DUOMIND: ENABLING DISTRIBUTED MULTI-ROBOT COORDINATION WITH SEMANTIC COMMUNICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language models (VLMs) and vision–language–action models (VLAs) have recently driven rapid progress in general-purpose robots, yet most progress has focused on single-robot settings. Extending these capabilities to multi-robot systems remains challenging because robots must coordinate long-horizon behaviors while maintaining reliable, fine-grained execution. We introduce DuoMind, a distributed hierarchical framework for multi-robot coordination through semantic communication. Each robot uses a VLA-based action model for low-level execution and a VLM-based orchestrator for high-level reasoning and inter-agent coordination. At each planning step, the orchestrator at each robot reasons over the task instruction, local observations, and messages received from other robots. It then generates low-level instructions for the action model and semantic messages for peer robots. This architecture exploits the complementary strengths of pretrained models by combining the semantic reasoning capabilities of VLMs with the precise action-generation capabilities of VLAs. To address the scarcity of benchmarks for multi-robot coordination, we further develop RoboPoly, a benchmark comprising long-horizon manipulation tasks that require coordinated, closed-loop execution under distributed control. Experiments on RoboPoly and RoboTwin demonstrate that DuoMind improves multi-robot task performance, while ablation studies confirm the contributions of hierarchical orchestration and semantic communication. More details are available on our project page.

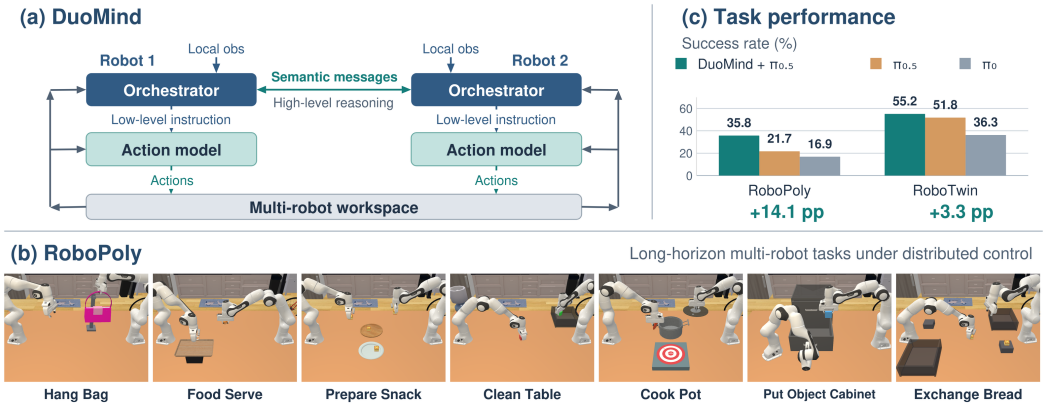


Figure 1: (a) An illustration of DuoMind, a distributed framework combining an orchestrator and action model for multi-robot coordination. (b) RoboPoly is a benchmark we develop for long-horizon coordination under distributed robot control. (c) DuoMind consistently outperforms the baselines across both benchmarks.

1 INTRODUCTION

Recent advances in vision-language models (VLMs) and vision-language-action models (VLAs) have substantially expanded the capabilities of general-purpose robots. VLMs (Beyer et al., 2024; Yang et al., 2025) provide strong visual understanding and high-level reasoning, while VLAs translate visual observations and natural-language instructions into robot actions (Zitkovich et al., 2023;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025b; Team et al., 2025). Building on these complementary capabilities, recent design of robot agents (Song et al., 2025a; Intelligence et al., 2025b; Team et al., 2025; Intelligence et al., 2025a) combines a high-level embodied reasoning model with a low-level VLA policy. In this design, the reasoning model aims to understand the environment and then decompose complex tasks into more specific instructions, while the VLA executes these instructions through fine-grained robot actions. Such a separation of functionalities allows the high-level model to focus on semantic understanding and long-horizon planning, while the VLA specializes in executing detailed instructions with precise low-level control.

Notably, most of the above progress has focused on a single robot acting in isolation, but many practical tasks are infeasible for a single agent and inherently require multi-robot collaboration, and others can benefit from multiple robots working together to improve efficiency. Effective multi-robot coordination requires each robot to construct appropriate subgoals and continuously adapt its behavior as the shared workspace evolves. For many practical settings, distributed execution (Amirkhani & Barshooi, 2022; Lo et al., 2024; Chen et al., 2024a) is particularly desirable, where each robot makes decisions primarily from its own observations rather than relying on a centralized controller, making the system more scalable as the number of robots increases (Nayak et al., 2023). Yet combining distributed multi-agent decision making with complex robotic manipulation creates a particularly challenging coordination problem, requiring robots to reason about both team-level cooperation and precise low-level task execution.

Conventional multi-agent learning and planning methods are often studied in relatively simple environments (Feng et al., 2026; Li et al., 2024; Zou et al., 2025; Wong et al., 2023; Liang et al., 2025a), with limited physical interaction and shorter task horizons. More recently, language-model-based robotic systems have leveraged the high-level reasoning capabilities of LLMs to decompose complex objectives and schedule robot behaviors (Chen et al., 2024b; He et al., 2026; Song et al., 2025b; Liu et al., 2025; Mandi et al., 2024). However, most of these systems focus primarily on high-level planning, while low-level control is executed through simply predefined action primitives or rule-based skills. An important gap remains between high-level multi-agent reasoning and end-to-end robot control. In addition, existing benchmarks provide limited support for systematic evaluation of multi-robot coordination, leaving the field without a comprehensive testbed for measuring such capabilities.

To address the above challenges, we introduce DuoMind, a distributed hierarchical framework for multi-agent robotic coordination, together with RoboPoly, a benchmark designed specifically for long-horizon multi-robot coordination. Each robot in DuoMind operates with a VLM as its orchestrator for high-level reasoning and a VLA as its action model for low-level control. At each planning step, each orchestrator reasons over the high-level instruction, its local observation, and the shared message from other robots. It then generates both a low-level instruction for the action model and a structured message for other robots. These two outputs serve complementary roles. The low-level instruction guides the action model to produce fine-grained robot actions, while the message communicates the robot's intention, subgoal, and task belief, allowing each agent to remain aware of its peers despite partial observability (Liang et al., 2025b) in the distributed setting. In this way, DuoMind provides an agentic harness around the action model: the orchestrator dynamically reasons and invokes the action model to realize long-horizon coordinated behavior at a relatively low frequency, while the action model focuses on high-frequency execution. Importantly, this decoupled design allows DuoMind to capitalize on the strong pretrained capabilities of existing VLM and VLA backbones. Rather than heavily fine-tuning models specifically for multi-robot coordination, high-level coordination and low-level manipulation can be largely handled by off-the-shelf VLM models, requiring only limited adaptation to the multi-robot setting. To systematically evaluate the coordination capabilities, we introduce RoboPoly, which provides challenging long-horizon manipulation tasks in which multiple robots must coordinate and complete the task through closed-loop physical execution rather than high-level planning alone. We further evaluate DuoMind across diverse cooperative scenarios and systematically study the contributions of hierarchical design and inter-agent communication.

The main contributions of this paper are summarized as follows:

- We introduce DuoMind, a distributed hierarchical framework for end-to-end multi-agent robotic coordination. DuoMind utilizes a VLM as the orchestrator at each robot for high-level reasoning and a VLA as the action model for low-level control, enabling inter-agent

semantic communication with natural language. This design harnesses the strong reasoning and control capabilities of existing foundation models to enable effective coordination in complex multi-robot tasks.

- We develop RoboPoly, a benchmark for long-horizon multi-robot coordination under distributed observations and control. The benchmark emphasizes closed-loop physical execution and includes tasks that are challenging for a single agent, requiring multiple agents to coordinate and schedule their behaviors throughout task execution.
- We conduct extensive experiments and ablation studies to evaluate DuoMind across challenging cooperative manipulation tasks on both RoboPoly and RoboTwin. Our results demonstrate the effectiveness of the proposed hierarchical design in multi-robot coordination and further quantify the contributions of integrating orchestrator and inter-agent communication.

2 DuoMind: Distributed Hierarchical Multi-Agent Framework

We introduce DuoMind, a distributed hierarchical framework for multi-agent robotic coordination. We first present the VLM-based orchestrator for high-level planning (2.1). We then describe how we bridges high-level planning and low-level control via embodied reasoning and agentic harness (2.2). Finally, we highlight the structured inter-agent communication (2.3).

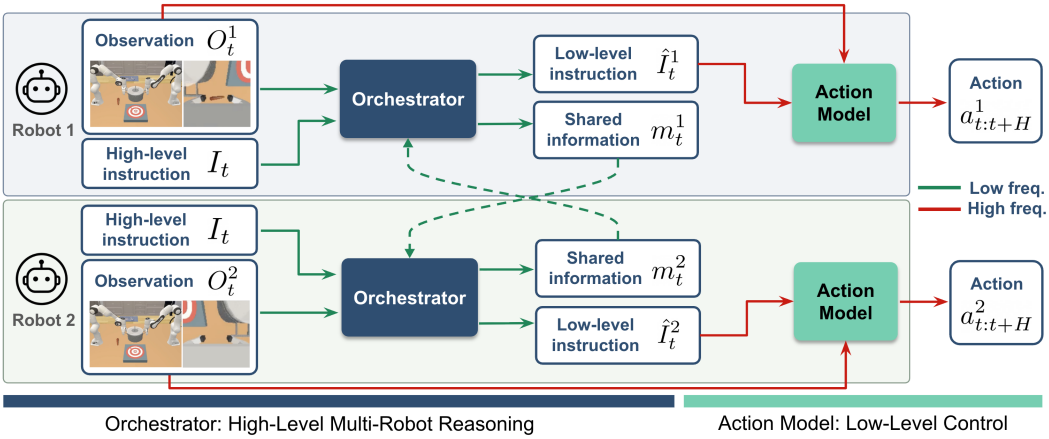


Figure 2: In DuoMind, each robot operates with a VLM as its orchestrator for high-level reasoning and a VLA as its action model for low-level control. At each high-level planning step, the orchestrator at each robot reasons and generates both a low-level instruction for the action model and a structured message for other robots. The action model subsequently follows the low-level instruction and produces fine-grained actions.

2.1 Multi-Agent Coordination via Embodied Reasoning

Existing studies on hierarchical robotic architectures have shown that action models are effective at low-level control and instruction following when provided with specific commands (Team et al., 2025). To extend this capability toward more general and long-horizon tasks, a promising approach is to introduce a reasoning model as an orchestrator, leveraging its strong visual understanding, semantic grounding, and planning capabilities to generate appropriate instructions for the action model. In distributed multi-agent settings, where task coordination are substantially more complex, we extend this hierarchical paradigm to distributed multi-robot coordination in DuoMind.

Specifically, DuoMind decomposes multi-agent coordination into two complementary levels: high-level multi-agent reasoning and low-level execution. Each robot employs a VLM as an orchestrator that reasons over the task-level instruction, its local visual observation, and message shared by other agents. The orchestrator determines the robot's current subtask and translates it into a single-agent, short-horizon instruction for low-level control. The VLA, serving as the action model, then follows the low-level instruction using local observations to generate fine-grained robot actions. This division of responsibility allows each module to focus on the capability for which it is best suited: the

orchestrator handles high-level understanding and inter-agent coordination in multi-agent setting, while the action model performs reliable low-level control.

Importantly, this decomposition avoids requiring the action model to directly interpret complex multi-agent state that includes the joint behavior of multiple robots and environment, which can be extremely challenging since action models are primarily trained with single-agent demonstrations. It therefore decouples multi-agent reasoning from low-level control, reducing the need for the action model itself to possess strong reasoning capabilities for multi-agent coordination. Moreover, DuoMind does not require the orchestrator, the action model, or the interface between them to be extensively redesigned for multi-agent settings. These foundation models developed for single-agent systems can be reused largely off the shelf, with only lightweight adaptation when necessary. This greatly extends the possibility of deploying DuoMind to diverse multi-robot tasks and offers strong compatibility to future upgrades on components.

The design of DuoMind naturally supports coordination among multiple robots. However, the complexity of task design, data collection, and evaluation increases substantially as the number of agents grows. Considering the high computational cost when the number of robots scales up, we focus on two-agent settings in this work and leave coordination at larger team scales for future study.

2.2 Bridging High-Level Reasoning and Low-Level Control via Natural Language Based Semantic Communication

Hierarchical robotic architectures bridge high-level reasoning and low-level control through different intermediate representations, including natural-language subgoals or motion descriptions (Shi et al., 2025; Belkhale et al., 2024), spatial representations such as trajectories or visual subgoals (Li et al., 2025; Zhao et al., 2025), and learned latent representations in tightly coupled dual-system models (Chen et al., 2025a; Bjorck et al., 2025). Among these approaches, natural language offers interpretability and broad compatibility with language-conditioned models, whereas other representations can convey compact information but typically require stronger coupling between the two modules, making them harder to replace or adapt across architectures and embodiments.

Our framework adopts natural language as a unified interface between the high-level orchestrator and the low-level action model. The orchestrator harnesses the action model by generating low-level instructions in natural language, thereby decoupling high-level reasoning from low-level control. This design allows existing VLMs and VLAs to be incorporated as off-the-shelf components with minimal adaptation, without requiring them to be jointly trained through a specialized interface. It also enables either component to be independently replaced as stronger reasoning or action models become available.

Natural language further provides a common interface for inter-agent semantic communication. Each orchestrator can interpret messages from other agents even when they use different reasoning models, facilitating coordination across heterogeneous agents. The same interface also supports human participation by allowing users to inspect or revise instructions using ordinary language. More broadly, this shared linguistic interface provides a scalable foundation for future multi-robot systems, where heterogeneous robots and multiple human users can coordinate through a common communication protocol.

2.3 Inter-Agent Information Sharing

Distributed multi-agent systems face the challenges of partial observability and non-stationarity: each agent has only partial knowledge of the global task state, while the environment continues to evolve as other agents act. Insufficient awareness of such information can lead to conflicting actions or ineffective collaboration. To address these challenges, DuoMind enables agents to share and integrate structured natural-language messages during high-level reasoning. These messages convey key information about other agents' decisions and task-relevant changes in the environment, providing additional context for more precise high-level reasoning. To support effective decision-making, we structure each shared message using four fields:

Intention. It is identical to the low-level instruction sent to the action model and describes the next subtask that the ego agent intends to execute. Sharing this intention makes other agents aware of the ego agent's next action, helping them avoid behaviors that may interfere with its execution.

Subgoals. This field specifies the expected task state after the ego agent completes its current subtask. It explicitly communicates the future state that the ego agent aims to achieve, allowing other agents to act proactively to facilitate task completion or avoid undesirable outcomes.

Belief of task. This part describes task-relevant information observed or inferred by the ego agent, such as the location of a key object or whether a particular subtask has been completed. For example, in a kitchen task, a robot operating near the cabinets may inform another robot preparing to cook that "the pot is in the third cabinet." Sharing such information reduces the impact of partial observability by propagating locally available knowledge across agents.

Uncertainty of belief. This optional field communicates the uncertainty associated with the ego agent's task belief. It is particularly useful in complex tasks, where information inferred through reasoning may not always be reliable. This uncertainty provides an additional confidence signal that allows the orchestrator to appropriately weigh different beliefs during high-level reasoning, especially when agents hold inconsistent beliefs or when the available information is ambiguous.

Through this structured information sharing, DuoMind provides each agent with compact contextual information about the intentions, subgoals, beliefs, and associated uncertainty of other agents. This additional context enables the orchestrator to generate more precise low-level instructions, thereby improving the safety and efficiency of multi-robot coordination.

3 ROBOPOLY

RoboPoly is designed to facilitate research on multi-robot manipulation by providing challenging long-horizon tasks with explicit cooperation requirements. Unlike RoboTwin Chen et al. (2025c), which primarily targets dual-arm manipulation with two arms on the same embodiment and does not necessarily require explicit cooperation, RoboPoly considers two independent robots and tasks specifically designed around multi-robot coordination. These tasks are either infeasible or inefficient for a single robot to complete independently, requiring each robot to reason and determine when cooperation is needed, and act at the appropriate time.

RoboPoly highlights distributed execution, where each robot acts based on its own observations and policy while cooperating with another robot. The benchmark is built upon the ManiSkill3 framework and contains seven multi-robot manipulation tasks. In this section, we describe the task suite, basic settings, and dataset generation.

3.1 TASKS FOR MULTI-ROBOT COORDINATION

All tasks share a common workspace configuration with two robot arms positioned on opposite sides of a table. The workspace and object placement are designed such that a single robot cannot always complete the task independently or efficiently, making cooperation necessary. RoboPoly contains the following seven tasks: Food Serve, Prepare Snack, Clean Table, Cook Pot, Put Object Cabinet, Hang Bag, Exchange Bread. We further detail the task description in A.

3.2 OBSERVATIONS, ACTIONS, AND ROBOTS

Each environment contains three RGB cameras: one global camera with a resolution of 512×384 and two wrist cameras with a resolution of 128×128. The global camera is positioned beside the table to provide an overview of the workspace, while each robot is equipped with a wrist camera that provides an egocentric view of its gripper. We adopt a distributed observation setting in which each robot receives observations from the global camera and its own wrist camera, but has no access to the wrist camera of the other robot. This design follows the distributed system setting and requires the robots to coordinate using only their locally available observations.

Both robots use 7-DoF Franka Emika Panda manipulators equipped with grippers. Under the distributed control setting, each policy controls only one robot rather than directly generating the joint actions of both robots. To improve task diversity, key objects are randomized within predefined task-specific regions at the beginning of each episode.

3.3 DATASETS

We construct the datasets using the motion-planning-based trajectory generation pipeline provided by ManiSkill3. For each task, we first decompose the full manipulation procedure into a sequence of key states, such as grasping, releasing, or moving toward a target location. Given these key states, the generation script invokes the motion planner to construct smooth robot trajectories between consecutive states. The spatial and temporal constraints for each trajectories are carefully designed with the planner to generate the desired multi-robot coordination demonstrations.

In addition to robot trajectories, we associate each manipulation stage with a low-level instruction for each robot describing the current behavior. We also provide a high-level instruction that specifies the overall task objective. During data generation, the pipeline first plans a successful trajectory and records the corresponding robot actions and states. It then replays the trajectory in simulator to render the RGB camera observations. The resulting dataset therefore contains visual observations, states, robot actions, high-level, and low-level language instructions for both robots. We further open-source the dataset to foster research on multi-robot coordination.

4 EXPERIMENTS

We evaluate DuoMind on 7 tasks from our RoboPoly and 8 dual-arm tasks from RoboTwin 2.0 (Chen et al., 2025c), both under distributed control. Since RoboTwin originally adopts centralized control over the dual-arm robot, we adapt its setting by treating the two arms as independent robots and control them separately. Task overview are shown in fig. 5 and fig. 6. All the finetuning and evaluation tasks are conducted on 4 NVIDIA A100 GPU, each taking 10–24 hours.

In the RoboPoly experiments, each Franka Emika Panda robot has 7 DoF and is equipped with a wrist camera, while both robots share a global camera positioned beside the workspace. We adopt the same distributed setting in which each robot has its own observations and policy. For each task, we collect 50 dual-robot demonstrations using planning-based trajectory generation and then decompose them into a unified single-robot dataset containing 100 trajectories for finetuning.

On RoboTwin, we select the tasks that require the coordination of two arms for multi-robot experiment. We treat the dual-arm Aloha-AgileX as two individual robots, each operating with its own observations and independently controlled by a separate policy. Each arm has 6 DoF with a wrist camera and a shared head camera. At finetuning stage, we use 50 official dual-arm demonstrations on each task. Each demonstration is decomposed into separate left- and right-arm trajectories, where each trajectory is paired with observations from the corresponding wrist camera and the shared head camera. We then merge the trajectories from both arms into a unified single-arm dataset, resulting in 100 demonstrations per task.

4.1 EXPERIMENT SETTINGS

Low-Level Control. We adopt $\pi_{0.5}$ (Intelligence et al., 2025b) as the low-level VLA controller in DuoMind due to its strong grounding in single-arm manipulation and generalization capability across diverse tasks and environments. We fine-tune $\pi_{0.5}$ using LoRA with batch sizes ranging from 256 to 1024 for 2k–10k training steps on RoboPoly and with batch sizes 32 for 20k training steps on RoboTwin, depending on the task. Detailed fine-tuning configurations are provided in D.

In RoboPoly experiments, the dataset includes low-level instructions, which are directly used to fine-tune the VLA. In RoboTwin experiments, however, the official dataset does not include low-level instructions and instead provides a high-level instruction, which we use during fine-tuning. At evaluation time, we retain the same hierarchical setting as in RoboPoly and provide decomposed low-level instructions to the VLA. Although these instructions does not fully match the fine-tuning dataset, the decomposition still benefits execution through the inherent generalization capability of the pretrained VLA. We further elaborate on this effect in section 4.2.

High-Level Reasoning. We integrate Qwen3-VL-4B-Instruct (Bai et al., 2025) as the high-level reasoning module in DuoMind, as it provides strong visual grounding while remaining compact enough for practical inference. To prevent the model from generating low-level instructions that exceed the instruction-following capability of the VLA policy or fall outside its training distribution,

Table 1: Comparison of success rates on RoboPoly tasks.

Method	Hang Bag	Food Serve	Prepare Snack	Clean Table
DuoMind w/ $\pi_{0.5}$	**78.00**	**32.75**	**18.00**	**24.75**
$\pi_{0.5}$ only	52.25	29.75	16.25	16.50
π_0 only	58.75	22.25	3.75	0.00

Method	Cook Pot	Put Object Cabinet	Exchange Bread
DuoMind w/ $\pi_{0.5}$	**39.25**	**26.50**	**31.00**
$\pi_{0.5}$ only	1.00	18.50	17.50
π_0 only	0.25	14.50	18.50

Table 2: Comparison of success rates on RoboTwin tasks.

Method	Hanging Mug	Pick Diverse Bottles	Stack Bowls Two	Put Bottle Dustbin
DuoMind w/ $\pi_{0.5}$	**24.75**	**58.50**	**85.50**	**34.50**
$\pi_{0.5}$ only	17.25	48.50	82.50	31.25
π_0 only	12.50	42.50	33.75	21.25

Method	Lift Pot	Handover Mic	Blocks Ranking RGB	Put Object Cabinet
DuoMind w/ $\pi_{0.5}$	**94.75**	**98.00**	**2.00**	43.25
$\pi_{0.5}$ only	94.00	94.75	0.00	**46.50**
π_0 only	77.50	84.50	0.25	18.00

we restrict each generated instruction to a predefined set of admissible examples. This constraint improves the reliability and consistency of low-level execution. In addition, to prevent the system from remaining indefinitely at an intermediate stage, we introduce a simple progress mechanism that forces the high-level instruction to advance when reasoning stagnates.

Baseline. To compare against other control models, we fine-tune $\pi_{0.5}$ and π_0 (Black et al., 2024) on each task using the corresponding high-level instruction as baseline methods. Except for the instruction setting, all fine-tuning configurations are kept identical to those used for DuoMind.

Evaluation. During evaluation, DuoMind and all baseline methods receive the same high-level instruction and observations as input from the simulation environment. We evaluate DuoMind and the baselines over 400 rollouts per task and report the corresponding success rates in table 2 and table 1. Detailed evaluation settings are provided in D.

4.2 Main Results

We evaluate DuoMind on seven tasks in RoboPoly and eight tasks in RoboTwin against strong baselines, with success rates reported in table 1 and table 2, respectively. Detailed task rollout examples are provided in section E. Overall, DuoMind matches or outperforms the baseline methods across most tasks. We attribute these gains to the high-level reasoning and inter-agent communication enabled by the hierarchical design.

For long-horizon multi-robot tasks, a global high-level instruction provides only coarse guidance throughout the entire task execution and offers limited information about what each robot should do at the current stage or how it should adapt to the behavior of its peers. With the hierarchical design, in contrast, the orchestrator reasons over the current task state together with messages from other agents, decomposes the task into appropriate subtasks, and provides specific low-level instructions to the action model. This allows each robot to adapt its behavior as the task evolves while avoiding conflicts with other agents.

Inter-agent communication is particularly important under partial observability, where local observations may not reveal the intentions or future actions of other robots. Explicit message sharing provides complementary information for high-level reasoning, enabling the orchestrator to make better-informed coordination decisions. With more precise low-level instructions, the action model can focus on executing the assigned subtask rather than implicitly reasoning about the overall task progression and multi-agent coordination. As shown in fig. 3, the robots coordinate effectively and complete the task in the proper sequence without conflicts.

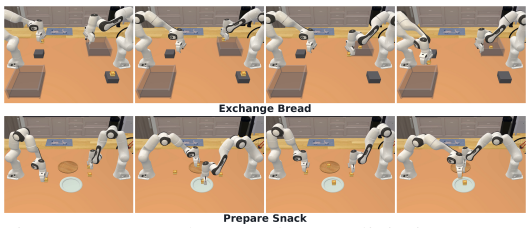
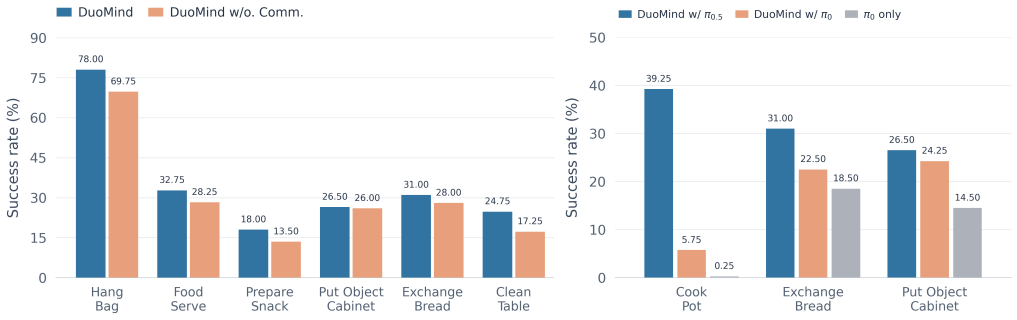


Figure 3: We observe that explicit inter-agent communication enables the robots to better coordinate their behaviors, such as exchanging bread sequentially or placing objects on different plates. By avoiding conflicting or redundant actions in a narrow shared workspace, the robots achieve more efficient and safer coordination.

This distinction becomes especially important when execution deviates from the planned behavior. Policies such as π_0 and $\pi_{0.5}$ that continuously follow a fixed high-level instruction may become uncertain about the appropriate next action after unexpected interactions or execution errors. Such situations occur frequently in multi-robot manipulation, where the behavior of one robot can alter another robot's workspace. The orchestrator can recognize these changes and adaptively issue new instructions, allowing the action model to recover and continue the task.

Additionally, tasks in RoboPoly generally have longer horizons and require more stable coordination than those in RoboTwin, resulting in lower overall success rates. Nevertheless, the performance improvement of DuoMind over the baselines is more significant on RoboPoly, further demonstrating the effectiveness of high-level reasoning and inter-agent communication and suggesting that their benefits become increasingly significant as the task horizon and coordination complexity increase.

4.3 Ablation of Inter-Agent Communication



(a) Inter-agent communication ablation.

(b) Action model ablation.

Figure 4: Ablation studies of DuoMind. (a) Success rates when inter-agent communication is disabled. (b) Success rates with different action models.

Inter-agent communication provides complementary information for decision making and helps mitigate partial observability. To evaluate its effect, we conduct an ablation, following the evaluation setting in section 4.2 except that messages from peer agents are disabled. As shown in fig. 4a, removing inter-agent communication leads to more frequent conflicts and asynchronous behaviors, resulting in degraded coordination performance. These results demonstrate that explicitly sharing peer information is important for effective multi-robot coordination under distributed observations. We further elaborate the ablation study in B.

4.4 Action Model Compatibility

DuoMind decouples high-level orchestration from low-level control, allowing different action models to be integrated into the framework. To study the influence of the action model, we evaluate DuoMind with π_0 in place of $\pi_{0.5}$. As shown in fig. 4b, since π_0 is a weaker action model than $\pi_{0.5}$, DuoMind with π_0 achieves lower success rates than DuoMind with $\pi_{0.5}$, as expected. Nevertheless, it substantially outperforms the π_0-only baseline, demonstrating that the hierarchical orchestration of DuoMind consistently improves multi-robot coordination across different action models. At the same time, the stronger action model provides more reliable low-level execution, which further improves the overall task success rate. We detail this analysis in C.

5 Related Work

5.1 Learning-Based Multi-Agent Coordination

Classical multi-agent learning studies decentralized decision-making under partial observability and non-stationarity through centralized training, value decomposition, and learned communication (Lowe et al., 2017; Rashid et al., 2020; Yu et al., 2022; Foerster et al., 2016). These methods are effective in cooperative games and simulation, but often require task-specific joint training and encode coordination in latent representations that are hard to interpret or integrate with pretrained robot foundation models. Recent work extends coordination to physical robots through decentralized imitation learning, shared representations, and VLA policies (Dong et al., 2025; He et al., 2025; Qin et al., 2025; Doshi et al., 2026). In contrast, DuoMind uses decentralized reasoning with explicit natural-language communication for interpretable exchange of task-relevant information.

5.2 Robot Learning and Multi-Robot Planning

Generalist robot policies leverage large-scale robot data and pretrained vision/language representations to map observations or language instructions to actions (Zitkovich et al., 2023; Kim et al., 2024; Black et al., 2024; Zhou et al., 2026; Team et al., 2025; Gao et al., 2026). Hierarchical approaches further separate high-level semantic reasoning from low-level continuous control (Shi et al., 2025; Intelligence et al., 2025b; Song et al., 2025a; Team et al., 2025), but mainly focus on single-robot execution. LLM- and VLM-based multi-robot systems explore task decomposition, allocation, communication, feedback, and replanning (Kannan et al., 2024; Chen et al., 2024b; Mandi et al., 2024; Liu et al., 2025; Chen et al., 2025b; Wang et al., 2024; He et al., 2026), while typically relying on predefined skills, symbolic actions, waypoints, or motion planners. DuoMind instead combines distributed high-level reasoning and communication with VLA-based physical execution.

5.3 Robot Learning and Multi-Agent Benchmarks

Established robot-learning benchmarks provide diverse manipulation tasks, demonstrations, language instructions, and long-horizon evaluation, but largely focus on single robots or centrally controlled embodiments (Yu et al., 2020; James et al., 2020; Gu et al., 2023; Mees et al., 2022; Liu et al., 2023). Recent multi-agent benchmarks study collaborative planning, heterogeneous capabilities, and manipulation (Mandi et al., 2024; Liu et al., 2025; Chen et al., 2025b; Chang et al., 2025; Qin et al., 2025). However, they often emphasize either abstract high-level planning or low-level manipulation. RoboPoly instead evaluates long-horizon coordination and closed-loop execution under distributed observation and control.

6 Conclusion

We presented DuoMind, a distributed hierarchical framework for multi-robot coordination that combines high-level reasoning, low-level control, and semantic inter-agent communication. We further introduced RoboPoly, a benchmark for evaluating long-horizon multi-robot manipulation under distributed observations and control. Experiments on RoboPoly and RoboTwin demonstrate that hierarchical orchestration improves multi-robot task execution, while ablation studies highlight the importance of explicit information sharing for coordination under partial observability. The decoupled design also allows different reasoning and action models to be integrated without redesigning the overall framework. As stronger VLMs and VLAs continue to emerge, DuoMind can naturally incorporate these advances to further improve its coordination capabilities. Leveraging such advances toward more capable and efficient real-world multi-robot coordination is a promising direction for future work.

7 AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic datasets and implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, assisting with translation, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, interpreting results. Additionally, we used generative AI tools for modifying figures, editing code, and polishing the paper. We have carefully reviewed all AI-assisted work, including checking summarized related work for potential hallucinations and verifying AI-generated code for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

8 REPRODUCIBILITY STATEMENT

We provide detailed descriptions of DuoMind, dataset generation, model training, and evaluation protocols in the main paper and appendix. section A specifies the task settings, while section D reports the fine-tuning and evaluation configurations, including training steps, batch sizes, learning rates, action horizons, and orchestrator intervals. We additionally provide the system prompt used by the orchestrator in section F. Together, these details are intended to facilitate reproduction of the experimental setup and results.

REFERENCES

Abdollah Amirkhani and Amir Hossein Barshooi. Consensus in multi-agent systems: a review. *Artificial Intelligence Review*, 55(5):3897–3935, 2022.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, et al. Partnr: A benchmark for planning and reasoning in embodied multi-agent tasks. In *International Conference on Learning Representations*, volume 2025, pp. 65205–65268, 2025.

Hao Chen, Jiaming Liu, Chenyang Gu, Zhuoyang Liu, Renrui Zhang, Xiaoqi Li, Xiao He, Yandong Guo, Chi-Wing Fu, Shanghang Zhang, et al. Fast-in-slow: A dual-system foundation model unifying fast manipulation within slow reasoning. *arXiv preprint arXiv:2506.01953*, 2025a.

Jinchi Chen, Jie Feng, Weiguo Gao, and Ke Wei. Decentralized natural policy gradient with variance reduction for collaborative multi-agent reinforcement learning. *Journal of Machine Learning Research*, 25(172):1–49, 2024a.

Junting Chen, Checheng Yu, Xunzhe Zhou, Tianqi Xu, Yao Mu, Mengkang Hu, Wenqi Shao, Yikai Wang, Guohao Li, and Lin Shao. Emos: Embodiment-aware heterogeneous multi-robot operating system with llm agents. In *International Conference on Learning Representations*, volume 2025, pp. 56670–56690, 2025b.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025c.

Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4311–4317. IEEE, 2024b.

Dayi Dong, Maulik Bhatt, Seoyeon Choi, and Negar Mehr. Mimic-d: Multi-modal imitation for multi-agent coordination with decentralized diffusion policies. *arXiv preprint arXiv:2509.14159*, 2025.

Ria Doshi, Tian Gao, Annie Chen, Chelsea Finn, and Jeannette Bohg. Chorus: Decentralized multi-embodiment collaboration with one vla policy. *arXiv preprint arXiv:2606.12352*, 2026.

Zhaohan Feng, Ruiqi Xue, Lei Yuan, Yang Yu, Ning Ding, Meiqin Liu, Bingzhao Gao, Jian Sun, Xinhu Zheng, and Gang Wang. Multi-agent embodied ai: Advances and future directions. *Science China Information Sciences*, 69(5):151202, 2026.

Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 29, 2016.

Dechen Gao, Boqi Zhao, Andrew Lee, Ian Chuang, Hanchu Zhou, Hang Wang, Zhe Zhao, Junshan Zhang, and Iman Soltani. Vita: Vision-to-action flow matching policy. In *International Conference on Learning Representations*, volume 2026, pp. 100903–100933, 2026.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023.

Chengyang He, Gadiel Sznaier Camps, Xu Liu, Mac Schwager, and Guillaume Sartoretti. Latent theory of mind: A decentralized diffusion architecture for cooperative manipulation. *arXiv preprint arXiv:2505.09144*, 2025.

Yi-Xiang He, Lan Wei, Haoming Cen, Jian-Jian Jiang, Zhuohao Li, Guanxing Lu, Yihan Yang, Dandan Zhang, and Wei-Shi Zheng. A closed-loop multi-agent framework for robust multi-robot manipulation. *arXiv preprint arXiv:2607.06990*, 2026.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: A VLA That Learns from Experience. *arXiv preprint arXiv:2511.14759*, 2025a.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025b.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

Shyam Sundar Kannan, Vishnunandan LN Venkatesh, and Byung-Cheol Min. Smart-llm: Smart multi-agent robot task planning using large language models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12140–12147. IEEE, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model, 2024. *URL https://arxiv. org/abs/2406.09246*, 1(2):4, 2024.

Huao Li, Hossein N Mahjoub, Behdad Chalaki, Vaishnav Tadiparthi, Kwonjoon Lee, Ehsan Moradi-Pari, Michael Lewis, and Katia Sycara. Language grounded multi-agent reinforcement learning with human-interpretable communication. *Advances in Neural Information Processing Systems*, 37:87908–87933, 2024.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Caelan Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, et al. Hamster: Hierarchical action models for open-world robot manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 24040–24068, 2025.

Fangzhou Liang, Tianshi Zheng, Chunkit Chan, Yauwai Yim, and Yangqiu Song. Llm-hanabi: Evaluating multi-agent gameplays with theory-of-mind and rationale inference in imperfect information collaboration game. *arXiv preprint arXiv:2510.04980*, 2025a.

Jiaxin Liang, Haotian Miao, Kai Li, Jianheng Tan, Xi Wang, Rui Luo, and Yueqiu Jiang. A review of multi-agent reinforcement learning algorithms. *Electronics*, 14(4):820, 2025b.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Kehui Liu, Zixin Tang, Dong Wang, Zhigang Wang, Xuelong Li, and Bin Zhao. Coherent: Collaboration of heterogeneous multi-robot system with large language models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10208–10214. IEEE, 2025.

Yat Long Richie Lo, Biswa Sengupta, Jakob Foerster, and Michael Noukhovitch. Learning multi-agent communication with contrastive learning. In *International Conference on Learning Representations*, volume 2024, pp. 51029–51046, 2024.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Zhao Mandi, Shreeya Jain, and Shuran Song. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 286–299. IEEE, 2024.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Siddharth Nayak, Kenneth Choi, Wenqi Ding, Sydney Dolan, Karthik Gopalakrishnan, and Hamsa Balakrishnan. Scalable multi-agent reinforcement learning through intelligent information aggregation. In *International conference on machine learning*, pp. 25817–25833. PMLR, 2023.

Yiran Qin, Li Kang, Xiufeng Song, Zhenfei Yin, Xiaohong Liu, Xihui Liu, Ruimao Zhang, and Lei Bai. Robofactory: Exploring embodied agent collaboration with compositional constraints. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10075–10085. IEEE, 2025.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

Haoming Song, Delin Qu, Yuanqi Yao, Qizhi Chen, Qi Lv, Yiwen Tang, Modi Shi, Guanghui Ren, Maoqing Yao, Bin Zhao, et al. Hume: Introducing system-2 thinking in visual-language-action model. *arXiv preprint arXiv:2505.21432*, 2025a.

Siqi Song, Xuanbing Xie, Zonglin Li, Yuqiang Li, Shijie Wang, and Biqing Qi. Leveraging adaptive group negotiation for heterogeneous multi-robot collaboration with large language models. *arXiv preprint arXiv:2602.06967*, 2025b.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

Yongdong Wang, Runze Xiao, Jun Younes Louhi Kasahara, Ryosuke Yajima, Keiji Nagatani, Atsushi Yamashita, and Hajime Asama. Dart-llm: Dependency-aware multi-robot task decomposition and execution using large language models. *arXiv preprint arXiv:2411.09022*, 2024.

Annie Wong, Thomas Bäck, Anna V Kononova, and Aske Plaat. Deep multiagent reinforcement learning: challenges and directions: A. wong et al. *Artificial Intelligence Review*, 56(6):5023–5056, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in neural information processing systems*, 35:24611–24624, 2022.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713. IEEE, 2025.

Hanchu Zhou, Brendan Lynch, Raman Goyal, Dechen Gao, Begum Kasap, Boqi Zhao, and Junshan Zhang. Agile-wam: An agile tactile world action model for contact-rich robot control. *arXiv preprint arXiv:2609.20761*, 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on robot learning*, pp. 2165–2183. PMLR, 2023.

Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, et al. Latent collaboration in multi-agent systems. *arXiv preprint arXiv:2511.20639*, 2025.

A Task Settings of RoboPoly

RoboPoly contains the following seven tasks:

Food Serve. Two pieces of bread and a serving tray are placed across the workspace such that one robot cannot reach both the bread and the tray. The robots must transfer the bread between each other when necessary and place both pieces onto the tray.

Prepare Snack. Two plates are placed near the center of the table, while four pieces of bread are distributed across the two sides of the workspace. The robots must place two pieces on each plate. Both robots can operate simultaneously and must coordinate their motions to efficiently complete the task while avoiding collisions.

Clean Table. Two cubes and a container are placed on the table such that neither robot can independently reach all cubes and container. Completing the task requires a cube to be passed between the robots when the container lies outside the workspace of a robot.

Cook Pot. A carrot, a pot with a lid, and an oven are placed around the center of the workspace. The robots must first remove the lid, place the carrot into the pot, and replace the lid. They should then jointly grasp the two handles of the pot and transport it to the oven.

Put Object Cabinet. A cube and a cabinet are placed in the workspace. The robots must open the bottom drawer of the cabinet, place the cube inside, and close the drawer. The task requires coordination between two robot arms for higher efficiency.

Hang Bag. A bag and a hook are placed on opposite sides of the workspace such that neither robot can reach both. One robot must first pass the bag to the other robot, which then hangs the bag onto the hook.

Exchange Bread. A piece of bread and a target box are placed beside each robot arm. Each robot must place its own bread into the box on the opposite side of the table. Since the target box lies outside its reachable workspace, the robot must first hand the bread over to the other robot, which then places it into the corresponding box.

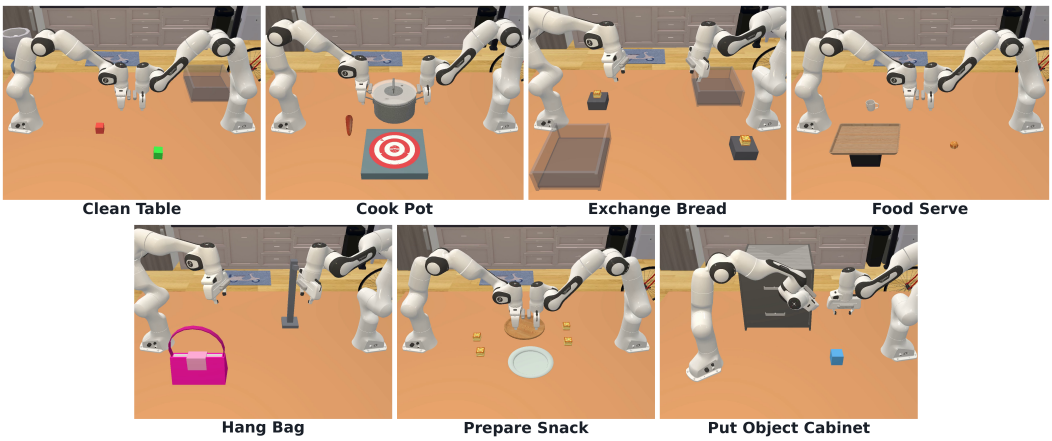


Figure 5: The task overview of RoboPoly.

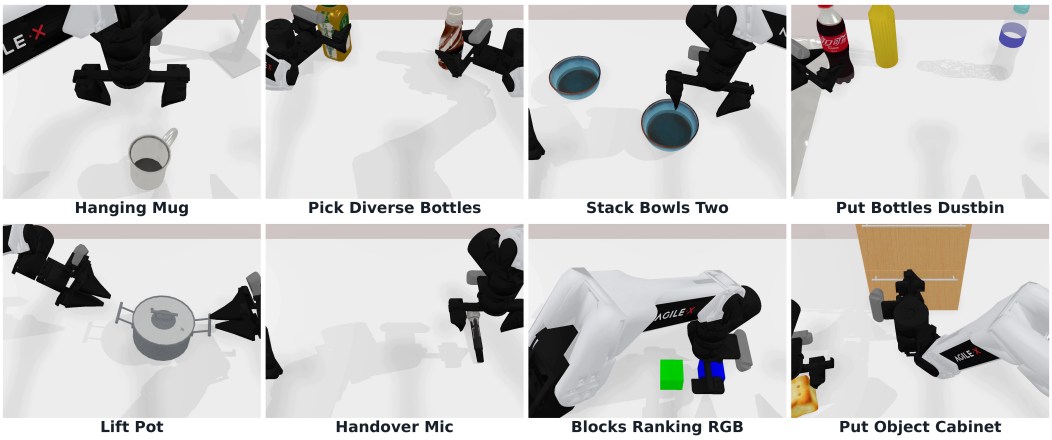


Figure 6: The task overview of selected RoboTwin tasks.

B Effect of Inter-Agent Communication

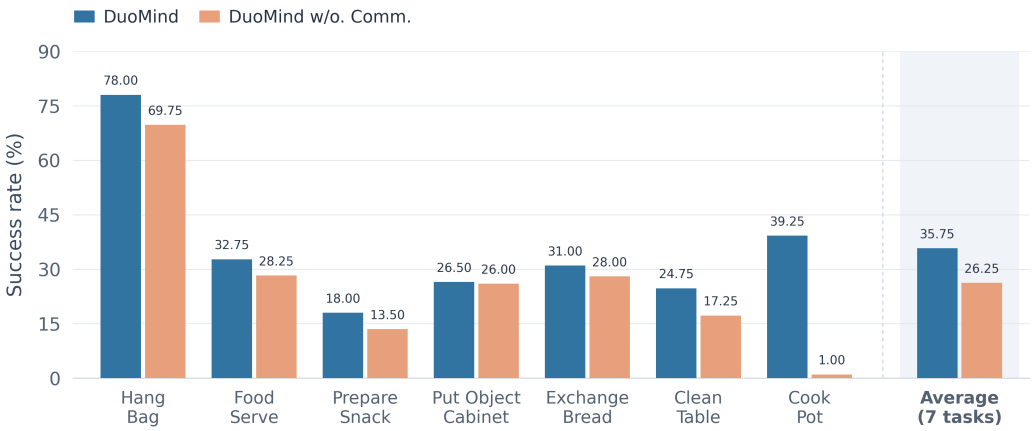


Figure 7: The result of ablation study that disable the inter-agent communication in DuoMind.

Inter-agent communication shares complementary information for decision making and helps mitigate the partial observability inherent in distributed multi-robot settings. To further study the effect of the shared messages, we conduct an ablation using DuoMind with $\pi_{0.5}$ as the action model, following the evaluation setting in section 4.2 except that messages from peer agents are disabled. Under this setting, each orchestrator receives no information from the other agent during reasoning and generates low-level instructions based only on its local observation and the task instruction.

The results are reported in fig. 7. Without inter-agent communication, we observe more frequent conflicts and asynchronous behaviors between the two robots. For example, in *Cook Pot*, the two robots are required to lift the pot handles in a coordinated manner. Without communication, one robot often starts lifting while the other has only begun grasping the handle, causing the coordinated manipulation to fail. In *Prepare Snack*, the two robots may independently select the same plate as their target and attempt to place bread simultaneously, which can lead to collisions between the robot arms above the plate.

These failure cases illustrate a limitation of relying solely on local observations: an individual agent may not have sufficient information to infer the peer agent's current intention or upcoming behavior. Explicit inter-agent communication provides a direct and efficient mechanism for sharing such information, thereby reducing conflicts and improving synchronized and safe multi-robot coordination.

C Action Model Compatibility

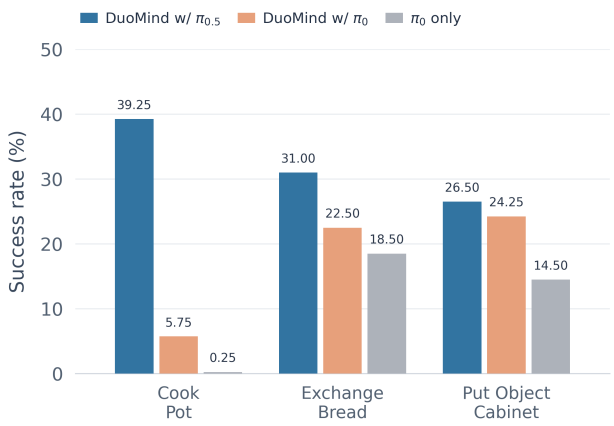


Figure 8: Results of the ablation study using different action models in DuoMind.

DuoMind decouples the high-level orchestrator from the low-level action model, enabling different action models to be integrated without changing the overall coordination framework. To analyze the influence of the action model, we conduct an additional experiment using π_0 in place of $\pi_{0.5}$, while following the same fine-tuning procedure described in section 4.2.

As shown in fig. 8, because π_0 provides weaker low-level control capability than $\pi_{0.5}$, DuoMind with π_0 achieves lower success rates than DuoMind with $\pi_{0.5}$, as expected. However, DuoMind with π_0 still substantially outperforms the π_0-only baseline. This comparison shows that the performance gain is not solely determined by the underlying action model. By introducing high-level reasoning and subtask decomposition through the orchestrator, DuoMind provides more appropriate low-level instructions for the evolving multi-agent state and improves coordination over the action-model-only setting.

The comparison also highlights the complementary roles of the two components. The orchestrator improves high-level reasoning and multi-agent coordination, while a stronger action model provides more reliable and stable low-level execution. Consequently, improving the capability of the action model can further increase the overall success rate. These results demonstrate that DuoMind is compatible with different action models while continuing to benefit from advances in low-level VLA policies.

D Training and Evaluation Configurations

The fine-tuning and evaluation settings of RoboPoly and RoboTwin are summarized in tables 3 and 4. All methods use LoRA fine-tuning and 400 evaluation rollouts per experiment. The batch sizes reported in the table refer to global batch size. For RoboTwin, the policy takes the head camera and each robot's own wrist image as input, while on RoboPoly it takes the global camera and each robot's own wrist image. The fine-tuning and evaluation of $\pi_{0.5}$ and π_0 adopt the same settings.

In the tables, *Execute / horizon* denotes the number of actions executed from each predicted action chunk and the total action-chunk horizon, respectively. *VLM interval* denotes the number of policy chunks executed between two consecutive calls to the high-level VLM orchestrator. We use a linear warmup over the first 1,000 training steps to reach the peak learning rate, followed by cosine decay to a final learning rate of 2.5×10^{-6}.

Table 3: Fine-tuning and evaluation settings on RoboPoly.

Task	Method	Train steps	Batch	Peak LR	Execute / horizon	VLM interval
Hang Bag	$\pi_{0.5}/\pi_0$ only	10k	256	2.5×10^{-5}	12/16	–
Hang Bag	DuoMind	10k	256	2.5×10^{-5}	12/16	4
Food Serve	$\pi_{0.5}/\pi_0$ only	10k	256	2.5×10^{-5}	16/16	–
Food Serve	DuoMind	10k	256	2.5×10^{-5}	16/16	4
Prepare Snack	$\pi_{0.5}/\pi_0$ only	2k	512	1×10^{-4}	16/16	–
Prepare Snack	DuoMind	2k	512	1×10^{-4}	16/16	4
Put Object Cabinet	$\pi_{0.5}/\pi_0$ only	4k	512	1×10^{-4}	8/16	–
Put Object Cabinet	DuoMind	4k	512	1×10^{-4}	8/16	4
Exchange Bread	$\pi_{0.5}/\pi_0$ only	4k	512	1×10^{-4}	16/16	–
Exchange Bread	DuoMind	4k	512	1×10^{-4}	16/16	3
Clean Table	$\pi_{0.5}/\pi_0$ only	4k	512	1×10^{-4}	16/16	–
Clean Table	DuoMind	4k	512	1×10^{-4}	16/16	4
Cook Pot	$\pi_{0.5}/\pi_0$ only	4k	512	1×10^{-4}	16/16	–
Cook Pot	DuoMind	4k	512	1×10^{-4}	16/16	4

Table 4: Fine-tuning and evaluation settings on RoboTwin.

Task	Method	Train steps	Batch	Peak LR	Execute / horizon	VLM interval
Hanging Mug	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Hanging Mug	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Pick Diverse Bottles	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Pick Diverse Bottles	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Stack Bowls Two	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Stack Bowls Two	DuoMind	20k	32	2.5×10^{-5}	50/50	2
Put Bottles Dustbin	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Put Bottles Dustbin	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Lift Pot	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Lift Pot	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Handover Mic	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Handover Mic	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Blocks Ranking RGB	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Blocks Ranking RGB	DuoMind	20k	32	2.5×10^{-5}	50/50	4
Put Object Cabinet	$\pi_{0.5}/\pi_0$ only	20k	32	2.5×10^{-5}	50/50	–
Put Object Cabinet	DuoMind	20k	32	2.5×10^{-5}	50/50	4

E EVALUATION VISUALIZATION

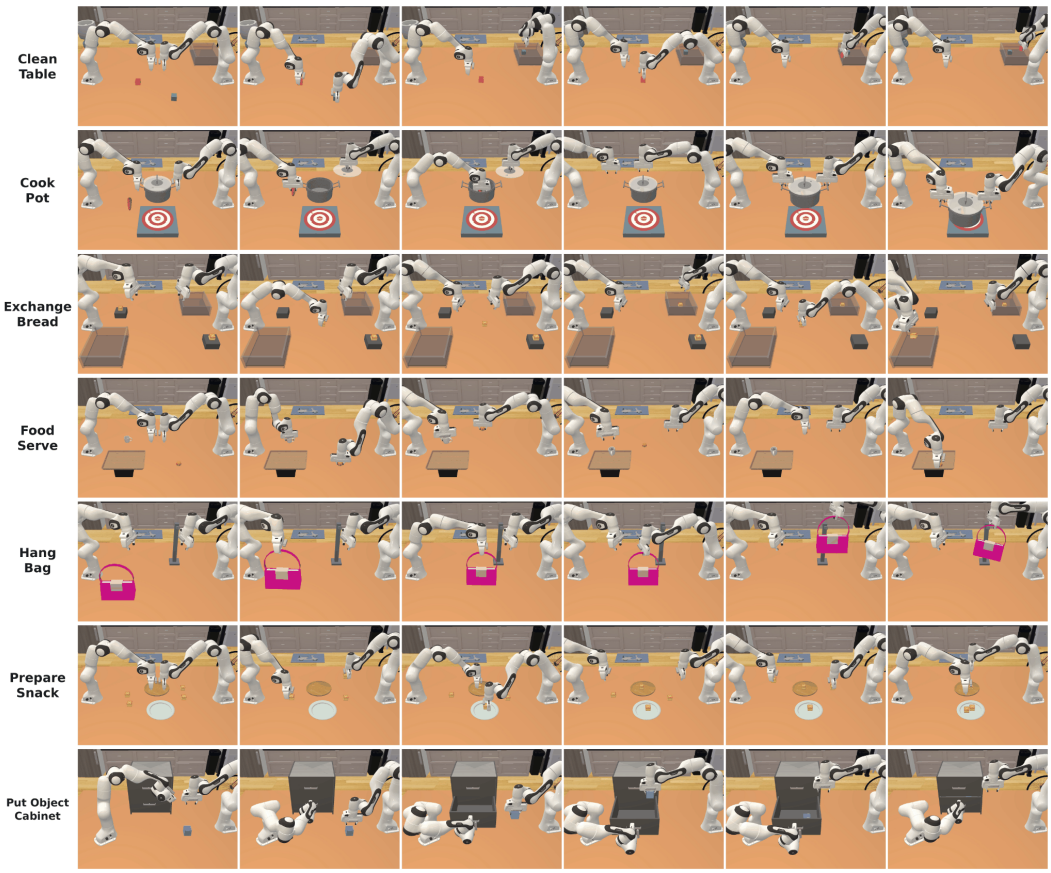


Figure 9: We evaluate DuoMind on 7 RoboPoly tasks.

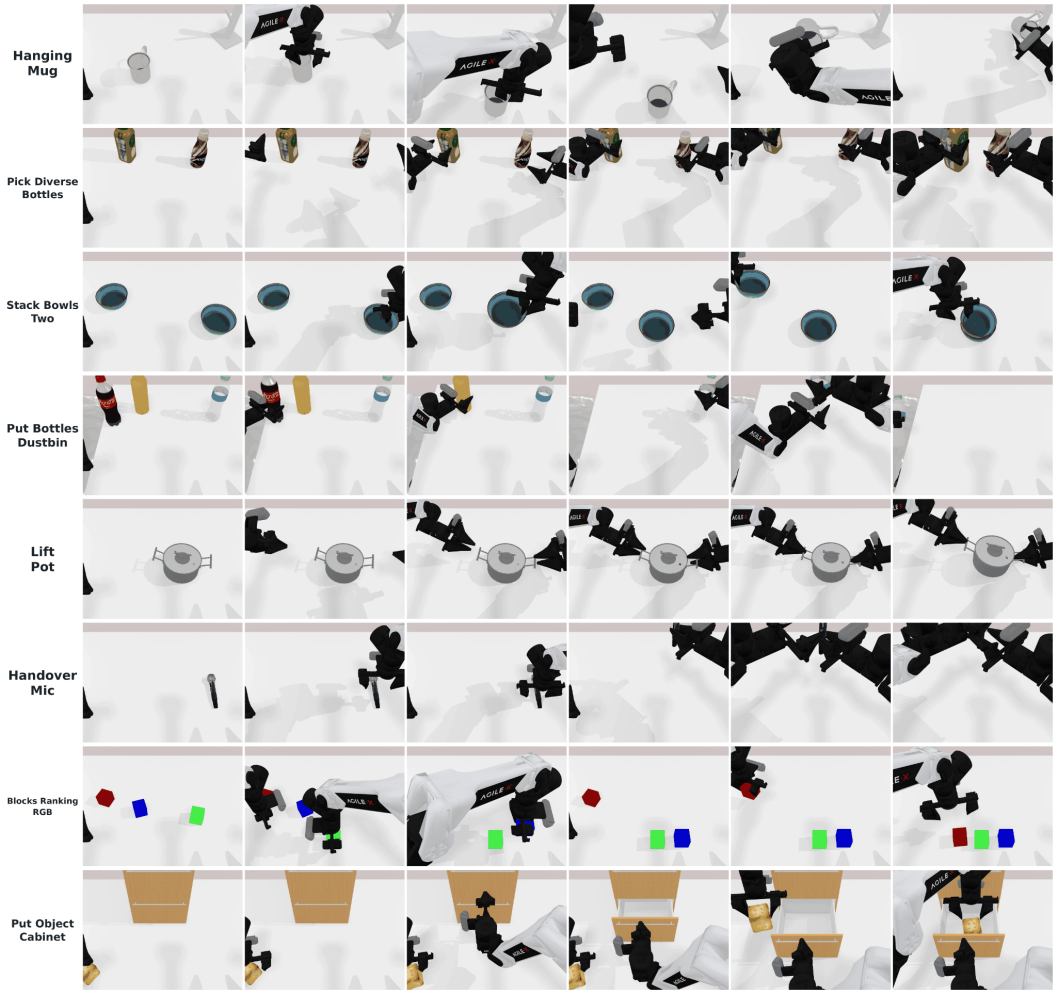


Figure 10: We evaluate DuoMind on 8 selected RoboTwin tasks.

F VLM System Prompt

We guide the orchestrator with a general system prompt shared across agents and tasks, which specifies its role and the type of low-level instruction to generate for the action model. The orchestrator serves two roles. First, it performs high-level planning by reasoning over the ego agent's reachable workspace, the peer agent's intention, and the current task state. It selects a complementary subgoal that avoids conflicts with other agents and contributes effectively to the overall task. Second, it generates a structured message for inter-agent communication, providing concise information about the ego agent's intention, belief, and uncertainty to support the peer agent's reasoning and coordination.

System Prompt for Orchestrator

```
You are the system-2 planner for agent_<AGENT_ID>, controlling the <ARM_SIDE> arm
(panda-<AGENT_ID>).
Use the global camera image, the high-level task, and the peer message to choose one short
low-level instruction for this agent's VLA.

You receive the current global-camera image.
High-level instruction: <HIGH_LEVEL_INSTRUCTION>
This agent's currently accepted instruction: <PREVIOUS_ACCEPTED_INSTRUCTION>
This agent's own recent planning history (oldest to newest): <OWN_RECENT_HISTORY_JSON>
Other agent shared message: {"intention": "<PEER_INTENTION>", "subgoals":
"<PEER_SUBGOALS>", "belief": "<PEER_BELIEF>"}

Without a specific requirement to act on a particular object or region, prefer objects and
subgoals in this agent's reachable workspace and choose a target that complements the
peer.

The low-level instruction must look like these training instructions:
<ALL_TRAINING_INSTRUCTION_EXAMPLES>

This ManiSkill task supplies an arm-specific candidate set. The L1/R1-style prefixes are
metadata tags, not part of the low-level instruction. Select exactly one candidate, but
put only the sentence after its colon in low_level_instruction; do not invent, paraphrase,
include the tag, or select another arm's instruction.
Permitted candidates for this <ARM_SIDE> arm:
<THIS_ARM_CANDIDATES>

The numeric candidate suffixes are ordered stages for this arm. On the first planning
event, select the lowest-numbered stage. On later events, repeat the current stage until
its result is visibly complete, then advance by at most one stage. Never skip or return to
an earlier stage.

Progress rule: do not keep issuing a subgoal that is already complete. Review the current
image and your own history when available. If visual evidence or your own prior belief
says the current subgoal's observable result is complete, you MUST select a different
permitted candidate that advances the unfinished high-level task. Otherwise repeat the
current candidate. Do not advance merely because time passed, and visually verify claims
from the other agent.

Return only valid JSON with this exact schema:
{
  "low_level_instruction": "short command for this agent VLA",
  "shared_message": {
    "intention": "same as low_level_instruction",
    "subgoals": "observable condition when the low-level instruction is done",
    "belief": "task-relevant CURRENT observed state; never describe an intended future
    result as already complete"
  }
}
```