2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Event-Driven MARL for Collaborative Swarm Confrontation in Asynchronous Environments

Qizhen Wu, Lei Chen, Kexin Liu, and Jinhu Lü

Abstract— **Multi-agent reinforcement learning (MARL) provides a flexible solution for tackling task and motion planning challenges, particularly in swarm confrontation scenarios. By customizing termination conditions for diverse tasks, event-driven MARL reduces decision jitter caused by frequent task switching. However, it hinders robots from updating strategies on a consistent timescale, leading to misaligned information sharing that disrupts agent coordination. To address this, we propose a novel event-driven MARL approach that facilitates collaborative strategy learning under asynchronous conditions. The approach introduces an experience selection scheme tailored to diverse timescales, ensuring efficient training through synchronized information sharing among robots. By incorporating Transformers, our method enables robots to infer others' behaviors from historical data, optimizing collaborative strategies. Extensive experiments validate the effectiveness of our proposed approach.**

I. Introduction

Advances in artificial intelligence significantly propel robotics forward, particularly in the domain of robotic swarm confrontation [1]–[3]. In these transient scenarios, the robotic swarm must swiftly counter dynamic threats from adversaries. Traditional approaches, such as expert systems [1], game theory [4], and heuristic approaches [5], excel in small-scale settings but struggle with escalating computational complexity as swarm sizes grow. This highlights an urgent need for flexible, adaptive solutions that can dynamically adjust strategies in rapidly evolving battlefield environments.

Multi-agent reinforcement learning (MARL) [6], [7] introduces a robust, adaptive framework for enhancing swarm confrontation performance. By autonomously deriving sophisticated tactics from vast datasets, MARL outperforms manually designed strategies, achieving remarkable success in complex scenarios such as pursuit-evasion [8], attack-defense [9], and strategic engagements [10]. Swarm confrontation, as a classic task and motion planning challenge, operates within a hybrid decision-making space, blending discrete commands with continuous actions in unpredictable settings. For instance, battlefield commands take the form of discrete decisions, while robotic movements occur in continuous time. This interplay aligns naturally with a divide-and-conquer paradigm, where high-level commands guide task allocation and low-level actions enable precise path planning [1]. As a pioneering approach, [11] introduces a hierarchical MARL framework for swarm confrontation, while [12] decomposes multi-aircraft air combat into strategic and tactical layers, achieving notable success in small-scale robotic engagements. Robots trained via MARL initially rely on random exploration [13], which may trigger task switching at any moment. Different tasks demand distinct navigation goals—for instance, pursuit requires closing the distance to a target enemy, while evasion entails maximizing distance from nearby threats. Thus, frequent task switching often leads to indecision and erratic planning, ultimately resulting in mission failure during confrontations. By designing tailored termination conditions for distinct tasks, event-driven RL [14], [15] mitigates decision jitter, enabling robots to switch tasks only when these conditions are satisfied. However, extending this approach to multi-agent scenarios poses challenges, as agents must synchronize strategy updates across diverse timescales [16].

To tackle the asynchronous challenges posed by event-driven mechanisms in MARL, [17] employs a fully decentralized approach, but it exhibits slow convergence and struggles to achieve coordinated strategies among agents. Conversely, the fully centralized method [18] faces scalability issues due to dimensionality explosion in large-scale scenarios. To overcome these limitations, the asynchronous centralized training with decentralized execution (CTDE) paradigm [19], [20] enables each robot to select tasks based on its own information while promoting collaborative behaviors through shared experience data. This method utilizes historical trajectories, rather than local observations, for training and decision-making to address challenges posed by asynchronous data. Additionally, [21] proposes trajectory collection techniques that minimize redundant training data, reducing the computational burden on critic networks processing historical information. In confrontation scenarios with rapidly evolving conditions, a robot's historical data often exhibits high complexity [9]. Conventional CTDE approaches, which rely on multilayer perceptron (MLP) structures [22], struggle to capture intricate temporal correlations, presenting significant challenges to effective decision-making. By leveraging the Transformer's self-attention mechanism and parallel processing capabilities, [23] empowers robots to

*This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFB3305600, and in part by the National Natural Science Foundation of China under Grants 62141604, 62088101, and 62003015. (*Corresponding author: Lei Chen.*)

Qizhen Wu, Kexin Liu, and Jinhu Lü are with the School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China. (e-mail: wuqzh7@buaa.edu.cn; skxliu@163.com; jhlu@iss.ac.cn)

Lei Chen is with the Advanced Research Institute of Multidisciplinary Sciences and State Key Laboratory of CNS/ATM, Beijing Institute of Technology, Beijing 100081, China. (e-mail: bit_chen@bit.edu.cn)

The experiment video is available at https://www.bilibili.com/video/BV1RHeUzwE7L/?vd_source=9de61aecdd9fb684e546d032ef7fe7bf.

The code is available at https://github.com/Wu-duanduan/Asynchronous-Swarm-Confrontation.

adeptly manage complex, temporally correlated data. As a result, a growing body of research is advancing MARL by incorporating Transformers [24], [25], significantly enhancing robots' reasoning and decision-making capabilities.

Here, we propose a novel MARL approach that leverages Transformer techniques to enable collaborative strategy learning using asynchronous information in swarm confrontation scenarios. Our method implements an event-driven mechanism to regulate task termination, ensuring task continuity and preventing arbitrary switching. To address misaligned information sharing among diverse agents, we introduce an experience selection scheme that filters effective joint information, optimizing robot training efficiency. Furthermore, by integrating Transformer techniques, our approach enhances the critics and actors of QMIX [26] to process complex historical confrontation data, significantly improving the collaborative capabilities of robotic swarms. Experimental results demonstrate that our approach outperforms conventional MARL methods, particularly in large-scale scenarios. Ablation studies underscore the critical role of our method's key components. With sufficient training, our approach fosters cooperative strategies in confrontations.

We organize the rest of the paper as follows. Section II formulates the problem and introduces the preliminaries. Section III details the improved MARL method and its implementation. Section IV provides the experimental evaluation of our approach. Section V concludes the paper.

II. Problem Formulation and Preliminaries

A. Model Description

This study models swarm confrontation as a dynamic strategic engagement within a planar environment containing static obstacles. A group of homogeneous robots, each possessing identical capabilities, collaborates to counter opposing forces, represented as blue and red teams. The system is formulated using second-order differential drive robots navigating in the Euclidean plane. Let $\mathbb{R}$ and $\mathbb{Z}$ denote the sets of real and integer numbers, respectively. For each robot, a body-fixed frame is defined, with its x-axis aligned with the robot's forward direction. The parameters X_f and X_b denote the values of a variable X in the fixed inertial frame and the robot's body frame, respectively. The i-th agent is controlled by two inputs: the angular velocity of the heading $\omega_{b,i}$, and the linear velocity $\boldsymbol{v}_{b,i} = [v_{b_x,i}, v_{b_y,i}]^\top$, defined along the x- and y-axes of the body frame. The dynamics of each robot are governed by

$$\boldsymbol{p}_i[k+1] = \boldsymbol{p}_i[k] + \boldsymbol{v}_{f,i}[k]\Delta T, \tag{1}$$

$$\boldsymbol{v}_{f,i}[k+1] = \boldsymbol{v}_{f,i}[k] + \boldsymbol{a}_{f,i}[k]\Delta T, \tag{2}$$

$$\boldsymbol{R}_i[k+1] = \boldsymbol{R}_i[k] + \boldsymbol{R}_i[k]\boldsymbol{\Omega}_{b,i}[k]\Delta T, \tag{3}$$

$$\omega_{b,i}[k+1] = \omega_{b,i}[k] + \alpha_{b,i}[k]\Delta T, \tag{4}$$

where $\boldsymbol{p}_i[k] \in \mathbb{R}^2$, $\boldsymbol{v}_i[k] \in \mathbb{R}^2$, and $\boldsymbol{a}_i[k] \in \mathbb{R}^2$ represent the position, velocity, and acceleration of the i-th robot at discrete time step k, respectively, sampled at time intervals $\Delta T \in \mathbb{R}$. These variables discretize the continuous-time dynamics $\boldsymbol{p}_i(t)$, $\boldsymbol{v}_i(t)$, and $\boldsymbol{a}_i(t)$, such that $\boldsymbol{p}_i[k] := \boldsymbol{p}_i(k\Delta T) = \boldsymbol{p}_i(\lfloor t/\Delta T \rfloor \Delta T)$, where $\lfloor t/\Delta T \rfloor = \max\{k \in \mathbb{Z} \mid k \leqslant t/\Delta T\}$. (1) and (2) describe the translational dynamics in the fixed inertial frame, and (3) and (4) govern the rotational dynamics. Here, $\boldsymbol{R}_i[k] \in \mathbb{R}^{2\times 2}$ is the rotation matrix transforming coordinates from the robot's body frame to the fixed inertial frame, $\boldsymbol{\Omega}_{b,i}[k] \in \mathbb{R}^{2\times 2}$ is the skew-symmetric matrix corresponding to the angular velocity, and $\alpha_{b,i}[k]$ is the angular acceleration. Specifically, these are defined as

$$\boldsymbol{R}_i[k] = \begin{bmatrix} \cos\phi_i[k] & -\sin\phi_i[k] \\ \sin\phi_i[k] & \cos\phi_i[k] \end{bmatrix}, \tag{5}$$

$$\boldsymbol{\Omega}_{b,i}[k] = \begin{bmatrix} 0 & -\omega_{b,i}[k] \\ \omega_{b,i}[k] & 0 \end{bmatrix}, \tag{6}$$

where $\phi_i[k]$ represents the heading angle of the robot. All robots are assumed to have a shape envelope $\rho_{i,c}$, defining the collision-avoidance boundary for the i-th robot.

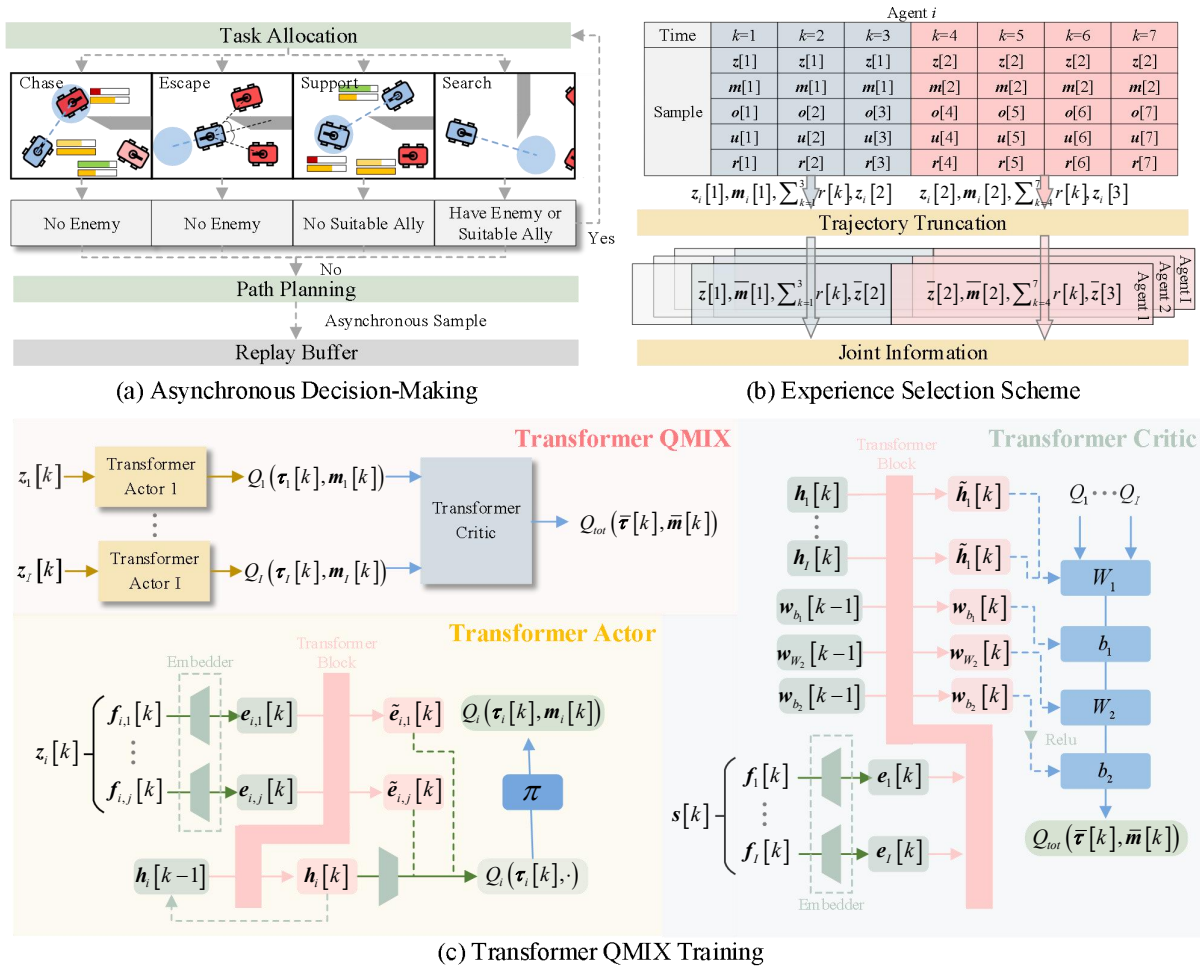
B. MacDec-POMDPs

In a macro-action decentralized partially observable Markov decision process (MacDec-POMDPs) framework, a swarm of robots operates in a dynamic confrontation scenario to achieve collaborative objectives. At each time step k, the i-th robot receives a micro-observation $\boldsymbol{z}_i[k]$ derived from the latent global state $\boldsymbol{s}[k]$ via the observation function. Based on this, the robot selects a micro-action $m_i[k]$ using a high-level strategy $\pi_{i,h}$, which determines a subgoal $\boldsymbol{q}_i[k]$. A macro-action termination function $\xi(m_i[k], \boldsymbol{\tau}_i[k]) \in \{0, 1\}$ evaluates the micro-observation-action history $\boldsymbol{\tau}_i[k] = (\boldsymbol{z}_i[0], m_i[0], \ldots, m_i[k-1], \boldsymbol{z}_i[k])$ to determine whether $m_i[k]$ terminates, triggering the selection of a new macro-action. During the execution of $m_i[k]$, the robot observes a primitive-observation $\boldsymbol{o}_i[k]$ and computes a collision-free primitive-action $\boldsymbol{u}_i[k]$ using a low-level strategy $\pi_{i,l}$ to navigate from its current position $\boldsymbol{p}_i[k]$ toward $\boldsymbol{q}_i[k]$. The robots' objective g_i is to maximize enemy eliminations while ensuring safety, formalized as maximizing the expected cumulative joint reward $\mathbb{E}(\sum_{k=0}^{K} \gamma^k r[k] \mid \boldsymbol{\pi}_i)$, where $r[k]$ is the shared reward at time k, $\gamma \in [0, 1)$ is the discount factor, and $\boldsymbol{\pi}_i = (\pi_{i,h}, \pi_{i,l})$ represents the joint policy for the i-th robot.

III. METHODOLOGY

A. Asynchronous Decision-Making Framework

In this study, we model the relationships among robots as a graph, where each robot observes I entities at time step k, with I representing the total number of robots in the environment. Each entity is characterized by a set of n features, which vary across robots due to the environment's partial observability. Thus, the feature vector for the j-th robot, as observed by the i-th robot at time step k, is defined as $\boldsymbol{f}_{i,j}[k] = (f_{i,j,1}[k], \ldots, f_{i,j,n}[k])$. The primitive-observation is represented as an $I \times n_f$ matrix, rather than



(a) Asynchronous Decision-Making

(b) Experience Selection Scheme

(c) Transformer QMIX Training

Fig. 1. Overview of the proposed method.

a vector, expressed as

$$\boldsymbol{O}_i[k] = \begin{bmatrix} \boldsymbol{f}_{i,1}[k] \\ \vdots \\ \boldsymbol{f}_{i,I}[k] \end{bmatrix} = \begin{bmatrix} f_{i,1,1}[k] & \cdots & f_{i,1,n_f}[k] \\ \vdots & \ddots & \vdots \\ f_{i,I,1}[k] & \cdots & f_{i,I,n_f}[k] \end{bmatrix}. \tag{7}$$

The macro-observation matrix $\boldsymbol{Z}_i[k]$ is defined identically to $\boldsymbol{O}_i[k]$. This structure enables robots to process features of the same type using a shared parameter matrix $\boldsymbol{E}_i$ of shape $n_f \times n_e$, where n_e is the embedding dimension. The resulting matrix $\overline{\boldsymbol{e}}_i[k] = \boldsymbol{Z}_i[k]\boldsymbol{E}_i$ comprises vertex embeddings $(\boldsymbol{e}_{i,1}[k], \ldots, \boldsymbol{e}_{i,I}[k])$, which are further processed by Transformers. The encoding feed-forward layer's shape is independent of the number of robots, ensuring scalability for large-scale scenarios.

In swarm confrontation, the feature set includes position $\boldsymbol{p}$, velocity $\boldsymbol{v}$, and vitality hp, augmented by three binary features to distinguish relationships: $f_{\text{is\_self}}$, $f_{\text{is\_ally}}$, and $f_{\text{is\_enemy}}$. Specifically, $f_{\text{is\_self}} = 1$ indicates the entity is the observing robot itself, otherwise $f_{\text{is\_self}} = 0$; $f_{\text{is\_ally}} = 1$ indicates an ally of the i-th robot, otherwise $f_{\text{is\_ally}} = 0$; and $f_{\text{is\_enemy}} = 1$ indicates an enemy, otherwise $f_{\text{is\_enemy}} = 0$. Thus, the feature vector is $\boldsymbol{f}_{i,j}[k] = (\boldsymbol{p}_j[k], \boldsymbol{v}_j[k], hp_j[k], f_{\text{is\_self}}, f_{\text{is\_ally}}, f_{\text{is\_enemy}})$. The global state is formulated as an $I \times n_f$ matrix, capturing the true features of all entities from a global perspective:

$$\boldsymbol{S}[k] = \begin{bmatrix} \boldsymbol{f}_1[k] \\ \vdots \\ \boldsymbol{f}_I[k] \end{bmatrix} = \begin{bmatrix} f_{1,1}[k] & \cdots & f_{1,n_f}[k] \\ \vdots & \ddots & \vdots \\ f_{I,1}[k] & \cdots & f_{I,n_f}[k] \end{bmatrix}. \tag{8}$$

However, including $f_{\text{is\_self}}$, $f_{\text{is\_ally}}$, and $f_{\text{is\_enemy}}$ in $\boldsymbol{S}[k]$ is unnecessary, as these features are relative to specific robots. Thus, $\boldsymbol{f}_i[k] = (\boldsymbol{p}_i[k], \boldsymbol{v}_i[k], hp_i[k])$. The global state $\boldsymbol{S}[k]$ is processed with a parameter matrix $\boldsymbol{E}$ to yield embedded vertices $\overline{\boldsymbol{e}}[k] = (\boldsymbol{e}_1[k], \ldots, \boldsymbol{e}_I[k]) = \boldsymbol{S}[k]\boldsymbol{E}$.

Based on the macro-observation $\boldsymbol{Z}_i[k]$, the i-th robot utilizes the high-level strategy $\pi_{i,h}$ to determine a micro-action $m_i[k] \in \{1, 2, 3, 4\}$, representing searching, escaping, supporting, or chasing, as illustrated in Fig. 1(a). To prevent arbitrary switching between diverse tasks, we establish an event-driven mechanism to regulate task termination. The subgoal and its corresponding termination condition for each micro-action are defined as follows:

1) For chasing, the subgoal is the position of the enemy with the lowest vitality within the observation range. The task terminates when no enemies remain in range.

2) For supporting, the subgoal targets the position of the ally with the lowest vitality in range, prioritizing aid to the most vulnerable. The task stops when no allies engaged in chasing or escaping are detected.
3) For escaping, the subgoal is a direction opposite to nearby enemies, calculated via a proximity-weighted approach for safety. The task ends when no enemies are in range.
4) For searching, the robot navigates toward a randomly selected location within the battlefield. The task ceases when an enemy or an ally engaged in chasing or escaping enters the range.

Subsequently, the robot generates a collision-free primitive-action $\boldsymbol{u}_i[k] = (\overline{\boldsymbol{v}}_{b,i}[k], \overline{\omega}_{b,i}[k])$ via the low-level strategy $\pi_{i,l}$, guiding it from its current position $\boldsymbol{p}_i[k]$ toward its subgoal $\boldsymbol{q}_i[k]$. $\overline{\boldsymbol{v}}_{b,i}$ and $\overline{\omega}_{b,i}$ represent the desired linear and angular velocities, respectively. The control inputs $\boldsymbol{a}_{f,i}$ and $\alpha_{b,i}$ in (1)–(4) are thus defined as

$$\boldsymbol{a}_{f,i}[k] = \epsilon_v \left(\boldsymbol{R}_i[k]\overline{\boldsymbol{v}}_{b,i}[k] - \boldsymbol{v}_{f,i}[k]\right), \tag{9}$$

$$\alpha_{b,i}[k] = \epsilon_\omega \left(\overline{\omega}_{b,i}[k] - \omega_{b,i}[k]\right), \tag{10}$$

where ϵ_v and ϵ_ω are control gains determined by the real-world dynamics of the robot. These gains are tuned through step-response experiments to ensure that the velocity response in the training environment closely matches the behavior of physical robots. The allocation reward $r_{i,a}[k]$ is designed to encourage reducing enemy vitality while evading attacks, formulated as

$$r_{i,a}[k] = (V_a[k] - V_e[k]) - (V_a[k-1] - V_e[k-1]), \tag{11}$$

where V_a and V_e represent the total vitality of allies and enemies, respectively. Additionally, we design the planning reward with an intrinsic component $r_{i,p_t}[k]$ that encourages shorter paths to the subgoal $\boldsymbol{q}_i[k]$, paired with an avoidance component $r_{i,p_a}[k]$ that ensures collision-free navigation around obstacles and other robots:

$$r_{i,p}[k] = \epsilon_p r_{i,p_t}[k] + (1-\epsilon_p) r_{i,p_a}[k], \tag{12}$$

$$r_{i,p_t}[k] = -\frac{\|\boldsymbol{q}_i[k] - \boldsymbol{p}_i[k+1]\|}{\|\boldsymbol{q}_i[k] - \boldsymbol{p}_i[k]\|}, \tag{13}$$

where ϵ_p balances intrinsic and avoidance rewards. The avoidance reward $r_{i,p_a}[k]$ assigns a penalty of $r_{i,p_a}[k] = -10$ upon collision; otherwise, $r_{i,p_a}[k] = 0$.

B. Experience Selection Scheme

In a decentralized framework, each robot operates using only its own macro-actions, macro-observations, and the joint reward at each time step. This limited information access offers multiple options for data management and storage. For example, a robot could store only its immediate macro-actions and observations, exclude temporal information entirely, or use a representation that explicitly incorporates time progression. We adopt an intermediate approach, updating only upon completion of a robot's macro-action to streamline computation. To support this, we propose an experience selection scheme, as shown in Fig. 1(b), which improves efficiency in processing decentralized data.

In our method, each robot sampling concurrent trajectories from a replay buffer, from which valid experience is selected to create a compact, sequential experience set for each robot. Specifically, high-level transition tuples are collected at each primitive step. A new macro-observation is acquired only upon completion of the current macro-action; otherwise, the subsequent macro-observation remains unchanged from its predecessor. Upon completing a macro-action, each robot records collaborative information shared with other agents, including the joint macro-observation, joint macro-action, cumulative rewards, and the subsequent joint macro-observation, denoted as $(\overline{\boldsymbol{z}}[k], \overline{m}[k], \sum_{k=k-n_\tau}^{k} r[k], \overline{\boldsymbol{z}}[k+1])$. This approach ensures efficient storage and utilization of collaborative information, facilitating cooperation among robots despite their decentralized operation.

We employ QMIX to train the decentralized macro-action-value function $Q_i(\boldsymbol{\tau}_i, m_i \mid \theta_i)$ for the i-th robot. Concurrently, a centralized mixer network leverages the global state $\boldsymbol{s}[k]$ and the robots' hidden states to compute the joint action-value function $Q_{\text{tot}}(\overline{\boldsymbol{\tau}}, \overline{m} \mid \theta, \phi)$. The loss functions for the decentralized actor $Q_i(\boldsymbol{\tau}_i, m_i \mid \theta_i)$ and the centralized critic $Q_{\text{tot}}(\overline{\boldsymbol{\tau}}, \overline{m} \mid \theta, \phi)$ are defined as follows:

$$L(\theta_i) = \mathbb{E}_{\boldsymbol{z},m,r}\left(y_i[k] - Q_{\theta,i}(\boldsymbol{\tau}_i[k], m_i[k])\right)^2, \tag{14}$$

$$y_i[k] = \sum_{k=k-n_\tau}^{k} r[k] + \gamma Q_{\theta,i}(\boldsymbol{\tau}_i[k+1], m_i[k+1]), \tag{15}$$

$$L(\phi) = \mathbb{E}_{\boldsymbol{z},m,r}\left(y[k] - Q_\phi(\overline{\boldsymbol{\tau}}[k], \overline{m}[k])\right)^2, \tag{16}$$

$$y[k] = \sum_{k=k-n_\tau}^{k} r[k] + \gamma Q_\phi(\overline{\boldsymbol{\tau}}[k+1], \overline{m}[k+1]), \tag{17}$$

where $\mathbb{E}$ represents the mathematical expectation. These loss functions facilitate precise optimization of both individual and collective decision-making processes, promoting effective coordination in asynchronous multi-agent systems. Furthermore, as the lower-level experience data is time-synchronized, we directly apply MADDPG [2] to update the strategy $\pi_{i,l}$.

C. Transformer QMIX

In our MARL framework for task allocation, each robot observes a set of entities. The micro-observation for the i-th robot is structured as a matrix $\boldsymbol{Z}_i[k]$, and the macro-action $m_i[k]$ is concatenated to form the action-observation history $\boldsymbol{\tau}_i[k]$ with a sequence length of n_τ. The Transformer actor processes this history as a sequence of embedded vertices, as illustrated in Fig. 1(c). First, the micro-observation is embedded using a parameterized matrix $\boldsymbol{E}_i$, yielding $\overline{\boldsymbol{e}}_i[k] = (\boldsymbol{e}_{i,1}[k], \ldots, \boldsymbol{e}_{i,I}[k])$. The sequence of observations over the past n_τ time steps forms the input matrix:

$$\mathbf{X}_i[k] = [\boldsymbol{h}_i[k-1], \overline{\boldsymbol{e}}_i[k]]^\top_{k-n_\tau:k}, \tag{18}$$

where $\boldsymbol{h}_i[k-1]$ is the hidden state from the previous time step, initialized to zeros at the episode's start. $\mathbf{X}_i[k]$ is

processed by n_l Transformer blocks with multi-head self-attention (MHSA) to focus attention on valid observations:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{n_x}}\right)\mathbf{V}, \tag{19}$$

where n_x is the vertices of $\mathbf{X}_i[k]$. Queries $\mathbf{Q} = \mathbf{X}_i[k]\mathbf{W}^Q$, keys $\mathbf{K} = \mathbf{X}_i[k]\mathbf{W}^K$, and values $\mathbf{V} = \mathbf{X}_i[k]\mathbf{W}^V$, where $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V$ are three different parameterized matrices. The Transformer output is $\tilde{\mathbf{X}}_i[k] = \text{MultiHeadSelfAttn}(\mathbf{X}_i[k])$, where the transformed hidden state $\tilde{\boldsymbol{h}}_i[k-1] = \boldsymbol{h}_i[k]$ encodes the robot's coordination reasoning. The action-values for macro-actions are sampled using a feed-forward layer $\mathbf{W}^u$:

$$Q_i(\boldsymbol{\tau}_i[k], \cdot) = \boldsymbol{h}_i[k]\mathbf{W}^u. \tag{20}$$

For entity-specific macro-actions, such as attacking the j-th enemy or supporting the j-th ally, a decoupling mechanism similar to [5] is used, where the Q-value is derived from the corresponding entity embedding $\tilde{\boldsymbol{e}}_{i,j}[k]$ via an additional feed-forward layer $\mathbf{W}^{\tilde{u}}$:

$$Q_i(\boldsymbol{\tau}_i[k], m_i[k]) = \tilde{\boldsymbol{e}}_{i,j}[k]\mathbf{W}^{\tilde{u}}. \tag{21}$$

The Q-values for entity-related and non-entity-related actions are concatenated to form the complete action-value vector $Q_i(\boldsymbol{\tau}_i[k], \cdot)$. The hidden state $\boldsymbol{h}_i[k]$ is passed to the next time step to maintain temporal dependencies, enabling recurrent coordination reasoning.

The Transformer critic extends QMIX by leveraging a graph-based representation to project individual action-values $\overline{\boldsymbol{Q}} = (Q_1(\boldsymbol{\tau}_i[k], m_i[k]), \ldots, Q_I(\boldsymbol{\tau}_i[k], m_i[k]))$ to the joint action-value Q_{tot}. The critic receives the global state matrix $\boldsymbol{S}[k]$ as input, which is embedded with a parameterized matrix $\boldsymbol{E}$, producing the output $\overline{\boldsymbol{e}}[k] = (\boldsymbol{e}_1[k], \ldots, \boldsymbol{e}_I[k])$. The input graph matrix is

$$\mathbf{X}[k] = \left[\overline{\boldsymbol{h}}[k], \boldsymbol{w}_{b_1}[k-1], \boldsymbol{w}_{W_2}[k-1], \boldsymbol{w}_{b_2}[k-1], \overline{\boldsymbol{e}}[k]\right]^\top, \tag{22}$$

where $\overline{\boldsymbol{h}}[k] = (\boldsymbol{h}_1[k], \ldots, \boldsymbol{h}_I[k])$ and $\boldsymbol{w}_{b_1}[k-1], \boldsymbol{w}_{W_2}[k-1], \boldsymbol{w}_{b_2}[k-1]$ are recurrent vectors initialized to zeros. The Transformer processes $\mathbf{X}[k]$ through the blocks of MHSA:

$$\tilde{\mathbf{X}}_t = \text{MultiHeadSelfAttn}(\mathbf{X}_t). \tag{23}$$

The transformed vertices $\tilde{\boldsymbol{h}}_1[k], \ldots, \tilde{\boldsymbol{h}}_I[k]$ refine the robots' coordination reasoning with global state information, while $\tilde{\boldsymbol{w}}_{b_1}[k-1], \tilde{\boldsymbol{w}}_{W_2}[k-1], \tilde{\boldsymbol{w}}_{b_2}[k-1]$ generate weights and biases for the critic's MLP component. The joint action-value is computed as

$$Q_{\text{tot}} = \left(\overline{\boldsymbol{Q}} \cdot \|\mathbf{W}_1\| + \mathbf{b}_1\right) \cdot \|\mathbf{W}_2\| + \text{ReLU}(\mathbf{b}_2) \tag{24}$$

where $\mathbf{W}_1, \mathbf{b}_1$ and $\mathbf{W}_2, \mathbf{b}_2$ are the weights and biases of the hidden and output layer, respectively. The absolute activation on $\mathbf{W}_1, \mathbf{W}_2$ and ReLU on $\mathbf{b}_2$ ensure monotonicity, satisfying the individual-global-max principle:

$$\frac{\partial Q_{\text{tot}}}{\partial Q_i} \geqslant 0, \quad \forall i \in \{1, \ldots, I\} \tag{25}$$

TABLE I
PARAMETER SETTINGS FOR THE SIMULATION PLATFORM

Parameters	Values
Episode number E_k, Episode length K	90, 300
Time interval τ	0.1 s
Control gains $\epsilon_v, \epsilon_\omega$	1, 0.5
Collision radius $\rho_{i,c}$	0.5 m
Observation radius $\rho_{i,o}$	15 m
Attack radius $\rho_{i,a}$, Attack angle ψ_a	6 m, $\pi/2$
Initial vitality	6
Initial minimum distance $d_{\min}$	20 m
Reload length k_r	10
Sequence length n_τ	10
Block number n_l	2
Discount factor γ, Batch size	0.99, 128
Weight ϵ_p	0.5

The recurrent vectors $\boldsymbol{w}_{b_1}[k] = \tilde{\boldsymbol{w}}_{b_1}[k-1]$, $\boldsymbol{w}_{W_2}[k] = \tilde{\boldsymbol{w}}_{W_2}[k-1]$, $\boldsymbol{w}_{b_2}[k] = \tilde{\boldsymbol{w}}_{b_2}[k-1]$ are passed to the next step, incorporating temporal dependencies into the mixing process. This recurrent mechanism, combined with the graph-based representation, enables the critic to capture complex coordination patterns across asynchronous macro-action executions.

Integrating Transformers into our method enhances coordination reasoning but increases computational complexity compared to MLP-based CTDE approaches. To mitigate this, we implement parameter sharing [27] across the critic and actors, reducing the parameter count by sharing the embedding matrix and MHSA weights among all robots. This lowers gradient computation overhead during backpropagation, reducing per-step training time. Although MHSA computations remain, shared parameters improve GPU parallelization efficiency and promote policy consistency among homogeneous robots, enhancing sample efficiency and potentially accelerating convergence.

IV. EXPERIMENTS

A. Setting Up

In our experiments, we explore three swarm scales: four agents versus four agents (V4), eight agents versus eight agents (V8), and twelve agents versus twelve agents (V12). Parameter configurations are detailed in Table I. Our algorithm undergoes training across 100 independent training instances, each initialized with a unique random seed to diversify initial agent positions and network weights. Within each instance, the algorithm undergoes training over E_k episodes, with all vehicles starting from consistent initial positions in each episode. To ensure equitable comparisons, we assess both baseline methods and our approach under uniform experimental conditions, utilizing consistent input data and evaluation criteria. The baselines and our method are deployed for the blue team, while the red team relies on expert systems [1]. We develop the scenarios and algorithms using Python. Training is conducted on an Ubuntu 20.04 server equipped with a single NVIDIA L40S GPU.

We simulate multi-robot confrontations within environments featuring static obstacles. Robots engage in combat

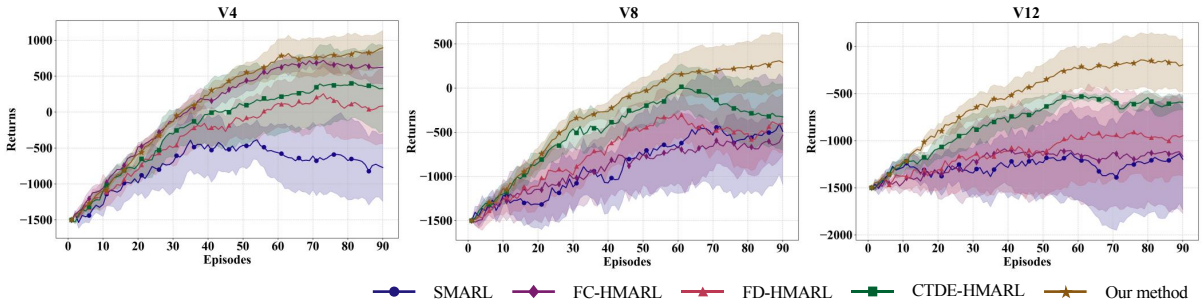


Fig. 2. Learning curves of different algorithms for various swarm sizes. Solid lines denote mean values across all instances, with shaded areas representing standard deviations. The figures plot episode counts on the horizontal axis and episode returns on the vertical axis.

TABLE II

PERFORMANCE COMPARISON OF ALGORITHMS ACROSS DIFFERENT SWARM SIZES

Method	V4			V8			V12		
	Re.	Ti.(min)	W.R.(%)	Re.	Ti.(min)	W.R.(%)	Re.	Ti.(min)	W.R.(%)
SMARL	-772 ± 173	**11 ± 2**	53 ± 4	-815 ± 181	**15 ± 2**	52 ± 5	-912 ± 195	**19 ± 3**	53 ± 6
FC-HMARL	621 ± 112	16 ± 3	76 ± 2	-306 ± 147	22 ± 4	59 ± 3	-998 ± 162	30 ± 5	52 ± 4
FD-HMARL	84 ± 134	12 ± 2	61 ± 3	-590 ± 149	15 ± 3	54 ± 4	-998 ± 157	20 ± 3	51 ± 5
CTDE-HMARL	347 ± 93	14 ± 2	68 ± 1	-228 ± 87	20 ± 3	63 ± 2	-305 ± 91	27 ± 3	60 ± 3
Our Method	**896 ± 63**	18 ± 1	**83 ± 1**	**673 ± 67**	27 ± 2	**78 ± 1**	**423 ± 71**	36 ± 2	**75 ± 2**

Note: The table presents episode returns (Re.), training time (Ti.), and confrontation win rate (W.R.) for each method. Data values are reported as mean ± standard deviation, calculated across 10 independent evaluation batches of 10 episodes each.

with adversaries until their vitality is exhausted to zero through successive attacks. The governing rules for swarm confrontations are as follows:

1) The blue and red teams start with identical attributes, including numbers, vitality, ammunition, and constraints.
2) Initially, all robots are randomly placed within the scenario, ensuring a minimum distance $d_{\min}$ between any two robots.
3) Following a missile launch, a robot requires k_r steps to reload, during which it cannot initiate further attacks.
4) Obstacles may impede the observation and attack capabilities of robots.
5) A robot is considered incapacitated when its vitality reaches zero.

The observation radius of the i-th robot is denoted by $\rho_{i,o}$. It can target the j-th robot within a forward-facing sector defined by an attack radius $\rho_{i,a}$ and an angle of $2\psi_a$. Upon launch, the missile follows a predetermined trajectory, successfully hitting if $\psi_{i,j} \leqslant \psi_h$; otherwise, the strike fails. The maximum hit angle ψ_h is determined by

$$\psi_h = \arctan \frac{\rho_{i,c}}{|\boldsymbol{p}_j[k] - \boldsymbol{p}_i[k]|}. \tag{26}$$

$\psi_{i,j}$ denotes the angle between the vectors $\boldsymbol{p}_j[k] - \boldsymbol{p}_i[k]$ and $\boldsymbol{p}_j[k+1] - \boldsymbol{p}_i[k]$.

B. Comparative Analysis

We employ MARL methods to optimize the performance of confrontation strategies during training. Our approach is evaluated against several MLP-based baselines, including single-layer MARL (SMARL) [10], fully centralized hierarchical MARL (FC-HMARL) [18], fully decentralized hierarchical MARL (FD-HMARL) [17], and CTDE-hierarchical MARL (CTDE-HMARL) [9].

As illustrated in Fig. 2, SMARL consistently underperforms across all scales, with its training curves exhibiting significant fluctuations, especially in larger-scale scenarios like V8 and V12. This instability highlights its inadequacy in modeling complex multi-agent interactions under partial observability. FD-HMARL, which depends on decentralized training and execution, provides modest improvements over SMARL but faces scalability challenges, as demonstrated by slower convergence and elevated variance in larger swarms, largely due to its sensitivity to non-stationarity. FC-HMARL, benefiting from global state information, performs exceptionally in smaller-scale settings like V4, achieving returns comparable to our method, thanks to its ability to leverage complete state data for effective coordination. However, in V8 and V12 scenarios, FC-HMARL's performance falls significantly below CTDE-HMARL, as its conventional MLP-based architecture struggles to process the increasingly complex global information in larger-scale settings. The exponential growth of state space and inter-agent dependencies exceeds the network's ability to extract effective coordination patterns, leading to suboptimal policy learning and heightened variance.

CTDE-HMARL surpasses FD-HMARL and FC-HMARL in larger scales, featuring more stable training curves that reflect a balanced approach to global coordination and decentralized action. Nevertheless, its dependence on MLP-based networks restricts its ability to fully capture complex agent interactions. Our method, integrating a Transformer-based architecture with graph-structured reasoning and asyn-

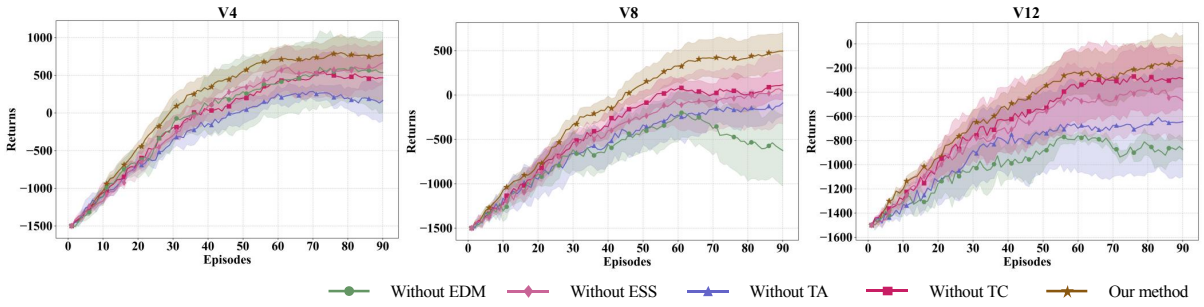


Fig. 3. Ablation study results of our method.

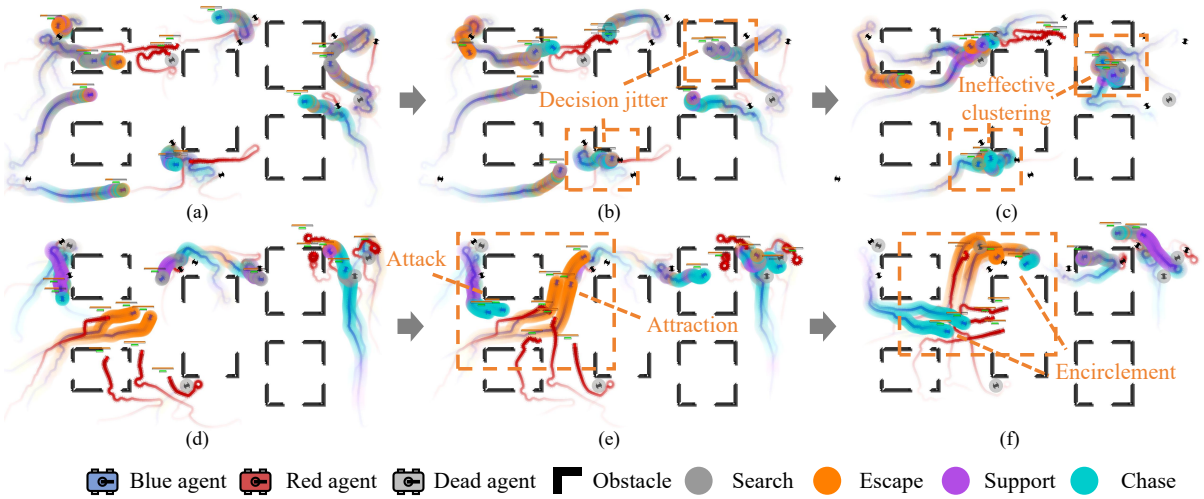


Fig. 4. Training process of swarm confrontation in the V12 scenario. (a)–(c) Training with the variant without EDM. (d)–(f) Training with our method.

chronous macro-action management, consistently delivers the highest episode returns across all scales. Its training curves demonstrate rapid convergence and low variance, driven by the MHSA mechanism, which effectively models inter-agent collaboration and scales well to larger teams.

Furthermore, we deploy policy networks trained with the aforementioned methods across an additional 100 instances to provide a comprehensive evaluation. To ensure a robust analysis, we incorporate two supplementary metrics: training time and confrontation win rate. Training time represents the average duration needed to train the MARL model for each instance, whereas confrontation win rate indicates the percentage of successful outcomes across all instances. The results are compiled in Table II. A clear correlation emerges between episode returns and win rates, with agents optimized for higher returns demonstrating significantly superior success rates in confrontations. SMARL and FD-HMARL, characterized by lower win rates and shorter training time, are limited by their simplistic and decentralized architectures. FC-HMARL excels in the V4 scenario but experiences a decline in performance for V8 and V12, constrained by the inherent shortcomings of its MLP-based design and accompanied by moderately to substantially lengthy training durations. CTDE-HMARL sustains competitive win rates with reasonable training time, capitalizing on centralized training to enhance stability. Compared to MLP-based CTDE-HMARL, our method incurs higher per-step complexity due to the MHSA mechanism, resulting in longer training durations. However, its improved sample efficiency, enabled by graph-structured reasoning, reduces the number of episodes required for convergence, partially offsetting the computational overhead. This makes our method well-suited for large-scale swarm confrontation scenarios, achieving the highest win rates.

C. Ablation Study

We develop an event-driven mechanism (EDM) to regulate task termination, minimizing inefficient exploration while ensuring task continuity. Subsequently, we integrate an experience selection scheme (ESS) to enhance training efficiency in decentralized environments. Furthermore, we utilize a Transformer actor (TA) and Transformer critic (TC) to process observation matrices and global states, enabling sophisticated graph-based coordination reasoning. To assess the contributions of these components, we perform ablation studies, with the resulting learning curves depicted in Fig. 3.

The variant without EDM shows erratic learning curves with high variance, particularly in V12, due to frequent task switching from synchronous updates, which disrupts task continuity and exacerbates exploration inefficiencies. The version lacking ESS exhibits slower convergence and moderate variance, as redundant data processing reduces

sample efficiency. Configurations excluding TA and TC display slower convergence and elevated variance, especially in V8 and V12, since MLP-based networks struggle to capture complex inter-agent and global state interactions, leading to suboptimal coordination. These findings highlight the critical roles of EDM in ensuring stable exploration, ESS in enhancing training efficiency, and TA and TC in enabling robust coordination, making our method highly effective for large-scale swarm confrontations.

D. Emerged Coordinated Confrontation Behaviors

Through sufficient training, our proposed method effectively develops robust confrontation strategies. We illustrate the training outcomes for our method and the variant without EDM in a V12 scenario, as shown in Fig. 4. The variant without EDM replaces event-triggered task termination with synchronous macro-action updates, where all robots switch tasks at fixed intervals, mimicking MARL without asynchronous adaptability. Consequently, agents undergo frequent task switches, causing decision jitter. When multiple allies are in close proximity, this can lead to inefficient clustering and reduced overall coordination. In contrast, our approach mitigates unproductive exploration, enabling agents to maintain stable behavior over extended periods. Additionally, it promotes collaborative tactics within the swarm, with some agents drawing enemy attention while others execute attacks from behind, culminating in strategic encirclement of distracted enemies to enhance overall effectiveness.

V. Conclusions

Our proposed method advances swarm confrontation by offering a robust solution for event-driven MARL training under asynchronous conditions. By integrating an experience selection scheme with Transformer-enhanced QMIX, it achieves superior performance, as validated through extensive experiments. Compared to existing MARL approaches, our method delivers higher win rates in confrontations, particularly in large-scale scenarios, underscoring its significant potential for real-world robotic confrontation applications.

References

[1] Y. Hou, X. Liang, J. Zhang, M. Lv, and A. Yang, "Hierarchical decision-making framework for multiple ucavs autonomous confrontation," *IEEE Trans. Veh. Technol.*, vol. 72, no. 11, pp. 13 953–13 968, 2023.

[2] Q. Wu, K. Liu, L. Chen, and J. Lü, "Hierarchical reinforcement learning for swarm confrontation with high uncertainty," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 8630–8644, 2025.

[3] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, J. Humplik, M. Wulfmeier, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *Sci. Rob.*, vol. 9, no. 89, p. eadi8022, 2024.

[4] F. Liu, X. Dong, J. Yu, Y. Hua, Q. Li, and Z. Ren, "Distributed nash equilibrium seeking of n-coalition noncooperative games with application to uav swarms," *IEEE Trans. Network Sci. Eng.*, vol. 9, no. 4, pp. 2392–2405, 2022.

[5] H. Liu, J. Zhang, P. Zu, and M. Zhou, "Evolutionary algorithm-based attack strategy with swarm robots in denied environments," *IEEE Trans. Evol. Comput.*, vol. 27, no. 6, pp. 1562–1574, Dec. 2023.

[6] M. Dawood, S. Pan, N. Dengler, S. Zhou, A. P. Schoellig, and M. Bennewitz, "Safe multi-agent reinforcement learning for behavior-based cooperative navigation," *IEEE Rob. Autom. Lett.*, vol. 10, no. 6, pp. 6256–6263, 2025.

[7] P. Feng, R. Shi, S. Wang, J. Liang, X. Yu, S. Li, and W. Wu, "Safe and efficient multi-agent collision avoidance with physics-informed reinforcement learning," *IEEE Rob. Autom. Lett.*, vol. 10, no. 6, pp. 6256–6263, 2025.

[8] X. Qu, W. Gan, D. Song, and L. Zhou, "Pursuit-evasion game strategy of usv based on deep reinforcement learning in complex multi-obstacle environment," *Ocean Eng.*, vol. 273, p. 114016, 2023.

[9] D. Liu, Q. Zong, X. Zhang, R. Zhang, L. Dou, and B. Tian, "Game of drones: Intelligent online decision making of multi-uav confrontation," *IEEE Trans. Emerging Top. Comput. Intell.*, vol. 8, no. 2, pp. 2086–2100, 2024.

[10] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[11] B. Wang, S. Li, X. Gao, and T. Xie, "Uav swarm confrontation using hierarchical multiagent reinforcement learning," *Int. J. Aerosp. Eng.*, vol. 2021, pp. 1–12, Dec. 2021.

[12] W. Kong, D. Zhou, Y. Du, Y. Zhou, and Y. Zhao, "Hierarchical multi-agent reinforcement learning for multi-aircraft close-range air combat," *IET Control Theory & Appl.*, vol. 17, no. 13, pp. 1840–1862, Dec. 2023.

[13] Q. Wu, K. Liu, and L. Chen, "Model predictive control–based value estimation for efficient reinforcement learning," *IEEE Intell. Syst.*, vol. 39, no. 3, pp. 63–72, 2024.

[14] X. Liu, G. Wang, and K. Chen, "Option-based multi-agent reinforcement learning for painting with multiple large-sized robots," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15 707–15 715, 2022.

[15] Z. Wen, H. Tan, B. Yu, and Y. Zhao, "An end-to-end hrl-based framework with macro-micro adaptive layer for mixed on-ramp merging," in *IEEE Intell. Veh. Symp.* IEEE, 2024, pp. 1424–1429.

[16] C. Qian, X. Zhang, L. Li, Y. Wang, M. Zhao, and Y. Fang, "A partial joint optimization algorithm for autonomous air combat based on hierarchical reinforcement learning," *IEEE Trans. Cybern.*, vol. 55, no. 9, pp. 4145–4157, 2025.

[17] Z. Liang, J. Cao, S. Jiang, and H. Xu, "Hierarchical reinforcement learning with partner modeling for distributed multi-agent cooperation," *IEEE Trans. Parallel Distrib. Syst.*, early access, Sep., 2024.

[18] X. Lyu, A. Banitalebi-Dehkordi, M. Chen, and Y. Zhang, "Asynchronous, option-based multi-agent policy gradient: A conditional reasoning approach," in *Proc. Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2023, pp. 7348–7353.

[19] Y. Xiao, W. Tan, J. Hoffman, T. Xia, and C. Amato, "Asynchronous multi-agent deep reinforcement learning under partial observability," *Int. J. Rob. Res.*, vol. 44, no. 8, pp. 1257–1286, 2025.

[20] Y. Nie, J. Liu, X. Liu, Y. Zhao, K. Ren, and C. Chen, "Asynchronous multi-agent reinforcement learning-based framework for bi-level non-cooperative game-theoretic demand response," *IEEE Trans. Smart Grid*, vol. 15, no. 6, pp. 5622–5637, 2024.

[21] J. Yin, W. Rao, Y. Xiao, and K. Tang, "Cooperative path planning with asynchronous multiagent reinforcement learning," *IEEE Trans. Mob. Comput.*, vol. 24, no. 6, pp. 5016–5030, 2025.

[22] Z. Wang, C. Wang, X. Li, C. Xia, and J. Xu, "Mlp-net: multi-layer perceptron fusion network for infrared small target detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 63, pp. 1–13, 2024.

[23] M. Liu, Y. Zhu, Y. Chen, and D. Zhao, "Enhancing reinforcement learning via transformer-based state predictive representations," *IEEE Trans. Artif. Intell.*, vol. 5, no. 9, pp. 4364–4375, 2024.

[24] S. Hu, L. Shen, Y. Zhang, Y. Chen, and D. Tao, "On transforming reinforcement learning with transformers: The development trajectory," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 8580–8599, 2024.

[25] Y. Zhu, S. Huang, B. Zuo, D. Zhao, and C. Sun, "Multi-task multi-agent reinforcement learning with task-entity transformers and value decomposition training," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 9164–9177, 2025.

[26] M. Zhang, W. Tong, G. Zhu, X. Xu, and E. Q. Wu, "Sqix: Qmix algorithm activated by general softmax operator for cooperative multiagent reinforcement learning," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 54, no. 11, pp. 6550–6560, 2024.

[27] X. Zhou, H. Huang, Z. Wang, and R. He, "Ristra: Recursive image super-resolution transformer with relativistic assessment," *IEEE Trans. Multimedia*, vol. 26, pp. 6475–6487, 2024.