Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.063703

ARTICLE

An Improved Multi-Actor Hybrid Attention Critic Algorithm for Cooperative Navigation in Urban Low-Altitude Logistics Environments

Chao Li[1,3,#], **Quanzhi Feng**[1,3,#], **Caichang Ding**[2,*] **and Zhiwei Ye**[1,3]

[1]School of Computer Science, Hubei University of Technology, Wuhan, 430068, China

[2]School of Computer and Information Science, Hubei Engineering University, Xiaogan, 432000, China

[3]Hubei Provincial Key Laboratory of Green Intelligent Computing Power Network, School of Computer Science, Hubei University of Technology, Wuhan, 430068, China

*Corresponding Author: Caichang Ding. Email: ccding_hb@hbeu.edu.cn

[#]These authors contributed equally to this work

Received: 21 January 2025; Accepted: 22 May 2025; Published: 03 July 2025

ABSTRACT: The increasing adoption of unmanned aerial vehicles (UAVs) in urban low-altitude logistics systems, particularly for time-sensitive applications like parcel delivery and supply distribution, necessitates sophisticated coordination mechanisms to optimize operational efficiency. However, the limited capability of UAVs to extract state-action information in complex environments poses significant challenges to achieving effective cooperation in dynamic and uncertain scenarios. To address this, we presents an Improved Multi-Agent Hybrid Attention Critic (IMAHAC) framework that advances multi-agent deep reinforcement learning (MADRL) through two key innovations. Firstly, a Temporal Difference Error and Time-based Prioritized Experience Replay (TT-PER) mechanism that dynamically adjusts sample weights based on temporal relevance and prediction error magnitude, effectively reducing the interference from obsolete collaborative experiences while maintaining training stability. Secondly, a hybrid attention mechanism is developed, integrating a sensor fusion layer—which aggregates features from multi-sensor data to enhance decision-making—and a dissimilarity layer that evaluates the similarity between key-value pairs and query values. By combining this hybrid attention mechanism with the Multi-Actor Attention Critic (MAAC) framework, our approach strengthens UAVs' capability to extract critical state-action features in diverse environments. Comprehensive simulations in urban air mobility scenarios demonstrate IMAHAC's superiority over conventional MADRL baselines and MAAC, achieving higher cumulative rewards, fewer collisions, and enhanced cooperative capabilities. This work provides both algorithmic advancements and empirical validation for developing robust autonomous aerial systems in smart city infrastructures.

KEYWORDS: Unmanned aerial vehicles; multiagent deep reinforcement learning; attention mechanism

1 Introduction

The low-altitude economy is an emerging economic model distinguished by technological innovation, operational flexibility, and multi-sector integration, serving as a driving force for future development. Unmanned Aerial Vehicles (UAVs) have been extensively applied in key areas of the low-altitude economy, such as stereoscopic agricultural [1], logistics delivery [2], and activity monitoring [3,4].

In the field of logistics and transportation, the cooperative objective optimization problem for multi-agent systems is both critical and challenging [5–7]. Multi-UAV systems outperform single-UAV systems in time efficiency and collaboration [8]. Wu et al. [9] and Kong et al. [10] applied MADRL to UAV systems in

the logistics and transportation domain, demonstrating high efficiency. MADRL enables agents to interact and collaborate, making it more effective than traditional models for addressing complex multi-agent optimization challenges.

MADRL approaches encounter a non-stationarity problem due to behavior changes caused by agent interactions during training. The Centralized Training with Decentralized Execution (CTDE) paradigm effectively mitigates this issue by utilizing global information during training to improve stability [11]. Under CTDE, decentralized actor networks infer actions based on local observations, while centralized critic networks provide a comprehensive global perspective. Notably, the MAAC model achieves superior performance, particularly in environments with restricted inter-agent communication [12].

However, when applying the aforementioned methods to the UAV logistics models, we observed that relying solely on similar information among UAVs often leads to poor decision-making due to the inability to account for differing information. To address these problems, this paper proposes a novel attention mechanism fused into the MAAC model, aiming to enhance collaboration efficiency among UAVs while reducing system energy consumption. The main contributions of this paper are as follows:

- **Hybrid Attention Mechanism Design:** The proposed IMAHAC algorithm enhances the MAAC framework by adding a sensor fusion layer in the actor network, integrating data from sensors like radar and inertial systems for improved UAV navigation. Additionally, a dissimilarity in the critic network computes state difference weights, enhancing training stability and avoiding errors from over-reliance on similar information.
- **TT-PER Method Application:** We introduce a prioritized experience replay (PER) method and improve it to Temporal Difference Error and time-based prioritized experience replay (TT-PER). This approach minimizes the influence of outdated experiences, enhancing training efficiency and stability.
- **Partially Observable Markov Decision Process (POMDP):** Multi-UAV cooperative tasks demand fast operation, high delivery precision, highly collaborative, and collision-free transportation planning. To tackle these challenges, the problem is modeled as a POMDP. Reinforcement learning enables each UAV to make task decisions based on its observations and state, ensuring efficient collaboration.

2 Related Work

We briefly review here some recent advances most related to our proposed work in the context of UAV-based Multi-Agent Reinforcement Learning (MARL).

2.1 MADRL Applications in UAVs

MADRL is widely applied in scenarios involving multi-UAV systems, such as path planning [13–15], target tracking [16,17], and task allocation [18,19]. Xu et al. [20] proposed an improved version of Multi-Agent Deep Deterministic Policy Gradient (MADDPG) by introducing a novel MARL framework for autonomous cooperative control of UAV swarms. Similarly, Jin et al. [21] introduced the Boids-PE framework, which synergizes Boids swarm intelligence with Deep Reinforcement Learning (DRL) to optimize collision-aware formation control and evasion path planning through Apollonian Circles geometric optimization and self-play training. Li et al. [22] developed a knowledge-assisted multi-agent reinforcement learning method for computation offloading and trajectory planning in multi-UAV mobile edge devices. The primary objectives of their approach were to maximize the success rate of computational task processing and enhance system fairness while minimizing processing delays. However, in dynamic and complex environments, the above algorithms may struggle to adapt to environmental changes in real time.

2.2 Attention Mechanisms in MARL

The attention mechanism is a widely adopted method in machine learning for handling various tasks. It can be divided into categories such as hierarchical attention [23], cooperative attention and multi-head attention [24] based on its structure. Yan et al. [25] introduced hierarchical attention into RL and proposed the HRAMA (Hierarchical Recurrent Attention Multi-Agent Actor-Critic) algorithm. This approach efficiently learns subtask policies at different stages and supports online learning of multi-level strategies. Jiang et al. [26] developed an attention-based communication model that learns when to communicate and how to integrate shared information for cooperative decision-making, achieving both efficiency and effectiveness in multi-agent collaboration. Wang et al. [27] proposed AHAC (Multi-Actor Hierarchical Attention Critic), a method within the CTDE framework. By combining hierarchical and multi-head attention mechanisms, this method enhances information processing and supports decision-makers in achieving better outcomes. Although the multi-head attention mechanism expands MADRL's ability to focus on diverse elements and hierarchically integrate information for improved environmental perception, its reliance on similar information can sometimes cause it to overlook critical differences, leading to suboptimal decisions.

2.3 The Development of UAVs in Logistics

Multi-UAV collaboration has demonstrated significant potential in the fields of logistics and transportation. Jiang et al. [28] designed a deep reinforcement learning algorithm based on graph neural networks, which aggregates the feature vectors of neighboring agents to address the challenges of heterogeneous multi-agent coordination. Liu et al. [29] focused on route planning for UAV logistics and proposed a resource allocation method based on Double Deep Q-Network (DDQN) to maximize the average total capacity of links between UAVs. However, task conflicts during multi-UAV collaboration may arise, potentially impacting overall transportation efficiency.

3 Multi-UAV Cooperative Collaboration Problem

IMAHAC model focuses on integrating the sensor fusion layer and dissimilarity layer into the attention mechanism to effectively address the multi-UAV cooperation challenges in logistics. Multi-UAV cooperative logistics involves large-scale data and demands transportation planning that ensures fast operation, high accuracy, strong collaboration, and collision-free planning. In this paper, the collaborative cooperation of multi-logistics agents is modeled as a POMDP and solved using a policy gradient approach.

3.1 Partially Observable Markov Decision Processes

The POMDP [30] is a classical reinforcement learning model suitable for decision-making in partially observable environments. A POMDP is defined as a tuple $[S, A, \mathrm{T}, P, O, R, \gamma]$, where S represents the state space, defined as the set of states observed by the agents. A is the action space, representing the set of actions that agents execute at each time step. The state transition function $T : S \times A_1 \times \cdots \times A_N \to P$. P represents the state transition probability, defining the probability distribution over the next possible states s' given the current state s and the actions a of the agents. O represents the observations of the environment perceived by the agents. R is the reward function. $\gamma \in [0,1]$ is the discount factor.

3.2 Definition of State, Action and Reward Functions

In this paper, we model the collaboration among agents as a POMDP. The state, action, and reward in the environment constitute the fundamental elements of the POMDP. The state represents the observations received by the UAVs, the action denotes the available movement options, and the reward reflects the feedback or return based on the UAVs' actions.

State Space *S*: The UAVs acquire information from three distinct sensors: ray-cast sensors for collision avoidance, inertial navigation system (INS) sensors for monitoring flight status and self-awareness, and radar (RADAR) sensors for identifying the positions of other UAVs and the distribution centers.

Action Space *A*: The UAVs can perform seven different types of actions: up, down, forward, backward, left, right, and stationary immobility. The stationary state represents either a no-command condition or a collision avoidance mechanism.

Reward Fuction *R*:

Travel Reward r_{travel}: To encourage UAVs to transport cargo along the shortest possible path, a reward mechanism is implemented at each time step. This reward is determined by the difference in distance between the current time step, d_{curr}, and the previous time step, d_{pre}, where each distance represents the UAV's proximity to its target. Before picking up cargo, the target is the nearest package, while after pickup, the target is the delivery destination. If the UAV moves farther from the target during the current time step compared to the previous one, an immediate negative reward is applied. The travel reward function is mathematically expressed as: $r_{driving} = \left(d_{pre} - d_{curr}\right) \times 0.5$.

Cooperative Reward r_{coop}: To train multi-UAV cooperative logistics transportation, the reward mechanism is assigned based on the type of action and the nature of collaboration. Picking up and completing small cargo tasks are non-collaborative actions, where only the UAV responsible for the task receives a reward of +20.0. In contrast, large cargo tasks require collaboration between two UAVs; the first UAV and the second UAV receive rewards of +10.0 and +20.0, respectively, for picking up the cargo, and both earn +30.0 upon completing the transportation. Meanwhile, improper actions lead to penalties—for example, a single UAV dropping large cargo incurs a penalty of −8.0, while both UAVs dropping cargo simultaneously result in a penalty of −15.0 each. Different rewards and penalties are assigned based on various scenarios, aiming to improve the efficiency of UAVs in delivering cargo.

Collision penalty $r_{penalty}$: During training, UAVs utilize ray-casting to detect and avoid collisions with obstacles such as buildings and other UAVs. Any collision results in an immediate penalty, with a negative reward of −10 assigned to discourage such behavior.

Based on the above definition, the final formula for reward is: $r = r_{travel} + r_{coop} + r_{penalty}$.

4 IMAHAC Model

4.1 IMAHAC Structure and Training Algorithm

To maximize the utility of sensor-collected data, mitigate errors caused by the attention mechanism's over-reliance on similarity-based information, and improve the efficiency of information processing during operations, several enhancements were made to the MAAC model. Specifically, a sensor fusion layer was introduced to process each UAV's local observations, enabling more effective integration of sensor data. Additionally, a dissimilarity layer was incorporated to optimize the use of global information shared among UAVs. To further enhance training efficiency, the experience buffer was modified to prioritize sampling based on the Temporal Difference Error (TD-error) of experiences and the time they were stored in the buffer, thereby minimizing the influence of outdated experiences on agent training. The overall architecture of the proposed IMAHAC model is illustrated in Fig. 1.

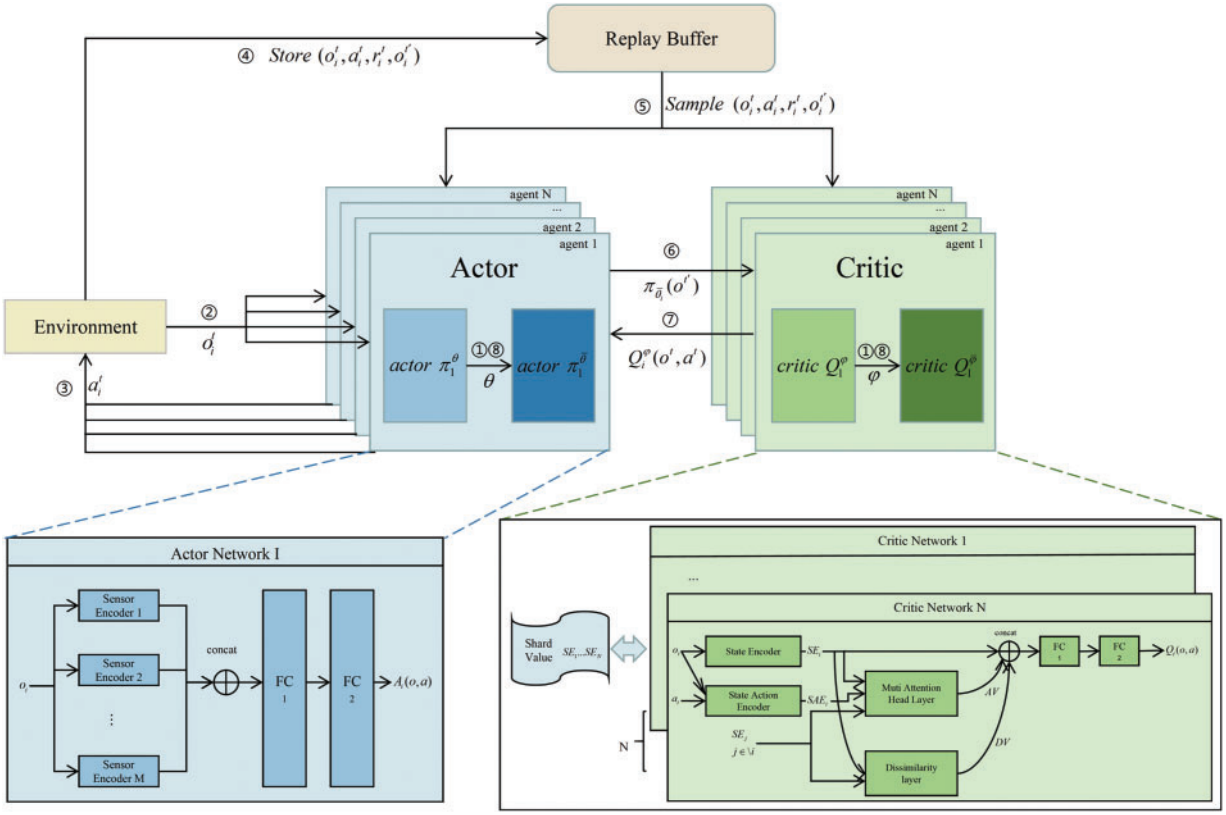

Figure 1: Algorithm structure

The overall workflow of IMAHAC is built upon the foundation of the MAAC model, including the gradient computation of both the loss function and the objective function. Each agent is equipped with its own independent actor and critic networks, operating under a CTDE paradigm. During the training phase, the observations of all agents are used as inputs to the critic network of the corresponding agent. In the execution phase, the decentralized actor network relies solely on the agent's local observations of input data to select actions for inference. The IMAHAC model is applicable to N agents equipped with M types of sensors. The IMAHAC training algorithm is shown in Algorithm 1.

Algorithm 1: IMAHAC training algorithm

Input : Environment state space O, action space A, reward function R
output : Optimized policy π_i^{θ} for each agent i
Initialize : Critic networks Q_i^{ψ} and actor networks π_i^{θ};
Synchronize target networks $Q_i^{\overline{\psi}} \leftarrow Q_i^{\psi}$, $\pi_i^{\overline{\theta}} \leftarrow \pi_i^{\theta}$;
Experience replay buffer B;
Step 1: Collect initial experiences
repeat
for *each time step t* **do**
Observe state o_i^t from the environment;
Compute action $a_i^t = \pi_i^{\theta}(o_i^t)$ using the actor network;

(Continued)

Algorithm 1 (continued)

Execute action a_i^t in the environment;
Receive next state $o_i^{t'}$ and reward r_i^t;
Store transition $(o_i^t, a_i^t, r_i^t, o_i^{t'})$ in the experience buffer B;
end
until *Replay buffer B contains E experiences*
Step 2: Training process
repeat
Step 2.1: Sample from buffer;
Sample a batch of transitions $(o_i^t, a_i^t, r_i^t, o_i^{t'})$ from B;
Step 2.2: Update critic network;
Compute the critic loss:
$L_Q(\psi_i) = \mathbb{E}_{(o_i^t, a_i^t, r_i^t, o_i^{t'})\sim B}\left[\left(Q_i^{\psi}(o_i^t, a_i^t) - y_i\right)^2\right]$
where the target value y_i is defined as:
$y_i = r_i^t + \gamma Q_i^{\overline{\psi}}(o_i^{t'}, \pi_i^{\overline{\theta}}(o_i^{t'}))$
Perform gradient descent on $L_Q(\psi_i)$ to update ψ_i;
Step 2.3: Update actor network;
Compute the actor objective:
$J(\pi_i^{\theta}) = \mathbb{E}_{o_i^t\sim B}\left[\nabla_{\theta_i} \log \pi_i^{\theta}(a_i^t \mid o_i^t) \cdot \left(Q_i^{\psi}(o_i^t, a_i^t) - b(o^t, a_{\setminus i}^t) - \alpha \log(\pi_i^{\theta}(a_i^t \mid o_i^t))\right)\right]$
Perform gradient ascent on $J(\pi_i^{\theta})$ to update θ_i
Step 2.4: Update target networks;
Update the target networks using soft updates:
$\overline{\psi} \leftarrow (1-\tau)\cdot\overline{\psi} + \tau\cdot\psi$
$\overline{\theta} \leftarrow (1-\tau)\cdot\overline{\theta} + \tau\cdot\theta$
until *End of the episode*;

4.2 IMAHAC Actor Networks

Multiple sensors can capture diverse types of information; however, directly inputting this data into the network may result in suboptimal utilization of information and negatively impact decision-making accuracy. The sensor fusion layer enhances feature extraction, enabling UAVs to make more precise navigation decisions in complex and dynamic environments, thereby improving energy efficiency and operational stability during task execution.

As shown in the Fig. 2, a deep fusion layer is incorporated into the actor network of the MAAC algorithm to improve its operational efficiency. Observational data are categorized based on different sensor types, enabling feature extraction tailored to each category. In our virtual environment, three distinct sensor types are utilized for the UAVs: an INS for self-awareness of flight states, a ray-cast sensor for collision avoidance with nearby obstacles, and a RADAR for determining the positions of other UAVs and relay stations.

The data collected from each sensor type are processed through their respective sensor encoders, after which the encoded outputs are concatenated and passed through two fully connected layers. Each sensor's raw data is processed through a dedicated encoder subnetwork to extract task-specific features, with the specific design shown in Table 1.

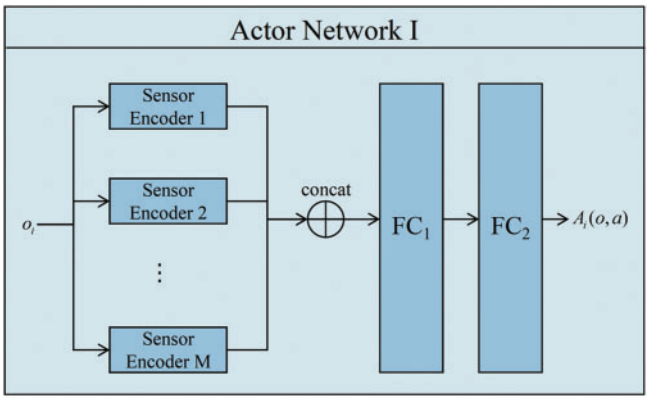

Figure 2: Deep fusion layer in actor networks

Table 1: Sensor observation space

Sensor type	Size	Description
INS	3	(x, y, z)-coordinates of UAVi.
	3	(x, y, z)-velocity of UAVi.
	3	cargo type (not holding, small cargo, and big cargo).
Ray-cast	1 × 9	Distance to obstacles in 9 directions.
	2 × 9	Encoding of the detected object (nothing, building) of 9 direction.
RADAR	6	(x, y, z, x, y, z)-coordinates of a big cargo hub and a small cargo hub.
	2	Distance from UAV to big and small cargo hubs.
	6	(x, y, z, x, y, z)-coordinates of each recently sized cargo.
	2	Distances from UAVi to the nearest big and small cargos.
	4	(x, y, z, d)-if there is a cargo on the UAVi, the coordinates and distance of the destination are given.
	7 × 4	Coordinates of UAVj (size 3), cargo type of UAVj (size 3), and distance from UAVi to UAVj (size 1).

Note: UAVi is the current, and UAVj are the rest of all UAVs except UAVi.

The deep fusion layer processes multimodal sensor inputs (27-dimensional Ray-cast data, 9-dimensional INS data, and 48-dimensional RADAR data) through three parallel sensor-specific encoders. Each encoder employs an independent fully-connected network to extract hierarchical features from its corresponding sensor modality. These modality-specific representations are subsequently concatenated and passed through successive fully-connected layers for cross-modal feature integration and dimensionality reduction, ultimately generating a unified latent representation that encapsulates comprehensive environmental awareness. The mathematical formulation of the deep fusion layer is as follows:

$$Output = FC_2(FC_1(Concat(SNE_1(sensor_1), SNE_2(sensor_2), SNE_3(sensor_3)))) \quad (1)$$

where FC_1, FC_2 and the sensor encoders $(SNE_{1\ldots3})$ represent fully connected layers responsible for feature extraction and integration.

4.3 IMAHAC Critic Networks

The attention mechanism in the MAAC model primarily relies on similarity information [24]. However, in collaborative tasks, information that is dissimilar to the current state can sometimes be more critical. For instance, observations from a UAV that is farther away but closer to the target may be more important than those from a nearby UAV. Introducing a dissimilarity layer helps to mitigate misjudgments by the attention mechanism in specific scenarios, such as distant critical information, similar information with different tasks, and noise interference. This enhances the model's efficiency in utilizing information in complex environments.

In the Critic Network, each agent is assigned an independent critic network. while the state encoder is shared among all agents. As illustrated in the Fig. 3, the observation o_i of agent i is processed by the state encoder to produce a state embedding vector (SE_i), and the action a_i is passed through the state-action encoder to generate the state-action embedding (SAE_i). Meanwhile, the state embeddings (SE_j) from other agents are simultaneously input into both the multi-head attention layer and the dissimilarity layer.

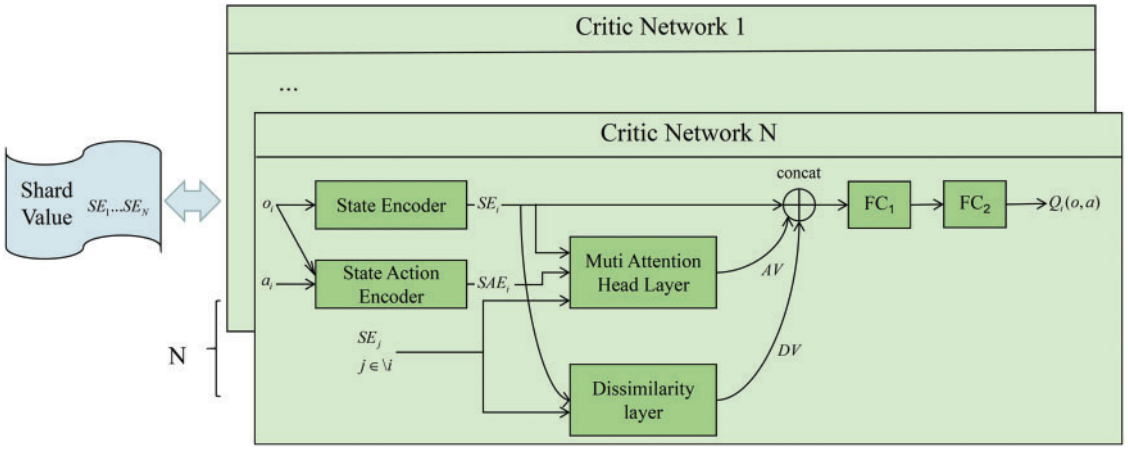

Figure 3: Dissimilarity layer in Critic networks

The multi-head attention layer, constructed based on the scaled dot-product mechanism, calculates the similarity between the encoded observations of agent i and those of agent j to generate the attention values (AV), enabling selective aggregation of relevant information. UAVs within a neighboring distance often exhibit similar observations; by assigning higher weights to these similar inputs, each UAV can obtain a broader field of view, which helps reduce the likelihood of collisions among agents.

The purpose of the multiple attention mechanism is to spread attention in many different directions. However, the information captured in the subspace is uncontrollable and too much concentration can lead to a single message. To overcome these challenges, We introduce a dissimilarity layer in the critic network, which leverages the cosine distance between the outputs of different attention heads as a regularization term. This layer helps to alleviate performance issues caused by excessive focus within the attention mechanism, quantifies the dispersion of attention across subspaces, and improves the overall stability and robustness of the learning process.

The dissimilarity weights between the observation data of agents are derived by scaling the cosine similarity value with a negative scalar, as expressed in Eq. (2):

$$CD(SE_i, SE_n) = -1 \cdot \frac{SE_i \cdot SE_n}{\max(\|SE_i\|_2 \cdot \|SE_n\|_2, \varepsilon)} \tag{2}$$

Negative dissimilarity values are set to zero to emphasize information from agents with distinct observation patterns. Each agent's observation data is then weighted by the cosine dissimilarity values, and the resulting data are concatenated. This concatenated output is fed into a fully connected layer. The concatenated representation of AV, (SE_i), and the dissimilarity values (DV) is then passed through fully connected layers to enhance the critic value estimation.

4.4 Normalization Layer

Although multi-head attention mechanisms have been successfully utilized in reinforcement learning [31], their integration can heavily influence the learning rate during pretraining. If the learning rate is too high, it may cause abrupt gradient fluctuations, resulting in unstable training. Conversely, a low learning rate can hinder progress by slowing down the learning process. To mitigate these challenges, a normalization layer is incorporated between the FC_1 and FC_2 layers in the critic network. This normalization layer improves training stability and facilitates a more efficient learning process.

After calculating the influence of other agents using the DV, a normalization layer is integrated to establish a Pre-Normalization Layer Norm (Pre-LN) paradigm [32]. This approach has been demonstrated to be more effective in mitigating the instability of attention mechanisms, particularly when operating under high learning rates, thereby improving overall training robustness.

4.5 Prioritized Experience Replay

When training the UAV with the IMAHAC algorithm, the gradient is updated incrementally, which can lead to the loss of valuable experience data crucial for future updates and negatively affect training efficiency. To overcome this limitation, the TT-PER method is introduced, prioritizing experience samples based on their TD-error and storage time in the replay buffer. During sampling, the selection probability of each experience is determined by its priority, which not only enhances training efficiency but also minimizes the influence of outdated experiences, thereby optimizing the agent's policy and ensuring greater training stability.

The sampling probability of an experience in this paper is defined as:

$$P(i) = \frac{p_i^\sigma}{\sum_k p_k^\sigma} \tag{3}$$

here, $p_i > 0$ denotes the priority assigned to experience sample i, and the hyperparameter σ controls the degree to which priority influences the sampling process. When $\sigma = 0$, the sampling reduces to a uniform distribution. We set $\sigma = 0.6$ in this paper.

In multi-agent systems, outdated experiences can hinder decision-making and reduce success rates, often showing large TD-error values, assigning high priority to such outdated samples is unreasonable. To address this, the proposed method incorporates a temporal factor based on the time an experience is stored in the replay buffer. Older samples receive lower priorities, while recent ones are prioritized. The priority of an experience sample is defined as:

$$p_i = \delta_i^2 \times \left(1 - e^{-K \times T_i}\right) + \varepsilon \tag{4}$$

Here, e represents the base of the natural logarithm, $\varepsilon = 10^{-4}$ is a small constant to prevent samples with zero TD-error from being entirely ignored, and T_i denotes the time value of experience sample i. The initial value of T is set to 1 and incrementally increases incrementally during training. The parameter K, we set to $K = 0.95$ in this paper, quantifies the influence of time on the priority of experiences. Finally, $\delta_i = \gamma Q_i^{\psi}(o', a') - Q_i^{\psi}(o, a)$ represents the TD-error of the i-th experience sample.

The priority formulation achieves a balanced trade-off between TD-error and temporal relevance, improving both training stability and efficiency.

5 Experiments

We evaluate the performance of the IMAHAC algorithm in a UAV logistic delivery service (UAV LDS) environment built on OpenAI Gym [33] and compare it with other baseline algorithms. The simulations are implemented in Python 3.11.

5.1 UAS-LDS Virtual Environment

The UAV-LDS is a virtual drone logistics and delivery environment that connects ground logistics with aerial drone logistics, facilitating cargo transportation within a three-dimensional urban airspace.

The UAV-LDS simulation environment includes modules for obstacles such as buildings, warehouses, and transportable cargo. In this setup, UAVs are responsible for transporting both large and small cargo between distribution centers and destinations. A key feature of the environment is that large cargo requires the collaboration of two drones to transport, reflecting real-world scenarios where multiple drones must work together to manage heavier loads. This structure allows for the assessment of the effectiveness of drone cooperation in logistics operations.

Fig. 4 depicts the UAV-LDS environment where UAVs are tasked with transporting cargo. In the illustration, gray blocks represent buildings, blue blocks indicate small cargo, and red blocks represent large cargo. The cargo is generated at the blue distribution centers on the ground, with large cargo assigned to pink areas and small cargo to green areas. When a UAV approaches cargo within a specified threshold distance, the cargo attaches to the UAV, simulating real-world logistics processes and streamlining the delivery operations.

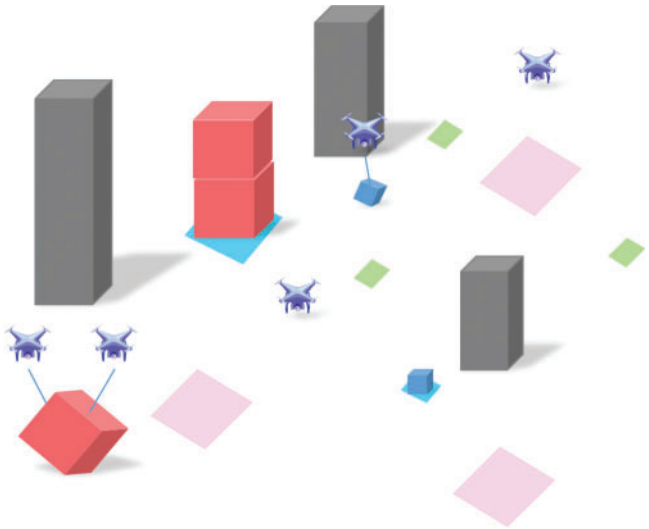

Figure 4: The UAV-LDS environment

5.2 Parameter Setting

The UAS-LDS environment can be customized using the Gym API. In this paper, the experimental parameters used for training and evaluation are outlined in Table 2.

Table 2: Environmental parameter setting

Parameter description	DDPG	MADDPG	MAPPO	MASAC	MAAC	IMAHAC
Total number of UAVs	5	5	5	5	5	5
Number of episodes	1000	1000	1000	1000	1000	1000
Steps per update	250	250	250	250	250	250
Batch size	1024	1024	1024	1024	1024	1024
Buffer length	1e6	1e6	1e6	1e6	1e6	1e6
Number of attention heads	–	–	–	–	4	4
Policy hidden dimension	128	128	128	128	128	128
Learning rate of critic q_lr	0.01	0.01	0.001	0.001	0.001	0.001
Learning rate of policy pi_lr	0.01	0.01	0.001	0.001	0.001	0.001
Discount factor γ	0.99	0.99	0.99	0.99	0.99	0.99
Width of the unity window	480 pixels	480 pixels	480 pixels	480 pixels	480 pixels	480 pixels
Height of the unity window	270 pixels	270 pixels	270 pixels	270 pixels	270 pixels	270 pixels
Number of buildings	3 units	3 units	3 units	3 units	3 units	3 units
MaxSmallbox	100 units	100 units	100 units	100 units	100 units	100 units
MaxBigbox	100 units	100 units	100 units	100 units	100 units	100 units

5.3 Comparison of Baseline Algorithms

The proposed IMAHAC algorithm is compared with the baseline algorithms Deep Deterministic Policy Gradient (DDPG), MADDPG, Multi-Agent Proximal Policy Optimization (MAPPO) and Multi-Agent Soft Actor-Critic (MASAC), using the average episodic reward as the evaluation metric. Fig. 5 illustrates the learning curves of the three algorithms over 20,000 training episodes, where the horizontal axis represents the number of training episodes, and the vertical axis represents the average episodic reward. The curves have been smoothed for clarity.

- **DDPG:** The DDPG algorithm demonstrates slow learning progress, with its average episodic reward showing a slight increase during the early stages (2000 episodes) before plateauing. Eventually, it converges around −30. This indicates that the algorithm exhibits limited learning capability and struggles to adapt to the complex environmental dynamics in the experimental scenario.
- **MADDPG:** Compared to DDPG, MADDPG exhibits better performance, with its average episodic reward rising more rapidly during the early training phase and converging to approximately −20 after around 5000 episodes. Although its performance surpasses that of DDPG, the relatively low convergence value indicates certain limitations when handling multi-agent cooperative tasks.
- **MAPPO:** MAPPO exhibits a rapid performance improvement during the initial training phase; however, however, its mean episode rewards fluctuate around −10 for the remainder of the training process, without significant further improvement. This observation suggests that although MAPPO demonstrates a certain degree of adaptability in the early stages, it encounters limitations in long-term policy optimization and cooperative decision-making in multi-agent environments.

- **MASAC:** The MASAC algorithm demonstrates a stable and continuous performance improvement over the entire training period. Its mean episode rewards increase steadily from the beginning and ultimately reach approximately 60 by the 20,000th episode. Although slightly inferior to the final performance of IMAHAC, MASAC significantly outperforms the other baseline methods, indicating its superior learning efficiency and policy stability in complex multi-agent scenarios.
- **IMAHAC:** The IMAHAC algorithm significantly outperforms both DDPG and MADDPG, with a steep learning curve that improves performance in a robust growth trend. Eventually, it converges to approximately 80 after 20,000 episodes. This result indicate that IMAHAC possesses robust environmental adaptability and multi-agent collaboration capabilities, significantly improving task completion efficiency.

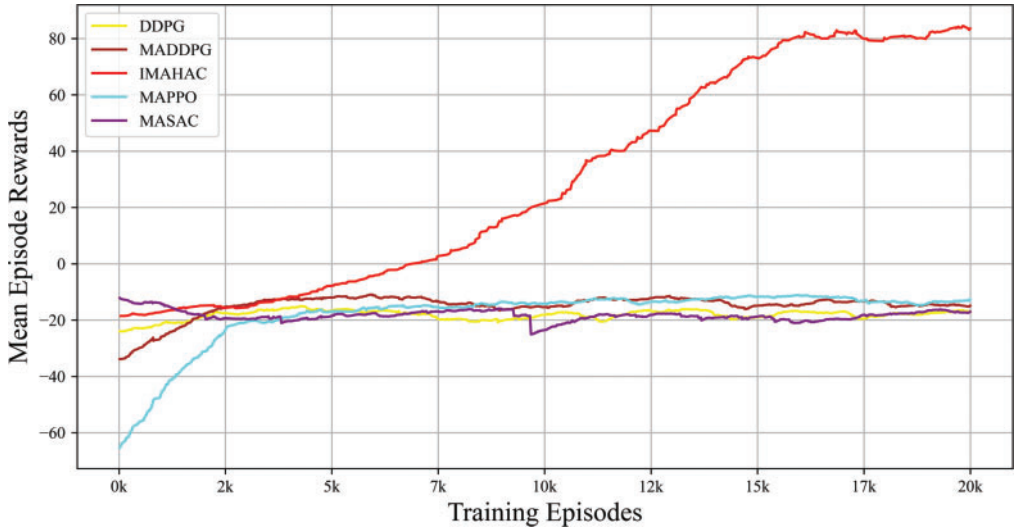

Figure 5: Comparison of baseline algorithms

The superior performance of IMAHAC over other baseline algorithms can be attributed to its architectural enhancements, namely the inclusion of a sensor fusion layer in the actor network and a dissimilarity layer in the critic network. These additions allow the model to better capture complex inter-agent dynamics and environmental features, resulting in improved coordination and learning efficiency. In contrast, algorithms such as DDPG and MADDPG suffer from limited representational capacity and coordination mechanisms, leading to slower convergence and lower overall performance. MAPPO demonstrates moderate early-stage improvements but tends to plateau due to its lack of structured attention or cross-agent representation modeling. MASAC achieves relatively better performance through entropy-regularized training, yet still falls short compared to IMAHAC due to the absence of these critical architectural components.

Table 3 presents the un-smoothed average rewards. As shown in the table, under the same number of drones, IMAHAC outperforms the other four comparison algorithms in terms of rewards.

Table 3: Average rewards

Arithmetic	Number of UAVs	Average rewards
IMAHAC	5	52.87
MASAC	5	21.26
MAPPO	5	−12.84
MADDPG	5	−13.12
DDPG	5	−17.56

5.4 Comparison ith MAAC Algorithm

We further selected the classic multi-agent algorithm MAAC for comparison. Both algorithms were run in the UAS-LDS virtual environment, and a consistent set of training parameters were used for comparison. The results are shown in the Fig. 6.

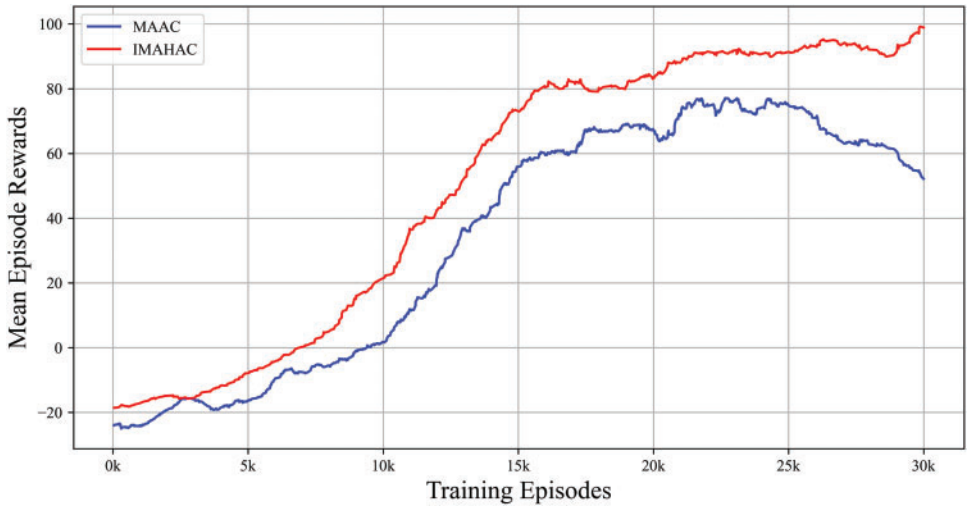

Figure 6: Comparison with MAAC algorithm

The results show that, compared to MAAC, IMAHAC demonstrates a significant performance advantage in the later stages of training. Particularly after 20,000 episodes, its reward value rapidly increases and stabilizes at a high level above 80, while the final reward value of MAAC fluctuates around 60. Additionally, IMAHAC exhibits smaller fluctuations in the later stages, indicating that its policy is more stable and it has a stronger ability to adapt to environmental changes.[1]

To assess the delivery accuracy of different algorithms, we analyzed the number of successful deliveries for both small and large payloads. As illustrated in Fig. 7, the horizontal axis depicts the performance of MAAC and IMAHAC algorithms across different task scenarios, while the vertical axis shows the total number of successfully completed delivery tasks. Additionally, the UAV performance scores were calculated using the following formula:

$$Score = N_{small} + 1.5 * N_{l\,arg\,e} \tag{5}$$

[1]The real-time UAV decision-making comparison videos of MAAC and our method are uploaded on GitHub. Check them out at https://github.com/jidebeibiji/IMAHAC.git (accessed on 21 May 2025).

where N_{small} and N_{large} represent the number of successfully delivered small and large payloads, respectively. The weight of 1.5 for large payloads reflects the 50% higher reward assigned during the training phase.

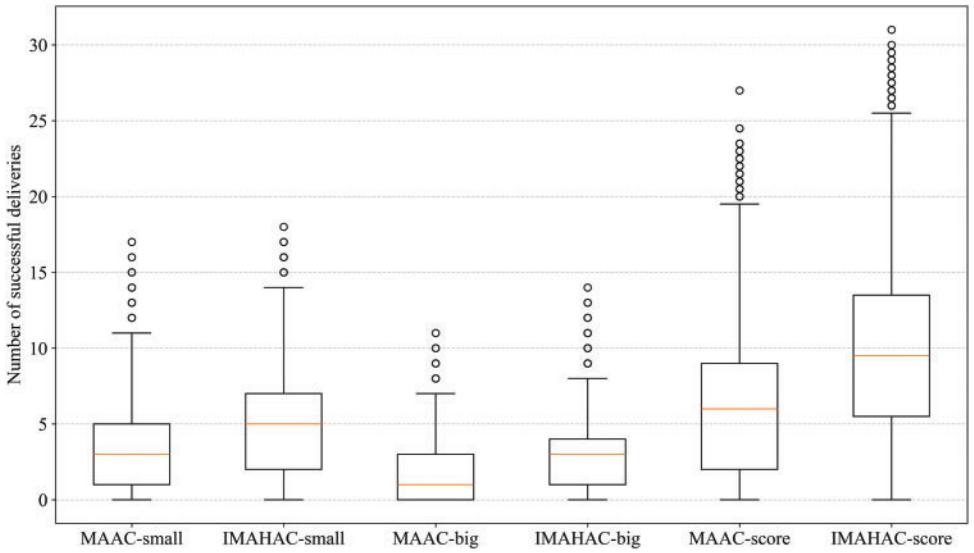

Figure 7: Comparison of delivery performance

Experimental results demonstrate that the IMAHAC algorithm significantly outperforms MAAC across all types of tasks. Notably, in scenarios requiring multi-UAV cooperative transportation, IMAHAC not only increases the median number of successful deliveries but also exhibits a larger upper quartile range. This indicates enhanced cooperative transport capabilities and adaptability. These findings highlight the distinct advantages of IMAHAC in improving UAV collaborative delivery efficiency and its superior adaptability to complex transportation demands.

Finally, we compared the average runtime per episode (in seconds) and the average number of collisions for IMAHAC against other baseline algorithms. The results are presented in the Fig. 8.

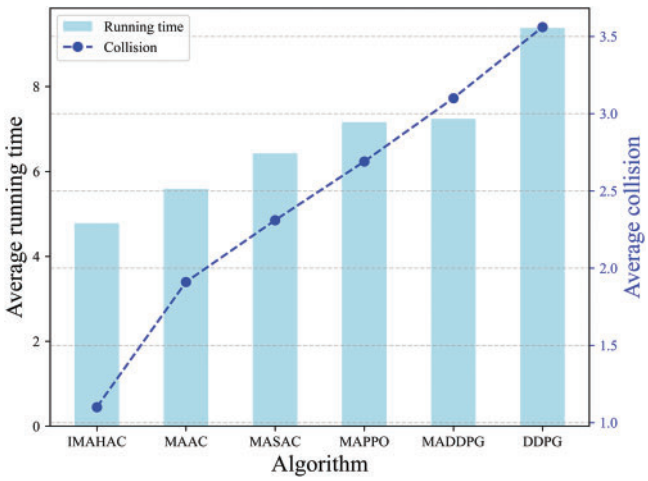

Figure 8: Comparison with running time and number of collisions

The experimental results clearly demonstrate the superior performance of IMAHAC. It achieved the shortest average running time (4.78 s) and the lowest average number of collisions (1.10), indicating both high efficiency and strong safety. MAAC and MASAC followed, with slightly higher runtimes (5.59 and 6.43 s) and moderate collision rates (1.91 and 2.33). In contrast, MADDPG and DDPG showed significantly higher runtimes (7.16 and 9.38 s) and collision counts (3.10 and 3.56), suggesting reduced adaptability and coordination. In general, IMAHAC outperformed all baselines in terms of efficiency and collision avoidance.

6 Discussion and Conclusions

6.1 Discussion

The proposed IMAHAC algorithm demonstrates strong potential in the emerging field of urban low-altitude logistics. In complex urban environments, efficient coordination of UAV-based transportation is crucial, while traditional methods often suffer from poor adaptability and limited scalability.

IMAHAC adopts an attention-based actor-critic architecture integrated with a CTDE strategy, enabling improved performance in task allocation and cooperative control. Moreover, the introduction of a well-designed reward mechanism significantly enhances the obstacle avoidance capability, thereby reducing collision risks in urban low-altitude airspace and improving the overall safety of UAV operations.

Application scenarios for UAV logistics include last-mile delivery, where autonomous parcel distribution can be achieved in congested urban areas, and emergency logistics, such as rapid supply delivery during disasters or critical events. In addition, with the gradual development of urban infrastructure—such as aerial corridors and vertical take-off and landing (VTOL) platforms—IMAHAC is well suited for managing UAV traffic and task coordination in increasingly complex low-altitude airspace.

6.2 Conclusions

To address the issues of low efficiency and poor coordination in multi-UAV transport tasks, we propose an IMAHAC algorithm, developed within a virtual logistics environment based on the OpenAI Gym system. The IMAHAC algorithm integrates a hybrid attention mechanism with the TT-PER prioritized replay mechanism and embeds them into the MAAC framework. The performance of the proposed model is evaluated through the average rewards obtained by multiple UAVs. Experimental results demonstrate that IMAHAC outperforms other baseline algorithms and proves effective in multi-UAV logistics scenarios. The detailed description is as follows:

1. In the proposed model, the sensor fusion layer is incorporated into the actor network, extracting features from various sensors to enable UAVs to efficiently utilize diverse sensor data. Meanwhile, the dissimilarity layer is employed in the critic network, leveraging cosine similarity to provide the target UAV with highly dissimilar data from other UAVs. Training in the UAV-LDS simulation environment demonstrates that IMAHAC outperforms traditional reinforcement learning models in terms of energy efficiency and collaborative capability.
2. The TT-PER prioritized experience replay determines the priority of experiences based on the TD-error and the time they are stored in the experience buffer. This approach helps to prevent outdated experiences from adversely affecting UAV decision-making.
3. Under identical experimental conditions, IMAHAC transported more cargo compared to MAAC, demonstrating superior collaboration capabilities and enhanced obstacle avoidance performance.

Acknowledgement: Not applicable.

Funding Statement: This research was supported by the Hubei Provincial Technology Innovation Special Project and the Natural Science Foundation of Hubei Province under Grants 2023BEB024, 2024AFC066, respectively.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Writing—review & editing: Chao Li; Writing—original draft, Resources, Data curation, Investigation: Quanzhi Feng; Visualization, Supervision, Formal analysis: Caichang Ding; Supervision: Zhiwei Ye. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Liu X, Wang L, Ma Y, Shao P. Collaborative trajectory planning for stereoscopic agricultural multi-UAVs driven by the aquila optimizer. Comput Mater Contin. 2025;82(1):1349–76. doi:10.32604/cmc.2024.058294.
2. Li J, Xiong X, Yan Y, Yang Y. A survey of indoor UAV obstacle avoidance research. IEEE Access. 2023;11:51861–91. doi:10.1109/ACCESS.2023.3262668.
3. Yang Y, Xiong X, Yan Y. UAV formation trajectory planning algorithms: a review. Drones. 2023;7(1):62. doi:10.3390/drones7010062.
4. Noguchi T, Komiya Y. Persistent cooperative monitoring system of disaster areas using UAV networks. In: 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI); 2019 Oct 14–18; San Francisco, CA, USA. Piscataway, NJ, USA: IEEE. p. 1595–600.
5. Li Y, Liu L, Wu J, Wang M, Zhou H, Huang H. Optimal searching time allocation for information collection under cooperative path planning of multiple UAVs. IEEE Trans Emerg Top Comput Intell. 2022;6(5):1030–43. doi:10.1109/TETCI.2021.3107488.
6. Du PF, He X, Cao HT, Garg S, Kaddoum G, Hassan MM. AI-based energy-efficient path planning of multiple logistics UAVs in intelligent transportation systems. Comput Commun. 2023;207(21):46–55. doi:10.1016/j.comcom.2023.04.032.
7. Wang Z, Hu T, Long L. Multi-UAV safe collaborative transportation based on adaptive control barrier function. IEEE Trans Syst Man Cybern Syst. 2023;53(11):6975–83. doi:10.1109/TSMC.2023.3292810.
8. Shen G, Lei L, Zhang X, Li Z, Cai S, Zhang L. Multi-UAV cooperative search based on reinforcement learning with a digital twin driven training framework. IEEE Trans Veh Technol. 2023;72(7):8354–68. doi:10.1109/TVT.2023.3245120.
9. Wu G, Fan M, Shi J, Li Z, Chen X. Reinforcement learning based truck-and-drone coordinated delivery. IEEE Trans Artif Intell. 2021;4(4):754–63. doi:10.1109/TAI.2021.3087666.
10. Kong F, Li J, Jiang B, Liu Y, Sun X, Wang X. Trajectory optimization for drone logistics delivery via attention-based pointer network. IEEE Trans Intell Transp Syst. 2022;24(4):4519–31. doi:10.1109/TITS.2022.3168987.
11. Nekoei H, Badrinaaraayanan A, Sinha A, Li X, Wang Y, Zhang Z et al. Dealing with non-stationarity in decentralized cooperative multi-agent deep reinforcement learning via multi-timescale learning. In: Conference on Lifelong Learning Agents; 2023 Jul 24–28; Honolulu, HI, USA. p. 376–98.
12. Iqbal S, Sha F. Actor-attention-critic for multi-agent reinforcement learning. In: Proceedings of the International Conference on Machine Learning; 2019 Jun 9–15; Long Beach, CA, USA. Piscataway, NJ, USA: IEEE. p. 2961–70.
13. Yang S, Yuan J, Zhang Z, Chen Z, Zhang H, Li W, et al. Multi-uav collaborative mission planning method for self-organized sensor data acquisition. Comput Mater Contin. 2024;81(1):1529–63. doi:10.32604/cmc.2024.055402.

14. Yuksek B, Umut Demirezen M, Inalhan G, Sekercioglu IH, Polat E. Cooperative planning for an unmanned combat aerial vehicle fleet using reinforcement learning. J Aerosp Inf Syst. 2021;18(10):739–50. doi:10.2514/1.I010961.
15. Xu D, Chen G. The research on intelligent cooperative combat of UAV cluster with multi-agent reinforcement learning. Aerosp Syst. 2022;5(1):107–21. doi:10.1007/s42401-021-00105-x.
16. Wang Z, Wang L, Yi Q, Li X, Liu Y. A MARL based multi-target tracking algorithm under jamming against radar. arXiv:2412.12547. 2024.
17. Zhu S, Han G, Lin C, Li Y, Wang X, Zhang Y, et al. Underwater multiple AUV cooperative target tracking based on minimal reward participation-embedded MARL. IEEE Trans Mob Comput. 2024;24(5):4169. doi:10.1109/TMC.2024.3521028.
18. Liu D, Dou L, Zhang R, Li X, Wang Y, Chen G. Multi-agent reinforcement learning-based coordinated dynamic task allocation for heterogenous UAVs. IEEE Trans Veh Technol. 2022;72(4):4372–83. doi:10.1109/TVT.2022.3228198.
19. Zhao G, Wang Y, Mu T, Li X, Liu Y, Zhang Z, et al. Reinforcement learning assisted Multi-UAV task allocation and path planning for IIoT. IEEE Internet Things J. 2024;11(16):26766. doi:10.1109/JIOT.2024.3370152.
20. Xu D, Chen G. Autonomous and cooperative control of UAV cluster with multi-agent reinforcement learning. Aeronaut J. 2022;126(1300):932–51. doi:10.1017/aer.2021.112.
21. Jin W, Tian X, Shi B, Zhao B, Duan H, Li Y, et al. Enhanced UAV pursuit-evasion using boids modelling: a synergistic integration of bird swarm intelligence and DRL. Comput Mater Contin. 2024;80(3):3523–53. doi:10.32604/cmc.2024.055125.
22. Li X, Qin Y, Huo J, Wang Y, Zhang Z, Chen G. Computation offloading and trajectory planning of multi-UAV-enabled MEC: a knowledge-assisted multiagent reinforcement learning approach. IEEE Trans Veh Technol. 2023;73(5):7077. doi:10.1109/TVT.2023.3338612.
23. Seo PH, Lin Z, Cohen S, Li Y, Wang X. Progressive attention networks for visual attribute prediction. arXiv:1606.02393. 2016.
24. Vaswani A, Shazeer NM, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Neural Information Processing Systems; 2017 Dec 4–9; Long Beach, CA, USA. Red Hook, NY, USA: Curran Associates. p. 5998–6008.
25. Yan C, Wang C, Xiang X, Li Y, Wang X, Zhang Z, et al. Collision-avoiding flocking with multiple fixed-wing UAVs in obstacle-cluttered environments: a task-specific curriculum-based MADRL approach. IEEE Trans Neural Netw Learn Syst. 2023;35(8):10894. doi:10.1109/TNNLS.2023.3245124.
26. Jiang J, Lu Z. Learning attentional communication for multi-agent cooperation. Adv Neural Inf Process Syst. 2018;31:947–57.
27. Wang Y, Shi D, Xue C, Li X, Liu Y, Zhang Z. AHAC: actor hierarchical attention critic for multi-agent reinforcement learning. In: Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics; 2020 Oct 11–14; Toronto, ON, Canada. Piscataway, NJ, USA: IEEE. p. 3013–20.
28. Jiang Z, Chen Y, Song G, Li X, Wang Y, Zhang Z. Cooperative planning of multi-UAV logistics delivery by multi-graph reinforcement learning. In: International Conference on Computer Application and Information Security (ICCAIS 2022); 2023 Mar 20–22; Cairo, Egypt. Bellingham, WA, USA: SPIE. p. 129–37.
29. Liu C, Huang L, Dong Z. A two-stage approach of joint route planning and resource allocation for multiple UAVs in unmanned logistics distribution. IEEE Access. 2022;10:113888–901. doi:10.1109/ACCESS.2022.3218134.
30. Shani G, Pineau J, Kaplow R. A survey of point-based POMDP solvers. Auton Agent Multi-Agent Syst. 2013;27(1):1–51. doi:10.1007/s10458-012-9200-2.
31. Chen S, Sheen H, Wang T, Li X, Liu Y, Zhang Z. Training dynamics of multi-head softmax attention for in-context learning: emergence, convergence, and optimality. arXiv:2402.19442. 2024.
32. Huang N, Kümmerle C, Zhang X. UnitNorm: rethinking normalization for transformers in time series. arXiv:2405.15903. 2024.
33. OpenAI Gym [Internet]. San Francisco, CA, USA: OpenAI; 2022 [cited 2025 May 21]. Available from: https://github.com/openai/gym.