Optimizing Multi-UAV Formation Cooperative Control Strategies with MCDDPG Approach

Jinsheng Xiao, *Senior Member, IEEE,* Bolun Yan, Honggang Xie*, Qiuze Yu, Linkun Li, and Yuan-Fang Wang

***Abstract*—In recent years, the application of Unmanned Aerial Vehicle (UAV) devices in military, industrial, and civilian sectors has become increasingly widespread. Consequently, research on multi-UAV formation cooperative control strategies has garnered significant attention. However, current multi-agent reinforcement learning algorithms often struggle with unguided exploration, making it challenging for agents to develop efficient action strategies for complex collaborative tasks. To address this issue, this paper introduces a Multi-Critic Deep Deterministic Policy Gradient (MCDDPG) algorithm. This algorithm designs a Multi-Critic (MC) structure based on the DDPG algorithm. This structure guides UAVs using physical models for tracking and obstacle avoidance, while deep learning models are employed to facilitate cooperative coordination among UAVs. Furthermore, to address the weight allocation issue among different Critic modules in the MC structure, a dynamic difficulty priority weight optimization algorithm is implemented. This enhances the algorithm's collaborative capabilities. To validate the collaborative planning capability of the proposed algorithm, a simulation scenario involving multi-coupled tasks is designed in the Multi-Agent Particle Environment (MPE). In this scenario, the MCDDPG algorithm demonstrates the fastest convergence speed and the optimal collaborative strategy, outperforming other state-of-the-art Multi-Agent Deep Reinforcement Learning (MADRL) algorithms currently in use.**

***Index Terms*—Multi-agent deep reinforcement learning(MADRL), Formation control of unmanned aerial vehicles(UAV), gravitational force model, optimal multi-module weights.**

I. Introduction

With the continuous development of intelligent devices such as drones, unmanned vehicles, and unmanned ships, various intelligent agents can now perform specific tasks in different environments. These agents can significantly reduce labor consumption and improve personal safety due to their strong information acquisition ability.

At present, the development of single UAV control technology is relatively mature, but there is still room for development in the cooperative control technology of multi-agents, such as UAV formation control technology. While this technology can quickly and efficiently complete complex coupling tasks such as battlefield environment exploration, forest fire rescue, and post-disaster personnel search.The current challenge is mainly the limitation of poor coordination among agents and control strategies that struggle to adapt to dynamic and complex environments increase the difficulty of practical application for multi-agent control technology.

Therefore, In the field of multi-agent cooperative planning, researchers focus on enabling intelligent agents to share data and coordinate effectively to achieve specific tasks. Two primary research strategies are currently employed to enhance cooperative capabilities in multi-agent systems. The first way is cooperative algorithms inspired by biological principles are designed to enable agent collectives to adapt their group functionalities across different environments. The second way is a combination of deep learning and reinforcement learning techniques allows agents to learn optimal cooperative strategies based on policy value functions and policy gradients [1].

Currently, Bio-inspired algorithms draw inspiration from biological evolutionary processes, these algorithms have decent performance in the field of coordination control optimization [2]–[4] and multi-agent task allocation [5]. However, the drawback of bio-inspired algorithms difficulty lies in handling the interconnection between agents and dynamic environments. They lack the capability to adapt to changes in the environment.

In order to make the UAV cooperation capability achieve human-level intelligent control, researchers combine deep neural networks with reinforcement learning that effectively mitigate the shortcomings of bio-inspired methods. For instance, Mnih et al. proposed the value-based DQN algorithm [6], and Van Hasselt et al. introduced the DDQN algorithm [7], both of which guide the agent's decision-making process through value function learning, which can be either state value or action-value functions. In contrast to value-based deep reinforcement learning algorithms, Schulman et al. proposed the TRPO algorithm [8] and introduced the PPO algorithm [9], focusing on policy networks as the update target and directly learning the mapping from states to actions to identify the optimal policy. Additionally, Volodymyr Mnih et al. proposed the A2C reinforcement learning architecture [10] and Lillicrap et al proposed DDPG algorithm [11] based on A2C . The A2C architecture allows for direct policy learning while also utilizing value functions to inform policy updates. This hybrid approach combines the strengths of both value and policy functions, thus expanding the application domain of deep reinforcement learning techniques.

Building on deep reinforcement learning, research into

This work was supported in part by the Major Program (JD) of Hubei Province (2023BAA026).The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

J. Xiao, B. Yan, Q. Yu and L. Li are with the School of Electronic Information, Wuhan University, Wuhan 430072, China. e-mail: {xiaojs, yanbl1, yuhenry007, lilinkun}@whu.edu.cn.

H. Xie is with the School of Electrical and Electronic Engineering, Hubei University of Technology, Wuhan 430068, China(e-mail:xiehg@hbut.edu.cn).

Y. Wang is with the Department of Computer Science, University of California, Santa Barbara, CA 93106-5110,USA. (e-mail: yfwang@cs.ucsb.edu).

multi-agent reinforcement learning algorithms has also progressed significantly. Yu, C et al. conducted an in-depth study of the performance of the PPO algorithm in cooperative multi-agent settings, subsequently proposing the MAPPO algorithm [12]. Pu, Y et al. introduced a novel decomposed multi-agent soft actor-critic MASAC method, which supports efficient non-policy learning and partially addresses the credit assignment problem in both discrete and continuous action spaces , making it suitable for tasks with large action spaces [13]. Additionally, Lowe et al. developed the MADDPG algorithm, which takes into account the actions of other agents and successfully learns complex multi-agent coordination tasks. This method demonstrates superior performance compared to other multi-agent reinforcement learning algorithms in cooperative-competitive task scenarios [14].

At present, multi-agent reinforcement learning algorithms have been applied in various multi-agent systems that require the consideration and collaboration of state information among different control units, including traffic signal control [15], resource allocation [16], competitive games [17], and robot collaboration [18]. These applications demonstrate the effectiveness of multi-agent systems in real-world scenarios.

Multi-agent reinforcement learning techniques have been widely applied to various tasks in drone formations, demonstrating their potential in advancing the capabilities of autonomous UAV operations. For instance, reference [19] proposes a cooperative trajectory design method (KMAPPO) to minimize interaction overhead and optimize deployment efficiency for emergency communications in disaster areas. Reference [20] introduces a multi-agent deep deterministic policy gradient method (3A-MADDPG) based on attention mechanisms and adaptive precision, enhancing path planning accuracy during formation adjustments. Reference [21] presents the GPR-MADDPG algorithm, which employs a multi-UAV system for continuous tracking and end-to-end coverage of mobile fleets, providing safety and surveillance support. Reference [22] investigates multi-UAV competitive games in air combat scenarios using a graph attention-based multi-agent soft actor-critic reinforcement learning target prediction network (GA-MASAC-TP). Lastly, references [23], [24] improve the MASAC and MADDPG algorithms, respectively, to enhance cooperative performance among different types of UAVs.

Aforementioned MARL algorithms have been optimized to address the demands of UAV swarm path planning [25]–[29] and coordinated tasks [30]–[33] in UAV formations. These optimized algorithms are capable of handling simple coordination tasks. However, they still face challenges with complex coupled tasks, including slow convergence, limited scalability, and effectiveness only in specific scenarios.

To address these shortcomings, this paper proposes an MCDDPG algorithm for handling coupled tasks. MCDDPG builds upon the DDPG algorithm, which combines value and policy functions. Specifically, it introduces a Multi-Critic (MC) architecture to enhance the optimization of policy networks. Unlike traditional MARL approaches where a single value function guides policy function optimization, the MC architecture allows the algorithm to utilize multiple Critic modules with diverse value functions. Some Critic modules are model-based, focusing on tasks like tracking, obstacle avoidance, and collision prevention, while others are deep neural network-based, guiding policy networks in coordinating actions among agents.In this paper, an application scenario is designed to verify the practicability of the MCDDPG algorithm. The MCDDPG algorithm is used to realize the test task of multi-angle cooperation of UAV formation, which verifies that the algorithm has certain practicability and versatility in the field of multi-UAV multi-target cooperative control, and can provide multi-angle image information for small targets. It can assist the extraction of target feature information [34] and improve the average accuracy of small target detection tasks [35].

In summary, the contributions of this paper are as follows:

1) This paper designs a MC framework and proposes MCDDPG algorithm. The framework solves the problem that the A2C architecture adopted by MARL algorithm acts randomly in the early stage of training, which makes it difficult to explore the effective empirical data.
2) A Dynamic Difficulty Priority(DDP) Weight Optimization algorithm is proposed to adjusts the evaluation weights of Critic modules in an MC architecture dynamically. This method prevents the performance degradation due to improper weight allocation among different Critic modules.

The structure of the subsequent chapters is as follows. Section II provides the physical model of the UAV and describes the problem statement for the coupled cooperative planning tasks. Section III provides a detailed description of the proposed MCDDPG algorithm and the mathematical principles and ideas behind the multi-critic module weight optimization algorithm. Section IV presents an analysis of the algorithm's simulation results. Finally, Section V presents the conclusions and a summary of the approach proposed in this paper.

II. Physical Model and Problem Description

The coordination planning of UAVs is subject to constraints from three aspects: the UAV themselves, the environment, and the Target. Therefore, it's necessary to define the physical attributes of the UAVs, Obstacles, and Targets to facilitate the subsequent training of deep reinforcement learning algorithms.

A simulation mathematical model is established for UAVs to address their characteristic properties.

$$\begin{cases} Uav = Uav_i(i = 1, 2, 3, ..., n) \\ Uav_{attr} \ = \ [p, r_s, v, a, c, \theta] \end{cases} \tag{1}$$

In the Uav_{attr} model, p stands for the coordinates of the UAV's position, r_s is used to denote the safe operating radius, v represents the current velocity of the UAV, a is the current acceleration of the drone, c signifies whether UAV can access each other's current states, and θ represents the UAV orientation Angle, the physical model of the UAV is shown in Fig.1.

The environment is primarily composed of obstacles, a target, and others UAVs. Essentially, for each individual UAV,

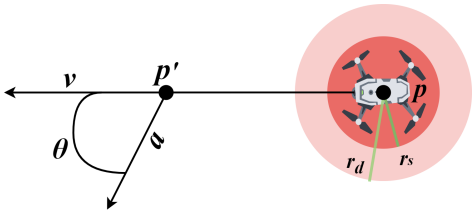


Fig. 1. Physical Model of UAV

any objects excluding themselves are considered as environmental states. The action strategies of the UAVs must be tailored in response to the various environmental variables. Consequently, it is crucial to model the attributes of the environment. The environment attributes are shown in Eq.(2), Eq.(3) and Eq.(4):

$$\begin{cases} Obs &= Obs_i(i = 1, 2, 3, ..., n) \\ Obs_{attr} &= [p, r_c] \end{cases} \tag{2}$$

$$Tar_{attr} = \begin{cases} [p, r, v, a, \theta], & movable = True \\ [p, r_v], & movable = False \end{cases} \tag{3}$$

$$Col_s^i = Uav_s^i(s = 1, 2 \cdots, n, i = 1, 2, 3, \cdots, n, i \neq s) \tag{4}$$

In these model, *Obs* is a set that comprises multiple obstacles, with Obs_{attr} serving as the attribute model for these obstacles. Within Obs_{attr}, p signifies the position coordinates of the obstacle, and r_c denotes the collision radius of the obstacle. Tar_{attr} is used to represent the Target attribute model, which can be divided into two categories: immovable targets(indicated by $movable = False$) and movable targets(indicated by $movable = True$). For immovable targets, the attributes are limited to the coordinate point p and the effective radius r_v. On the other hand, movable targets possess additional attributes such as velocity v, acceleration a, and direction deviation angle θ. Lastly, Col_s^i consists of other UAVs that may collide with each other.

When Unmanned Aerial Vehicles (UAVs) are tasked with flight path planning, they generally ascend to a predetermined altitude before embarking on their missions. As such, it can be inferred that UAVs, obstacles, and targets objects all interact within the same plane. The Eq.(5) describes the kinematic model for dynamic objects within this environment:

$$\begin{cases} v_x = v_t \times \cos\theta \\ v_y = v_t \times \sin\theta \\ v_{t+\triangle t} = v_t + a_t \times \triangle t \\ x_{t+\triangle t} = x_t + v_x \times \triangle t \\ y_{t+\triangle t} = y_t + v_y \times \triangle t \end{cases} \tag{5}$$

In the equations, v_x,v_y denote the components of velocity v_t along the x and y axes. x_t and y_t denote the position of UAV at time t. v_t and θ_t are the velocity and orientation angle at time t. a_t is the acceleration at time t; $\triangle t$ is the motion state sample interval. As shown in Fig.2.

Simultaneously, constraints are imposed on both the state spaces and action spaces of the UAV and the target to more realistic in the scene design. The constraints on the action

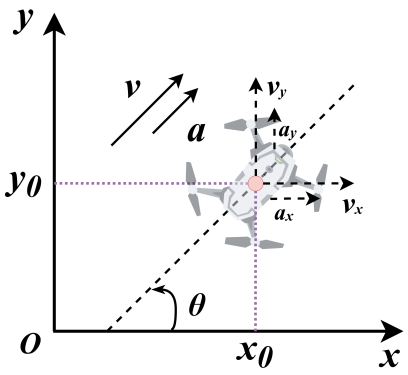


Fig. 2. Kinematic model of the UAV and the target

space include limitations on velocity v_u, v_r and acceleration a_u, a_r. In the state space, constraints are applied to the total number of actions *steps*(reflecting energy constraints of the UAV), the perception distance r_d (the environment perceived by the UAV is dynamically changing), and the bounds on the operational range x_u, x_r, y_u, y_r.

The MCDDPG algorithm needs to complete the cooperative strategy exploration of UAV swarm under this constraint condition. All constraints are expressed as follows:

$$\begin{cases} 0 \leq v_u \leq \max(v_u) \\ 0 \leq v_r \leq \max(v_r) \\ 0 \leq a_u \leq \max(a_u) \\ 0 \leq a_r \leq \max(a_r) \\ steps \leqslant STEP \\ r_d = r_s \times d \\ lb \leq x_u, x_r \leq ub \\ lb \leq y_u, y_r \leq ub \end{cases} \tag{6}$$

The $\max()$ function is utilized to represent the maximum constraint, which is applied to limit the maximum speed and acceleration of both the UAVs and targets. The number of action steps of the UAV must not exceed the maximum energy step limit $STEP$. The perception range r_d is d times the safe collision radius r_s. The lower and upper bounds of the coordinate axes are denoted by *lb* and *ub* respectively, and these are employed to confine the range of the operational space.

To validate the algorithm's relevance in real-world environments, it is necessary to incorporate various primary constraints encountered in practical considerations into the simulation scenarios. Currently, the physical constraints of UAV primarily focus on three aspects: communication limitations [36]–[38], energy constraints [39], [40], and environmental restrictions [41], [42]. In terms of communication, unmanned devices typically rely on wireless communication; however, existing signal transmission scenarios mainly address static environments [43]. Regarding energy constraints, fixed-wing UAVs require gradual adjustments in tilt, speed, and acceleration when turning, and operating at high angles and speeds increases energy consumption [40].

To address these limitations, when utilizing the MCDDPG algorithm to learn optimal control strategies, this paper constrains the maximum action steps ($STEP$), speed (v),

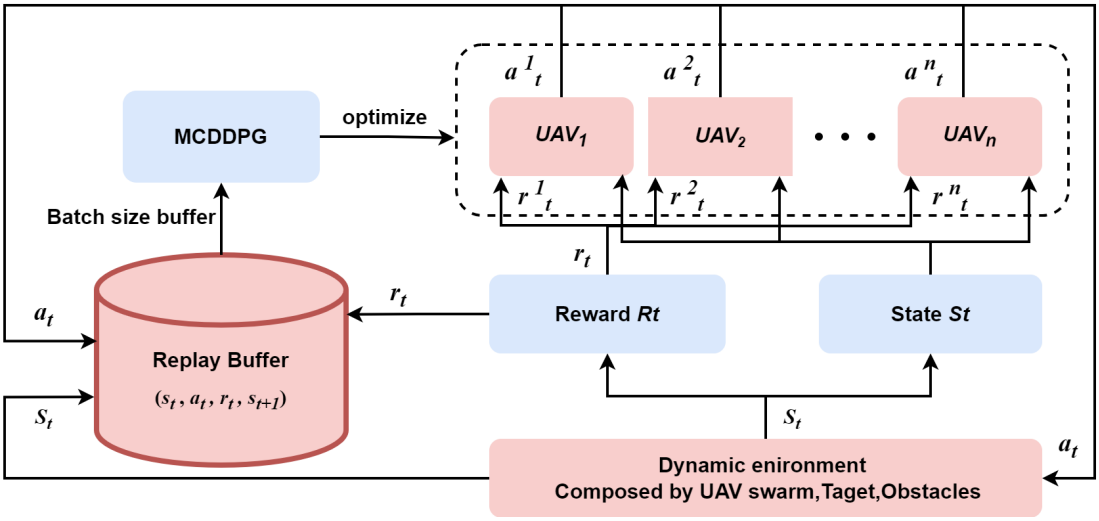


Fig. 3. MARL architecture based on MCDDPG algorithm

acceleration (a), and direction angle (θ) during model training. The other UAVs is regarded as part of the environment to mitigate the adverse effects of various communication problems. And we introduces the concept of a visibility range r_d as a constraint to address obstacle avoidance and formation coordination in dynamic environments during UAV formation target tracking.

Although the simulation scenario considers major influencing factors encountered by UAVs in real operational settings, there are still numerous other factors such as wind speed, humidness and magnetic field strength limitations. These limitations affect the control accuracy and long-term service life of Uavs to some extent, they have little relevance for the mission scenario studied in this paper, so these limitations are not considered.

III. Proposed Multi-agent Reinforcement Learning Framework

This paper proposes a MARL algorithm solving the problem of UAV swarm cooperative planning, as shown in Fig.3. Firstly, we model this problem as a Markov game and employ the MCDDPG algorithm to train the action strategy network parameters of the UAV formation. Secondly, considering that the execution of cooperative tasks involves coupled tasks and requires complex optimization, we design an MC architecture in this paper. MC architecture enables UAV formation to optimize decision exploration and accelerate the improvement of cooperative planning capabilities. Ultimately, we obtain action strategies for UAV formation cooperative tasks.

A. Scalable MC Architecture

In MARL algorithms, the A2C architecture is commonly used. Both the Critic and Actor modules in A2C are constructed using deep neural networks (DNN). The Critic module, formed by a Deep Neural Network , acts as a proficient mentor guiding the agent to explore optimal decision-making strategies. However, it lacks practical experience. This paper introduces the MCDDPG algorithm based on the Multi-Critic (MC) architecture. Unlike A2C, MC employs multiple Critic to guide the agent's learning, as depicted in Fig.4.

Some Critic components, akin to Experts, are composed of specific models endowed with prior knowledge and substantial practical experience. These models advise the agent on action selection in specific scenarios. Other Critic modules consist of deep neural networks and are model-free, dedicated to learning scenarios that Experts cannot resolve. MC architecture by employing different types of Critics, the exploration process of agents is streamlined, thereby enhancing the efficiency of agents training.

Fig. 4. Each Critic module in MC architecture jointly trains the agent action policy

The MCDDPG algorithm extends the DDPG framework by introducing a Multi-Critic architecture, as shown in Fig.5. This MC architecture incorporates multiple Critic modules that collectively evaluate and guide the Actor module. The agent's historical information is stored in an experience pool, and DNN are used to extract features from the trajectories of the agent's historical actions, approximating the optimal action strategy.

The policy network μ is used to output the probability distribution of actions for the agent given its current state. The Multi-Critic modules synthesize evaluations on the effectiveness of current action executions. These evaluations are then used to update the parameters within the agent's policy network.

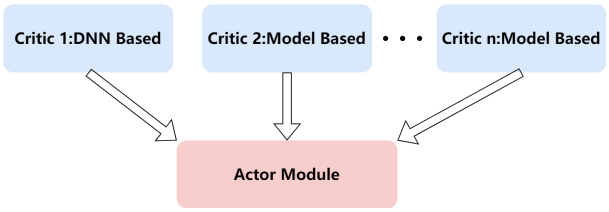


Fig. 5. Multi-Critic architecture

Therefore, for the total evaluation Q-value for the current action output by composed of MC rchitecture is shown in Eq.(7) and Eq.(8):

$$Q = \beta_1 Q_1 + \beta_2 Q_2 + ... + \beta_n Q_n \tag{7}$$

$$\beta_1 + \beta_2 + ...\beta_n = 1 \tag{8}$$

In the equation, Q represents the evaluation value of the current UAV action, $Q_1, Q_2, ..., Q_n$ is the critic module of n different principles, $\beta_1, \beta_2, ..., \beta_n$ is the weight of these critic modules. Through the MARL algorithms comprising MC architecture, theoretically, issues such as the increase in the number of UAVs or Targets, high task coupling, and exploding dimensions in backpropagation optimization can be addressed.

By combined model-based Critic modules (such as cluster grouping models for UAVs, target selection models, etc.) with the Critic modules composed of DNNs, model can converge efficiently, improve of UAVs' ability to coordinate actions in complex task environments, thereby enhancing exploration and coordination capabilities.

B. Complexity analysis

As the number of UAVs and targets continues to increase, it is necessary to analyze the changes in data complexity within the empirical data. The complexity of the data in the experience pool directly impacts the convergence performance of the RL algorithm and the design of MC architecture. Therefore, analyzing data complexity is significant for constructing the MC architecture.

This paper utilizes Differential Entropy(DE) to evaluate the uncertainty and complexity of the data in the experience pool. DE serves as a measure of the uncertainty of continuous random variables, where a higher value indicates greater data complexity. As shown in Eq.(9).

$$DE\left(X_{DE}\right) = -\int p\left(x\right) \log\left(p\left(x\right)\right) dx \tag{9}$$

In the equation, $DE\left(X_{DE}\right)$ represents the calculated value of the DE, where X_{DE} denotes the experience replay buffer data of each UAV or Target, and $p\left(x\right)$ represents the probability of occurrence of data x in X_{DE}.

TABLE I
EMPIRICAL REPLAY BUFFER DATA COMPLEXITY ANALYSIS

Indicator	3 UAVs and 1 Target	6 UAVs and 2 Target	9 UAVs and 3 Target
Dimension	12	28	40
Differential Entropy	0.450	0.643	0.855
Mutual Information	0.725	0.607	0.479

Mutual information(MI) indicates whether there is a relationship between the data of two agents, as well as the strength of that relationship. It serves as a metric for measuring the correlation between two variables. A larger value suggests a greater influence of the target's actions on the UAV, indicating stronger correlation between them. Conversely, a smaller value implies weaker correlation between a single UAV and the target. The calculation is shown in Eq.(10).

$$MI\left(X,Y\right) = \int_Y \int_X p\left(x,y\right) \log\left(\frac{p\left(x,y\right)}{p\left(x\right) \cdot p\left(y\right)}\right) dxdy \tag{10}$$

In the Eq.(10), $MI\left(X,Y\right)$ represents the mutual information between the UAV's experience data X and the Target's experience data Y. $p\left(x,y\right)$ denotes the joint probability distribution of X and Y occurring together, while $p\left(x\right)$ and $p\left(y\right)$ represent the individual probabilities of X and Y, respectively.

As shown in Table I, with the increase of the number of UAVs and targets increases, the dimensions of the experience replay buffer become more numerous, leading to increased data complexity. Additionally, the DE of the experience replay buffer data gradually increases, indicating a rise in data disorder. The MI value decreases, suggesting a diminishing correlation between individual UAV and individual target, while the complexity of the collaborative task progressively increases.

The analysis results indicate that as the number of UAVs and targets increases, the complexity of the tasks also gradually rises. If conventional reinforcement learning structures were employed, the time cost incurred would be substantial. However, the MC architecture can reduce unnecessary exploration through MC modules, thereby saving exploration time while still achieving a robust policy network for completing collaborative tasks. This effectiveness of the MC architecture in complex scenarios is further demonstrated in the subsequent experimental results in Section IV.

C. Method and principle

This section utilizes the MCDDPG algorithm to tackle the specific task of multi-angle coordination among UAV clusters. In this task scenario, factors such as dynamic changes in obstacle environments during task execution and complex adjustments in UAV group path coordination are taken into account. The MC architecture is specifically designed with two Critic modules, Critic1 and Critic2, to cater to the needs of this task scenario. The value function of the Critic1 module is approximated by an MLP network, while the Critic2 module employs an gravitational field model for guidance. Furthermore, the DDP (Dynamic Difficulty Prioritization) algorithm is used to dynamically optimize the weights of the two Critic

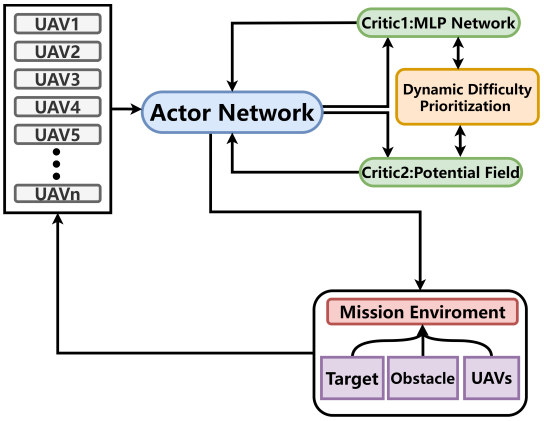


Fig. 6. Specific architecture of MCDDPG algorithm in UAV swarm cooperative task scenario

modules. The specific structure of the MCDDPG algorithm is depicted in Fig.6.

The specific design of the Critic1 and Critic2 modules within the MC architecture is depicted in Fig.7. During the initial phase of the follow-tracking task, the parameter updates of the Actor module's policy network should primarily focus on the value network of the Critic2 module. This phase mainly considers the attractive force between the UAV and the target point, and the repulsive force between the obstacles to ensure to move towards the target and avoidance collisions. Once the UAV formation reaches the target points, the task transitions to multi-angle coordinated formation planning. In this phase, the focus should shift towards the value network of the Critic1 module, utilizing MLP to delve into deeper cooperative relationships among UAV groups within a localized area. This enables strategic coordination actions among UAV formations in small-scale movements.

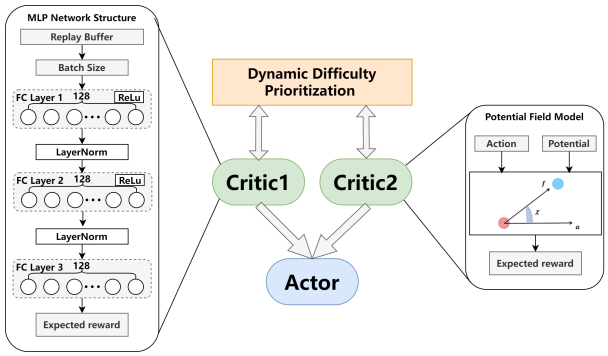


Fig. 7. Critic1, Critic2 modules consisting of MLP neural network and gravitational field model

The text below elucidates the Critic1 and Critic2 modules within the MC module, the Actor module, and the policy gradient update method of MCDDPG.

The Critic1 module employs the value network from the DDPG algorithm to assess the action output of the actor model. It investigates the update methods for cooperative strategies for UAV swarms within a limited range, with the objective of achieving multi-angle encirclement deployment planning around target points. The value function of this Critic1 network is represented by Eq.(11), γ_1 indicating the conversion coefficient of Critic1.

$$Q_{C1}(s,a) \triangleq E\left[\sum_{k=1}^{T}\gamma_1^{k-1} r(s_k,a_k) \mid s_1=s, a_1=a\right] \quad (11)$$

The value network of the Critic2 module is constructed based on the gravitational field model. The attraction model, a method for route planning, calculates the attraction and repulsion between the agent and objects, taking into account their distances and action tendency relationships. It operates under the assumption that the dynamic model of the environment is known, with the target generating attraction towards the UAV and obstacles creating repulsion. The movement direction and path strategy of the UAV formation are planned in accordance with the resultant force of attraction and repulsion. The gravitational model between the target and the UAV is defined as per Eq.(12).

$$F_{gra}(i) = \frac{1}{2}\varepsilon \times d^2\left(p^{Uav^i}, p^{tar}\right) \quad (12)$$

In the equation, ε represents the gravitational coefficient used to quantify the strength of attraction exerted by the target on the UAV. p^{UAV^i} and p^{tar} are positional attributes of the UAV and the target, respectively, where d(·) is used to calculate the Euclidean distance between them. The repulsion force model is represented by Eq.(13).

$$F_{rep}(i) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{d\left(p^{Uav_i}, p\right)} - \frac{1}{d_0}\right)^2 & if\ d < d_0 \\ 0 & if\ d \geqslant d_0 \end{cases} \quad (13)$$

In the given equation, η symbolizes the repulsion coefficient, while d_0 represents the safe operational distance for the UAV. As illustrated in the equation, when the distance between the UAV and obstacles, other UAVs, or the target falls below the safe distance, the UAV will encounter a repulsion force. On the other hand, when the distance between the UAV and other objects surpasses the safe distance, indicating no risk of collision, the repulsion force experienced becomes zero.

The cumulative force of repulsion and attraction experienced by the UAV during its operation is termed as the UAV's gravitational field, which is represented by Eq.(14).

$$F(i) = F_{gra}(i) + F_{rep}(i) \quad (14)$$

By integrating the gravitational model into the algorithm via the MC architecture, preliminary information is supplied to steer the UAV towards exploration in the direction of attraction. In other words, it approaches the target position while distancing itself from other UAVs and obstacles. Consequently, in the MCDDPG algorithm, the Critic2 network, which is built using the gravitational model, is defined by Eq.(15).

$$q_F(s,a) \triangleq -F(s)\left[1 - cos(\chi)\right] \quad (15)$$

As illustrated in Fig.8, the angle χ between the actual action values derived from the action space and the guided actions from the gravitational field model is depicted. This suggests

that as the angular difference between the two escalates, the influence of the gravitational field on the UAV's output action strategy intensifies. This can be harnessed to rectify the UAV's irrational exploration directions.

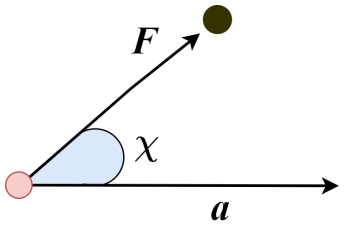


Fig. 8. The angle between the direction of the UAV's acceleration and the gravitational force

Hence, the integration of the gravitational model as the Critic2 module in the MCDDPG algorithm can be articulated as Eq.(16), where γ_2 symbolizes the weighting coefficient of Critic2 and $step$ denotes the action step.

$$Q_F(s,a) \triangleq E\left[\sum_{k=1}^{step} \gamma_2^{k-1} q_F(s_k,a_k) \,|s_1=s, a_1=a\right] \quad (16)$$

The Critic2 module, grounded in the gravitational model, facilitates a unified evaluation of action space selection for UAV swarms. Each UAV utilizes the same Critic to guide the update of action planning strategies, thereby reducing the randomness in decision-making among multiple agents in dynamically changing environments. This process accelerates and stabilizes the exploration process of the Actor module.

The action policy network, Actor, for the UAV is composed of a multi-layer perceptron (MLP). Its aim is to secure the optimal reward values for the state distribution space and the corresponding action space. The Actor module computes the probabilities of actions that yield the highest reward values for the current state and future states at this step, iteratively updating the policy network. The policy network is represented as Eq.(17).

$$\rho^u(s',\gamma) = \int_s \sum_{t=1}^{step} \gamma^{t-1} p_1(s)\, p(s \to s', t, \pi)\, ds \quad (17)$$

In the given equation, $p_1(s)$ signifies the initial distribution of the state space s , $p(s \to s', t, \pi)$ denotes the probability distribution of transitioning from state space s to $s\prime$ at time t when executing the action outputted by the policy network, and γ is the discount factor of the policy network, which serves to mitigate the influence of future actions on the current actor decision.

As outlined above, the objective function of the MCDDPG network is steered by both the Critic1 and Critic2 modules to update the Actor module. The aim is to secure the maximum reward value for the UAV over the course of an episode trajectory. The objective function is represented by Eq.(18).

$$\begin{aligned} J(\mu_\theta) = \beta \int_s \rho^\mu(s,\gamma_1)\, r(s,\mu_\theta(s))\, ds \\ + (1-\beta) \int_s \rho^\mu(s,\gamma_2)\, q_{PF}(s,\mu_\theta(s)) \end{aligned} \quad (18)$$

In the equation, $r(s,\mu_\theta(s))$ represents the reward function corresponding to the impact of the current action on the state, This is determined by the task requirements of the UAV swarm, and additional explanations will be provided regarding the reward function. As depicted in the equation, the first part of the objective function is based on the MLP-based Critic1 module, while the second part is grounded in the gravitational model-based Critic2 module. β represents the weight parameters of the two Critic networks.

By utilizing Eq.(19) to update the gradient of the objective function, we train to secure the optimal actor network parameters, thereby accomplishing multi-perspective cooperative planning tasks for UAV formations.

$$\begin{aligned} \nabla_\theta J(\mu_\theta) = \beta E_{s\sim\rho^\mu(s,\gamma_1)}\left[\nabla_\theta \mu_\theta(s)\, \nabla_a Q_{C1}(s,a)\,|_{a=\mu_\theta(s)}\right] \\ + (1-\beta)\, E_{s\sim\rho^\mu(s,\gamma_2)}\left[\nabla_\theta \mu_\theta(s)\, \nabla_a Q_F(s,a)\,|_{a=\mu_\theta(s)}\right] \end{aligned} \quad (19)$$

D. Reward function

In MARL, an agent's policy network updates its parameters based on historical data stored in the experience replay buffer. By assigning reward values to historical actions and environmental states, the agent assesses the performance of various actions under different environmental conditions. These reward values enable the agent to conduct backpropagation, guiding the optimization of its policy network parameters.

This paper employs a sparse reward function to guide the policy network updates for the agent. This reward mechanism aligns more closely with human intuition and experience, enhancing the agent's exploratory capabilities. With sparse rewards, the agent receives rewards only upon reaching intermediate goals or completing the task, which also increases the difficulty of optimal policy convergence for various algorithms. The reward values are shown in Table II.

TABLE II
REWARD FUNCTION

Description	Values
Task completion	10
Target detection	1
Action steps	-0.1
Collision or out of the boundary	-1

A task completion reward is given when the UAV successfully performs multi-angle tracking of the target. During task execution, a collision penalty is applied if a collision occurs. A action steps penalty is also introduced to encourage the UAV to complete the collaborative task in as few decision steps as possible. Additionally, a detection reward is provided to guide the UAV in actively searching for the target.

Hence, the reward function for an individual UAV is represented by Eq.(20).

$$R_i = r_1 + r_2 + r_3 + r_4 \quad (20)$$

The overall reward score for the agent is expressed as Eq.(21).

$$R^T = \sum_{i=1}^{n} R_i \tag{21}$$

R^T represents the sum of the reward functions for all UAVs participating in the cooperative task. In the MCDDPG algorithm, it is necessary to search for the highest reward for each individual UAV, while also aiming to maximize the overall reward value R^T for the entire UAV formation.

E. Weight Optimization for multi-critic module

The weight value β of the MC module significantly influences the policy planning performance of the MCDDPG algorithm. Determining an optimal value for β is essential to maximize the performance of the MCDDPG algorithm and improve the success rate of multi-angle cooperative tasks for UAV formations.

In this MC architecture, the Critic1 module Q_{C1} is based on MLP, while the Critic2 module Q_{C2} employs a gravitational field model. These two modules collaboratively evaluate the actions output by the Actor module μ. The Critic2 module uses a fixed model, where it only necessitates the input of parameters to obtain the action evaluation values it focuses on. Therefore, in the MCDDPG algorithm, updates are required for the MLP network parameters w in the Critic1 module and the Actor module netword parameters θ. The network model expressions for these two modules are as follows:

$$\mu\left(s|\theta\right) = MLP_{Actor}\left(s|\theta\right) \tag{22}$$

$$Q_{C1}\left(s,a\right) = MLP_{C1}\left(s,a|w\right) \tag{23}$$

s represents the state, w are the parameters of Critic1's MLP network, and θ are the parameters of the Actor's policy network.

$$a_t = \mu\left(s_t|\theta\right) \tag{24}$$

The Critic1 module takes the action a_t and state s_t of UAV at time t as inputs into its value estimation network, and subsequently provides the reward evaluation value V_{C1} for the action selected by UAV.

$$V_{C1} = Q_{C1}\left(s_t,a_t\right) \tag{25}$$

In the training process, the MCDDPG algorithm persistently updates the weight parameters w of the Critic1 module. This is done to achieve precise evaluations of the reward values for actions. The algorithm employs the root mean square error (MSE) as the error function, which is the difference between the evaluation outputs V_{C1} and the actual reward values y_i. By minimizing this error, the Critic1 module is updated. y_i is obtained by adding the true reward at time step t and the evaluated value of the action $\mu\prime$ at time step $t+1$ multiplied by the distance discount coefficient γ_1.

$$\begin{cases} L_1 = \dfrac{1}{N}\sum\limits_{i}^{N}\left(y_i - V_{C1}\right)^2 \\ y_i = r_i + \gamma_1 Q\prime_{C1}\left(s_{t+1}, \mu\prime\left(s_{t+1}|\theta\prime\right)|w\prime\right) \end{cases} \tag{26}$$

For Critic2, at time step t, both the state and action at time t are directly fed into the Critic2 module. This is done to generate the current action's value evaluation V_{C2} under the gravitational field model Q_F, error value L_2 is equal to the evaluated value V_{C2}.

$$\begin{cases} L_2 = V_{C2} \\ V_{C2} = Q_F\left(s_t,a_t\right) \end{cases} \tag{27}$$

Therefore, by minimizing the L_1 error function, the parameters w of the MLP network in the Critic1 module are updated to their iterated values $w\prime$. In this equation, τ represents the learning rate parameter, which is used to adjust the step size of the network parameter updates.

$$\tau w + \left(1-\tau\right) w\prime \rightarrow w\prime \tag{28}$$

The evaluation values, produced by the Critic1 and Critic2 modules that form the MC architecture, directly steer the update of the weight parameters in the Actor module.

$$\tau\theta + \left(1-\tau\right)\theta\prime \rightarrow \theta\prime \tag{29}$$

For the overall MC module, the total value function Q is defined as follows:

$$Q\left(s_t,a_t\right) = \beta_1 Q_{C1}\left(s_t,a_t|w\right) + \beta_2 Q_F\left(s_t,a_t\right) \tag{30}$$

Therefore, under the guidance of the MC architecture and in accordance with the gradient ascent update formula presented in Eq.(18), policy network update formula for the MCDDPG algorithm as shown in Eq.(31).

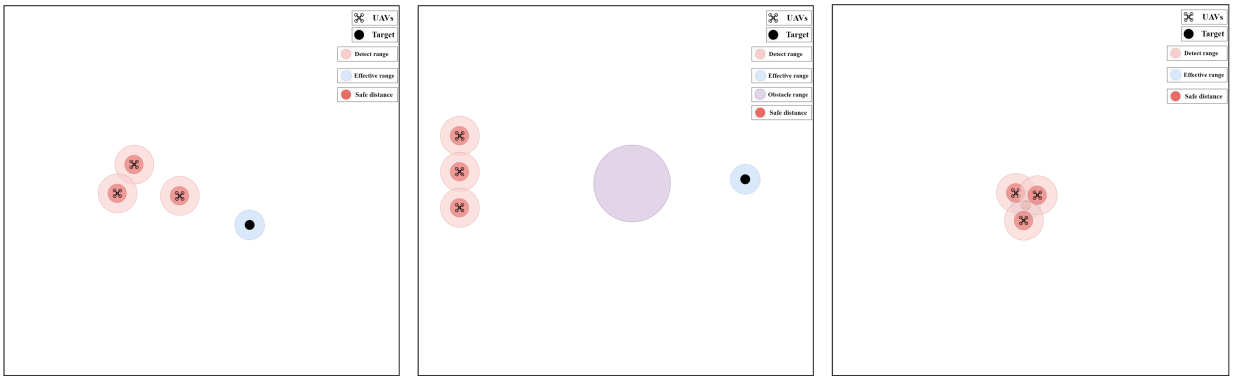
$$\begin{aligned} \nabla_\theta J\left(\mu_\theta\right) = \frac{1}{N}\sum_i \nabla_\theta \mu\left(s_i|\theta\right)\left[\beta_1 \nabla_w Q_{C1}\left(s_i,a_i|w\right)\right. \\ \left. +\beta_2 \nabla_a Q_F\left(s_i,a_i\right)\right] \end{aligned} \tag{31}$$

Reference [44], [45] dynamically adjusts the weights of the multiple loss functions based on the rate of gradient descent. The principle of the multi-Critic structure utilized in this paper bears conceptual similarity to the adjustment of weights for multiple loss functions. Consequently, this paper introduces a DDP method. The DDP method dynamically allocates weights based on the changes in the gradients of the error functions: the more the gradient of the error function L_i for the i-th Critic module changes during iteration of the whole error function L_k, the higher the weight it is assigned, as illustrated in Eq.(32).

$$DDP\left(\beta_i\right) = \frac{\nabla L_i}{\sum_{k=1}^{N}\nabla L_k} \tag{32}$$

For the MCDDPG algorithm designed in this paper, the variable $k=2$.

When the gradient of the evaluation error for Critic1, denoted as ∇L_1, exhibits significant variation, it is assigned greater weights. This prioritizes the guidance of UAVs in collaborative exploration tasks. On the other hand, when the evaluation error variation for Critic2, denoted as ∇L_2, is substantial, it concentrates on guiding UAVs in tracking and obstacle avoidance exploration tasks. This dynamic adjustment



(a) Subtask 1: Track the target (b) Subtask 2: Obstacle avoidance (c) Subtask 3: Multi-angle collaboration

Fig. 9. Simulation of subtasks for multi-angle collaborative planning: (a) target tracking, (b) obstacle avoidance, and (c) Bounding collaboration

strategy aims to optimize the weight β_i of the MC module, thereby enhancing the training efficiency of the MCDDPG algorithm.

Therefore, the iterative update formula of the MCDDPG algorithm after optimization with DDP is shown as Eq.(33).

$$\nabla_\theta J(\mu_\theta) = \frac{1}{N}\sum_i \nabla_\theta \mu(s_i|\theta)\left[DDP(\beta_1)\nabla_w Q_{C1}(s_i,a_i|w) + DDP(\beta_2)\nabla_a Q_F(s_i,a_i)\right] \tag{33}$$

Based on the principles and formulas outlined in this section, the weight assignment of the MC architecture can be dynamically adjusted to further improve the algorithm performance. The proposed MCDDPG approach is summarized in Algorithm 1.

F. MCDDPG convergence

To ensure the convergence of the MCDDPG algorithm, we analyze the convergence of each core component: the policy gradient update, the Q-value update, and the joint convergence of multiple agents.

In MCDDPG, each UAV has an independent policy and a Q-value network . The objective of each agent's policy update is to maximize the expected return under the current estimate of the Q-value function. The policy gradient update rule is defined as Eq(19).

Monotonic Improvement of Policy Updates.The policy gradient method ensures that each update increases the objective function Eq.(18) $J(\mu_\theta)$ [14].For the policy gradient update to converge, the following conditions are required:(1) The learning rate should be sufficiently small to ensure stable gradient updates.(2) The Q-value estimate Q must gradually approach the true Q-value function (discussed further in the Q-value convergence section).(3) Assuming the action space $A \neq \infty$ is compact ensures that the gradient remains bounded. Given these conditions, the policy gradient update will converge to a optimum, thereby ensuring monotonic improvement of the policy with each update.

Convergence of the Q-Value Update. Q-value update follows the Bellman equation, minimizing the target error to update the Q-value network parameters. The Q-value update rule is as follows:

$$w \leftarrow \underset{w}{arg\,min} E_{(s,a,r,s')\ D}\left(Q_w(s,a) - y_i\right)^2 \tag{34}$$

y_i is defined as Eq.(26).

The Q-value update can be viewed as an application of the Bellman operator T^μ. For a given state-action pair (s,a), the

Algorithm 1 MCDDPG algorithm

Initialize the network parameters of Critic1 module w and Actor module θ.
Receive initial observation state S_1
for episode = $\{1,2,\ldots,M\}$ **do**
  Initialize a random process $\mathcal{N}$ for action exploration for each UAV.
  Receive empirical data of the random exploration round
  for $t = 1, STEP$ **do**
    Select action a_t according to the current policy $a_t = u(s_t|\theta)$
    Execute action a_t and observe reward r_t and next state s_{t+1}
    Compose data groups $\overline{s_t} = \left(s_t, a_t^1, a_t^2, ..., a_t^n\right)$ and $\overline{s_{t+1}} = \left(s_{t+1}, a_{t+1}^1, a_{t+1}^2, ..., a_{t+1}^n\right)$, according to the actions $a_t^{i\in n}$, $a_{t+1}^{i\in n}$ at time t and time $t+1$ of each UAV in the formation, and states s_t, s_{t+1}.
    Store transition $(\overline{s_t}, a_t, r_t, \overline{s_{t+t}})$ in Replay buffer.
    Sample a batch transitions $(\overline{s_t}, a_t, r_t, \overline{s_{t+t}})$ from Replay buffer.
    Set y_i: $y_i = r_i + \gamma_1 Q'_{C1}(s_{t+1}, \mu'(s_{t+1}|\theta)|w')$.
    Update critic 1 by minimizing the loss: $L_1 = \frac{1}{N}\sum_i^N (y_i - V_{C1})^2$.
    Calculate the value of critic 2: $V_{C2} = Q_F(s_t, a_t)$
    Calculate the β_i by DDP algorithm according to Eq.(32).
    Calculate the Q value of MC architecture: $Q(s_t,a_t) = \beta_1 Q_{c1}(s_t,a_t|w) + \beta_2 Q_F(s_t,a_t)$.
    Update the actor policy according to Eq.(33)
  end for
end for

Bellman operator is defined as:

$$T^{\mu}Q\left(s,a\right)=r\left(s,a\right)+\gamma E_{s'}\left[Q\left(s',\mu\left(s'\right)\right)\right] \tag{35}$$

According to Bellman's optimality principle, repeated application of T^{μ} the Q-value function to converge to the optimal Q-value Q^{*}.

In summary, Under this framework, the multi-agent joint policy can be represented as a collective policy vector $\pi=(\pi_1,\cdots,\pi_n)$. With each update, policies and Q-values undergo a convergence iteration. Due to the monotonicity of the Bellman operator and policy gradient update, the system will converge to a Nash equilibrium.

IV. Analysis And Disscussion Of Simulation Results

In this section, we implement the physical models of UAVs, targets, and obstacles in the OpenAI Multi-Agent Particle Environment (MPE). We devise simulation experiments for various scenarios to evaluate the performance of the MCDDPG algorithm. We compare the MCDDPG algorithm with other MARL algorithms such as MADDPG, MAPPO, MASAC, IPPO [46] and ISAC [47], all utilizing the same DNN architecture. We create diverse simulation scenarios, including tracking, obstacle avoidance, and cooperation, to assess the planning effectiveness of the MCDDPG algorithm in controlling UAV formation in these scenarios, as illustrated in Fig.9 below. We conduct ablation experiments to demonstrate the efficacy of each functional module of the MCDDPG algorithm in optimizing control policies. By comparing performance metrics of the MCDDPG algorithm under dynamically adjusted weight parameters and fixed parameter settings for the optimizer, we validate the enhancement in algorithm performance achieved by the dynamic difficulty priority weight optimization method. Further details of the simulation experiments are provided in the subsequent sections of this article.

A. Parameter setting

Before starting the experiments, it is necessary to initialize the physical properties of the UAVs, target points, and obstacles in the MPE environment. The physical properties of the UAVs and targets include a radius $r=0.035$, velocity $v=0.5$, acceleration $a=1$, and a maximum field of view $c=120$. The obstacles, which are immovable, only require setting their radius to $r=0.2$. Unless otherwise specified, each episode consists of a maximum of 200 $STEPS$, the detection radius parameter d. The hyperparameter Settings of the MCDDPG algorithm are shown in Table III below.

TABLE III
PARAMETER SETTING OF SIMULATION SCENARIO

Parameter	Values
Policy learning rate γ^{θ}	0.001
Critic learning rate γ^{Q}	0.001
Discount factor γ	0.99
Update frequency τ	0.001
Training episodes	500000
Buffer size	300000
Batch size	1024

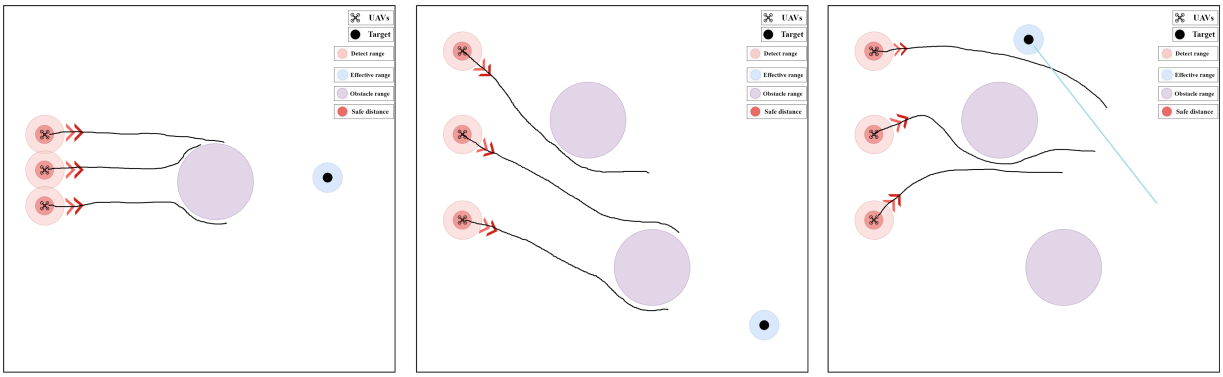
and the physical attribute parameters of the UAV, target, and obstacle are shown in Table IV below.

TABLE IV
PARAMETER SETTING OF SIMULATION SCENARIO

Object	Parameter	Value
UAV	*Radius* r_S	0.035
	Velocity v	0.5
	Acceleration a	0.5
	Cooperation for view angle c	120
	The detection radius d	2
	STEPS	200
Target	*Radius* r_T	0.035
	Velocity v	0.5
	Acceleration a	0.5
Obstacle	*Radius* r_O	0.2

B. Simulation scenarios

To verify the capability of the MCDDPG algorithm in multi-angle cooperative planning tasks for UAV swarm formations, this section designs simulation scenarios 1 to 5 ranging from simple to complex, and from single-task to

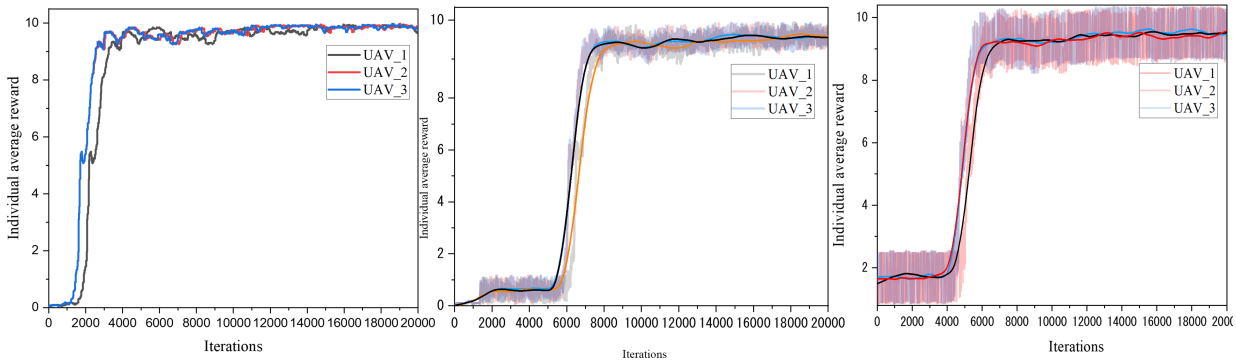


(a) Scenario 1 : UAVs formation obstacle avoidance path planning

(b) Scenario 2 : Multi-angle cooperative planning for static

(c) Scenario 3 : Multi-angle cooperative planning for dynamic targets

Fig. 10. Simulation effect of UAV formation execution action output by MCDDPG algorithm in (a) obstacle avoidance; (b) Multi-angle coordination of static targets; (c) Dynamic target multi-angle cooperation



(a) UAV formation learning curve in training scenario 1

(b) UAV formation learning curve in training scenario 2

(c) UAV formation learning curve in training scenario 3

Fig. 11. Uav formation of three Uavs, learning curve for each UAV

coupled-task scenarios. These scenarios aim to demonstrate the algorithm's ability to achieve excellent cooperative strategies across different task environments.

We first verify the reward value convergence effect of the MCDDPG algorithm under three simple scenarios 1,2,3 as depicted in Fig.10. These scenarios will assess whether the algorithm exhibits robust convergence and whether it can be effectively applied across diverse single task scenarios.

As shown in Fig.11 presents the learning curves of UAV formations across three task scenarios. In scenario 1, the MCDDPG algorithm only needs to address obstacle avoidance and collisions between UAVs, without addressing coordination issues. Therefore, the algorithm can rapidly converge and obtain the optimal obstacle avoidance strategy, as deplicted in Fig.11(a). Despite the increased complexity of environmental conditions in scenarios 2 and 3, the MCDDPG algorithm retains its capability to train UAV formations to excel in executing collaborative tasks. The corresponding learning curves from the training process are illustrated in Fig.12(b) and Fig.13(c).

C. Composition of MC Architecture

As proposed in Chapter 2, the MC architecture requires a combination of model-based and deep neural network-based Critic modules. By extending the model-based Critic module, the algorithm's fitting capability for more complex tasks can be enhanced. In the A2C architecture, the Critic module iteratively updates its network parameters without guidance. During the early training phase, poor evaluation accuracy from the Critic can result in slow updates for the Actor module, making it challenging to explore effective actions. However, the MC architecture can address this shortcoming through the model-based Critic module.

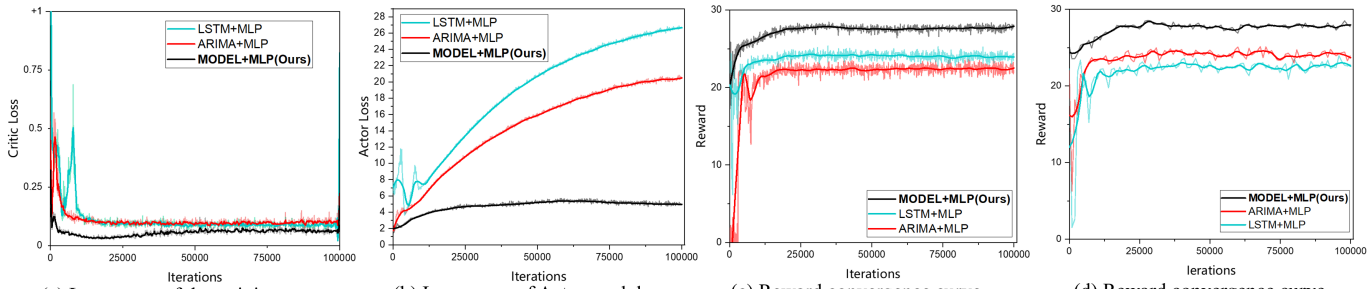
To validate the effectiveness of different MC architecture compositions, we conducted comparative experiments to demonstrate that the combination of the model-based Critic module and the DNN-based Critic module exhibits superior performance compared to other function approximators. We compared the MC architectures consisting of LSTM neural networks and MLP neural networks, ARIMA models and MLP neural networks, as well as the proposed Model-base combined with MLP neural networks. Each configuration was trained on tasks in Scenario 3, with results shown in Fig.12.

From the loss function convergence curves in Fig.12(a) and Fig.12(b), it is evident that the MC architecture composed of the Model and MLP allows for rapid convergence of the loss values for both the Critic and Actor modules. This is attributed to the fact that the model-based Critic module can provide accurate evaluation results when the MLP module's assessment accuracy is insufficient, thereby accelerating the convergence of the MLP network and, subsequently, the Actor module. In contrast, the MC architectures formed by LSTM+MLP and ARIMA+MLP are unable to provide effective evaluations in the early stages of the algorithm, resulting in inferior convergence capabilities compared to the MODEL+MLP configuration.

From the reward value convergence curves in Fig.12(c) and Fig.12(d), it can be observed that the MC architecture composed of MODEL+MLP achieves high reward values early in the training process and continues to improve from that point. In contrast, the MC architecture formed by LSTM+MLP exhibits very low reward values during the initial training phase, with significant fluctuations. This inconsistency arises because the LSTM+MLP architecture extracts different features from the data, making it difficult for the MC architecture to obtain a unified and accurate Critic neural network parameter. The MC architecture composed of ARIMA+MLP benefits from the autoregressive characteristics and linear constraints of ARIMA, resulting in better convergence compared to LSTM+MLP. However, it still does not outperform the MODEL+MLP architecture. The statistical results for each model in Scenario 3 are shown in Fig.13, which also reflects that the MC architecture comprising MODEL+MLP achieves the best task completion performance.

D. Ablation experiments

The ablation study to demonstrate the positive role played by the MC architecture proposed in this paper during agent training. Specifically, we aim to elucidate the guiding roles of Critic1 and Critic2 modules within the MC structure throughout the training process, and to illustrate their significant



(a) Loss curve of the training process of Critic module

(b) Loss curve of Actor module training process

(c) Reward convergence curve during Training

(d) Reward convergence curve during Test

Fig. 12. In Scenario 3, the MCDDPG algorithm training curve. The MC architecture consists of dual DNN network (LSTM+MLP), statistical MODEL and DNN network (ARIMA+MLP), and physical model and DNN network (MODEL+MLP) groups, respectively

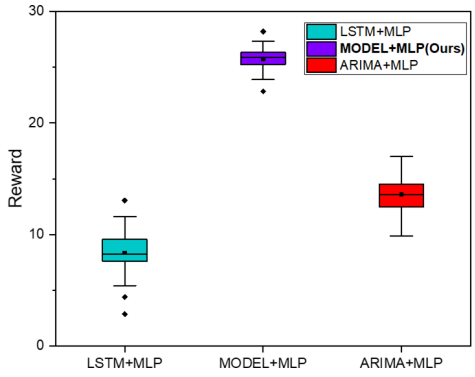


Fig. 13. In Scenario 3, the MCDDPG algorithm training curve. The MC architecture consists of dual DNN network (LSTM+MLP), statistical MODEL and DNN network (ARIMA+MLP), and physical model and DNN network (MODEL+MLP) groups, respectively

contributions to enhancing the decision-making capabilities of the agents in terms of action strategies.

In the ablation study, Scenario 3 serves as the test environment, which includes tasks such as dynamic target tracking, obstacle avoidance, and multi-angle coordination within a UAV swarm. This environment effectively underscores the roles that each Critic module plays in the exploration and training processes of the MCDDPG algorithm. The experimental results, as shown in Fig.14, reveal the following:

Over the training iterations, the MCDDPG algorithm, equipped with the full MC architecture, rapidly progresses through the initial exploration phase. It steers the UAV swarm to update the action strategy network parameters for tasks such as tracking and obstacle avoidance. As a result, the decision model orchestrates the UAV swarm to achieve an average completion rate of nearly 100% in multi-angle cooperative tasks.

When the Critic1 module is removed, the Critic2 module alone forms the value function using an attraction-repulsion field model to guide the UAV swarm's exploration and action decisions. This attraction-repulsion field model evaluates the combined forces of repulsion between UAVs and obstacles, and attraction towards the target, to determine the direction of UAV swarm movement.

In the absence of the Critic1 module, the MC architecture can only guide the UAVs towards the target location based on Critic2's gravitational field model. However, without Critic1, the MC architecture lacks the capability to acquire multi-angle cooperative strategies. It can only guide the UAV swarm in obstacle avoidance and target tracking tasks. This limitation prevents the MCDDPG algorithm from achieving coordinated UAV swarm behavior for multi-angle coordination tasks, as intended.As a result, the training curve labeled NO_Critic1 in Fig.14 fails to converge. This outcome illustrates that the MC architecture, without Critic1, cannot achieve the strategic objective of multi-angle coordination among UAVs, thereby highlighting the indispensable role of Critic1 in enabling the MCDDPG algorithm to achieve coordination task within UAV formation.

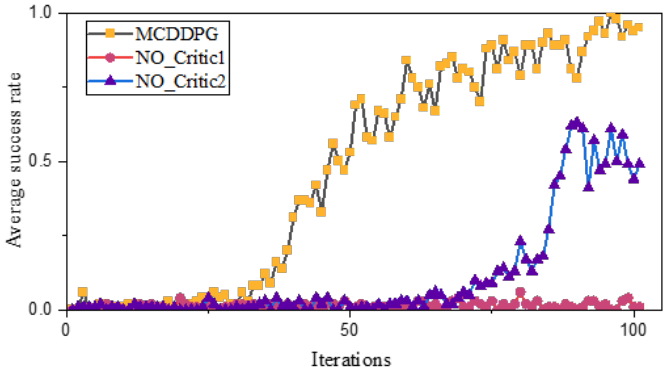


Fig. 14. Ablation Experiment: When $\beta = 0$, Critic1 module is ablated; when $\beta = 1$, Critic2 module is ablated; Otherwise, MCDDPG operates normally. The success rate of UAV formation tasks driven by the MCDDPG algorithm under these three modes is also evaluated.

When the Critic2 module is removed, leaving only the Critic1 module, the MCDDPG algorithm loses the structured guidance provided by the attraction-repulsion field model during the exploration phase for action selection. As a result, agents resort to trial-and-error, leading to less efficient updates of the MLP network weights in Critic1.

In this scenario, agents find it challenging to comprehend even simple tasks such as tracking and obstacle avoidance, due to the absence of structured guidance from Critic2. This leads to a significant increase in the number of training episodes required to achieve understanding and convergence in these tasks, the success rate achieved with only Critic1 is

around 50%, indicating that the agents often fail to effectively coordinate and accomplish tasks without the guidance of Critic2's attraction-repulsion model. The training curve labeled NO_Critic2 in Fig.14 illustrates this extended convergence process compared to the complete MCDDPG algorithm.

Therefore, based on the results of the ablation study depicted in Fig.14, we can conclude that each Critic module within the MC architecture contributes to enhancing the algorithm's convergence capability.

E. Weight optimization

To validate the effectiveness of the DDP algorithm on the MC architecture, this paper compares the success rate convergence curves and average reward convergence curves for the scenarios with optimized β values and fixed β values in both Task Scenario 3 and Task Scenario 4.

In Task Scenario 3, as shown in Fig.15(a) and Fig.15(b), the MCDDPG algorithm employing the DDP algorithm to optimize the weight parameter β converges faster than the algorithm with fixed β values. Additionally, both the success rate and reward values are higher than those of the fixed β value algorithm, and the training process is more stable. This indicates that, in Task Scenario 3, the DDP algorithm can dynamically adjust the β value for the MC architecture based on the difficulty of the current subtask, thereby enhancing the algorithm's performance.

In Task Scenario 4, the tasks for the UAV formation are more complex compared to those in Scenario 3. The MC architecture designed in this paper only extends the Critic module of the gravitational field model. As a result, the MLP module in the MCDDPG algorithm still needs to explore target allocation strategies. This leads to a significant decrease in the task success rate curve during the mid-training phase, as the number of UAVs allocated to one of the targets is insufficient to complete the collaborative task due to the underdeveloped allocation strategy.

From the training curves in Fig.15(c) and Fig.15(d), it can also be observed that as β increases, providing more weight to the MLP module results in better convergence of the rewards and success rates obtained by the UAVs. When $\beta < 0.7$, the guidance provided by the MLP to the Actor module is insufficient, which hinders the MC module's ability to instruct the Actor module in learning allocation strategies, making convergence difficult. The convergence curves in Scenario 4 indicate that even in more complex task scenarios, the DDP-optimized MCDDPG algorithm achieves higher success rates and reward values compared to the fixed β parameter MCDDPG algorithm, with smaller fluctuations and greater stability during training.

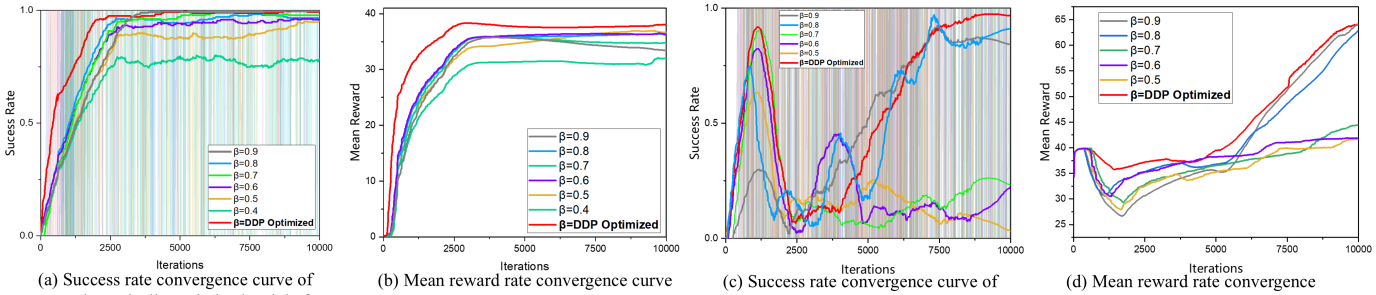
F. Comparison of other MADRL algorithms

This section presents comparative experiments among various MARL algorithms. These experiments are designed to test cooperative tasks that range from single-target to multi-target and single-team to multi-team scenarios. The objective is to validate that the MCDDPG algorithm demonstrates superior convergence speed and task completion capability compared to other MARL algorithms. This superiority is particularly evident in task scenarios 3, 4, and 5, which involve 3 vs 1, 6 vs 2, and 9 vs 3 formations, respectively. All these scenarios aim to achieve multi-angle coordination for UAV swarms towards dynamic targets.

In each of these scenarios, the physical properties and reward functions remain consistent across all algorithms to ensure fair performance evaluations. Various deep reinforcement learning algorithms, including MCDDPG, are trained to develop cooperative control strategies for UAV swarms under identical conditions. These conditions include the same number of training episodes, reward functions, and physical parameter models.

Performance testing reveals that under these standardized conditions, MCDDPG consistently outperforms other algorithms in the domain of multi-angle cooperative planning. Detailed simulation procedures and results are further elaborated in the subsequent sections.

Firstly, the multi-angle information gathering task is conducted under Scenario 3. In this scenario, the UAV swarm is required to track moving targets and form multi-angle coordinated formations when targets reach designated positions, all while ensuring collision-free operation among the UAVs. Six deep reinforcement learning algorithms, including MCDDPG, MAPPO, MADDPG, MASAC, IPPO [32], and ISAC [33], are



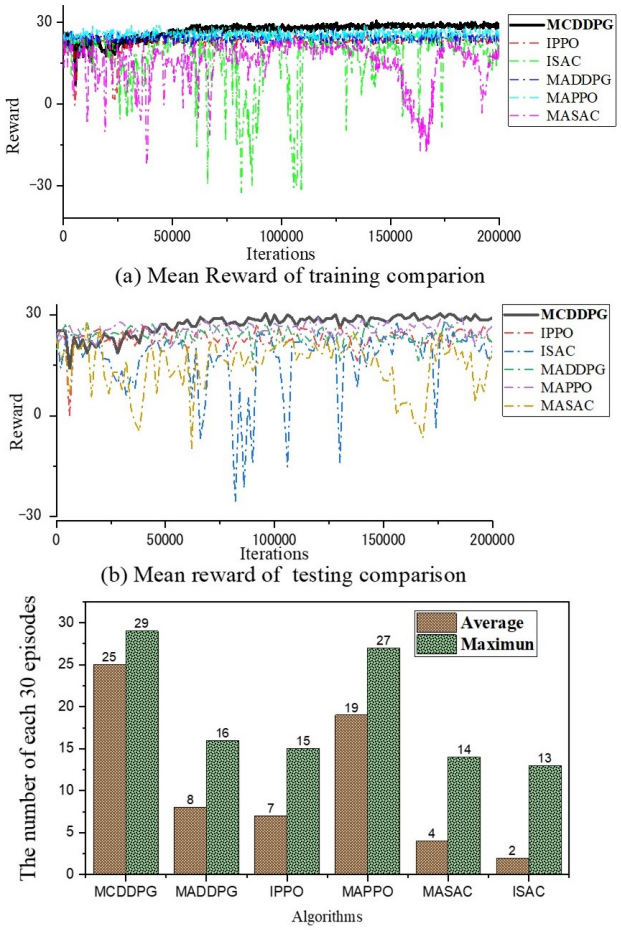
(a) Success rate convergence curve of DDP dynamically optimized weight β and fixed weight β in scenario 3

(b) Mean reward rate convergence curve of DDP dynamically optimized weight β and fixed weight β in scenario 3

(c) Success rate convergence curve of DDP dynamically optimized weight β and fixed weight β in scenario 4

(d) Mean reward rate convergence curve of DDP dynamically optimized weight β and fixed weight β in scenario 4

Fig. 15. In Scenario 3, the task success rate convergence curve with optimized β parameters and fixed β parameters is shown in Figure 13(a), and the reward function convergence curve is shown in Figure 13(b). Similarly, in Scenario 4, the task success rate convergence curve with optimized β parameters and fixed β parameters is shown in Figure 13(c), and the reward function convergence curve is shown in Figure 13(d).



(a) Mean Reward of training comparion

(b) Mean reward of testing comparison

(c) Number of tasks completed by different algorithms in each 30 episodes

Fig. 16. Comparison between MCDDPG algorithm and other MADRL algorithms.

trained. Their training curves are depicted in Fig.16(a), while their test curves are shown in Fig.16(b). This confirms that the proposed algorithm successfully handles a series of coupled tasks such as tracking, obstacle avoidance, and multi-angle coordination.

In the scenario 3, the training learning curves shown in Fig.16(a) and Fig.16(b) do not clearly demonstrate the advantages and disadvantages among MCDDPG, MAPPO, and MADDPG. In scenarios with few targets and a small number of collaborative UAVs, all three algorithms can achieve relatively high rewards. Although the average success rate statistics in Fig.16(c) indicate that the MCDDPG algorithm exhibits a more stable task completion rate, relying solely on this metric may not provide conclusive evidence. Therefore, we further compare and validate the MCDDPG, MAPPO, and MADDPG algorithms using more complex collaborative scenarios.

In both scenario 4 and scenario 5, the UAVs performs the same task as in Scenario 3 and shares the same reward function. The simulation results of the MCDDPG algorithm for these scenarios are depicted in Fig.17(a) and Fig.17(b), while the learning curves from the training of different algorithms are shown in Fig.18(a) and Fig.18(c), the task success rate convergence curves are shown in Fig.18(b) and Fig.18(d).

From Fig.18, it can be observed that as the number of UAVs in the fleet and the number of target points to be planned increase, the MCDDPG algorithm shows improvements compared to the other two algorithms in terms of convergence speed, robustness, and the acquisition of reward values. Moreover, in terms of task completion success rate, MCDDPG achieves the highest success rate among the selected deep reinforcement learning algorithms, as shown in Table V.

TABLE V
COMPARISON OF DIFFERENT ALGORITHMS BASED ON THEIR SUCCESS RATE AND AVERAGE STEP COST ACROSS THREE SCENARIOS.

Algorithms	Metric	scenario 3	Scenario 4	Scenario 5
ISAC	Success rate	0.43	*None*	*None*
ISAC	Average step cost	172.58	*None*	*None*
MASAC	Success rate	0.47	*None*	*None*
MASAC	Average step cost	166.29	*None*	*None*
IPPO	Success rate	0.50	*None*	*None*
IPPO	Average step cost	106.23	*None*	*None*
MADDPG	Success rate	0.53	0.43	0.26
MADDPG	Average step cost	92.79	101.15	14813
MAPPO	Success rate	0.90	0.75	0.56
MAPPO	Average step cost	70.42	89.56	96.77
MCDPPG	**Success rate**	**0.97**	**0.86**	**0.73**
MCDPPG	**Average step cost**	**56.30**	**62.22**	**75.85**

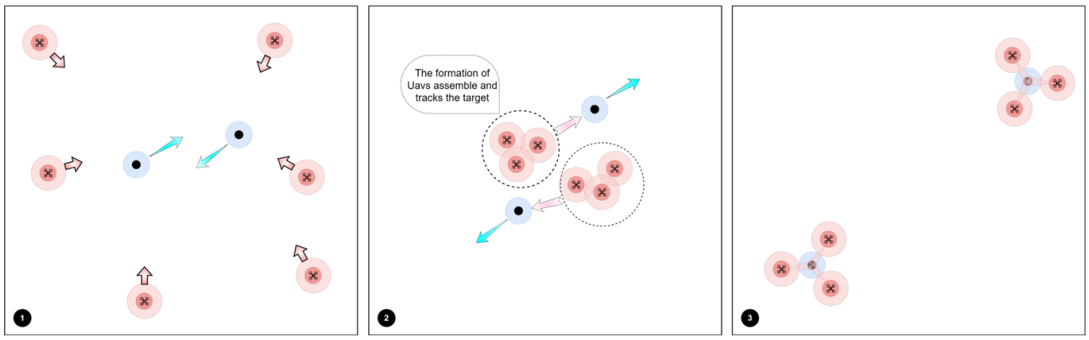
The MADDPG algorithm employs deterministic policy gradient methods, where the Actor generates specific actions based on the state during parameter iteration updates. This characteristic leads to unstable fluctuations during training, resulting in increased training volumes and computational complexity. As depicted in Fig.18, the training results of various MADDPG algorithms show the most noticeable variability in success rates after convergence.

In contrast, MAPPO utilizes actor networks that output action probability distributions rather than deterministic actions. It leverages the PPO algorithm to explore the environment with stochastic policies. The Critic network evaluates the state value function, predicting the expected return the agent can achieve in specific states. The probabilistic nature of MAPPO's actor outputs introduces uncertainty during exploration. Additionally, the computational overhead associated with computing probability distributions contributes to higher training costs. As shown in Fig.18, MAPPO exhibits greater stability and smaller success rate fluctuations after convergence compared to MADDPG, albeit requiring more training episodes during initial exploration.

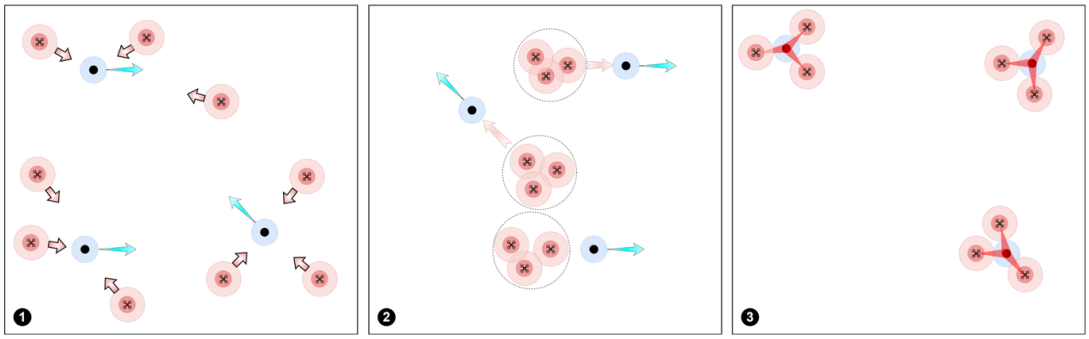
The MCDDPG algorithm proposed in this paper addresses the challenges of lengthy initial exploration periods and post-convergence fluctuations through its MC architecture. This enhances the algorithm's convergence and robustness. Consequently, the convergence speed and stability of reward values and success rates in MCDDPG surpass those of both MADDPG and MAPPO algorithms.

V. CONCLUSION

The proposed MCDDPG algorithm, based on the MC architecture, not only offers an efficient solution for multi-angle coordination tasks in UAV swarm formations, but also has applicability in various other multi-agent cooperative planning scenarios. By integrating model-based Critic modules and

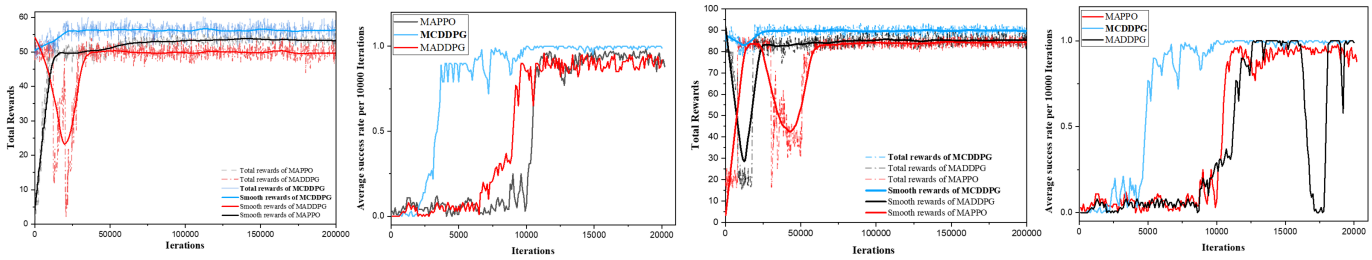


(a) Scenario 4 : 2 targets are acquired by the 6 UAVs



(b) Scenario 5 : 3 targets are acquired by the 9 UAVs

Fig. 17. Collaborative planning simulation effect of multi-UAV formation to obtain multi-target points.



(a) Training learning curves of different algorithms in Scenario 4

(b) Task completion rate curves of each algorithm under Scenario 4

(c) Training learning curves of different algorithms in Scenario 5

(d) Task completion rate curves of each algorithm under Scenario 5

Fig. 18. Comparison of training learning curves of MCDDPG, MADDPG and MAPPO algorithms.

DNN-based Critic modules within the MC architecture, it enhances the adaptability and coordination efficiency of multi-agent systems in complex dynamic environments. This approach introduces novel perspectives for addressing decision-making and coordination challenges in multi-agent systems.

Future research directions could include integrating the MC architecture with other advanced reinforcement learning techniques or optimization algorithms to further enhance the performance of UAV swarms in extremely complex environments, so that multiple UAV formations can have a more intelligent cooperative action plan, and better complete the more complex task scene of multiple target tracking and image collection.

REFERENCES

[1] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[2] G. Sun, R. Zhou, Z. Ma *et al.*, "Mean-shift exploration in shape assembly of robot swarms," *Nature Communications*, vol. 14, no. 3476, 2023.

[3] Z. Deng, J. Xu, Q. Song, B. Hu, T. Wu, and P. Huang, "Robustness of multi-agent formation based on natural connectivity," *Applied Mathematics and Computation*, vol. 366, p. 124636, 2020.

[4] M. Shafiq, Z. A. Ali, A. Israr, E. H. Alkhammash, and M. Hadjouni, "A multi-colony social learning approach for the self-organization of a swarm of uavs," *Drones*, vol. 6, no. 5, 2022. [Online]. Available: https://www.mdpi.com/2504-446X/6/5/104

[5] J. Long, S. Wu, X. Han, Y. Wang, and L. Liu, "Autonomous task planning method for multi-satellite system based on a hybrid genetic algorithm," *Aerospace*, vol. 10, no. 1, 2023.

[6] V. Mnih, K. Kavukcuoglu, D. Silver *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[7] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," 2015. [Online]. Available: https://arxiv.org/abs/1509.06461

[8] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," 2017. [Online]. Available: https://arxiv.org/abs/1502.05477

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[10] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016. [Online]. Available: https://arxiv.org/abs/1602.01783

[11] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019. [Online]. Available: https://arxiv.org/abs/1509.02971

[12] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative, multi-agent games," 2022. [Online]. Available: https://arxiv.org/abs/2103.01955

[13] Y. Pu, S. Wang, R. Yang, X. Yao, and B. Li, "Decomposed soft actor-critic method for cooperative multi-agent reinforcement learning," 2021. [Online]. Available: https://arxiv.org/abs/2104.06655

[14] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," 2020. [Online]. Available: https://arxiv.org/abs/1706.02275

[15] L. Li, R. Zhu, S. Wu, W. Ding, M. Xu, and J. Lu, "Adaptive multi-agent deep mixed reinforcement learning for traffic light control," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 2, pp. 1803–1816, 2024.

[16] X. Liu, C. Xu, H. Yu *et al.*, "Multi-agent deep reinforcement learning for end-edge orchestrated resource allocation in industrial wireless networks," *Front. Inform. Technol. & Electron. Eng.*, vol. 23, pp. 47–60, 2022.

[17] O. Vinyals, I. Babuschkin, W. M. Czarnecki *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, pp. 350–354, 2019.

[18] M. Jiang, T. Hai, Z. Pan, H. Wang, Y. Jia, and C. Deng, "Multi-agent deep reinforcement learning for multi-object tracker," *IEEE Access*, vol. 7, pp. 32 400–32 407, 2019.

[19] Y. Guan, S. Zou, H. Peng, W. Ni, Y. Sun, and H. Gao, "Cooperative uav trajectory design for disaster area emergency communications: A multiagent ppo method," *IEEE Internet of Things Journal*, vol. 11, no. 5, pp. 8848–8859, 2024.

[20] J. Wu, D. Li, Y. Yu, L. Gao, J. Wu, and G. Han, "An attention mechanism and adaptive accuracy triple-dependent maddpg formation control method for hybrid uavs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 9, pp. 11 648–11 663, 2024.

[21] A. Garg and S. S. Jha, "Deep deterministic policy gradient based multi-uav control for moving convoy tracking," *Engineering Applications of Artificial Intelligence*, vol. 126, no. PD, Feb. 2023.

[22] 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 Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 2, pp. 2086–2100, 2024.

[23] Y. Cao, Y. Luo, H. Yang, and C. Luo, "Uav-based emergency communications: An iterative two-stage multiagent soft actor–critic approach for optimal association and dynamic deployment," *IEEE Internet of Things Journal*, vol. 11, no. 16, pp. 26 610–26 622, 2024.

[24] Q. Cui, X. Zhao, W. Ni, Z. Hu, X. Tao, and P. Zhang, "Multi-agent deep reinforcement learning-based interdependent computing for mobile edge computing-assisted robot teams," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 5, pp. 6599–6610, 2023.

[25] W. Wu, J. Xu, and Y. Sun, "Integrate assignment of multiple heterogeneous unmanned aerial vehicles performing dynamic disaster inspection and validation task with dubins path," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 4, pp. 4018–4032, 2023.

[26] C. Dai, K. Zhu, and E. Hossain, "Multi-agent deep reinforcement learning for joint decoupled user association and trajectory design in full-duplex multi-uav networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 6056–6070, 2023.

[27] N. Wen, Y. Long, R. Zhang, G. Liu, W. Wan, and D. Jiao, "Colregs-based path planning for usvs using the deep reinforcement learning strategy," *Journal of Marine Science and Engineering*, vol. 11, no. 12, 2023.

[28] H. Kang, X. Chang, J. Mišić, V. B. Mišić, J. Fan, and Y. Liu, "Cooperative uav resource allocation and task offloading in hierarchical aerial computing systems: A mappo-based approach," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 497–10 509, 2023.

[29] S. Zhou, W. Ren, X. Ren, Y. Wang, and X. Yi, "Independent deep deterministic policy gradient reinforcement learning in cooperative multiagent pursuit games," in *Artificial Neural Networks and Machine Learning – ICANN 2021*, I. Farkaš, P. Masulli, S. Otte, and S. Wermter, Eds. Cham: Springer International Publishing, 2021, pp. 625–637.

[30] Z. Feng, M. Huang, D. Wu, E. Q. Wu, and C. Yuen, "Multi-agent reinforcement learning with policy clipping and average evaluation for uav-assisted communication markov game," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 14 281–14 293, 2023.

[31] D. Xue, D. Wu, A. S. Yamashita, and Z. Li, "Proximal policy optimization with reciprocal velocity obstacle based collision avoidance path planning for multi-unmanned surface vehicles," *Ocean Engineering*, vol. 273, p. 114005, 2023.

[32] Y. Li, X. Li, X. Wei, and H. Wang, "Sim-real joint experimental verification for an unmanned surface vehicle formation strategy based on multi-agent deterministic policy gradient and line of sight guidance," *Ocean Engineering*, vol. 270, p. 113661, 2023.

[33] L. Yue, R. Yang, J. Zuo, Y. Zhang, Q. Li, and Y. Zhang, "Unmanned aerial vehicle swarm cooperative decision-making for sead mission: A hierarchical multiagent reinforcement learning approach," *IEEE Access*, vol. 10, pp. 92 177–92 191, 2022.

[34] J. Xiao, Y. Wu, Y. Chen, S. Wang, Z. Wang, and J. Ma, "Lstfe-net: Long short-term feature enhancement network for video small object detection," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 14 613–14 622.

[35] J. Xiao, H. Guo, J. Zhou, T. Zhao, Q. Yu, Y. Chen, and Z. Wang, "Tiny object detection with context enhancement and feature purification," *Expert Systems with Applications*, vol. 211, p. 118665, 2023.

[36] Y. Jin, R. He, B. Ai, T. Wu, H. Zhang, B. Liu, Y. Li, and J. Li, "A novel geometry-based stochastic channel model in integrated sensing and communication scenarios," *IEEE Wireless Communications Letters*, vol. 13, no. 7, pp. 2018–2022, 2024.

[37] Q. Zhang, X. Wang, Z. Li, and Z. Wei, "Design and performance evaluation of joint sensing and communication integrated system for 5g mmwave enabled cavs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 6, pp. 1500–1514, 2021.

[38] R. Ahmed, Y. Chen, B. Hassan, L. Du, T. Hassan, and J. Dias, "Hybrid machine-learning-based spectrum sensing and allocation with adaptive congestion-aware modeling in cr-assisted iov networks," *IEEE Internet of Things Journal*, vol. 9, no. 24, pp. 25 100–25 116, 2022.

[39] N. Gupta, S. Agarwal, and D. Mishra, "Joint trajectory and velocity-time optimization for throughput maximization in energy-constrained uav," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 24 516–24 528, 2022.

[40] X. Ji and T. Wang, "Energy minimization for fixed-wing uav assisted full-duplex relaying with bank angle constraint," *IEEE Wireless Communications Letters*, vol. 12, no. 7, pp. 1199–1203, 2023.

[41] Z. Hu and X. Jin, "Formation control for an uav team with environment-aware dynamic constraints," *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 1465–1480, 2024.

[42] A. Chowdhury and D. De, "Rgso-uav: Reverse glowworm swarm optimization inspired uav path-planning in a 3d dynamic environment," *Ad Hoc Networks*, vol. 140, p. 103068, 2023.

[43] X. Cheng, H. Zhang, J. Zhang, S. Gao, S. Li, Z. Huang, L. Bai, Z. Yang, X. Zheng, and L. Yang, "Intelligent multi-modal sensing-communication integration: Synesthesia of machines," *IEEE Communications Surveys and Tutorials*, vol. 26, no. 1, pp. 258–301, 2024.

[44] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Springer International Publishing, 2018, pp. 282–299.

[45] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," 2018. [Online]. Available: https://arxiv.org/abs/1711.02257

[46] C. S. de Witt, T. Gupta, D. Makoviichuk, V. Makoviychuk, P. H. S. Torr, M. Sun, and S. Whiteson, "Is independent learning all you need in the starcraft multi-agent challenge?" 2020. [Online]. Available: https://arxiv.org/abs/2011.09533

[47] C. Banerjee, Z. Chen, and N. Noman, "Improved soft actor-critic: Mixing prioritized off-policy samples with on-policy experiences," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 3, pp. 3121–3129, 2024.

Jinsheng Xiao (Senior member, IEEE) received the Ph.D. degree in computational mathematics from Wuhan University, China in 2001. From 2014 to 2015, he was a Visiting Scholar with the University of California at Santa Barbara, USA. He is currently a Professor in Wuhan University. He has authored over 50 papers in international conference proceedings and journals. His works focus on image and video processing, computer vision, and Artificial intelligence.

Bolun Yan received the M.sc degree in Electrical Engineering from Hubei University of Technology , Wuhan, China, in 2023. He is currently pursuing Ph.D degree in electronic information at Wuhan University . His current work primarily focuses on deep reinforcement learning algorithms, cooperative planning of unmanned equipment formation, and Multi-agents control combined with computer visual.

Honggang Xie received the Ph.D. degree in Information and Communication Engineering from Wuhan University, Wuhan, China, in 2011. He is currently working at the School of Electrical and Electronic Engineering, Hubei University of Technology. His research interests include computer vision and Multimedia network communication.

Qiuze Yu received the Ph.D. degree in electronic engineering from Huazhong University of Science and Technology, China in 2004. From 2008 to 2012, he served as a Postdoctoral Fellow and an Associate Professor with Shanghai Jiao Tong University. In 2013, he worked as a Visiting Scholar with the California Institute of Technology, USA. He is currently a Professor in Wuhan University. He has authored over 40 papers in international conference proceedings and journals. His research interests include object recognition and UAV navigation.

Linkun Li received the bachelor's degree in communication engineering from Wuhan University of Technology, China, in 2023. He is currently pursuing M.sc degree in electronic information at Wuhan University . His current work primarily focuses on deep reinforcement learning algorithms, Multi-agent control and collaborative planning.

Yuan-Fang Wang received his Ph.D. degree in electrical and computer engineering from the University of Texas at Austin. He joined the Department of Computer Science at the University of California at Santa Barbara in 1987, where he is currently a Professor. His research interests include Computer Vision, Machine Learning, Computer Graphics, and Robotics.