Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

A Closed-Loop Multi-Agent Framework for Robust Multi-Robot Manipulation

Yi-Xiang He[1], **Lan Wei**[1], **Haoming Cen**[1], **Jian-Jian Jiang**[1], **Zhuohao Li**[1,5], **Guanxing Lu**[4], **Yihan Yang**[1], **Dandan Zhang**[3], **Wei-Shi Zheng**[1,2,5,†]
[1]Sun Yat-sen University, [2]Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, China
[3]Imperial College London, [4]Tsinghua Shenzhen International Graduate School, [5]Shenzhen Loop Area Institute
[†]Corresponding author.

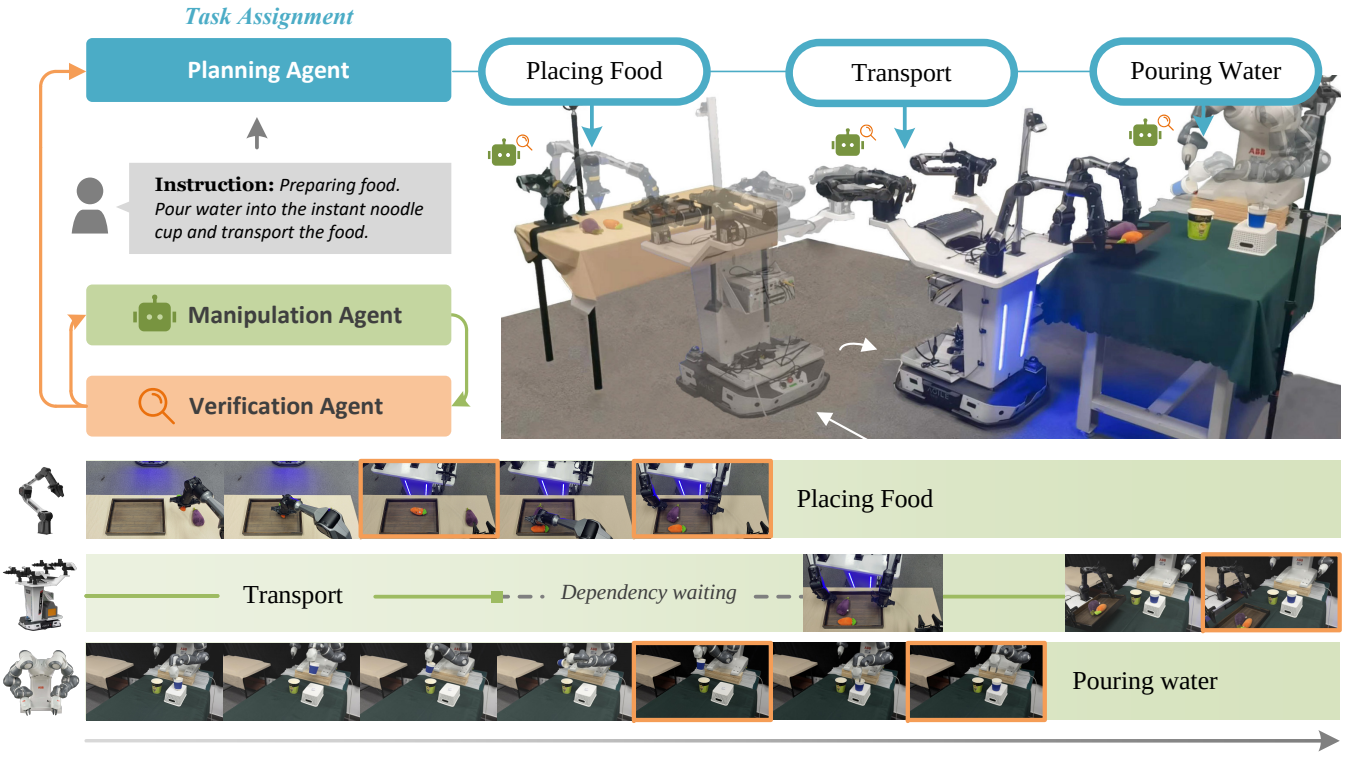

Fig. 1: We propose a closed-loop multi-agent framework for robust multi-robot manipulation. The system is driven by three specialized agents: **the Planning Agent** decomposes instructions into dependency-aware sub-tasks, **the Manipulation Agent** grounds these sub-tasks into physical actions through adaptive tool use, and **the Verification Agent** identifies execution errors to trigger hierarchical re-planning. Extensive real-world experiments demonstrate that our framework achieves versatile manipulation capabilities, effectively handles external disturbances and execution uncertainties, and adapts from single-robot operations to multi-robot collaborative tasks.

***Abstract*—Multi-robot systems provide the parallelism and redundancy necessary for long-horizon tasks, while Large Language Models (LLMs) offer the reasoning capabilities to decompose these objectives into actionable plans. However, effectively grounding this high-level reasoning in physical multi-robot execution remains an open challenge. Existing LLM-based approaches fall mainly into two categories: Single-robot methods achieve robust contact-rich manipulation but lack the coordination mechanisms required for tasks spanning multiple workspaces. Current multi-robot frameworks focus on high-level planning, often treating manipulation as an idealized primitive that fails to account for real-world execution uncertainties. To address this, we propose a hierarchical closed-loop agentic LLM-based framework to ensure robust multi-robot manipulation. Our system consists of three specialized agents: the Planning Agent decomposes instructions into allocated sub-tasks, the Manipulation Agent for each robot executes actions via adaptive tool use, and the Verification Agent closes the loop by monitoring physical outcomes and feeding back semantic corrections. Extensive real-world experiments demonstrate that our framework achieves superior success rates, ensures robust adaptability ranging from single to cross workspace manipulation, and offers a generalizable approach for diverse manipulation tasks.**

I. Introduction

Multi-robot systems demonstrate significant potential in diverse environments, from domestic service to industrial manufacturing [2, 7, 30], by expanding manipulation capabilities and enhancing efficiency to offer great system extensibility. However, achieving such systems imposes a substantial burden

on manipulation complexity, necessitating coordination between heterogeneous robots and effective emergency response to unexpected failures during execution.

Recently, the integration of Large Language Models (LLMs) has offered new avenues for robotic manipulation by leveraging their capabilities in semantic understanding and logical reasoning [52, 9, 1]. Existing LLM-based approaches fall broadly into two categories. The first uses LLMs to synthesize single-robot manipulation policies, code, grasping guidance, or constraints [32, 15, 18, 46, 48, 33]. While these methods demonstrate promising physical grounding, they lack the coordination mechanisms required for tasks that span multiple workspaces. The second paradigm targets multi-robot systems but focuses primarily on high-level task allocation and symbolic planning [51, 50, 29, 24, 35], typically validating plans prior to execution while treating manipulation as an idealized primitive. This abstraction assumes deterministic success and neglects real-world execution uncertainties such as grasp failures, contact variability, external disturbances, or unreachable workspace.

To achieve robust and self-correction capabilities in real-world tasks, we propose a closed-loop agentic workflow for robust multi-robot manipulation. Our core insight is that tackling real-world uncertainties requires a shift from static execution pipelines to an iterative agentic process that incorporates a hierarchical recovery mechanism. Unlike previous methods that restrict validation to the pre-execution phase and subsequently rely on idealized actions, an agentic workflow empowers the system to actively reason, act, and reflect on its performance. By integrating semantic planning, physical grounding, and outcome verification, our architecture enables robots to perceive environmental feedback and adaptively recover from failures, achieving robust and verifiable multi-robot manipulation and ensuring reliability by allowing agents to actively address execution uncertainties in real-world environments.

Specifically, we instantiate this workflow with three specialized agents that operate in a closed-loop cycle of reasoning, acting, and reflecting, enabling refinement of both high-level collaboration decisions and low-level execution. The information flow begins with the Planning Agent, which serves as the reasoning core to handle task decomposition and allocation while balancing semantic instruction logic with the specific capabilities of heterogeneous robots. To ensure accurate semantic grounding, it employs a bidirectional interaction mechanism to query bottom-up perception from the Manipulation Agent whenever instructions are ambiguous. Following this, the Manipulation Agent takes charge of the acting phase. It translates sub-tasks into fine-grained operations by adaptively leveraging tools such as keypoint localization and grasp generation models to ground semantic plans into reality. Finally, to tackle inevitable execution uncertainties, the verification functions as a reflecting mechanism. It monitors the status of each sub-operation, and upon detecting a failure, it utilizes historical interaction data and current observations to diagnose the cause of error and trigger hierarchical recovery strategies. For execution-level errors, it initiates local re-planning, whereas for capability constraints, it escalates to global re-planning to seek collaboration from other robots. This multi-level mechanism effectively resolves errors at the appropriate layer and prevents failures from disrupting the global multi-robot workflow. Together, these agents form a collaborative system that ensures reliable deployment for complex multi-robot manipulation.

Extensive real-world experiments on six tasks validate our framework in three aspects: (1) **Versatility:** outperforming both learning-based and LLM-based baselines in diverse skills ranging from pick-and-place to articulated object interactions; (2) **Robustness:** evidenced by a multi-level recovery mechanism that handles physical disturbances via local re-planning and resolves capability constraints through inter-robot collaborative recovery; (3) **Adaptability:** demonstrated by the seamless transition by our framework from tabletop setups to cross-workspace collaborations involving heterogeneous robots.

II. Related works

A. LLMs for Robotics

While conventional robotic manipulation models [8, 54, 3, 25, 58, 59, 22, 47] demonstrate promising performance, their understanding of tasks and environments remains constrained. To address this limitation, Large Language Models (LLMs) are increasingly integrated into robotic systems to leverage their strengths in semantic reasoning and high-level planning [28, 4, 26]. Early works, such as [20, 14, 16, 19], demonstrate the efficacy of LLMs in decomposing semantic instructions into sequential sub-tasks. Leveraging the robust code generation capabilities of LLMs [37, 11, 23], a specific stream of research focuses on synthesizing executable policies directly. [32, 42] utilize LLMs to generate Pythonic control logic that invokes predefined action primitives. To improve physical grounding, methods like [17, 18, 44, 40] formulate manipulation as constraint optimization problems, using the geometric information of the scene, translating natural language into cost maps or spatial constraints to guide motion planning. While these approaches achieve promising manipulation capabilities in single-robot settings, they are inherently limited to individual workspaces. They lack the requisite coordination mechanisms to manage spatio-temporal constraints across multiple robots, making them insufficient for collaborative tasks that demand synchronized execution and long-horizon cooperative behaviors.

In contrast to these single-robot-centric approaches, our framework extends LLMs manipulation paradigm to the multi-robot domain. We introduce a collaborative planner through an agentic workflow that enables heterogeneous robots to achieve parallel execution and cross-workspace collaboration.

B. Multi-Robot Planning and Collaboration

Prior to the advent of LLMs, multi-robot planning focuses on classical symbolic planning and optimization methods.

Traditional multi-robot Task and Motion Planning frameworks typically utilize Planning Domain Description Language (PDDL) or temporal logic to solve long-horizon collaboration problems under geometric constraints [10, 13, 5]. While these methods ensure logical correctness and collision-free execution, they rely on rigid, predefined domain rules and struggle to generalize to open-ended semantic instructions.

Recognizing the limitations of traditional approaches in semantic understanding and adaptability, recent research has extended LLMs capabilities to multi-robot systems [41, 57, 6, 43, 36, 39, 53]. A prominent line of work treats multi-robot collaboration as a communication and consensus problem mimicking human teams. For instance, Mandi et al. [38], Zhang et al. [56] and Zhang et al. [55] introduce multi-turn dialog mechanisms where robots discuss strategies to reach a consensus on task distribution. Other frameworks explore centralized paradigms. Kannan et al. [24] utilizes a single centralized LLM to decompose instructions into multi-stage task planning. In a more structured approach, Liu et al. [35] adopts a centralized hierarchical pipeline that generates sub-task proposals, allocating specific actions to heterogeneous robots based on their capabilities.

Despite their success in high-level planning, most existing LLM-based frameworks limit their validation before execution, which implicitly assume that the scheduled tasks will be executed successfully. They tend to treat the interaction process as idealized abstract actions, neglecting the execution uncertainties[34, 31] inherent in the physical world. Consequently, these planners lack the requisite dynamic verification and feedback mechanisms during deployment, often leading to cascading system-wide failures when local errors occur.

In contrast to the aforementioned approaches, our work bridges the gap between single-robot manipulation capabilities and multi-robot collaboration. We propose a closed-loop multi-agent architecture that bridges low-level physical manipulation with high-level collaborative task planning. By incorporating a Verification Agent, our framework moves beyond open-loop execution, enabling active outcome monitoring and adaptive failure recovery to ensure robust real-world collaboration.

III. Method

A. System Overview

In this section, we present a closed-loop multi-agent framework designed for reliable multi-robot manipulation in real-world environments. As shown in Fig. 2, our framework is structured as a collaborative agentic workflow. It takes language instructions and a coarse-grained scene description, which include the robot's position and the general location of objects as input. It integrates semantic planning, physical manipulation, and verification into an iterative cycle coordinated by three specialized agents: the Planning Agent handles high-level reasoning and task decomposition, the Manipulation Agent translates the sub-task into the real world and the Verification Agent assesses action feasibility and execution outcomes. Crucially, these agents dynamically interact to translate abstract instructions into real-world primitives, while robustly adapting to execution-time uncertainties. Detailed designs of each agent and their collaborative mechanisms will be elaborated in the following sections. Specific model selections are detailed in Appendix.III.

B. Planning Agent: Semantic Reasoning and Task Allocation

The Planning Agent serves as the semantic reasoning core of the framework. As shown in Fig. 2(a), its primary role is to interpret user instructions and generate a high-level task graph that bridges the gap between abstract semantic goals and robot-specific capabilities through two key mechanisms.

Dependency-Aware Task Decomposition. Given a natural language instruction and coarse-grained scene descriptors, the Planning Agent decomposes the global objective into a **directed acyclic graph** (DAG) of logical sub-tasks. Unlike generating action sequences, this agent focuses on identifying task dependencies, determining which sub-tasks must be executed sequentially and which can be performed simultaneously. It outputs a schedule of sub-tasks augmented with a `parallel` flag, allowing robots to perform operations in parallel during this stage which enables concurrent execution by heterogeneous robots to optimize operational efficiency.

Capability-Based Allocation & Interactive Refinement. The Planning Agent assigns each sub-task to the most suitable robot by matching the sub-task requirements (e.g., transport) with the heterogeneous robots' capabilities (e.g., mobile base and fixed base). Furthermore, to handle potential uncertainties, the agent is equipped with an interactive refinement capability. In scenarios where the initial instruction or scene information is insufficient (e.g., an ambiguous "Tidy up" command or the object to manipulate is unknown), the Planning Agent can conditionally send a `further_perception` flag. This triggers a targeted query to the corresponding Manipulation Agent to obtain fine-grained observations, allowing the Planning Agent to refine the task graph based on the updated semantic context.

C. Manipulation Agent: Physical Action Grounding

The Manipulation Agent functions as the embodied executor within our agentic workflow. As shown in Fig. 2(b), upon receiving a logical sub-task from the Planning Agent, its objective is to translate this semantic instruction into action primitives that can control the robot directly. To achieve this, the agent leverages a flexible mechanism of adaptive tool invocation. Rather than following a fixed pipeline, it dynamically calls upon a suite of specialized modules, ranging from VLM-based decomposition to visual prompting tools to decompose a sub-operation, perceive targets, and get action parameters for the specific physical context. Specifically, the agent re-perceives the environment between sub-operations in response to potential changes occurring after an action (e.g., opening a drawer).

Role-Based Operation Resolution. Unlike Planning Agent that decomposes tasks at a high-level, Manipulation Agent focuses on breaking down sub-tasks into executable fine-grained actions. Before execution, the agent first divides sub-task into manipulation steps through VLM-based reasoning.

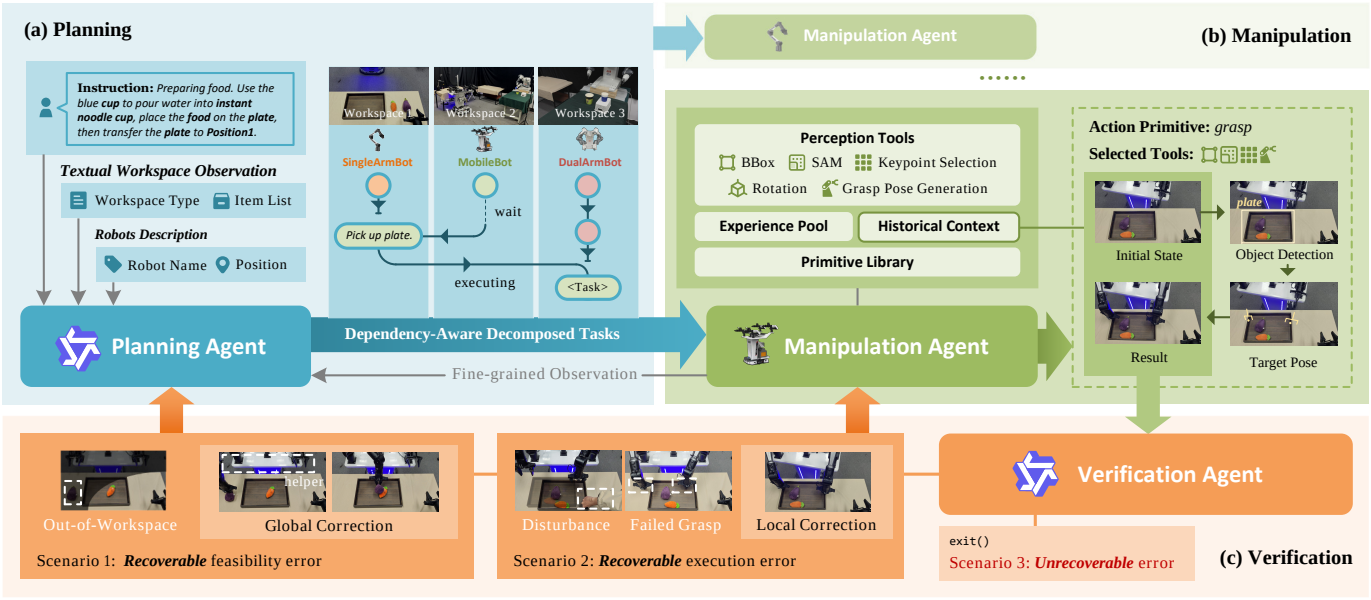

Fig. 2: **Overview of our Closed-Loop Multi-Agent Framework.** The agentic LLM-based workflow integrates three specialized agents: (a) **The Planning Agent** processes language instructions and coarse-grained workspace observations. It reasons about robot capabilities to generate a dependency-aware task graph, allocating sub-tasks to different robots. (b) **The Manipulation Agent** grounds these abstract sub-tasks into grounded action primitives using a suite of visual perception tools and historical context. (C) **The Verification Agent** monitors the physical state to ensure robustness. It implements a hierarchical recovery, handling execution failures through local correction and capabilities violations through global correction. Ensuring robust collaboration in real-world environments.

Based on the interaction logic, it not only categorizes entities into active and passive roles but also determines their semantic part descriptions (e.g., "handle", "surface") for each object. Simultaneously, the agent evaluates object attributes to determine whether a dual-arm action is required at this stage. This structured resolution ensures that subsequent tools focus on the correct entities.

Tool-Augmented Visual Perception. To bridge the gap between semantic roles and action parameters, the agent executes a comprehensive visual grounding pipeline utilizing the part descriptions extracted in the previous stage. *Object Detection & Segmentation:* The agent first invokes a VLM-based detector to identify bounding boxes for active and passive objects, followed by the Segment Anything Model (SAM) [27] to generate precise pixel-level masks. *Affordance-Aware Keypoint Selection:* The agent employs a visual prompting tool [44] to localize optimal interaction points, such as the upper part of a cup handle. For dual-arm grasping, the agent applies a geometric heuristic to identify the key points for center symmetry of objects to ensure stability. Crucially, these 2D keypoints are projected into 3D space using depth information, establishing the spatial anchors required for manipulation.

Action Primitives Generation with Adaptive Tool Invocation. Once the spatial keypoints are established, the agent synthesizes the operation by querying an action primitive library containing skills in Tab I. During this generation process, the agent adaptively invokes specialized tools if a primitive requires physical grounding. For instance, when instantiating a grasp primitive, semantic keypoints alone may be insufficient for physical stability. Current 6-DoF grasp generation methods

TABLE I: **Library of Action Primitives.** Modes indicate support for Single-arm (S) or Dual-arm (D) execution.

Primitive	Mode	Function
Basic Manipulation		
`Grasp`	S / D	Execute a grasp and lift up the object.
`Place`	S / D	Move to the target, put down and release gripper.
`Lift Up`	S / D	Vertical motion ($+\Delta z$).
`Put Down`	S / D	Vertical motion ($-\Delta z$).
`Reset Home`	S / D	Retreat to a predefined safe configuration.
Spatial Adjustment		
`Move XY`	S / D	Planar translation while maintaining height.
`Move Pose`	S	Move to (x, y, z) keeping current pose.
`Rotate`	S	Re-orient the object around the predicted axis.
`Align`	S	Minimize the distance between semantic keypoints.
Constraint Interaction		
`Open`	S	Follow the articulation trajectory (prismatic).
`Close`	S	Restore the articulated object to a closed state.

are diverse [45, 21, 12, 49], the agent triggers AnyGrasp [12] to generate candidate 6-DoF grasping poses (details are provided in Appendix.II.C), which are then matched against the extracted 3D semantic keypoints to select the optimal pose that minimizes spatial alignment error. In addition to spatial precision provided by the keypoints selection, many manipulation tasks require appropriate object orientation to satisfy specific semantic requirements. To addressing this, the agent can activate the rotation perception tool. Leveraging rigid-body assumption, this tool projects the robot's end-effector frame onto the object's center and a VLM then performs spatial reasoning to analyze the object's current state and the task requirements, predicting the necessary rotation axis, direction, and angle. This allows the agent to execute intuitive reorientation actions.

Memory-Augmented Reasoning. To enhance the manipulation efficiency and reduce the ambiguity in sub-tasks description, we implemented a dual-layer memory mechanism. First, for the short-term memory, the agent maintains an interaction history of previous manipulation steps. It will be used in the sub-task decomposition phase which resolves ambiguous sub-task instructions and verification phase. For example, when receiving "Pour water", it can determine from historical manipulations whether the grasping action has already been performed or if only the pouring action needs to be executed. Second, for manipulation efficiency, we implement an experience pool for the sub-task decomposition and action primitives generation. For a sub-task, the agent first identifies the actions required for the task and the objects involved in the interaction and records them as a template signature (e.g., `<Action> Active` $\rightarrow$ `Passive`). After this sub-task is successfully executed, the agent will record the results of sub-task decomposition and the corresponding action primitives. Next time the agent encounters a sub-task exactly matching the signature, it triggers a shortcut, using the validated sub-task decomposition and the primitives directly. The experience pool reduces the number of VLM API calls which accelerates execution time, enabling manipulation to proceed directly into the perception phase.

D. Verification Agent: Local Monitoring and Hierarchical Recovery

The Verification Agent acts as the reflection component in the agentic workflow to ensure system reliability. As shown in Fig. 2(c), it transforms the linear execution into a robust closed-loop system by monitoring the task states and leveraging a hierarchical recovery mechanism. Each Manipulation Agent is equipped with a Verification Agent which obtains the execution status and triggers different strategies based on the types of failures: executing a local self-correction when execution fails, while interacting with the Planning Agent for global re-planning when encountering feasibility errors.

Discrete Visual Outcome Validation. To verify each sub-operation acted by the Manipulation Agent, Verification Agent employs a visual feedback tool to perform a discrete and semantic assessment. Upon the completion of a sub-operation, the agent analyzes pre-action and post-action images relative to the instructions, performing logic validation to generate a binary success prediction.

Hierarchical Error Recovery. When a failure is detected, the Verification Agent diagnoses the error type and triggers recovery at the appropriate system level. For execution failures, where the task remains feasible (e.g., a grasping slip or misalignment), the agent initiates a local recovery loop without disturbing the global plan. By referencing the interaction history and the visual states, the agent autonomously constructs a corrective sequence to re-establish the necessary states. These recovery steps are dispatched directly to the Manipulation Agent, forming an efficient inner loop for rapid self-correction.

Conversely, if an action is identified as physically infeasible, typically due to workspace constraints or missing objects , the Verification Agent will send the failure to the Planning Agent, triggering a global recovery loop. It will submit a failure feedback containing the specific error context and state, which activates re-allocation rules in the Planning Agent. Consequently, the Planning Agent generates a new task graph based on the cause of the error and the current state of robots, while continuing to pursue the original task objectives. For example, when a fixed-based robot arm can't pick up an object out of its working space, Planning Agent will dispatch a mobile robot to solve the problem. This hierarchical mechanism ensures task continuity through agentic collaboration, preventing single-robot failures from stopping the entire workflow.

IV. EXPERIMENTS

In this section, we evaluate the effectiveness of our proposed framework through extensive real-world experiments. We designed 6 distinct tasks of varying complexity, ranging from single-robot manipulation to heterogeneous multi-robot collaboration. The task execution processes are shown in Fig.3.

A. Experimental Setup

Hardware Configuration: Our system comprises three types of robots: (1) an Agilex Cobot Magic dual-arm mobile robot, (2) an ABB-YuMi dual-arm desktop robot, and (3) an Agilex Piper single-arm desktop robot. Based on the number of robots, we have categorized 6 tasks into two categories:

a) Table Top Dual-Arm Tasks: (1) Block Stacking: A precision task requiring the robot to place a rectangular block across two cylinders (bridge) and stack a cube on the top. (2) Table Organization: The robot needs to tidy up a workspace by multi-round pick & place 5 odds and ends into a basket sequentially under fuzzy instructions without knowing the objects on the table. (3) Object Stowing: An articulated object manipulation task which contains multiple stages involving opening a drawer, placing an object inside, and closing drawer.

b) Multi-Robot Collaboration: (1) Basket Shelving: A Piper robot places an object into a basket. The Cobot Magic robot lifts the basket with dual-arm and transports it to a shelf in a different workspace. (2) Collaborative Pouring: Receiving an instruction to pour a bowl of water, Cobot Magic first needs to move to a table which has a bowl on it, transport it to ABB's table, then place it in an appropriate place for ABB to pour water into it. (3) Food Preparation: A three-robot task. The ABB robot pours water into an instant noodle cup and places the cup back. Simultaneously, the Piper places two food objects on the plate on a different table. Finally, the Cobot Magic transports the prepared food plate from Piper's table to the ABB's table.

B. Dual-arm Table Top Task

We validate our framework's manipulation ability against (1) OpenVLA-OFT [25] and (2) π_0 [3], representing mainstream learning-based end-to-end policies. (3) ReKep [18], a learning-free method that employs VLMs to optimize relational keypoint constraints for robot manipulation. We report

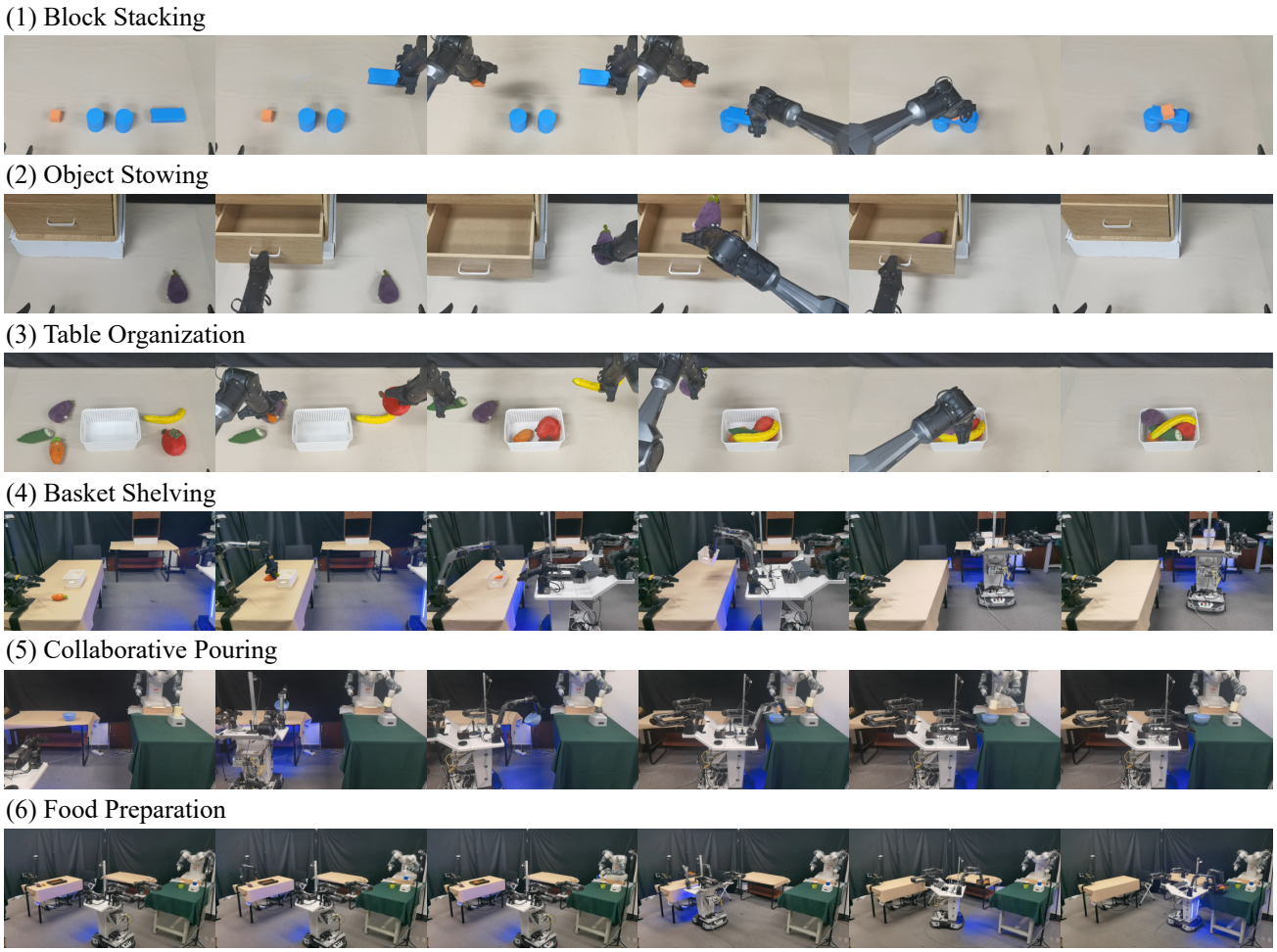

Fig. 3: **Execution Process of Six Real-world Experiments.** We validate our framework on (1-3) dual-arm manipulation tasks and (4-6) multi-robot collaborative scenarios, demonstrating the framework's versatility across varying complexities and robot configurations.

TABLE II: **Table Top Tasks Performance Comparison.** We evaluate OpenVLA-OFT, π_0 (learning-based), ReKep (LLM-based), and our method (Ours) across three tasks. For each method and task, we report: (1) number of successful trials out of 20, and (2) average task completion percentage.

Method	**Block Stacking**		**Table Org.**		**Object Stowing**	
	Succ.	Comp.(%)	Succ.	Comp.(%)	Succ.	Comp.(%)
OpenVLA-OFT [25]	1/20	35	5/20	55	4/20	31
π_0 [3]	9/20	46	9/20	75	8/20	60
ReKep Annot. [18]	7/20	56	6/20	44	0/20	0
Ours	**14/20**	**85**	**11/20**	**76**	**14/20**	**78**

success trials and task completion (Comp. %) over 20 trials for each task.

Experimental Setup. Both learning-based baselines (OpenVLA-OFT and π_0) are fine-tuned in a single-task setting using 50 demonstrations per task. To isolate perception errors and strictly evaluate manipulation capabilities, we provide ReKep with ground-truth keypoint annotations manually during execution.

As summarized in Tab. II, our method significantly outperforms all baselines on three tasks. The learning-based methods struggled in contact-rich tasks like *Block Stacking* and *Object Stowing*. Lacking precise geometric grounding, minor deviations in the predicted end-effector's pose frequently led to hard collisions with blocks and drawer, causing immediate failures. In long-horizon task *Table Organization*, baselines often fail to grasp the target object yet continue to execute the subsequent trajectory, letting these execution errors remain undetected and unchecked throughout the sequence. The learning-free method ReKep completely failed in *Object Stowing* because their original method could not define the end key points when operating drawers. In *Block Stacking*, the optimization solver often returns sub-optimal solutions that failed to strictly avoid collisions while attempting to reach the goal. In contrast, our framework achieved the highest success rates across all tasks, demonstrating that our tool-augmented perception ensures the precision required for stacking, the diverse combination action primitives provide the necessary versatility, and the Verification Agent provides the closed-loop robustness necessary to detect and try to recover from the execution errors.

C. Robustness to External Disturbances

To evaluate the robustness of our framework against real-world uncertainties, we designed three distinct disturbances

TABLE III: **Table Top Tasks Performance Comparison under External Disturbances.** Comparison of Success Rates (SR) across three desktop tasks. Baselines include OpenVLA-OFT and π_0.

Method	Stacking	Table Org.	Stowing	Avg. (%)
OpenVLA-OFT	0/10	0/10	2/10	7
π_0	2/10	1/10	3/10	20
Ours	**5/10**	**6/10**	**8/10**	**63**

TABLE IV: **System Robust Analysis under External Disturbances.** We evaluate the closed-loop pipeline under external disturbances. Metrics include Failure Identification rate (ID), Re-planning success rate (Re-Plan), Physical Recovery success rate (Phy-Rec), and the Final Task Success Rate (SR).

Task	Scenario	ID (%)	Re-Plan (%)	Phy-Rec (%)	Final SR (%)
Dual-Arm	Block Stacking	90	100	78	50
	Table Organization	100	100	80	60
	Object Stowing	90	100	100	80
Multi-Robot	Basket Shelving	90	100	89	40

ranging from local interference to global failure. We bring these disturbances at random intervals during the execution of both desktop and multi-robot tasks.

a) Action Level: Human operators remove or displace the target object while the robot is attempting to interact, simulating environmental changes or interaction failure.

b) Sub-task Level: After a sub-task is completed, the operator resets the environment state, specifically testing its ability to execute multi-step recovery sequences.

c) Global Level: In multi-robot task, a failure may occur if the target is placed outside the workspace of a fixed-base robot. The system must invoke a collaborative recovery and dispatch other robots to assist in the manipulation.

As shown in Tab. III, learning-based baselines struggled significantly under these perturbations, resulting in low average success rates. Due to the absence of a closed-loop feedback mechanism, these policies often continued to execute subsequent trajectories despite environmental change. While they may attempt a simple re-grasp if a failure occurs during the initial phase, they fail to recover from complex disturbances that require multi-step reasoning. In contrast, our method maintained a robust average success rate of **63%**. Compared to the undisturbed condition, the success rate shows only a slight decrease, demonstrating the critical necessity of our verification loop.

D. Verification Agent Analysis

To demonstrate the reliability of our Verification Agent, we tracked four metrics under the aforementioned disturbance conditions: Failure Identification Rate (ID), Re-planning Success Rate (Re-Plan), Physical Recovery Success Rate (Phy-Rec) which quantifies the robot's ability to physically resolve the disturbance conditioned on a valid plan and Final Task Success Rate (Task SR). The results are shown in Tab. IV. The Verification Agent shows robust failure identification capabilities around **90–100%** failure identification rate across all

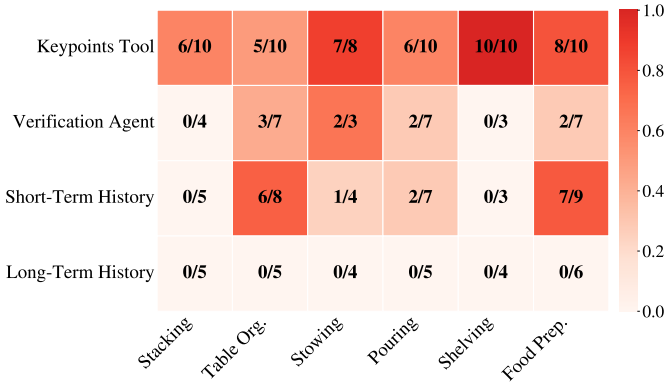

Fig. 4: **Analysis of Failure Contributions by Module Ablation.** We visualize the impact of removing specific components on task failures. The ratio (x/y) denotes the number of failures (x) directly attributed to the ablated module relative to the total number of failures (y) observed in that task.

tasks. This confirms that our VLM-based verification module effectively distinguishes between successful outcomes and varying types of errors.

In single-robot tasks, the system primarily relies on an internal recovery loop. The Re-planning Success Rate was consistently **100%**. However, there is a gap between detection and final success in execution, as irreversible states may occur during task execution, such as collapse of the building blocks, which require more meticulous operational skills to recover. In contrast, *Object Stowing* achieved a high 80% Final SR, proving the system's effectiveness in handling reversible state regressions. In the multi-robot task, we validate the global recovery capability of our framework in the *Basket Shelving* task. We placed an object outside the fixed-base robot's workspace, making it unreachable for the required operation. The system successfully identified the error caused by exceeding the workspace and performed global re-planning with **100%** re-planning success. The only exception occurred when the object moved out of the field of view after a failed placement operation.

E. Ablation Study

To analyze the contributions of each module within our framework, we conducted an ablation study across 6 tasks. We evaluated the system by systematically removing: (1) the visual perception tool, (2) the Verification Agent, (3) short-term memory (interaction history recorded by the Manipulation Agent), and (4) long-term memory (experience pool). Quantitative results are summarized in Tab. V, and Fig. 4 details module contributions to task failures.

Impact of visual perception tool. We employ the same visual perception method from ReKep[18] to replace our process, which uses position and feature clustering to obtain representative key points throughout the image. As shown in Fig. 4, the absence of the object-specific keypoint selection process resulted in a catastrophic performance drop. The results reveal the problems of previous multi-robots planning methods, they treat execution as idealized process, ignoring

TABLE V: **Ablation Study on Key Components.** We evaluate the impact of: (1) Keypoint Perception Tool, (2) Verification Agent, (3) Short-Term Memory, and (4) Long-Term Memory. The full model includes all components. Metrics: Success Rate (SR) and Task Completion (Comp. %).

Model Variant	Block Stacking		Table Org.		Object Stowing		Collab. Pouring		Basket Shelving		Food Prep.	
	Succ.	Comp. (%)	Succ.	Comp. (%)	Succ.	Comp. (%)	Succ.	Comp. (%)	Succ.	Comp. (%)	Succ.	Comp. (%)
Full Model (Ours)	14/20	85	11/20	76	14/20	78	11/20	79	12/20	84	10/20	69
– w/o Keypoint Perception Tool	0/10	35	0/10	32	2/10	45	0/10	16	0/10	50	0/10	26
– w/o Verification Agent	6/10	85	3/10	71	7/10	80	3/10	70	7/10	87	3/10	56
– w/o History in Planning	5/10	78	2/10	46	6/10	83	3/10	61	7/10	95	1/10	51
– w/o Exp.	5/10	78	5/10	65	6/10	80	5/10	84	6/10	80	4/10	70

that physical manipulation is coupled with actual perceptual process. For LLM-based manipulation approaches, the precise keypoint selection serves as the essential bridge connecting the high-level plan to the real-world environment. It is not merely an auxiliary step but essential for any successful manipulation.

Impact of Verification Agent. The ablation of the Verification Agent led to a decline particularly in long-horizon tasks like *Food Preparation* and *Table Organization*. In these tasks, every individual operation carries a non-zero failure probability. Without Verification Agent, these minor execution errors accumulate, eventually becoming fatal to the overall task. This validates that a closed-loop mechanism for error detection and recovery after each execution is essential for the successful completion of complex multi-robot tasks.

Impact of Short-Term Memory. Removing short-term memory caused a drop in sequential manipulation tasks, most notably in *Table Organization* and *Food Preparation*. To address changes in the environment following an operation, we decompose the sub-operation into action primitives. Therefore, we observed that without historical context, some sub-task instructions might be ambiguous, such as "place" command could mean that an object can be placed directly, or needs to be grasped before being placed. This was particularly evident in pouring sequences. Short-term memory eliminates these redundancies by recording the interaction history and state, helping Manipulation Agent generate sub-operations that are logically consistent with the robot's current status.

Impact of Long-Term Memory. For the long-term memory module, our primary objective is to enhance manipulation efficiency and reduce the frequency of LLM invocations. Experimental results indicate that removing this module only results in a slight decline and contributes to zero failures in method performance. This demonstrates the robustness of our Manipulation Agent in decomposing sub-tasks and generating action primitives, proving the framework does not strictly rely on memorization. Second, it demonstrates that the design of the experience pool enhances stability. By retrieving validated actions for repetitive sub-tasks, the system avoids the variance introduced by VLM generation, ensuring higher consistency and efficiency in execution.

F. System Error Breakdown

To identify the limitations of our current framework, we record the failure distribution across individual modules based on the results from the full model experiments in Tab. V. As illustrated in Fig. 5, keypoint selection constitutes the dominant failure mode. This bottleneck arises because the VLM struggles to localize keypoints that precisely satisfy the semantic constraints, resulting in a mismatch between the VLM's reasoning and the actual environment. Action primitives generation failures followed, sometimes the robot failed to find a kinematic solution or the VLM failed to correctly reason the relation between keypoints, resulting in incorrect parameter input. Beyond these two modules, failures such as Verification Agent errors and grasp failures account for a low percentage. Additionally, irrecoverable failures occasionally arise due to physical constraints, such as robotic arm collisions, noisy sensor data from depth cameras, and occasional navigation drift. However, each of these failures represents a relatively minor portion of the total cases.

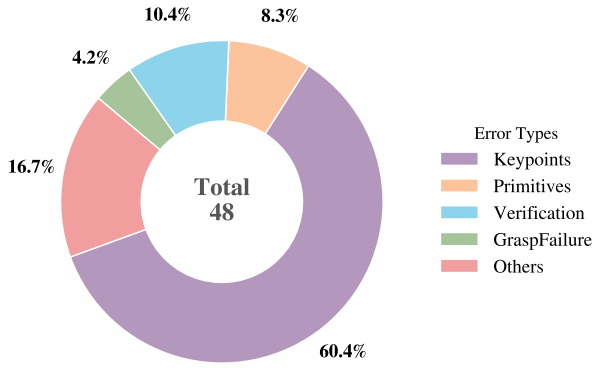

Fig. 5: **Error Breakdown in System Deployment.** We analyzed the total 48 failure cases encountered by the Full Model across all tasks reported in Tab. V.

V. Conclusions

In this paper, we propose a closed-loop multi-agent framework to enable robust and collaborative manipulation for a multi-robot system. By structuring the system into three specialized agents-Planner, Manipulation, and Verification, we bridge the gap between high-level reasoning and low-level physical execution. Extensive experiments across fine-grained desktop tasks to multi-robot collaborative tasks validate our framework possesses high versatility in executing diverse skills, ranging from precision pick-and-place to articulated object interactions. Furthermore, our framework demonstrates

robustness by maintaining continuous multi-robot workflows under disturbances through hierarchical recovery. Finally, transition from desktop setups to cross-workspace collaborations confirms the framework's adaptability in scaling from single-robot to heterogeneous multi-robot systems.

Limitations and Future Work. Currently, our system relies on a library of action primitives, which limits its ability to handle more complex manipulation tasks. Additionally, the current workflow is constrained by the execution speeds of heterogeneous robots. The system primarily employs a dependency check, lacking a dynamic scheduling mechanism to optimize the global efficiency. In future work, we aim to integrate learning-based primitive discovery and explore concurrent task allocation methods to minimize idle time and enhance the efficiency of multi-robot collaboration.

ACKNOWLEDGMENTS

This work was supported partially by NSFC (92470202), Guangdong NSF Project (No.2023B1515040025), Guangdong Key Research and Development Program (No.-2024B0101040004, No.2025B0909020002).

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. URL https://doi.org/10.48550/arXiv.2303.08774.

[2] Patrick Benavidez, Mohan Kumar, Sos S. Agaian, and Mo M. Jamshidi. Design of a home multi-robot system for the elderly and disabled. In *System of Systems Engineering Conference*, 2015. URL https://doi.org/10.1109/SYSOSE.2015.7151907.

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. *Robotics: Science and Systems*, 2024. URL https://doi.org/10.48550/arXiv.2410.24164.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[5] Jennifer Buehler and Maurice Pagnucco. A framework for task planning in heterogeneous multi robot systems based on robot capabilities. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2014. URL https://www.researchgate.net/publication/286126681_A_framework_for_task_planning_in_heterogeneous_multi_robot_systems_based_on_robot_capabilities.

[6] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? In *IEEE International Conference on Robotics and Automation*, 2024. URL https://doi.org/10.48550/arXiv.2309.15943.

[7] Zhe Chen, Javier Alonso-Mora, Xiaoshan Bai, Daniel D Harabor, and Peter J Stuckey. Integrated task assignment and path planning for capacitated multi-agent pickup and delivery. *IEEE Robotics and Automation Letters*, 2021. URL https://doi.org/10.48550/arXiv.2110.14891.

[8] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2025. URL https://journals.sagepub.com/doi/full/10.1177/02783649241273668.

[9] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. URL https://doi.org/10.48550/arXiv.2507.06261.

[10] Neil T. Dantam, Zachary K. Kingston, Swarat Chaudhuri, and Lydia E. Kavraki. Incremental Task and Motion Planning: A Constraint-Based Approach. In *Robotics: Science and Systems*, 2016. URL http://www.roboticsproceedings.org/rss12/p02.html.

[11] Yihong Dong, Xue Jiang, Jiaru Qian, Tian Wang, Kechi Zhang, Zhi Jin, and Ge Li. A survey on code generation with llm-based agents. *arXiv preprint arXiv:2508.00083*, 2025. URL https://doi.org/10.48550/arXiv.2508.00083.

[12] Haoshu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. AnyGrasp: Robust and Efficient Grasp Perception in Spatial and Temporal Domains. *IEEE Transactions on Robotics*, 2023. URL https://doi.org/10.1109/TRO.2023.3281153.

[13] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. PDDLStream: Integrating Symbolic Planners and Blackbox Samplers via Optimistic Adaptive Planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2020. URL https://ojs.aaai.org/index.php/ICAPS/article/view/6739.

[14] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong,

Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with Language Model is Planning with World Model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://doi.org/10.18653/v1/2023.emnlp-main.507.

[15] Haoxu Huang, Fanqi Lin, Yingdong Hu, Shengjie Wang, and Yang Gao. CoPa: General Robotic Manipulation through Spatial Constraints of Parts with Foundation Models. In *IEEE International Conference on Intelligent Robots and Systems,*, 2024. URL https://doi.org/10.1109/IROS58592.2024.10801352.

[16] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Tomas Jackson, Noah Brown, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner Monologue: Embodied Reasoning through Planning with Language Models. In *Conference on Robot Learning*, 2022. URL https://proceedings.mlr.press/v205/huang23c.html.

[17] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models. In *Conference on Robot Learning*, 2023. URL https://proceedings.mlr.press/v229/huang23b.html.

[18] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. ReKep: Spatio-Temporal Reasoning of Relational Keypoint Constraints for Robotic Manipulation. In *Conference on Robot Learning*, 2024. URL https://proceedings.mlr.press/v270/huang25g.html.

[19] William Hunt, Sarvapali D Ramchurn, and Mohammad D Soorati. A survey of language-based communication in robotics. *arXiv preprint arXiv:2406.04086*, 2024. URL https://doi.org/10.48550/arXiv.2406.04086.

[20] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Conference on Robot Learning*, 2022. URL https://proceedings.mlr.press/v205/ichter23a.html.

[21] Jian-Jian Jiang, Xiao-Ming Wu, Zibo Chen, Yi-Lin Wei, and Wei-Shi Zheng. igrasp: An interactive 2d-3d framework for 6-dof grasp detection. In *International Conference on Pattern Recognition*, 2024. URL https://link.springer.com/chapter/10.1007/978-3-031-78113-1_22.

[22] Jian-Jian Jiang, Xiao-Ming Wu, Yi-Xiang He, Ling-An Zeng, Yi-Lin Wei, Dandan Zhang, and Wei-Shi Zheng. Rethinking bimanual robotic manipulation: Learning with decoupled interaction framework. In *International Conference on Computer Vision*, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Jiang_Rethinking_Bimanual_Robotic_Manipulation_Learning_with_Decoupled_Interaction_Framework_ICCV_2025_paper.html.

[23] Xue Jiang, Yihong Dong, Lecheng Wang, Zheng Fang, Qiwei Shang, Ge Li, Zhi Jin, and Wenpin Jiao. Self-planning code generation with large language models. *ACM Transactions on Software Engineering and Methodology*, 2024. URL https://doi.org/10.48550/arXiv.2303.06689.

[24] Shyam Sundar Kannan, Vishnunandan L. N. Venkatesh, and Byung-Cheol Min. SMART-LLM: Smart Multi-Agent Robot Task Planning using Large Language Models. In *IEEE International Conference on Intelligent Robots and Systems*, 2024. URL https://doi.org/10.1109/IROS58592.2024.10802322.

[25] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. *Robotics: Science and Systems*, 2025. URL https://doi.org/10.48550/arXiv.2502.19645.

[26] Yeseung Kim, Dohyun Kim, Jieun Choi, Jisang Park, Nayoung Oh, and Daehyung Park. A survey on integration of large language models with intelligent robots. *Intelligent Service Robotics*, 2024. URL https://doi.org/10.48550/arXiv.2404.09228.

[27] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloé Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment Anything. In *International Conference on Computer Vision*, 2023. URL https://doi.org/10.1109/ICCV51070.2023.00371.

[28] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

[29] Peihan Li, Zijian An, Shams Abrar, and Lifeng Zhou. Large language models for multi-robot systems: A survey. *arXiv preprint arXiv:2502.03814*, 2025. URL https://doi.org/10.48550/arXiv.2502.03814.

[30] Zhi Li, Ali Vatankhah Barenji, Jiazhi Jiang, Ray Y. Zhong, and Gangyan Xu. A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand. *Journal of Intelligent Manufacturing*, 2020. URL https://doi.org/10.1007/s10845-018-1459-y.

[31] Zhuohao Li, Yinghao Li, Jian-Jian Jiang, Lang Zhou, Tianyu Zhang, and Wei-Shi Zheng. ReViP: Reducing False Completion in Vision-Language-Action Models with Vision-Proprioception Rebalance. *arXiv*, 2026. URL https://arxiv.org/abs/2601.16667.

[32] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol

Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control. In *IEEE International Conference on Robotics and Automation*, 2023. URL https://doi.org/10.1109/ICRA48891.2023.10160591.

[33] Yuhao Lin, Yi-Lin Wei, Haoran Liao, Mu Lin, Chengyi Xing, Hao Li, Dandan Zhang, Mark R. Cutkosky, and Wei-Shi Zheng. TypeTele: Releasing Dexterity in Teleoperation by Dexterous Manipulation Types. In *Conference on Robot Learning*, 2025. URL https://arxiv.org/abs/2507.01857.

[34] Jiayi Liu, Jiaming Zhou, Ke Ye, Kun-Yu Lin, Allan Wang, and Junwei Liang. EgoTraj-Bench: Towards Robust Trajectory Prediction Under Ego-view Noisy Observations. *arXiv*, 2025. URL https://arxiv.org/abs/2510.00405.

[35] 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 *IEEE International Conference on Robotics and Automation*, 2025. URL https://arxiv.org/abs/2409.15146.

[36] Xinzhu Liu, Peiyan Li, Wenju Yang, Di Guo, and Huaping Liu. Leveraging large language model for heterogeneous ad hoc teamwork collaboration. *arXiv preprint arXiv:2406.12224*, 2024. URL https://doi.org/10.48550/arXiv.2406.12224.

[37] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language Models of Code are Few-Shot Commonsense Learners. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2022. URL https://doi.org/10.18653/v1/2022.emnlp-main.90.

[38] Zhao Mandi, Shreeya Jain, and Shuran Song. RoCo: Dialectic Multi-Robot Collaboration with Large Language Models. In *IEEE International Conference on Robotics and Automation*, 2024. URL https://doi.org/10.1109/ICRA57147.2024.10610855.

[39] Kazuma Obata, Tatsuya Aoki, Takato Horii, Tadahiro Taniguchi, and Takayuki Nagai. LiP-LLM: Integrating Linear Programming and Dependency Graph With Large Language Models for Multi-Robot Task Planning. *IEEE Robotics and Automation Letters*, 2025. URL https://doi.org/10.1109/LRA.2024.3518105.

[40] Mingjie Pan, Jiyao Zhang, Tianshu Wu, Yinghao Zhao, Wenlong Gao, and Hao Dong. Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025. URL https://doi.org/10.48550/arXiv.2501.03841.

[41] Guangyao Shi, Yuwei Wu, Vijay Kumar, and Gaurav S Sukhatme. PIP-LLM: Integrating PDDL-Integer Programming with LLMs for Coordinating Multi-Robot Teams Using Natural Language. *arXiv preprint arXiv:2510.22784*, 2025. URL https://doi.org/10.48550/arXiv.2510.22784.

[42] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models. In *IEEE International Conference on Robotics and Automation*, 2023. URL https://doi.org/10.1109/ICRA48891.2023.10161317.

[43] Kun Song, Shentao Ma, Gaoming Chen, Ninglong Jin, Guangbao Zhao, Mingyu Ding, Zhenhua Xiong, and Jia Pan. CollaBot: Vision-language guided simultaneous collaborative manipulation. *arXiv preprint arXiv:2508.03526*, 2025. URL https://doi.org/10.48550/arXiv.2508.03526.

[44] Chenyu Su, Weiwei Shang, Chen Qian, Fei Zhang, and Shuang Cong. ReSemAct: Advancing Fine-Grained Robotic Manipulation via Semantic Structuring and Affordance Refinement. *arXiv preprint arXiv:2507.18262*, 2025. URL https://doi.org/10.48550/arXiv.2507.18262.

[45] An-Lan Wang, Nuo Chen, Kun-Yu Lin, Yuan-Ming Li, and Wei-Shi Zheng. Task-oriented 6-dof grasp pose detection in clutters. In *IEEE International Conference on Robotics and Automation*, 2025. URL https://ieeexplore.ieee.org/abstract/document/11128749.

[46] Yi-Lin Wei, Jian-Jian Jiang, Chengyi Xing, Xiantuo Tan, Xiao-Ming Wu, Hao Li, Mark R. Cutkosky, and Wei-Shi Zheng. Grasp as You Say: Language-guided Dexterous Grasp Generation. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5367f6d58cc98dc929e1b27fcaf2b0a6-Abstract-Conference.html.

[47] Yi-Lin Wei, Haoran Liao, Yuhao Lin, Pengyue Wang, Zhizhao Liang, Guiliang Liu, and Wei-Shi Zheng. CycleManip: Enabling Cyclic Task Manipulation via Effective Historical Perception and Understanding. *arXiv preprint arXiv:2512.01022*, 2025. URL https://arxiv.org/abs/2512.01022.

[48] Yi-Lin Wei, Mu Lin, Yuhao Lin, Jian-Jian Jiang, Xiao-Ming Wu, Ling-An Zeng, and Wei-Shi Zheng. AffordDexGrasp: Open-set Language-guided Dexterous Grasp with Generalizable-Instructive Affordance. In *International Conference on Computer Vision*, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Wei_AffordDexGrasp_Open-set_Language-guided_Dexterous_Grasp_with_Generalizable-Instructive_Affordance_ICCV_2025_paper.html.

[49] Yi-Lin Wei, Zhexi Luo, Yuhao Lin, Mu Lin, Zhizhao Liang, Shuoyu Chen, and Wei-Shi Zheng. OmniDexGrasp: Generalizable Dexterous Grasping via Foundation Model and Force Feedback. *arXiv*, 2025. URL https://arxiv.org/abs/2510.23119.

[50] Jianan Xie, Wei Zhang, Hongming Chen, Jiayu Zeng, Yuyang Gao, Zhen Xu, and Kenji Hashimoto. Centralized LLM-Driven Multi-Robot Coordination for Cooperative Object Transportation. In *IEEE International*

Conference on Advanced Robotics and its Social Impacts, 2025. URL https://ieeexplore.ieee.org/document/11124964.

[51] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 2013. URL https://journals.sagepub.com/doi/10.5772/57313.

[52] 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. URL https://doi.org/10.48550/arXiv.2505.09388.

[53] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. Co-navgpt: Multi-robot cooperative visual semantic navigation using large language models. *arXiv preprint arXiv:2310.07937*, 2023. URL https://doi.org/10.48550/arXiv.2310.07937.

[54] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. In *Robotics: Science and Systems*, 2024. URL https://arxiv.org/abs/2403.03954.

[55] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models, journal=arXiv preprint arXiv:2307.02485. 2023. URL https://doi.org/10.48550/arXiv.2307.02485.

[56] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B. Tenenbaum, Tianmin Shu, and Chuang Gan. Building Cooperative Embodied Agents Modularly with Large Language Models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=EnXJfQqy0K.

[57] Xiaopan Zhang, Hao Qin, Fuquan Wang, Yue Dong, and Jiachen Li. Lamma-p: Generalizable multi-agent long-horizon task allocation and planning with lm-driven pddl planner. In *IEEE International Conference on Robotics and Automation*, 2025. URL https://doi.org/10.48550/arXiv.2409.20560.

[58] Jiaming Zhou, Teli Ma, Kun-Yu Lin, Zifan Wang, Ronghe Qiu, and Junwei Liang. Mitigating the Human-Robot Domain Discrepancy in Visual Pre-training for Robotic Manipulation. In *Proceedings of the Computer Vision and Pattern Recognition*, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhou_Mitigating_the_Human-Robot_Domain_Discrepancy_in_Visual_Pre-training_for_Robotic_CVPR_2025_paper.html.

[59] Jiaming Zhou, Ke Ye, Jiayi Liu, Teli Ma, Zifan Wang, Ronghe Qiu, Kun-Yu Lin, Zhilin Zhao, and Junwei Liang. Exploring the Limits of Vision-Language-Action Manipulation in Cross-task Generalization. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/cc92809cd8dfbd035801966ab4896741-Paper-Conference.pdf.