectives. Simulation results demonstrate that compared to conventional multi-agent methods, PMI-MADDPG shows significant advantages in task reward acquisition and network convergence efficiency, while revealing the impact of UAV quantity on system stability. The

proposed approach provides an innovative solution for cooperative path planning tasks of UAV swarms in complex environments.

Keywords: UAV swarm, path planning, multi-agent reinforcement learning, partially observable Markov decision process, reward shaping

1 Introduction

In recent years, the rapid advancement of intelligent decision-making technologies and expansion of the low-altitude economy have significantly enhanced the utilization of unmanned aerial vehicles (UAVs). Unmanned aerial vehicles have achieved groundbreaking applications in civilian domains such as precision agricultural spraying [1, 2], mining terrain mapping[3], and forest fire search-and-rescue [4], as well as military scenarios including electronic warfare [5] and cooperative strike [6]. However, collaborative path planning for UAV swarms in dynamic environments remains a critical challenge, especially when facing partial observability and non-stationary tar-gets.

The ultimate goal of path planning is to ensure that the UAV reaches the target loca-tion safely and efficiently while optimizing the path length. Traditional path planning al-gorithms have relatively simple principles and can meet most path planning requirements. For example, in graph-based search algorithms, the Dijkstra algorithm [7] determines the optimal path by traversing all nodes forward, while the A* algorithm [8] improves search efficiency by introducing an evaluation function to expand nodes. The D* algorithm [9] further enhances the path planning process by first computing an initial path to the target and then dynamically updating nodes using an evaluation function to adjust the next stage of the path based on the current state. However, since path planning is a typical NP-hard problem [10], the computational complexity of traditional algorithms increases significantly in complex environments, making them susceptible to the curse of dimensionality. Heuristic path planning algorithms can effectively mitigate the curse of dimensionality. For example, particle swarm optimization, ant colony optimization, and artificial bee colony algorithms [11] have been increasingly applied to path planning tasks. Reference [12] proposes an improved hybrid particle swarm optimization algorithm that integrates the simulated annealing mechanism to enhance path planning performance in complex environments. However, both traditional and heuristic algorithms are prone to local optima and are more suitable for static path planning. In complex environments with unknown global information and dynamic path planning problems, these algorithms often struggle to adapt to environmental changes effectively and fail to guarantee the global optimal solution.

The design of deep reinforcement learning (DRL) methods has become a popular re-search direction in the field of artificial intelligence and has been widely applied in various domains, such as multi-UAV target search [13], UAV power inspection [14], and traffic signal control [15]. DRL [16] is an effective approach for solving the dynamic path planning problem for UAVs. It integrates the decision-making capability of reinforcement learning with the perception ability of deep learning, enabling UAVs to autonomously plan paths in complex and dynamic environments. Reference [17]

employs the Competitive Deep Q-Network (CDQN) algorithm to address collision-avoidance path planning at a fixed altitude. Reference [18] applies an improved Deep Deterministic Policy Gradient (DDPG) algorithm for glider path planning in ocean current environments. However, these methods are all based on single-agent deep reinforcement learning. While they perform well in single-UAV and multi-UAV path planning tasks, they struggle to achieve efficient multi-agent collaboration in cooperative UAV missions. When extended to multi-agent scenarios, these algorithms of-ten lack robustness and scalability, making them unsuitable for large-scale UAV swarm systems.

The development of multi-agent deep reinforcement learning (MADRL)[19, 20] has emerged as a prominent research frontier in autonomous systems, demonstrating considerable potential across diverse application scenarios such as automated traffic control [21] UAV communication system optimization [22], and UAV-assisted Edge Computing [23]. In practical applications, coordinated path planning is essential when multiple UAVs collaborate to accomplish single or multiple tasks. Study [24] implemented distributed UAV trajectory management using the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) framework, with optimization objectives focusing on enhancing user service fairness and minimizing global energy consumption. The Deep Learning-Genetic Algorithm (DL-GA) proposed in [25] synergizes the feature extraction capabilities of deep learning with the global search advantages of genetic algorithms. It employs GA to collect multi-scenario state-path data for training deep neural networks, enabling rapid generation of optimized paths. Although this method significantly improves convergence efficiency, its computational complexity escalates exponentially in highly dynamic environments.

To address the critical challenges in cooperative path planning for UAV swarms, this study proposes an Enhanced Multi-Agent Deep Deterministic Policy Gradient algorithm [26]. The innovative integration of neural network-based positive pointwise mutual information (PPMI) [27] estimation enables intelligent path decision-making through a reward reshaping mechanism. Compared with prior works, the principal contributions of this research are threefold:

• Methodological Innovation: Conventional approaches face dual limitations: classical algorithms suffer from the curse of dimensionality, while heuristic methods are prone to local optima, both inadequate for dynamic UAV path planning. The single-agent deep reinforcement learning (DRL) framework inherently lacks multi-agent coordination mechanisms. To overcome these, we establish a distributed decision-making system using the Multi-Agent Deep Reinforcement Learning (MADRL) framework for co-operative path planning in UAV swarms.

• Currently, there are very few open-source reinforcement learning environments avail-able for UAV swarm path planning. To address this gap, this study develops a reinforcement learning environment for UAV swarms within an airspace scenario, featuring a visualized simulation. In this environment, UAV movements are constrained by a dynamical model while incorporating collision avoidance and boundary constraints to ensure safety compliance

• This study formulates the UAV swarm path planning problem as a partially observable Markov decision process (POMDP) and proposes an improved MADDPG

algorithm to solve this model. In the algorithm design, a neural network is employed to estimate pointwise mutual information in the path planning task, and the reward function is reshaped based on this estimation to more effectively capture the relationship between UAV collaboration and mission objectives. This enhancement not only improves the accuracy of the reward signal but also significantly accelerates the algorithm's convergence and enhances its final performance.

The remaining sections of this paper are organized as follows: Section 2 establishes the system model and mathematical formulation for UAV swarm path planning under partial observability. Section 3 introduces the PMI-MADDPG algorithm with its reciprocal reward mechanism and network architecture. Section 4 presents simulation experiments and comparative results analysis. Section 5 concludes the paper and outlines future re-search directions.

2 System Modeling and Decision Framework

2.1 Reinforcement Learning-Based Air-to-Air Path planning Model

The standardized multi-UAV path planning framework establishes a swarm system with agents and target points, where targets are randomly distributed within an unknown operational domainΩ. During the search process, each UAV dynamically selects individual or multiple targets for path planning optimization. All agents navigate in a 2D Euclidean space with constrained mobility $v \leqslant 10$km/h, where motion control inputs are modeled continuously varying stochastic variables to accurately reflect authentic UAV kinematic characteristics. In the multi-UAV path planning framework, each agent is governed by a point-mass kinematic model mathematically expressed as:

$$\begin{cases} \phi = \omega_\phi \, \mathrm{d}t, \quad -10 < \omega_\phi < 10 \\ \omega_\phi = 9.81 \dfrac{m}{F} \cdot \mathrm{d}t \cdot \tan\phi \\ \phi = \omega_\phi \, \mathrm{d}t, \quad -20 < \omega_\phi < 20 \\ v_x = \sin\phi \cdot \dfrac{F}{m} \\ v_y = \cos\phi \cdot \dfrac{F}{m} \\ x = v_x \, \mathrm{d}t \\ y = v_y \, \mathrm{d}t \end{cases} \tag{1}$$

Where ϕ represents the roll angle of the drone, ω_ϕ represents the roll angular velocity, F represents the driving force, ω_ψ represents the yaw angular velocity, θ represents the yaw angle of the drone, m represents the mass of the drone, $\boldsymbol{v_x}$ and $\boldsymbol{v_y}$ are the velocity components of the drone along the X-axis and Y-axis respectively in the two-dimensional plane, x and y represent the positions of the drone along the X-axis and Y-axis in the two-dimensional plane, and $\mathrm{d}t$ represents the differential variable with respect to time t. It is assumed that the action space of the drone is a continuous

space, expressed as a two-dimensional velocity vector $[\boldsymbol{v_x}, \boldsymbol{v_y}]$. The scenario allows the drone to move in any direction at variable speeds.

In order to effectively simulate the movement behavior of actual drones, the drone's action is set as a vector $[\boldsymbol{F}, \boldsymbol{\omega_\phi}]$ composed of the driving force and the roll angular velocity.Then, the flight speed and yaw angle of the drone at time t are obtained by the following equations:

$$\begin{cases} v_{x,i,t} = v_{x,i,t-1} + \sin(\phi_{i,t-1}) \cdot \dfrac{F_{i,t}}{m_i} \cdot \mathrm{d}t \\ v_{y,i,t} = v_{y,i,t-1} + \cos(\phi_{i,t-1}) \cdot \dfrac{F_{i,t}}{m_i} \cdot \mathrm{d}t \\ \phi_{i,t} = \phi_{i,t-1} + \omega_{\phi,i} \cdot \mathrm{d}t \end{cases} \tag{2}$$

Where v represents the speed of the drones, i is the number of drones, and Δt denotes the time interval. The position of the target in the scenario is random, and drones are unable to acquire the target's position information in advance. Drones are capable of observing the target. When the target is within the monitoring range of a drone, the drone can sense and monitor it. Additionally, drones can receive communication information from other adjacent drones; yet they are unable to determine the quantity and position information of other targets.

A single drone is capable of monitoring multiple target points simultaneously. Since there is no clear target allocation, the drone should always keep the targets within its sensing range and search for as many target points as possible. Moreover, the drone should avoid collisions and flying out of boundaries to meet the safety constraints.

2.2 POMDP-based Cooperative Search Framework

The decision-making process of the drone swarm is defined using a partially observable Markov decision process. Each drone can only obtain local observation information and cannot access the global state. At each time step, each drone makes its action decisions based on its local observation information, and all drones execute their joint actions to update the environment.

Suppose the drone swarm comprises n drones. The partially observable Markov decision process is defined as $(N, S, A, P, R, O, Z, \gamma)$. Here, N is the set of n drones, with $i \in N$. The state space S is composed of the state s of each drone, where $s \in S$. The joint action space of all drones is denoted by $A : \{A^{(1)}, A^{(2)}, \ldots, A^{(n)}\}$, and the joint action $a^{(i)}$ of the i-th drone satisfies $a^{(i)} \in A^{(i)}$. The state-transition function of the environment, $p(s'|s, a) \to [0, 1]$, gives the probability of transitioning from state s to the subsequent state s' upon execution of the joint action $a : \{a^{(1)}, a^{(2)}, \ldots, a^{(n)}\}$. The joint reward $R(s, a) : \{r^{(1)}, r^{(2)}, \ldots, r^{(n)}\}$ is obtained after performing the joint action in state s. The joint observation space of all drones is $O : \{O^{(1)}, O^{(2)}, \ldots, O^{(n)}\}$. The function $Z : o \to Z(s, i)$ represents the individual observation model of each drone in state s, where $o^{(i)} \in O^{(i)}$. The discount factor $\gamma \in [0, 1]$ is used to balance the relative importance between long-term and current rewards.

Each drone makes action decisions based on its local information. The action policy of the i-th drone is $\pi^{(i)} : O^{(i)} \to A^{(i)}$. The joint action policies of all drones are denoted

as $\pi : \{\pi^{(1)}, \pi^{(2)}, \ldots, \pi^{(n)}\}$. When the joint observation of drones is o and the joint action policy is π, the state-value function is expressed as:

$$V_{\pi}^{(i)}(s) = \mathbb{E}_{\pi}\left[\sum_{l=0}^{\infty} \gamma^{l} r^{(i)} | o_0 = o\right] \tag{3}$$

When the joint action is executed, the action-value function is expressed as:

$$Q_{\pi}^{(i)}(o, a) = \mathbb{E}_{\pi}\left[\sum_{l=0}^{\infty} \gamma^{l} r^{(i)} | (o, a)_t = (o, a)\right] \tag{4}$$

Where $\mathbb{E}_{\pi}$ represents the expectation of the action policy.

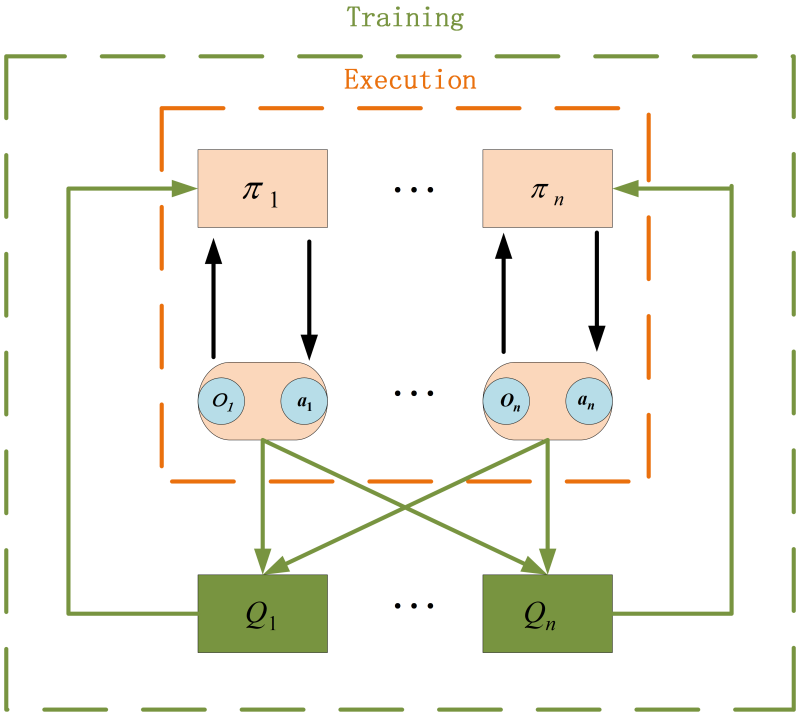


Fig. 1 Centralized Training and Distributed Execution Structure Diagram

In the independent learning process of agents, the environment they are in often exhibits no-stationarity. Agents conduct evaluations and make decisions based on local observations. However, such local observations may lead to inaccuracies in the evaluation mechanism (Critic network) and low efficiency of the decision-making mechanism (Actor network). Nevertheless, during the training simulation phase, agents can obtain more information and are not affected by communication barriers, which offers the possibility of alleviating the non-stationarity of the environment. Therefore, we adopt a framework of Centralized Training with Decentralized Execution (CTDE). Currently,

this type of framework has become one of the most popular training paradigms in the field of multi-agent reinforcement learning.

The Multi-Agent Deep-Deterministic Policy Gradient (MADDPG) algorithm is an innovation first proposed and applied to the DDPG algorithm within this framework. In this algorithm, for each agent (taking drone i as an example), during the centralized training period, the critic network will receive the state-action pairs of all drones for global optimization. In the decentralized execution phase, the actor network makes action decisions based only on the local observation states of the drones themselves. This approach effectively combines the advantages of global information and the flexibility of local execution, providing an effective solution for the coordination and autonomy of multi-agent systems, as shown in Fig 1.

3 Multi-UAV Cooperative Path Planning: A MARL Approach

3.1 MADDPG Algorithm

The Multi-Agent Deep-Deterministic Policy Gradient (MADDPG) algorithm adopts the Centralized Training with Decentralized Execution (CTDE) structure. Under the CTDE structure, agents are allowed to use additional information (global information) for learning during the training phase and make decisions using local information during the execution phase. We use $\theta = [\theta_1, \theta_2, \ldots, \theta_n]$ to represent the parameters of the policies of n agents, and $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$ to represent the policies of n agents.

The cumulative expected reward of the i-th agent is expressed as:

$$J(\theta_i) = \mathbb{E}_{s_t \sim p^\pi, a_t \sim \pi_{\theta_i}} \left[\sum_{t=0}^{\infty} \gamma^t r_{i,t} \right] \tag{5}$$

The gradient of the stochastic policy is expressed as:

$$\nabla_{\theta_i} J(\theta_i) = \mathbb{E}_{s_t \sim p^\pi, a_t \sim \pi_{\theta_i}} \left[\nabla_{\theta_i} \log \pi_i(a_i|o_i) Q^\pi(x_i, a_1, \ldots, a_n) \right] \tag{6}$$

Where o_i represents the observation state of the i-th agent, $x = [o_1, \ldots, o_n]$ represents the observation vector, and $Q^\pi(x, a_1, \ldots, a_n)$ is the centralized state-action function of the i-th agent. Since agents can independently learn their own Q^i functions, each agent can have a different reward function, enabling them to complete cooperative or competitive tasks. The following is the extension to the deterministic policy μ^i and the gradient formula expressed as:

$$\nabla_{\theta_i} J\left(\theta_i\right) = \mathbb{E}_{x,a \sim D} \left[\nabla_{\theta_i} \mu_i(a_i|o_i) \nabla_{a_i} Q_i^\mu(x, a_1, \ldots, a_n) \big|_{a_i = \mu_i(o_i)} \right] \tag{7}$$

The update method of the centralized critic network draws on the temporal difference method in the DQN algorithm. The loss function of the MADDPG algorithm is:

$$L\left(\theta_i\right) = \mathbb{E}_{x,a,r,x'} \left[\left(Q_i^\mu(x, a_1, \ldots, a_n) - y \right)^2 \right] \tag{8}$$

$$y = r_i + \gamma \overline{Q}_i^{\mu'}(x', a'_1, \ldots, a'_n)\big|_{a_j = \mu'_j(o_j)} \tag{9}$$

Where $\overline{Q}_i^{\mu'}$ represents the target network, and $\mu' = [\mu'_1, \ldots, \mu'_n]$ represents the updated target policy. From the above, we can see that the critic network learns using global information, while the actor network only uses local observation information.

For the i-th UAV, the gradient of the expected return, that is, the gradient of the Actor network $J(\mu_i)$ is shown in the following formula:

$$\nabla_{\theta_i} J(\mu_i) = \mathbb{E}_{(o,a_{-i})\sim D}\left[\nabla_{\theta_i}\pi_i(o_i;\theta_i)\nabla_{a_i} Q_i^{\mu}(o, a_i, a_{-i};\mu_i)\big|_{a_i=\mu_i(o_i)}\right] \tag{10}$$

Where J represents the expected value of rewards, E represents the expectation, r_i represents the reward of the i-th UAV, $a_i = [a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n]$ represents the joint action vector of all UAVs except UAV i, D is the experience replay buffer that records information of all UAVs, $Q^{\mu}(o, a, a_{-i};\mu)$ is a centralized state-action value function, whose inputs include the joint observation state of the UAV swarm and the joint actions a of all agents, and μ_i represents the deterministic policy of the i-th UAV.

The output of the Critic network is the centralized state-action value function Q^{μ}, and the loss function is defined as follows:

$$L(\theta_i) = \mathbb{E}_{o,a,r_i,o'}\left[\left(Q_i^{\mu}(o,a) - y\right)^2\right] \tag{11}$$

$$y = r_i + \gamma \bar{Q}_i^{\bar{\mu}}(o', a';\bar{\mu})\big|_{a'=\bar{\mu}(o')} \tag{12}$$

Where $\overline{Q}_i^{\mu}$ is the value function output by the critic-target network, y is the target value function, o' is the observation state of the UAV at the next time step, a' is the action of the UAV at the next time step, μ is the parameter of the actor-target network, and $\overline{\mu}$ is softly updated regularly using the latest parameters μ of the critic network.

The input of the UAV consists of an action vector and an observation state vector. The action vector is composed of $a_i = [F_i, r_i, j]$, and the observation state vector is composed of $o_i = [s_{i,t}, s_{r,i,t}, e_{j,i}]$. The observation state vector contains three types of state information, which are:

1. Current UAV Onboard State Information

The observation state vector of the $\mathbf{o}_{i,t} = [s_{i,t}, s^r_{i,t}, e_{j,t}]$ contains relevant information.

The observation state vector of the i-th UAV at time t is expressed as $\mathbf{o}_{i,t} = [s_{i,t}, s^r_{i,t}, e_{j,t}]$, where $s_{i,t}$ represents the state and action information of the UAV itself, $\mathbf{s}^r_{i,t}$ represents the postural feature information of other UAVs in the same team, and $e_{j,t}$ represents the coordinate information of the target area, where $j \in m$ and m is the set of target areas. For the i-th UAV itself, $\mathbf{s}_{i,t} = [x_{i,t}, y_{i,t}, v_{x_{i,t}}, v_{y_{i,t}}, \varphi_{i,t}]$ represents the state information of the UAV itself.

2 Postural feature vector information of other UAVs:

The UAV observes the state feature vector $\mathbf{s^r_{i,t}}$ Thus, $\mathbf{s^r_{i,t}}$ contains $n-1$ sub-vectors, which is defined as follows:

$$\mathbf{s^-_{i,t}} = [\hat{s}_{1,t}, \hat{s}_{2,t}, \cdots, \hat{s}_{i-1,t}, \hat{s}_{i,t}, \cdots, \hat{s}_{N,t}] \tag{13}$$

Where $\hat{s}_{k,t}$ is the observation state sub-vector of the k-th UAV, as shown below:

$$\hat{\mathbf{s}}_{\mathbf{k},\mathbf{t}} = [d_{i,k,t}, x_{k,t}, y_{k,t}, v_{x,k,t}, v_{y,k,t}, a_{k,-1}] \tag{14}$$

Where $a_{k,-1}$ represents the action of the k-th UAV in the previous time step, $k \in N, i \neq k$. The distance $d_{i,k,t}$ represents the relative distance between the i-th UAV and the k-th UAV teammate, which is described by the 2-norm as follows:

$$d_{i,k,t} = \|p_{i,t} - p_{k,t}\|_2 = \sqrt{(p_{i,t} - p_{k,t})^H (p_{i,t} - p_{k,t})} \tag{15}$$

Where H represents the conjugate transpose matrix, $p_{i,t}$ represents the two-dimensional coordinate vector of the i-th UAV at time t, and $p_{k,t}$ represents the two-dimensional coordinate vector of the k-th UAV at time t.

3 Coordinate vector information of the target area:

The UAV observes the state-feature information of the target area and can obtain the observation vector $\mathbf{e}_{\mathbf{j},\mathbf{t}}$ by observing the target area. It includes m sub-vectors, defined as follows:

$$\mathbf{e}_{j,t} = [s^*_{1,t}, s^*_{2,t}, \ldots, s^*_{j,t}, \ldots, s^*_{m,t}] \tag{16}$$

Where $s^*_{j,t}$ represents the sub-vector for observing the j-th target, specifically as follows:

$$s^*_{j,t} = [d_{i,j,t}, x_{j,t}, y_{j,t}] \tag{17}$$

Where $d_{i,j,t}$ represents the distance between the UAV and the target, and $x_{j,t}, y_{j,t}$ represent the position of the target.

3.2 Pointwise Mutual Information Method

This paper employs Positive Pointwise Mutual Information (PPMI) to capture the dependency relationships among unmanned aerial vehicles (UAVs). Then, the dependency index $d^{(ik)}$ between the i-th UAV and the k-th UAV is expressed as:

$$d^{(ik)} = PPMI(l_i, a_i, l_k, a_k) = \max\left(\log \frac{p(l_i, a_i, l_k, a_k)}{p(l_i, a_i)P(l_k, a_k)}, 0\right) \tag{18}$$

Where l represents the local information of each UAV, where $l_i = (s_{i,t}, s^r_{i,t}, e_{j,t}, b_{i,t})$ and b_i denotes the boundary information of the i-th UAV. Also, l_k is given by $l_k = (s_{k,t}, s^r_{k,t}, e_{j,t}, b_{k,t})$ with b_k being the boundary information of the k-th UAV, and $i \neq k$. The function $p()$ represents the joint distribution function.

The Positive Pointwise Mutual Information $PPMI(l_i, a_i, l_k, a_k)$ is estimated by maximizing the Jensen-Shannon (JS) divergence of the mutual information $I_{JS}(L^{(i)}, A^{(i)}, L^{(k)}, A^{(k)})$ using the neural network $f_w(l^{(i)}, a^{(i)}, l^{(k)}, a^{(k)})$. $L^{(i)}$ represents all the joint observations of UAV i, and $A^{(k)}$ represents all the joint actions of UAV k.

Suppose the joint distribution of two random variables (X, Y) is $P(X, Y)$ and their marginal distributions are $P(X)$ and $P(Y)$ respectively. The mutual information (MI)

of (X, Y) is expressed as:

$$I(X;Y) = D_{KL}(P(X,Y), P(X)P(Y)) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{19}$$

Where D_{KL} represents the divergence. The greater the correlation between X and Y, the larger the value of $I(X;Y)$. If X and Y are independent, then $I(X;Y) = 0$. For a specific pair of random events (x, y), Pointwise Mutual Information (PMI) is used to measure the degree of their dependence:

$$PMI(x;y) = \log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(x|y)}{p(x)} = \log \frac{p(y|x)}{p(y)} \tag{20}$$

Both mutual information and pointwise mutual information are symmetric. $I(X;Y)$ is the weighted sum of the pointwise mutual information for all possible pairs (x, y). To avoid the situation where $\log 0 = -\infty$, Positive Pointwise Mutual Information (PPMI) is defined as:

$$PPMI = \max\left(\log \frac{p(x,y)}{p(x)p(y)}, 0\right) \tag{21}$$

Each UAV has a predefined environmental reward. A major challenge in calculating reciprocal rewards lies in determining the dependency index between two cooperating UAVs. In a UAV swarm, each UAV has partial observability and makes action decisions based on its local information, such as communication with neighbors, target observation, boundary information, and state-related data. Since all UAVs operate within the same environment and the global state remains unknown, the local information and actions of two adjacent UAVs may not be independent of each other. We employ Pointwise Mutual Information (PMI) to capture the interdependencies among UAVs. The dependency index $d^{(ik)}$ between the i-th UAV and the k-th UAV is expressed as:

$$d^{(ik)} = PMI(l_i, a_i, l_k, a_k) = \log \frac{p(l_i, a_i, l_k, a_k)}{p(l_i, a_i)p(l_k, a_k)} \tag{22}$$

Where l denotes the local information of each unmanned aerial vehicle (UAV). Specifically, $l_i = (s_{i,t}, s^r_{i,t}, e_{j,t}, b_{i,t})$, in which $b_{i,t}$ represents the boundary-related information of the i-th UAV. Moreover, $l_k = (s_{k,t}, s^r_{k,t}, e_{j,t}, b_{k,t})$, where $b_{k,t}$ refers to the boundary-related information of the k-th UAV, with $i \neq k$. The symbol $p()$ signifies the joint distribution function.

Due to the difficulty of directly deriving Pointwise Mutual Information (PMI) in complex environments, we can obtain an estimation of the optimal PMI by maximizing Mutual Information (MI). MI can be expressed as the expected value of the divergence between the joint probability distribution and the product of marginal probability distributions. Based on different measurement methods, numerous works have been conducted on the variational lower bounds for maximizing MI. Among them, I_{JS} estimates MI using the Jensen-Shannon (JS) divergence, inspired by the neural estimation

of MI; this paper employs a neural network to estimate PMI. For the sake of concise and convenient derivation, we use X_1 and X_2 to substitute for $l^{(i)}$, $a^{(i)}$, and $l^{(k)}$, $a^{(k)}$, and then estimate PMI with the following lemma.

For random variables X_1 and X_2, their JS-based mutual information is defined as:

$$\begin{aligned} I_{JS}(X_1, X_2) &= D_{JS}\left(P(X_1, X_2) \| P(X_1)P(X_2)\right) \\ &= \frac{1}{2} D_{KL}\left(P(X_1, X_2) \| M\right) + \frac{1}{2} D_{KL}\left(P(X_1)P(X_2) \| M\right) \end{aligned} \tag{23}$$

Where $M = \frac{P(X_1,X_2)+P(X_1)P(X_2)}{2}$, The variational lower-bound of $I_{JS}(X_1, X_2)$ is given by

$$\begin{aligned} I_{JS}(X_1, X_2) \geq \sup_{f_w} & E_{P(X_1,X_2)}\left[-sp(-f_w(x_1, x_2))\right] \\ & - E_{P(X_1)\times P(X_2)}[sp(f_w(x_1, x_2))] \end{aligned} \tag{24}$$

Where $f_w(x_1, x_2)$ is a fitting function parameterized by w, and $sp(x) = \log(1 + e^x)$. When the variational lower-bound reaches its maximum, we can find an optimal fitting function:

$$f_w^*(x_1, x_2) = \log \frac{p(x_1, x_2)}{p(x_1)p(x_2)} = PMI(x_1; x_2) \tag{25}$$

In fact, when there is no dependency relationship between two random events, the minimum pointwise mutual information is 0. However, the output of f_w may be less than 0. Therefore, this paper proposes using Positive Pointwise Mutual Information (PPMI) to capture the dependency relationships among unmanned aerial vehicles (UAVs), and further to reshape the reward function.

3.3 Reward Function

To encourage UAVs to learn cooperative strategies, each UAV must not only reach the perceived target efficiently but also avoid redundant sensing to maximize the number of detected targets. Additionally, they should avoid flying out of the mission area to ensure flight safety. These expectations are represented by potential reward shaping. Therefore, it is necessary to design appropriate reward functions to guide the learning process. The reward function is designed to encourage UAVs to reach their targets while minimizing the traveling distance between UAVs. The set of distances from each UAV to its target is computed, and the UAV reward $r_1^{(i)}$ is given as follows:

$$r_1^{(i)} = -\min(d) \tag{26}$$

When two observation areas overlap, it is referred to as a pseudo-collision. Similarly, when UAVs physically collide with each other, it is called a real collision. In training, the frequency of real collisions is reduced by penalizing the occurrence of pseudo-collisions. The collision reward $r_2^{(i)}$ is as follows:

$$r_2^{(i)} = \begin{cases} -(\sigma - \text{dist}_2), & \text{if } \text{dist}_2 < \sigma \\ 0, & \text{else} \end{cases} \tag{27}$$

Among them, dist_2 represents the distance between two unmanned aerial vehicles (UAVs), and σ denotes the observation radius of a UAV.

Each UAV operates within the mission area. When a UAV is too close to the boundary, part of its perception range may extend beyond the mission area, and the perception of the non-mission area may be useless. The minimum distance from a UAV to the boundary is defined as $d_{\min}$, and the boundary reward $r_3^{(i)}$ is given as follows:

$$r_3^{(i)} = \begin{cases} -\exp\left(\frac{\sigma - d_{\min}}{\sigma}\right), & d_{\min} < \sigma \\ 0, & \text{else} \end{cases} \tag{28}$$

Repeated tracking means that a single target is being tracked simultaneously by multiple UAVs, which results in resource wastage and an increased collision risk. When the relative distance between two UAVs is greater than twice the perception radius, repeated tracking does not occur.Therefore, The repeated-tracking reward $r_4^{(i)}$ is defined as follows:

$$r_4^{(i)} = \begin{cases} -\exp\left(\frac{2\sigma - \text{dist}_2}{2\sigma}\right), & \text{if } \text{dist}_2 < 2\sigma \\ 0, & \text{else} \end{cases} \tag{29}$$

Consequently, the per-step environmental reward $r_{\text{env}}^{(i)}$ for a UAV is:

$$r_{\text{env}}^{(i)} = r_1^{(i)} + r_2^{(i)} + r_3^{(i)} + r_4^{(i)} \tag{30}$$

In the multi-agent path-planning problem, it is assumed that if a UAV can cooperate with its adjacent UAVs to obtain better rewards, it should be appropriately encouraged. Otherwise, when a neighboring UAV receives a negative reward, the UAV should bear part of the responsibility and be penalized accordingly. Therefore, the immediate reward $r^{(i)}$ for each UAV is modified as follows:

$$r^{(i)} = (1 - \alpha) r^{(i)} + \alpha r_c^{(i)} \tag{31}$$

3.4 Reward Function

The PMI-MADDPG framework, as depicted in Fig. 2, extends the CTDE paradigm by integrating PMI Actors and Reciprocal Critics across UAV units to address swarm

coordination. Each UAV unit operates as follows: PMI Actor: Utilizing local observations o_i and pointwise mutual information (PMI), the actor computes actions a_i that maximize both individual rewards and swarm-wide correlations. The PMI term evaluates interdependencies between UAVs' states (s_i, s_j), enabling adaptive cooperation. Reciprocal Critic: This module estimates the reciprocal reward r_i^c by aggregating PMI-weighted rewards from neighboring UAVs. It employs a temporal-difference error (TD-error) to guide policy updates, ensuring UAVs prioritize actions beneficial to the swarm's global state.

During training, the CTDE mechanism synchronizes decentralized PMI Actors and centralized Reciprocal Critics. The critic network receives global observations $(s_1, s_2, \ldots, s_n)$ and joint actions $(a_1, a_2, \ldots, a_n)$ to compute the Q-value, while actors execute locally optimized policies. The reward signal r_{env}^i combines individual task rewards r_{env}^i and reciprocal incentives r_i^c, dynamically balanced via gradient-based coefficients. Experience Pool (EP) stores historical transitions (s, a, r, s') from all UAVs, enabling efficient experience replay during training. By sampling mini-batches from EP, the framework mitigates non-stationarity and enhances learning stability. This architecture ensures stable learning through target networks and experience replay, addressing non-stationarity in swarm environments. Arrows in Fig. 2 illustrate the flow of observations, actions, and reward gradients, enabling UAVs to achieve decentralized autonomy with centralized coordination awareness.

4 Analysis of Simulation Results

4.1 Simulation Environment and Parameter Setting

In the path planning of UAV swarms for multiple target points, it is not only necessary for individual UAVs to find the target points, but also for the UAV swarm to cooperate, enabling more UAVs to locate the target points. In the UAV swarm environment, UAVs have partial observability and cannot obtain all the information. The cooperative path planning of the UAV swarm constructed in this paper divides the scenario into a rectangular environment of 2000m $\times$ 2000m. The detection range of the target is 10 meters, and the positions of the UAV swarm and the target points appear randomly.

In the simulation experiments, the maximum number of training episodes is set to 1500, with each episode limited to 1000 timesteps. The network parameters are updated every 10 timesteps, and the final training results are analyzed. The specific definitions of the environment, learning process, and neural network model parameters are provided in Table 1. The neural network employs orthogonal initialization to accelerate learning, while the Adam optimizer is selected for its consideration of both momentum and adaptive parameter adjustments.

4.2 Model Performance And Result Analysis

The performance analysis initiates with a rigorous evaluation of training outcomes across three state-of-the-art algorithms: MADDPG, PMI-MADDPG, and MAPPO. The selection of MAPPO as a comparative algorithm is due to its high optimization efficiency and strong adaptability, which makes it a high-performance benchmark

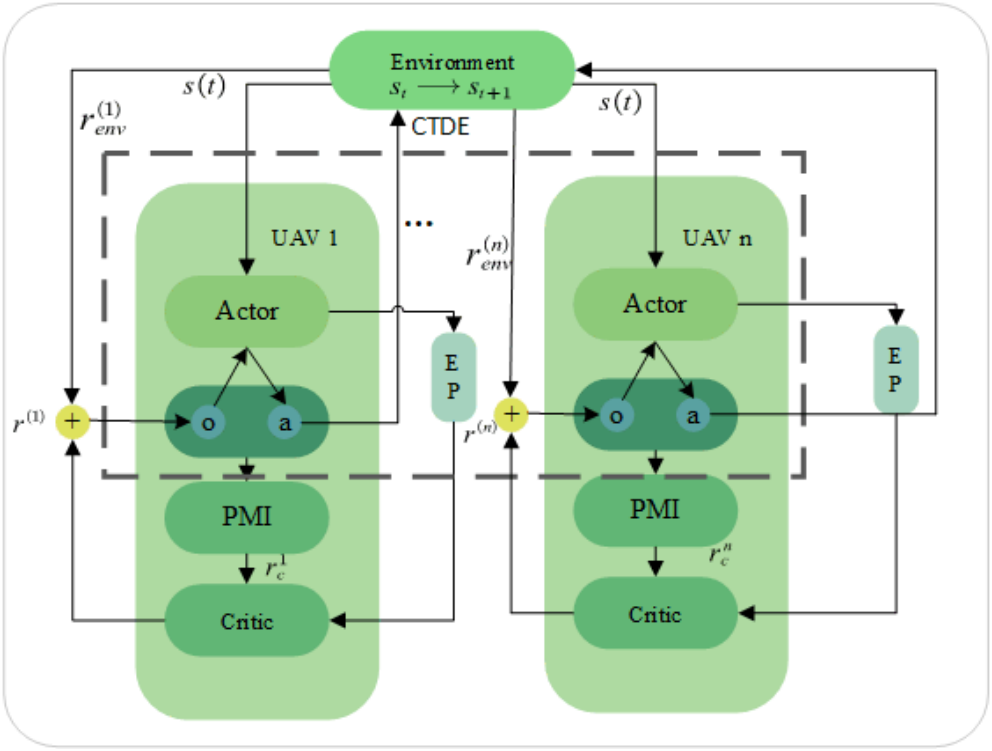


Fig. 2 PMI-MADDPG algorithmic framework

algorithm for addressing multi-agent cooperation issues. To ensure fairness, all models share identical network configurations (layers and neurons), while algorithm-specific hyperparameters (e.g., learning rates, discount factors) are independently optimized. Each experiment is rigorously repeated three times with random seeds 1, 2, and 3, and final metrics are averaged to ensure statistical robustness.

This study employs the Multi-Agent Particle Environment (MPE) and enhances the UAV dynamics model based on the MPE framework while redesigning the reward function. To evaluate the performance of multi-agent reinforcement learning algorithms in UAV swarm path planning, the episode reward, Critic loss, and Actor loss are used as evaluation metrics.

The reward values for UAV swarm path planning based on the MADDPG (3 UAVs) method and the PMI-MADDPG (3 UAVs) method are presented in Figure 2. During the initial training phase, the reward values of the UAV swarm are similar because the Critic network has not yet gathered sufficient experimental data to train its own network. As the experiment progresses, it becomes evident that, at episode 600, the UAV swarm based on the PMI-MADDPG (3 UAVs) method achieves higher rewards. This indicates the effectiveness of the reward function reshaping using the positive point mutual information method. Furthermore, both methods converge in the later stages of training, suggesting that UAV swarms utilizing different approaches

Table 1 The simulation environment and network parameter settings.

Parameter	Value
Number of UAVs	2&3
Number of episodes	1500
Maximum timestep per episode	1000
Actor learning rate	1.5×10^{-4}
Critic learning rate	3×10^{-4}
Discount factor	0.99
Batch size	256
Experience replay buffer size	1×10^{7}
Network update frequency	10
Optimizer	Adam
Number of network layers	4
Activation function	ReLU
Number of neurons in hidden layers	128

learn stable path planning strategies. The MAPPO algorithm is too conservative in its exploration strategy, making it difficult to handle complex environments and resulting in convergence to a suboptimal solution.

In Figure 2, it is also observed that the PMI-MADDPG (2 UAVs) experiment converges faster than the PMI-MADDPG (3 UAVs) experiment. The increase in the number of UAVs leads to an exponential growth in the action space, causing a slower learning speed for larger UAV swarms. Additionally, their reward values are similar, indicating that smaller UAV swarms are able to learn better action decisions. In order to further validate the experimental results, three algorithms, MADDPG, PMI-MADDPG, MAPPO, were tested. The model with the random seed of 3 was chosen for model testing, and each algorithm was tested 10 rounds so as to take the average value as the test result; other environmental parameters were consistent with the model training above. The model test results are shown in Table 2.

Table 2 UAV Swarm Reward Analysis Values

Algorithm	Maximum Reward	Minimum reward	Average Reward
PMI-MADDPG (3 UAVs)	-205.15	-311.67	-243.46
MADDPG (3 UAVs)	-241.88	-309.43	-261.69
MAPPO (3 UAVs)	-284.08	-320.13	-293.27

As shown in Table 2, during the early stages of training, the reward values obtained using the PMI-MADDPG (3 UAVs) method are slightly lower than those achieved with the MADDPG (3 UAVs) method, indicating that the strategies of both methods are similar, with some overlap in action selection. However, as training progresses, the PMI-MADDPG (3 UAVs) method enhances the UAV swarm's collaboration capability through the reshaping of the reward function via positive point mutual information.

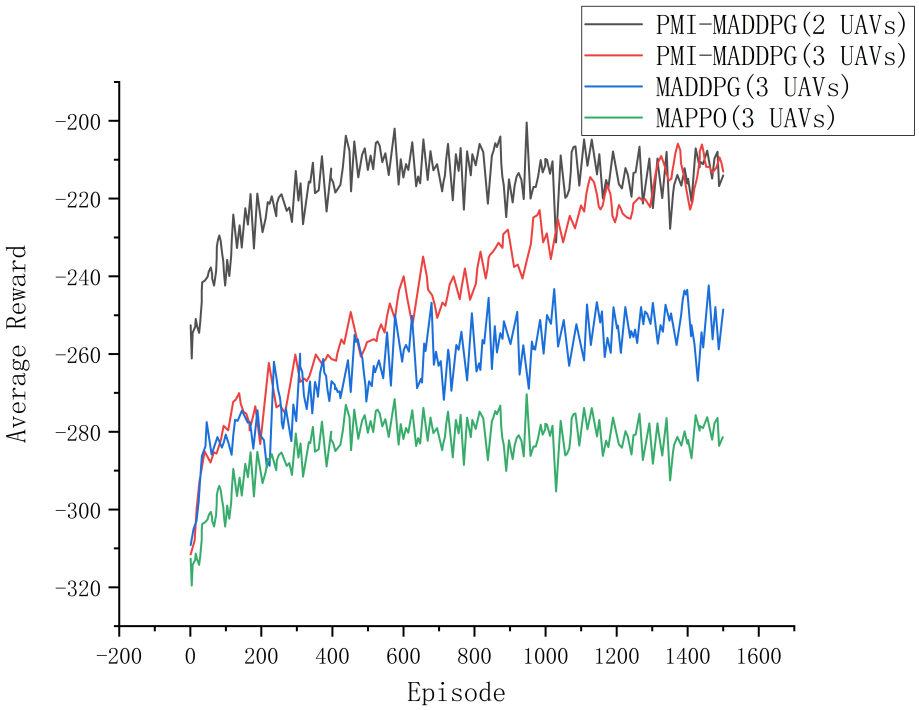


Fig. 3 Average Reward of the UAV Swarm

From the perspective of the maximum reward achieved, the PMI-MADDPG (3 UAVs) method significantly outperforms the MADDPG (3 UAVs) method[1].

In this study, the average reward value is used to measure the advantage of the PMI-MADDPG (3 UAVs) method. As shown above, in the path planning task, the average reward value for the PMI-MADDPG (3 UAVs) method is -243.46, while for the MADDPG (3 UAVs) method, it is -261.69. This indicates that the PMI-MADDPG (3 UAVs) method achieves approximately a 7.5% improvement over the MADDPG (3 UAVs) method in terms of average reward.

As shown in Figure 2, the Critic loss function value gradually converges as training progresses, indicating that the Critic network's estimation of state-action pairs under the current policy becomes increasingly stable and closer to the true values. The value function provided by the Critic network is critical for guiding the policy updates of the Actor network. The gradual convergence and stabilization of the Critic loss function value suggest that the feedback it provides is sufficiently accurate to help the Actor network effectively update the policy gradient, which is crucial for optimizing the policy in the overall learning environment. Additionally, this signifies the stability of

[1]Detailed analysis of reward values shows that the PMI-MADDPG (3 UAVs) achieves better overall performance.

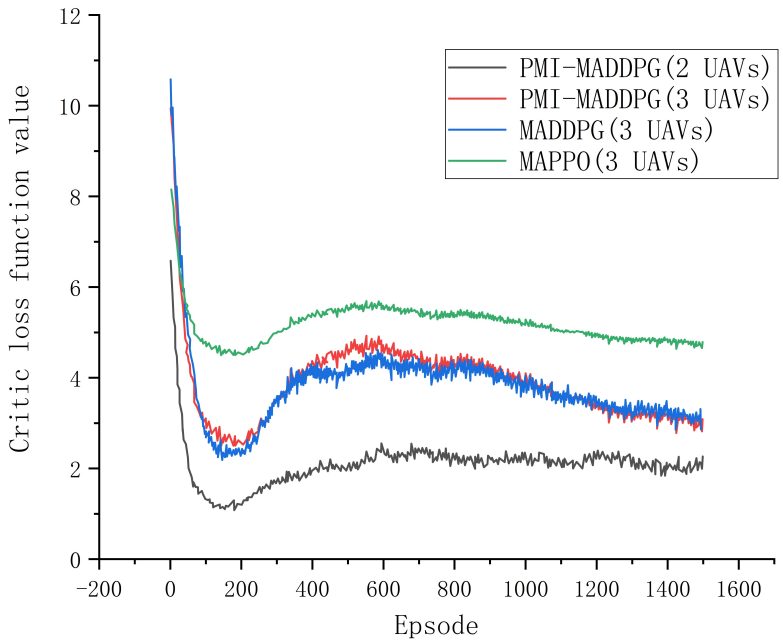


Fig. 4 Critic Loss Function Value

the learning process. In Figure 2, it can be observed that the Critic loss function value converges more stably with 2 UAVs compared to 3 UAVs, indicating that the number of agents significantly impacts multi-agent reinforcement learning algorithms. In the same environment, an increase in the number of agents leads to greater instability in the experiment.

As shown in Figure 5, the Actor loss function value gradually converges as training progresses. The Actor network is responsible for outputting the optimal action for the cur-rent state in the path planning model. Its loss function is typically designed to maximize the expected return of the policy, as estimated by the Critic network. When the Actor net-work's loss function converges, it indicates that the policy learned by the network effectively maximizes the expected return within the current learning environment and frame-work. Additionally, during policy learning, it is important to balance the exploration of unknown actions with the exploitation of known effective actions. For UAV swarms with different numbers of agents, the convergence of the Actor network's loss function suggests that, under the current exploration and exploitation strategy, the UAVs have found a relatively optimal action execution strategy. However, compared to the Actor network loss function with 3 UAVs, the loss

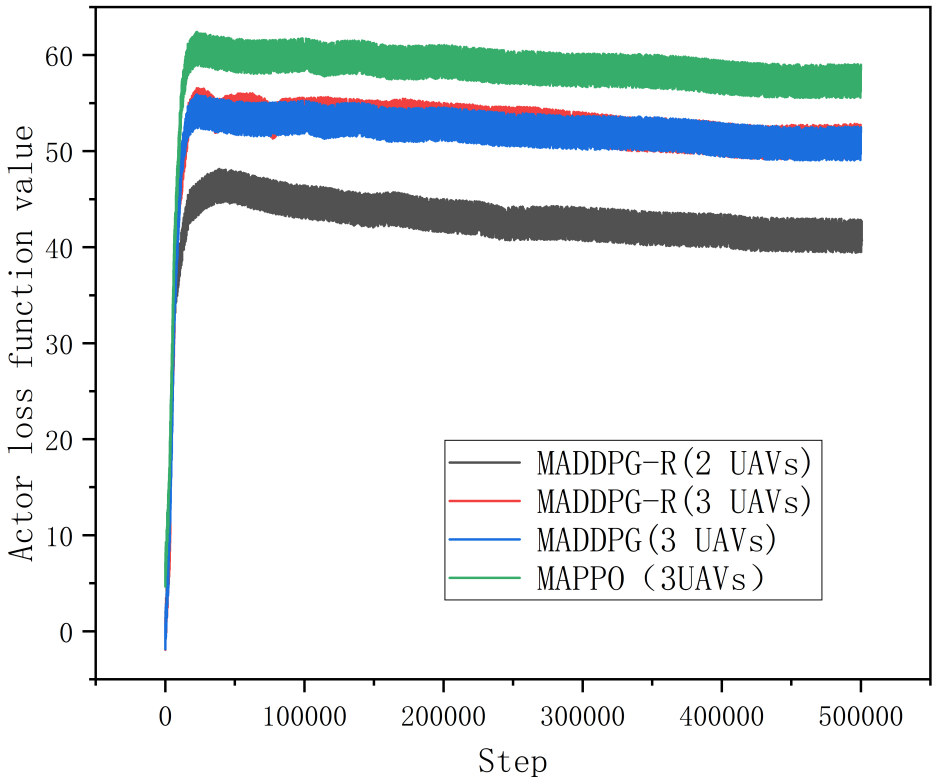


Fig. 5 Actor Loss Function Value

function value for the 2 UAVs swarm is slightly lower, indicating that the smaller UAV swarm has a smaller action space and can more easily find the optimal action decision.

5 Conclusion

This study presents PMI-MADDPG, an enhanced multi-agent reinforcement learning framework addressing cooperative path planning for UAV swarms in dynamic partially observable environments. The algorithm introduces two key innovations: a neural net-work-based pointwise mutual information estimator for quantifying inter-drone dependencies, and a dynamic reward-balancing mechanism that adaptively weights individual versus swarm-level objectives. Experimental validations demonstrate significant improvements over conventional MADDPG, with the proposed method achieving higher task rewards and faster convergence rates across varying swarm sizes. The centralized training-decentralized execution architecture ensures stable learning dynamics, as evidenced by the Critic/Actor networks' smooth convergence patterns despite increasing environmental complexity. Future work will focus on applying the proposed PMI-MADDPG algorithm to scenarios with a larger scale of UAVs.

At the same time, while optimizing the performance of the algorithm and the training efficiency of the model.

Acknowledgments. This work is supported by the Open Fund of State key Laboratory of Extreme Environment Optoelectronic Dynamic Testing Technology and Instrument, North University of China (2023-SYSJJ-04), the Aeronautical Science Foundation of China (20240055054001), the Open Fund of State Key Laboratory of Comprehensive Space time Network and Equipment Technology (CEPNT2023A01), the Open Fund of Key Laboratory of Spatio-temporal Sensing and Intelligent Processing, Ministry of Natural Resources of the People's Republic of China (232203), the Key Laboratory of Technology and Equipment of Tianjin Urban Air Transportation System (TJKL-UAM-202305), the Open Fund of Key Laboratory of Civil Aviation Flights Wide Area Surveillance and Safety Control technology of Civil Aviation University of China (202105).

References

[1] Murata, H., Masui, S., Tsuchida, Y.: Efficacy evaluation of low-volume concentrate application of pesticides by unmanned aerial vehicle (uav) using an indoor spraying device. Applied Entomology and Zoology **59**(2), 103–110 (2024)

[2] Srivastava, A., Prakash, J.: Techniques, answers, and real-world uav implementations for precision farming. Wireless Personal Communications **131**(4), 2715–2746 (2023)

[3] Zhu, M., Yu, X., Tan, H., Yuan, J.: Integrated high-precision monitoring method for surface subsidence in mining areas using d-insar, sbas, and uav technologies. Scientific Reports **14**(1), 12445 (2024)

[4] Manoj, S., Valliyammai, C.: Drone network for early warning of forest fire and dynamic fire quenching plan generation. EURASIP Journal on Wireless Communications and Networking **2023**(1), 112 (2023)

[5] Xin, H., Dang, F., Wang, J., Zhang, K., Wang, Y., Pang, X.: Analysis of uav swarm combat technology development trend. In: International Conference on Advanced Unmanned Aerial Systems, pp. 80–86 (2024). Springer

[6] Su, W., Gao, M., Gao, X., Xuan, Z.: A decision-making method for distributed unmanned aerial vehicle swarm considering attack constraints in the cooperative strike phase. International Journal of Aerospace Engineering **2023**(1), 6568359 (2023)

[7] Dijkstra, E.W.: A note on two problems in connexion with graphs. In: Edsger Wybe Dijkstra: His Life, Work, and Legacy, pp. 287–290 (2022)

[8] Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and

Cybernetics **4**(2), 100–107 (1968)

[9] Stentz, A.: Optimal and efficient path planning for partially-known environments. In: Proceedings of the 1994 IEEE International Conference on Robotics and Automation, pp. 3310–3317 (1994). IEEE

[10] Ershen, W., Jing, G., Chen, H., Hongfan, R., Aidong, C., Xinna, S.: Cooperative confrontation model of uav swarm with random spatial networks. Journal of Beijing University of Aeronautics and Astronautics **49**(1), 10–16 (2021)

[11] Dewangan, R.K., Shukla, A., Godfrey, W.W.: Three dimensional path planning using grey wolf optimizer for uavs. Applied Intelligence **49**, 2201–2217 (2019)

[12] Wang, E., Liu, F., Hong, C., *et al.*: Masac-based confrontation game method of uav clusters. Science China Information Sciences **52**(10), 2254–2269 (2022)

[13] Han, Z., Chen, M., Shao, S., Zhou, T., Wu, Q.: Path planning of unmanned autonomous helicopter based on hybrid satisficing decision-enhanced swarm intelligence algorithm. IEEE Transactions on Cognitive and Developmental Systems **15**(3), 1371–1385 (2022)

[14] Nair, R., Tambe, M., Roth, M., Yokoo, M.: Communications for improving policy computation in distributed pomdps. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004., pp. 1098–1105 (2004). IEEE

[15] Yu, Z., Si, Z., Li, X., Wang, D., Song, H.: A novel hybrid particle swarm optimization algorithm for path planning of uavs. IEEE Internet of Things Journal **9**(22), 22547–22558 (2022)

[16] Su, K., Qian, F.: Multi-uav cooperative searching and tracking for moving targets based on multi-agent reinforcement learning. Applied Sciences **13**(21), 11905 (2023)

[17] Xuan, Y., Zhang, X.-Y., Li, C., Wang, H., Mu, C.: Lam-yolov11 for uav transmission line inspection: overcoming environmental challenges with enhanced detection efficiency. Multimedia Systems **31**(2), 1–13 (2025)

[18] Ren, Y., Zhang, H., Du, L., Zhang, Z., Zhang, J., Li, H.: Stealthy black-box attack with dynamic threshold against marl-based traffic signal control system. IEEE Transactions on Industrial Informatics (2024)

[19] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., Freitas, N.: Dueling network architectures for deep reinforcement learning. In: International Conference on Machine Learning, pp. 1995–2003 (2016). PMLR

[20] Yin, S., Xiang, Z.: Adaptive operator selection with dueling deep q-network for

evolutionary multi-objective optimization. Neurocomputing **581**, 127491 (2024)

[21] Wang, X., Gursoy, M.C., Erpek, T., Sagduyu, Y.E.: Learning-based uav path planning for data collection with integrated collision avoidance. IEEE Internet of Things Journal **9**(17), 16663–16676 (2022)

[22] Lan, W., Jin, X., Chang, X., Wang, T., Zhou, H., Tian, W., Zhou, L.: Path planning for underwater gliders in time-varying ocean current using deep reinforcement learning. Ocean Engineering **262**, 112226 (2022)

[23] Oroojlooy, A., Hajinezhad, D.: A review of cooperative multi-agent deep reinforcement learning. Applied Intelligence **53**(11), 13677–13722 (2023)

[24] Canese, L., Cardarilli, G.C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., Spanò, S.: Multi-agent reinforcement learning: A review of challenges and applications. Applied Sciences **11**(11), 4948 (2021)

[25] Benhamza, K., Seridi, H., Agguini, M., Bentagine, A.: A multi-agent reinforcement learning based approach for intelligent traffic signal control. Evolving Systems **15**(6), 2383–2397 (2024)

[26] Li, X., Zhang, X., Li, J., Luo, F., Huang, Y., Zhang, X.: Blocklength allocation and power control in uav-assisted urllc system via multi-agent deep reinforcement learning. International Journal of Computational Intelligence Systems **17**(1), 138 (2024)

[27] Wang, L., Wang, K., Pan, C., Xu, W., Aslam, N., Hanzo, L.: Multi-agent deep reinforcement learning-based trajectory planning for multi-uav assisted mobile edge computing. IEEE Transactions on Cognitive Communications and Networking **7**(1), 73–84 (2020)