Modeling Dynamic Obstacle Avoidance Strategy of Drone Swarms Combined with Multi-Agent Reinforcement Learning

Xianghua Fang[1], Ken Chen[2], Chenghao Ren[3], HongChuan Jiang[2], Bing Li[3]
[1]College of Forestry Science and Technology, Lishui Vocational & Technical College, Lishui 323000, Zhejiang, China
[2]Zhejiang Rongqe Technology Co., Ltd., Lishui 323000, Zhejiang, China
[3]Shenzhen Hozon-funtion Technology Co., Ltd., Shenzhen 518000, Guangdong, China

Corresponding author: Ken Chen (e-mail: chenken@rongqe.cn)

ABSTRACT This paper proposes the Locally-decoupled and Embedding-enhanced Multi-Agent Deep Deterministic Policy Gradient (LDE-MADDPG) algorithm to address poor scalability and delayed response in drone swarm dynamic obstacle avoidance under complex cooperative environments. Such autonomous coordination capabilities are also important for distributed sensing, wireless networking, and electromagnetic information exchange in future intelligent aerial systems. The algorithm introduces three key innovations beyond standard MADDPG: a Graph Attention Network module that encodes variable-length observations into fixed-dimensional embeddings for swarm-size generalization; a dual-path critic with a global branch guiding policy updates and a local branch specializing in obstacle avoidance evaluation; and a hierarchical reward integrating multi-objective signals. Evaluated across eight static and dynamic obstacle scenarios, LDE-MADDPG achieves significantly lower collision rates (2.1%–4.2% in static scenarios and 3.8%–7.2% in dynamic scenarios) than state-of-the-art baselines and reaches a 97.5% mission completion rate in 100 random scenarios. The proposed framework demonstrates robust scalability and real-time coordination capability for dynamic environments, while providing a reliable decision-making paradigm for intelligent multi-agent systems operating in communication-intensive and electromagnetically complex application scenarios.

INDEX TERMS drone swarm, multi-agent reinforcement learning, graph attention network, electromagnetic environment, intelligent wireless systems

I. INTRODUCTION

THE growing demand for robust, collaborative systems in complex environments has led to significant technological advancements, exemplified by the success of UAV (Unmanned Aerial Vehicle) swarms in scenarios such as formation flying, regional coverage, and emergency response [1, 2]. However, the need for such resilient systems extends to industrial domains [3,4]. Traditional centralized control methods [5,6] are difficult to adapt to scale changes and local disturbances, and existing reinforcement learning strategies mostly rely on fixed-dimensional state inputs, which can easily lead to model failure when the cluster scale is dynamically adjusted. This paper applies the LDE-MADDPG (Local Decoupled and Embedding Multi-Agent Deep Deterministic Policy Gradient) algorithm to address the poor scalability and delayed obstacle avoidance response of drone swarms in dynamic environments, caused by the coupling of scale changes and individual behaviors. The algorithm makes three key technical contributions: First, a state feature learning module based on a graph attention network is designed to encode variable-length neighborhood observations into fixed-dimensional embedding vectors. This effectively addresses the state space mismatch problem caused by the changing number of drones in traditional methods and improves policy generalization. Second, a decoupled dual-path critic network is constructed to separate global collaborative evaluation from local behavior value calculation. The global critic optimizes the overall formation performance of the swarm, while the local critic independently supervises individual obstacle avoidance behaviors, enhancing sensitivity to collision risk and improving the credit allocation mechanism. Finally, a hierarchical reward structure is applied to integrate collision penalties, obstacle avoidance penalties, boundary violation penalties, and target location arrival, achieving unified optimization of macro-guidance and micro-safety.

II. RELATED WORK

As an important application of multi-agent systems, cooperative control of drone swarms has demonstrated significant potential in military reconnaissance, disaster relief, and intelligent transportation. Early research primarily employed

consensus protocols for formation control [7,8], though such methods struggle with sudden obstacles and communication interruptions in dynamic environments. With advances in reinforcement learning [9], data-driven approaches have emerged as effective solutions for high-dimensional nonlinear control problems. MADDPG [10,11], a representative centralized-training-decentralized-execution algorithm, enhances multi-agent cooperation through a global critic network that evaluates joint actions, and has been widely applied to drone formation and obstacle avoidance tasks. However, conventional MADDPG relies on fixed-dimensional state inputs, leading to state-space mismatch when swarm size varies [12]. Its single-critic structure also complicates credit assignment for individual behaviors, particularly during emergency obstacle avoidance, ultimately impacting system stability. While some studies introduce communication mechanisms or attention models to improve observability [13], most assume a fixed number of agents and lack scalability. Thus, developing policy representations adaptable to dynamic topologies remains a key challenge for enhancing swarm robustness. Graph neural networks have been integrated into multi-agent reinforcement learning to enhance state representation scalability. Graph Attention Networks (GAT) dynamically weight neighbor features [14], enabling fixed-dimensional embeddings from variable observations [15]. However, existing methods inadequately decouple individual and group evaluation [16], causing local obstacle avoidance to be overshadowed by global strategies. While decoupled critics show promise in other domains [17,18], they remain under-explored for drone swarms. The challenge persists in balancing collaboration and safety during formation switching with moving obstacles, necessitating frameworks that combine scalable encoding with fine-grained assessment. Recent multi-agent reinforcement learning advances for UAV coordination address communication efficiency and environmental dynamics. QMIX [19] enables value decomposition but struggles with continuous actions, while MAPPO [20,21] improves stability but lacks scalability. Attention mechanisms enhance local perception but overlook motion dynamics. Hierarchical approaches [22] face optimization challenges, and current obstacle modeling [23,24] lacks predictive capability. Most evaluations use fixed swarm sizes [25], failing to validate scalability. A comprehensive framework balancing real-time obstacle avoidance with cooperative performance remains needed.

III. METHOD

A. PROBLEM MODELING: MULTI-AGENT MARKOV DECISION PROCESS

To formally describe the collaborative obstacle avoidance and formation control tasks of UAV swarms [26,27] in complex dynamic environments, this paper models it as a partially observable multi-agent Markov decision process, defined as a seven-tuple:

$$M = \langle I, S, \{O_i\}_{i\in I}, \{A_i\}_{i\in I}, P, \{r_i\}_{i\in I}, \gamma \rangle \qquad (1)$$

The elements in formula (1) are defined as follows:

$I = \{1, 2, \ldots, N\}$: The set of UAV agents, N is the current cluster size, which is allowed to change dynamically in different tasks; $S \subseteq \mathbb{R}^{N\times d_s}$: The global state space, which contains the complete state information of all agents. $s = [p_1, v_1, \ldots, p_N, v_N, p_{\text{goal}}, \{p_k^{\text{obs}}, v_k^{\text{obs}}\}_{k=1}^{M}] \in S$. p_{goal} is the center position of the target formation, and $\{p_k^{\text{obs}}, v_k^{\text{obs}}\}$ is the position and velocity of M obstacles; $O_i \subseteq S$: The local observation space of the i-th agent, which is limited by the perception radius $r_{\text{sense}} > 0$ and can only observe neighboring UAVs that meet $\|p_j - p_i\| \le r_{\text{sense}}$ and detectable obstacles. $A_i = [v_{\min}, v_{\max}]^2 \subseteq \mathbb{R}^2$: The continuous action space, where the i-th UAV performs the speed control action $a_i = (v_{ix}, v_{iy})^\top$, which is constrained by physical dynamics; $P(s'|s, a)$: The state transition function, which is dominated by the first-order integral dynamics model, and the obstacle state is updated according to the preset motion model; $r_i : O_i \times A_i \to \mathbb{R}$: Local reward function. $\gamma \in (0, 1)$: Discount factor, used to balance immediate and long-term rewards. The control goal is to learn a set of optimal distributed strategies $\pi^* = \{\pi_1^*, \pi_2^*, \ldots, \pi_N^*\}$, where $\pi_i : O_i \to A_i$, so that the cluster can complete the assembly from the initial distribution to the target formation configuration within a limited time, while minimizing the collision probability, cross-boundary risk and path deviation, and maximizing the mission success rate and system robustness. The optimization problem is expressed as:

$$\max_{\pi} \mathbb{E}_{o_i\sim\rho^{\pi}(\cdot),\, a_i\sim\pi_i(\cdot|o_i)} \left[\sum_{t=0}^{T} \gamma^t r_i^t\right], \quad \forall i \in I \qquad (2)$$

In Equation (2), $\rho_\pi(\cdot)$ is the state-observation distribution induced by policy π, and T is the maximum task step size. Because the environment is partially observable and dynamic, this problem exhibits highly nonstationary and multi-objective coupling characteristics. This problem is efficiently solved using a multi-agent reinforcement learning framework. The relative positional relationships between the drone, obstacles, and targets are shown in Figure 1.

The drone [28,29] senses its position relative to the target in real time to determine its direction and speed. Sensors detect the distribution, distance, and geometric characteristics of surrounding obstacles to construct a local environmental model. Obstacles between the drone and the target block the path. The dynamic relationship between these three factors requires the drone to approach the target along the optimal path while maintaining a safe distance.

B. OVERALL ARCHITECTURE OF THE LDE-MADDPG ALGORITHM

The overall architecture is shown in Figure 2.

The GAT encoding → feature embedding module takes the local observation features of each agent as input, and its dimension is determined by relative position and velocity information, specifically 4 dimensions. Neighborhood

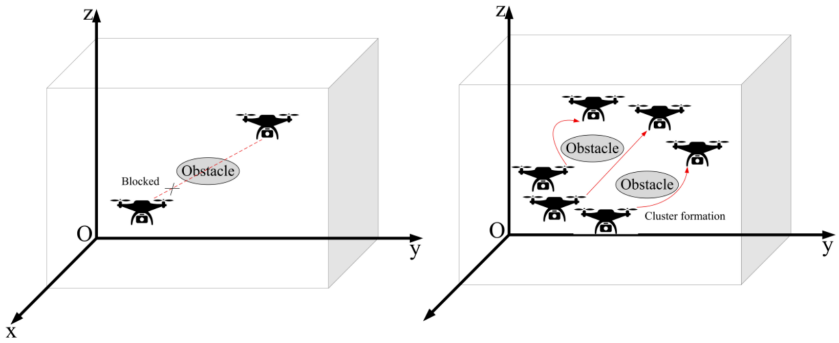

FIGURE 1. Relative positional relationships among drones, obstacles, and targets

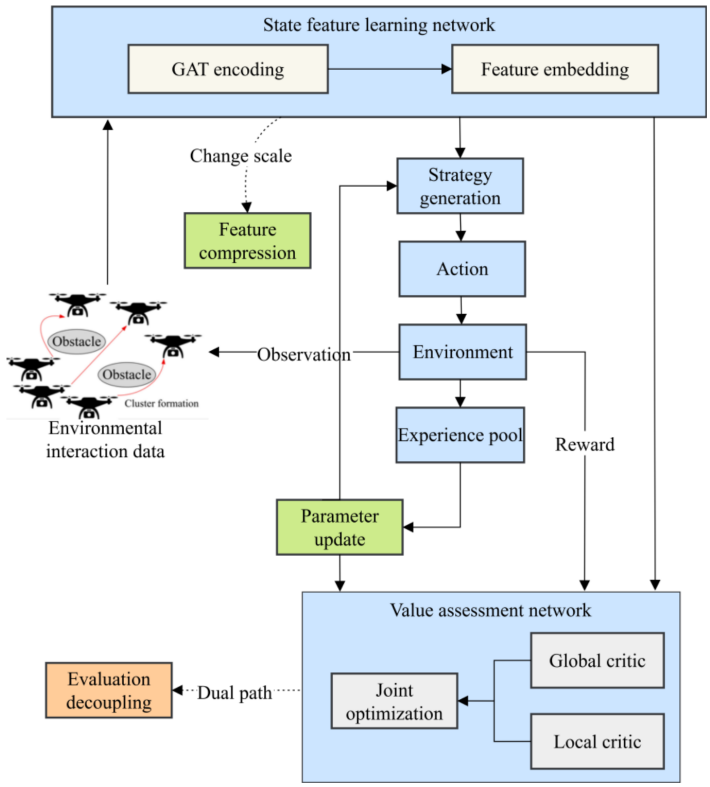

FIGURE 2. LDE-MADDPG algorithm framework

aggregation adopts a graph attention mechanism based on multi head attention to achieve weighted fusion of node features; All nodes share the same set of attention weights and parameter vectors, ensuring that the model has scale invariance and generalization ability, thereby supporting efficient embedding representation of variable scale clusters. This article constructs a decoupled dual path critic network to achieve the separation of global collaboration and local behavior evaluation. The global critic takes the joint state action embedding of all agents as input and learns a centralized value function, with the core objective of optimizing the overall formation performance and collaborative efficiency of the cluster; Local critics independently construct a value function for each agent, evaluating the value of local safety behaviors such as obstacle avoidance based solely on its own state embedding and actions. This structural decoupling achieves the refinement of credit allocation: global commentators directly guide the collaborative optimization of policies through policy gradients, while local commentators focus on improving individual response sensitivity and behavioral rationality to immediate risks. State Feature Learning Network: Graph Attention Network is used to encode variable-length neighborhood observations into fixed-dimensional embedding vectors, ensuring policy generalization across different cluster sizes [30,31]. Decoupled Dual-Path Critic Network: It separates global collaborative evaluation from local behavior value calculation, improving credit assignment accuracy. Actor Network [32,33]: It outputs continuous velocity actions based on local embedded features, enabling distributed execution.

Hierarchical Reward Function: It integrates macro-formation objectives with micro-safety constraints. Experience Replay and Target Network Mechanism: It ensures training stability. Let o_i be the local observation of the i-th agent. The state feature learning network extracts its neighborhood structural features through GAT and outputs a fixed-dimensional embedding $z_i \in \mathbb{R}^{d_z}$. The actor network executes the policy $\mu_i(z_i;\theta_i)$ and outputs an action $a_i = \mu_i(z_i;\theta_i) \in A_i$. The critic module adopts a two-branch structure. The global critic receives the embeddings and actions of all agents and evaluates the long-term reward of the joint policy; the local critic [34,35] evaluates the value of each individual's behavior based solely on individual information. The joint critic loss function is defined as:

$$L_{\text{critic}} = \mathbb{E}_{(s,a,r,s')\sim D}\Big[(y - Q^{\text{global}}(z_{1:N}, a_{1:N};\omega_g))^2 + \lambda \sum_{i=1}^{N}(y_i - Q_i^{\text{local}}(z_i, a_i;\omega_l))^2\Big]. \tag{3}$$

In formula (3), the target Q value is $y = r + \gamma Q_{\text{target}}(z'_{1:N}, a'_{1:N};\omega_g^-)$, y_i is the local target value of the i-th agent, $\lambda > 0$ is the balance coefficient, and ω_g^- represents the target network parameter. Actor policies are updated via deterministic policy gradients:

$$\nabla_{\theta_i} J \approx \mathbb{E}_{z_i\sim\rho^\mu}\Big[\nabla_{a_i} Q^{\text{global}}(z_{1:N}, a_{1:N}) \cdot \nabla_{\theta_i}\mu_i(z_i;\theta_i)\Big]. \tag{4}$$

During the training process, an experience replay pool is used to store transfer samples, and the target network parameters are maintained through a soft update mechanism:

$$\omega^- \leftarrow \tau\omega + (1-\tau)\omega^-, \quad \tau \ll 1 \tag{5}$$

C. STATE FEATURE LEARNING NETWORK

1) Local Observation Modeling

This paper designs a state feature learning network to map high-dimensional, variable-length local observations into compact, fixed-dimensional embeddings, eliminating the input dimensionality inconsistency caused by the dynamic number of neighbors. The local observation of the i-th drone at time t is represented as:

$$o_i = \left[p_i, v_i, \{p_j - p_i, v_j\}_{j\in N_i}, p_{\text{goal}} - p_i, \{p_k^{\text{obs}} - p_i\}_{k\in M_i}\right] \tag{6}$$

In formula (6), $p_i = [x_i, y_i]^\top \in \mathbb{R}^2$ and $v_i \in \mathbb{R}^2$ are the current position and velocity of drone i, respectively. The observation model is translationally invariant (based on relative coordinates) and partially observable, which aligns with the actual sensor input characteristics. Because the cardinality of N_i and M_i changes dynamically with the environment, directly inputting o_i into the neural network results in non-fixed input dimensions, making it difficult to adapt to batch training and policy generalization. The neighbor sets N_i and M_i are dynamic sets, and their update mechanism is based on the agent's perception radius r_{sense}. Specifically, at each time step, the unmanned aerial vehicle j that satisfies the condition $\|p_j - p_i\| \le r_{\text{sense}}$ is included in N_i, and the obstacle k that satisfies $\|p_k^{\text{obs}} - p_i\| \le r_{\text{sense}}$ is included in M_i. On the contrary, when neighboring individuals move out of the perception range, they are removed from the corresponding set, thus achieving dynamic updates of the neighborhood topology structure.

2) Graph Attention Encoding and Feature Embedding

This paper uses GAT to construct a state feature learning module, achieving efficient encoding of local neighborhood structure and fixed-dimensional feature embedding. This method not only has good scalability but also adaptively weights the contributions of different neighbors, improving the strategy's sensitivity to key neighboring individuals. The perception neighborhood of the i-th drone is modeled as an undirected graph $G_i = (V, E)$, where the node set $V = \{j \mid j \in N_i\}$ represents all of its communicable or perceptible neighboring drones, and the edge set $\epsilon \subseteq V \times V$ represents the connections between drones. Edge weights are implicitly determined by relative distance, while proximity is dynamically reflected in the attention mechanism. The initial feature vector of each node is defined as:

$$h_j^{(0)} = [p_j - p_i, v_j] \in \mathbb{R}^4 \tag{7}$$

Formula (7) includes the position offset and velocity information relative to agent i, ensuring that the input is translationally invariant and preserves the motion state. Feature propagation and aggregation are performed through an L-layer graph attention network. At layer l, the feature update for node j is:

$$h_j^{(l+1)} = \sigma\left(\sum_{k\in N(j)\cup\{j\}} \alpha_{jk}^{(l)} W^{(l)} h_k^{(l)}\right) \tag{8}$$

In formula (8), $W^{(l)}$ is a learnable linear transformation matrix, σ is a nonlinear activation function, and $N(j)$ is the set of neighbors of node j. The attention coefficient measures the influence of node k on node j and is calculated as:

$$\alpha_{jk} = \frac{\exp\left(\text{LeakyReLU}\left(a^\top\left[Wh_j^{(l)} \,//\, Wh_k^{(l)}\right]\right)\right)}{\sum_{k'\in N(j)\cup\{j\}} \exp\left(\text{LeakyReLU}\left(a^\top\left[Wh_j^{(l)} \,//\, Wh_{k'}^{(l)}\right]\right)\right)} \tag{9}$$

In formula (9), a is the learnable parameter vector of the attention mechanism. A global average pooling operation is used to generate a fixed-dimensional embedding vector:

$$z_i = \frac{1}{|N_i|}\sum_{j\in N_i} h_j^{(L)} \in \mathbb{R}^{d_z} \tag{10}$$

In formula (10), z_i encodes the overall structural characteristics and motion situation of the neighborhood.

D. DECOUPLED DUAL-PATH CRITIC NETWORK

1) Global Critic

The input to the global critic is the state embedding features and corresponding actions of all N agents, forming a joint observation representation $z_{1:N} = [z_1, \ldots, z_N]$ and a joint action $a_{1:N} = [a_1, \ldots, a_N]$. This joint input encodes the overall cluster topology, motion state, and control behavior, capturing the interactions between individuals and the group's collaborative tendencies. A multilayer perceptron is used to implement nonlinear mapping:

$$h^{(0)} = [z_{1:N}, a_{1:N}], \quad h^{(l+1)} = \sigma^{(l)}\left(W^{(l)}h^{(l)} + b^{(l)}\right) \tag{11}$$

$$Q^{\text{global}} = w_o^{\top} h^{(L)} + b_o \tag{12}$$

The learning goal is to minimize the timing difference error:

$$L_{\text{global}} = \mathbb{E}_{(s,a,r,s')\sim D}\Big[\big(r + \gamma Q^{\text{target}}(z'_{1:N}, a'_{1:N}; \omega_g^-) - Q^{\text{global}}(z_{1:N}, a_{1:N}; \omega_g)\big)^2\Big]. \tag{13}$$

In formula (13), r is the team shared reward and Q^{target} is the target network output.

2) Local Critic

The input to the local critic consists of two parts: the state feature embedding of the i-th agent, generated by the state feature learning network, which encodes its neighborhood structure and motion state; and the action $a_i = (v_{ix}, v_{iy})^{\top} \in A_i$ it performs. To further incorporate group context information to avoid completely isolated evaluation, a global context vector is also fused into the input:

$$\bar{z} = \frac{1}{N}\sum_{j=1}^{N} z_j \tag{14}$$

Formula (14) represents the average embedding state of the entire cluster, which is used to provide a reference for formation consistency. The network structure is implemented using a lightweight multi-layer perceptron:

$$h^{(0)} = [z_i, a_i, \bar{z}] \tag{15}$$

$$h^{(l+1)} = \sigma^{(l)}\left(W_l^{(l)}h^{(l)} + b_l^{(l)}\right) \tag{16}$$

$$Q_i^{\text{local}} = w_o^{\top} h^{(L_l)} + b_o \tag{17}$$

The learning goal is to minimize the temporal difference error at the individual level:

$$L_{\text{local},i} = \mathbb{E}_{(o_i,a_i,r_i,a'_i)\sim D}\Big[\big(r_i + \gamma Q_i^{\text{local,target}}(z'_i, a'_i; \omega_l^-) - Q_i^{\text{local}}(z_i, a_i; \omega_l)\big)^2\Big] \tag{18}$$

E. ACTOR NETWORK AND STRATEGY OPTIMIZATION

1) Actor Network Structure

The actor network is responsible for distributed execution, meaning each drone agent independently generates continuous control actions based on its local embedded features. The actor network for the i-th agent is implemented as a deterministic policy function $\mu_i(z_i; \theta_i) : \mathbb{R}^{d_z} \to A_i$, whose goal is to output the optimal velocity control vector to drive the drones toward the target formation and avoid obstacles in real time. The network input is a fixed-dimensional embedding vector extracted by a state feature learning network. This vector encodes the relative position and velocity information of neighboring drones and the local environment structure. It is translation-invariant and scale-independent, making it suitable for dynamically changing swarm topologies. The network architecture utilizes a multilayer, fully connected feedforward neural network, with each layer having dimensions of 256 and 128, respectively. The output layer is a two-dimensional vector representing the desired velocity component. To ensure that the output meets physical constraints, the final action is normalized using a tanh activation function and linearly mapped to the actual velocity range:

$$a_i = \mu_i(z_i; \theta_i) = v_{\text{scale}} \cdot \tanh\left(W_o h^{(L)} + b_o\right) + v_{\text{bias}} \tag{19}$$

In formula (19), $v_{\text{scale}} = (v_{\max} - v_{\min})/2$, $v_{\text{bias}} = (v_{\max} + v_{\min})/2$, $a_i \in [v_{\min}, v_{\max}]^2$ is guaranteed.

2) Policy Gradient Update

Actor network policy optimization follows the deterministic policy gradient theorem, maximizing the expected value of long-term cumulative rewards through gradient ascent. The policy objective of the i-th agent is defined as the expected reward under the initial state distribution:

$$J(\theta_i) = \mathbb{E}_{s\sim\rho^{\pi},\, a_{1:N}\sim\pi}\left[\sum_{t=0}^{T}\gamma^t r_t\right] \tag{20}$$

The gradient of the policy parameter θ_i can be expressed as:

$$\nabla_{\theta_i} J = \mathbb{E}_{s\sim\rho^{\pi}}\left[\nabla_{\theta_i}\mu_i(z_i;\theta_i)\cdot\nabla_{a_i}Q^{\pi}(s, a_{1:N})\big|_{a_i=\mu_i(z_i;\theta_i)}\right] \tag{21}$$

In formula (21), the gradient consists of two parts: the sensitivity of the policy network to the parameters $\nabla_{\theta_i}\mu_i$, and the gradient $\nabla_{a_i}Q^{\pi}$ of the Q function to the action in the action space, which reflects the direction of the impact of the current action adjustment on the long-term return. Since Q^{π} is unknown, the global critic network $Q^{\text{global}}(z_{1:N}, a_{1:N}; \omega_g)$ is used as an approximate estimate. The policy gradient update direction of the actor is:

$$\nabla_{\theta_i} J \approx \mathbb{E}_{z_{1:N}, a_{1:N} \sim D} \Bigg[\nabla_{\theta_i} \mu_i(z_i; \theta_i) \cdot \Bigg(\frac{\partial Q^{\text{global}}(z_{1:N}, a_{1:N}; \omega_g)}{\partial a_i} + \beta \cdot \frac{\partial Q_i^{\text{local}}(z_i, a_i; \omega_l)}{\partial a_i} \Bigg) \Bigg] \Bigg|_{a_i = \mu_i(z_i; \theta_i)} \tag{22}$$

In formula (22), the Actor's policy parameter θ_i is updated via a composite gradient that combines two complementary critic signals. The first term derives from the global critic Q^{global}, which guides the Actor toward macro-level formation coordination and cooperative task completion. The second term derives from the local critic Q_i^{local}, which introduces an individual-level safety gradient that directly penalizes actions driving the agent into high-risk proximity with obstacles. The weighting coefficient $\beta > 0$ controls the relative contribution of the local safety signal to the overall policy update direction, and is set to $\beta = 0.3$ based on pre-experimental grid search. The gradient is calculated through backpropagation and the Actor parameters are updated using gradient ascent:

$$\theta_i \leftarrow \theta_i + \alpha_\pi \nabla_{\theta_i} J \tag{23}$$

In formula (23), $\alpha_\pi > 0$ is the Actor learning rate.

F. HIERARCHICAL REWARD FUNCTION DESIGN AND TRAINING MECHANISM

To achieve multi-objective collaborative optimization of drone swarm formation and safe obstacle avoidance in a dynamic environment, this paper designs a hierarchical reward function that organically combines macro-mission objectives with micro-safety constraints. This guides the strategy to respond quickly to individual high-risk behaviors while ensuring overall coordination. The reward mechanism uses a weighted linear combination form, taking into account both interpretability and optimization efficiency. The formula is:

$$r_i = w_g r_{global} + w_c r_{collision,i} + w_o r_{obstacle,i} + w_b r_{boundary,i} + w_t r_{arrival,i} \tag{24}$$

In formula (24), r_i is the local reward obtained by the i-th agent at the current time step. r_{global}, $r_{collision,i}$, $r_{obstacle,i}$, $r_{boundary,i}$, and $r_{arrival,i}$ are the global formation reward, collision penalty, obstacle avoidance penalty, boundary violation penalty, and target reach reward, respectively. The reward weights are determined based on grid search and ablation experiments, with values of $w_g = 0.6$, $w_c = 0.8$, $w_o = 0.7$, $w_b = 0.3$, $w_t = 1.0$. This configuration balanced convergence speed and final performance in the pre experiment. The global formation reward measures the degree to which the entire cluster converges to the target configuration and is defined as the negative average formation error:

$$r_{global} = -\frac{1}{N} \sum_{j=1}^{N} \| p_j - p_j^{target} \| \tag{25}$$

In formula (25), p_j^{target} is the target position of the j-th UAV in the desired formation. This term serves as a macro-guidance signal, driving the coordinated movement of the entire swarm. The collision penalty is used to prevent collisions between UAVs and is given by:

$$r_{collision,i} = \begin{cases} -C_c, & \exists j \neq i, \ \| p_i - p_j \| < d_{safe}, \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

In formula (26), $d_{safe} = 2.0$ m is the safety distance threshold. The obstacle avoidance penalty applies a continuous penalty to the approach behavior of static and moving obstacles, and uses a Gaussian kernel function to enhance the response sensitivity:

$$r_{obstacle,i} = -C_o \sum_{k \in M_i} \exp\left(-\frac{\| p_i - p_k^{obs} \|^2}{2\sigma^2} \right) \tag{27}$$

In formula (27), σ controls the scope of the penalty, ensuring that evasive action is triggered in advance when an obstacle approaches. The out-of-bounds penalty limits the flight area to prevent the drone from flying out of the mission airspace.

The formula is:

$$r_{boundary,i} = \begin{cases} -C_b, & p_i \notin \Omega, \\ 0, & \text{otherwise.} \end{cases} \tag{28}$$

The goal arrival reward is to give positive incentives to individuals who successfully approach the target location:

$$r_{arrival,i} = \begin{cases} +R_t, & \| p_i - p_i^{target} \| < \epsilon, \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

ϵ is the tolerance threshold for the target's arrival position, set to ϵ = 0.3 m based on the task accuracy requirements and drone positioning error. When the distance between the drone and the target position is less than this threshold, it is judged as a successful arrival.

IV. EXPERIMENTAL SETUP

A. SIMULATION PLATFORM AND ENVIRONMENT CONFIGURATION

The simulation is run with a fixed time step of $\Delta t = 0.1$s, and the UAV dynamics model employs a first-order integrator. The environment includes both static and dynamic obstacles, supporting various formation configurations and target switching. This article builds a multi-agent simulation environment based on Python 3.8 and PyTorch 1.12 frameworks, and uses the Gymnasium API as the environment interface. This environment simulates the dynamics of a drone swarm in three-dimensional spaces. The simulation environment realizes functions such as obstacle interaction, local observation calculation, and distributed strategy execution, providing a standardized testing platform for algorithm training and evaluation. To ensure experimental

reproducibility and engineering rationality, all key environmental parameters are set based on the flight performance and perception capabilities of small multirotor UAVs and standardized according to mission requirements. The core parameters for the UAV formation simulation experiment are shown in Table 1.

TABLE 1. Core parameters of UAV formation simulation experiment

Parameter category	Parameter name	Value
Spatial settings	Maximum mission execution time	40 s
	Formation completion error threshold	0.3 m
	Operational size	0.25 m
	Minimum safety separation	0.5 m
Drone physical parameters	Minimum flight speed	0 m/s
	Maximum flight speed	10 m/s
	Initial speed	0 m/s
Perception and communication	Perception radius	10 m
	Communication delay	0 s
Obstacle settings	Static obstacle radius	5 m
	Dynamic obstacle radius	5 m
	Dynamic obstacle maximum speed	1.5 m/s
	Obstacle avoidance reward trigger distance	0.8 m

The experiment uses the Adam optimizer with a batch size of 512, a discount factor $\gamma = 0.95$, and a soft update coefficient $\tau = 0.01$. The hyperparameter information for the training of this algorithm is shown in Table 2.

TABLE 2. Training hyperparameters

Parameter	Value	Functions
Episodes	200	Controls the total number of rounds in the training process
Actor learning rate	1×10^{-4}	Controls the update step size of the policy network parameters
Global critic learning rate	1×10^{-3}	Adjusts the learning rate of the global value function
Local critic learning rate	1×10^{-3}	Ensures synchronized convergence of local value estimates
Discount factor	0.95	Weighs immediate rewards against future rewards
Batch size	512	Number of samples randomly drawn from the experience pool during each training session
Target network update rate	0.01	Controls the target network parameter update rate

B. UAV CLUSTERS AND FORMATION CONFIGURATIONS

Six typical formations are tested to verify the generalization capability, and the set assembly formations are shown in Figure 3.

This paper designs six typical formation configurations to validate the algorithm's generalization capabilities: triangles and diamonds are used to test its adaptability to symmetrical configurations; squares and rectangles are used to assess coordination accuracy under regular grid layouts; and a cross is simulated to test the extended structure used in detection and coverage missions. These formations, varying in size (8–12 aircraft) and geometric characteristics, comprehensively tested the algorithm's scalability and robust formation control performance in complex dynamic environments. This paper designs eight typical obstacle scenarios, including four static and four dynamic ones, covering a spatial area of $[0, 150] \times [0, 150] \times [0, 150]$ m^3. In all scenarios, the drone swarm starts from an initial position and converges towards the target, avoiding obstacles along the way. Scenarios 1–4 are static obstacle scenarios; Scenarios 5–8 apply moving obstacles based on their respective static scenarios. Motion patterns include uniform linear motion, simple harmonic oscillation, circular motion, and random walk. Each scenario includes an obstacle with a radius of 5 meters and a safe avoidance distance of 2.0 meters. The obstacle scenario settings are shown in Table 3.

C. MODEL COMPARISON

To verify the performance of LDE-MADDPG, this paper compares it with existing mainstream algorithms. The comparison model information is shown in Table 4.

The algorithms listed in Table 4 are widely used in collaborative control, robot swarms, and traffic scheduling, forming a comprehensive comparison system from basic to cutting-edge. By comparing these seven models, the comprehensive performance of LDE-MADDPG can be comprehensively evaluated in terms of scalability, obstacle avoidance response, and collaborative efficiency, and its advancedness and robustness can be verified in complex dynamic environments. All baseline algorithms are implemented under the same PyTorch code framework and trained on the same hardware platform (NVIDIA RTX 3080 GPU, 16GB RAM). To ensure fairness in comparison, all models share the following core training settings: training epochs (200 episodes), batch size (512), discount factor (0.95), experience replay buffer size (1e6), target network update rate (0.01), and use Adam optimizer. Algorithm specific hyperparameters, such as the Critic learning rate 1e-3 and Actor learning rate 1e-4 of MADDPG, are used to ensure that each algorithm participates in performance comparisons with its optimal configuration.

V. RESULTS

A. TRAINING PROCESS ANALYSIS RESULTS

In Scenario 7 Dynamic Obstacle, the target assembly formation is set to 9 "diamonds". The cumulative rewards per round for a single drone and a swarm are shown in Figure 4.

Figure 4(a) shows the cumulative rewards of nine drones over 200 training rounds, exhibiting a typical S-shaped reinforcement learning convergence curve. In the early stages (rounds 1-50), the reward fluctuation is between -6940 and -4000, with significant fluctuations, reflecting the high policy variance characteristic of the exploration phase. Rewards rise rapidly in the middle stages (50-150 rounds). In the

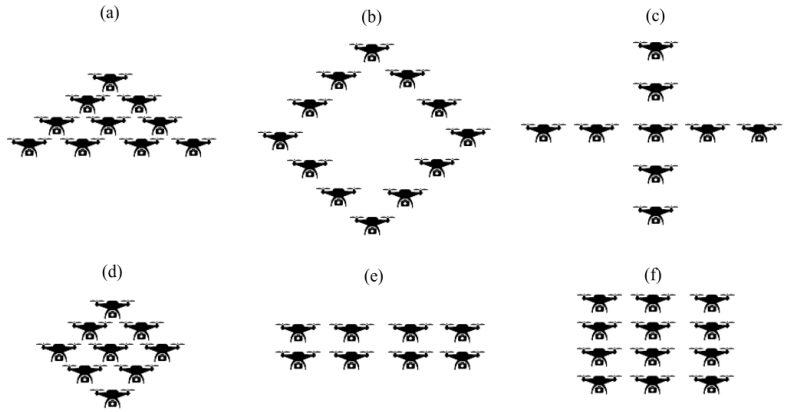

FIGURE 3. Assembly formations: (a) 10-person "triangle" formation; (b) 12-person "square" formation; (c) 9-person "cross" formation; (d) 9-person "diamond" formation; (e) 8-person "rectangle" formation; (f) 12-person "rectangle" formation

TABLE 3. Obstacle scene settings

Scenario	Type	Description
Scenario 1	Static	A single spherical obstacle is located at (30, 30, 60), blocking the area directly below, simulating a tall building or a no-fly column.
Scenario 2	Static	Dual obstacle channel structure: located at (30, 30, 60) and (50, 30, 60), with a spacing of 30 meters, forming a linear low-altitude channel.
Scenario 3	Static	Triangularly distributed three obstacles: located at (30, 30, 60), (50, 30, 60), and (30, 50, 60).
Scenario 4	Static	Circular static obstacle group: four obstacles are evenly distributed on a horizontal circle with a radius of 15 meters centered at (30, 30, 60), simulating a circular obstacle structure.
Scenario 5	Dynamic	Based on Scenario 1, the obstacle moves in a straight line at a constant speed of $v = (1.2, 0.8, 0.0)$ m/s, traversing the area along the $z = 60$ meter plane, simulating a moving vehicle or floating platform.
Scenario 6	Dynamic	Based on Scenario 3, the right obstacle performs simple harmonic oscillation along the z-axis.
Scenario 7	Dynamic	Based on Scenario 3, the obstacle located at (50, 30, 60) undergoes harmonic vibration along the Z-axis. $z(t) = 60 + 8\sin(0.2\pi t)$, with a range of $z \in [52, 68]$ m.
Scenario 8	Dynamic	Based on Scenario 3, the central obstacle walks randomly on the horizontal plane at a maximum speed of 1.2 m/s, and its direction of motion changes randomly with a probability of 0.5 at each time step, simulating an irregular moving interference target.

TABLE 4. Comparison model information

Algorithm	Features	Advantages
MADDPG	Centralized critic evaluation of joint actions, distributed actor execution	Supports continuous action spaces, laying the foundation for multi-agent collaborative control
MAPPO	Combining PPO with a centralized value function, using policy gradient optimization	Stable training and good convergence, suitable for high-dimensional collaborative tasks
QMIX	Value function decomposition method, integrating individual Q-values via a monotonic network	Effectively solves the credit assignment problem, suitable for discrete action collaboration
COMA (Counterfactual Multi-Agent Policy Gradients)	Applying a counterfactual baseline to evaluate the contribution of individual actions to the overall reward	Improves local behavior sensitivity, improving credit assignment accuracy
TarMAC (Targeted Multi-Agent Communication)	Enabling dynamic communication based on a soft attention mechanism	Enables interpretable information exchange, adapting to partially observable environments
GAT-MADDPG	Integrating a graph attention network for neighbor feature aggregation	Supports variable-scale input, improving policy generalization
ROMA (Role-Oriented Multi-Agent Reinforcement Learning)	Applying a role discovery mechanism to implicitly learn agent functional roles	Enables behavioral specialization, improving large-scale collaborative efficiency

later stages (150-200 rounds), rewards reach a stable convergence phase, approaching stability with minimal individual variation. The cluster reward curve graph 4 (b) reveals the collaborative dynamics of multi-agent systems: the cumulative reward continues to increase in the first 93 rounds and turns positive around the 94th round. This phenomenon marks a critical turning point in the strategic performance of the intelligent agent cluster: at this stage, the positive benefits obtained through exploration by the strategy begin to systematically surpass the cumulative punishment it bears due to exploration behavior. This reflects the natural learning process of the algorithm transitioning from the initial stage of exploration with negative rewards to the mature stage of effectively utilizing learned knowledge and obtaining net

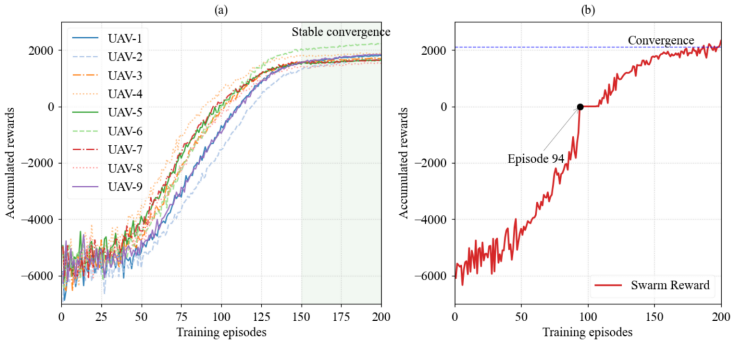

FIGURE 4. Cumulative rewards per round for a single drone and a swarm: (a) Single drone rewards; (b) Swarm rewards

positive returns. The emergence of this transition window validates the effectiveness of GAT feature embedding and dual path critic architecture in promoting strategy stable co evolution.

B. TRAJECTORY VISUALIZATION ANALYSIS

In the static obstacle environment of Scenario 3, the target assembly formation is a "diamond" formation of 9 sorties, and the trajectory visualization is shown in Figure 5.

Figure 5 shows the complete trajectory of a drone swarm starting from the starting point (0,0,0) (indicated by black dots in the figure) and finally assembling into a designated 'diamond' formation at the target position (marked by a pentagram) in the environment of Scene 3. Obstacles in the environment are indicated by light brown spheres. The trajectory visualization is clear and all drones can smoothly navigate around obstacles during flight. Specifically, when the drone approaches obstacles, its trajectory shows a clear circular arc. Individuals located on the flank of the formation can safely pass from the outside of obstacles by making early turns and adjusting their speed.

C. ABLATION EXPERIMENT ANALYSIS

This paper conducts an ablation experiment on reward items. While retaining the global formation reward, it removes the collision penalty, obstacle avoidance penalty, boundary violation penalty, and target arrival reward. The ablation information is shown in Table 5.

Note: The reward mechanism used in this model is Mechanism A. Figure 6 shows the dynamic obstacle avoidance performance of drone swarms using different reward mechanisms.

1. Obstacle avoidance response time is defined as the interval between the moment an obstacle enters the sensing radius and the first avoidance maneuver (speed and direction change $> 30°$). 2. Data are the mean $\pm$ standard deviation of 20 independent runs. 3. Mission completion rate is defined as the percentage of drones completing formation assembly within the specified time (40 seconds) without any collisions. Obstacle avoidance response time results show that the gradual removal of key local rewards delays the swarm's obstacle avoidance response in various scenarios. Under the full reward mechanism (A), the average response time remains between 0.32 and 0.42 seconds, demonstrating good real-time performance and agility. When the collision penalty is removed (B), the response time increases slightly, indicating that the reward has a certain restraining effect on individual safety behavior. Further removing the obstacle avoidance penalty increases the average response time to 0.68 seconds or more, indicating that the obstacle avoidance penalty is the core driving signal for proactive avoidance maneuvers. This phenomenon validates the effectiveness of our proposed layered reward design: global rewards cannot replace the guiding role of local safety factors in emergency behavior. Furthermore, while the absence of out-of-bounds penalties has a minor impact, their combined effect further deteriorates response performance, suggesting a synergistic enhancement effect among the various micro-constraints. The overall trend suggests that the lack of a local penalty mechanism can lead to a policy that is less sensitive to potential threats, severely weakening the system's robustness in highly dynamic environments. Analysis of task completion rates further reveals the contribution of each reward to overall collaborative performance. Mechanism A, in its entirety, achieves a high completion rate exceeding 92% in all scenarios, demonstrating the proposed reward system's robust task-enabling capabilities. Mechanism B's lack of a collision penalty leads to increased inter-individual conflict. Mechanism C, however, shows a sharp drop in performance after removing the obstacle avoidance penalty, reaching as low as 58.7% in complex dynamic scenarios, demonstrating the crucial importance of obstacle avoidance rewards for successful traversal of obstacles. Due to the persistent lack of multiple constraints, the D and E mechanisms tend to adopt aggressive or chaotic strategies, leading to mission failures often caused by collisions or deviations from the target area. Notably, the E mechanism, which only retains the global strategy, performs the worst, demonstrating that relying solely on macro-level guidance cannot guarantee safety and feasibility. Experimental results strongly confirm that local rewards are not auxiliary factors but rather a critical component to mission success. Their coordinated optimization with global rewards is the core mechanism for achieving efficient and safe formation control.

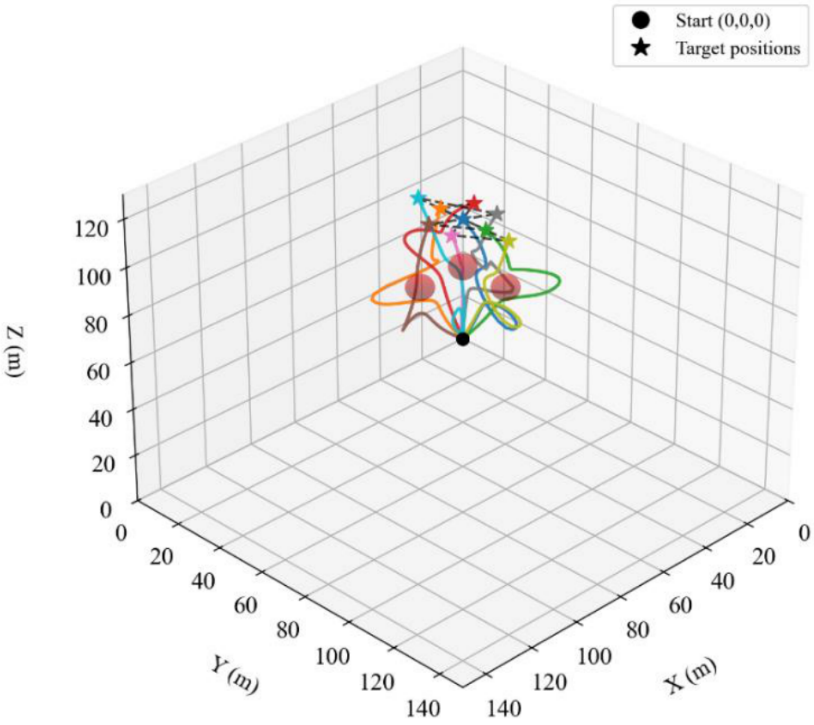

FIGURE 5. Trajectory visualization

TABLE 5. Ablation information

Reward mechanism	Global formation reward	Obstacle avoidance penalty	Boundary exit penalty	Target reach reward	Collision penalty
A	√	√	√	√	√
B	√	×	√	√	√
C	√	×	×	√	√
D	√	×	×	×	√
E	√	×	×	×	×

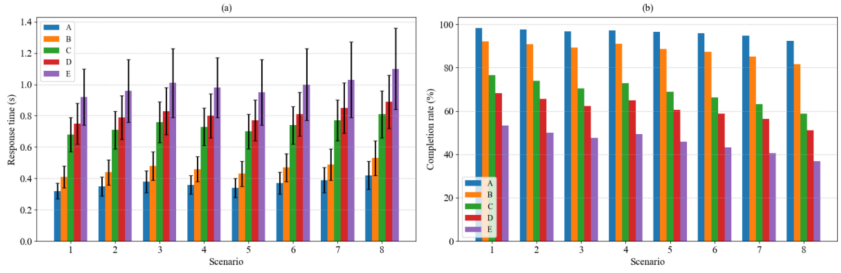

FIGURE 6. Performance of Different Reward Mechanisms: (a) Obstacle Avoidance Response Time (s); (b) Task Completion Rate (%)

D. PERFORMANCE COMPARISON OF MULTIPLE ALGORITHMS

In eight scenarios, including static obstacle scenarios (Scenarios 1-4) and dynamic obstacle scenarios (Scenarios 5-8), our algorithm is compared with MADDPG, MAPPO, QMIX, COMA, TarMAC, GAT-MADDPG, and ROMA. The collision rate results are shown in Figure 7.

In static obstacle scenarios, LDE-MADDPG achieves a lower collision rate (2.1%-4.2%) than other algorithms, particularly traditional MADDPG (12.7%-18.6%) and QMIX (14.5%-20.3%). This advantage is primarily attributed to its decoupled dual-path critic architecture: a global critic optimizes formation coordination through GAT feature embedding, while local critics independently evaluate obstacle avoidance behavior, avoiding the collision risk associated with ambiguous credit assignment in traditional methods. Despite incorporating a graph attention network, GAT-MADDPG (5.6%-8.9%) still lags behind due to its lack

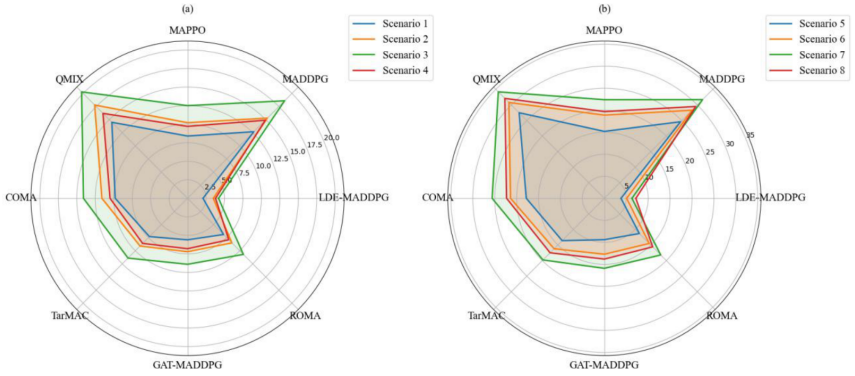

FIGURE 7. Collision rate of multiple algorithm performance (%): (a) Static failure scenario; (b) Dynamic failure scenario

of separation between global and local evaluation. Notably, TarMAC (7.3%-11.4%) improves obstacle avoidance coordination through communication mechanisms, but its reliance on fixed-dimensional state inputs makes it less adaptable in complex static layouts. Data shows that the performance differences among algorithms in static environments are primarily determined by their state representation capabilities (GAT embedding) and their evaluation mechanism (dual-path critic), while LDE-MADDPG achieves an optimal balance by combining these two. Dynamic scenarios further magnify the performance gap between algorithms. LDE-MADDPG maintains the lowest collision rate (3.8%-7.2%), thanks to its hierarchical reward design: obstacle avoidance penalties trigger avoidance actions early via a Gaussian kernel, while global formation rewards prevent overly conservative paths. Comparatively, TarMAC and ROMA algorithms perform poorly in dynamic environments because they do not explicitly model obstacle motion trends. QMIX suffers from poor adaptability in continuous action spaces due to the monotonicity constraints of its value function decomposition, resulting in the highest collision rate (27.5%-34.2%). The core challenge in dynamic scenarios lies in the trade-off between real-time response and long-term coordination. The performance advantage of LDE-MADDPG comes from the synergistic effect of GAT embedding, layered reward, and dual path critic. Among them, the dual path structure enhances obstacle avoidance behavior through the specialized learning of safety rewards by local commentators, enabling the global value function to more accurately reflect obstacle avoidance needs, thereby indirectly strengthening obstacle avoidance behavior during policy updates. This mechanism, together with GAT's state encoding capability and the multi-objective orientation of hierarchical rewards, forms the basis for improving algorithm performance.

E. RANDOM SCENARIO GENERALIZATION ABILITY TEST

It randomly generates 100 dynamic scenarios in the space between the starting point and the target formation. It evaluates the generalization ability of random scenarios using the flight time, energy consumption, and mission completion rate of the drone swarm. The results are shown in Figure 8.

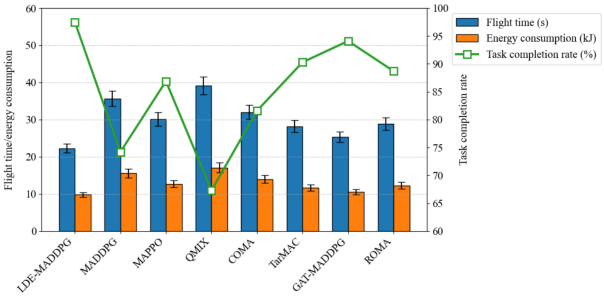

FIGURE 8. Generalization ability of random scenes

LDE-MADDPG achieves superior performance with lower flight time (22.3±1.2s) and energy consumption (9.8±0.6kJ) while maintaining 97.5% mission completion. This stems from three key innovations: GAT embedding enabling generalization to random obstacles, dual-path critics reducing redundant maneuvers through local behavior evaluation, and hierarchical rewards accelerating safety responses. Comparative results show QMIX suffers from prolonged flight time (39.2s) due to value decomposition constraints, while TarMAC (90.3% completion) exhibits higher energy consumption without decoupled evaluation. Although GAT-MADDPG approaches in efficiency metrics (25.4s, 10.5kJ), its 3.4% lower completion rate confirms that enhanced state representation alone cannot resolve credit assignment challenges. These results collectively validate LDE-MADDPG's comprehensive advantage in dynamic scenarios.

VI. CONCLUSION

This study presents LDE-MADDPG to overcome scalability and responsiveness limitations in drone swarm obstacle avoidance. The algorithm integrates three key components: GAT-based feature embedding for processing variable-scale observations, a dual-path critic separating global and local evaluation, and a hierarchical reward scheme. Experimental validation demonstrates collision rates of 2.1%-4.2% (static) and 3.8%-7.2% (dynamic), with 97.5% mission completion

in random scenarios. The fusion of GAT embedding with dual-path critics enables effective modeling of variable-scale swarms, advancing robust multi-agent coordination in dynamic environments.

AUTHOR CONTRIBUTIONS

Conceptualization – Xianghua Fang, Ken Chen, Chenghao Ren, HongChuan Jiang and Bing Li; methodology – Xianghua Fang, Ken Chen, Chenghao Ren, HongChuan Jiang and Bing Li; investigation – Xianghua Fang, Ken Chen, Chenghao Ren, HongChuan Jiang and Bing Li; writing-original draft preparation – Xianghua Fang, Ken Chen, Chenghao Ren, HongChuan Jiang and Bing Li. All authors have read and agreed to the published version of the manuscript.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

FUNDING

This research was supported by the City key research and development project forest "two situation two resources" UAV digital platform system development, Lishui City Science and Technology Bureau, Zhejiang Province 2023zdyf02.

ACKNOWLEDGEMENTS

Not applicable.

REFERENCES

[1] G. Kumar, A. Anwar, A. Dikshit, A. Poddar, U. Soni, and W. K. Song, "Obstacle avoidance for a swarm of unmanned aerial vehicles operating on particle swarm optimization: a swarm intelligence approach for search and rescue missions," Journal of the Brazilian Society of Mechanical Sciences and Engineering, vol. 44, no. 2, Art. no. 56, 2022, doi: 10.1007/s40430-022-03362-9.

[2] D. Marek, P. Biernacki, J. Szyguła, M. Paszkuta, and M. Szczygiel, "Collision avoidance mechanism for swarms of drones," Sensors, vol. 25, no. 4, pp. 1141-1141, 2025, doi: 10.3390/s25041141.

[3] L. Zhao, B. Chen, and F. Hu, "Research on cooperative obstacle avoidance decision making of unmanned aerial vehicle swarms in complex environments under end-edge-cloud collaboration model," Drones, vol. 8, no. 9, pp. 461-461, 2024, doi: 10.3390/drones8090461.

[4] R. A. Saeed, M. Omri, S. Abdel-Khalek, E. S. Ali, and M. F. Alotaibi, "Optimal path planning for drones based on swarm intelligence algorithm," Neural Computing and Applications, vol. 34, no. 12, pp. 10133-10155, 2022, doi: 10.1007/s00521-022-06998-9.

[5] G. Jia and J. Wang, "A review of research methods for UAV swarm mission planning," Systems Engineering & Electronics, vol. 43, no. 1, pp. 99-99, 2021, doi: 10.3969/j.issn.1001-506x.2021.01.13.

[6] X. Fu and J. Pan, "Distributed formation control of UAV swarm to avoid dynamic obstacles," Systems Engineering & Electronics, vol. 44, no. 2, pp. 529-529, 2022, doi: 10.12305/j.issn.1001506x.2022.02.22.

[7] D. Yan, W. Zhang, H. Chen, and J. Shi, "Research on sliding mode consensus formation control of multi-UAV with time delay and interference constraints," Journal of Northwestern Polytechnical University, vol. 38, no. 2, pp. 420-426, 2020, doi: 10.1051 /jnwpu/20203820420.

[8] Y. Wu and T. Liang, "UAV formation control based on improved consensus algorithm," Acta Aeronautica Sinica, vol. 41, no. 9, pp. 323848-323848, 2020, doi: 10.7527/S10006893.2020.23848.

[9] Z. Xue and T. Gonsalves, "Vision based drone obstacle avoidance by deep reinforcement learning," Ai, vol. 2, no. 3, pp. 366-380, 2021, doi: 10.3390/ai2030023.

[10] A. Novikov, S. Yakovlev, and I. Gushchin, "Exploring the possibilities of MADDPG for UAV swarm control by simulating in Pac-Man environment," Radioelectronic and Computer Systems, vol. 2025, no. 1, pp. 327-337, 2025, doi: 10.32620/reks.2025.1.21.

[11] J. Liu, S. Wei, B. Li, T. Wang, W. Qi, X. Han, et al., "Dual-timescale hierarchical MADDPG for Multi-UAV cooperative search," Journal of King Saud University Computer and Information Sciences, vol. 37, no. 6, pp. 1-17, 2025, doi: 10.1007/s44443-025-00156-6.

[12] E. Zhao, N. Zhou, C. Liu, H. Su, Y. Liu, J. Cong, et al., "Time-aware MADDPG with LSTM for multi-agent obstacle avoidance: A comparative study," Complex & Intelligent Systems, vol. 10, no. 3, pp. 4141-4155, 2024, doi: 10.1007/s40747-024-01389-0.

[13] J. Li, Z. Yan, K. Yan, Y. Zhao, R. Tan, C. Liang, et al., "UAV ground target detection algorithm based on attention and channel rearrangement," Journal of Ordnance Equipment Engineering, vol. 45, no. 3, pp. 306-306, 2024, doi: 10.11809/bqzbgcxb2024.03.040.

[14] X. He, X. Shi, J. Hu, and Y. Wang, "Multi-robot navigation with graph attention neural network and hierarchical motion planning," Journal of Intelligent & Robotic Systems, vol. 109, no. 2, pp. 25-25, 2023, doi: 10.1007/S10846-023-01959-3.

[15] Q. Wang, D. Zhuang, and H. Xie, "Identification of influential nodes for drone swarm based on graph neural networks," Neural Processing Letters, vol. 53, no. 6, pp. 4073-4096, 2021, doi: 10.1007/S11063-021-10583-X.

[16] X. Zhang, J. Zheng, T. Su, H. Liu, and Q. Gao, "Planning method for cooperative search and tracking mission of UAV swarm," Radar Science and Technology, vol. 20, no. 5, pp. 480-491, 2022, doi: 10.3969/j.issn.1672-2337.2022.05.002.

[17] H. Dong, J. Yang, S. Li, J. Wang, and Z. Duan, "Research progress of robot motion control based on deep reinforcement learning," Control and Decision, vol. 37, no. 2, pp. 278-292, 2022, doi: 10.13195/j.kzyjc.2020.1382.

[18] Y. Jia, Y. Song, B. Xiong, J. Cheng, W. Zhang, S. Yang, et al., "Hierarchical perception-improving for decentralized multi-robot motion planning in complex scenarios," IEEE Transactions on Intelligent Transportation Systems, vol. 25, no. 7, pp. 6486-6500, 2024, doi: 10.1109/TITS.2023.3344518.

[19] W. Wang, Y. Chen, Y. Zhang, Y. Chen, and Y. Du, "Collaborative Search Algorithm for Multi-UAVs Under Interference Conditions: A Multi-Agent Deep Reinforcement Learning Approach," Drones, vol. 9, no. 6, pp. 445-445, 2025, doi: 10.3390/drones9060445.

[20] D. Wei, L. Zhang, Q. Liu, H. Chen, and J. Huang, "UAV Swarm Cooperative Dynamic Target Search: A MAPPO-Based Discrete Optimal Control Method," Drones, vol. 8, no. 6, pp. 214-214, 2024, doi: 10.3390/drones8060214.

[21] P. Zhang and G. Li, "A cooperative dynamic target search approach for multi-UAV systems utilizing the MAPPO algorithm," Discover Artificial Intelligence, vol. 5, no. 1, pp. 153-153, 2025, doi: 10.1007/S44163-025-00411-9.

[22] C. Ning, J. Fan, and S. Sun, "A review of research on multi-UAV collaborative planning," Journal of Computer Engineering & Applications, vol. 61, no. 1, pp. 42-42, 2025, doi: 10.3778/j.issn.1002-8331.2405-0110.

[23] Y. Sun, C. Yan, X. Xiang, D. Tang, H. Zhou, J. Jiang, et al., "Multi-UAV cooperative capture method based on hierarchical reinforcement learning," Control Theory & Applications/Kongzhi Lilun Yu Yinyong, vol. 42, no. 1, pp. 96-96, 2025, doi: 10.7641/CTA.2024.30439.

[24] J. Wu, C. Luo, Y. Luo, and K. Li, "Distributed UAV swarm formation and collision avoidance strategies over fixed and switching topologies," IEEE transactions on cybernetics, vol. 52, no. 10, pp. 10969-10979, 2021, doi: 10.1109/TCYB.2021.3132587.

[25] S. Argiliana, E. Ekawati, and F. Mukhlish, "Adaptive Strategies for Dynamic Obstacle Avoidance and Formation Control in Multi-Agent Drone Systems: A Review," Journal of Robotics and Control (JRC), vol. 6, no. 4, pp. 1710-1720, 2025, doi: 10.18196/jrc.v6i4.26243.

[26] R. Fan, J. Wang, W. Han, and B. Xu, "UAV swarm control based on hybrid bionic swarm intelligence," Guidance, Navigation and Control, vol. 3, no. 02, pp. 2350008-2350008, 2023, doi: 10.1142/S2737480723500085.

[27] H. Muller, V. Niculescu, and T. Polonelli, "Robust and efficient depth-based obstacle avoidance for autonomous miniaturized uavs," IEEE Transactions on Robotics, vol. 39, no. 6, pp. 4935-4951, 2023, doi: 10.1109/TRO.2023.3315710.

[28] M. H. Harun, S. S. Abdullah, M. S. M. Aras, and M. B. Bahar, "Collision avoidance control for Unmanned Autonomous Vehicles (UAV): Recent advancements and future prospects," Indian Journal of Geo-Marine Sciences (IJMS), vol. 50, no. 11, pp. 873-883, 2022.

[29] C. C. Ekechi, T. Elfouly, A. Alouani, and T. Khattab, "A Survey on UAV Control with Multi-Agent Reinforcement Learning," Drones, vol. 9, no. 7, pp. 484-484, 2025, doi: 10.3390/drones9070484.

[30] X. Wei, W. Cui, X. Huang, L. Yang, X. Geng, Z. Tao, et al., "Hierarchical RNNs with graph policy and attention for drone swarm," Journal of Computational Design and Engineering, vol. 11, no. 2, pp. 314-326, 2024, doi: 10.1093/jcde/qwae031.

[31] M. M. Alam, S. A. Trina, T. Hossain, M. S. Ahmed, and M. Y. Arafat, "Variations in Multi-Agent Actor–Critic Frameworks for Joint Optimizations in UAV Swarm Networks: Recent Evolution, Challenges, and Directions," Drones, vol. 9, no. 2, pp. 153-153, 2025, doi: 10.3390/drones9020153.

[32] R. Tang, J. Tang, M. S. A. Talip, N. K. Aridas, and X. Xu, "Enhanced multi agent coordination algorithm for drone swarm patrolling in durian orchards," Scientific Reports, vol. 15, no. 1, pp. 9139-9139, 2025, doi: 10.1038/S41598-025-88145-7.

[33] P. Cao, L. Lei, S. Cai, G. Shen, X. Liu, X. Wang, et al., "Computational intelligence algorithms for UAV swarm networking and collaboration: A comprehensive survey and future directions," IEEE Communications Surveys & Tutorials, vol. 26, no. 4, pp. 2684-2728, 2024, doi: 10.1109/COMST.2024.3395358.

[34] S. A. Mustafa and A. S. M. Kakshar, "Analysis and Prospect of Existing Path Planning Algorithms for Multi-Drone Systems," QALAAI ZANIST SCIENTIFIC JOURNAL, vol. 10, no. 1, pp. 1508-1543, 2025.