OC-HMAS: Dynamic Self-Organization and Self-Correction in Heterogeneous Multiagent Systems Using Multimodal Large Models

Ping Feng, Tingting Yang, Mingyang Liang, Lin Wang, and Yuan Gao

***Abstract*—Heterogeneous multiagent systems (HMASs) leverage diverse agent capabilities to address complex tasks in dynamic environments, yet traditional approaches face limitations in autonomy and generalization when adapting to evolving scenarios. To overcome these challenges, we propose OC-HMAS, an IoT-integrated framework that synergizes self-organization and self-correction through multimodal perception. The system processes RGB images, LiDAR point clouds, and instance segmentation maps for real-time environmental awareness, while vision-language models and large language models (LLMs) jointly enable context-aware task decomposition, role allocation, and adaptive planning. Integrated path optimization and obstacle avoidance mechanisms further ensure operational safety and scalability across logistics, inspection, and search-and-rescue operations. Experimental validation demonstrates the framework's superiority over SMRC-LLM, with 5.15% higher success rates and 14.2% faster task completion in logistics, alongside 4.69% accuracy gains and 12.9% time reduction in inspection scenarios. These results validate its enhanced adaptability in IoT-augmented environments, establishing a new benchmark for autonomous HMAS deployment.**

***Index Terms*—Embodied intelligence, environmental perception, Internet of Things (IoT), large language model (LLM), multiagent system (MAS), task scheduling, vision language model (VLM).**

Received 4 November 2024; revised 8 January 2025 and 1 February 2025; accepted 9 February 2025. Date of publication 3 March 2025; date of current version 9 May 2025. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1806800; in part by the Key Projects of the National Defense Foundation Strengthening Plan under Grant 2020-JCJQ-ZD-020-05; in part by the Peng Cheng Laboratory Project under Grant PCL2021A02; in part by the Guangdong Province Basic and Applied Basic Research Foundation under Grant 2019B1515120084, Grant 2022A1515110787, and Grant 2024A1515012065; in part by the Key-Area Research and Development Program for Guangdong Province under Grant 2019B010136001; in part by the Shenzhen Science and Technology Program under Grant JSGGKQTD20221101115656029; and in part by the Longgang District Shenzhen's Ten Action Plan for Supporting Innovation Projects under Grant LGKCSDPT2024002. *(Corresponding authors: Yuan Gao; Tingting Yang.)*

Ping Feng and Tingting Yang are with the School of Navigation, Dalian Maritime University, Dalian 116026, China, and also with the Department of Human Resources and Education, Pengcheng Laboratory, Shenzhen 518066, China (e-mail: yangtt@pcl.ac.cn).

Mingyang Liang is with the Sino-German Intelligent Manufacturing College, Shenzhen Technology University, Shenzhen 518122, China.

Lin Wang is with the Intelligent Robotics Center, Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518172, China.

Yuan Gao is with the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen 518172, China, and also with The Chinese University of Hong Kong, Shenzhen 518172, China (e-mail: gaoyuan@cuhk.edu.cn).

Digital Object Identifier 10.1109/JIOT.2025.3545496

I. Introduction

HETEROGENEOUS multiagent systems (HMAS) have become increasingly integral for executing complex tasks in dynamic and uncertain environments [1]. By leveraging the diverse capabilities of various agents, HMAS can effectively address significant challenges related to environmental uncertainty, task complexity, and the need for flexible coordination [2]. Furthermore, embodied intelligence integrates physical agents with cognitive processes, enabling meaningful interactions and adaptations within their environments. The evolution of HMAS began with early decentralized control and cooperative behavior models [3]. It progressed through advanced frameworks like decentralized partially observable Markov decision processes (Dec-POMDPs) [4] and now integrates machine learning techniques for greater autonomy and adaptability [5]. Despite their potential, existing heterogeneous HMAS often encounter substantial adaptability and real-time responsiveness limitations across multiple scenarios [1]. Traditional systems are typically constrained by predefined roles and scenario-specific parameters, which restrict their flexibility and scalability [6]. These constraints make it challenging for HMAS to adjust efficiently to real-time environmental changes, hindering effective task execution under varying conditions.

Moreover, conventional HMAS often depend on human intervention for error correction, leading to delayed responses and diminished robustness in dynamic settings [7]. This reliance highlights the critical need for more autonomous systems capable of real-time adaptation and self-organization. Recent advancements have explored the integration of machine learning techniques, particularly reinforcement learning and deep learning, to enhance the intelligence and autonomy of HMAS [8], [9]. However, achieving seamless adaptability and robust performance in diverse and unpredictable environments remains a significant challenge.

Motivated by the expanding convergence of HMAS with the IoT [10], this article introduces a Self-Organizing and Self-Correcting HMAS (OC-HMAS). OC-HMAS integrates multimodal sensing with VLM and LLM to address these limitations. The proposed framework is optimized for a variety of applications, including logistics, inspection, and search-and-rescue operations. By employing multimodal sensors, such as RGB cameras, InstanceSeg, and LiDAR, the system achieves precise real-time environmental perception. The integration of VLM and LLM facilitates intelligent

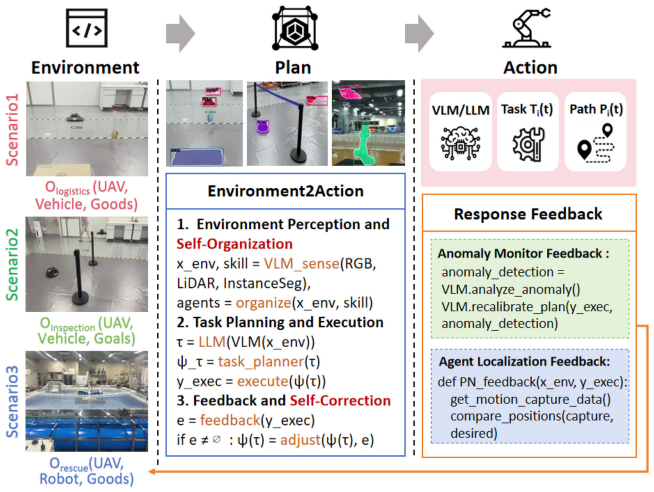


Fig. 1. OC-HMAS framework leverages multimodal data (RGB, LiDAR, InstanceSeg, et al.) and advanced models (VLMs and LLMs) to enable dynamic self-organization and self-correction in HMASs. Its stages facilitate real-time environmental perception, self-organization (including role assignment, adaptive task allocation, and path planning), and self-correction feedback (providing continuous feedback for anomaly monitoring and localization). Applied across logistics, inspection, and rescue scenarios, OC-HMAS enhances efficiency, safety, and scalability, outperforming traditional frameworks in broadly generalizable IoT-enabled environments.

task decomposition and dynamic planning, enabling seamless adaptation to varying conditions. Additionally, OC-HMAS's dynamic perception allows agents to continuously interpret and respond to environmental changes, ensuring sustained operational effectiveness. The system also incorporates self-correction and adaptive collaboration mechanisms to enhance resilience and flexibility. Fig.1 illustrates the overall architecture of the proposed OC-HMAS framework, which comprises three main stages: 1) real-time environmental perception; 2) self-organization; and 3) self-correction feedback. In the perception stage, multimodal data integration provides comprehensive situational awareness. The self-organization stage leverages VLMs and LLMs for dynamic role assignment, adaptive task allocation, and path planning, enabling agents to autonomously organize themselves based on real-time data. The self-correction feedback stage ensures continuous monitoring and localization of anomalies, allowing the system to adjust and optimize its operations autonomously. This comprehensive framework enables OC-HMAS to perform efficiently across various applications by enhancing flexibility, scalability, and robustness in dynamic IoT-enabled environments.

A central planner within the system processes user instructions and dynamically allocates tasks and roles based on real-time scenario demands. This adaptive role assignment enhances the system's flexibility and scalability, allowing agents to switch roles to meet evolving task requirements. Furthermore, the framework implements a two-tiered self-correction mechanism to ensure robust performance: the central brain provides real-time feedback by processing multimodal data, while individual agents conduct local self-assessments by integrating sensor inputs with central brain reasoning. This hierarchical feedback loop facilitates continuous task optimization and autonomous error correction, significantly improving the system's ability to handle complex and unpredictable environments. The contributions of this article are as follows.

1) We propose a self-organizing framework with high autonomy and multitask adaptability, surpassing traditional fixed-task MAS architectures. The central planner translates user instructions into task and time assignments, allowing agents to adapt roles across diverse scenarios, enhancing flexibility and scalability.
2) A two-layer self-correction mechanism mitigates latency and reduces human intervention. The system adjusts global feedback based on multimodal inputs, while agents perform local self-evaluation by integrating sensor data with the central planner. This cloud-edge feedback loop boosts system resilience and robustness.
3) Our system enables precise, real-time environmental perception through the fusion of multimodal data (RGB, InstanceSeg, and LiDAR). The integration of VLM and LLM facilitates adaptive task decomposition and planning, enabling dynamic optimization of task allocation and path planning. This combination enhances the system's ability to operate safely and efficiently in complex environments.

II. Related Works

A. Multiagent Deep Reinforcement Learning Algorithms

Recent advancements in multiagent deep reinforcement learning (MARL) have greatly improved agent coordination and scalability in dynamic environments [12]. Algorithms like multiagent proximal policy optimization (MAPPO) adapt proximal policy optimization (PPO) for multiagent settings, promoting stable policy updates while balancing exploration and exploitation [13]. Enhancements, such as heterogeneous agent trust region policy optimization (HATRPO) account for the diverse capabilities of agents, making these methods effective for complex, real-world tasks [14].

Decentralized MARL approaches have also tackled communication and scalability challenges. FacMAC, introduced by Peng et al. [15], employs a factorized multiagent communication mechanism to minimize interagent communication overhead while maintaining effective collaboration. FacMAC achieves this by focusing on shared latent representations, which are updated only when essential, thereby optimizing coordination in static settings. However, this reliance on predefined communication structures reduces its adaptability to dynamic and time-sensitive environments. For instance, rescue missions requiring rapid role changes and real-time decision-making challenge FacMAC's rigid frameworks.

Our experiments reveal that while FacMAC excels in reducing communication latency and optimizing coordination in controlled, static environments, it struggles with dynamic, time-critical scenarios. This limitation underscores the need for frameworks like OC-HMAS, which leverage multimodal data and dynamic task allocation to handle real-time adaptability and scalability. Additionally, compared to IC3Net, which relies on LSTM-based memory mechanisms that may not capture long-term dependencies effectively, OC-HMAS utilizes

advanced perception models to maintain situational awareness, thereby enhancing overall coordination and performance in dynamic tasks.

B. Large Models for Multiagent Systems

The integration of large models, including VLMs and LLMs, has significantly advanced multiagent systems (MASs) [17]. VLMs, such as CLIP and multimodal GPT-4V, proficiently process visual and textual data, facilitating applications like object detection and robotic planning through multimodal inputs. By merging vision and language, these models enhance agents' environmental reasoning capabilities, proving indispensable in tasks, such as spatial reasoning and combinatorial optimization [18].

SMRC-LLM [33], focuses on improving interagent communication through large-scale language models capable of interpreting and generating natural language instructions. This mechanism enables seamless coordination in collaborative tasks by reducing ambiguities inherent in traditional communication methods. While effective in structured tasks, such as warehouse logistics and assembly operations, SMRC-LLM's high computational demands and dependence on centralized processing limit its applicability in scenarios requiring rapid real-time decision-making, such as inspection or search-and-rescue operations.

In contrast, OC-HMAS addresses these limitations by implementing a hybrid architecture, distributing computational loads between centralized nodes and edge agents. This design leverages LLM capabilities for high-level reasoning while integrating low-layer, multimodal awareness at the edge to enhance responsiveness and reduce computational latency. As a result, OC-HMAS achieves superior adaptability and robustness in dynamic multiagent scenarios. Furthermore, unlike IC3Net and FACMAC, which may require extensive retraining to incorporate new tasks or environments, OC-HMAS's modular architecture allows for easier integration of additional models or data sources, enhancing its scalability and flexibility in diverse operational contexts.

C. Multimodal Data Fusion Processing

Recent developments in multimodal agent collaboration systems have enabled agents to integrate data from diverse sensors, such as RGB cameras, LiDAR, and thermal imaging [21]. These systems enhance object detection, depth perception, and situational awareness in real-time, particularly under challenging conditions like low-light environments [22]. By employing early fusion techniques, multimodal agents improve the accuracy of 2-D and 3-D object detection, thereby increasing their reliability for inspection, surveillance, and disaster management tasks.

Joint representation learning techniques have further augmented the robustness and adaptability of multimodal systems [23], [24]. By aligning and processing multiple data streams simultaneously, agents can make informed decisions and coordinate more effectively across various tasks, including patrolling and search-and-rescue operations. Additionally, real-time data processing ensures that agents can dynamically respond to environmental changes, facilitating seamless coordination in time-sensitive missions [25].

While models like IC3Net effectively manage large-scale temporal memory challenges through recurrent network structures, their scalability is limited in highly dynamic environments. Conversely, OC-HMAS integrates the advanced capabilities of large multimodal models, surpassing traditional methods in adaptability, robustness, and real-time task management. This alignment of multimodal processing with dynamic task allocation ensures a more resilient and scalable framework for HMASs. Additionally, compared to FACMAC's reinforcement learning-based communication, OC-HMAS's multimodal fusion allows for richer and more context-aware interactions among agents, enhancing overall mission performance and reliability.

III. System Modeling

The mathematical nomenclature and definitions utilized in the system modeling are comprehensively summarized in Table I, offering a clear reference for interpreting the variables, constraints, and optimization criteria discussed herein. This study centers on a heterogeneous multiagent framework comprising autonomous aerial vehicles (AAVs), autonomous vehicles (AVs), and robotic arms, with the primary objective of optimizing task execution and enhancing collaboration across diverse operational scenarios. Each agent's state at a given time t is denoted as $U_{n_t}^t$ for AAVs, $V_{n_q}^t$ for AVs, and $R_{n_w}^t$ for robotic arms, where n_t, n_q, and n_w represent the respective indices of the agents. The optimization problem is formulated to minimize the cost function $f_{\pi_\theta}(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}})$, which integrates task completion time, path efficiency, and resource utilization. This cost function is subject to several constraints aimed at ensuring both safety and efficiency within the system

$$\arg\min_{\theta} \quad f_{\pi_\theta}\left(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}}\right). \tag{1}$$

This equation represents the core objective of the optimization problem, where the goal is to determine the parameter set θ that minimizes the cost function f_{π_θ}. The function f_{π_θ} encapsulates the key performance metrics of task completion time, path efficiency, and resource utilization across the heterogeneous agents $\widetilde{U}$ AAVs, $\widetilde{V}$ AVs, and $\widetilde{\mathcal{R}}$ (robotic arms). The parameters π_θ govern the policies of the agents, dictating their actions and interactions to achieve optimal system performance

$$\mathcal{S}(t) = \{s_i(t)\}_{i=1}^{N}, \quad \mathcal{T}(t) = \{T_i(t)\}_{i=1}^{N} \tag{2}$$

here, $\mathcal{S}(t)$ denotes the state space at time t, comprising the states $s_i(t)$ of all N agents within the system. Concurrently, $\mathcal{T}(t)$ represents the task set at time t, consisting of tasks $T_i(t)$ assigned to each agent. These definitions are fundamental for modeling the dynamic interactions and task allocations that occur within the multiagent framework

$$J(\mathcal{T}, \mathcal{P}) = \sum_{i=1}^{N_U+N_V+N_R} \left(\alpha_1 \cdot T_{\text{completion}}(T_i) + \alpha_2 \cdot E_i(\mathcal{P}_i) + \alpha_3 \cdot \text{Safety}(P_i)\right) \tag{3}$$

TABLE I
MATHEMATICAL NOMENCLATURE AND DEFINITIONS

Symbol	Definition / Meaning	Symbol	Definition / Meaning
θ	Parameters governing agents' policies	f_{π_θ}	System cost function for performance metrics
g_{π_θ}	Constraint function for safety and resources	$g^*_{\pi\theta}$	Path planning constraint for agent n_p, ensuring safety
$k_{\pi\theta}$	Coordination limits between UAVs and AVs	c_{π_θ}	System state constraint for task execution and stability
u, v, R	States of UAVs, AVs, and robotic arms	$P_i(t)$	Path of agent i at time t
d_{safe}	Minimum safe distance from obstacles	$\mathbf{p}_i(t)$	Position vector of agent i at time t
$\mathbf{q}_i(t)$	Quaternion representing agent i's orientation	$\mathbf{P}_i^{\text{desired}}(t)$	Desired path of agent i at time t
$\Delta\mathbf{p}_i(t)$	Position error for agent i	$\Delta\mathbf{q}_i(t)$	Orientation error for agent i
$\alpha_1, \alpha_2, \alpha_3$	Weights for task time, energy, and safety	$\mathcal{S}(t)$	State space of the entire system at time t
$\mathcal{T}(t)$	Task set assigned to agents at time t	$J(T, P)$	Cost function for task and path planning
J_i	Individual cost function for agent i	J_{path}	Path-specific cost function
J_{total}	Total cost function across all agents	$V(t)$	Lyapunov function ensuring system stability
$\frac{dV(t)}{dt}$	Time derivative of Lyapunov function	$\epsilon_{\max}$	Maximum allowable trajectory error
$\mathbf{u}_i(t)$	Control input for agent i at time t	$\mathbf{I}^{\text{RGB}}(t)$	RGB image input at time t
$\mathbf{P}^{\text{LiDAR}}(t)$	LiDAR point cloud data at time t	InstSegNet	InstanceSeg neural network
$S(t)$	Scene description generated from visual data	$\tilde{S}(t)$	Fused scene description with multi-modal data
LLM	LLM for decision-making	VLM	VLM for visual data processing
VLM_{RGB}	VLM for RGB data	$\text{VLM}_{\text{InstSeg}}$	VLM for InstanceSeg data
ϕ_{LiDAR}	LiDAR feature extraction function	Feature2Text	Function to convert features to text
Γ	Classification function for anomaly types	Θ	Function for adjusting tasks based on anomaly
γ^{role}	Assignment function for roles	$\mathcal{A}$	Set of anomaly categories
$\mathcal{R}$	Set of roles	$A(t)$	Type of anomaly detected at time t
$\mathbf{T}_i(t)$	Task set for agent i at time t	$\mathbf{T}_i^{\text{new}}(t)$	Updated task set for agent i after anomaly
$r_i(t)$	Role of agent i at time t	$\epsilon_{\text{pos}}, \epsilon_{\text{att}}$	Threshold values for position and attitude errors
$\mathbf{S}_j(t)$	State feedback of agent j at time t	$\mathbf{P}_i^{\text{actual}}(t)$	Actual path of agent i at time t
$\mathbf{P}_i^{\text{desired}}(t)$	Desired path of agent i at time t	$\mathbf{E}_i(t)$	Trajectory error of agent i at time t

the cost function $J(\mathcal{T}, \mathcal{P})$ aggregates the performance metrics across all agents, where N_U, N_V, and N_R denote the number of AAVs, AVs, and robotic arms, respectively. The term $T_{\text{completion}}(T_i)$ represents the time required to complete task T_i, $E_i(\mathcal{P}_i)$ denotes the energy consumed by agent i along its path $\mathcal{P}_i$, and Safety(P_i) measures the safety of the path P_i, considering obstacles and environmental constraints. The coefficients α_1, α_2, and α_3 are weights that determine the relative importance of each metric in the overall cost function. By adjusting these weights, the optimization process can prioritize different aspects of performance based on the specific requirements of the operational scenario.

The optimization problem can thus be formally expressed as (1), subject to the constraints

$$\begin{aligned} g_{\pi_\theta}\left(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}}\right) &\le \mathcal{C}_1 \\ g^*_{\pi_\theta}\left(\mathcal{P}_{n_p}, \widetilde{\mathcal{R}}_{n_p}\right) &\le \mathcal{C}_2 \quad \forall \mathcal{P}_{n_p}, \widetilde{\mathcal{R}}_{n_p} \in \left(\widetilde{U} \cup \widetilde{V}\right) \\ k_{\pi_\theta}(\widetilde{v}, \widetilde{u}) &\le \widetilde{\mathcal{C}}_3 \\ c_{\pi_\theta}\left(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}}\right) &= 0. \end{aligned} \tag{4}$$

In this formulation, the constraints ensure that the system operates within predefined safety and resource limits. The first constraint $g_{\pi_\theta}(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}}) \le \mathcal{C}_1$ enforces resource limitations, such as battery capacity and payload for each agent. The second constraint $g^*_{\pi_\theta}(\mathcal{P}_{n_p}, \widetilde{\mathcal{R}}_{n_p}) \le \mathcal{C}_2$ mandates collision avoidance by maintaining a minimum safety distance during path planning. The coordination constraint $k_{\pi_\theta}(\widetilde{v}, \widetilde{u}) \le \widetilde{\mathcal{C}}_3$ ensures synchronization between AAVs and AVs during critical operations like package handovers. Finally, $c_{\pi_\theta}(\widetilde{U}, \widetilde{V}, \widetilde{\mathcal{R}}) = 0$ maintains operational consistency across all agents, thereby ensuring system stability.

IV. METHOD

A. General Framework

The system goes through five key stages in the proposed heterogeneous multiagent framework, as shown in Fig. 2, and each stage is crucial for optimizing task execution in a dynamic environment. The first stage, *Environmental Perception*, involves acquiring multimodal sensor data, such as RGB images, InstanceSeg, and LiDAR. These sensor inputs provide foundational data for thorough scene understanding, including object identification and obstacle detection. By leveraging these inputs, agents can perceive their surroundings in real time, allowing them to adapt effectively to dynamic changes, such as emerging obstacles along the robot's path.

Next, *Role Assignment* capitalizes on multimodal perception and recognition to evaluate robotic capabilities and create an inventory of environmental objects and their properties. Following this, roles are assigned to agents through scene definition and role assignment, facilitating general task decomposition. With their roles clearly defined, each agent can take on specific tasks and responsibilities. Complex processes, such as logistics operations, are broken down into smaller, more manageable subtasks, such as selecting optimal routes, inspecting operational conditions, ensuring obstacle avoidance, and completing deliveries.

In the third stage, *Task Decomposition*, multiagent parallel execution is achieved, essential for effective task allocation among agents. For instance, in a logistics scenario, subtasks might include determining the optimal transport route, assigning inspection duties for route safety, and coordinating search and rescue efforts if items are lost. At this point, task parallelization is critical for boosting efficiency. Path

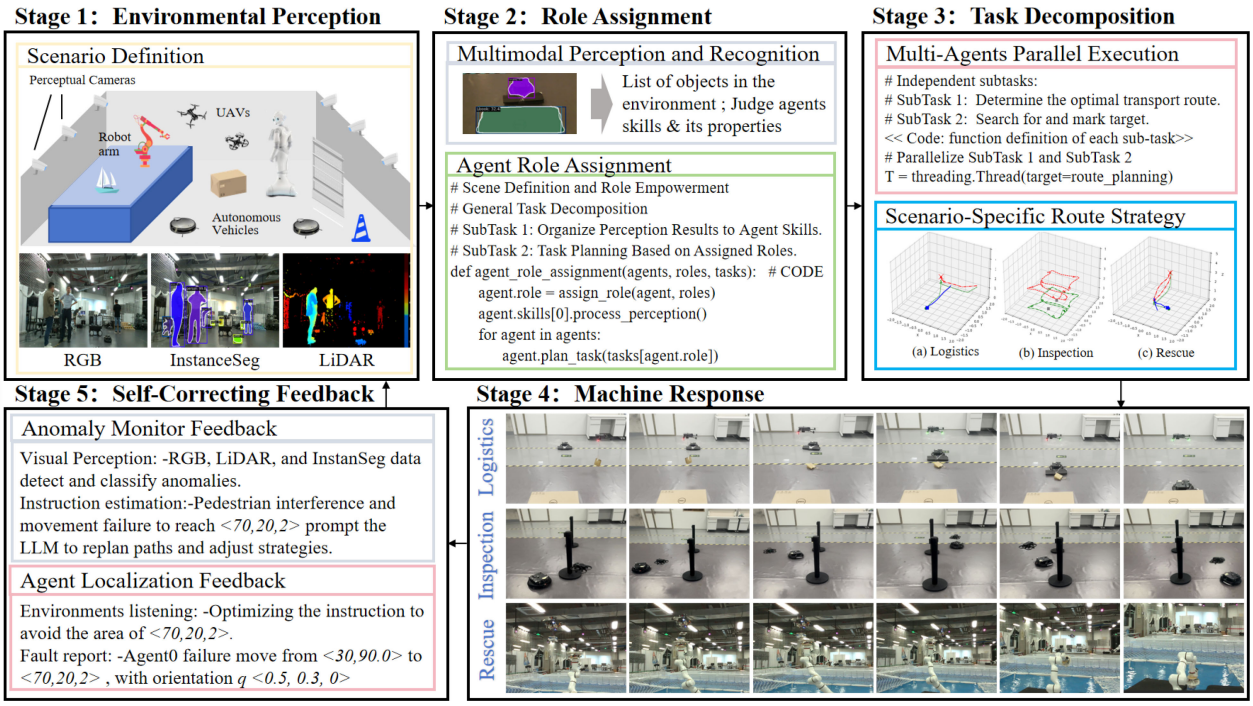


Fig. 2. Framework of the OC-HMAS illustrates five essential stages for efficient and adaptive task execution: Environmental Perception, Role Assignment, Task Decomposition, Machine Response, and Self-Correction Feedback. The system achieves self-organization and Self-Correction through multimodule collaboration, with each stage contributing to its capacity for autonomous operation in complex and dynamic environments.

planning leverages 3-D coordinates with a scenario-specific routing strategy to accommodate spatial and temporal constraints. Moreover, obstacle avoidance and route optimization algorithms, such as PID controllers [26] and nonlinear model predictive control (NMPC) [27], are applied to guarantee safe and efficient transportation.

The fourth stage, *Machine Response*, highlights the system's capability to perform real-time tasks across various operational environments, such as logistics, inspection, and search-and-rescue scenarios. During logistics operations, the system closely monitors transport status, reports task progress (e.g., vehicle in transit), and provides completion rates (e.g., 50% completion when an item is mid-transport). Meanwhile, during inspections, agents assess the condition of equipment along transport routes, delivering real-time updates on their progress. In search-and-rescue scenarios, agents focus on locating and retrieving target items, with real-time monitoring ensuring tasks are executed efficiently.

Finally, *Self-Correction Feedback* forms the foundation for the system's adaptive capabilities. As environmental conditions evolve, the system detects these changes—such as shifts in obstacle locations or variations in speed—and adjusts its responses accordingly. Feedback from the environment informs the system of these modifications, which might lead to task execution issues, such as an inability to move between coordinates due to unexpected obstacles. When this occurs, a self-correcting mechanism is activated, allowing the system to recalculate paths and modify task planning to maintain efficiency dynamically. This continuous feedback loop supports the ongoing optimization of task execution, enhancing the system's robustness and autonomy in ever-changing environments.

The following sections provide a detailed explanation of the two key mechanisms: 1) self-organization, which includes environmental perception, role assignment, and task decomposition; 2) self-correction, which consists of machine response, anomaly monitoring feedback and dynamic agent localization feedback.

B. Self-Organization Mechanism

1) Environment Perception and Multimodal Data Fusion: In HMASs, scene adaptation and environmental anomaly detection are vital for maintaining efficient and self-organized system behavior. The system achieves this by analyzing real-time environmental data, primarily through a VLM and a LLM, ensuring agents can dynamically adjust tasks in response to environmental changes.

Environment perception begins with the acquisition of visual and LiDAR data. For scene analysis, we collect RGB images $\mathbf{I}^{\mathrm{RGB}}(t) \in \mathbb{R}^{H\times W\times 3}$ and LiDAR point clouds $\mathbf{P}^{\mathrm{LiDAR}}(t) \in \mathbb{R}^{N_p\times 3}$ at time t, where H and W represent the dimensions of the RGB image, and N_p denotes the number of points in the LiDAR data. The RGB image provides rich visual context, while the LiDAR data contributes spatial information for depth and location awareness.

To extract visual information, we utilize advanced InstanceSeg techniques, segment anything model (SAM) [29] and Faster R-CNN [30], generating InstanceSeg results

$$\mathbf{S}^{\mathrm{InstSeg}}(t) = \mathrm{InstSegNet}\left(\mathbf{I}^{\mathrm{RGB}}(t)\right). \tag{5}$$

This yields segmentation masks and object boundaries, which facilitate understanding object-level details in the environment.

Concurrently, the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$ is processed by a VLM (GPT-4o [31]) to generate a high-level scene description

$$S^{\mathrm{RGB}}(t) = \mathrm{VLM}_{\mathrm{RGB}}\left(\mathbf{I}^{\mathrm{RGB}}(t)\right). \tag{6}$$

The scene analysis results $S^{\mathrm{RGB}}(t)$ represent the outcome of real-world scene interpretation, essential for executing tasks. In parallel, the InstanceSeg result $\mathbf{S}^{\mathrm{InstSeg}}(t)$ is transformed into a semantic description via a separate VLM instance

$$S^{\mathrm{InstSeg}}(t) = \mathrm{VLM}_{\mathrm{InstSeg}}\left(\mathbf{S}^{\mathrm{InstSeg}}(t)\right). \tag{7}$$

These two descriptions—one derived from raw RGB data and the other from segmentation results—are then fused using a concatenation function

$$S(t) = \mathrm{Concat}\left(S^{\mathrm{RGB}}(t), S^{\mathrm{InstSeg}}(t)\right). \tag{8}$$

This process ensures that visual data from different sources (raw images and segmented objects) are aligned and integrated into a comprehensive scene description.

For the LiDAR data $\mathbf{P}^{\mathrm{LiDAR}}(t)$, we extract spatial features using point cloud feature extraction networks (e.g., PointNet or Dynamic Graph CNN)

$$\mathbf{f}^{\mathrm{LiDAR}}(t) = \phi_{\mathrm{LiDAR}}\left(\mathbf{P}^{\mathrm{LiDAR}}(t)\right). \tag{9}$$

The extracted LiDAR features are either retained as spatial information or converted into textual descriptions $S^{\mathrm{LiDAR}}(t)$ using a feature-to-text model

$$S^{\mathrm{LiDAR}}(t) = \mathrm{Feature2Text}\left(\mathbf{f}^{\mathrm{LiDAR}}(t)\right). \tag{10}$$

Finally, the comprehensive scene description and LiDAR data are fused to form the final input for task planning

$$\tilde{S}(t) = \mathrm{Concat}\left(S(t), S^{\mathrm{LiDAR}}(t)\right). \tag{11}$$

The LLM (GPT-4 [32]) utilizes this fused environmental representation to generate task plans for each agent in the system, taking into account the environmental state and dynamically assigning tasks

$$\mathbf{T}_i(t) = \mathrm{LLM}\left(\tilde{S}(t), \{\mathbf{S}_j(t)\}_{j=1}^{N}\right), \quad i = 1, 2, \ldots, N \tag{12}$$

where $\mathbf{S}_j(t)$ represents the state feedback from agent j.

2) Role Assignment and Task Decomposition: We utilize the environmental perception result $\tilde{S}(t)$ to assign roles to the agents. At time t, roles are assigned based on the list of objects collected from the environment and the skills and attributes of the agents. Let $\mathbf{A}_i$ denote the attributes of agent i, $\mathbf{O}(t)$ represent the list of objects in the environment, and each object be denoted as $o_k(t) \in \mathbf{O}(t)$. The process of role assignment can be expressed as

$$r_i(t) = \gamma^{\mathrm{role}}\left(\tilde{S}(t), \mathbf{A}_i, \mathbf{O}(t)\right), \quad r_i(t) \in \mathcal{R} \tag{13}$$

where, $\gamma^{\mathrm{role}}(\cdot)$ represents the role assignment function, while $\mathcal{R}$ denotes the set of roles. Based on the agent's properties and the characteristics of the objects in the environment, the system assigns an appropriate role to each agent.

The task set $\mathbf{T}(t)$ can be expressed as a decomposition problem in task decomposition. Assuming the task goal is $\mathbf{G}(t)$, we decompose the overall task into several subtasks by analyzing the environment state $\tilde{S}(t)$

$$\mathbf{T}(t) = \Delta\left(\tilde{S}(t), \mathbf{G}(t)\right) \tag{14}$$

where $\Delta(\cdot)$ represents the task decomposition function. Each agent i is assigned to a subtask $\mathbf{T}_i(t)$, which can be determined by the agent's skills and the state of the environment

$$\mathbf{T}_i(t) = \delta(\mathbf{A}_i, \mathbf{T}(t)), \quad i = 1, 2, \ldots, N. \tag{15}$$

After assigning roles to agents, the LLM dynamically adjusts the tasks of the agents based on the anomaly type $A(t)$ and the current task planning $\mathbf{T}_i(t)$

$$\mathbf{T}_i^{\mathrm{new}}(t) = \Theta(\mathbf{T}_i(t), A(t)). \tag{16}$$

Simultaneously, the system dynamically assigns roles to agents based on the new task planning. Define the set of roles $\mathcal{R}$, the role of agent i at time t is determined by the assignment function γ^{role}

$$r_i(t) = \gamma^{\mathrm{role}}\left(\mathbf{T}_i^{\mathrm{new}}(t)\right), \quad r_i(t) \in \mathcal{R}. \tag{17}$$

After completing role assignment and task planning, the LLM generates a new path plan for the agent. The path planning problem is formalized as an optimization problem aimed at finding the optimal path $\mathbf{P}_i(t)$ to minimize the cost function J_i

$$\mathbf{P}_i(t) = \arg\min_{\mathbf{P}} J_i\left(\mathbf{P}, \mathbf{T}_i^{\mathrm{new}}(t), \tilde{S}(t)\right). \tag{18}$$

The cost function J_i may include factors, such as path length, time, energy consumption, and safety. The overall system performance metric is defined as

$$J_{\mathrm{total}} = \sum_{i=1}^{N} J_i\left(\mathbf{P}_i(t), \mathbf{T}_i^{\mathrm{new}}(t), \tilde{S}(t)\right). \tag{19}$$

The optimization goal is to minimize the total cost function

$$\{\mathbf{P}_i^*(t)\}_{i=1}^{N} = \arg\min_{\{\mathbf{P}_i(t)\}} J_{\mathrm{total}}. \tag{20}$$

To ensure system stability, we define a Lyapunov function

$$V(t) = \frac{1}{2}\sum_{i=1}^{N} |\mathbf{E}_i(t)|^2, \tag{21}$$

and its time derivative is given by

$$\frac{dV(t)}{dt} = \sum_{i=1}^{N} \mathbf{E}_i^{\top}(t)\dot{\mathbf{E}}_i(t). \tag{22}$$

By designing an appropriate control strategy, we ensure that

$$\frac{dV(t)}{dt} \leq 0. \tag{23}$$

This expression indicates that the derivative of $V(t)$ with respect to time is nonpositive, thereby guaranteeing the stability of the system at the equilibrium point.

In summary, the central scheduling system utilizes VLM and LLM to acquire and process visual information. The processed LiDAR point cloud features $\mathbf{f}^{\mathrm{LiDAR}}(t)$ are fused with the scene description $S(t)$ and input into the LLM for

decision-making. In this manner, the system achieves a deep understanding of the environment and dynamic task planning. Combined with the self-correction mechanism, the system can adaptively adjust the behavior of agents in dynamic environments, ensuring efficient task completion and system stability.

C. Self-Correction Mechanism

1) Route Strategy and Machine Response: To enhance the synergy between peripheral network feedback and the VLM and LLM in respective visual data acquisition and task planning, we integrate peripheral network data with visual information and dynamic task adaptation, forming a closed-loop feedback system. This system merges the agents' physical states with environmental perception, ensuring the accuracy and responsiveness of task planning.

At any time t, the position of each agent i, denoted by $\mathbf{p}_i(t)$, is defined as

$$\mathbf{p}_i(t) = \begin{bmatrix} x_i(t) & y_i(t) & z_i(t) \end{bmatrix} \in \mathbb{R}^3 \tag{24}$$

where $x_i(t)$, $y_i(t)$, and $z_i(t)$ represent the spatial coordinates of agent i at time t.

The orientation of the agent is represented by a quaternion $\mathbf{q}_i(t)$, defined as

$$\mathbf{q}_i(t) = \begin{bmatrix} q_{w_i}(t) & q_{x_i}(t) & q_{y_i}(t) & q_{z_i}(t) \end{bmatrix} \in \mathbb{H} \tag{25}$$

where $\mathbb{H}$ denotes the set of quaternions, with $q_{w_i}(t)$ as the real part, and $q_{x_i}(t)$, $q_{y_i}(t)$, and $q_{z_i}(t)$ as the imaginary parts.

The system uses a motion capture perceptual networks listener to acquire the spatial coordinates and orientation of each agent in real-time. This data is used in the task planning process, providing real-time physical state feedback to ensure that the agents' motion and orientation conform to the desired goals.

The desired trajectory $\mathbf{p}_i^{\text{desired}}(t)$ and desired orientation $\mathbf{q}_i^{\text{desired}}(t)$ of agent i are defined as

$$\mathbf{p}_i^{\text{desired}}(t) = \begin{bmatrix} x_i^{\text{desired}}(t) \\ y_i^{\text{desired}}(t) \\ z_i^{\text{desired}}(t) \end{bmatrix} \in \mathbb{R}^3 \tag{26}$$

$$\mathbf{q}_i^{\text{desired}}(t) = \begin{bmatrix} q_{w_i}^{\text{desired}}(t) \\ q_{x_i}^{\text{desired}}(t) \\ q_{y_i}^{\text{desired}}(t) \\ q_{z_i}^{\text{desired}}(t) \end{bmatrix} \in \mathbb{H}. \tag{27}$$

The trajectory error $\Delta\mathbf{p}_i(t)$, defined as the difference between the desired and actual positions, is given by

$$\Delta\mathbf{p}_i(t) = \mathbf{p}_i^{\text{desired}}(t) - \mathbf{p}_i(t) = \begin{bmatrix} x_i^{\text{desired}}(t) - x_i(t) \\ y_i^{\text{desired}}(t) - y_i(t) \\ z_i^{\text{desired}}(t) - z_i(t) \end{bmatrix}. \tag{28}$$

The attitude error $\Delta\mathbf{q}_i(t)$ is defined as the difference between the desired and current quaternion. The error is computed using quaternion multiplication

$$\Delta\mathbf{q}_i(t) = \mathbf{q}_i^{\text{desired}}(t) \otimes \mathbf{q}_i^{-1}(t) \tag{29}$$

where $\otimes$ denotes quaternion multiplication and $\mathbf{q}_i^{-1}(t)$ is the inverse of the current quaternion. This computation captures the rotational difference between the agent's current and desired orientations.

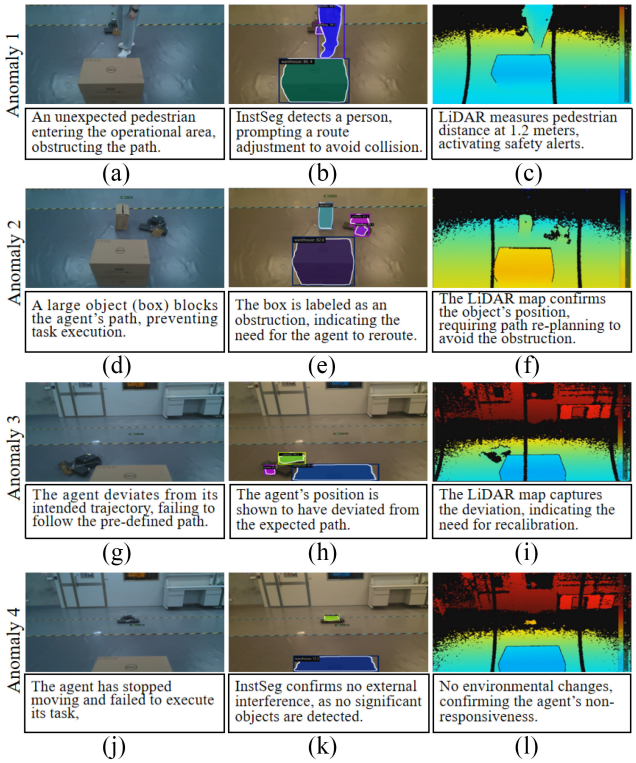


Fig. 3. Four common types of anomalies: pedestrian appearance ($A = 1$), object occlusion ($A = 2$), trajectory deviation ($A = 3$), and device non-response ($A = 4$). Each anomaly is depicted across three types of data: RGB image $\mathbf{I}^{\text{RGB}}(t)$ (first column), instance segmentation $\mathbf{S}^{\text{InstSeg}}(t)$ (second column), and LiDAR feature map $\mathbf{f}^{\text{LiDAR}}(t)$ (third column). The data illustrate the nature of each anomaly and how it is detected through multiple sensor modalities.

The peripheral network feedback data is combined with VLM and LLM-driven task planning, providing real-time physical state feedback to ensure the agents' motions and orientations continuously adjust to meet the desired goals.

2) Anomaly Monitoring and Location Feedback: In order to realize environmental anomaly detection, it is assumed that under normal circumstances, the fusion scene description $\tilde{S}(t)$ obeys the probability distribution $p_{\text{normal}}(\tilde{S})$. By collecting data under normal circumstances, a reference model is established. For the current moment, the logarithmic probability density is calculated as follows:

$$\log p(t) = \log p_{\text{normal}}\left(\tilde{S}(t)\right). \tag{30}$$

Set the anomaly detection threshold δ. If $\log p(t) < \delta$, an environmental anomaly is detected. In order to identify the specific types of anomalies, a set of anomaly categories $\mathcal{A}$ is defined, and the classification function Γ of LLM is used to analyze the fused scene description

$$A(t) = \Gamma\left(\tilde{S}(t)\right), \quad A(t) \in \mathcal{A}. \tag{31}$$

Common anomaly types include pedestrian appearance ($A = 1$), object occlusion ($A = 2$), trajectory deviation ($A = 3$), and device nonresponse ($A = 4$). Fig. 3 presents four potential abnormal scenarios, where the first column

(a, d, h, j) illustrates the data captured by the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$, the second column (b, e, i, k) shows the corresponding semantic segmentation results $\mathbf{S}^{\mathrm{InstSeg}}(t)$, and the third column (c, f, j, l) displays the radar feature map $\mathbf{f}^{\mathrm{LiDAR}}(t)$.

For Anomaly 1, depicted in the first row, the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$ (a) reveals an unexpected pedestrian entering the operational area, potentially obstructing the agent's path and delaying its response. The semantic segmentation result $\mathbf{S}^{\mathrm{InstSeg}}(t)$ (b) identifies the pedestrian as an independent entity, distinguishing it from other objects, such as boxes. The LiDAR feature map $\mathbf{f}^{\mathrm{LiDAR}}(t)$ (c) captures the presence of the pedestrian, thereby triggering an alert due to the proximity of the obstacle.

In Anomaly 2, represented in the second row, the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$ (d) shows a large object (box) obstructing the agent's path, preventing task execution. The segmentation result $\mathbf{S}^{\mathrm{InstSeg}}(t)$ (e) classifies the box as an obstruction, indicating the necessity for the agent to reroute. The LiDAR map $\mathbf{f}^{\mathrm{LiDAR}}(t)$ (f) confirms the position of the object, necessitating replanning to avoid the obstacle.

For Anomaly 3, depicted in the third row, the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$ (h) indicates that the agent has deviated from its intended trajectory, failing to follow the predefined path. The segmentation result $\mathbf{S}^{\mathrm{InstSeg}}(t)$ (i) highlights that the agent's position has deviated from the expected path. The LiDAR feature map $\mathbf{f}^{\mathrm{LiDAR}}(t)$ (j) captures the trajectory deviation, indicating a need for recalibration.

Anomaly 4, presented in the fourth row, shows in the RGB image $\mathbf{I}^{\mathrm{RGB}}(t)$ (j) that the agent has stopped moving and failed to execute its task, suggesting a possible system malfunction. The segmentation result $\mathbf{S}^{\mathrm{InstSeg}}(t)$ (k) indicates no external interference, as no significant objects are detected. The LiDAR map $\mathbf{f}^{\mathrm{LiDAR}}(t)$ (l) further verifies the lack of environmental changes, confirming that the issue lies with the agent's nonresponsiveness.

After the abnormality is detected, the abnormality monitoring feedback mechanism is entered to ensure the system's robustness. If the agent's trajectory or posture error exceeds the predefined threshold, the system will mark the abnormality and replan to start the self-correction process.

The trajectory error is deemed anomalous if

$$\|\Delta\mathbf{p}_i(t)\| > \epsilon_{\mathrm{pos}}. \tag{32}$$

The attitude error is deemed anomalous if

$$\|\Delta\mathbf{q}_i(t)\| > \epsilon_{\mathrm{att}} \tag{33}$$

where ϵ_{pos} and ϵ_{att} represent the threshold values for position and attitude errors, respectively.

Upon detection of an anomaly, an environmental feedback signal $e_i(t)$ is generated, which is defined by the error function $f(\cdot)$

$$e_i(t) = f(\|\Delta\mathbf{p}_i(t)\|,\ \|\Delta\mathbf{q}_i(t)\|). \tag{34}$$

If $e_i(t) = 1$, indicating an anomaly, the system triggers the LLM to replan the task path and adjust control strategies. Agents continuously provide feedback on their states $\mathbf{S}_i(t)$, including actual trajectory $\mathbf{P}_i^{\mathrm{actual}}(t)$ and task progress. The LLM uses this feedback to update task planning and system state in real-time.

To implement the self-correction mechanism, define the trajectory error of agent i as

$$\mathbf{E}_i(t) = \mathbf{P}_i^{\mathrm{desired}}(t) - \mathbf{P}_i^{\mathrm{actual}}(t). \tag{35}$$

If the error magnitude exceeds a preset threshold ϵ

$$\|\mathbf{E}_i(t)\| > \epsilon \tag{36}$$

then the LLM adjusts the control instructions for agent i accordingly.

The control input is updated as follows:

$$\mathbf{u}_i(t) = \mathbf{u}_i(t) - k_p\Delta\mathbf{p}_i(t) - k_q\Delta\mathbf{q}_i(t) \tag{37}$$

where k_p and k_q are the position and attitude control gain matrices. Proper tuning of these control gains is critical for ensuring a fast response while maintaining system stability.

In multiagent cooperation, based on the output of the LLM, the path planning module generates a new path to adapt to the current environment. The path planning objective for agent i is defined by the goal position $\mathbf{p}_i^{\mathrm{goal}}$, and the path generation problem is formulated as the following optimization problem

$$\min_{\mathbf{p}_i(t)} \sum_{t=0}^{T} \|\mathbf{p}_i^{\mathrm{goal}} - \mathbf{p}_i(t)\|^2. \tag{38}$$

During the path planning process, obstacles are considered to ensure the safety of the agents. Specifically, the distance between an agent and the nearest obstacle $\mathbf{o}_j(t)$ must satisfy the following safe distance constraint

$$\|\mathbf{p}_i(t) - \mathbf{o}_j(t)\| \geq d_{\mathrm{safe}} \tag{39}$$

where d_{safe} is the minimum safe distance.

D. Workflow Overview

Algorithm 1 outlines the workflow of OC-HMAS, which achieves dynamic task allocation and self-organization through perception and planning loops. The system begins by processing input data, specifically RGB images $\mathbf{I}^{\mathrm{RGB}}(t)$ and LiDAR point clouds $\mathbf{P}^{\mathrm{LiDAR}}(t)$, which serve as the basis for environmental perception and decision-making.

The RGB images are initially processed using an InstanceSeg network to extract instance-level details, $\mathbf{S}^{\mathrm{InstSeg}}(t) = \mathrm{InstSegNet}(\mathbf{I}^{\mathrm{RGB}}(t))$. Concurrently, high-level scene descriptions are generated using VLMs, represented as $S^{\mathrm{RGB}}(t) = \mathrm{VLM}_{\mathrm{RGB}}(\mathbf{I}^{\mathrm{RGB}}(t))$ and $S^{\mathrm{InstSeg}}(t) = \mathrm{VLM}_{\mathrm{InstSeg}}(\mathbf{S}^{\mathrm{InstSeg}}(t))$. The final environmental representation, $\tilde{S}(t)$, is formed by combining the RGB-based, segmentation-based, and LiDAR-derived features, offering a comprehensive understanding of the scene.

The output from the Environmental Perception stage, namely $\tilde{S}(t)$, is seamlessly passed to the Role Assignment stage. Here, each agent's capabilities are evaluated based on $\tilde{S}(t)$, ensuring that task assignments are contextually relevant and optimized for the current environment. The Role Assignment output, $\mathbf{T}_i(t)$, is then utilized in the Task

Algorithm 1: OC-HMAS Workflow

Input : RGB images $\mathbf{I}^{\text{RGB}}(t)$, LiDAR point cloud data $\mathbf{P}^{\text{LiDAR}}(t)$

Output: Task allocation, path planning, control instructions

Extract InstanceSeg from RGB image using InstSegNet($\mathbf{I}^{\text{RGB}}(t)$);

Generate scene description using $\text{VLM}_{\text{RGB}}(\mathbf{I}^{\text{RGB}}(t))$ and $\text{VLM}_{\text{InstSeg}}(\mathbf{S}^{\text{InstSeg}}(t))$;

Fuse visual and LiDAR data to form $\tilde{S}(t)$;

for *each agent* i **do**

- Evaluate agent capabilities and assign roles;
- Decompose tasks based on scene $\tilde{S}(t)$;
- Assign specific task $\mathbf{T}_i(t)$ using $\text{LLM}(\tilde{S}(t), \{\mathbf{S}_j(t)\}_{j=1}^{N})$;
- Perform task decomposition for parallel execution;
- Plan path $\mathbf{P}_i(t)$ to minimize cost function $J_i(\mathbf{P}, \mathbf{T}_i^{\text{new}}(t), \tilde{S}(t))$;
- Ensure safety with constraint $\left\|\mathbf{p}_i(t) - \mathbf{o}_j(t)\right\| \geq d_{\text{safe}}$;
- Execute assigned tasks (e.g., logistics, inspection, search and rescue);
- Monitor task progress and report state $\mathbf{S}_i(t)$;
- Calculate trajectory error $\Delta\mathbf{p}_i(t)$ and attitude error $\Delta\mathbf{q}_i(t)$;
- **if** $\|\Delta\mathbf{p}_i(t)\| > \epsilon_{\text{pos}}$ *or* $\|\Delta\mathbf{q}_i(t)\| > \epsilon_{\text{att}}$ **then**
  - Generate feedback signal $e_i(t) = 1$;
  - Replan path and adjust control strategies using VLM & LLM;
  - Update control input: $\mathbf{u}_i(t) = \mathbf{u}_i(t) - k_p\Delta\mathbf{p}_i(t) - k_q\Delta\mathbf{q}_i(t)$;
- **end**

end

Decomposition stage to break down complex tasks into manageable subtasks tailored to each agent's strengths. This hierarchical flow ensures that each subsequent stage receives precise and actionable information, facilitating efficient task execution and adaptability.

Based on the comprehensive environmental state, each agent is assigned specific tasks using a LLM, with the assignment formulated as $\mathbf{T}_i(t) = \text{LLM}(\tilde{S}(t), \{\mathbf{S}_j(t)\}_{j=1}^{N})$. This approach allows dynamic task allocation by leveraging the environment's attributes and each agent's capabilities. Path planning aims to minimize a cost function J_i for each agent while considering task requirements and environmental factors, such as avoiding obstacles and optimizing task efficiency. Safety is enforced by maintaining a minimum distance from obstacles, denoted as d_{safe}. Once the tasks are planned, each agent follows the optimal path and continuously monitors its progress during execution.

A self-correcting feedback mechanism is implemented to ensure system robustness. Trajectory and attitude errors, denoted as $\Delta\mathbf{p}_i(t)$ and $\Delta\mathbf{q}_i(t)$, respectively, are computed to track deviations from desired paths. If these deviations exceed predefined thresholds, corrective actions are taken to adjust control inputs, such as updating $\mathbf{u}_i(t) = \mathbf{u}_i(t) - k_p\Delta\mathbf{p}_i(t) - k_q\Delta\mathbf{q}_i(t)$, thereby ensuring agents stay aligned with their intended routes.

The transitions between stages are managed to ensure data integrity and relevance. For instance, the Task Allocation output directly informs the Path Planning stage, where each agent's specific tasks dictate the optimization criteria for their respective paths. Additionally, feedback from the Self-Correcting Mechanism is relayed back to both Path Planning and Task Execution stages, allowing real-time adjustments that enhance overall system performance. This structured information flow is pivotal in maintaining synchronization across all stages, thereby optimizing task execution and responsiveness to dynamic environmental changes.

The output of this workflow includes task allocations, optimized paths $\mathbf{P}_i(t)$, and control instructions $\mathbf{u}_i(t)$ for each agent, thereby ensuring effective task execution and adaptability to environmental changes. In summary, Algorithm 1 effectively integrates multimodal perception, role assignment, path planning, and real-time feedback to enable OC-HMAS to operate adaptively in dynamic environments. To enhance the clarity and effectiveness of our framework, this section elucidates the training and parameter tuning strategies employed for the VLM and LLM. These strategies encompass the selection process, optimization methods, and the management of computational tradeoffs essential for achieving optimal performance.

Our framework leverages a VLM and a LLM to serve distinct roles in environmental perception and task planning. The VLM is trained through a two-stage strategy: 1) pretraining on ImageNet to establish general visual feature extraction capabilities, followed by 2) domain-specific fine-tuning using logistics-oriented datasets collected from real-world scenarios. This enables precise recognition of delivery-related objects and urban environments. Meanwhile, the LLM undergoes sequential optimization: initial pretraining on diverse textual corpora ensures linguistic competence, followed by domain adaptation with logistics-specific task protocols and multiagent coordination dialogues. Iterative fine-tuning based on system feedback further refines its ability to generate context-aware task allocation strategies.

To address computational constraints, we prioritize inference optimization over full retraining. Domain adaptation via targeted fine-tuning preserves performance without incurring prohibitive retraining costs. For distributed multiagent deployment, we implement a hybrid approach: critical decisions are assigned to high-capacity models (e.g., GPT-4o), while computationally intensive operations utilize lightweight classical algorithms. Dynamic resource allocation mechanisms balance real-time responsiveness and computational load, ensuring efficient utilization of hardware resources under varying task complexities.

V. Experiments

A. Experimental Setup

1) Experimental Scenarios: In this experiment, we developed and simulated three different scenarios—logistics,

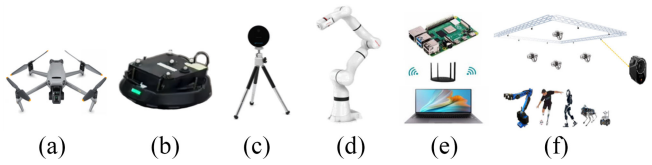


Fig. 4. Overview of hardware components, including AAVs, AVs, LiDAR sensors, robotic arms, and the communication network. Each component plays a critical role in multimodal data integration, supporting dynamic task execution and real-time feedback.

inspection, and rescue—each designed to test the ability of multiple intelligent agents to collaborate in real-world-inspired tasks. The following simulation experiments model each scenario in detail to ensure that agent behaviors and environments reflect real-world operational challenges. The following sections provide an overview of each scenario and its respective simulation setup.

Logistics Scenario: The main goal of this scenario is to simulate a coordinated transportation task where multiple agents, such as AAVs and AVs and warehouses work together. The system integrates AAVs and AVs to manage transportation and delivery tasks. The simulation also considers real-world conditions, such as road occupancy ranging from 5% to 20% and random vehicle movements to replicate a dynamic warehouse environment.

Inspection Scenario: In this scenario, the goal is to conduct coordinated facility inspection and monitoring. AAVs and AVs navigate along S-shaped curves and stop at inspection points. Wireless networks are used to communicate with fixed sensor nodes to ensure a comprehensive inspection process. The simulation introduces real-world complexities, including a 10% probability of communication delay and a 5% probability of data loss, to test the resilience of the system under challenging conditions.

Rescue Scenario: This scenario focuses on the delivery of relief supplies using a drone. The drone flies to the target airdrop, and the robotic arm receives the airdropped supplies and places them stably on the water surface at the final location. The main goal is to ensure that the supplies are delivered to the target surface vessel accurately and in a timely manner. To simulate real-world conditions, we use Monte Carlo simulation methods to simulate severe sea conditions and random wind direction changes to reflect the challenges of delivering supplies in an unpredictable environment.

2) Hardware Introduction: The primary hardware components of OC-HMAS are depicted in Fig. 4, comprising (a) AAV, (b) AV, (c) observation lidar, (d) robotic arm, (e) communication system, and (f) peripheral network. Within the three experimental scenarios, AAVs, AVs, and robotic arms primarily function as intelligent agents executing specific subtasks computed and dispatched by the VLM-LLM scheduling centre. The observation lidar is the primary sensor for visual data acquisition, while the communication system interconnects the central scheduler with each intelligent agent via Wi-Fi. These hardware components are integrated with advanced algorithms to ensure accuracy, efficiency, and reliability.

AAV: The system employs DJI Mavic 3 AAVs,[1] customized to meet specific scenario requirements. These AAVs have high-definition cameras, infrared sensors, and object-release mechanisms, facilitating precise route planning and targeted payload delivery. In both logistics and rescue scenarios, they handle aerial transport, receive route plans from the VLM-LLM scheduling centre, and are equipped with payload-release systems to ensure the accurate delivery of supplies.

AVs: For ground transportation in logistics scenarios, Our AV is based on the ROS intelligent robot Spark[2] with NVIDIA Jetson Xavier NX. These AVs use a hybrid control system—PID for basic trajectory tracking and NMPC for dynamic route optimization and collision avoidance.

Observation Lidar: The observation lidar plays a critical role in OC-HMAS, acquiring and detecting RGB and LiDAR data within the system. The gathered data, $\mathbf{I}$ and $\mathbf{P}^{\text{LiDAR}}$, is transmitted via the communication system to the scheduling centre for data processing. The observation Intel RealSense LiDAR Camera L515[3] is responsible for visual data acquisition and anomaly detection, supporting the self-correcting capability of OC-HMAS.

Robotic Arm: The ROKAE seven-axis robotic arm is a critical component in rescue scenarios, explicitly designed for aerial-sea collaboration in challenging water environments. It is equipped with force sensors and real-time motion positioning capabilities. It allows precise collaboration with AAVs to receive and place rescue supplies even in rough sea and windy conditions.[4]

Communication System: Reliable communication among agents is crucial in all scenarios. A 5G wireless network ensures low-latency, high-bandwidth communication between AAVs, AVs, warehouse robots, mobile robots, and robotic arms. The system also supports Wi-Fi 6 for long-distance, low-bandwidth communication where 5G coverage is limited, enabling seamless coordination, real-time feedback, and task adjustments in large or complex environments.

Perceptual Network: The perceptual network is established using a virtual reality perceptual network based on optical motion capture.[5] By equipping intelligent agents with marker sensing spheres, the perceptual network can accurately capture each agent's trajectory $\mathbf{p}_i$ and orientation $\mathbf{q}_i$. This setup ensures precise localization of agents and updates their positional information at a frequency of 180 Hz, guaranteeing real-time system responsiveness and robustness.

The OC-HMAS framework integrates various IoT devices to enhance task execution across logistics, inspection, and rescue scenarios. These include AAVs (DJI Mavic 3), AVs (ROS-based Spark robot), LiDAR sensors (Intel RealSense L515), robotic arms (ROKAE seven-axis), and a 5G-enabled communication system. The seamless data exchange between these devices and the central VLM-LLM scheduling centre ensures that information flows efficiently, enabling real-time

[1]https://www.dji.com/au/support/product
[2]https://www.nxrobo.com/ProDetail
[3]https://www.intelrealsense.com/lidar-camera-l515/
[4]https://www.rokae.com/en/product.html
[5]https://www.olshuzi.com/oulei/products.html

decision-making and task optimization. This integration supports dynamic task execution and real-time feedback, allowing the system to adapt swiftly to changing environments and task requirements. The integrated hardware setup, combined with advanced planning and control algorithms, ensures effective task execution and high system coordination across all scenarios, enhancing overall performance in logistics, inspection, and rescue operations.

3) Evaluation Metrics: In this experiment, we comprehensively evaluated the performance of each system across multiple task scenarios using five key performance metrics:

Task Success Rate (Success): This metric measures the system's reliability in successfully completing specific tasks. In the experiment, each system was run 100 times per scenario under different initial conditions and random environmental variables to simulate real-world uncertainties. The number of successful task completions was recorded to calculate the success rate.

Number of Task Executions (Steps): This metric tracks the average number of operational steps required by the system to complete a single task, encompassing subtasks, such as information processing, path planning, and execution. By repeating the tasks multiple times, we recorded the steps needed for each execution and computed the average task execution steps across different scenarios.

Task Completion Time (Completion Time): This metric measures the total time (in seconds) taken by the system from task initiation to successful completion, reflecting the system's overall efficiency. In each scenario, the completion time was recorded over multiple tests, and the average time was used to assess the system's responsiveness.

Adaptability: This metric evaluates the system's ability to self-adjust in response to dynamic environmental changes (e.g., the appearance of obstacles or changes in target location). During the experiment, we introduced sudden changes in the environment to observe whether the system could quickly adjust its strategy and effectively complete the task.

Scalability: To assess the system's performance stability under expanding task complexity or multiagent collaboration, this metric tests how the system handles increased task complexity or concurrent tasks. The system's performance in high-load scenarios was evaluated to quantify its scalability.

B. Experimental Statistics and Analysis in Simulation

The simulation-based experiments compared four task execution methods under standardized hardware and software configurations: LSTM-based heterogeneous multiagent methods (IC3Net [28]), RL-based systems (FACMAC [15]), LLM-based systems (SMRC-LLM [33]), and our proposed OC-HMAS. Evaluations were conducted across three scenarios—Logistics, Inspection, and Rescue—designed with PyBullet physics engine (v3.2.5) to simulate real-world operational challenges. To ensure comparability, all methods were deployed on identical virtual machines equipped with Intel Xeon Gold 6248R CPUs and NVIDIA A100 GPUs, using Ubuntu 20.04 LTS with Docker containers for dependency isolation. Random seeds were fixed across trials to control stochastic factors. Five metrics were quantified: Success Rate (task completion), Steps (action efficiency), Completion Time (s), Adaptability (% environmental variation tolerance), and Scalability (% performance retention under agent scaling).

In the logistics scenario, the OC-HMAS exhibited the highest success rate of 80.18%, significantly outperforming both the SMRC-LLM system with a success rate of 75.03% and the IC3Net approach with 61.79%. The efficiency of OC-HMAS is further evidenced by the average number of steps required, with OC-HMAS needing only 8.12 steps, representing a considerable reduction compared to FACMAC (10.87 steps) and IC3Net (12.34 steps). Additionally, the system's ability to integrate multimodal data contributed to a reduction in completion time to 16.84 s, highlighting its operational efficiency in the logistics domain.

An important discovery during the logistics experiments is the advantage of using VLM to analyze global environment states. By inputting RGB path images into VLM, the system captures both semantic and spatial context, enabling agents to anticipate and proactively resolve potential conflicts. This method improves global planning robustness, especially in dense obstacle scenarios. Furthermore, compared to FACMAC, which relies heavily on reinforcement learning and predefined policies, OC-HMAS demonstrates greater flexibility in adapting to unforeseen changes in the environment, thereby enhancing overall task success rates.

In the inspection scenario, OC-HMAS achieved a success rate of 83.31%, outperforming competing methods by 4.69 percentage points (78.62% to 83.31%). The system's adaptability and real-time decision-making capabilities are reflected in the reduced number of steps (6.12) and a completion time of 14.67 s, marking reductions of 17.63% and 12.89%, respectively, compared to SMRC-LLM's metrics (7.43 steps, 16.84 s). These quantitative improvements clearly highlight the advantages of integrating visual perception with language-based planning, contributing to superior adaptability (by 5.50 % points) and task efficiency.

The integration of LLM and VLM significantly outperforms standalone LLM approaches in this scenario. While LLM tracks agent positions, VLM enhances global scene understanding by analyzing fused multimodal data. Experiments show a 12% reduction in task execution time and a 10% decrease in anomaly-triggered delays when both models operate synergistically. Compared to IC3Net, which struggles with maintaining context over longer tasks, OC-HMAS maintains higher consistency and accuracy in task execution. Additionally, FACMAC's reinforcement learning approach, while effective in static environments, fails to adapt as efficiently to dynamic changes as OC-HMAS, underscoring the latter's superior adaptability in real-world applications. This validates the superiority of multimodal collaboration in dynamic environments.

In the rescue scenario, characterized by high-pressure and real-time operational constraints, OC-HMAS achieved the highest success rate at 69.53%. Furthermore, OC-HMAS consistently required fewer operational steps, averaging 9.87 steps, and completed the task in 18.56 s, outperforming the other

TABLE II
THIS TABLE COMPARES THE PERFORMANCE OF FOUR METHODS (IC3NET, FACMAC, SMRC-LLM, AND OC-HMAS) ACROSS THREE SCENARIOS: LOGISTICS, INSPECTION, AND RESCUE

Scenario	Property	IC3Net (LSTM-based)	FACMAC (RL-based)	SMRC-LLM (LLM-based)	OC-HMAS (Ours)
Logistics	Success	61.79%	68.21%	75.03%	**80.18%**
	Steps	12.34	10.87	9.76	**8.12**
	Completion Time	25.89s	22.45s	19.63s	**16.84s**
	Adaptability	54.21%	65.18%	72.72%	**75.18%**
	Scalability	60.45%	68.72%	70.13%	**73.21%**
Inspection	Success	64.62%	71.06%	78.62%	**83.31%**
	Steps	10.12	8.76	7.43	**6.12**
	Completion Time	22.18s	19.56s	16.84s	**14.67s**
	Adaptability	55.23%	67.18%	74.16%	**78.24%**
	Scalability	62.34%	70.42%	72.16%	**75.23%**
Rescue	Success	50.23%	57.29%	66.61%	**69.53%**
	Steps	15.78	13.12	11.45	**9.87**
	Completion Time	28.34s	25.67s	21.12s	**18.56s**
	Adaptability	52.12%	59.98%	66.45%	**68.21%**
	Scalability	55.23%	62.78%	63.23%	**65.53%**

methods across all metrics. These results underline the effectiveness of OC-HMAS in handling dynamic environments that require high adaptability and scalability.

Additionally, fine-tuning the model with scenario-specific datasets (e.g., specialized rescue missions) provided an approximate 5% improvement in task success rates. This demonstrates the value of adapting the framework to specific operational conditions for enhanced performance. Compared to SMRC-LLM, which relies on centralized processing and thus introduces latency in decision-making, OC-HMAS's distributed architecture allows for faster real-time responses. Moreover, while IC3Net and FACMAC perform adequately in controlled settings, their performance degrades in the unpredictable and rapidly changing conditions typical of rescue operations. OC-HMAS, leveraging multimodal data fusion and dynamic task allocation, maintains robust performance, highlighting its superiority in critical, time-sensitive scenarios.

Fig. 5 illustrates the simulated results generated by the proposed OC-HMAS system across three distinct case studies, each highlighting different collaborative tasks involving HMASs. The results demonstrate the system's ability to efficiently plan and execute coordinated tasks, ensuring seamless interaction between agents.

This scenario simulates a collaborative transportation task involving two heterogeneous agents, specifically a drone and an AV. In this task, the drone (depicted by the red dashed trajectory) and the AV (represented by the blue solid trajectory) are responsible for jointly transporting an object, whose position is shown by the green dashed line. The results highlight OC-HMAS's capability to effectively coordinate between the different agents, ensuring accurate path planning in a complex environment. Throughout the scenario, both agents synchronize their movements, avoiding collisions while maintaining efficient task execution. Notably, in the final time steps, the drone and the AV converge toward the target simultaneously, showcasing the system's excellent coordination and collaboration capabilities. The smooth convergence of trajectories demonstrates OC-HMAS's strength in handling dynamic multiagent tasks.

In intelligent inspection Scenario, the agents must navigate a cluttered environment with multiple obstacles and checkpoints. The drone's trajectory (red dashed line) and the AV's trajectory (blue solid line) are again displayed, with the inspection paths marked by the green dashed line. OC-HMAS successfully plans and executes the inspection task, with each agent efficiently performing its assigned role. The system demonstrates outstanding adaptability by distributing tasks between the drone and the AV based on real-time feedback. The drone performs aerial surveillance, providing the AV with continuous updates, which adjusts its path to avoid obstacles. This coordinated division of labor ensures the completion of all inspection checkpoints in an efficient manner, further proving the system's ability to manage complex, real-time tasks with dynamic constraints.

The final scenario models an emergency disaster relief task, where agents must rapidly transport supplies to affected areas. The drone's path (red dashed line) is responsible for delivering supplies, while the ground robot (blue solid line) ensures precise material placement in target locations. OC-HMAS exhibits remarkable path optimization and time efficiency in this high-pressure scenario. The drone navigates through difficult terrain, reaching the designated drop-off points swiftly. Meanwhile, the ground robot follows closely, ensuring precise and timely delivery of the relief supplies. The smooth, coordinated trajectory planning further validates the system's potential for real-world emergency response applications.

As shown in Fig. 6, during obstacle avoidance experiments, OC-HMAS demonstrated significant advantages in dynamic adaptation. By leveraging LiDAR and peripheral motion capture data, the system achieved real-time trajectory adjustments and avoided collisions effectively. In comparison, SMRC-LLM exhibited delayed responses, occasionally failing to detect or avoid obstacles in time. These results highlight the importance of multimodal integration for dynamic self-correction. Furthermore, compared to FACMAC, which relies on reinforcement learning policies that may not generalize well to unforeseen obstacles, OC-HMAS's multimodal perception

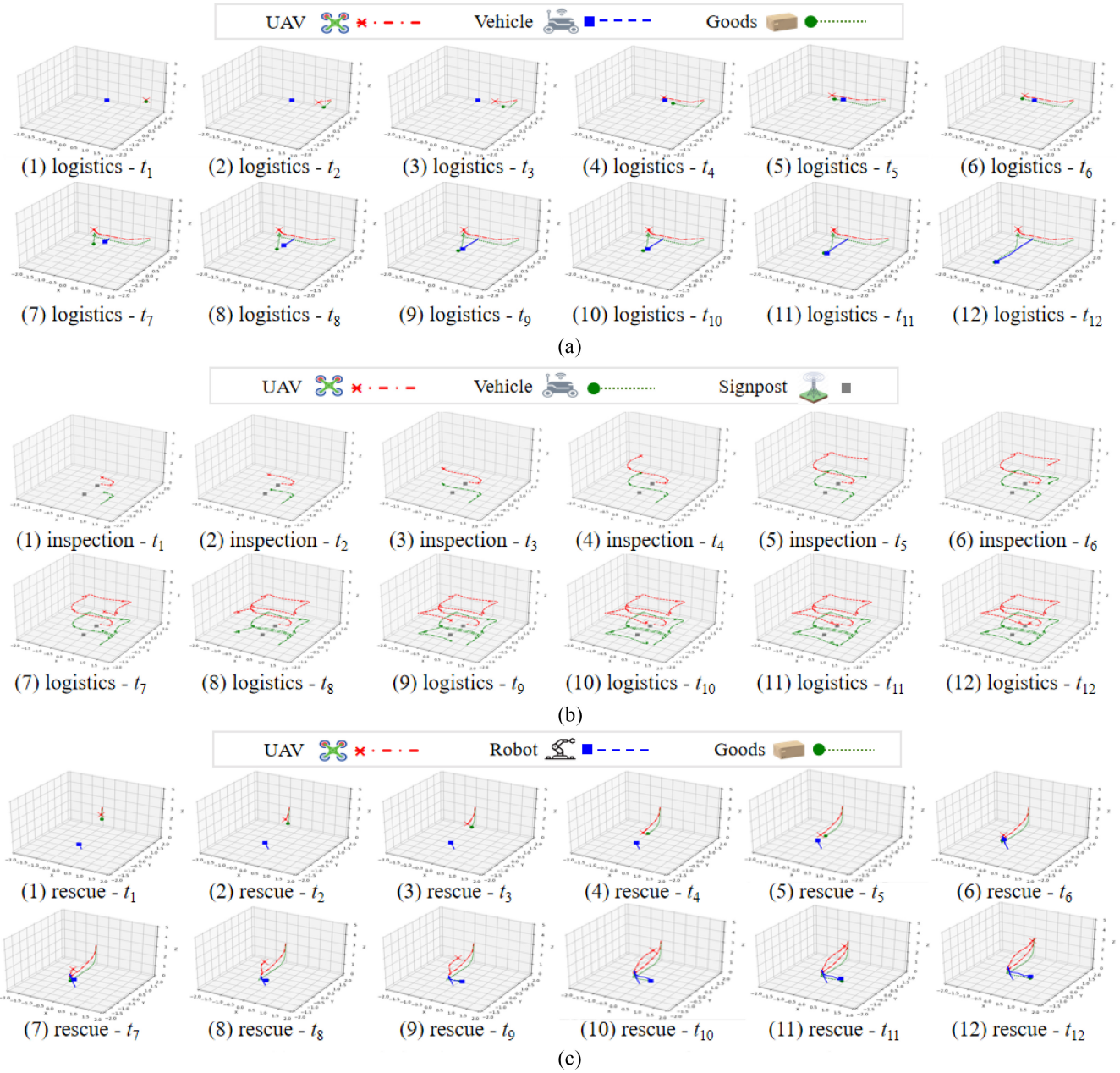


Fig. 5. Diagram illustrates the trajectory planning and collaboration of heterogeneous agents in three scenarios - Logistics, Inspection, and Rescue, managed by the OC-HMAS system over time steps from t_1 to t_{12}. In the Logistics scenario (a), a drone (red dashed line), an AV (blue solid line), and cargo (green dotted line) coordinate. (a).1 to (a).6 depict the drone transferring the package to the vehicle; (a).7 to (a).9 show the drone releasing the package for the AV to transport; (a).10 to (a).12 illustrate the AV transporting the package to its destination. The Inspection scenario (b) features a drone and a vehicle performing a figure-eight pattern around an inspection signpost (gray), demonstrating efficient real-time coordination. In the Rescue scenario (c), (c).1 to (c).5 show the drone (red dashed line) carrying a package (green dotted line) to a water robot, represented by the mechanical arm's end-effector path (blue solid line); (c).6 to (c).7 depict the drone releasing the package onto the water robot; (c).8 to (c).12 illustrate the robot arm securely placing the package at the designated location on the water surface, followed by the drone returning to its starting point. This comprehensive coordination across different scenarios highlights the OC-HMAS system's ability to manage complex multiagent interactions efficiently.

allows for more accurate and timely adjustments, ensuring safer and more reliable navigation in complex environments.

C. Real-World Experiments and Algorithm Validation

The experimental results presented in the figures provide a comprehensive demonstration of the proposed OC-HMAS algorithm across three distinct real-world scenarios: 1) Logistics; 2) Inspection; and 3) Search and Rescue. Each scenario illustrates the algorithm's superior capabilities in terms of real-time adaptability, scalability, and coordination among HMASs, highlighting the robustness of the proposed system in dynamic environments.

1) Logistics Scenario: In the logistics scenario, multiple agents, including AAVs, AVs (AVs), and warehouse robots, are tasked with the coordinated delivery of goods to a designated location. As shown in State 1, the system initiates the task by positioning Agent0 (drone) for transport, Agent1 (warehouse robot) for sorting, and Agent2 (AV) for environmental

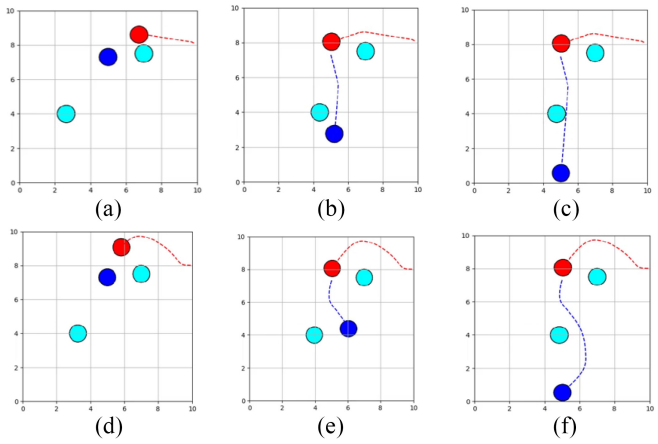


Fig. 6. Obstacle avoidance during logistics tasks, showcasing AAV and AV coordination. The red trajectory represents AAV movement, while the blue line indicates AV navigation. Cyan circles highlight static and dynamic obstacles. OC-HMAS ensures adaptive path planning to maintain safety and efficiency. Compared to SMRC-LLM, OC-HMAS offers superior real-time updates, enabling safer navigation.

monitoring. In subsequent states, the agents dynamically adapt to their roles, with Agent0 handling the delivery of goods, Agent1 completing the sorting process, and Agent2 monitoring the surroundings to ensure smooth task execution. The system continuously updates the perception and status of the agents, ensuring accurate path planning and obstacle avoidance. Upon reaching the final state, the goods are successfully delivered, and all agents return to their initial positions, ready for the next mission.

The results from this experiment underscore the scalability and coordination facilitated by the OC-HMAS algorithm. Despite the complexity of the logistics task, the agents operate autonomously and collaboratively, demonstrating the algorithm's ability to handle increased task complexity while ensuring efficient execution. Moreover, the system's ability to integrate multimodal sensory data enables precise and dynamic decision-making, ensuring real-time responsiveness to environmental changes.

2) Inspection Scenario: The inspection scenario involves AAVs and AVs collaborating to inspect predefined targets along a facility route. In State 1, Agent0 (AV) begins by navigating around the first signpost, while Agent1 (AAV) performs aerial monitoring to ensure comprehensive coverage. As the inspection progresses through States 2–5, the agents remain synchronized, with the AAV providing continuous oversight from above and the AV executing ground-level inspections. The system's perception module updates the status of the agents and inspection points in real time, ensuring that the agents proceed efficiently without redundant movement or misalignment.

This experiment highlights the precision and synchronization capabilities of the proposed algorithm. The OC-HMAS system ensures that both agents maintain optimal task alignment, avoiding delays or miscommunication between the aerial and ground units. Additionally, the system's adaptability to potential communication delays and sensor noise illustrates its robustness in complex real-world inspection tasks, where maintaining synchronization between heterogeneous agents is critical for operational success.

3) Rescue Scenario: In the search and rescue scenario, the task involves delivering rescue supplies via AAV, coordinated with a robotic arm (ROKAE) for precise handling in challenging environmental conditions near water surfaces. As illustrated in State 1, Agent0 (robotic arm) prepares to receive the goods, while Agent1 (AAV) positions itself above the robotic arm for delivery. The agents operate in parallel, with the AAV dynamically adjusting its flight path to account for real-time environmental factors, such as wind and surface disturbances. In State 3, the AAV carefully lowers the goods, and the robotic arm manipulates the supplies into position. Upon task completion, both agents return to their starting positions, indicating successful task execution.

The self-correcting mechanism of the OC-HMAS algorithm is particularly evident in this scenario. The system's ability to autonomously adjust the agents' paths and actions in response to real-time feedback ensures accurate and efficient task execution, even under unpredictable environmental conditions. This capability is crucial in high-risk environments, such as search and rescue operations, where timely and precise delivery of supplies can be critical.

D. Ablation Study

The ablation study results in Table III offer insights into the individual contributions of each core component within the OC-HMAS framework across three scenarios: 1) Logistics; 2) Inspection; and 3) Rescue. Specifically, the configurations analyzed include the complete OC-HMAS model and models with key components systematically removed—namely, the VLM, the LLM, and the self-correction mechanism. By isolating these elements, this study elucidates the impact of each component on performance metrics, such as Success Rate, Steps, Completion Time, Adaptability, and Scalability. This approach aids in understanding how each component strengthens the OC-HMAS architecture, enabling it to perform effectively in complex, real-world scenarios.

The results demonstrate the superior performance of the complete OC-HMAS model, where the integration of VLM, LLM, and the self-correction mechanism consistently enhances system metrics across all scenarios. In the Logistics scenario, for instance, the full configuration achieves a Success Rate of 80.18%, a Completion Time of 16.84 s, and an Adaptability score of 75.18%, notably surpassing models where individual components are removed. This performance highlights the crucial role of multimodal environment perception, supported by VLM, in enabling real-time, high-precision environmental awareness. Removing VLM (w/o VLM configuration) led to considerable reductions in Success Rate and Adaptability across all scenarios, underscoring the importance of robust multimodal sensing for maintaining efficiency and flexibility in diverse operational conditions.

Similarly, the LLM contributes significantly to the OC-HMAS system, enhancing high-level reasoning and dynamic task decomposition. This component enables sophisticated interagent communication and adaptive task allocation, which

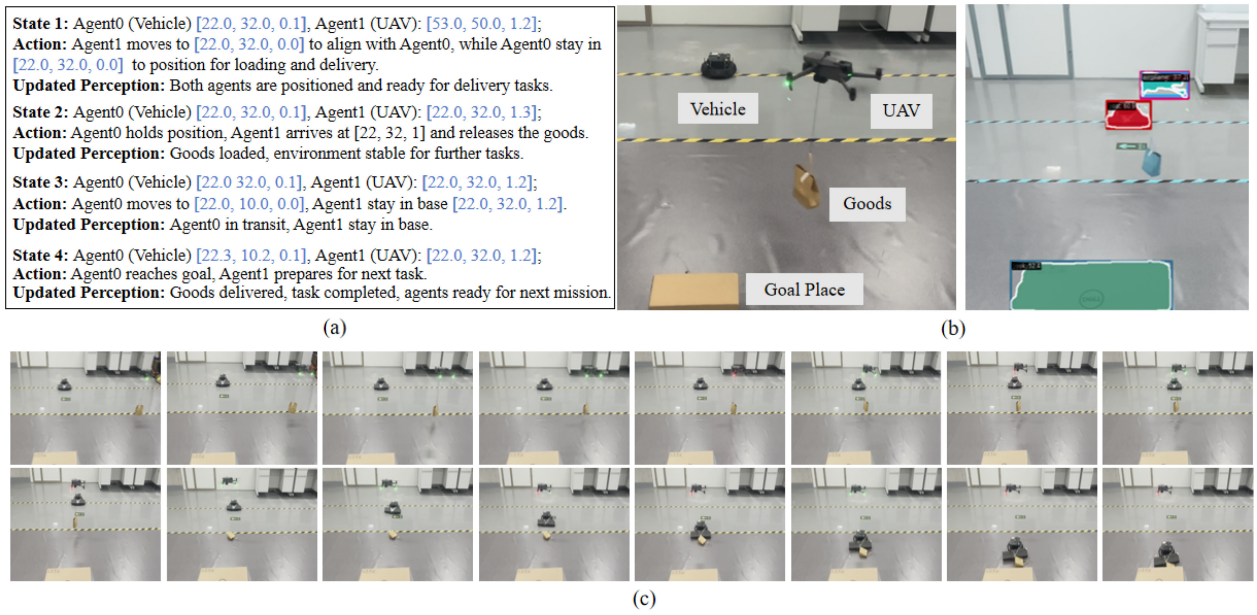


Fig. 7. In logistics task, agents (AAV, AV, etc.) autonomously organize to transport goods. The AAV picks up goods and coordinates with the AV to deliver them to the designated goal. (a) shows the self-organizing actions and state transitions of the agents. (b) displays the agent setup with markers and InstanceSeg detection results, ensuring accurate tracking of AAV, vehicle, and goods. (c) illustrates the real-world execution from t_1 to t_{16}, where the agents dynamically adjust actions to complete the logistics task efficiently.

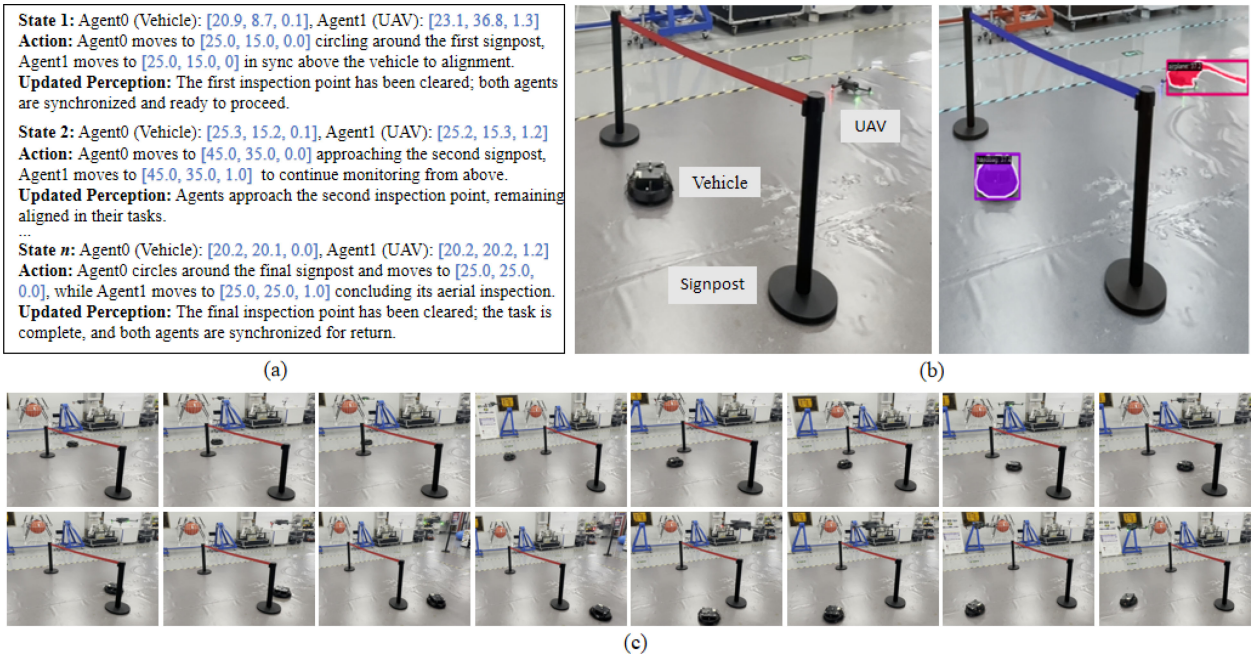


Fig. 8. In the inspection scenario, the OC-HMAS system directs a AAV and AV to inspect key locations. (a) demonstrates how the agents self-organize to perform synchronized aerial and ground inspections. (b) provides InstanceSeg results, showing accurate detection of agents and signposts. (c) depicts the real-world execution of the inspection task, where agents synchronize to inspect the signposts across t_1 to t_{16}, ensuring full coverage.

is critical for scenarios demanding real-time adjustments. When the LLM was removed (w/o LLM configuration), the system's self-correction capability was impaired, particularly in managing error mitigation and adaptive responses. This limitation resulted in increased Steps and Completion Time and reduced Adaptability, especially in the Inspection and Rescue scenarios, where dynamic adjustments are essential.

The self-correction mechanism also plays a vital role in enhancing system resilience, particularly in unpredictable environments, such as the Rescue scenario. Without this component, the system encountered higher rates of trajectory and orientation errors, evidenced by a decline in Success Rate and increased Completion Time. This continuous feedback loop ensures system stability and accuracy under variable conditions. Overall, the ablation study validates the architectural choices of OC-HMAS, showing that each component—whether for multimodal perception, large model-based self-correction, or dynamic task planning—collectively

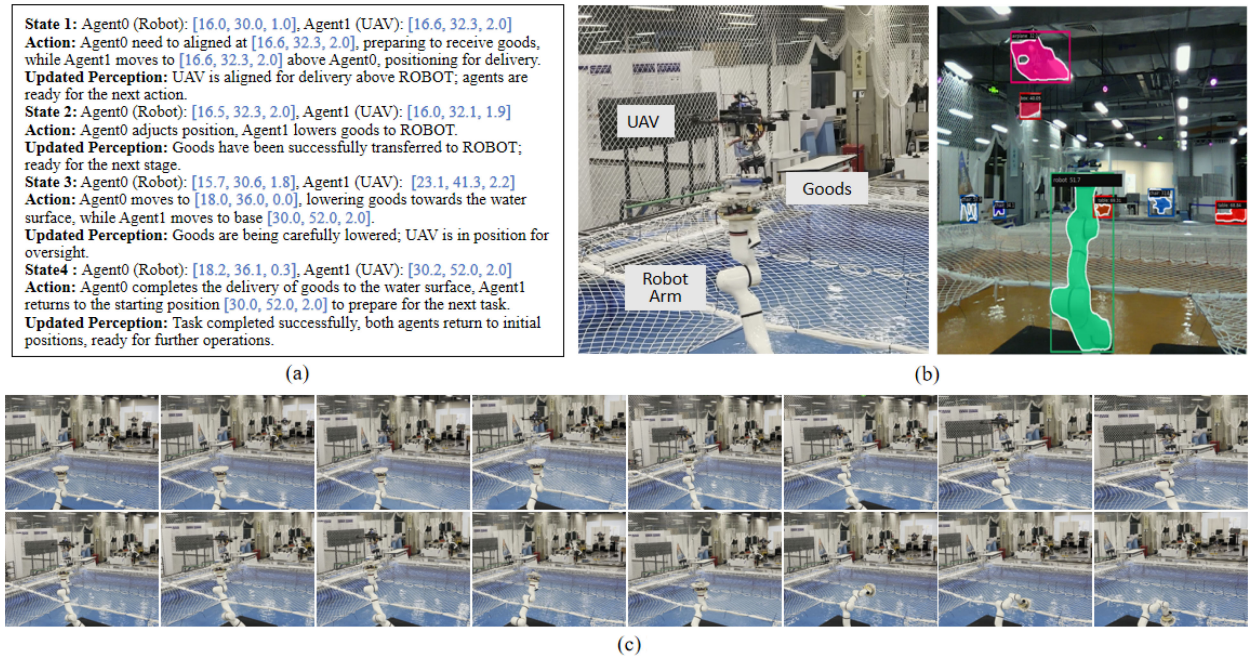


Fig. 9. In the rescue mission, OC-HMAS coordinates a AAV and robotic arm to deliver goods to a water surface. (a) shows the agents' self-organization, where the AAV transfers goods to the robotic arm. (b) illustrates the agents and goods detection using InstanceSeg, supporting precise task execution. (c) depicts the real-world scenario from t_1 to t_{16}, where the AAV and robotic arm work in harmony to complete the goods transfer in the dynamic rescue environment.

TABLE III
ABLATION STUDY RESULTS FOR OC-HMAS COMPONENTS IN LOGISTICS, INSPECTION, AND RESCUE SCENARIOS. THE TABLE EVALUATES SUCCESS RATE, ADAPTABILITY, AND COMPLETION TIME ACROSS DIFFERENT ABLATIONS OF KEY SYSTEM COMPONENTS: VLM, LLM, AND SELF-CORRECTION

Scenario	Configuration	VLM	LLM	Self-Correction	Success Rate	Steps	Completion Time (s)	Adaptability	Scalability
Logistics	Full Model	✓	✓	✓	**80.18%**	**8.12**	**16.84**	**75.18%**	**73.21%**
	w/o VLM		✓	✓	71.42%	9.78	19.46	62.45%	67.12%
	w/o LLM	✓		✓	68.21%	10.45	21.34	59.32%	65.67%
	w/o Self-Correction	✓	✓		73.67%	9.21	18.78	64.53%	69.78%
Inspection	Full Model	✓	✓	✓	**83.31%**	**6.12**	**14.67**	**78.24%**	**75.23%**
	w/o VLM		✓	✓	74.56%	7.45	16.89	65.78%	70.15%
	w/o LLM	✓		✓	71.06%	8.23	18.23	63.21%	68.54%
	w/o Self-Correction	✓	✓		76.78%	7.12	15.45	68.43%	72.89%
Rescue	Full Model	✓	✓	✓	**69.53%**	**9.87**	**18.56**	**68.21%**	**65.53%**
	w/o VLM		✓	✓	61.34%	11.12	21.23	57.45%	60.32%
	w/o LLM	✓		✓	57.29%	12.34	23.45	53.12%	58.78%
	w/o Self-Correction	✓	✓		63.87%	10.56	20.32	60.78%	62.34%

contributes to the system's robustness and Adaptability in handling complex, multiagent tasks.

VI. DISCUSSION

The proposed OC-HMAS framework leverages multimodal sensors, VLMs, and LLMs to enhance adaptability and efficiency in dynamic environments, but several challenges remain. Its reliance on cloud computing for model computation and training introduces latency risks, as real-time scheduling demands rapid computation, high network bandwidth, and low-latency connections. Continuous and stable network connectivity is essential for dynamic task allocation and feedback loops, making performance susceptible to bandwidth disruptions. Additionally, the system's agent localization capabilities must adapt to indoor and outdoor environments—while indoor applications rely on motion capture systems, outdoor scenarios depend on GPS, which is incompatible with indoor use. OC-HMAS focuses on GPS-limited indoor environments with stable networks, limiting its scalability to outdoor or poorly connected spaces. Future research should explore hybrid cloud-edge architectures to reduce latency and develop adaptive perception solutions for seamless indoor-outdoor transitions, broadening the applicability and autonomy of OC-HMAS across diverse scenarios.

VII. CONCLUSION

This article introduced OC-HMAS, a self-organizing and self-correcting MAS that leverages multimodal sensors, VLMs, LLMs, and IoT technologies to enhance adaptability and intelligence in dynamic environments. OC-HMAS addresses the limitations of traditional HMAS, such as limited autonomy and weak generalization, by enabling dynamic task decomposition, role assignment, and adaptive planning based on real-time environmental data. Integrating path planning,

obstacle avoidance, and self-correction mechanisms further strengthens the system's ability to operate efficiently and safely across complex scenarios. In rigorous testing, OC-HMAS achieved a 14.2% faster completion time in logistics than SMRC-LLM, demonstrating the significant algorithmic advantage of dynamic task planning and real-time self-correction. These results highlight OC-HMAS as a robust and scalable solution for advanced automation in real-world applications.

REFERENCES

[1] A. Amirkhani and A. H. Barshooi, "Consensus in multi-agent systems: A review," *Artif. Intell. Rev.*, vol. 55, no. 5, pp. 3897–3935, 2022.

[2] Y. Gao et al., "Asymmetric self-play-enabled intelligent heterogeneous multirobot catching system using deep multi-agent reinforcement learning," *IEEE Trans. Robot.*, vol. 39, no. 4, pp. 2603–2622, Aug. 2023.

[3] K. Zhang, Z. Yang, and T. Basar, "Decentralized multi-agent reinforcement learning with networked agents: Recent advances," *Front. Inf. Technol. Electron. Eng.*, vol. 22, no. 6, pp. 802–814, 2021.

[4] W. Koops, S. Junges, and N. Jansen, "Approximate Dec-POMDP solving using multi-agent A," 2024, *arXiv:2405.05662*.

[5] I. H. Ahmed et al., "Deep reinforcement learning for multi-agent interaction," *AI Commun.*, vol. 35, no. 4, pp. 357–368, 2022.

[6] D. Gao et al., "AgentScope: A flexible yet robust multi-agent platform," 2024, *arXiv:2402.14034*.

[7] A. Håkansson, Y. C. Dündar, and R. L. Hartung, "Towards robustness analysis for adaptive artificial intelligence in multi-autonomous agent systems," *Procedia Comput. Sci.*, vol. 225, pp. 4657–4666, Dec. 2023.

[8] G.-P. Antonio and C. Maria-Dolores, "Multi-agent deep reinforcement learning to manage connected autonomous vehicles at tomorrow's intersections," *IEEE Trans. Veh. Technol.*, vol. 71, no. 7, pp. 7033–7043, Jul. 2022.

[9] H. Wu, D. Qiu, L. Zhang, and M. Sun, "Adaptive multi-agent reinforcement learning for flexible resource management in a virtual power plant with dynamic participating multi-energy buildings," *Appl. Energy*, vol. 374, Nov. 2024, Art. no. 123998.

[10] L. Portilla et al., "Wirelessly powered large-area electronics for the Internet of Things," *Nat. Electron.*, vol. 6, no. 1, pp. 10–17, 2023.

[11] S. Lee and J. Kim, "Future directions in multi-agent systems: Enhancing adaptability and robustness," *IEEE Trans. Cybern.*, vol. 53, no. 2, pp. 1234–1245, 2023.

[12] B. Ren et al., "A multi-agents deep reinforcement learning autonomous security management approach for Internet of Things," *IEEE Internet Things J.*, vol. 11, no. 15, pp. 25600–25612, Aug. 2024.

[13] C. Yu et al., "The surprising effectiveness of PPO in cooperative multi-agent games," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 24611–24624.

[14] H. Li and H. He, "Multiagent trust region policy optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 9, pp. 12873–12887, Sep. 2024.

[15] B. Peng et al., "FACMAC: Factored multi-agent centralised policy gradients," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 12208–12221.

[16] Z. Tian et al., "Decompose a task into generalizable subtasks in multi-agent reinforcement learning," in *Proc. 37th Conf. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–19.

[17] N. Zhong et al., "CASIT: Collective intelligent agent system for Internet of Things," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19646–19656, Jun. 2024.

[18] M. Elhenawy et al., "Visual reasoning and multi-agent approach in multimodal large language models (MLLMs): Solving TSP and mTSP," *Mach. Learn. Knowl. Extr.*, vol. 6, no. 3, pp. 1894–1920, 2024.

[19] L. Zhang, B. Wang, Y. Zhao, Y. Yuan, T. Zhou, and Z. Li, "Collaborative multimodal fusion network for multiagent perception," *IEEE Trans. Cybern.*, vol. 55, no. 1, pp. 486–498, Jan. 2025.

[20] B. Jiang, Y. Xie, X. Wang, W. J. Su, C. J. Taylor, and T. Mallick, "Multimodal and multi-agent systems meet rationality: A survey," in *Proc. Workshop LLMs Cogn.*, 2024, pp. 1–16.

[21] M. Brienza, F. Argenziano, V. Suriani, D. D. Bloisi, and D. Nardi, "Multi-agent planning using visual language models," 2024, *arXiv:2408.05478*.

[22] W. Zhang, M. Cai, T. Zhang, Y. Zhuang, and X. Mao "EarthGPT: A universal multi-modal large language model for multi-sensor image comprehension in remote sensing domain," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5917820.

[23] G. Zhang, G. Yuan, D. Cheng, L. Liu, J. Li, and S. Zhang, "Mitigating propensity bias of large language models for recommender systems," 2024, *arXiv:2409.20052*.

[24] G. Zhang, S. Zhang, and G. Yuan, "Bayesian graph local extrema convolution with long-tail strategy for misinformation detection," *ACM Trans. Knowl. Discov. Data*, vol. 18, no. 4, pp. 1–21, 2024.

[25] Y. Zhou et al., "ALGPT: Multi-agent cooperative framework for open-vocabulary multi-modal auto-annotating in autonomous driving," *IEEE Trans. Intell. Veh.*, early access, Sep. 16, 2024, doi: 10.1109/TIV.2024.3461651.

[26] M. A. Johnson and M. H. Moradi, *PID Control*. London, U.K.: Springer, 2005.

[27] S. J. Qin and T. A. Badgwell, "An overview of nonlinear model predictive control applications," in *Proc. Nonlinear Model Predictive Control*, 2000, pp. 369–392.

[28] A. Singh, T. Jain, and S. Sukhbaatar, "Learning when to communicate at scale in multi-agent cooperative and competitive tasks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–15.

[29] A. Kirillov et al., "Segment anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.

[30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[31] T. Brown, M. Mann, and N. Ryder (OpenAI, San Francisco, CA, USA). *GPT-4o: A Robust and Efficient AI Model*. 2024. [Online]. Available: https://openai.com/research/gpt-4o

[32] T. Brown, M. Mann, and N. Ryder (OpenAI, San Francisco, CA, USA). *GPT-4 Technical Report*. 2023. [Online]. Available: https://openai.com/research/gpt-4

[33] Y. Chen, J. Arkin, Y. Zhang, N. Roy, and C. Fan, "Scalable multi-robot collaboration with large language models: Centralized or decentralized systems?" in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 4311–4317.

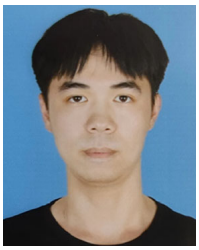
Ping Feng received the M.S. degree in Internet of Things Engineering from Dalian Maritime University, Dalian, China, in 2023, where she is currently pursuing the Ph.D. degree.

She is also affiliated with Pengcheng National Laboratory, Shenzhen, China, focusing her research on advanced artificial intelligence. Her work primarily revolves around multiagent collaboration using large language models, optimization of heterogeneous multimachine systems, and innovations in distributed network architectures, particularly within the Internet of Things domain.

Tingting Yang received the Ph.D. degree from Dalian Maritime University, Dalian, China, in 2010.

She currently holds the position of Professor with both the Navigation College of Dalian Maritime University and Peng Cheng Laboratory, Shenzhen, China. Since September 2012, she has been affiliated with the Broadband Communications Research (BBCR) Lab at the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, as a Visiting Scholar. Her research focuses on 6G networking, maritime wideband communication networks, DTN networks, and space-air-ground integrated systems.

Prof. Yang serves as an Associate Editor for *IEEE Network Magazine*, *SpringerPlus*, and *IET Communications*. She has also contributed as a Guest Editor for the IEEE INTERNET OF THINGS JOURNAL. Her involvement extends to serving as a TPC Member for IEEE ICC'14 and ICC'15. Furthermore, she has been invited to serve as a Tutorial Co-Chair for IEEE ICC 2021, ICC 2023, and ICCC 2019. Notably, she is the initiator of the Wireless Edge Intelligence KubeEdge Wireless (KEW) project within the global open-source K8S community.



Mingyang Liang is currently pursuing the bachelor's degree with Shenzhen Technology University, Shenzhen, China.

In 2024, he interned with the Special Robotics Center of the Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen. His research focuses on autonomous aerial vehicles (AAVs) and heterogeneous multiagent collaboration, with a particular interest in AAV control applications and the perception and decision-making processes of autonomous vehicles.

Lin Wang received the master's degree from the School of Mechanical and Power Engineering, Guangdong Ocean University, Guangdong Provincial Ocean Equipment and Manufacturing Engineer Technology Research Center, Zhanjiang, China, in 2022.

He is currently an Assistant Engineer with the Shenzhen Institute of Artificial intelligence and robotics for Society, Shenzhen, China. His research interests include reinforcement learning, embodied artificial intelligence, heterogeneous multirobot system, and socialized heterogeneous multimachine system.

Yuan Gao received the Ph.D. degree in the area of machine learning and robotics from Uppsala University, Uppsala, Sweden, under the supervision of Prof. Ginevra Castellano and Prof. D. Kragic.

He is currently an Associate Research Scientist with the Shenzhen Institute of Artificial intelligence and Robotics for Society, Shenzhen, China, and a part-time Assistant Professor with Chinese University of Hong Kong, Shenzhen. He has authored or co-authored different IEEE journals (e.g., IEEE TRANSACTIONS ON ROBOTICS, *IEEE/ASME Transactions on Mechatronics*, IEEE ROBOTICS AND AUTOMATION LETTERS). His research interests include machine behavior analysis, reinforcement learning, robotics, and general machine learning. His research interests include machine behavior analysis, reinforcement learning, robotics, and general machine learning.