

RALLY: Role-Adaptive LLM-Driven Yoked Navigation for Agentic UAV Swarms

ZIYAO WANG^{1,3}, RONGPENG LI² (Senior Member, IEEE), SIZHAO LI², YUMING XIANG²,
HAIPING WANG³, ZHIFENG ZHAO^{2,3} (Senior Member, IEEE), AND HONGGANG ZHANG⁴ (Fellow, IEEE)

(Invited Paper)

¹Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 311500, China

²Zhejiang University, Hangzhou 310027, China

³Zhejiang Lab, Hangzhou 310012, China

⁴Macau University of Science and Technology, Macau, China

CORRESPONDING AUTHOR: ZHIFENG ZHAO (e-mail: zhaozf@zhejianglab.org).

This work was supported in part by the National Key Research and Development Program of China under Grant 2024YFE0200600, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LR23F010005, in part by Huawei Cooperation Project under Grant TC20240829036, and in part by the Big Data and Intelligent Computing Key Lab of CQUPT under Grant BDIC-2023-B-001.

ABSTRACT Intelligent control of Uncrewed Aerial Vehicles (UAVs) swarms has emerged as a critical research focus, and it typically requires the swarm to navigate effectively while avoiding obstacles and achieving continuous coverage over multiple mission targets. Although traditional Multi-Agent Reinforcement Learning (MARL) approaches offer dynamic adaptability, they are hindered by the semantic gap in black-boxed communication and the rigidity of homogeneous role structures, resulting in poor generalization and limited task scalability. Recent advances in Large Language Model (LLM)-based control frameworks demonstrate strong semantic reasoning capabilities by leveraging extensive prior knowledge. However, due to the lack of online learning and over-reliance on static priors, these works often struggle with effective exploration, leading to reduced individual potential and overall system performance. To address these limitations, we propose a Role-Adaptive LLM-Driven Yoked navigation algorithm RALLY. Specifically, we first develop an LLM-driven semantic decision framework that uses structured natural language for efficient semantic communication and collaborative reasoning. Afterward, we introduce a dynamic role-heterogeneity mechanism for adaptive role switching and personalized decision-making. Furthermore, we propose a Role-value Mixing Network (RMIX)-based assignment strategy that integrates LLM offline priors with MARL online policies to enable offline training of role selection strategies. Experiments in the Multi-Agent Particle Environment (MPE) and a Software-In-The-Loop (SITL) platform demonstrate that RALLY outperforms conventional approaches in terms of task coverage, convergence speed, and generalization, highlighting its strong potential for collaborative navigation in agentic multi-UAV systems.

INDEX TERMS Intelligent UAV swarm control, large language model, multi-agent reinforcement learning, role-heterogeneous network, agentic AI.

I. INTRODUCTION

Nowadays, Uncrewed Aerial Vehicles (UAVs) have demonstrated significant potential in multi-agent pursuit-evasion game (e.g., disaster responses [1]). Typically, the UAV swarm shall have the capability to avoid pursuing enemies and environmental obstacles while moving towards multiple target areas in certain formations, which is often known as a Dynamic Swarm coordination with Cooperative Evasion

and Formation Coverage (DS-CEFC) task [2]. By enabling UAVs to adjust their functions in response to real-time environmental conditions, dynamic role adaptation often leads to optimized coordination, improved task coverage, and enhanced robustness in complex and unpredictable scenarios [3], [4]. Nevertheless, the underlying difficulty in coordinating roles and decision-making across multiple agents in swarms poses significant challenges. For example, traditional

control-based algorithms [5], [6] are contingent on fully connected topologies and lack the required adaptability and scalability to large-scale dynamic environments, thus degrading the practicality in the real world [7]. Meanwhile, despite the incorporation of inter-agent communications [8], [9] and cooperation [10], [11], it still remains inevitable for decentralized Multi-Agent Reinforcement Learning (MARL) to yield conflicting roles and decisions [12], [13]. Given the inherent semantic reasoning capabilities and pretrained experience of Large Language Models (LLMs) [14], [15], they offer a promising alternative to mitigate the critical issues of conflicting roles and inconsistent decision-making that plague purely MARL-based approaches.

A. RELATED WORKS

1) MARL-BASED UAV SWARM CONTROL

Deep Reinforcement Learning (DRL) [16] has significantly enhanced agent adaptability in complex tasks. However, in Multi-Agent Systems (MAS), critical challenges, such as environmental non-stationarity, rapidly expanding state spaces, and difficulty in credit assignment, hinder the ability of traditional methods to learn effective cooperative policies. To address this problem, the Centralized Training with Decentralized Execution (CTDE) paradigm is introduced with exemplary algorithms like MADDPG [17] and MAPPO [18]. During training, it also involves techniques such as policy distillation [19] and imitation learning [20] to improve coordination in complex obstacle-laden environments. Nevertheless, these approaches struggle to assess an individual agent's contribution, as they typically optimize a global value function while neglecting the importance of localized utility. Value decomposition methods (e.g., VDN [10], QMIX [21], QTRAN [22]) ameliorate this issue by decomposing the joint value function, thereby enabling analysis of individuals' contributions to cooperative decision-making. Furthermore, more advanced attention mechanisms or neural communication protocols (e.g., TarMAC [11], IMANet [23], DAACMP [24]) can boost the effectiveness of filter messages. However, the direct communication of numerical vectors [11], [23], [24], which lacks interpretability and cannot convey task semantics, leads to information redundancy and bandwidth bottlenecks [8], [9], [25], [26], and greatly limits algorithm generalization. Although the leader-follower architecture [27] enables role differentiation, its static role assignments lack the flexibility to adapt to dynamic environmental conditions and varying formation sizes. Therefore, researchers resort to integrating hierarchical control with consensus inference [2], [28], [29], so as to simplify the inference interpretability and boost convergence speed. However, agents in these methods commonly remain homogeneous, and the policy networks learned via CTDE cannot inherently leverage the advantages of the natural heterogeneity in swarms. Therefore, building a scalable, efficient, dynamically adaptive, and interpretable heterogeneous multi-agent collaboration mechanism for UAV swarm control remains an open problem.

2) LLM-BASED MULTI-AGENT SYSTEMS

LLMs have demonstrated near-human performance [14] in complex reasoning and planning tasks, providing new impetus for environment understanding and decision explainability in UAV swarms [15]. Leveraging vast prior knowledge, LLMs can not only be used for single-agent path planning (e.g., CoNavGPT [30], RoCo [31]), but also facilitate high-level communication with low-level trajectory planning, significantly improving task efficiency, adaptability, and generalization [32], [33], [34]. Moreover, in MetaGPT [35], CAMEL [36] and ChatDev [37], LLMs can decompose complex missions into a number of subtasks handled collaboratively by different agents, thereby reducing “hallucinations” [38] and enhancing the ability to solve complex problems. More importantly, these LLM-driven Multi-Agent (LLM-MA) systems [39], [40], [41] often customize LLMs into specialized or personalized roles. Therefore, through multi-agent collaboration, they replicate human-like collective intelligence and further enhance overall situational understanding and decision explainability, making inter-agent interactions more meaningful. However, LLM-based decision-making still heavily relies on prior knowledge and lacks exploration, often getting stuck in local optima [42]. Additionally, these methods leverage individual memory and self-evolution mechanisms to optimize agents independently, while neglecting potential synergistic effects arising from multi-agent interactions. Furthermore, applying LLM-MA to build global consensus for DS-CEFC remains scarce, though some studies [41] start to focus on consensus negotiation among agents in a simplified scenario. On the other hand, some approaches [39] utilize diverse “personalities”, empowered by the LLM, for creative collaboration; nevertheless, the underlying fixed definitions of roles still struggle to accommodate dynamic task switching in DS-CEFC. In other words, enabling dynamic role adaptation with balanced exploration and exploitation remains a critical challenge for DS-CEFC.

3) INTEGRATION OF MARL AND LLM

There has been some light shed on deeply integrating MARL and LLMs [43]. Prominently, semantic capabilities in LLMs can be leveraged as natural language interfaces to bridge human feedback and MAS [44], but the consensus inference therein often relies more on human supervision than on autonomous collaboration. Other studies distill LLM knowledge into smaller executable models or empower LLMs with human-in-the-loop feedback for policy generation, so as to accelerate MARL training and improve performance in complex environments [45], [46], [47]. Nevertheless, these methods often depend on offline human annotations and feedback, making them ill-suited for dynamic real-time changes; meanwhile, agent roles remain homogeneous, with only the number of agents being scaled up. Some recent work treats LLMs as the core of heterogeneous agents in MARL [14], assigning different “personalities” to agents. But these roles tend to be fixed and cannot adapt to environmental changes in

DS-CEFC. Additionally, deploying LLMs in UAV swarms significantly strains computational resources and energy, limiting their practical use [48]. Thus, it is meaningful to calibrate superior means to integrate MARL and LLMs, thus better harnessing heterogeneous swarm intelligence for improved generalization and adaptability in DS-CEFC.

B. CONTRIBUTIONS

To accomplish consensus inference of multiple UAVs with heterogeneous roles for DS-CEFC, we propose a novel LLM-MARL-integrated framework called RALLY (Role-Adaptive LLM-Driven Yoked navigation). Built on our previous work [2], which unifies target selection, obstacle-avoidance navigation, and flight-control execution, RALLY serves as a significantly enhanced high-level consensus inference module towards establishing role-adaptive cooperative navigation. Specifically, RALLY comprises two core modules: an LLM-based two-stage semantic reasoning module and a Role-value Mixing Network (RMIX)-based credit-distribution mechanism. The integration of semantic reasoning in LLM and policy learning in MARL makes it qualified for coordinating roles and decision-making across swarms for DS-CEFC. The main contributions of this work can be summarized as follows:

- We introduce RALLY, which consists of an LLM-driven semantic reasoning architecture for goal inference alongside a RMIX-based credit-distribution mechanism for role selection. RALLY accelerates consensus formation, improves convergence speed, and optimizes cooperative behaviors within the UAV swarm to fulfill DS-CEFC.
- We implement a two-stage LLM-based semantic reasoning of intention and consensus inference. Replacing traditional numerical vector-based methods with more interpretable text contributes to the consensus inference efficiency.
- Unlike static role assignment, RMIX dynamically assigns agents' roles during cooperative navigation. By integrating offline LLM experiences with online MARL training, RMIX optimizes credit assignment and accelerates group coordination across diverse scenarios.
- To meet the distributed deployment and parallel inference demands on edge devices, we propose a capacity-migration algorithm that significantly reduces runtime memory footprint. We validate RALLY in the Multi-Particle Environment (MPE) [49] and the Software-In-The-Loop (SITL) platform [50] based on Gazebo-ROS-PX4. Experimental results demonstrate that RALLY outperforms the latest MARL [2] and pure LLM-driven approaches [30], [51] in terms of task completion, convergence speed, generalization, and interpretability.

C. PAPER STRUCTURE

The remainder of this paper is organized as follows. Section II introduces the system model and formulates the problem. Section III outlines the algorithmic framework of RALLY by elaborating on the RMIX role assignment mechanism and

TABLE 1. Mainly Used Notations in This Paper

Symbol	Description
n	Number of UAVs
$\mathcal{N}$	Set of UAVs
$\mathcal{T}$	Set of target regions
$\mathbf{p}_{tr}, \kappa_{tr}^t$	Target positions/urgency
$\mathbf{p}_e^t, \mathbf{v}_e^t$	Enemy's position/velocity
$\mathbf{p}_i^t, \mathbf{v}_i^t$	Agent i 's position/velocity
$\mathbf{s}^t$	Global state
$\mathcal{N}_i^t$	Set of agent i 's neighbors at time t
$\mathbf{o}_i^t$	Local Observation of agent i
$\mathbf{z}_i^t$	Local information of agent i
k_i^t	Role assignment of agent i
$\mathcal{G}$	Set of candidate target goals
$g_i'^t$	Initial goal intention of agent i
g_i^t	Final consensus goal of agent i
RL _{HI}	Credit-assignment role selection
LLM _{HC}	Two-stage LLM-based consensus inference
LLM _{init}	Initial intent generation policy
LLM _{cons}	Consensus-refinement policy
X _{task}	Task instruction prompt
Y _{init}	Initial LLM prompt
Y _{cons}	Consensus-refinement LLM prompt
M _{CoT}	Chain-of-Thought guidance prompts

context-migration algorithm. Section IV presents experimental results and discussion. Finally, Section V summarizes this work with possible research directions.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. SYSTEM MODEL

Beforehand, commonly used variables are summarized in Table 1.

Hereinafter, we consider a DS-CEFC task, wherein n communication-limited UAVs in set $\mathcal{N}$ cooperatively cover multiple target regions $\mathcal{T}$, with 2D positions $\mathbf{p}_{tr}^t$ and urgency levels κ_{tr}^t that decreases over coverage time, $tr \in \mathcal{T}$, alongside the blocking from static obstacles and chasing from PPO-based [52] adversarial “enemy”. Therefore, during navigation, agents shall dynamically change the target region from a set of candidate target goals $\mathcal{G}^t \subset \mathcal{T}$ to balance evasion and coverage efficiency. Due to the partial observability and communication limitation within the observation range δ_{obs} , as illustrated in Fig. 1, each agent i obtains an environmental observation as

$$\mathbf{o}_i^t := \{(\mathbf{p}_i^t, \mathbf{v}_i^t), (\mathbf{p}_e^t, \mathbf{v}_e^t), (\mathbf{p}_{tr}^t, \kappa_{tr}^t)_{tr \in \mathcal{T}}\}, \quad (1)$$

where $\mathbf{p}_i^t = [p_{x_i}^t, p_{y_i}^t]$ and $\mathbf{v}_i^t = [v_{x_i}^t, v_{y_i}^t]$ are the 2D position and velocity of agent i , respectively, while $\mathbf{p}_e^t = [p_{x_e}^t, p_{y_e}^t]$, $\mathbf{v}_e^t = [v_{x_e}^t, v_{y_e}^t]$ are those of enemy agent. Based on its local observation $\mathbf{o}_i^t$, agent i first generates a target intention $g_i^t \in \mathcal{T}$ and role assignment $k_i^t \in \mathcal{K} = \{\text{Commander, Coordinator, Executor}\}$ (i.e.,

Step 1. Intention Generate

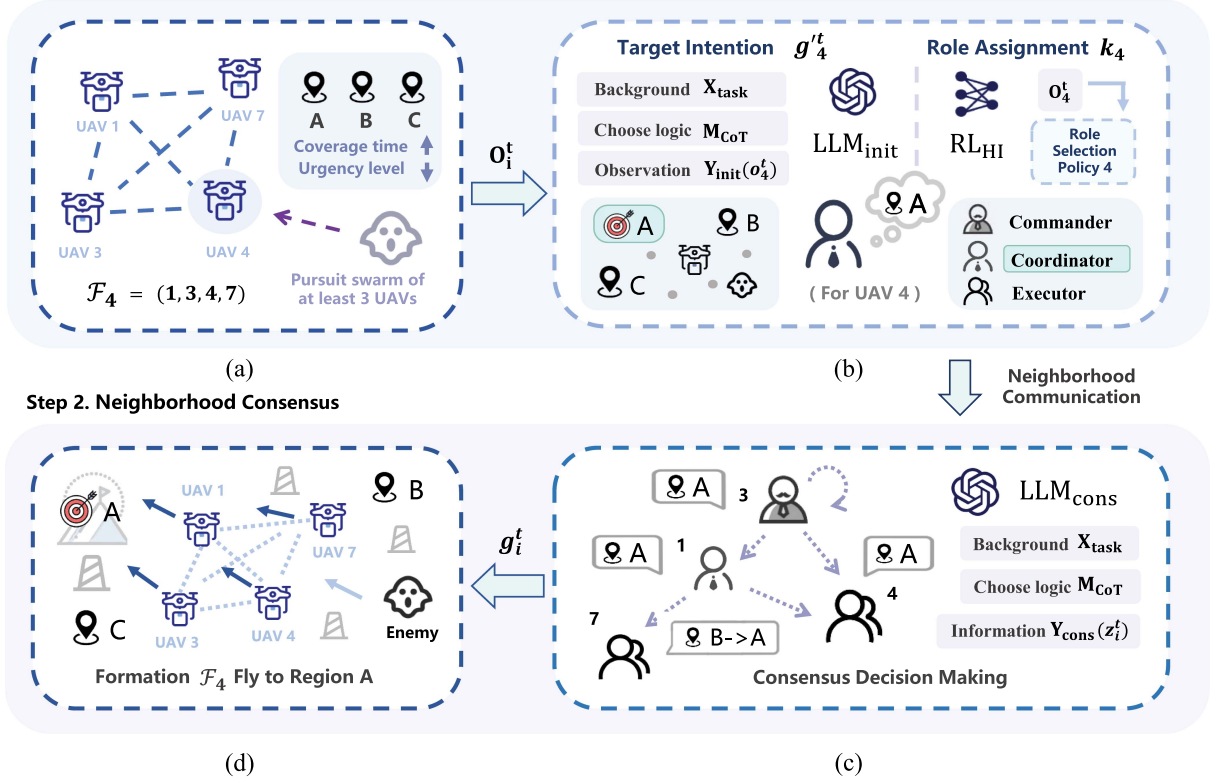


FIGURE 3. Flowchart of the RALLY algorithm. (a) Taking formation $\mathcal{F}_4$ as an example, UAVs 1, 3, 4, and 7 adaptively form the team. (b) Based on local observations o_i^t , each agent selects target intention g_i^t and role k_i . Based on background prompt descriptions X_{task} , reasoning logic M_{CoT} , and local observation prompts $Y_{init}(o_i^t)$, UAVs generate target intentions via policy LLM_{init} , and then determine roles k_i through a role selection policy RL_{HI} . (c) After neighborhood communications, LLM_{cons} refines the goal to reach a final, converged consensus decision g_i^t (e.g., Region A), incorporating individual role preferences. (d) The resulting formation $\mathcal{F}_4$ then collectively navigates toward Region A.

design a task-specific instruction X_{task} that outlines agents' mission requirements and task conditions, and craft the prompt Y_{init} to capture the observation o_i^t of agent i . The LLM's natural-language output is then parsed into numeric goals g_i^t as

$$g_i^t = LLM_{init}(X_{task}, Y_{init}(o_i^t), M_{CoT}). \quad (5)$$

Next, the Role Selection Policy RL_{HI} takes the current observation o_i^t as input to optimize the role selection for each agent. The final selected role k_i^t is given as

$$k_i^t = RL_{HI}(o_i^t; g_i^t), \quad (6)$$

which determines the agent's action strategy based on its role.

After communication with neighbors j within a range δ_{obs} (i.e., $\|p_i^t - p_j^t\| < \delta_{obs}, \forall j \in \mathcal{N}_i^t$), the local information available to agent i at time t can be denoted as:

$$z_i^t = \left(o_i^t, \left\{ (p_j^t, v_j^t, g_j^t, k_j^t) \right\}_{j \in \mathcal{N}_i^t} \right). \quad (7)$$

On this basis, each agent i constructs a new prompt Y_{cons} describing its intents, RL_{HI} -produced role k_i^t , neighbors'

roles, as well as environmental constraints. We further incorporate task-driven Chain-of-Thought (CoT) prompts M_{CoT} , such as "Cluster or disperse based on the threats from enemy" and "needs cluster with other two teammates", to strengthen the LLM's reasoning and minimize hallucinations. The LLM is instructed to output a *refined consensus* like "I recommend going to target $[x, y]$ " which is converted into the final numeric target:

$$g_i^t = LLM_{cons}(X_{task}, M_{CoT}, Y_{cons}(z_i^t)). \quad (8)$$

This refinement embeds a balance between maximizing swarm utility and ensuring safe avoidance of enemy under adversarial and coverage requirements.

Example prompts for X_{task} and M_{CoT} are illustrated in Fig. 4, while more prompt details, such as Y_{init} and Y_{cons} , are provided in the Appendix. Notably, as evidenced in Fig. 19 of the Appendix, the design of prompts shows considerable sensitivity to the environmental changes and yields significantly different responses.

For occasional illegal LLM outputs, we implement hierarchical contingency strategies: Commander maintains initial intent; Coordinator defers to valid Commander (else

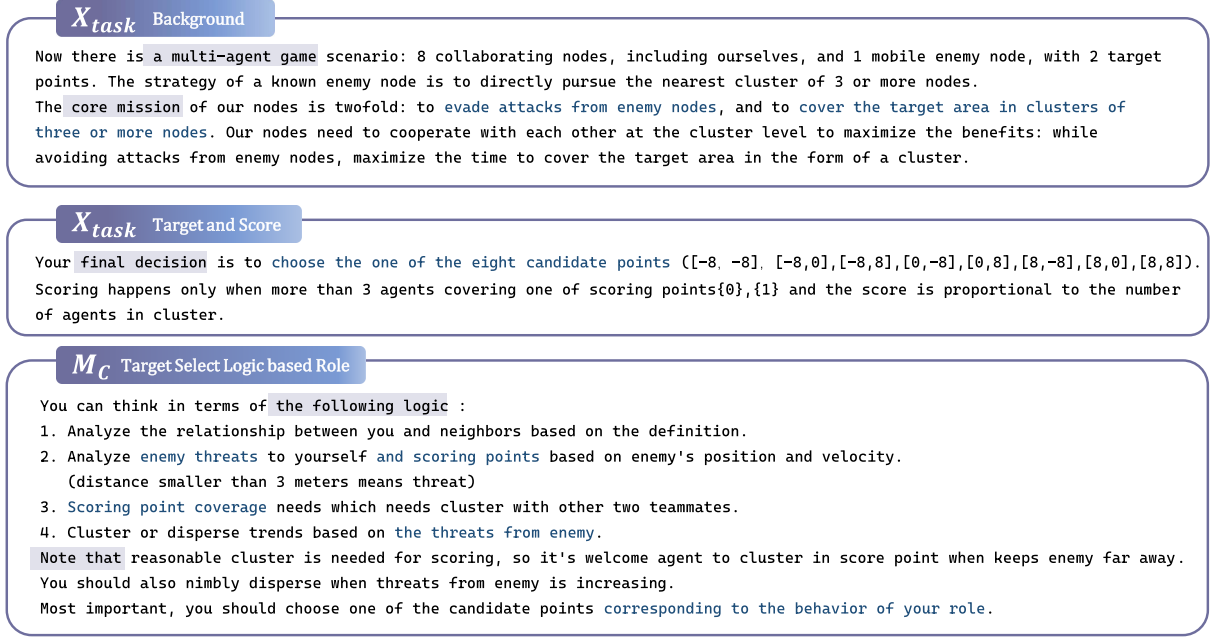


FIGURE 4. Example prompts used as inputs to the LLM.

self-reverts); Executor follows any available Commander/Coordinator. This design eliminates performance dips, and stabilizes system bounds by preventing collective deviations.

We compare the two-stage coordination policy $\pi_H^{\text{two}} = (\text{LLM}_{\text{cons}} \circ \text{LLM}_{\text{init}}) \rightarrow g_i^t$ against a one-stage baseline policy $\pi_H^{\text{one}} = \widetilde{\text{LLM}}_{\text{HC}} \rightarrow \tilde{g}_i^t$, where the symbol $\circ$ marks a strategy synergy operation. Beforehand, we introduce the following definition and assumption.

Definition 1 (Joint Policies Definition): For agent i , the action-value function at time t is:

$$\begin{aligned} Q_i(o_i^t, a_i^t) &= \mathbb{E} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} R(s^{t'}, a^{t'}) \mid o_i^t, a_i^t \right] \\ &= \mathbb{E} \left[\sum_{t'=t}^{\infty} \gamma^{t'-t} R(s^{t'}, a^{t'}) \mid o_i^t, k_i^t, g_i^t \right]. \end{aligned} \quad (9)$$

For global state s^t and all agents' goal decisions $\mathbf{g}^t := (g_1^t, \dots, g_N^t)$, the global value function Q_{tot} is defined by a mixing network f , which will be detailed in Section III-B:

$$Q_{\text{tot}}(s^t, \mathbf{a}) = f(Q_1, Q_2, \dots, Q_N; s^t, \mathbf{a}^t). \quad (10)$$

Assumption 1 (Monotonic Value Factorization): Under a monotonic value factorization [21], every weight in f is constrained to be non-negative, which guarantees the monotonicity property:

$$\frac{\partial Q_{\text{tot}}(s^t, \mathbf{g})}{\partial Q_i} \geq 0, \quad \forall i. \quad (11)$$

Assumption 2 (Performance Improvement of Extra Contextual Reasoning): Similar to [54], [55], [56], the extra

contextual reasoning helps guide the decision-making process toward more robust and effective outcomes, which leads to higher-quality Q -values.

Theorem 1 (Two-Stage Superiority): Under Assumption 1 and Assumption 2, the two-stage policy achieves strictly higher expected return:

$$J(\pi^{\text{two}}) > J(\pi^{\text{one}}), \quad (12)$$

provided there exists at least one reachable context where LLM_{cons} improves upon $\widetilde{\text{LLM}}_{\text{HC}}$.

Proof: Under Assumption 2, for any (o_i^t, k_i^t) , the two-stage consensus step selects

$$Q_i(o_i^t, k_i^t, g_i^t) \geq Q_i(o_i^t, k_i^t, g_i^{t'}), \quad \forall g_i^{t'} \quad (13)$$

by construction of LLM_{cons} . In contrast, the one-stage policy outputs $\tilde{g}_i = \widetilde{\text{LLM}}_{\text{HC}}(o_i^t, k_i^t)$ without refinement. In other words, Assumption 2 implies

$$Q_i(o_i^t, k_i^t, g_i^t) \geq Q_i(o_i^t, k_i^t, \tilde{g}_i^t). \quad (14)$$

Under Assumption 1, increasing any individual agent's value Q_i cannot decrease the total value, which implies:

$$Q_{\text{tot}}(s^t, \mathbf{g}^*) \geq Q_{\text{tot}}(s^t, \tilde{\mathbf{g}}), \quad (15)$$

with strict inequality if any agent achieves improvement. ■

Theorem 1 shows that under Assumptions 1 and 2, the two-stage policy $\pi_H^{\text{two}} = (\text{LLM}_{\text{cons}} \circ \text{LLM}_{\text{init}})$ achieves strictly higher expected cumulative reward than the one-stage baseline $\pi_H^{\text{one}} = \widetilde{\text{LLM}}_{\text{HC}}$. In other words, while it may not guarantee the optimal goal decision, the two-stage consensus process provides a more comprehensive evaluation than the initial decision, potentially leading to an improved target decision.

Algorithm 1: Credit-Assignment Enhanced Intention Generation Strategy (RMIX).

Require: Role candidate set

$\mathcal{K} = \{\text{Commander, Coordinator, Executor}\},$
RMIX parameters θ , target network $\bar{\theta}$,
Discount factor γ_{rmix} , learning rate α , batch size
 b ,
Experience replay buffer $\mathcal{B}_R$.

```

1: Step 1: Offline Experience Collection via LLM
2: for episode = 1 to  $N_{\text{pre}}$  do
3:   Initialize environment, observe  $\{o_i\}_{i=1}^n$ ;
4:   for each agent  $i$  do
5:      $k_{i\text{GPT}} \leftarrow \pi_{\text{GPT}}(o_i)$ ;  $\triangleright$  GPT assigns role
6:   end for
7:   Execute actions, receive reward  $R$ , state  $s$ ;
8:   Store  $\langle s, \{o_i, k_{i\text{GPT}}\}, R \rangle$  into buffer  $\mathcal{B}_R$ ;
9: end for
10: Step 2: Online Cooperative RMIX Training
11: for episode = 1 to  $N_{\text{epoch}}$  do
12:   Reset environment, observe  $\{o_i^0\}_{i=1}^N$ ;
13:   for  $t = 0$  to  $T - 1$  do
14:     for each agent  $i$  do
15:        $Q_i(o_i^t, k) \leftarrow \text{MLP}_{\theta}(o_i^t)$ ;
16:        $k_i^t \leftarrow \arg \max_{k \in \mathcal{K}} Q_i(o_i^t, k)$ ;
17:     end for
18:     Execute roles  $\{k_i^t\}$ , receive  $R^t, s^{t+1}, \{o_i^{t+1}\}$ ;
19:     Store  $\langle s^t, \{o_i^t, k_i^t\}, R^t, s^{t+1} \rangle$  into  $\mathcal{B}_R$ ;
20:     if  $|\mathcal{B}_R| \geq b$  then
21:       Sample batch  $\{(s^j, \{o_i^j, k_i^j\}, R^j, s^{j+1})\}_{j=1}^b$ ;
22:       Compute target values as follows:
23:        $y^j = R^j + \gamma_{\text{rmix}} \max_{k'} \bar{Q}(s^{j+1}, k'; \bar{\theta})$ ;
24:       Compute RMIX outputs  $Q_{\text{tot}}$  according to (17);
25:       Update:  $\theta \leftarrow \theta - \alpha \nabla_{\theta} \sum_j (y^j - Q_{\text{tot}}^j)^2$ ;
26:       Periodically update target:
27:        $\bar{\theta} \leftarrow \tau \theta + (1 - \tau) \bar{\theta}$ .
28:     end if
29:   end for
30: end for
31: return Trained RMIX network  $\text{RL}_{\text{HI}}$  for decentralized role selection

```

Next, we introduce role-based credit-assignment mechanism to ensure that Assumption 1 always holds.

B. CREDIT-ASSIGNMENT MECHANISM BASED ON ROLE-VALUE MIXING NETWORK

The underlying distributed nature of the DS-CEFC task makes a direct fine-tuning of LLM incompetent for reasonable multi-agent collaboration. As illustrated in Fig. 5, each distributed agent i feeds its current local observation o_i^t , and computes the obtainable optimal role satisfying:

$$k_i^t = \arg \max_{k_i^t \in \mathcal{K}} Q_i(o_i^t, k_i^t; g_i^t). \quad (16)$$

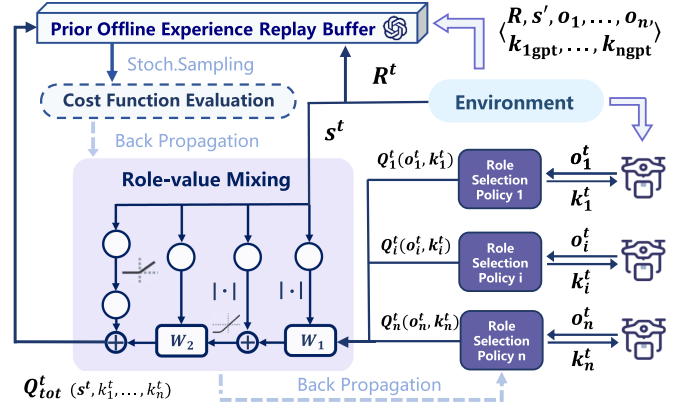


FIGURE 5. The training of the RMIX-based role selection mechanism. Each UAV employs a fully distributed role selection policy, using local observation o_i^t to select role k_i^t , and computes corresponding $Q_i(o_i^t, k_i^t)$. These Q_i are aggregated into the global action-value Q_{tot} via the RMIX network and stored in a prior offline experience replay buffer. Function evaluation cost is computed using the buffer data, followed by gradient backpropagation to update both the RMIX network and role selection policies.

By (9), in dynamic multi-agent adversarial tasks, instantaneous reward R cannot directly reflect the contribution from the choice of role k_i^t . To address this issue, we propose an RMIX-assisted credit-assignment mechanism to evaluate each agent's role choices. Also, we consider the significant inference latency of LLM, which results in higher costs for acquiring training samples. In particular, with 8 agents and 3 roles each, the dimension of joint role space turns 3^8 , and when combined with sparse, adversarial rewards, this makes pure online exploration extremely difficult. Therefore, we resort to an offline learning approach to improve sample efficiency. Similar to Curriculum Learning (CL) [57], we first exploit the zero-shot capability of LLM (e.g., GPT-4o [58]) via API to obtain its role assignment strategy π_{GPT} . To clarify, π_{GPT} refers to the online GPT4o-API version, rather than any locally fine-tuned large model. By interacting with the environment under local observation o_i^t , the mid-layer policy π_M and the physical flight controller π_L , we record GPT's preliminary role suggestions $k_{i\text{GPT}}$ and store the related transition quadruple $\langle o_i, k_{i\text{GPT}}, s', R \rangle$ into a replay buffer $\mathcal{B}_R$, where s' denotes the transitioned state. This offline data collection runs in parallel with LLM consensus reasoning and seeds the replay buffer with sensible role assignments, enabling RMIX to assist the LLM in understanding role assignments and reducing the cold-start overhead from ineffective exploration.

Inspired by [21], RMIX aggregates individual role-value estimates $Q_i(o_i^t, k_i^t)$ into a global value Q_{tot} via a two-layer Multi-Layer Perceptron (MLP). Let $\mathbf{Q}^t = [Q_1(o_1^t, k_1^t), \dots, Q_N(o_N^t, k_N^t)]$. A small hypernetwork conditioned on the global state s^t produces nonnegative weights, by enforcing:

$$Q_{\text{tot}}^t = \text{ReLU}(\mathbf{W}_2^\top (\text{ReLU}(\mathbf{W}_1 \mathbf{Q}^t + \mathbf{b}_1)) + \mathbf{b}_2), \quad (17)$$

where the two nonnegative weight vectors $\mathbf{W}_1 \in \mathbb{R}^{E \times N}$ and $\mathbf{W}_2 \in \mathbb{R}^{E \times 1}$, while biases $\mathbf{b}_1 \in \mathbb{R}^{E \times 1}$ and $\mathbf{b}_2 \in \mathbb{R}$, with E

Algorithm 2: Context Transfer Enhanced Lightweight Large Model Consensus Inference Algorithm.

Require: Task prompt X_{task} , prompt generator $Y(o_i)$, GPT-4o API, target set $\mathcal{G}_a$, rewards R , thresholds $\{\tau_r, L_{\min}, L_{\max}\}$, weights $\{w_{1,g}, \dots, w_{4,g}\}$, minimum samples M , batch size B , LoRA params ψ .

- 1: $\mathcal{B}_{\text{LLM}} \leftarrow \emptyset$
- 2: **while** $|\mathcal{B}_{\text{LLM}}| < M$ **do**
- 3: Observe (o_i) ;
- 4: $(g_i^*, \mathcal{J}_i^*) \leftarrow \text{GPT4o}(X_{\text{task}}, Y(o_i))$ cf. Eq. (20);
- 5: Append $(o_i, k_i, g_i^*, \mathcal{J}_i^*, R)$ to $\mathcal{B}_{\text{LLM}}$;
- 6: **end while**
- 7: $\mathcal{B}_{\text{fil}} \leftarrow \{x \in \mathcal{B}_{\text{LLM}} \mid \text{Check}(x) \geq 1\}$ cf. Eq. (21);
- 8: **for** epoch=1 **TO** N_{epoch} **do**
- 9: Sample batch $\mathcal{B} \subset \mathcal{B}_{\text{fil}}$ of size B ;
- 10: Compute loss via Eq. (22);
- 11: Update $\psi \leftarrow \psi - \eta \nabla_{\psi}$;
- 12: **end for**
- 13: **return** Fine-tuned LLM $_{\psi}$

denoting the hidden layer dimension of RMIX. $\text{ReLU}(\cdot)$ indicates the ReLU nonlinear activation function. Eq (17) ensures the monotonic mapping between Q_{tot}^t and Q^t , thus satisfying the Assumption 1.

Subsequently, we proceed to the standard CTDE online training. RMIX jointly learns from both offline and newly collected online samples to continuously explore and refine cooperative behaviors among agents. Generally, RMIX can be solved by a classical Stochastic Gradient Descent (SGD) approach, with the cost function of RMIX being formulated as:

$$L(\theta) = \sum_{i=1}^{|\mathcal{B}_R|} \left[(y_i^{\text{tot}} - Q_{\text{tot}}(s^t, k^t; \theta, g^t))^2 \right], \quad (18)$$

where θ denotes the concatenation of $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1$ and $\mathbf{b}_2$ while $|\mathcal{B}_R|$ denotes the number of samples sampled from the empirical memory $\mathcal{B}_R$. As an approximate estimation of the cumulative returns under the global state s^t , y^{tot} is given as:

$$y^{\text{tot}} = R^t + \gamma_{\text{rmix}} \max_{k'} \bar{Q}(s', k'; \bar{\theta}, g^t), \quad (19)$$

where for the transitioned state s' and the taken action k' , $\bar{Q}(s', k'; \bar{\theta})$ denotes the target network parameterized by $\bar{\theta}$ and γ_{rmix} is the discount factor.

Finally, Algorithm 1 summarizes the procedure of RMIX.

C. CONTEXT-BASED LLM FINE-TUNING

Albeit the proficiency of LLMs like GPT-4o [58], a lightweight version is preferred by resource-constrained multi-agent system. Therefore, through self-generated instruction tuning [59], we introduce a capacity-migration augmentation, so as to improve a smaller model's reasoning ability. Through a concerted effort, we successfully transfer task-understanding capabilities from a State-Of-The-Art

(SOTA) foundation model to a local version, and subsequently compress the model to under 5 GB of memory usage, thus potentially enabling distributed inference of the consensus reasoning module on onboard UAV GPUs [60].

Due to the limited dataset in DS-CEFC, we call an online GPT-4o model to generate samples as:

$$X_{\text{task}}, Y(o_i') \rightarrow g_i^*, \mathcal{J}_i^*, \quad (20)$$

where g_i^* denotes the manually labeled decision, and $\mathcal{J}_i^*$ constitutes additional consensus inference outputs, stored beyond Eq. (7), specifically for training purposes. In particular, to guide the generation of desired output in (8), human-annotated instructions, similar to those in Fig. 4 and 11, are provided. On this basis, GPT-4o's output g_i serves as RALLY's consensus decision for selecting the agent's target region. We then interact with the mid-layer policy π_M and the physical flight controller π_L to obtain the next local observation o_i^{t+1} and role k_i^{t+1} . This process repeats until acquiring a dataset $\mathcal{B}_{\text{LLM}}$ with sufficient inference samples and trajectory data $(o_i, k_i, g_i^*, \mathcal{J}_i^*, R)$.

The raw datasets $\mathcal{B}_{\text{LLM}}$ may contain low-quality or invalid outputs. Therefore, we apply a filtering mechanism to retain a high-quality subset $\mathcal{B}_{\text{fil}}$. To weed out low-quality samples, we check whether the yielded target region g_i^* belongs to the target set $\mathcal{T}$, the length of the inference content $\mathcal{J}_i^*$ is illegal with anomalous symbols, and it conforms to the task constraints based on the r_i . Mathematically,

$$\begin{aligned} \text{Check}(g_i^*, \mathcal{J}_i^*, R) = & \mathbb{I}(g_i^* \in \mathcal{G}_a) \cdot w_{1,g} \\ & + \mathbb{I}(\mathcal{J}_i^* \cap \Lambda = \emptyset) \cdot w_{2,g} \\ & + \mathbb{I}(L_{\min} \leq |\mathcal{J}_i^*| \leq L_{\max}) \cdot w_{3,g} \\ & + \mathbb{I}(R \geq \tau_r) \cdot w_{4,g}, \end{aligned} \quad (21)$$

where $\mathbb{I}(\cdot)$ is an indicator function that returns 1 when the condition is established and 0 otherwise; $w_{1,g}, w_{2,g}, w_{3,g}, w_{4,g}$ are the indicator importance weights; Λ is a pre-defined set of anomalous characters; $L_{\min}$ and $L_{\max}$ denote the minimum and maximum inference result lengths, respectively.

Notably, the fine-tuning aims to minimize the mean squared error (MSE) between model outputs and GPT-4o reference samples, that is:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{|\mathcal{B}_{\text{fil}}|} \sum_{(g_i^*, \mathcal{J}_i^*) \in \mathcal{B}_{\text{fil}}} \left(\text{LLM}_{\psi}(X_{\text{task}}, Y(o_i)) - (g_i^*, \mathcal{J}_i^*) \right)^2, \quad (22)$$

where ψ denotes the low-rank adaptation parameters. To reduce computation and storage requirements, we use the LLaMA-Factory framework [61] with LoRA [62] to fine-tune a smaller, ψ -parameterized LLM (e.g., Qwen2.5 or Llama3.1).

Finally, the pseudocode is presented in Algorithm 2.

TABLE 2. Key Environmental and Reward Function Parameter Configurations

Parameter	Setting
No. of UAVs N	{8, 9, 10, 11}
Formation patterns set $\mathcal{C}$	{3, 4, 5, 6, 7, 8}
UAV velocity range (m/s)	[-1, 1]
Adversary velocity range (m/s)	[-0.75, 0.75]
Observation distance δ_{obs} (m)	3
Weights $\omega_f, \omega_n, \omega_{tc}, \omega_e, \omega_c$ of reward R	(15, 4, 10, 100, 100)
Decrease factor ω_d	0.003
Joint policy discount factor γ	0.92
RMIX discount factor γ_{mix}	0.95
RMIX learning rate α	1×10^{-5}
RMIX hidden layer dimension E	128
Thresholds $\{\tau_r, L_{\min}, L_{\max}(\text{token})\}$	(-3, 000, 200, 400)
LoRa weights $w_{1,g} \dots w_{4,g}$	(0.45, 0.25, 0.2, 0.1)
Minimum No. of samples M	12,000

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

A. EXPERIMENTAL SETTINGS

In this section, we evaluate the effectiveness and superiority of RALLY on DS-CEFC tasks in MPE [49] and a fully distributed, high-fidelity SITL [50] simulation built on Gazebo-ROS-PX4. Concretely, each simulation runs for 1,000 steps, wherein every 50 steps, UAVs select their target regions according to π_H . The set of possible targets is $\mathcal{G}_a = \{(p_x, p_y) | p_x, p_y \in [-8, 8]\}$ with both coordinates initialized uniformly in $[-8, 8]$ m and re-sampled when a region's urgency reaches zero. The urgency level κ_{tr}^t is initialized as 1. Whenever there exists a formation of UAVs covering the region $tr \in \mathcal{T}$, κ_{tr}^t will decrease linearly with a scale ω_d until zero; otherwise, it remains unchanged. The pursuer's speed is set to be twice that of the evader to ensure it has the necessary mobility to chase down and capture the evaders effectively. Other key environment and mission parameters are listed in Table 2.

For RALLY, the RMIX network is implemented as a two-layer MLP with a hidden layer size of 16. During training, we use a batch size of 256, and update the target network with soft updates at a rate of 0.01. For the capacity-migration-augmentation strategy, we create $L = 50$ manually labeled examples to serve as the few-shot corpus. Given the inference latency, we set the few-shot sample count to $\rho = 1$. During the API-driven data collection, the GPT-4o model is leveraged to provide high-quality inferences. After simulation and filtering, we accumulate $|\mathcal{B}_{\text{fil}}| = 8,231$ samples for fine-tuning the local model Qwen2.5-7B. The adversary uses a PPO policy [52] with the same network structure as π_M , learning to chase the nearest cluster of at least three agents while avoiding obstacles.

We compare RALLY against three representative baselines, CIHRL [2], CoNavGPT [30], and DITTO [51], to evaluate its effectiveness on the DS-CEFC task. CIHRL [2], which does not incorporate role assignment, incorporates multi-agent communication and belongs to the SOTA MARL approach

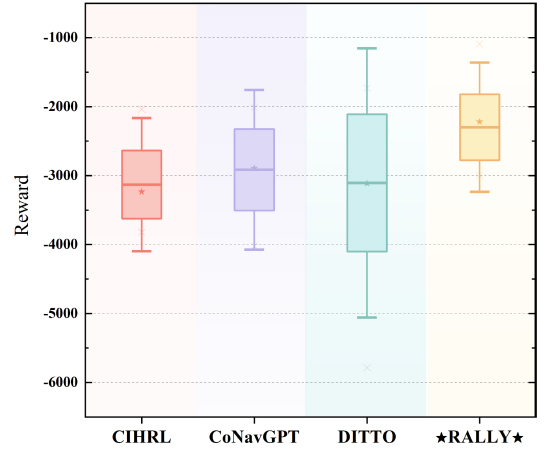


FIGURE 6. Performance comparison between RALLY and baselines.

for DS-CEFC. CoNavGPT [30] employs an LLM as a global planner without any training process, achieving high success rates and efficiency on the navigation task. DITTO [51] achieves good collaboration based on LLM, demonstrating strong role-based heterogeneity. All the non-API LLMs are deployed in a distributed fashion on 8 NVIDIA GeForce RTX 4090 GPUs (24 GB each).

B. PERFORMANCE COMPARISON WITH BASELINES

1) OVERALL PERFORMANCE

Fig. 6 presents the average rewards¹ over 30 test episodes and demonstrates the impact of consensus mechanisms on task completion. Particularly, RALLY attains the highest mean reward and the narrowest variance distribution, indicating minimal variance, fastest convergence, and consistent task completion across various episodes. In contrast, CIHRL behaves conservatively, yielding stable but modest rewards. CoNavGPT achieves higher average rewards than CIHRL, confirming the LLM's strong environmental understanding and decision-making, but it lacks online exploration and can get stuck in local optima. DITTO slightly outperforms CIHRL by using LLM-based self-cognition for role and action selection, yet its greedy role choices and absence of reinforcement feedback lead to high variance and unstable consensus. These results demonstrate that RALLY's integration of RL-based environmental feedback and LLM-driven semantic decision making effectively guides the multi-UAV swarm to reach robust consensus and execute high-quality collaboration in complex dynamic environments.

Next, we study the convergence of the credit-based role assignment mechanism. Particularly, "RMIX" uses the MLP-based fusion network for joint utility estimation detailed in Section III-B, while "VDN" aggregates Q_i via a simple weighted sum $Q_{\text{tot}} = \sum_{i=1}^n w_i Q_i(o_i, k_i)$. Fig. 7 presents

¹Consistent with our previous works Ref. [2], due to the large negative values of interference penalty R_c^t , and collision penalty R_c^t , the reward defined in RALLY is generally negative as well.

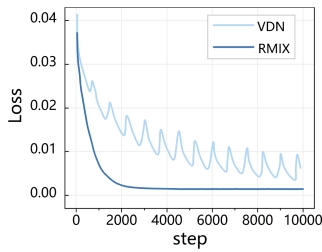


FIGURE 7. Training based on RMIX.

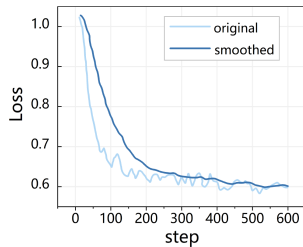


FIGURE 8. Fine-tuning of Qwen2.5-7B.

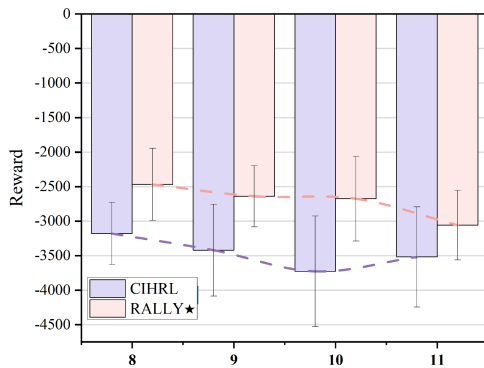


FIGURE 9. Generalization of RALLY to varying numbers of agents.

the corresponding results. Although both methods converge, RMIX converges faster and yields more accurate cumulative return estimates. Fig. 8 shows the LoRA fine-tuning loss curve, while the validation starting from step 500 indicates successful convergence.

2) GENERALIZATION

As depicted in Fig. 9, we evaluate RALLY and CIHRL with changing swarm sizes from 8, 9, 10 to 11 on the DS-CEFC task. As the swarm grows beyond this training configuration, CIHRL's performance degrades substantially. In other words, without retraining for larger formations, the involved agents fall into “habitual grouping” patterns that repeatedly form the learned clusters, preventing effective coverage of additional targets and leading to significant score losses. In contrast, RALLY, which encodes the maximum permitted formation size into its prompt, dynamically forges a split consensus to avoid excessive clustering. More importantly, RALLY preserves high scoring ability even as the swarm size increases to

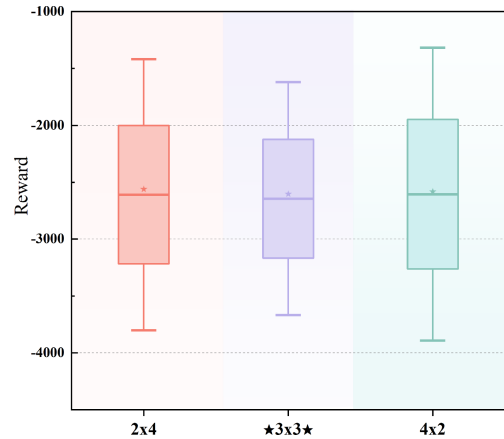


FIGURE 10. Generalization of RALLY to varying target areas.

9, 10, and 11. This clear contrast underscores RALLY's superior generalization to larger, unseen formations than CIHRL.

We further evaluate RALLY across three different target area configurations: the original 3×3 grid, a 2×4 grid, and a 4×2 grid. The target locations in each scenario are randomly generated, ensuring dynamic and diverse environments. As shown in Fig. 10, RALLY performs consistently well across all three scenarios, with no significant difference in reward performance, reinforcing its ability to adapt to varying environmental conditions.

C. PERFORMANCE ANALYSIS OF RALLY

1) CONTRIBUTION OF RMIX

To illustrate how RMIX enhances LLM semantic decision making, we examine the differences between the initial intention and inferred consensus of Agent #6, as shown in Fig. 11. In the initial LLM-only phase, Agent #6 computes its distance to the adversary (≈ 6.86 m), and driven by “safe aggregation” and “maximal scoring”, greedily adopts the Commander role with target $(-8, 8)$. While this choice yields short-term points, it overlooks team coordination and can destabilize consensus under complex threats. Therefore, the role-selected policy overrides this role to Coordinator, shifting the agent's motive from individual dominance to supporting and scheduling. Receiving a neighbor's intent (also $(-8, 8)$), the LLM then recommends $(-8, 8)$ again. Moreover, the LLM also issues explicit role alignment that maintains both proximity to the Executor and collaboration with the Commander. This refined decision reuses the initial safety assessment and integrates multi-party communication via role mapping, yielding a more holistic trade-off between individual scoring and team consistency. By correcting the LLM's isolated “Commander” bias, RALLY preserves LLM's semantic planning strengths while injecting MARL's distributed division of responsibility and information fusion. Consequently, the two-stage output achieves superior policy stability and collaborative performance in the dynamic DS-CEFC task.

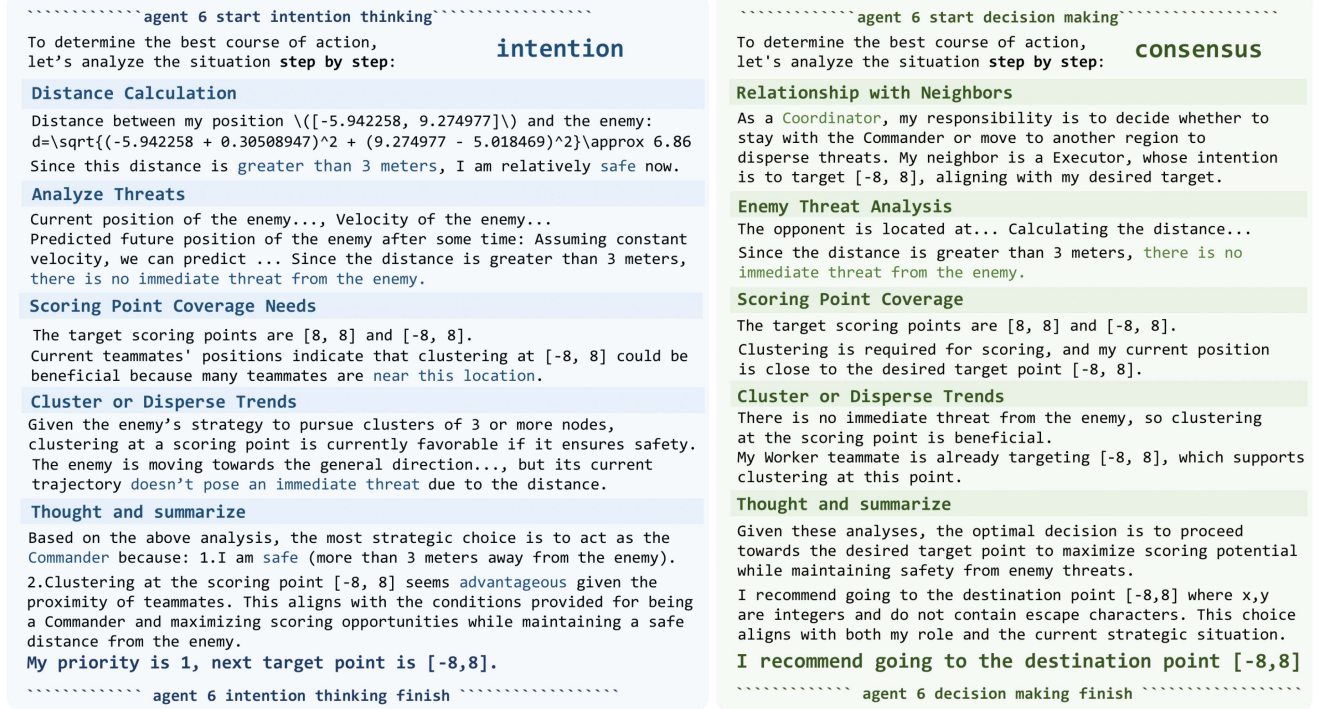


FIGURE 11. A two-phase decision-making procedure executed by Agent #6, where both stages are derived from the LLM outputs.

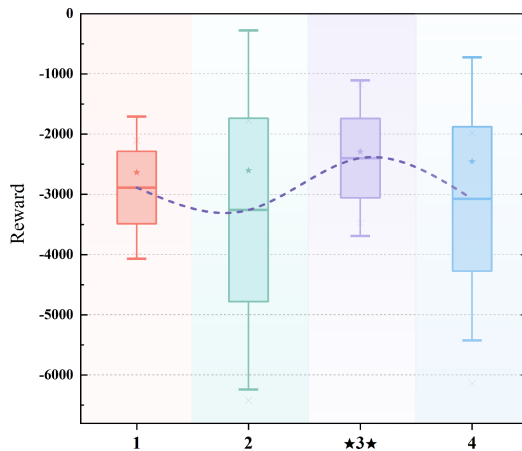


FIGURE 12. Impact of four different role configurations, including single role (Executor), two roles (Commander-Executor), three roles (Commander-Coordinator-Executor), and four roles (Commander-Coordinator-Executor-Decoy).

2) IMPACT OF ROLE NUMBER

Fig. 12 illustrates RALLY's reward distributions under four different role configurations, including single role (Executor), two roles (Commander-Executor), three roles (Commander-Coordinator-Executor), and four roles (Commander-Coordinator-Executor-Decoy), where the Decoy role is specifically designed to divert enemy attention. The single-role setup achieves the lowest mean reward ($\approx -3,000$), or severely limited performance

due to the absence of task decomposition and limited exploration-coverage trade-off. Introducing a dual-role hierarchy (Commander-Executor) yields a slight increase in mean reward but greatly enlarged whiskers, indicating that overreliance on the Commander's decisions amplifies fluctuation and undermines group synergy. In contrast, the three-role configuration combines high average performance with smaller variance, demonstrating that introducing a Coordinator role effectively mediates the semantic planning benefits from the LLM while enhancing consistency and robustness through MARL's exploration and credit-assignment. Unfortunately, adding a fourth Decoy role reduces average reward and inflates variance, suggesting that excessive role granularity raises coordination overhead and consensus costs, thereby detracting from overall effectiveness. Overall, the three-role (Commander - Coordinator - Executor) architecture strikes the optimal balance between performance and stability within our two-phase LLM-MARL convergence framework, fully leveraging semantic decision making and reinforcement-driven exploration to achieve superior formation coverage and convergence stability.

3) LLM FINETUNING

Fig. 13 present the fine-tuning performance for RALLY and compare with other models. It can be observed from Fig. 13 that a non-fine-tuned Qwen2.5-7B base model delivers markedly lower performance, while due to its occasional illegal outputs (e.g., "I suggest going to target point #8, #8." or "region 8"), a direct calling of GPT-4o for online interaction

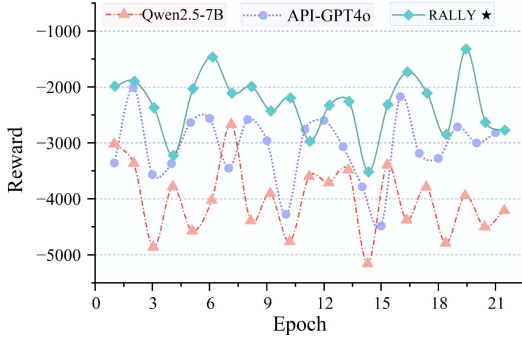


FIGURE 13. Inference performance comparison after model fine-tuning.

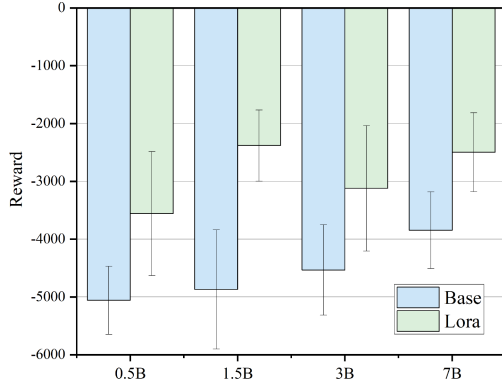


FIGURE 14. Impact of model parameter scale.

TABLE 3. Running Performance of Mainstream LLMs on an NVIDIA RTX 4090

Model	AIT (s)	MF (GB)	RO (GB)
Qwen2.5-7B	15.39	15	15.7
Qwen2.5-3B	17.63	5.8	7.17
Qwen2.5-1.5B	14.48	2.9	4.13
Qwen2.5-0.5B	15.45	1.2	1.77

possibly result in significant latency and network instability issues, degrading the performance. By contrast, RALLY, which fine-tunes Qwen2.5-7B base model under the dataset $\mathcal{B}_{\text{fil}}$, harmonizes the high-quality inference of API-GPT-4o with the efficiency and stability of a smaller model.

Next, we evaluate the performance after fine-tuning a LoRA-based family of Qwen2.5 models with varying parameter counts (0.5-B, 1.5-B, 3-B, and 7-B) on the DS-CEFC task. Fig. 14 summarizes the post-fine-tuning performance and demonstrates substantial performance gains. Furthermore, Table 3 reports the Average Inference Time (AIT), Memory Footprint (MF), and Runtime Overhead (RO) after running these models on an NVIDIA RTX 4090. It can be found that a Qwen2.5-1.5-B version strikes the balance by delivering robust decision quality with minimal inference overhead.

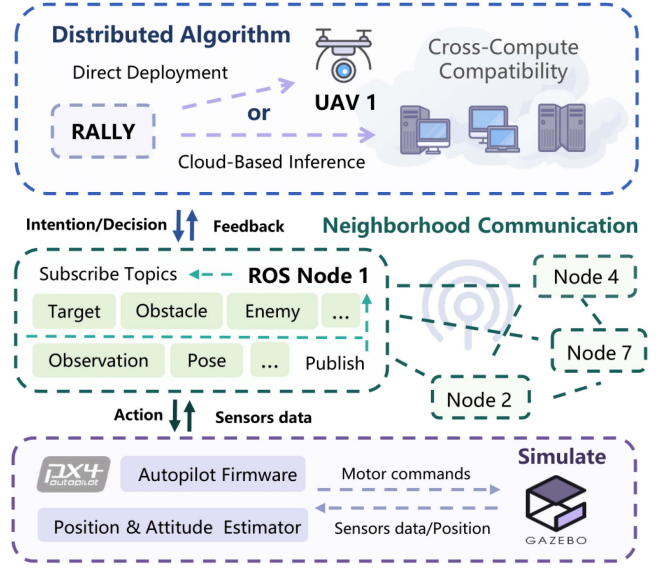


FIGURE 15. Task overview in Gazebo Simulator for SITL.

D. SOFTWARE-IN-THE-LOOP VALIDATION

To assess RALLY's real-world viability, we integrate it into a ROS-based SITL environment featuring Gazebo-Classical and the PX4 autopilot, as illustrated in Fig. 15. Furthermore, each UAV follows standard quadrotor dynamics [63]. Consistent with the mainstream MARL-based UAV studies, we assume fixed-altitude flight that $u_z \equiv 0$. Given the desired horizontal acceleration $\mathbf{u} = [u_x, u_y]$ from π_M , the PX4 flight controller will use a PID scheme [53] to compute thrust and angular rate commands, followed by a physics simulator which integrates the Newton–Euler equations [63] to update each UAV's pose and dynamic state. Unlike prior open-source frameworks such as XT-Drone, our setup enforces fully distributed decision-making: each quadcopter node operates on local observations within a limited communication radius amid multiple obstacles, scoring zones, and predator–prey interactions. Concretely, UAV #1 (and each peer) runs an independent off-board Python controller. The high-level RALLY consensus module adopts the aforementioned 1.5-B fine-tuned Qwen2.5 model, while the mid-level and low-level PX4 flight-control modules execute on a ground server to generate navigation commands. Using MAVROS over UDP, each UAV publishes its state and sensory data (including adversary, obstacle, target, and neighbor information) and subscribes to receive pertinent updates. The desired target region, output by RALLY, is then converted into horizontal accelerations and broadcast to PX4 via ROS topics. PX4, connected to Gazebo through TCP, receives these acceleration demands, computes motor and actuator setpoints via its PID controllers, and returns updated UAV poses and sensor readings for the next simulation step. This tightly coupled loop in Fig. 17 contributes to validating RALLY's distributed consensus and navigation performance under realistic quadcopter constraints.

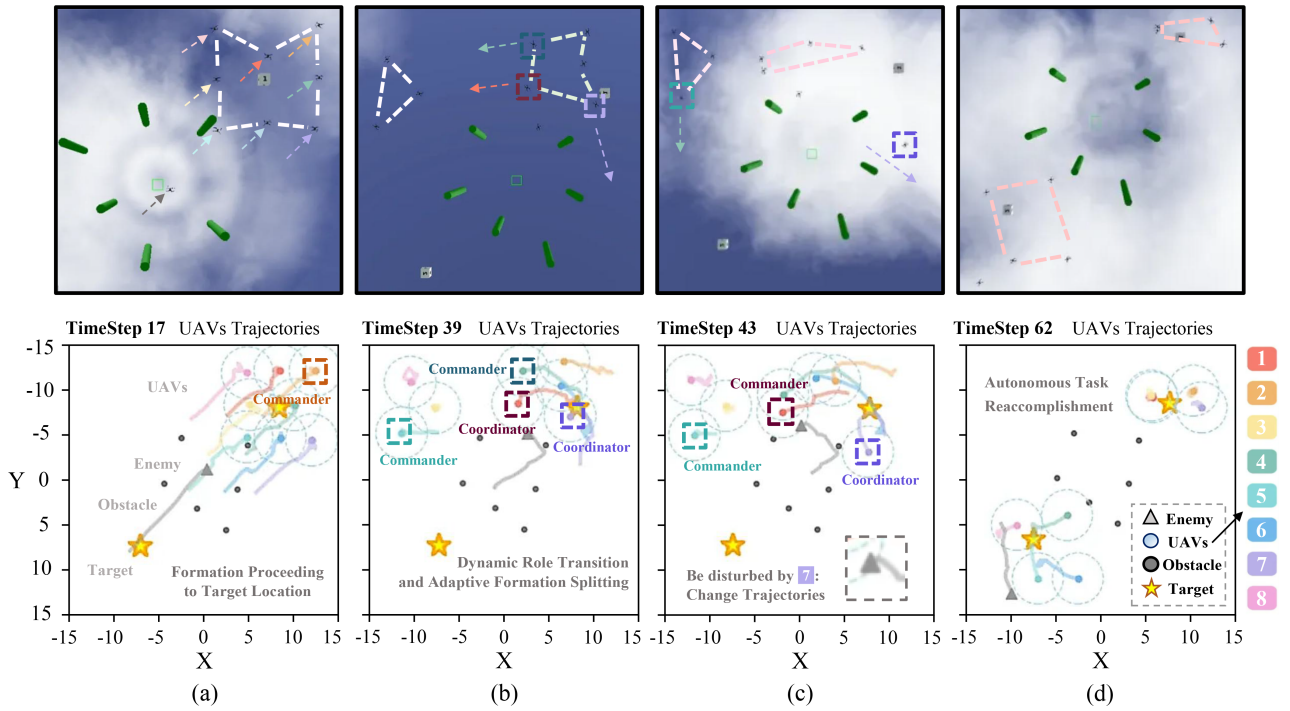


FIGURE 16. Four typical cooperation scenarios in one episode of SITL. (a), (b), (c), and (d) stand for snapshots at high-level decision steps 17, 39, 43, and 62, respectively.

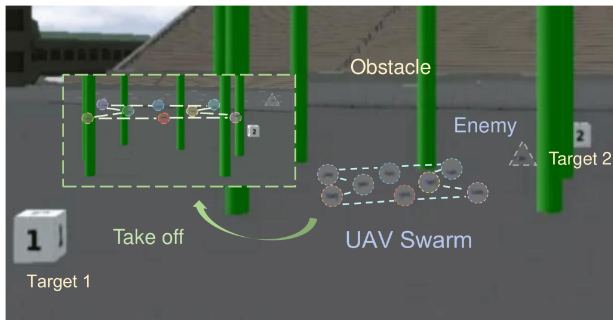


FIGURE 17. Initialization of the Gazebo-ROS-PX4 simulation environment.

Fig. 16 illustrates four representative consensus-building steps in the SITL environment, overlaid on each UAV's flight path derived from Gazebo-Classic. Each simulation time-step corresponds to one decision frame for consensus refinement. At the time-step 17 (Fig. 16(a)), UAV #2—being closest to Target #2 and farthest from the enemy—assumes the Commander role and selects the upper-right scoring zone. All teammates comply and proceed toward Target #2. At the time-step 39 (Fig. 16(b)), the approaching enemy forces a reconfiguration: UAVs 3, 5, and 8 split off as a three-agent squad (F_3), with UAV #5 promoted to Commander and guiding its group to Target #1. The remaining five UAVs form a separate F_5 team; UAV #4 takes on the Commander role and, alongside UAV #1 acting as Coordinator, leads its squad toward the same goal. UAV #7 selects to serve as

Coordinator, deliberately positioning itself between the adversary and the cluster to divert attention and safeguard its peers. At the time-step 43 (Fig. 16(c)), UAV #7's unexpected directional shift effectively confuses the enemy's pursuit vector, causing it to veer off and granting the other drones a clear corridor to bypass the threat and reorient toward the next target. Finally, at the time-step 62 (Fig. 16(d)), after successive rounds of LLM-driven intent generation and RMIX-guided role reassignment, both sub-clusters successfully evade the enemy and complete coverage of their respective scoring regions. This sequence confirms RALLY's ability to orchestrate dynamic role adaptation and robust distributed consensus in complex, adversarial scenarios.

V. CONCLUSION AND FUTURE WORKS

This paper introduces RALLY, an advanced LLM-MARL-integrated framework for UAV swarm control that combines LLM-driven semantic reasoning with MARL-based exploration. By integrating autonomous intent understanding, dynamic role assignment, and decentralized consensus building, RALLY enables each UAV to interpret local observations, select functional roles, and collaboratively decide on navigation goals under communication constraints. We validate RALLY via the MPE simulation environment and a high-fidelity Gazebo-ROS-PX4 SITL platform, demonstrating its superior task completion, collaborative effectiveness, and generalization compared to existing methods in the DS-CEFC scenario.

M_C Role Introduction

First, you should select your role in one of [commander, coordinator, executor] represented as number 1, 2, 3 and then choose one of the candidate points based on your role. Below are the roles' responsibility.

1. The commander will be firm in his opinions and do what is best for him in the decision-making stage.
2. The coordinator judges gains and losses depending on commander's intention and conduct executors to ensure the interests of the commander.
3. The executor always follows intention and choose the coordinator's target points.

 M_C Role Select Logic

When you are closer to the enemy than neighbor and it's better to change another points for covering scoring neighbor, you are supposed to be the coordinator and lead other neighbors.

If you are safe (away from enemy over 3 meters) and around target points{0},{1}, better to be the commander to maximize score.

If you are in a dilemma to make decision, you are encouraged to be the executor.

It is recommended that you answer by simple calculations and semantic analysis.

 $Y_{init}(o_i^t)$

The current scoring points are {} and {}, you are now in {} position. Given that your teammates are at {}, other {} teammates are out of communication range, and your opponents are at {} with velocity {}, you choose which target point to go to.

Step1 Output_Format

The last line requires output in the form: "My priority is [x], next target point is [y,z]."

where x can be 1, 2, 3 and [y,z] are integers and do not have escape characters.

 $Y_{cons}(z_i^t)$

Now only points {} and {} are scoring points, you are now in {} position and your role is {} with desired target point {}.

Given that your teammates are at {}, other {} teammates are out of communication range. Neighbors' roles are {} and prefer to targets {} respectively. Your opponents are at {} with velocity {}, you choose which target point to go to.

Step2 Output_Format

The last line requires output in the form: "I recommend going to the destination point [x,y]."

where x,y are integers and do not contain escape characters.

FIGURE 18. Part of the prompts used in RALLY.

Now only points [8. 8.] and [-8. 8.] are scoring points.

You are now in [-5.3419337 7.6574435] position and your role is **Commander** with desired target point:[-8. 8.].

Given that your teammates are at [[-9.43643379 9.16500187]], other 6 teammates are out of communication range.

Neighbors' roles are ['Executor'] and prefer to targets [array([-8. 8.])] respectively.

Your opponents are at [6.145471 -0.80852365] with velocity [-0.0719887 0.79896355].

LLM Final output: "I recommend going to the destination point [-8. 8.]."

Now only points [8. 8.] and [-8. 8.] are scoring points.

You are now in [7.85132313 6.48642683] position and your role is **Coordinator** with desired target point:[8. 8.].

Given that your teammates are at [[6.3594694 5.16088724]]\n [9.32564354 7.93882751]

\n [10.01850128 5.27130747]], other 4 teammates are out of communication range. Neighbors' roles are ['Executor', 'Commander', 'Coordinator'] and prefer to targets [array([8. 8.]), array([8. 8.]), array([8. 8.])] respectively.

Your opponents are at [6.145471 -0.80852365] with velocity [-0.0719887 0.79896355].

LLM Final output: "I recommend going to the destination point [8. 8.]."

Now only points [8. 8.] and [-8. 8.] are scoring points.

You are now in [6.3594694 5.16088724] position and your role is **Executor** with desired target point:[8. 8.].

Given that your teammates are at [[7.85132313 6.48642683]], other 6 teammates are out of communication range.

Neighbors' roles are ['Coordinator'] and prefer to targets [array([8. 8.])] respectively.

Your opponents are at [6.145471 -0.80852365] with velocity [-0.0719887 0.79896355].

LLM Final output: "I recommend going to the destination point [8. 8.]."

FIGURE 19. Specific prompts and corresponding responses by different agents running their distributed models in parallel. For brevity, we have omitted the repeated structural prompts and CoT reasoning, while retaining only the key data-containing components of the input and output.

Looking forward, we plan to optimize the lightweight LLM for faster inference and reduced communication latency, and enhance system robustness through more large-scale settings and advanced communication strategies. In addition, we will address the possible local optima issue in CoT reasoning by exploring test-time training strategies and diversifying reasoning paths to improve reasoning diversity, reduce convergence to suboptimal solutions, and enhance generalization. We also aim to investigate multimodal fusion and theoretical guarantees for rapid semantic consensus in larger UAV swarms. These efforts will pave the way for deploying intelligent, collaborative UAV systems in complex, resource-constrained missions.

APPENDIX

In the Appendix, we provide the detailed prompts in Fig. 18 and give the reasoning sensitivity in Fig. 19.

REFERENCES

- [1] T. Uzakov, T. P. Nascimento, and M. Saska, "UAV vision-based nonlinear formation control applied to inspection of electrical power lines," in *Proc. Int. Conf. Unmanned Aircr. Syst.*, Athens, Greece, 2020, pp. 1301–1308.
- [2] Y. Xiang, S. Li, R. Li, Z. Zhao, and H. Zhang, "Decentralized consensus inference-based hierarchical reinforcement learning for multi-constrained UAV pursuit-evasion game," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 18, 2025, doi: [10.1109/TNNLS.2025.3582909](https://doi.org/10.1109/TNNLS.2025.3582909).
- [3] Y. Xia, J. Zhu, and L. Zhu, "Dynamic role discovery and assignment in multi-agent task decomposition," *Complex Intell. Syst.*, vol. 9, pp. 6211–6222, Apr. 2023.
- [4] Y. Jin and Q. Liu, "Multi-agent self-motivated learning via role representation," in *Proc. Int. Joint Conf. Neural Netw.*, Yokohama, Japan, 2024, pp. 1–7.
- [5] G. Vásárhelyi et al., "Outdoor flocking and formation flight with autonomous aerial robots," in *Proc. Int. Conf. Intell. Robot. Syst.*, Chicago, IL, USA, 2014, pp. 3866–3873.
- [6] F. F. Lizzio, E. Capello, and G. Guglieri, "A review of consensus-based multi-agent UAV implementations," *Intell. Robot. Syst.*, vol. 106, no. 2, Oct. 2022, Art. no. 43.
- [7] Y. Yan et al., "Relative distributed formation and obstacle avoidance with multi-agent reinforcement learning," in *Proc. Int. Conf. Robot. Automat.*, Philadelphia, PA, USA, 2022, pp. 1661–1667.
- [8] K. Lu et al., "Rethinking modern communication from semantic coding to semantic communication," *IEEE Wireless Commun.*, vol. 30, no. 1, pp. 158–164, Feb. 2023.
- [9] Z. Lu, R. Li, M. Lei, C. Wang, Z. Zhao, and H. Zhang, "Self-critical alternate learning-based semantic broadcast communication," *IEEE Trans. Commun.*, vol. 73, no. 5, pp. 3347–3363, May 2025.
- [10] P. Sunehag et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proc. 17th Int. Conf. Auton. Agents MultiAgent Syst.*, Stockholm, Sweden, Jul. 2018, pp. 2085–2087.
- [11] A. Das et al., "TarMAC: Targeted multi-agent communication," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, Jun. 2019, vol. 97, pp. 1538–1546.
- [12] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proc. Int. Conf. Mach. Learn.*, Amherst, MA, USA, Jul. 1993, pp. 330–337.
- [13] A. Oroojlooy and D. Hajinezhad, "A review of cooperative multi-agent deep reinforcement learning," *Appl. Intell.*, vol. 53, no. 11, pp. 13677–13722, Oct. 2023.
- [14] Z. Xi et al., "The rise and potential of large language model based agents: A survey," *Sci. China Inf. Sci.*, vol. 68, no. 2, Jan. 2025, Art. no. 121101.
- [15] S. Javaid, H. Fahim, B. He, and N. Saeed, "Large language models for UAVs: Current state and pathways to the future," *IEEE Open J. Veh. Technol.*, vol. 5, pp. 1166–1192, 2024.
- [16] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
- [17] R. Lowe et al., "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Red Hook, NY, USA, 2017, pp. 6382–6393.
- [18] C. Yu et al., "The surprising effectiveness of PPO in cooperative multi-agent games," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, 2022, pp. 24611–24624.
- [19] S. Green, C. M. Vineyard, and C. K. Koc, "Distillation strategies for proximal policy optimization," 2019, *arXiv:1901.08128*.
- [20] A. Hussein et al., "Imitation learning: A survey of learning methods," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–35, Apr. 2017.
- [21] T. Rashid et al., "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, Stockholm, Sverige, Jul. 2018, pp. 4295–4304.
- [22] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *Proc. 36th Int. Conf. Mach. Learn.*, Long Beach, CA, USA, Jun. 2019, pp. 5887–5896.
- [23] Q. Zou et al., "Cooperative multiagent attentional communication for large-scale task space," *Wireless Commun. Mobile Comput.*, vol. 2022, no. 1, Jan. 2022, Art. no. 4401653.
- [24] H. Mao et al., "Learning multi-agent communication with double attentional deep reinforcement learning," *Auton. Agents Multi-Agent Syst.*, vol. 34, no. 1, Apr. 2020, Art. no. 32.
- [25] Z. Lu et al., "Semantics-empowered communications: A tutorial-cum-survey," *IEEE Commun. Surveys Tuts.*, vol. 26, no. 1, pp. 41–79, Firstquarter 2024.
- [26] T. Ren et al., "Separate source channel coding is still what you need: An LLM-based rethinking," *ZTE Commun.*, vol. 23, no. 1, pp. 30–44, Mar. 2025.
- [27] M. M. Alam, S. Poudel, S. M. A. Huda, and S. Moh, "Leader-follower formation strategy in a UAV swarm for tree plantation: Design and effectiveness," in *Proc. 3rd Int. Conf. Comput. Technol.*, Bali, Indonesia, 2024, pp. 99–102.
- [28] S. Pateria et al., "Hierarchical reinforcement learning: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–35, Jan. 2021.
- [29] Y. Cheng et al., "Multi-UAV collaborative path planning using hierarchical reinforcement learning and simulated annealing," *Int. J. Performability Eng.*, vol. 18, no. 7, Jan. 2022, Art. no. 463.
- [30] B. Yu, H. Kasaei, and M. Cao, "Co-NavGPT: Multi-robot cooperative visual semantic navigation using large language models," 2025, *arXiv:2310.07937*.
- [31] Z. Mandi, S. Jain, and S. Song, "RoCo: Dialectic multi-robot collaboration with large language models," in *Proc. Int. Conf. Robot. Automat.*, 2024, pp. 286–299.
- [32] Y. Du et al., "Improving factuality and reasoning in language models through multiagent debate," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 11733–11763.
- [33] T. Liang et al., "Encouraging divergent thinking in large language models through multi-agent debate," in *Proc. Empirical Methods Natural Lang. Process.*, Miami, FL, USA, Nov. 2024, pp. 17889–17904.
- [34] C.-M. Chan et al., "ChatEval: Towards better LLM-based evaluators through multi-agent debate," in *Proc. Int. Conf. Learn. Representations*, Vienna, Austria, May 2024, pp. 9079–9093.
- [35] S. Hong et al., "MetaGPT: Meta programming for multi-agent collaborative framework," in *Proc. Int. Conf. Learn. Representations*, Vienna, Austria, May 2024, pp. 1609–1627.
- [36] G. Li et al., "CSMEL: Communicative agents for 'mind' exploration of large scale language model society," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, Dec. 2023, pp. 51991–52008.
- [37] C. Qian et al., "ChatDev: Communicative agents for software development," in *Proc. Assoc. Comput. Linguistics*, Bangkok, Thailand, Aug. 2024, pp. 15174–15186.
- [38] Y. Zhang et al., "Siren's song in the AI ocean: A survey on hallucination in large language models," 2023, *arXiv:2309.01219*.
- [39] Z. Zeng et al., "PersLLM: A personified training approach for large language models," 2024, *arXiv:2407.12393*.
- [40] Y.-S. Chuang et al., "Simulating opinion dynamics with networks of LLM-based agents," in *Proc. Assoc. Comput. Linguistics*, Mexico City, Mexico, Jun. 2024, pp. 3326–3346.

- [41] H. Chen et al., “Multi-agent consensus seeking via large language models,” 2023, *arXiv:2310.20151*.
- [42] E. Eigner and T. Händler, “Determinants of LLM-assisted decision-making,” 2024, *arXiv:2402.17385*.
- [43] C. Sun, S. Huang, and D. Pompili, “LLM-based multi-agent reinforcement learning: Current and future directions,” 2024, *arXiv:2405.11106*.
- [44] P. D. Siedler and I. M. Gemp, “LLM-Mediated guidance of MARL systems,” 2025, *arXiv:2503.13553*.
- [45] H. Zhang et al., “Building cooperative embodied agents modularly with large language models,” in *Proc. Proc. Int. Conf. Learn. Representations*, Vienna, Austria, pp. 42734–42763, May 2024.
- [46] T. R. Sumers et al., “Cognitive architectures for language agents,” *Trans. Mach. Learn. Res.*, vol. 2024, Sep. 2023.
- [47] Y. Xu, Z. Jian, J. Zha, and X. Chen, “Emergency networking using UAVs: A reinforcement learning approach with large language model,” in *Proc. 23rd ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, Hong Kong, 2024, pp. 281/282.
- [48] Z. Yuan et al., “YOLO-MARL: You only LLM once for multi-agent reinforcement learning,” 2024, *arXiv:2410.03997*.
- [49] R. Lowe et al., “Multi-agent actor-critic for mixed cooperative-competitive environments,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Mar. 2017, pp. 6382–6393.
- [50] K. D. Nguyen and T.-T. Nguyen, “Vision-based software-in-the-loop-simulation for unmanned aerial vehicles using Gazebo and PX4 open source,” in *Proc. Int. Conf. System Sci. Eng.*, Dong Hoi, Vietnam, 2019, pp. 429–432.
- [51] K. Lu et al., “Large language models are superpositions of all characters: Attaining arbitrary role-play via self-alignment,” in *Proc. Assoc. Comput. Linguistics*, Bangkok, Thailand, Aug. 2024, pp. 7828–7840.
- [52] J. Schulman et al., “Proximal policy optimization algorithms,” 2017, *arXiv:1707.06347*.
- [53] L. Meier, D. Honegger, and M. Pollefeys, “PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms,” in *Proc. IEEE Int. Conf. Robot. Automat.*, Seattle, WA, USA, 2015, pp. 6235–6240.
- [54] T. B. Brown et al., “Language models are few-shot learners,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Dec. 2020, pp. 1877–1901.
- [55] J. Wei et al., “Chain-of-thought prompting elicits reasoning in large language models,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, Dec. 2022, pp. 24824–24837.
- [56] M. Turpin et al., “Language models don’t always say what they think,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, New Orleans, LA, USA, Dec. 2023, pp. 74952–74965.
- [57] X. Wang, Y. Chen, and W. Zhu, “A survey on curriculum learning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4555–4576, Sep. 2022.
- [58] OpenAI et al., “GPT-4o system card,” 2024, *arXiv:2410.21276*.
- [59] Y. Wang et al., “Self-instruct: Aligning language models with self-generated instructions,” in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, Toronto, ON, Canada, Jul. 2023, pp. 13484–13508.
- [60] Y. Park et al., “Any-precision LLM: Low-cost deployment of multiple, different-sized LLMs,” in *Proc. 41st Int. Conf. Mach. Learn.*, Vienna, Austria, Jul. 2024, pp. 39682–39701.
- [61] Y. Zheng et al., “LlamaFactory: Unified efficient fine-tuning of 100+ language models,” in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, Bangkok, Thailand, Aug. 2024, pp. 400–410.
- [62] J. E. Hu et al., “LoRA: Low-rank adaptation of large language models,” in *Proc. Int. Conf. Learn. Representations*, Apr. 2022, pp. 12513–12525.
- [63] M. Walid, N. Slaheddine, A. Mohamed, and B. Lamjed, “Modelling, identification and control of a quadrotor UAV,” in *Proc. 15th Int. Multi-Conf. Syst., Signals Devices*, Hammamet, Tunisia, 2018, pp. 1017–1022.