

Article

Multi-Agent Deep Reinforcement Learning (MADRL)-Based End-to-End Formation Control for UAV Swarm with Dynamic Topology

Yanping Chen ¹, Qingyang Xu ², Chi Zhang ^{1,*}  and Zhengmao Li ¹

¹ School of Artificial Intelligence and Big Data, Hefei University, Hefei 230601, China; chenypo@hfu.edu.cn (Y.C.); izm@hfu.edu.cn (Z.L.)

² School of Electrical Engineering and Automation, Anhui University, Hefei 230039, China

* Correspondence: hfuu_zc@hfu.edu.cn

Abstract

While MADRL has demonstrated significant potential in Unmanned Aerial Vehicle (UAV) swarm control, traditional architectures often rely on fixed-dimensional observation spaces. This rigid structural constraint severely limits the swarm's adaptability in dynamic environments, particularly when facing sudden topological changes such as node failures or dynamic reinforcements. To overcome these limitations, this paper proposes an end-to-end UAV swarm motion control framework incorporating a state-modulated Graph Attention Network (GAT). By modeling the swarm as a dynamic interaction graph, the proposed method dynamically aggregates neighbor features using attention weights modulated by the agents' real-time kinematic states. Furthermore, a virtual structure combined with an auction mechanism is introduced to achieve precise formation planning and target allocation. Evaluated in the Genesis 3D physics engine, the proposed Prioritized Experience Replay (PER)-MADDPG-Graph Attention Network (GAT) algorithm exhibits superior robustness and spatial adaptability. Extensive experiments, including dynamic node reduction and addition scenarios, confirm that the proposed framework seamlessly maintains swarm configurations without catastrophic policy degradation, outperforming baseline MADRL methods in both convergence speed and control precision.

Keywords: UAV swarm; GAT; MADDPG; end-to-end control

1. Introduction

In recent years, UAV swarms have been increasingly deployed in complex missions [1], such as area surveillance [2,3], search [4] and rescue [5], and distributed sensor networks. Compared to single-agent systems, UAV swarms [6,7] offer improved fault tolerance and broader spatial coverage. However, effective cooperative control remains a significant challenge, as the swarm must simultaneously navigate toward target destinations, avoid collisions, and maintain specific spatial configurations in highly dynamic environments.

To address the complex coordination tasks of UAV swarms, numerous motion control strategies have been proposed in recent literature. Traditional control paradigms, such as robust control [8] based on semidefinite programming, sliding mode control [9], and behavior-based algorithms [10], have established foundational frameworks for maintaining swarm formations. To further refine performance, two notable advances have been proposed. First, integrating Control Barrier Functions ensures safety-critical obstacle avoidance [11]. Second, advanced path-following strategies like decoupled nonuniform



Academic Editor: Milena Faria Pinto

Received: 5 May 2026

Revised: 20 June 2026

Accepted: 25 June 2026

Published: 1 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

guiding vector fields improve coordination accuracy on complex 3D manifolds by decoupling convergence and coordination terms [11]. Furthermore, integrating Control Barrier Functions with disturbance observers ensures rigorous, safety-critical obstacle avoidance and prescribed tracking performance for UAVs [12]. However, these model-based methods typically rely on explicit mathematical models and fixed environmental assumptions, severely limiting their adaptability when confronted with highly dynamic environments or unexpected topological disruptions [13].

Consequently, MADRL has emerged as a practical paradigm for end-to-end autonomous control [14,15]. Algorithms such as the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) allow agents to learn cooperative strategies—such as obstacle avoidance, target encirclement, and trajectory planning—through continuous trial-and-error interactions. By mapping environmental observations directly to continuous control commands, MADRL circumvents the need for explicitly programmed heuristic rules [16,17]. Furthermore, to address the scalability bottlenecks in multi-agent interactions, recent studies have begun integrating Graph Neural Networks with reinforcement learning, treating the swarm as a dynamic graph to efficiently aggregate spatial features among neighbors [18]. Beyond spatial coordination, UAV swarms increasingly function as complex nonlinear networked systems within industrial and surveillance applications. In these settings, addressing subsystem heterogeneity and data privacy under external disturbances has become a critical frontier. This has prompted the development of advanced federated learning and offset-free distributed control frameworks [19].

Despite these significant theoretical advancements, applying conventional MADRL and graph-based extensions to realistic 3D UAV swarms still presents several critical limitations. First, traditional Actor–Critic architectures require fixed-dimensional observation spaces [20], restricting agents to a pre-defined number of neighbors. Although recent hybridized frameworks utilize Graph Neural Networks [21] to handle variable topologies, their attention mechanisms predominantly rely on static spatial proximity. They lack the explicit integration of the UAV's dynamic kinematic intents (i.e., continuous linear and angular velocities). When swarms undergo highly dynamic flight or sudden reconfigurations, this purely distance-based perception causes topological lag, frequently leading to catastrophic inter-agent collisions. Second, existing on-policy MARL algorithms suffer from severe sample inefficiency in complex continuous control tasks. Consequently, many studies validate their algorithms using simplified dataset-driven or 2D point-mass kinematics [22,23], which fail to capture the 3D underactuated and strongly coupled dynamics of actual quadrotors. Finally, most existing MADRL approaches utilize purely reactive reward shaping. While this encourages basic aggregation, it often results in loose, unpredictable swarm morphologies, making it difficult to enforce strict geometric formations required for specific tactical tasks.

To address these limitations, this paper proposes an off-policy, sample-efficient end-to-end 3D UAV swarm motion control framework incorporating a state-modulated GAT [24] and PER [25]. By modeling the swarm as a dynamic interaction graph, the GAT module processes variable-length neighbor information, freeing the network from fixed-topology constraints. Unlike conventional graph-based methods that rely purely on spatial distance, our attention weights are dynamically modulated by the agents' real-time kinematic states. This structural contribution establishes a physical kinematics-driven semantic topology, ensuring rapid proactive collision avoidance even under sudden swarm reconfigurations. Furthermore, the integration of PER significantly mitigates the sample inefficiency bottleneck in 3D environments. To achieve strict geometric control, a virtual structure [26] method combined with a decentralized auction mechanism is introduced for formation planning and target allocation.

The main contributions of this paper are summarized as follows:

- **3D End-to-End MDP Formulation:** We construct a comprehensive Markov Decision Process (MDP) for 3D quadrotor control. A composite reward function is designed, incorporating attitude constraints, velocity penalties, and collision avoidance, bridging the gap between high-level navigation decisions and low-level physical actuation.
- **State-Modulated GAT for Dynamic Topologies:** We introduce a GAT module that dynamically aggregates neighbor features based on kinematic intent vectors. This design inherently accommodates variable neighbor counts, enabling the swarm to seamlessly adapt to sudden topological changes (e.g., node failures or additions) without requiring retraining.
- **Auction-Based Formation Planning:** We integrate a virtual structure approach with a decentralized auction mechanism to translate macroscopic swarm topologies into explicit relative-position tracking tasks, effectively minimizing the total maneuvering cost during formation reconfiguration.
- **3D Physical Validation:** The proposed PER-MADDPG-GAT algorithm is evaluated within the Genesis 3D physics engine. Extensive ablation studies confirm that the framework maintains high control precision and robustness under dynamic node reduction and reinforcement scenarios, significantly outperforming baseline MADRL methods.

The remainder of this paper is organized as follows. Section 2 details the proposed MADDPG-based UAV swarm control algorithm, including the MDP modeling, the state-modulated GAT module, and the formation planning mechanism. Section 3 presents the experimental settings, comparative results, and ablation studies in the 3D simulation environment. Finally, Section 4 concludes the paper.

2. GAT-Based UAV Swarm Control Algorithm

2.1. MDP Modeling for 3D End-to-End Swarm Control

Based on the core principles of reinforcement learning (RL), the highly dynamic 3D swarm coordination problem must be formally formulated into a Markov Decision Process (MDP) model to achieve end-to-end control of the UAV swarm [27]. It should be clarified that in this specific context, “end-to-end” refers to the direct policy mapping from local observation states to continuous kinematic commands, effectively bypassing traditional explicit path planning and trajectory optimization modules. Meanwhile, the underlying attitude and motor executions are still governed by pre-defined low-level controllers for fundamental flight stability. As shown in Figure 1, this section elaborates on the formal definitions from three dimensions: the high-dimensional state space, the low-level action space, and the composite reward function with physical constraints. Note that Figure 1 was conceptually drafted by the authors; AI (gemini-3.1-pro) was employed solely for non-substantive visual polishing—including color palette, shading, and aesthetic refinement—with all elements rigorously verified against the original draft to ensure scientific integrity.

State Space: In a multi-UAV swarm, the state space serves as the sole interface for the neural network to perceive physical world information. The rationality and completeness of its design directly determine the upper bound of algorithm convergence and control stability. Assuming there are N UAVs in the environment, to ensure that the agent grasps precise kinematic boundaries and avoids dangerous maneuvers that violate physical characteristics (such as extreme roll/pitch overloads), a single UAV i must first obtain the observation of its own ego-attitude $\Theta_i = [\phi_i, \theta_i, \psi_i]^T$, representing the roll, pitch, and yaw angles, respectively.

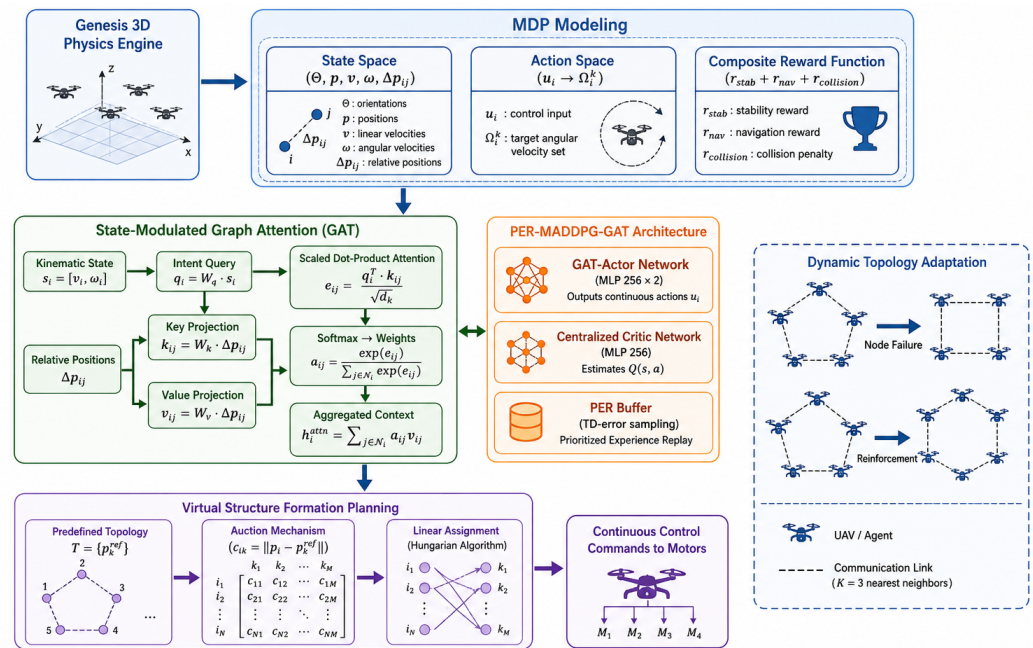


Figure 1. Schematic overview of the proposed GAT-based UAV swarm control framework. The diagram illustrates the end-to-end control pipeline: (1) multi-UAV environment state acquisition from the Genesis physics engine, (2) state-modulated graph attention mechanism for dynamic neighbor feature aggregation, (3) PER-MADDPG-GAT policy network for continuous action generation, and (4) virtual structure formation planning with decentralized auction-based node allocation. AI-assisted tools were used for visual enhancement of this figure.

To depict the absolute motion state of the UAV in 3D space, the network must be fed the UAV's position vector $\mathbf{p}_i = [x_i, y_i, z_i]^T$, linear velocity vector $\mathbf{v}_i = [v_{x_i}, v_{y_i}, v_{z_i}]^T$, and angular velocity vector $\boldsymbol{\omega}_i = [\omega_{x_i}, \omega_{y_i}, \omega_{z_i}]^T$ in the global coordinate system. The aforementioned physical quantities collectively constitute the agent's self-state observation component $\mathbf{o}_i^{\text{self}} = [\boldsymbol{\Theta}_i^T, \mathbf{p}_i^T, \mathbf{v}_i^T, \boldsymbol{\omega}_i^T]^T$.

To accomplish cooperative swarm tasks, relying solely on isolated individual information is far from sufficient to maintain the swarm formation. Adequate relative motion information of neighbors must be provided to enhance the UAV's perception of the swarm's spatial topology. Constrained by the communication range and bandwidth limits of real physical systems, a local topology judgment mechanism is introduced: setting a maximum communication radius d_c , state resolution is only performed for nodes within this range. In addition, the selection of the K nearest neighbors is analytically determined by balancing the geometric requirements of 3D collision avoidance with the communication bandwidth constraints of physical UAVs. From a spatial perspective, tracking the three closest neighbors effectively covers the immediate interaction neighborhood required for reliable collision evasion in a 3D partition. Furthermore, from a topological learning perspective, blindly increasing K not only escalates the communication payload but also introduces “over-smoothing” risks inherent to graph neural networks, which blurs the distinct local kinematic intents of the agents and degrades the responsiveness of the swarm. Under this rule, agent i acquires the relative position vector $\mathbf{p}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ and the relative velocity vector $\mathbf{v}_{ij} = \mathbf{v}_j - \mathbf{v}_i$ of neighbor j . Simultaneously, to guide the swarm in completing navigation tasks, the relative distance vector $\mathbf{p}_{ig} = \mathbf{p}_g - \mathbf{p}_i$ between the agent and the task target point (or the absolute hovering point in the desired swarm configuration) is introduced. The above interaction and target guidance features are combined to form the environmental observation component $\mathbf{o}_i^{\text{env}}$.

To accelerate the convergence of the neural network, all observation vectors undergo scaling and normalization. The final, complete state space of UAV i is defined as $\mathbf{o}_i = [\mathbf{o}_i^{\text{self}}, \mathbf{o}_i^{\text{env}}]^T$.

Action Space: A quadrotor UAV is a highly underactuated and strongly coupled non-linear system. To discard traditional attitude resolution controllers and achieve a direct mapping from network commands to real physical torques, the action space design must balance control precision with dynamic boundaries. The system adopts a continuous action space, where the output of the Actor network is designed as a 4D normalized vector $\mathbf{u}_i = [u_i^1, u_i^2, u_i^3, u_i^4]^T$, with each component $u_i^k \in [-1, 1]$.

In the Genesis engine, directly outputting unconstrained rotational speeds easily leads to rotor stall or overspeed crashes. Assuming the baseline motor rotational speed of the quadrotor in a fully loaded hover state is Ω_{hover} , this paper constructs a relative control mapping mechanism based on this baseline speed. The outputs of the Actor network are used to perform differential adjustments on the low-level propellers:

$$\Omega_i^k = (1 + c \cdot u_i^k) \Omega_{\text{hover}} \quad (1)$$

where c is the physical scaling coefficient for actions, set to $c = 0.8$ in this study. Under this hard constraint, the true physical rotational speed of a single motor is strictly confined within the closed interval $[0.2\Omega_{\text{hover}}, 1.8\Omega_{\text{hover}}]$, ensuring that system control commands always evolve smoothly around the steady-state hovering condition.

Composite Reward Function: In quadrotor UAV swarm control, the attitude stability of the UAV is of paramount importance. During the initial learning phase, a lack of reasonable attitude constraints can cause the UAV to tumble or lose control, violating basic physics. Therefore, attitude stability terms are designed to guide the UAV toward smooth and controllable flight. First, excessive roll and pitch angles are penalized to suppress severe attitude inclination:

$$r_i^{\text{att}} = -\lambda_{\text{att}}(|\phi_i| + |\theta_i|) \quad (2)$$

where $\lambda_{\text{att}} > 0$ is the weight coefficient that adjusts the impact of this term within the overall reward space, ensuring the UAV maintains a level attitude during flight and improving stability.

In addition to attitude angle deviations, excessive angular velocity also leads to flight instability. Particularly during swarm flight and cooperative maneuvers, violent attitude changes can disrupt the formation structure and increase the risk of collision. Thus, a penalty term based on the magnitude of the angular velocity is introduced to suppress rapid spinning behaviors:

$$r_i^{\text{ang}} = -\lambda_{\omega} |\omega_i| \quad (3)$$

where $\lambda_{\omega} > 0$ is the angular velocity penalty weight. Restricting angular velocity effectively reduces attitude oscillations, guiding the UAV toward smoother flight.

During swarm flight tasks, the yaw stability of the UAV is crucial for maintaining heading and coordinating maneuvers. Therefore, an exponential reward function based on yaw deviation is introduced. Let ψ_i be the yaw angle of the i -th UAV. After normalizing it to $(-\pi, \pi]$, the yaw stability reward is defined as:

$$r_i^{\text{yaw}} = \exp(-\lambda_{\psi} |\psi_i|) \quad (4)$$

where $\lambda_{\psi} > 0$ is the decay coefficient. This reward assumes a larger value when the yaw angle aligns with the target direction and decays rapidly as the deviation increases, guiding

the UAV to maintain a stable heading. Combining these three terms, the attitude stability reward function formulated in this paper is expressed as:

$$r_i^{\text{stab}} = r_i^{\text{att}} + r_i^{\text{ang}} + r_i^{\text{yaw}} \quad (5)$$

For UAVs to form a cohesive swarm, stable flight attitude is the fundamental prerequisite. During the pre-defined swarm formation process, each UAV must gradually navigate to its designated position within the formation. Therefore, a reward function is designed based on the relative distance between the UAV's position and the target point:

$$r_i^{\text{step}} = \lambda_{\text{step}} \left(\left| \mathbf{p}_i^t - \mathbf{p}_g^t \right| - \left| \mathbf{p}_i^{t+1} - \mathbf{p}_g^{t+1} \right| \right) \quad (6)$$

where $\lambda_{\text{step}} > 0$. Using the difference between consecutive frames serves as a dense feedback signal, allowing the UAV to explicitly perceive whether each step brings it closer to the target. Compared to a sparse reward granted only upon reaching the destination, this approach provides stronger guidance. However, while this design encourages general movement toward the target, achieving precise arrival and steady hovering requires an additional, more direct penalty:

$$r_i^{\text{goal}} = -\lambda_{\text{goal}} \left| \mathbf{p}_i - \mathbf{p}_g \right| \quad (7)$$

This term penalizes the absolute distance between the UAV and the target, reaching its maximum (zero) when the UAV arrives. By combining these two navigation rewards, the UAV exhibits superior performance in precise navigation sub-tasks.

In practical scenarios, as the number of UAVs increases, collision avoidance during swarm formation becomes critical. The UAVs must learn an autonomous obstacle avoidance policy, for which the following collision penalty function is designed (Equation (8)):

$$r_i^{\text{collision}} = \begin{cases} -k, & d < d_{\min} \\ -k \left(\frac{d - d_{\text{avoid}}}{d_{\text{avoid}}} \right)^{0.2}, & d_{\min} \leq d < d_{\text{avoid}} \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

where k is a positive penalty coefficient, and d represents the distance between the UAV and its nearest neighbor. When d is less than d_{avoid} , the UAV perceives the need to evade; when d falls below $d_{\min}$, indicating that a collision has occurred, the UAV receives the maximum penalty.

In summary, the total reward space of the UAV comprises three components: attitude stability, swarm navigation, and collision avoidance, as expressed in Equation (9):

$$r_i = r_i^{\text{stab}} + (r_i^{\text{step}} + r_i^{\text{goal}}) + r_i^{\text{collision}} \quad (9)$$

2.2. State-Modulated Graph Attention Mechanism

Traditional MADDPG algorithms concatenate neighbor observations into a fixed-length vector, which fails when the neighbor count is time-varying. To resolve this, the observation space of the i -th UAV is first decoupled as $\mathbf{o}_i = [\mathbf{o}_i^{\text{self}}, \Delta \mathbf{P}_i]^T$. Here, $\mathbf{o}_i^{\text{self}}$ contains the ego-states (velocity, attitude, angular velocity), and $\Delta \mathbf{P}_i = \{\Delta \mathbf{p}_{ij}\}_{j=1}^N$ denotes the relative positions of all neighbors within the sensing range. The relative position $\Delta \mathbf{p}_{ij}$ is defined as:

$$\Delta \mathbf{p}_{ij} = \begin{cases} \mathbf{p}_j - \mathbf{p}_i, & j \neq i \\ 0, & j = i \end{cases}$$

To prevent the curse of dimensionality and reflect the varying importance of spatial information under different motion patterns, a state-modulated intent vector is designed. The critical kinematic state $\mathbf{s}_i$ is defined as:

$$\mathbf{s}_i = [\mathbf{v}_i^T, \boldsymbol{\omega}_i^T]^T$$

where $\mathbf{v}_i$ and $\boldsymbol{\omega}_i$ represent the linear and angular velocities, respectively. This state is linearly mapped to an intent vector $\mathbf{q}_i$ via a parameter matrix $\mathbf{W}_q$:

$$\mathbf{q}_i = \mathbf{W}_q \mathbf{s}_i$$

This mapping explicitly endows the agent with dynamic attention preferences. For instance, during high-speed cruising, forward neighbors receive higher attention, whereas during aggressive turning maneuvers, lateral awareness is prioritized. To process the neighbor information, the raw relative position $\Delta \mathbf{p}_{ij}$ is linearly projected into a latent Key-Value space via learnable parameter matrices $\mathbf{W}_k$ and $\mathbf{W}_v$. Specifically, the spatial descriptor $\mathbf{k}_{ij}$ and the interaction feature $\mathbf{v}_{ij}$ are formally defined as:

$$\mathbf{k}_{ij} = \mathbf{W}_k \Delta \mathbf{p}_{ij}$$

$$\mathbf{v}_{ij} = \mathbf{W}_v \Delta \mathbf{p}_{ij}$$

Here, $\mathbf{k}_{ij}$ encapsulates the geometric relationship of neighbor j , acting as an identity tag for matching, while $\mathbf{v}_{ij}$ embeds the actual spatial context to be aggregated. Using the intent vector $\mathbf{q}_i$ and the spatial descriptor $\mathbf{k}_{ij}$, the raw attention score is calculated via a scaled dot-product model to mitigate vanishing gradients:

$$e_{ij} = \frac{\mathbf{q}_i^T \mathbf{k}_{ij}}{\sqrt{d_k}}$$

A Softmax function normalizes these scores to obtain the final attention weight a_{ij} :

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^N \exp(e_{ik})}$$

Finally, the interaction features $\mathbf{v}_{ij}$ are aggregated using these weights to generate a fixed-dimensional spatial context representation $\mathbf{h}_i^{\text{attn}}$ through a non-linear activation function f_{attn} :

$$\mathbf{z}_i = \sum_{j=1}^N a_{ij} \mathbf{v}_{ij}$$

$$\mathbf{h}_i^{\text{attn}} = f_{\text{attn}}(\mathbf{z}_i)$$

Unlike traditional graph-based MARL models that require exhaustive tuning for hyperparameters, the proposed state-modulated GAT is naturally robust. By mapping the intent query $\mathbf{q}_i$ directly to the UAV's real-time kinematic state $\mathbf{s}_i$, the attention weights a_{ij} are driven by actual flight mechanics rather than abstract hidden variables. Consequently, the parameter matrices ($\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$) simply align spatial geometry with the drone's movement intent, making the network much less sensitive to parameter shifts. Additionally, using a lightweight, single-head attention structure prevents the model from overfitting to fixed swarm formations.

2.3. Network Architecture and Algorithm Workflow

The proposed GAT–Actor network aims to fuse ego-perception with spatial relationship modeling. The network processes the raw observation by extracting the aforementioned $\mathbf{h}_i^{\text{attn}}$ using the state-modulated GAT module. This effectively compresses the variable-length relative position matrix $\Delta \mathbf{P}_i$ into a fixed-length feature, drastically reducing the training complexity in dynamic topologies. Specifically, the Actor is parameterized by a Multi-Layer Perceptron (MLP) with two hidden layers of 256 neurons.

Similarly, the centralized Critic network is enhanced to process the joint feature states and actions of all agents, providing a stable, low-variance value gradient for Actor updates. The Critic also adopts an MLP architecture with 256 hidden units. Both networks are trained using the Adam optimizer. Coupled with the PER mechanism—which assigns higher sampling probabilities to transitions with high Temporal-Difference (TD) errors—the proposed PER-MADDPG-GAT algorithm forces the network to focus on critical interaction scenarios, significantly boosting convergence stability.

For real-world UAV applications, the proposed decentralized framework offers three practical advantages: The computational complexity of the local GAT module is $\mathcal{O}(K \cdot d_f)$, where K is the neighbor count and d_f is the feature dimension. With a fixed $K = 3$, the complexity reduces to $\mathcal{O}(d_f)$, making the onboard computation independent of the total swarm size N and ensuring high scalability. Each UAV only broadcasts its low-dimensional kinematic states (position and velocity) to nearby neighbors. This avoids global state broadcasting, keeping the communication payload minimal and strictly within the limits of standard embedded wireless modules. The actor network utilizes a single-layer attention mechanism and a shallow MLP. This lightweight architecture requires minimal floating-point operations, enabling low-latency forward inference that satisfies the high-frequency execution requirements of embedded flight controllers.

2.4. Swarm Formation Planning and Node Allocation

While the GAT module provides dynamic perceptual capabilities, purely reactive rewards fail to enforce strict geometric formations. Therefore, a virtual structure approach is adopted to model the high-level objectives. Assuming the geometric center of the swarm is the origin of the reference frame, the predefined topology is represented by a set of desired relative node positions:

$$\mathcal{T} = \left\{ \mathbf{p}_k^{\text{ref}} \in \mathbb{R}^3 \mid k = 1, 2, \dots, N \right\}$$

where $\mathbf{p}_k^{\text{ref}}$ is the expected position vector of the k -th virtual node relative to the swarm center. To match physical UAVs with these virtual nodes optimally, a decentralized auction mechanism is employed. The bidding cost c_{ik} for UAV i to claim node k is defined by the Euclidean distance:

$$c_{ik} = \left\| \mathbf{p}_i - \mathbf{p}_k^{\text{ref}} \right\|$$

Based on this cost matrix, a linear assignment algorithm efficiently minimizes the total maneuvering cost during topology reconfiguration. This mechanism, seamlessly integrated with the UAV-based control policy, yields a highly robust cooperative control solution for dynamic environments.

3. Experimental Validation and Results Analysis

3.1. Experimental Parameter Settings

To ensure the simulation accurately reflects the dynamic characteristics of multi-UAV cooperative tasks in the real world, a 3D flight environment is constructed using the

Genesis physics engine. This platform provides rigid-body dynamics for the quadrotors. The physical attributes of the UAVs and the swarm parameters for the cooperative tasks are rigorously defined. Table 1 details the specific physical parameters, task settings, and reward function weights, which form the baseline for evaluating the effectiveness and stability of the control policies.

The coefficients of the joint reward function were empirically determined based on strict hierarchical priorities defined by actual UAV flight requirements. First, the quadcopter must maintain a stable attitude and fly in a manner consistent with the laws of physics. Second, the UAV must avoid collisions in space, which aligns with real-world mission requirements. Finally, there is a coefficient for mission completion, which serves to gradually guide the UAV's movement toward achieving the mission objective. During the training process described in this paper, although the convergence rate of the DRL policy exhibits moderate sensitivity to the absolute values of these weights, the final flight performance remains highly robust as long as this relative order of priority is strictly maintained. Conversely, if the weights of the reward coefficients are adjusted arbitrarily, the first challenge encountered is that training becomes ineffective; the drone will be unable to achieve normal flight capabilities even after tens of thousands of training iterations.

Table 1. Simulation Environment Parameters and Reward Function Weights.

Parameter Name	Symbol	Value
Initial Number of UAVs	N	5
Desired Swarm Radius	R	0.3 m
Target Flight Altitude	h_{target}	1.0 m
Simulation Frequency	f	100 Hz
Max Steps per Episode	T_{max}	500
Total Training Episodes	$Episodes$	30,000
Batch Size	N_{batch}	256
Map Dimensions	—	6 m $\times$ 6 m $\times$ 3 m
Target Tracking Reward	w_{dist}	3.0
Proximity Guidance Reward	w_{prox}	300.0
Altitude Maintenance Reward	w_h	1.0
Velocity Tracking Reward	w_v	1.0
Attitude Stability Reward	w_{pose}	10.0
Action Smoothness Penalty	w_{act}	0.02
Actor Learning rate	η_a	1×10^{-4}
Critic Learning rate	η_a	1×10^{-3}
Replay Buffer Capacity	N_{buffer}	1×10^6
PER prioritization exponent	α_{PER}	0.6
PER importance sampling exponent	β_{PER}	0.4

3.2. Experimental Results and Analysis

This section validates the performance of the multi-UAV swarm algorithm integrated with the GAT module within the Genesis environment. Initially, four baseline and proposed algorithms are trained under identical task scenarios to form specific swarm configurations. Subsequently, utilizing the converged policies of these four algorithms, disturbance rejection tests are conducted in two dynamic topology scenarios to rigorously verify the effectiveness of the GAT module in handling dynamic interactive environments.

To evaluate the learning efficiency and final policy quality of each algorithm, Figure 2 presents the average reward curves over 30,000 training episodes for MADDPG, PER-

MADDPG, MADDPG-GAT, and the proposed PER-MADDPG-GAT. The solid lines represent the mean values across 5 independent trials utilizing different random seeds, while the shaded regions denote the standard deviations, reflecting the volatility and stability of the training process. Observing the overall trends, all four algorithms experience a brief decline in rewards during the initial exploration phase, followed by a gradual ascent and eventual convergence as the policies are optimized. However, significant disparities emerge in convergence speed and final steady-state rewards: the original MADDPG algorithm exhibits the lowest learning efficiency. Its reward recovery is sluggish, remaining at a relatively low level throughout the training cycle. This indicates that relying solely on basic multi-agent gradient policies is insufficient to rapidly find optimal solutions when dealing with high-dimensional state spaces and complex cooperative tasks. A comparison between MADDPG and MADDPG-GAT reveals that the introduction of the graph attention mechanism plays a decisive role in elevating the policy's performance ceiling. The slope of the MADDPG-GAT curve is significantly steeper than that of the baseline algorithm, and its final converged reward is substantially higher. This demonstrates that by dynamically extracting key features of neighbor nodes, the GAT enhances the agents' accuracy in perceiving the environmental topology, enabling them to learn more efficient cooperative strategies. The PER-MADDPG-GAT algorithm proposed in this paper combines the advantages of both mechanisms, exhibiting optimal comprehensive performance. After the initial exploration, this algorithm ascends with the steepest gradient, reaching a high-score region unmatched by the others at approximately 10,000 episodes, and consistently maintains the highest average reward level. The exceedingly narrow shaded region in the later stages of training indicates that PER-MADDPG-GAT maintains stable convergence across different random seeds, showcasing exceptional robustness. Furthermore, to rigorously validate this performance enhancement, a Welch's *t*-test was conducted on the converged average rewards. The statistical analysis yielded a *p*-value of $p < 0.05$ when comparing the proposed PER-MADDPG-GAT against the baseline algorithms, providing strong statistical evidence that the improvement in policy quality is indeed significant. Upon the convergence of the four algorithms, two sets of cooperative navigation experiments are established to verify their configuration maintenance capabilities and navigation precision. After the UAVs are randomly initialized in the 3D space, the policy must output low-level actions based on current states to rapidly form a stable, predefined geometric configuration. Subsequently, the system relocates the target points in the environment to guide the swarm in executing simple straight-line and complex rectangular trajectory navigation, respectively. The stability of the policies is verified by observing the swarm's motion trajectories and configuration retention. Figure 3 displays two motion trajectories of the PER-MADDPG-GAT algorithm during the testing phase. It is evident that after spawning from the origin, the UAVs rapidly adjust their attitudes and positions to form a swarm at the yellow star marker in Figure 3a. They then navigate toward the target point to complete the mission. Throughout this process, the trajectory remains remarkably smooth without noticeable formation dispersion or collisions, proving the algorithm's stability in foundational navigation tasks. In Figure 3b, four continuous target points are set. The results clearly show that even when the swarm is required to execute 90-degree turns, the policy successfully maintains the geometric shape during target navigation.

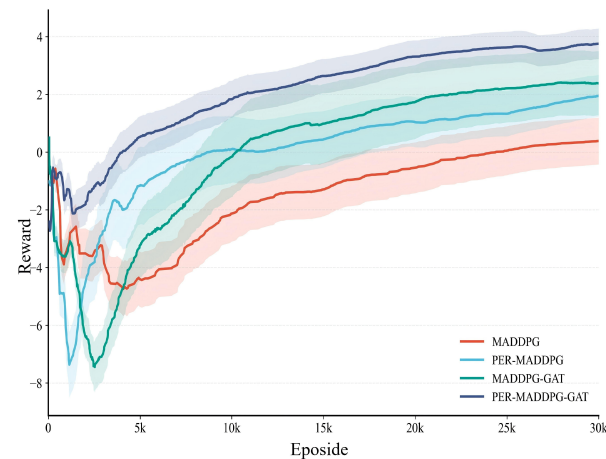


Figure 2. Reward Variation Curves During Training Process for Different Algorithms.

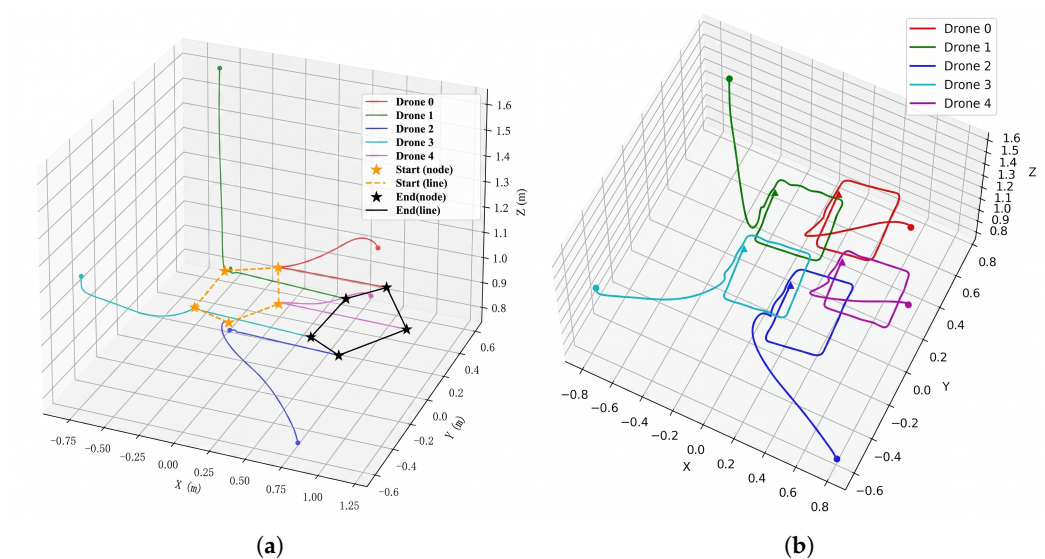


Figure 3. Different target navigation trajectories of the PER-MADDPG-GAT algorithm. (a) Straight-line trajectory: the UAV swarm assembles from random initial positions and navigates to a single target point while maintaining formation. (b) Rectangular trajectory: the swarm flies through four consecutive target points requiring 90-degree turns, demonstrating formation retention during complex maneuvers.

To further quantitatively evaluate the performance, we compare the swarm errors of the four algorithms across these two tasks. The swarm error E is defined in Equation (10):

$$E = \frac{1}{N} \sum_{i=1}^N \left\| \mathbf{p}_i - \frac{1}{N} \sum_{j=1}^N \mathbf{p}_j \right\|_2 - R \quad (10)$$

where N is the total number of UAVs, R is the predefined expected radius of the swarm, and $\mathbf{p}_i$ is the position of the i -th UAV. As shown in Figure 4, PER-MADDPG-GAT consistently maintains the lowest and most stable error among the four algorithms. This result not only verifies the baselines' capabilities to complete basic tasks but also highlights the precision advantage of the proposed method, providing a benchmark for evaluating performance under more complex conditions.

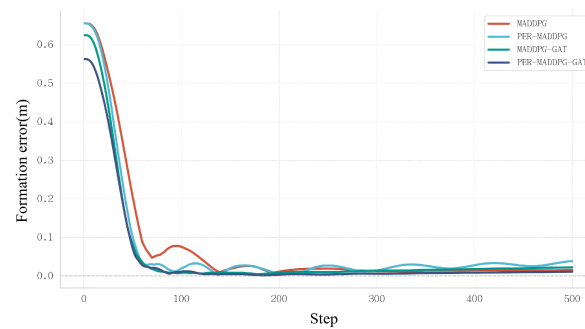


Figure 4. Average Swarm Error of Four Algorithms in Target Navigation Tasks.

The ablation studies in this section primarily aim to verify the adaptability and stability of the GAT module when the number of UAVs and the topological structure change. Two scenarios involving dynamic quantity variations are designed: UAV failure (node reduction) and dynamic UAV reinforcement (node addition). The four aforementioned algorithms are assigned to execute these tasks to validate the GAT module's superiority in processing UAV interaction relationships in complex environments. First, to verify the robustness and adaptability of the PER-MADDPG-GAT algorithm in extreme scenarios such as sudden hardware failures or communication disruptions, an ablation experiment with dynamically reducing UAV numbers is designed. The scenario is initialized with a 5-UAV swarm executing a straight-line navigation task. At step 200 of the simulation, a failure mechanism is artificially introduced, forcing one UAV to abruptly exit the swarm. At this juncture, the remaining 4 UAVs must perceive the topological change without human intervention, rapidly reconfigure from the original pentagonal formation into a square formation, and sustain this new geometry to complete the remaining navigation task.

The test results are shown in Figure 5. The UAV swarm initially gathers and maintains a stable pentagonal formation. Upon reaching the failure trigger point, the compromised UAV detaches. The remaining four UAVs respond swiftly to the topological shift, adjusting their relative positions to autonomously reconstruct the formation from a pentagon into a square. Post-reconfiguration, they continue to maintain the swarm shape and fly steadily toward the target point. The overall trajectory is smooth, exhibiting no system crashes or route divergence due to node loss, directly proving the proposed algorithm's adaptive capability in handling dynamic node dropouts. To visually demonstrate this topological reconfiguration, Figure 6 presents snapshots of the experimental scene in the Genesis environment before and after the dynamic reduction.

To quantitatively analyze the GAT module's contribution in such dynamic scenarios, Figure 7 compares the system error variations of the four algorithms before and after the failure. Initially, all algorithms successfully guide the UAVs to form a stable swarm. At step 200, when one UAV fails, the geometric center of the swarm abruptly shifts, necessitating a recalculation of the configuration allocation; consequently, the swarm errors for all algorithms spike. For the two algorithms lacking the GAT module, MADDPG (red curve) oscillates violently and fails to converge, indicating that the UAVs cannot maintain stable flight in the Genesis environment and the entire swarm enters a state of persistent oscillation. PER-MADDPG (light blue curve) converges, but its swarm error does not decrease, implying that the swarm fails to reform into a square geometry post-failure. These two algorithms demonstrate that a fixed-dimensional state space cannot accommodate variations in input dimensions, inevitably leading to swarm destabilization.

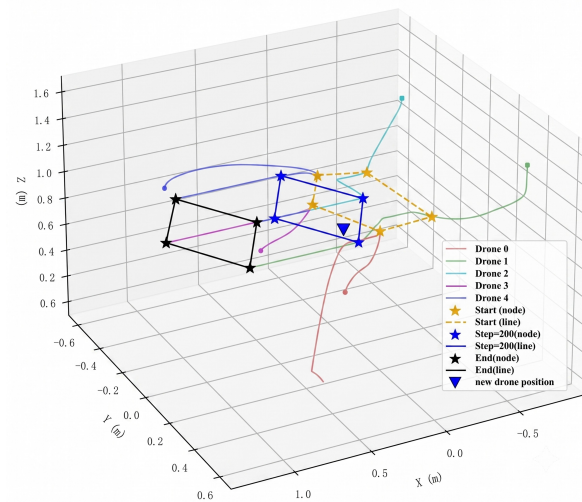


Figure 5. Reduced Drone Count, Swarm Mission Trajectory Map.

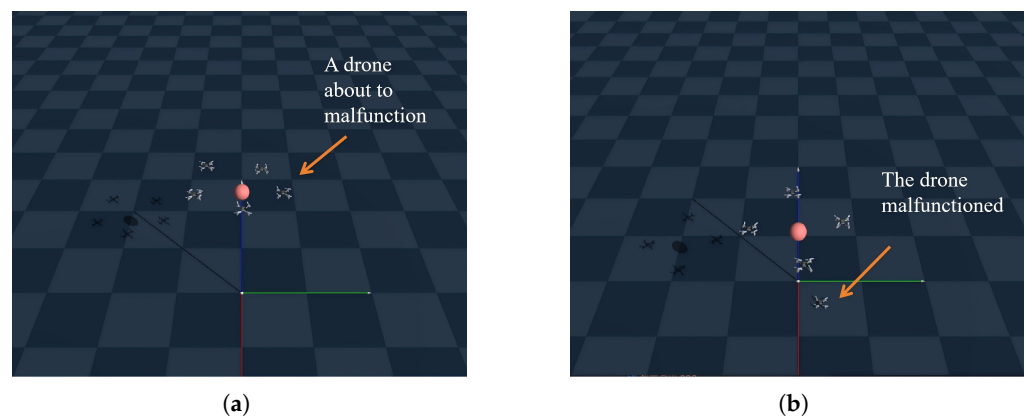


Figure 6. Dynamic reduction of UAVs in the Genesis simulation environment. (a) Before reduction: five UAVs maintain a stable pentagonal formation during the straight-line navigation task. (b) After reduction: following the forced removal of one UAV at step 200, the remaining four UAVs autonomously reconfigure into a square formation and continue the mission.

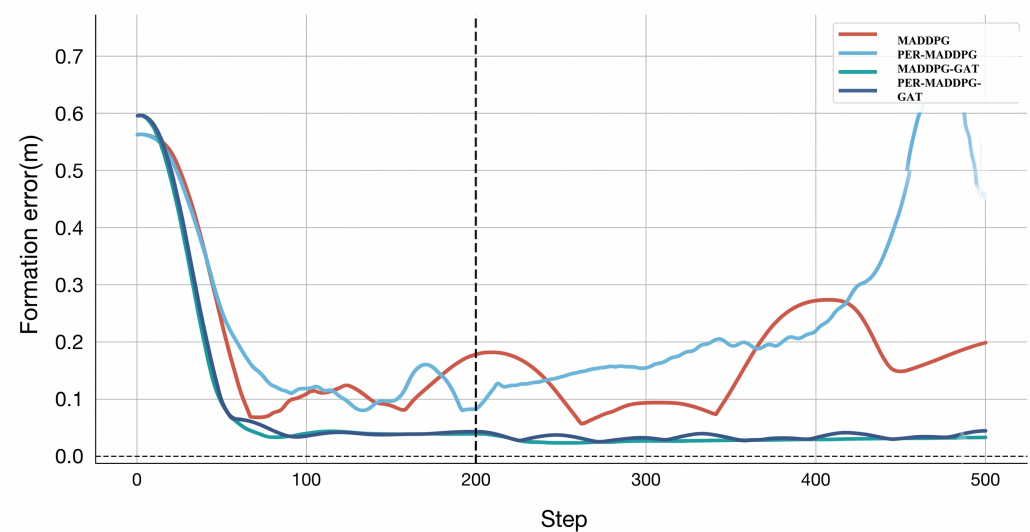


Figure 7. Swarm Error Before and After Reduction in Number for Four Algorithms, the black dashed line indicates the time where the number of drones changed.

Conversely, algorithms incorporating the GAT module exhibit significant advantages. Specifically, PER-MADDPG-GAT, after a brief perturbation, sees its error curve drop rapidly and stabilize near zero. This indicates that the GAT module effectively compensates for the information vacuum caused by the missing node by dynamically adjusting the attention weights of the remaining neighbors, enabling the swarm to quickly adapt to the new topology (a 4-UAV swarm), thereby verifying the mechanism's exceptional robustness in dynamic topological environments. To further validate the algorithm's adaptability in swarm scaling and dynamic task allocation, an ablation experiment involving dynamic UAV reinforcement is designed. The initial scenario features 5 UAVs executing a straight-line navigation task. At step 200, a new reinforcement UAV is introduced into the system. The swarm must perceive the integration of the new node, reallocate target points, and autonomously expand and reconfigure from the initial pentagonal topology to a hexagonal one without human intervention, subsequently flying to the destination while maintaining the new shape.

Figure 8 illustrates the 3D trajectory of the PER-MADDPG-GAT algorithm during this dynamic reinforcement task. Initially, the 5 UAVs rapidly assemble into a stable pentagon (marked by the yellow star). At step 200, as the reinforcement UAV cuts in from the flank, the swarm reacts promptly to this perturbation; the original UAVs proactively adjust their positions to yield space for the new node. As indicated by the blue hexagram, the swarm successfully accommodates the reinforcement, smoothly completing the topological transition from a pentagon to a hexagon. Post-reconfiguration, the 6 UAVs maintain a tight hexagonal formation until reaching the destination (black star). The entire reinforcement process features a smooth trajectory with no collisions or drastic evasive maneuvers between the old and new nodes, proving the algorithm's stability when handling dynamic node integration. Figure 9 similarly displays the swarm variations before and after the reinforcement's arrival in the Genesis environment.

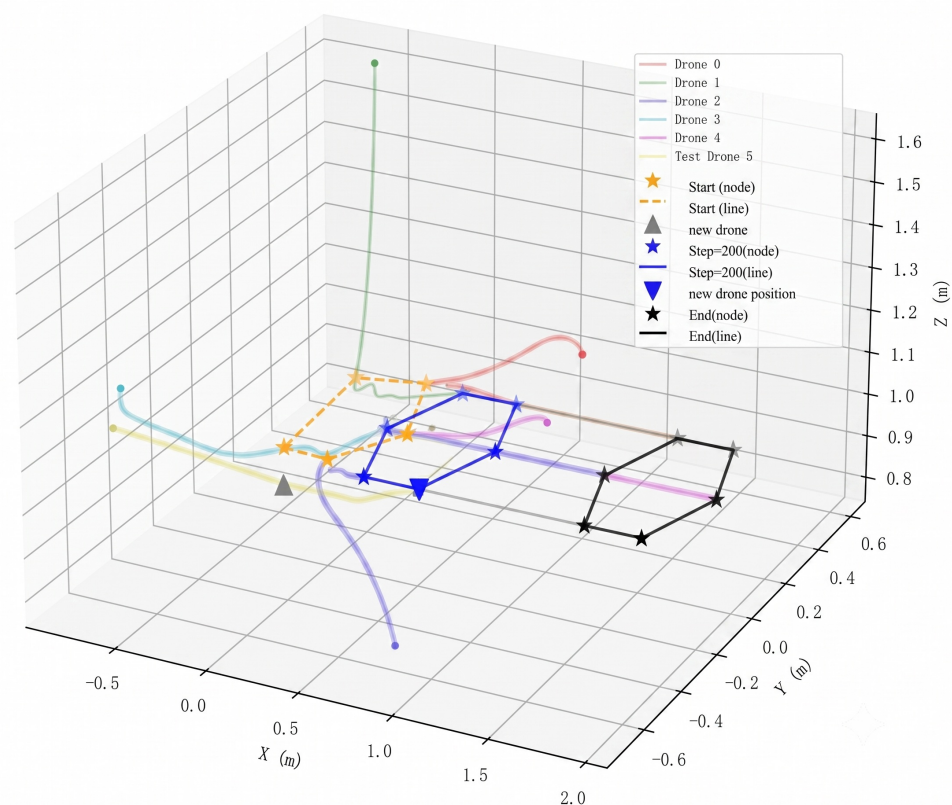


Figure 8. Increased Number of Drones, Swarm Mission Trajectory Map.

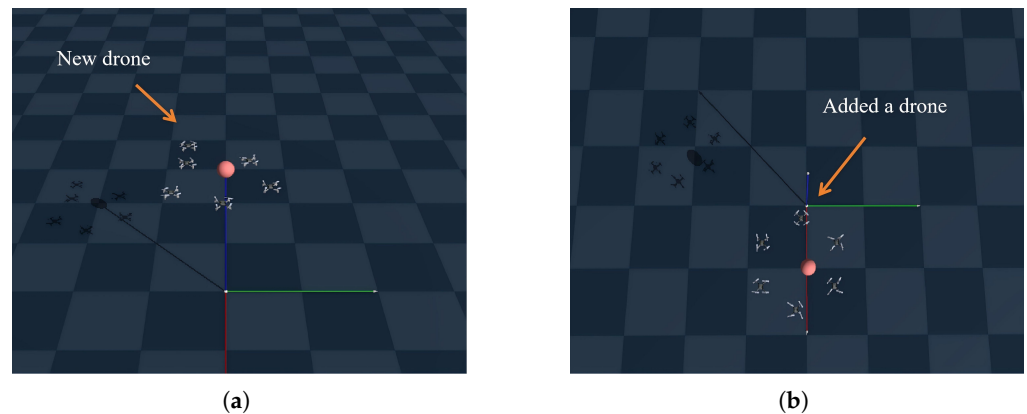


Figure 9. Increasing number of drones in the Genesis simulation environment. (a) Before addition: five UAVs execute the straight-line navigation task in a stable pentagonal formation. (b) After addition: a reinforcement UAV joins at step 200, and the swarm seamlessly expands from a pentagon to a hexagon without collision or formation disruption.

Figure 10 further quantifies the performance disparities among the four algorithms when confronting swarm scaling. At the exact moment the reinforcement UAV joins at step 200 (indicated by the dashed line), the relative relationships between UAVs alter. MADDPG and PER-MADDPG, lacking the graph attention mechanism, exhibit severe adaptability failures. As illustrated, the error for PER-MADDPG escalates sharply post-step 200 and shows a diverging trend. This demonstrates that a rigid network structure struggles to effectively process the interactive information of the added node, resulting in the complete collapse of the swarm configuration. In stark contrast, MADDPG-GAT and PER-MADDPG-GAT exhibit profound robustness. Benefiting from the inherent adaptability of graph neural networks to variable-length inputs, the UAV instantaneously assigns attention weights to the newly joined node. The experimental data shows that PER-MADDPG-GAT experiences only marginal error fluctuations at the moment of reinforcement integration before rapidly reconverging to a steady state.

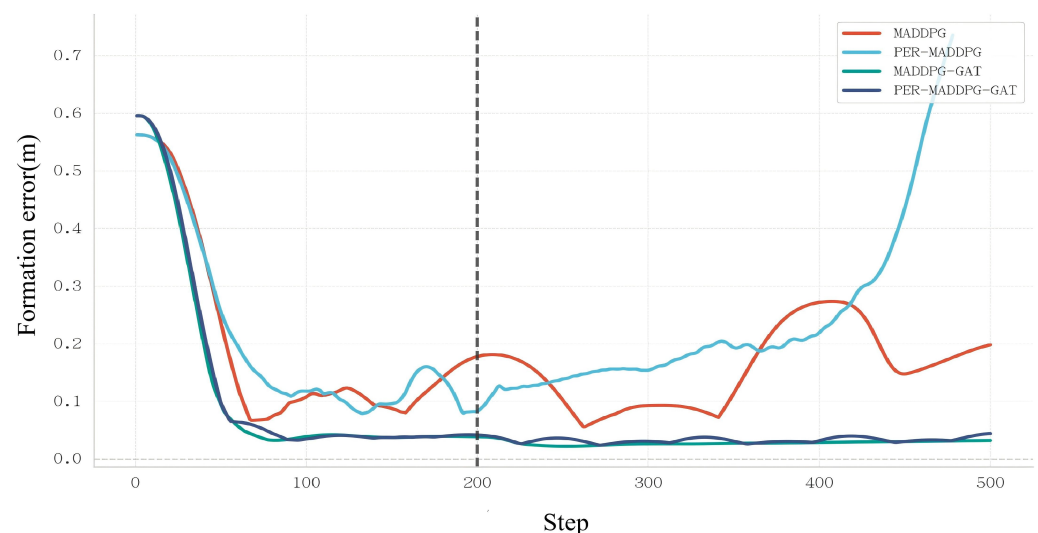


Figure 10. Swarm Error Before and After Increasing the Number for Four Algorithms. The black dashed line indicates the time when the number of drones changed.

4. Conclusions

This paper presented an end-to-end 3D motion control framework based on a state-modulated GAT for UAV swarms in dynamic topologies. To overcome the fixed-

dimensionality constraints of traditional MARL, the proposed PER-MADDPG-GAT algorithm dynamically aggregates variable-length neighbor features using kinematic-dependent attention weights. Additionally, integrating a virtual structure with a decentralized auction mechanism ensures precise macroscopic formation control.

Simulations in the Genesis 3D physics engine confirm that our framework seamlessly adapts to sudden topological shifts, such as node failures or reinforcements, under strict rigid-body constraints. Compared to baseline methods, it significantly improves convergence stability, trajectory precision, and robustness. Future work will focus on addressing communication latency and deploying the framework on physical quadrotors.

While the proposed framework demonstrates clear advantages in adapting to dynamic topologies, the current empirical validation is bounded by small-scale swarm variations and comparisons with MADDPG-based architectures. To address these limitations, we plan to extend this work in three main directions. First, to overcome the computational bottleneck of simulating full 3D rigid-body dynamics for numerous agents, we will utilize distributed computing clusters to scale the experiments to larger swarms under severe communication noise. Second, we aim to benchmark our framework against more recent on-policy or sequence-modeling multi-agent algorithms, such as MAPPO and Transformer-based architectures. Alongside this, we will conduct granular ablation studies to isolate and quantify the specific performance contribution of the kinematic state-modulation mechanism versus standard distance-based attention. Finally, we will continue working toward bridging the sim-to-real gap by testing the control policy on physical quadrotors.

Author Contributions: Conceptualization, Y.C. and C.Z.; Methodology, Y.C. and Q.X.; Software, Y.C. and Q.X.; Formal analysis, Y.C., Q.X. and C.Z.; Investigation, Y.C., Q.X. and Z.L.; Resources, Z.L.; Data curation, Q.X. and Z.L.; Writing—original draft, Y.C., Q.X., C.Z. and Z.L.; Writing—review & editing, Y.C., C.Z. and Z.L.; Visualization, Y.C. and Q.X.; Supervision, C.Z. and Z.L.; Project administration, C.Z.; Funding acquisition, C.Z. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Key Project of Scientific Research in Universities of Anhui Province (Natural Sciences), Research on Privacy Data Security Sharing Technology Based on Blockchain and CP-ABE with grant number: 2024 AH051527, Key Project of Teaching and Research at Hefei University under grant number: 2023 hfujyd05.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: During the preparation of this manuscript, the authors used deepseek-v4-pro and gemini-3.1-pro for the purposes of grammar checking and visual enhancement. The authors have reviewed and edited the output and take fully responsible for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
GAT	Graph Attention Network
PER	Prioritized Experience Replay
MADRL	Multi-Agent Deep Reinforcement Learning
MADDPG	Multi-Agent Deep Deterministic Policy Gradient

References

1. Zhang, J.; Zhang, L. Bio-inspired neighbor selection mechanism and performance evaluation for UAV Swarm: An extended visual-attention model. *IFAC-PapersOnLine* **2025**, *59*, 1569–1574.
2. Mishra, B.; Garg, D.; Narang, P.; Mishra, V. Drone-surveillance for search and rescue in natural disaster. *Comput. Commun.* **2020**, *156*, 1–10. [\[CrossRef\]](#)
3. Dilshad, N.; Hwang, J.; Song, J.; Sung, N. Applications and challenges in video surveillance via drone: A brief survey. In *Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC)*; IEEE: Piscataway, NJ, USA, 2020; pp. 728–732.
4. Albanese, A.; Sciancalepore, V.; Costa-Pérez, X. SARDO: An automated search-and-rescue drone-based solution for victims localization. *IEEE Trans. Mob. Comput.* **2021**, *21*, 3312–3325.
5. Hoang, M.T.O.; Grøntved, K.A.R.; Van Berkel, N.; Skov, M.B.; Christensen, A.L.; Merritt, T. Drone swarms to support search and rescue operations: Opportunities and challenges. In *Cultural Robotics: Social Robots and Their Emergent Cultural Ecologies*; Springer: Cham, Switzerland, 2023; pp. 163–176.
6. Chen, W.; Liu, J.; Guo, H.; Kato, N. Toward robust and intelligent drone swarm: Challenges and future directions. *IEEE Netw.* **2020**, *34*, 278–283. [\[CrossRef\]](#)
7. Saffre, F.; Hildmann, H.; Karvonen, H. The design challenges of drone swarm control. In *Proceedings of the International Conference on Human-Computer Interaction*, Washington, DC, USA, 24–29 July 2021; pp. 408–426.
8. Liu, H.; Suzuki, S.; Wang, W.; Liu, H.; Wang, Q. Robust control strategy for quadrotor drone using reference model-based deep deterministic policy gradient. *Drones* **2022**, *6*, 251. [\[CrossRef\]](#)
9. Ahn, H.; Hu, M.; Chung, Y.; You, K. Sliding-mode control for flight stability of quadrotor drone using adaptive super-twisting reaching law. *Drones* **2023**, *7*, 522.
10. Zhang, P.; Zhou, J.; Tian, D.; Duan, X.; Zhao, D.; Guo, K. Distributed Robust UAVs Formation Control Based on Semidefinite Programming. *IEEE Access* **2024**, *12*, 61123–61134.
11. Liu, J.; Zhang, Z.; Zhang, L.; Zhang, K.; Yan, C. A Decoupled Non-Uniform Guiding Vector Field for Multi-Robot Coordinated Path-Following on 2D Manifolds. *IEEE Trans. Aerosp. Electron. Syst.* **2026**, *62*, 5751–5766.
12. Yan, C.; Zhang, Z.; Bao, W.; Zhang, K.; Zhang, L. Collision-Free Robust Trajectory Tracking Control for Fixed-Wing UAV with Flexible Performance Constraints. *Aerosp. Sci. Technol.* **2025**, *168*, 111217.
13. Zhang, Y.; Li, Y.; Wu, Z.; Xu, J. Deep reinforcement learning for UAV swarm rendezvous behavior. *J. Syst. Eng. Electron.* **2023**, *34*, 360–373. [\[CrossRef\]](#)
14. Li, B.; Wang, J.; Song, C.; Yang, Z.; Wan, K.; Zhang, Q. Multi-UAV roundup strategy method based on deep reinforcement learning CEL-MADDPG algorithm. *Expert Syst. Appl.* **2024**, *245*, 123018.
15. Zhang, Y.; Wu, Z.; Ma, Y.; Sun, R.; Xu, Z. Research on autonomous formation of multi-UAV based on MADDPG algorithm. In *Proceedings of the 2022 IEEE 17th International Conference on Control & Automation (ICCA)*; IEEE: Piscataway, NJ, USA, 2022; pp. 249–254.
16. Xing, X.; Ma, Y.; Lei, Y.; Li, Y.; Xiao, B. Multi-UAV Rendezvous Trajectory Planning Based on Improved MADDPG Algorithm in Complex Dynamic Obstacle Environments. *IEEE Access* **2025**, *13*, 5580–5591. [\[CrossRef\]](#)
17. Chen, J.; Wang, Y.; Zhang, Y.; Lu, Y.; Shu, Q.; Hu, Y. Extrinsic-and-Intrinsic Reward-Based Multi-Agent Reinforcement Learning for Multi-UAV Cooperative Target Encirclement. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 17653–17665.
18. Yang, Q.; Lu, J.; Zhang, Y.; Shao, S. Multi-UAV Cooperative Path Planning via Graph Neural Network and Proximal Policy Optimization. *IEEE Access* **2025**, *13*, 193575–193588. [\[CrossRef\]](#)
19. Xu, Z.; Wang, Y.; Wu, Z.; Zheng, W.X.; Hu, C. Federated learning-based offset-free distributed control of nonlinear networked systems with application to IIoT. *IEEE Trans. Netw. Sci. Eng.* **2025**, *12*, 1859–1871.
20. Sun, J.; Zhu, Z.; Li, H.; Chai, Y.; Qi, G.; Wang, H.; Hu, Y.H. An integrated critic-actor neural network for reinforcement learning with application of DERs control in grid frequency regulation. *Int. J. Electr. Power Energy Syst.* **2019**, *111*, 286–299. [\[CrossRef\]](#)
21. Yadav, K.; Palaparthi, D.P.; Garg, G.; HB, H.; Priyanka, T.P.; Kaliappan, S. MAPPO-GAT: A Graph-Attention-Driven Multi-Agent Reinforcement Learning Framework for Entropy-Aware Autonomous Drone Swarm Coordination. In *Proceedings of the 2025 1st International Conference on Advancement in Futuristic Technologies (ICAFT)*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–8.
22. Fu, S.; Li, K.; Huang, H.; Ma, C.; Fan, Q.; Zhu, Y. Red-billed blue magpie optimizer: A novel metaheuristic algorithm for 2D/3D UAV path planning and engineering design problems. *Artif. Intell. Rev.* **2024**, *57*, 134.
23. Lyu, R.; Pang, J.; Tian, X.; Zhao, W.; Zhang, J. How to optimize the 2D/3D urban thermal environment: Insights derived from UAV LiDAR/multispectral data and multi-source remote sensing data. *Sustain. Cities Soc.* **2023**, *88*, 104287.
24. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
25. Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv* **2015**, arXiv:1511.05952.

26. Askari, A.; Mortazavi, M.; Talebi, H. UAV formation control via the virtual structure approach. *J. Aerosp. Eng.* **2015**, *28*, 04014047. [\[CrossRef\]](#)
27. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; MIT Press: Cambridge, MA, USA, 2018.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.