Long-Horizon Manipulation via Trace-Conditioned VLA Planning

Isabella Liu[1], An-Chieh Cheng[1], Rui Yan[1], Geng Chen[1], Ri-Zhao Qiu[1], Xueyan Zou[1], Sha Yi[1], Hongxu Yin[2†], Xiaolong Wang[1†], and Sifei Liu[2†]

[1] University of California, San Diego
[2] NVIDIA

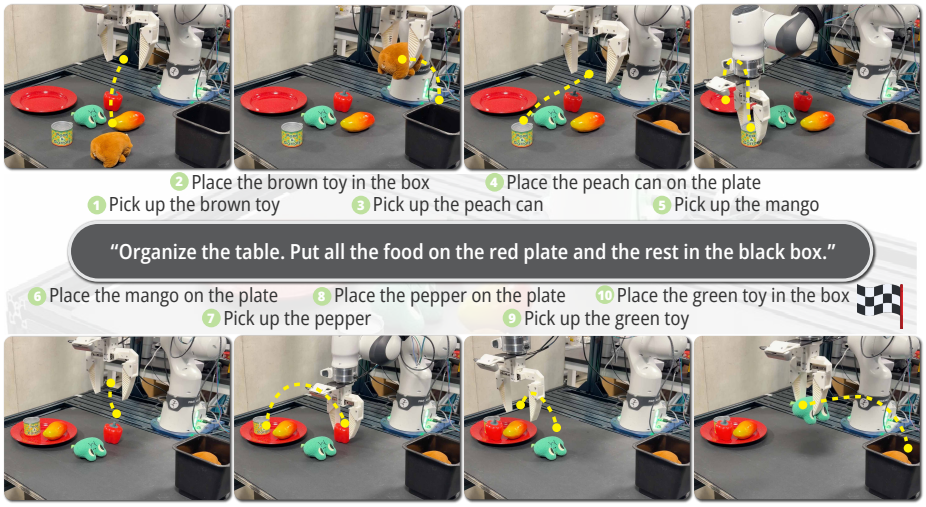

Fig. 1: Our task manager decouples high-level planning from low-level control by decomposing a high-level instruction into sequential sub-tasks and tracking their completion over time. For each step, it predicts a spatial visual trace that acts as an actionable prompt for a low-level VLA executor, enabling reliable execution of complex multi-step manipulation tasks. Project page: https://www.liuisabella.com/LoHoManip.

Abstract. Long-horizon manipulation remains challenging for vision-language-action (VLA) policies: real tasks are multi-step, progress-dependent, and brittle to compounding execution errors. We present LoHo-Manip, a modular framework that scales short-horizon VLA execution to long-horizon instruction following via a dedicated task-management VLM. The manager is decoupled from the executor and is invoked in a receding-horizon manner: given the *current* observation, it predicts a progress-aware *remaining plan* that combines (i) a subtask sequence with an explicit *done* + *remaining* split as lightweight language memory, and (ii) a visual trace—a compact 2D keypoint trajectory prompt specifying where to go and what to approach next. The executor VLA is adapted to condition on the rendered trace, thereby turning long-horizon decision-making

† Equal advising.

into repeated local control by "following the trace." Crucially, predicting the remaining plan at each step yields an implicit closed loop: failed steps persist in subsequent outputs, and traces update accordingly, enabling automatic continuation and replanning without hand-crafted recovery logic or brittle visual-history buffers. Extensive experiments spanning embodied planning, long-horizon reasoning, trajectory prediction, and end-to-end manipulation in simulation and on a real Franka robot demonstrate strong gains in long-horizon success, robustness, and out-of-distribution generalization.

Keywords: Long-horizon Manipulation · Vision Language Action Models · Vision Language Models

1 Introduction

Robots have made striking progress in short-horizon manipulation—picking, placing, opening, and pushing—driven by large-scale imitation learning [9, 13, 50, 71] and increasingly capable vision-language(-action) models [6–8, 36]. Yet the gap between these skills and long-horizon manipulation remains wide [20,43]. Real tasks rarely end after a single grasp. They require dozens of interdependent to-dos, state-dependent decisions (e.g., "the kettle is not full enough"), and robust recovery when any step fails. A household instruction such as "refill the kettle" implicitly expands into a long to-do list: locate a cup, grasp it, move to the faucet, fill it to an appropriate level, close the faucet, navigate to the kettle, open the lid, pour, close the lid, and so on. Long-horizon success demands not only low-level control but also *task management*: decomposition, progress tracking, and continuous re-evaluation as the world changes.

A common strategy is to increase the capability of a single monolithic policy—either by scaling a VLA model [8, 9, 71], or by embedding high-level planning within the same model that produces actions [19,27,31,68]. However, tightly coupling planning and execution introduces two persistent limitations. First, fragility under drift [22,32,55]: long sequences amplify small errors, and one-shot plans cannot reliably anticipate partial failures, occlusions, or object motion. Second, poor modularity [12,18,37]: if the planner is fused into the executor, upgrading or swapping the low-level VLA (e.g., changing embodiments, action spaces, or training domains) typically requires re-engineering the entire stack. This tension is especially acute in embodied settings, where task complexity, environment variability, and distribution shift are the norm.

We argue that long-horizon manipulation benefits from a cleaner separation of concerns: a high-level component should focus on what remains to be done, while a low-level component should focus on how to do the next short-term control. Concretely, we introduce LoHo-Manip, a hierarchical system centered around a dedicated task-management VLM that sits above any VLA policy. The manager is responsible for task-level reasoning and outputs two complementary artifacts at inference time:

- *Remaining subtasks*: a structured list describing the actions that still need to be completed from the current observation onward.
- *Visual trace*: a high-level 2D trajectory (a "roadmap") that specifies where the agent should move and which objects/regions should be approached next.

The trace is not merely an auxiliary visualization; it is a conditioning signal for the executor. By training or adapting the VLA to follow this trace, we convert a difficult long-horizon planning problem into a sequence of short-horizon control problems, where modern VLAs excel. Intuitively, the manager answers "*what/where next*" with an explicit visual pointer, while the VLA handles "*how to do it*" with precise actions.

A key design choice in LoHo-Manip is the use of *receding-horizon* task management via *remaining-plan prediction*. Instead of generating a complete plan only at the first frame, the manager is invoked periodically (or at every step) and always predicts the remaining plan from the current observation. This yields a simple but powerful closed-loop property: if a subtask fails (e.g., the cup was not grasped), the world state reflects that failure, and the manager continues to include the uncompleted item in its subsequent plan — often accompanied by an updated trace. As a result, LoHo-Manip exhibits implicit progress tracking, implicit replanning, and implicit recovery without requiring hand-crafted failure detectors or special-case logic.

Importantly, long-horizon management must represent *history* without becoming brittle or expensive. Feeding long observation histories can increase latency and expose the manager to distribution shift when execution becomes imperfect, while emitting only a single "next subtask" can lead to unstable instruction streams under repeated failures. LoHo-Manip adopts a lightweight alternative: the manager conditions solely on the *current* observation, while maintaining progress through a compact *textual memory* that summarizes what has already been completed (e.g., "done: a–c; remaining: d–f"). This simple design simultaneously preserves explicit task-state bookkeeping, avoids reliance on long visual histories, and keeps the subtask interface stable by predicting a *remaining* sequence rather than a one-step directive.

As the additional visual prompt to the VLA executor, the trace also offers a direct path to generalization. Many VLAs overfit to the visual and semantic statistics of their training distributions (e.g., a particular set of objects, textures, or colors). In contrast, a general-purpose VLM is often better at grounding language to unseen objects and producing a coarse spatial plan. The trace externalizes this grounding as a visual instruction (e.g., literally drawing a path toward an object of an unseen category). Once the executor learns the generic skill of trace-following, it can execute behaviors on targets it has never encountered, provided the manager can point to them.

We evaluate LoHo-Manip on multiple long-horizon embodied and manipulation benchmarks and show that our framework effectively bridges high-level reasoning with low-level control. Our experiments, spanning embodied planning, long-horizon reasoning, trajectory prediction, and end-to-end manipulation in both simulation and on a real Franka robot, demonstrate strong gains in long-horizon success, robustness, and out-of-distribution generalization. By decomposing complex instructions into grounded sub-task trajectories, our model maintains temporal consistency and significantly outperforms existing vision-language baselines in both simulated and real-world tabletop environments. We summarize the contributions as follows:

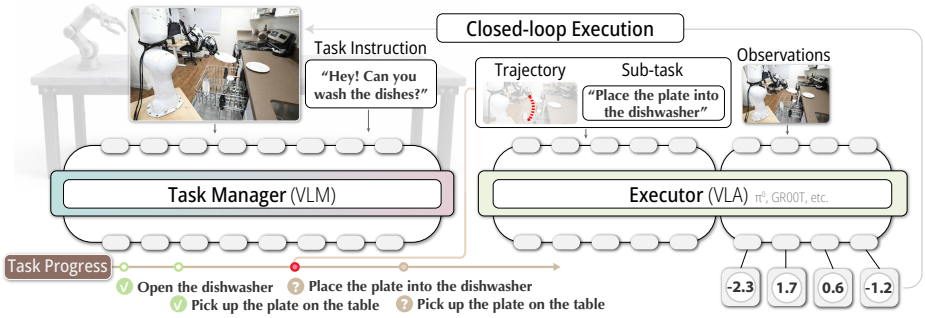

Fig. 2: Overview of LoHo-Manip framework. Given a task instruction and the current observation, a vision–language *task manager* predicts the next sub-task together with a spatial visual trace indicating the intended interaction trajectory. A low-level *executor* then performs short-horizon control conditioned on this guidance. The system operates in a *closed loop*: after executing actions, new observations are fed back to the manager to update task progress and generate the next sub-task and trace, enabling robust long-horizon manipulation and recovery from execution errors.

- We propose LoHo-Manip, a modular framework that separates a dedicated task-management VLM from a short-horizon VLA executor, enabling reusable high-level management across different low-level policies.
- We train the manager to predict *remaining* subtasks (and the corresponding trace) from the current observation throughout an episode, yielding implicit progress tracking, re-planning, and failure recovery without hand-crafted heuristics.
- We introduce a visual trace as an actionable prompt for the executor, turning long-horizon planning into local control and improving robustness and generalization to novel objects and environments.

2 Method

Directly learning a policy that maps observations to long-horizon action sequences is difficult in practice: the policy must implicitly track task progress, handle compounding execution errors, and maintain consistent behavior over many steps. As the horizon grows, these challenges lead to unstable credit assignment and distribution shift between training trajectories and real executions.

To address this, we adopt a hierarchical decomposition that separates *task management* from *short-horizon control*. The manager predicts the remaining high-level plan from the current observation, while a low-level policy executes locally conditioned actions. This formulation converts long-horizon manipulation into sequential short-horizon execution with explicit progress signals.

2.1 System Overview

Our system decomposes long-horizon manipulation into two components: a high-level **task manager** that reasons about task progress, and a low-level **executor** that performs short-horizon control.

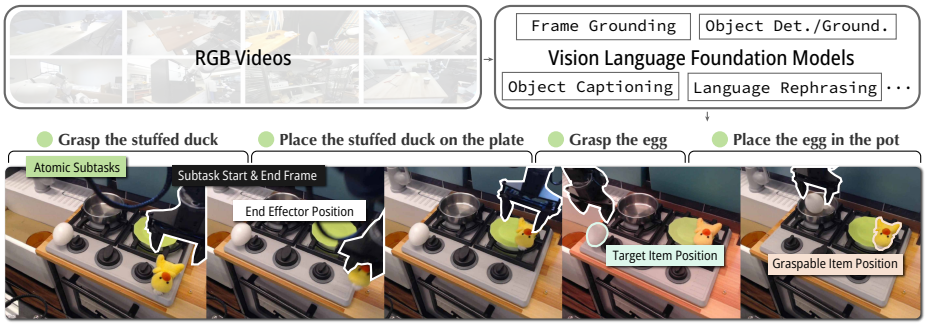

Fig. 3: Data Pipeline. Given RGB manipulation videos, we use vision–language foundation models to perform frame grounding, object detection, and captioning to identify interaction events. The trajectory is segmented into atomic subtasks with corresponding start and end frames, while the robot end-effector positions are extracted to form a 2D visual trace. Additional grounding identifies target and graspable objects, producing paired (subtask instruction, visual trace) supervision for training the task manager.

Task Manager. The task manager is a vision-language model that predicts the *remaining task structure* from the current observation. Given the instruction x and the current observation o_t, the manager produces two outputs: (1) a **subtask description** s_t that specifies the next short-horizon objective in natural language, and (2) a **visual trace** τ_t represented as a sequence of 2D keypoints indicating the spatial trajectory or interaction target associated with the subtask. Importantly, the task manager is designed as a *generalist module*. It does not depend on the specific low-level policy used for execution, allowing the same manager to interface with different VLA models.

Executor (VLA). The executor is a standard action policy (e.g., π_0, GR00T, or other VLA/VA backbones) that maps observations to robot actions. Modern VLA models are particularly effective at executing *short-horizon instructions*, making them well suited to follow the subtasks produced by the manager.

To connect the two components, the predicted visual trace τ_t is provided as an additional input to the executor. The trace is rendered into the observation and used as a spatial conditioning signal. After fine-tuning on this representation, the VLA learns to follow the predicted trajectory and execute the corresponding subtask.

Closed-loop execution. During execution, the system repeatedly invokes the task manager to predict the remaining plans from the current observation. The executor then performs locally conditioned actions guided by the predicted subtask and trace. This receding-horizon interaction enables explicit task progress tracking and robust recovery from execution errors.

2.2 Sub-Task and Trace Construction

Progress-aware plan representation. Given an instruction x and a trajectory of observations $\{o_t\}_{t=1}^{T}$, we represent the task as a sequence of *atomic interaction primitives* $\bar{S} = [\bar{s}^{(1)}, \ldots, \bar{s}^{(K)}]$. At each time t, instead

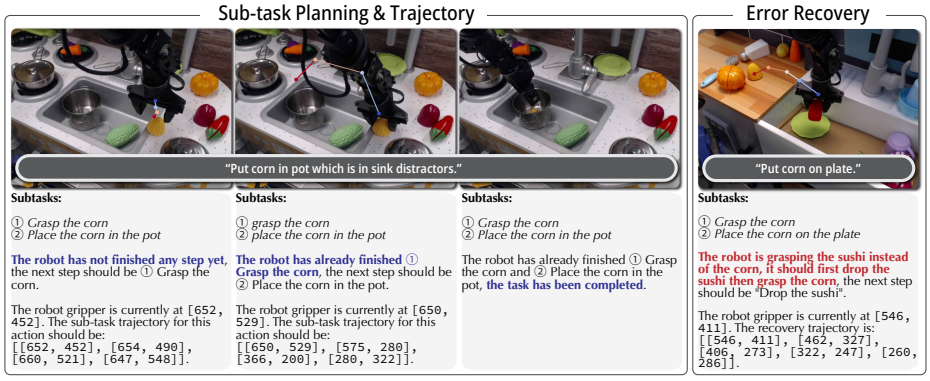

Fig. 4: Samples of Sub-task Planning and Error Recovery. *Left:* Sequential samples from a single episode demonstrating sub-task decomposition and trajectory grounding. The planning module maintains temporal context to track progress, while predicted trajectories map instructions to spatial coordinates to reduce ambiguity. *Right:* A curated error recovery sample. The model identifies a semantic error (grasping sushi instead of corn), triggers a corrective "Drop the sushi" sub-task, and generates a recovery trajectory to reset the state and resume the original goal.

of emitting only a single "next subtask", we define a *progress-aware plan text* that explicitly includes both the completed prefix and the remaining suffix:

$$C_t^\star = [\bar{s}^{(1)}, \ldots, \bar{s}^{(k(t)-1)}], \quad R_t^\star = [\bar{s}^{(k(t))}, \ldots, \bar{s}^{(K)}], \tag{1}$$

where $k(t)$ denotes the index of the current (or next) primitive at time t. We use $C_t^\star$ as a compact *language memory* summarizing what has been done, and $R_t^\star$ as the remaining plan to be executed.

Visual trace representation. We additionally associate the plan with a lightweight *visual trace* that encodes spatial intent. For each timestep, we obtain the 2D pixel coordinate of the robot end-effector $p_t \in \mathbb{R}^2$. We define the remaining trace label as the future portion of the end-effector trajectory:

$$\tau_t^\star = \{p_t, p_{t+1}, \ldots, p_{t_K^e}\}, \tag{2}$$

where t_K^e is the end index of the final primitive. In practice, $\tau_t^\star$ is stored in a compact form (e.g., resampled waypoints) and rendered as a visual prompt.

Current-frame conditioning. Although supervision is constructed from full trajectories, our task manager is conditioned on the *current frame only* during both training and inference. Concretely, the manager receives (x, o_t, C_{t-1}) and predicts (C_t, R_t, τ_t). This keeps the visual input distribution consistent between training and deployment, while C_t provides explicit progress context without requiring history frames.

Constructing primitives and traces. To construct $\bar{S}$ and $\{\tau^{(k)}\}$ from video trajectories, we (i) temporally segment the episode into interaction events and map each segment to an atomic primitive, and (ii) extract end-effector pixel locations to form traces. For segmentation, we leverage off-the-shelf vision–language models with video grounding capabilities to

identify physically grounded interaction events and their temporal spans (t_k^s, t_k^e). For trace extraction, we localize the robot end-effector per frame to obtain p_t, which directly yields $\tau^{(k)}$ for each segment. In simulation, the end-effector location can be obtained from state; on real robots, it can be obtained via visual localization. We defer model-specific choices and implementation details to Sec. 2.3.

Shift-resilient language memory. A practical challenge in long-horizon management is representing history without introducing train–test mismatch. Feeding long visual histories can be brittle when real executions deviate from smooth demonstrations, leading to distribution shift in the history frames (also noted in [60]). We therefore condition the manager on the *current* frame only, and encode history as a compact textual summary (C_t: "completed primitives") alongside the remaining plan (R_t). This preserves explicit progress context for decision making, avoids reliance on history frames, and keeps inference efficient.

2.3 Training Paradigm

Our training separates the *task manager* from the *executor*. The manager is trained to predict a progress-aware *remaining plan* and its associated trace from the *current* observation, while the executor is adapted to follow the trace prompt for short-horizon control. This decoupling keeps the manager executor-agnostic and allows reusing the same manager checkpoint across different VLA backbones, while each executor is fine-tuned to the trace interface.

Training data. We use two sources of supervision. (i) **Real-robot demonstrations** from the Bridge subset (in Open X-Embodiment format) provide video trajectories from which we extract atomic interaction primitives and end-effector traces following Sec. 2.2 [50]. (ii) **Auxiliary long-horizon reasoning/planning data** (RoboVQA and EgoPlan-BenchIT) provides additional instruction understanding and progress-reasoning signals that improve the manager's generalization beyond the manipulation distribution [45,56]. To improve the task manager's robustness to execution errors, we augment the training set with synthesized failure-recovery samples from the Bridge dataset. We filter for grasp-and-place episodes and identify the transition frames where grasping and placing occur. We then compose "fake" failure data by replacing the original grasped object with other detected graspable items in the scene. A fail recovery example is demonstrated in Fig. 4.

Task manager training. We initialize the task manager from a pretrained VLM [4]. We freeze the vision encoder and fine-tune the language model with supervised learning to predict: (a) a progress-aware plan text that includes both completed and remaining primitives, and (b) the associated 2D trace representation (e.g., waypoint sequence) for the remaining execution. Crucially, the manager is conditioned on the *current frame only* during training, with history provided via the textual progress summary, which avoids relying on long visual histories that may drift under imperfect rollouts.

Executor adaptation. For the low-level executor, we adopt the $\pi_{0.5}$ architecture [31] and initialize from its base checkpoint. We then fine-tune the executor to condition on the rendered trace prompt (and optionally the

current subtask text), so that it can reliably track the manager-provided trajectory and execute short-horizon interaction primitives.

Inference loop and memory update. At inference time, we run a receding-horizon closed loop. At each step (or every fixed interval), the manager takes the current observation and the textual progress summary and predicts an updated (completed, remaining) plan plus trace. The executor then acts for a short horizon conditioned on the trace prompt. We update the textual memory by taking the latest completed/remaining split from the manager output, without feeding history frames. This design preserves explicit progress context while remaining efficient and robust to distribution shift in long visual histories.

3 Experiments

Our evaluation validates LoHo-Manip across four key dimensions. We first verify the model's fundamental reasoning and trajectory prediction capabilities (Sec. 3.1). We then assess the model's sub-task planning accuracy on embodied agent benchmarks (Sec. 3.2). Moving to closed-loop control, we evaluate end-to-end manipulation in simulation (Sec. 3.3) to analyze the integration of high-level reasoning and low-level control. We conclude with real-world experiments (Sec. 3.4), showcasing the model's ability to handle long-horizon instructions in unconstrained environments.

3.1 Embodied Reasoning and Trajectory Prediction

We evaluate the embodied reasoning and spatial grounding capabilities of our task manager across four benchmarks: **RoboVQA** [56] for long-horizon reasoning, **EgoPlan-Bench2** [52] for human-level planning, and **ShareRobot-T** [33] and **VABench-V** [65] for 2D trajectory prediction. These capabilities are critical for a task manager to effectively bridge the gap between abstract user instructions and executable robot plans.

Quantitative results are reported in Tab. 1 and Tab. 2, where we compare LoHo-Manip against leading proprietary models (e.g., Gemini-3.0-Flash [24]), general-purpose open-source VLMs (e.g., Qwen3-VL [4]), and specialized embodied foundation models of comparable scale(e.g., ThinkAct [27]). Our method consistently outperforms all baselines, demonstrating superior accuracy in both high-level semantic planning and low-level spatial trajectory generation.

A key strength of our high-level task manager is its ability to generalize across robotic setups and environments, including varying manipulators, object categories, and scene. The manager acts as a modular component that can seamlessly interface with different downstream executor policies. This architectural decoupling allows LoHo-Manip to remain effective even when deployed on embodiments or datasets not seen during the training of the task manager, highlighting its potential as a foundation model for embodied planning. To further highlight these advantages, we provide qualitative samples in Fig. 8, showing results of trajectory prediction on out-of-distribution embodiments.

Methods	RoboVQA ↑					EgoPlan2 ↑				
	B-1	B-2	B-3	B-4	Avg.	Day.	Wrk.	Rec.	Hob.	Avg.
Proprietary Models (API)										
GPT-4V [1]	32.2	26.5	24.7	23.9	26.8	36.7	27.7	33.9	32.5	32.6
Gemini-2.5-Flash [14]	39.1	31.6	22.9	22.1	28.9	44.2	42.3	43.2	39.1	42.4
Gemini-3.0-Flash [24]	46.6	36.4	33.8	32.2	37.3	55.8	45.7	45.9	46.3	48.8
Open-source Models										
InternVL2.5-2B [11]	36.6	33.7	31.0	29.4	32.7	30.9	27.8	28.6	33.1	30.1
InternVL3-2B [70]	34.4	33.9	33.5	33.3	33.8	36.9	29.9	35.6	31.5	33.4
NVILA-2B [44]	38.7	34.3	31.1	29.2	33.3	34.6	26.7	33.3	31.6	31.4
Qwen2.5-VL-3B [5]	42.5	36.3	28.7	31.8	34.8	29.0	27.0	30.2	28.9	28.5
Qwen3-VL-4B [4]	62.5	52.2	47.0	43.2	51.2	32.3	31.2	33.8	28.9	31.3
Qwen3-VL-8B [4]	72.6	62.9	56.4	51.4	60.8	39.6	37.7	38.8	31.6	36.6
Embodied Foundation Models										
Magma-8B [63]	38.6	31.5	28.1	26.7	31.2	32.1	25.7	34.4	29.3	29.8
RoboBrain2.0-3B [59]	54.4	47.7	43.1	41.0	46.5	45.3	37.6	45.9	39.7	41.8
RoboBrain2.0-7B [59]	37.4	31.0	27.1	25.8	30.0	39.4	29.7	33.9	32.2	33.2
RoboBrain2.5-8B [58]	48.8	38.2	33.2	29.8	37.5	43.5	40.8	38.8	40.1	41.2
Fast-ThinkAct-3B [26]	70.1	63.0	57.2	53.0	60.8	50.3	44.3	46.4	43.2	46.4
ThinkAct-3B [27]	62.4	57.3	52.0	49.6	55.3	46.6	41.4	45.9	42.5	44.0
ThinkAct-7B [27]	69.1	61.8	56.0	52.4	59.8	50.1	49.8	44.8	45.2	48.2
RynnBrain-8B [16]	74.3	63.9	57.3	52.6	62.1	38.9	32.6	35.5	31.4	34.8
LoHo-Manip-4B	**75.1**	**65.0**	**58.6**	**53.5**	**63.1**	**60.4**	**56.4**	**51.9**	**54.8**	**56.7**

Table 1: Performance comparison across two key benchmarks: RoboVQA [56] for long-horizon reasoning and EgoPlan-Bench2 [52] for human-level planning. We evaluate our method against proprietary, open-source, and embodied foundation models of comparable scale. Our approach outperforms all baselines, achieving state-of-the-art results. We report the BLEU score for RoboVQA and Accuracy (%) for EgoPlan-Bench2.

3.2 Embodied Agent

To evaluate the high-level planning ability of our task manager, we conduct experiments on EmbodiedBench [64]. This benchmark is uniquely designed to evaluate embodied agents' proficiency in instruction understanding, commonsense reasoning, and long-horizon planning.

In this setup, the model is queried with an egocentric observation image and must output atomic commands (e.g., `PickUp`, `Open`) that are directly executable by robot APIs. By bypassing the complexities of continuous low-level motor control while requiring visually-grounded discrete actions, EmbodiedBench serves as the most representative benchmark for VLA capability at the planning level.

We focus our evaluation on two environments:

- **EB-Alfred:** Based on the ALFRED dataset, requiring multi-step household task completion.
- **EB-Habitat:** Utilizing the Habitat simulator for semantic navigation and interaction in 3D indoor scenes.

Tab. 3 reports the evaluation results. Our method achieves leading performance across both EB-Alfred and EB-Habitat compared with existing multimodal models of similar or even larger scale. These results demonstrate that LoHo-Manip can effectively interpret long-horizon instructions and produce accurate sub-task plans, highlighting the strength of our hierarchical task reasoning design.

Methods	ShareRobot-T			VABench-V		
	DFD ↓	HD ↓	RMSE ↓	DFD ↓	HD ↓	RMSE ↓
Qwen3-VL-4B [4]	0.3808	0.3294	0.2204	0.2792	0.2528	0.2037
MolmoAct-7B [38]	0.7764	0.7764	0.6771	0.8136	0.8136	0.6877
Hamster-13B [40]	0.4365	0.3919	0.3554	0.2124	0.2045	0.1825
Embodied-R1-3B [66]	0.3426	0.3002	0.2388	0.3028	0.2588	0.2129
LoHo-Manip-4B	**0.2309**	**0.2058**	**0.1559**	**0.2123**	**0.1821**	**0.1469**

Table 2: Performance comparison on trajectory prediction benchmarks. We evaluate on ShareRobot-T [33] and VABench-V [65] across three key metrics: Discrete Fréchet Distance (DFD), Hausdorff Distance (HD), and Root Mean Square Error (RMSE).

Methods	EmbodiedBench ↑						
	Base	Common	Complex	Long	Spatial	Visual	Average
EB-Alfred							
GPT-4o mini [49]	0.34	0.28	0.36	0.00	0.22	0.24	0.24
Llama3.2-11B [46]	0.24	0.08	0.16	0.06	0.06	0.22	0.14
InternVL3-8B [70]	0.20	0.14	0.14	0.02	0.00	0.12	0.10
Qwen3-VL-4B [4]	0.18	0.14	0.26	0.26	0.14	0.14	0.19
LoHo-Manip-4B	**0.48**	**0.36**	**0.54**	**0.31**	**0.30**	**0.28**	**0.38**
EB-Habitat							
GPT-4o mini [49]	0.74	0.22	0.32	0.14	0.32	0.22	0.33
Llama3.2-11B [46]	0.70	0.16	0.28	0.06	0.20	0.10	0.25
InternVL3-8B [70]	0.60	0.14	0.24	0.10	0.20	0.18	0.24
Qwen3-VL-4B [4]	0.68	0.16	0.38	0.10	0.26	0.20	0.30
LoHo-Manip-4B	**0.78**	**0.32**	**0.46**	**0.20**	**0.28**	**0.26**	**0.38**

Table 3: Performance comparison on EmbodiedBench [64]. LoHo-Manip substantially outperforms baselines, especially on the Complex configuration.

3.3 VLA in Simulation

We evaluate our method across two simulation benchmarks: LIBERO [42] and VLABench [67], which provide a comprehensive assessment of both foundational manipulation and complex reasoning.

VLABench is specifically designed to challenge the reasoning and generalization capabilities of VLA models, with a primary focus on long-horizon tasks. The benchmark features scenarios where instructions are often implicitly specified and visual observations deviate significantly from the training distribution. This setup necessitates that agents rely on semantic reasoning and robust visual grounding rather than simple memorization of action patterns. In contrast, while LIBERO focuses on rather shorter horizons, it serves as a widely adopted standard for evaluating the atomic capabilities and base motor skills of VLA models. By evaluating on both, we demonstrate our framework's versatility across different task complexities.

We report quantitative results in Tab. 5 and Tab. 4. Our method consistently outperforms prior approaches across all task suites. For VLABench, we further analyze performance using the Intention Score (IS) and Progress Score (PS), as shown in Fig. 7. The Progress Score is particularly meaningful for evaluating long-horizon success, as it captures the agent's ability to complete intermediate steps. Our superior performance on these

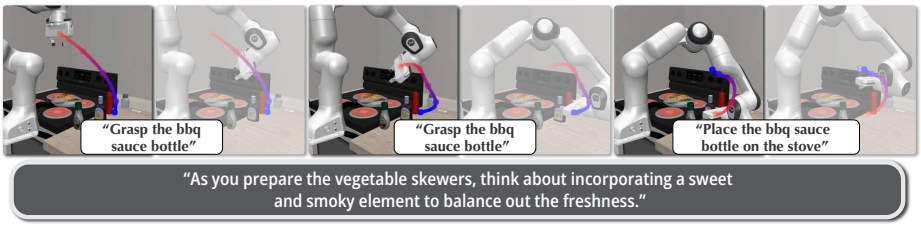

Fig. 5: Results on the semantic instruction track of VLABench. The task involves high-level abstract instructions that require reasoning to infer the appropriate manipulation actions. We visualize both the predicted subtasks and the corresponding trajectory generated by our method.

Methods	VLABench					
	In Distribution ↑	Cross Category ↑	Common Sense ↑	Semantic Instr. ↑	Unseen Texture ↑	**Average ↑**
π_0-fast [51]	0.29	0.18	0.21	0.20	0.24	0.22
$\pi_{0.5}$ [31]	0.37	0.22	0.21	0.17	0.25	0.24
LoHo-Manip	**0.54**	**0.23**	**0.36**	**0.42**	**0.39**	**0.39**

Table 4: Performance comparisons on VLABench [67]. LoHo-Manip consistently outperform our base VLA $\pi_{0.5}$.

metrics demonstrates the effectiveness of our hierarchical framework in successfully bridging high-level reasoning with precise low-level control for complex manipulation.

To further investigate the reasoning process, we provide qualitative results on VLABench in Fig. 5. We visualize the agent's ability to decompose complex, implicit instructions—such as "*As you prepare the vegetable skewers...*"—into a coherent sequence of executable atomic commands. These visualizations highlight how our method maintains consistent visual grounding throughout long-horizon executions, successfully navigating significant visual shifts and distractor objects that often cause baseline models to fail.

3.4 VLA in Real-World

To evaluate the effectiveness of LoHo-Manip in physical environments, we conduct experiments on a real-world robot setup. Our hardware configuration consists of a Franka arm equipped with two Intel RealSense cameras: one providing a static top-view of the workspace and another mounted on the gripper for wrist-view observations. We collected 100 demonstration trajectories via teleoperation [62], which were subsequently processed through our automated data pipeline (Sec. 2) to fine-tune both the high-level task manager and the low-level action executor. The evaluation is categorized into single-step and multi-step tasks. Single-step tasks focus on the precision and grounding of a single atomic command, while multi-step tasks require the task manager to maintain temporal consistency and correct sequencing over long horizons. In Fig. 6, we provide qualitative visualizations of two multi-step samples, demonstrating how our method successfully executes a chain of sub-goals to complete long-horizon manipulation sequences.

Methods	LIBERO				
	Spatial ↑	Object ↑	Goal ↑	Long ↑	**Average ↑**
Octo-Base [48]	78.9	85.7	84.6	51.1	75.1
OpenVLA [36]	84.7	88.4	79.2	53.7	76.5
DiT-Policy [17]	82.6	84.7	82.1	57.6	76.8
TraceVLA [69]	84.6	85.2	75.1	54.1	74.8
CoT-VLA [68]	87.5	91.6	87.6	69.0	83.9
SpatialVLA [53]	88.2	89.9	78.6	55.5	78.1
Nora-AC [30]	85.6	89.4	80.0	63.0	79.5
WorldVLA [10]	87.6	96.2	83.4	60.0	79.1
ThinkAct [27]	88.3	91.4	87.1	70.9	84.4
π_0-fast [51]	96.4	96.8	88.6	60.2	85.5
GR00T-N1.5 [47]	92.0	92.0	86.0	76.0	86.5
MolmoAct [38]	87.0	95.4	87.6	77.2	86.6
StarVLA [15]	97.8	98.6	96.2	93.8	96.6
LoHo-Manip	**98.0**	**98.6**	**98.0**	**95.2**	**97.5**

Table 5: Performance on LIBERO Benchmark. Our method achieves the highest average score across all four tracks.

We specifically focus our evaluation on Out-of-Distribution (OOD) scenarios to test the robustness of the decoupled architecture. We define OOD conditions based on the task complexity:

- Single-step OOD: Evaluated using novel object categories that were not present in the fine-tuning demonstrations, testing the model's zero-shot semantic grounding.
- Multi-step OOD: Evaluated through novel spatial arrangements and unseen language combinations. This requires the task manager to generalize its reasoning to new environmental layouts and instruction structures while maintaining the correct execution logic.

As shown in Fig. 7, our method significantly outperforms our base VLA model, the $\pi_{0.5}$ baseline, which was also fine-tuned on the same 100-sample dataset. The results demonstrate that decoupling high-level semantic planning from low-level execution allows the robot to maintain a higher success rate in OOD settings. In contrast, the baseline model often fails to localize the correct objects or correctly sequence the necessary sub-tasks when faced with novel visual and linguistic arrangements.

4 Related Work

4.1 Robot Planning with Foundation Models

Robot planning is traditionally formulated as coupling a discrete task structure with continuous feasibility, as in task-and-motion planning (TAMP) [23, 35]. Recently, foundation models have been used to instantiate the high-level planning layer for open-ended instructions, including grounded skill selection [2], embodied multimodal planners [19], and closed-loop feedback with language-state updates [21, 29]. Complementary efforts provide planning-oriented resources, including benchmarks for multimodal planning and long-horizon reasoning [52, 56] and large-scale embodied data/model engines [33, 59]. A complementary direction treats LLMs as program synthesizers that generate executable control

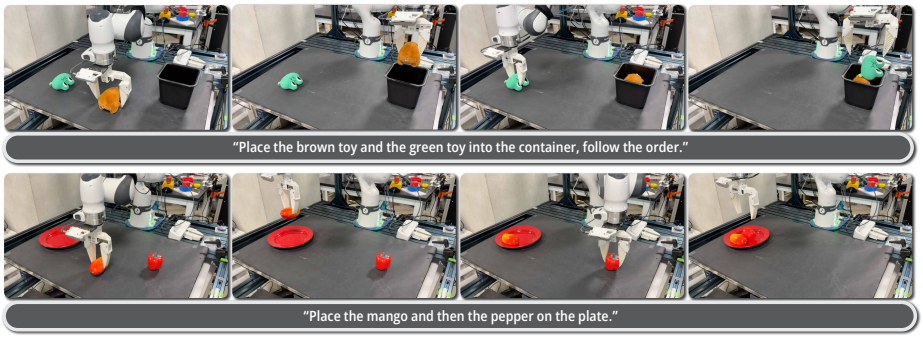

Fig. 6: Robot execution results on multi-step tasks. We evaluate performance under both in-distribution and out-of-distribution settings, including scenarios with unseen objects and novel subtask combinations.

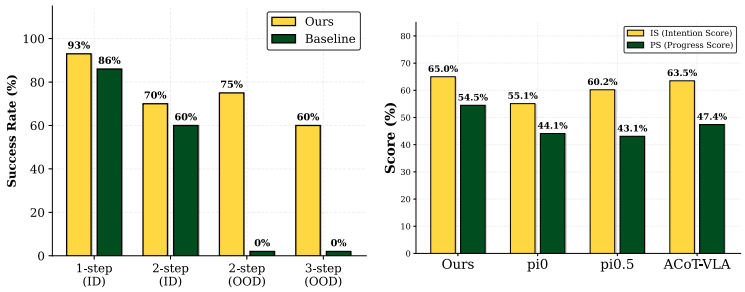

Fig. 7: (Left) **Real robot results**. LoHo-Manip significantly outperforms $\pi_{0.5}$ (our base VLA finetuned with the same data) in out-of-distribution settings involving novel instructions and objects, demonstrating superior generalization. (Right) **IS and PS comparison on VLABench**. LoHo-Manip consistently exceeds baselines in metrics requiring robust vision-language reasoning, moving beyond simple success rates.

logic and API calls [3, 41, 57]. In parallel, agentic multimodal foundation models study general-purpose interactive decision making for embodied agents [63]. We also note structured instantiations that couple LLM planning with explicit world models for scalability, e.g., 3D scene-graph grounded planning in SayPlan [54].

4.2 Long-Horizon Manipulation and Generalist VLA

Long-horizon manipulation requires maintaining task state over many steps, handling subtask dependencies, and recovering from compounding execution errors. Generalist VLA foundations provide strong short-horizon control and broad transfer [6–9, 31, 36, 50, 51, 71], but long-horizon performance is often limited by drift and partial observability. Therefore, recent work targets long-horizon capability more directly, e.g., by introducing long-horizon training/evaluation and stronger execution-time correction [20, 26, 27, 43, 68], or by integrating planning-style intermediate structure into a unified VLA loop [25, 61]. Another key ingredient is *memory* for progress tracking under partial observability: RoboVQA

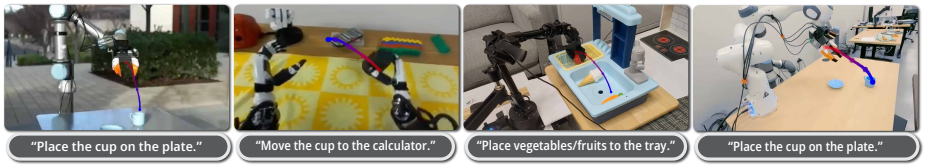

Fig. 8: Qualitative results of trajectory prediction on out-of-distribution embodiments. Our high-level task manager generalizes across diverse robotic setups and environments, including different manipulators, objects, and scenes. By decoupling high-level task planning from low-level control, the manager can seamlessly plug into downstream executor policies trained on different embodiments and datasets.

and MindExplore study multimodal long-horizon reasoning with explicit mechanisms to integrate observations over time [39, 56], and concurrent with our work, MEM equips VLAs with multi-scale memory via short-term visual context and longer-horizon language memory [60]. Our work contributes a complementary axis: rather than solely scaling an end-to-end VLA or embedding all reasoning within it, we separate long-horizon task management from short-horizon control and train the manager to predict *remaining* plans from the current observation. This receding-horizon "what remains" formulation provides a stable progress signal for closed-loop execution, and interfaces naturally with trace-conditioned policies for local control.

4.3 Visual Prompting and Trajectory Representations

Several lines of work bridge high-level intent to low-level policies by introducing explicit intermediate guidance. VIMA formulates manipulation as prompt-conditioned behavior via interleaved visual and textual tokens [34]. Planning-oriented systems also use intermediate representations (e.g., code, value maps, or waypoints) to connect semantic decisions to short-horizon execution [27, 28, 41]. More directly, TraceVLA encodes trajectories as a visual trace prompt to enhance spatial-temporal awareness for generalist robotic policies [69]. Our approach follows this theme by using a lightweight *visual trace* as a direct conditioning signal, reducing long-horizon manipulation into repeated short-horizon, locally-conditioned execution.

5 Conclusion

We present LoHo-Manip, a hierarchical framework for long-horizon manipulation that decouples high-level symbolic task planning from low-level sensorimotor execution. By predicting a joint distribution of subtask sequences and canonical visual traces, our approach reformulates complex long-horizon reasoning into a series of manageable, short-horizon control problems. Extensive experiments across embodied QA, simulation, and real-world benchmarks demonstrate that LoHo-Manip significantly improves success rates, generalization, and task-planning accuracy compared to monolithic VLA baselines. Our architecture provides a scalable pathway for integrating future advancements in both large-scale VLMs and high-frequency robot control policies.

Supplementary Material

A More Qualitative Results on VLABench

We provide additional qualitative results of our method on VLABench [67]. The VLABench contains five evaluation tracks, each consisting of 50 tasks with different scenes or instructions.

- **In-distribution**. Evaluates the policy's ability to learn tasks from in-domain episodes with a small and diverse dataset.
- **Cross-category**. Evaluates generalization across different object categories and instances.
- **Common-sense**. Evaluates the ability to apply common-sense reasoning to identify the target object.
- **Semantic-instruction**. Evaluates the understanding of instructions with complex semantic or contextual information.
- **Unseen-texture**. Evaluates robustness under changes in background and table textures.

Results from Fig. 9 demonstrate that our method generalizes well to both in-distribution and out-of-distribution scenes and tasks. For the common-sense and semantic-instruction tracks, which involve open-ended instructions, our method shows strong reasoning and generalization ability. For the unseen-texture track with varying textures, our method remains robust and continues to generate reasonable sub-tasks and actions.

B More Qualitative Results on Real Evaluation

We provide additional qualitative results from real robot experiments. The system takes a top-down camera view, a wrist-mounted camera view, and a high-level instruction as input. The task manager first predicts the next sub-task together with a corresponding visual trace. The VLA executor then generates low-level robot actions conditioned on these signals to perform the manipulation.

The results demonstrate that our framework can reliably decompose high-level instructions into executable sub-tasks and produce spatial traces that guide the robot toward the correct interaction targets. The predicted traces provide an intuitive intermediate representation that improves the stability of execution and helps the robot recover from minor deviations during long-horizon manipulation. Across different scenes and object configurations, the system maintains consistent task progress and successfully completes multi-step manipulation sequences.

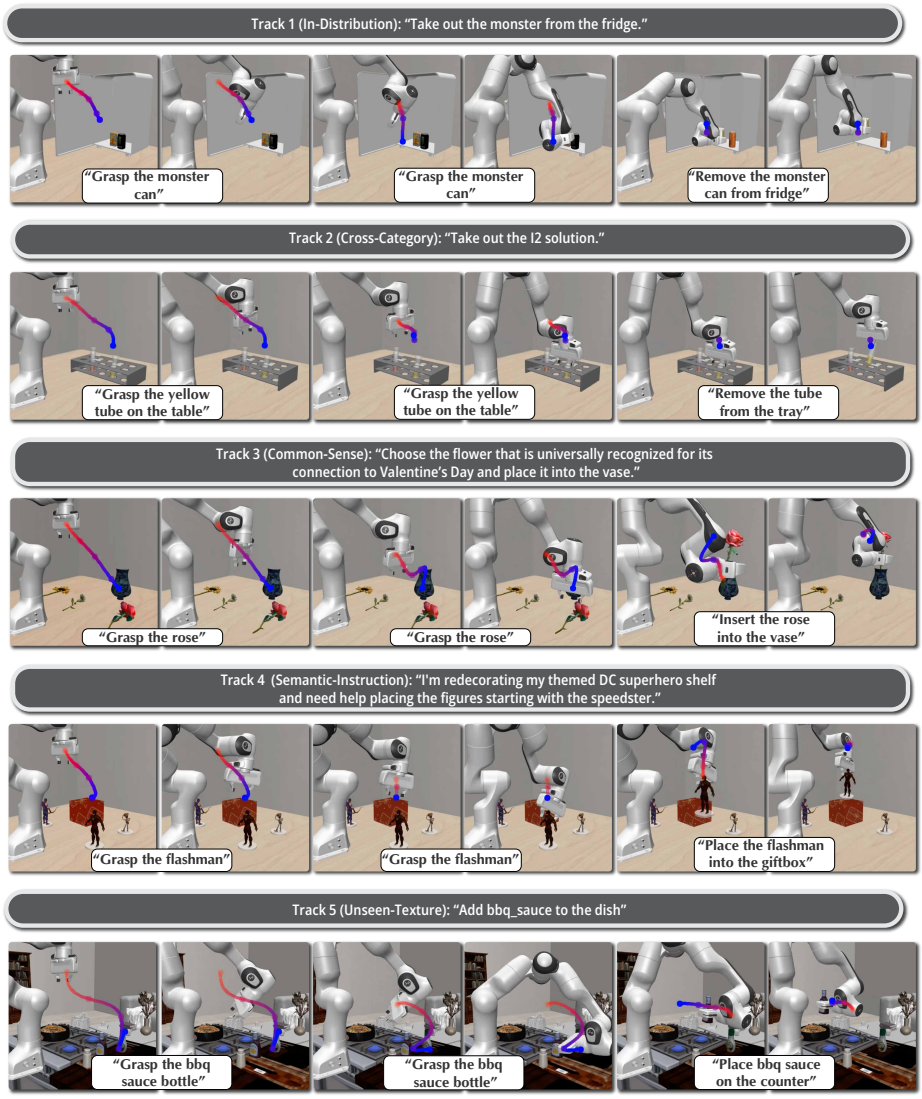

Fig. 9: Results on different tracks of VLABench. We show the predicted sub-tasks and their corresponding trajectory.

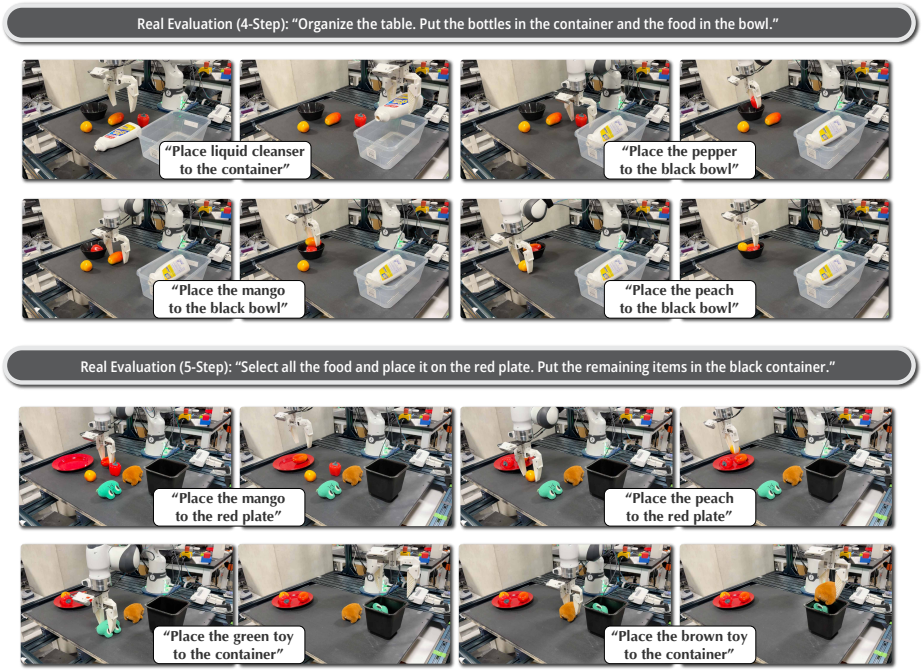

Fig. 10: Additional qualitative results from real robot experiments. We evaluate our model on different long-horizon manipulation tasks.

C More Ablation Study

C.1 Quality of Sub-Task and Trace Curation

We study the effectiveness of the proposed sub-task and trace construction in our data pipeline for generating training data from real robot demonstrations. We compare the performance before and after incorporating our curated data on trajectory prediction benchmarks on ShareRobot-T [33] and VABench-V [65] across three key metrics: Discrete Fréchet Distance (DFD), Hausdorff Distance (HD), and Root Mean Square Error (RMSE).
The results in Tab. 6 demonstrate that our method greatly improved the model's trajectory prediction ability on both benchmarks, validating the effectiveness of our curated sub-task and trajectory data.

C.2 Integration of Different VLA Executors

Since our framework adapts a modular design that decouples the high-level task manager from the low-level VLA executor. The task manager predicts progress-aware sub-tasks together with a spatial trace, which serves as a general interface that can be followed by different VLA policies. This design allows the same task manager to be reused with various executor architectures without modifying the planning component.

Methods	ShareRobot-T			VABench-V		
	DFD ↓	HD ↓	RMSE ↓	DFD ↓	HD ↓	RMSE ↓
W./o. subtask traj.	0.2437	0.2149	0.1628	0.2500	0.1984	0.1686
Ours	0.2309	0.2058	0.1559	0.2123	0.1821	0.1469

Table 6: Ablation on trajectory prediction benchmarks before and after incorporating our curated sub-task and trace data. Our data curation pipeline consistently improves trajectory prediction performance on both ShareRobot-T and VABench-V across all metrics (DFD, HD, RMSE).

To validate this property, we integrate our task manager with an alternative VLA model, StarVLA [15], replacing the default $\pi_{0.5}$ executor used in our main experiments. The predicted sub-tasks and visual traces from the manager are provided to the StarVLA policy as execution guidance, enabling the full pipeline to operate in the same closed-loop manner. We evaluate this configuration on the VLABench benchmark and report the results below in Tab. 7. The results show that our task manager can effectively interface with different VLA executors, demonstrating the flexibility and extensibility of our framework.

Methods	VLABench					
	In Distribution ↑	Cross Category ↑	Common Sense ↑	Semantic Instr. ↑	Unseen Texture ↑	**Average ↑**
StarVLA	0.30	0.08	0.19	0.15	0.18	0.18
StarVLA + ours	0.42	0.12	0.18	0.21	0.25	0.24

Table 7: Performance comparison on VLABench when integrating our task manager with different VLA executors. By providing sub-task and trace guidance, our framework consistently improves the performance of the StarVLA policy across all evaluation tracks.

C.3 Inference Efficiency

Our framework maintains efficient inference through a hierarchical execution scheme. In practice, the task manager (planner) is invoked once every 100 steps of the low-level VLA executor, while the executor continuously produces robot actions conditioned on the predicted sub-task and visual trace. Since the planner runs at a much lower frequency than the executor, the additional computational overhead is minimal, keeping the overall inference time comparable to using the executor alone.

We compare the inference latency in the same simulation task before and after integrating the task manager, as shown in Tab. 8.

In real robot deployment, on a machine equipped with an NVIDIA A6000 GPU, the planner runs at approximately 2 Hz, while the low-level executor operates at about 10 Hz. This scheduling allows the system to per-

Configuration	Latency (1 episode) (↓)
With task manager	∼86s
Without task manager	∼72s

Table 8: Inference latency comparison for a single episode rollout in simulation.

form high-level reasoning intermittently while maintaining fast low-level control during long-horizon manipulation.

D Details for Sub-Task and Trace Extraction

To generate the visual trace supervision described in Sec. 2.2, we extract the robot's end-effector trajectory from demonstration videos using a Vision-Language Model (VLM).
This process converts raw video frames into a sequence of normalized 2D pixel coordinates. For each frame, we query the VLM to localize the Franka robot's end-effector. The prompt defines the target specifically as the gripper and wrist assembly, excluding upstream arm links, to ensure spatial consistency. The model is constrained to a strict JSON output format, returning a bounding box for the detected region. We calculate the center point of this box to determine the robot's location.
The resulting coordinate sequence is normalized to a [0, 1000] range and resampled to create the compact visual trace representation used for training. Frames where the model fails to provide a valid detection are discarded. The prompting template used for this process is detailed in Tab. 9.

E Limitations

While our method demonstrates strong performance across multiple benchmarks, several aspects remain open for further improvement. First, our approach relies on accurate sub-task grounding and trace prediction from the task manager. Although the receding-horizon design allows the system to adapt when execution deviates from the plan, improvements in perception and spatial grounding could further enhance the stability of the overall pipeline.
Second, the visual trace representation provides a simple and effective interface between high-level reasoning and low-level control. However, it currently captures spatial intent using a 2D trajectory, which may not fully express more complex interaction patterns such as highly precise manipulation or contact-rich behaviors. Incorporating richer spatial or temporal guidance signals could further extend the applicability of this representation.
Finally, our experiments primarily focus on tabletop manipulation scenarios with a single robotic arm. While the modular design of our framework allows the task manager to interface with different executors, evaluating the approach on a broader range of embodiments, tasks, and dynamic environments would be an interesting direction for future work.

Data Type	Prompt Template
Robot Trace Detection	You are an expert in robotic vision. Locate the Franka robot's end-effector in this top-down image. ### Target Definition Detect the entire end-effector assembly only. This includes the wrist (hand) and the grippers/fingers. Do NOT include the forearm (Link 7) or any part of the articulated arm leading up to the wrist. ### Strict Output Format Return a valid JSON array of exactly one object, or an empty array [] if the end-effector is not visible. Format: [{"bbox_2d": [xmin, ymin, xmax, ymax], "label": "robot_end_effector"}] ### Rules - Coordinates must be integers in pixel units (Origin: top-left). - The box must be the tightest possible fit around the grippers and wrist housing. - Output ONLY the JSON. No markdown code blocks, no preamble, no commentary.
Sub-task Decomposition	You are an expert in robot action analysis. Your task is to: 1. Decompose the robot's behavior into atomic sub-tasks that directly fulfill the instruction: '{language}'. ### Step 1: Identify Atomic Sub-Tasks - List only atomic sub-tasks that involve a physical interaction changing an object's contact or state (e.g., grasp, place, insert, push, cut). - Exclude: motion-only actions (e.g., move, reach, approach), preparatory phrases (e.g., prepare to...), or non-interactive steps (e.g., release, hover). - Each sub-task must correspond to one clear change in object state or contact. - For each sub-task, specify the frame number at which the action is fully completed. ### Step 2: Output Format Return a valid JSON object with following keys: - 'subtasks': a dictionary mapping each sub-task (string) and its completion frame (as a stringified integer). ### Example Output { "subtasks": { "10": "Grasp the knife", "25": "Place the knife on the cutting board" } }

Table 9: Prompt templates used for extracting robot end-effector traces and decomposing demonstrations into atomic sub-tasks. A vision-language model is prompted to localize the robot gripper in each frame and to segment manipulation sequences into interaction primitives, producing the trace and sub-task supervision used for training.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al.: Do as i can, not as i say: Grounding language in robotic affordances. In: CoRL (2022)
3. Ahn, S., Choi, W., Lee, J., Park, J., Woo, H.: Towards reliable code-as-policies: A neuro-symbolic framework for embodied task planning. In: NeurIPS (2025)
4. Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., Deng, L., Ding, W., Gao, C., Ge, C., et al.: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025)
5. Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H., Lin, J.: Qwen2.5-vl technical report. arXiv preprint arXiv:2502.13923 (2025)
6. Barreiros, J., Beaulieu, A., Bhat, A., Cory, R., Cousineau, E., Dai, H., Fang, C.H., Hashimoto, K., Irshad, M.Z., Itkina, M., et al.: A careful examination of large behavior models for multitask dexterous manipulation. arXiv preprint arXiv:2507.05331 (2025)
7. Bjorck, J., Castañeda, F., Cherniadev, N., Da, X., Ding, R., Fan, L., Fang, Y., Fox, D., Hu, F., Huang, S., et al.: Gr00t n1: An open foundation model for generalist humanoid robots. arXiv preprint arXiv:2503.14734 (2025)
8. Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., et al.: π_0: A vision-language-action flow model for general robot control. In: RSS (2025)
9. Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al.: Rt-1: Robotics transformer for real-world control at scale. In: RSS (2023)
10. Cen, J., Yu, C., Yuan, H., Jiang, Y., Huang, S., Guo, J., Li, X., Song, Y., Luo, H., Wang, F., Zhao, D., Chen, H.: Worldvla: Towards autoregressive action world model. arXiv preprint arXiv: (2025)
11. Chen, Z., Wang, W., Cao, Y., Liu, Y., Gao, Z., Cui, E., Zhu, J., Ye, S., Tian, H., Liu, Z., et al.: Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. arXiv preprint arXiv:2412.05271 (2024)
12. Cheng, A.C., Ji, Y., Yang, Z., Gongye, Z., Zou, X., Kautz, J., Bıyık, E., Yin, H., Liu, S., Wang, X.: Navila: Legged robot vision-language-action model for navigation. In: RSS (2025)
13. Chi, C., Xu, Z., Feng, S., Cousineau, E., Du, Y., Burchfiel, B., Tedrake, R., Song, S.: Diffusion policy: Visuomotor policy learning via action diffusion. In: RSS (2023)
14. Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., 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)
15. Community, S.: Starvla: A lego-like codebase for vision-language-action model developing. arXiv preprint arXiv:2604.05014 (2026)
16. Dang, R., Guo, J., Hou, B., Leng, S., Li, K., Li, X., Liu, J., Mao, Y., Wang, Z., Yuan, Y., et al.: Rynnbrain: Open embodied foundation models. arXiv preprint arXiv:2602.14979 (2026)
17. Dasari, S., Mees, O., Zhao, S., Srirama, M.K., Levine, S.: The ingredients for robotic diffusion transformers. In: ICRA (2025)
18. Devin, C., Gupta, A., Darrell, T., Abbeel, P., Levine, S.: Learning modular neural network policies for multi-task and multi-robot transfer. In: ICRA (2017)
19. Driess, D., Xia, F., Sajjadi, M.S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al.: Palm-e: An embodied multimodal language model. In: ICML (2023)
20. Fan, Y., Ding, P., Bai, S., Tong, X., Zhu, Y., Lu, H., Dai, F., Zhao, W., Liu, Y., Huang, S., et al.: Long-vla: Unleashing long-horizon capability of vision language action model for robot manipulation. In: CoRL (2025)
21. Fang, H., Zhang, M., Dong, H., Li, W., Wang, Z., Zhang, Q., Tian, X., Hu, Y., Li, H.: Robix: A unified model for robot interaction, reasoning and planning. arXiv preprint arXiv:2509.01106 (2025)
22. Florence, P., Lynch, C., Zeng, A., Ramirez, O.A., Wahid, A., Downs, L., Wong, A., Lee, J., Mordatch, I., Tompson, J.: Implicit behavioral cloning. In: CoRL (2021)
23. Garrett, C.R., Chitnis, R., Holladay, R., Kim, B., Silver, T., Kaelbling, L.P., Lozano-Pérez, T.: Integrated task and motion planning. Annual Review of Control, Robotics, and Autonomous Systems (2021)
24. Google DeepMind: Gemini 3 Flash: frontier intelligence built for speed. https://deepmind.google/models/gemini/flash (2025)
25. Hu, Y., Zhang, J., Luo, Y., Guo, Y., Chen, X., Sun, X., Feng, K., Lu, Q., Chen, S., Zhang, Y., Li, W., Chen, J.: Bagelvla: Enhancing long-horizon manipulation via interleaved vision-language-action generation. arXiv preprint arXiv:2602.09849 (2026)
26. Huang, C.P., Man, Y., Yu, Z., Chen, M.H., Kautz, J., Wang, Y.C.F., Yang, F.E.: Fast-thinkact: Efficient vision-language-action reasoning via verbalizable latent planning. In: CVPR (2026)
27. Huang, C.P., Wu, Y.H., Chen, M.H., Wang, Y.C.F., Yang, F.E.: Thinkact: Vision-language-action reasoning via reinforced visual latent planning. In: NeurIPS (2025)
28. Huang, W., Wang, C., Zhang, R., Li, Y., Wu, J., Fei-Fei, L.: Voxposer: Composable 3d value maps for robotic manipulation with language models. In: CoRL (2023)
29. Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al.: Inner monologue: Embodied reasoning through planning with language models. arXiv preprint arXiv:2207.05608 (2022)

30. Hung, C.Y., Sun, Q., Hong, P., Zadeh, A., Li, C., Tan, U., Majumder, N., Poria, S., et al.: Nora: A small open-sourced generalist vision language action model for embodied tasks. arXiv preprint arXiv:2504.19854 (2025)
31. Intelligence, P., Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al.: $\pi_{0.5}$: A vision-language-action model with open-world generalization. arXiv preprint arXiv:2504.16054 (2025)
32. Janner, M., Du, Y., Tenenbaum, J.B., Levine, S.: Planning with diffusion for flexible behavior synthesis. In: ICML (2022)
33. Ji, Y., Tan, H., Shi, J., Hao, X., Zhang, Y., Zhang, H., Wang, P., Zhao, M., Mu, Y., An, P., et al.: Robobrain: A unified brain model for robotic manipulation from abstract to concrete. In: CVPR (2025)
34. Jiang, Y., Gupta, A., Zhang, Z., Wang, G., Dou, Y., Chen, Y., Fei-Fei, L., Anandkumar, A., Zhu, Y., Fan, L.: Vima: Robot manipulation with multimodal prompts. In: ICML (2023)
35. Kaelbling, L.P., Lozano-Pérez, T.: Hierarchical task and motion planning in the now. Tech. rep. (2010), https://dspace.mit.edu/handle/1721.1/54780
36. Kim, M.J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P., et al.: Openvla: An open-source vision-language-action model. In: CoRL (2024)
37. Kumar, A., Fu, Z., Pathak, D., Malik, J.: Rma: Rapid motor adaptation for legged robots. In: RSS (2021)
38. Lee, J., Duan, J., Fang, H., Deng, Y., Liu, S., Li, B., Fang, B., Zhang, J., Wang, Y.R., Lee, S., et al.: Molmoact: Action reasoning models that can reason in space. In: ICRA (2026)
39. Li, D., Zhang, Y., Cao, M., Liu, D., Xie, W., Hui, T., Lin, L., Xie, Z., Li, Y.: Towards long-horizon vision-language-action system: Reasoning, acting and memory. In: ICCV (2025)
40. Li, Y., Deng, Y., Zhang, J., Jang, J., Memmel, M., Yu, R., Garrett, C.R., Ramos, F., Fox, D., Li, A., et al.: Hamster: Hierarchical action models for open-world robot manipulation. In: ICLR (2025)
41. Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., Zeng, A.: Code as policies: Language model programs for embodied control. In: ICRA (2023)
42. Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., Stone, P.: Libero: Benchmarking knowledge transfer for lifelong robot learning. In: NeurIPS (2023)
43. Liu, Y., Hamid, J.I., Xie, A., Lee, Y., Du, M., Finn, C.: Bidirectional decoding: Improving action chunking via guided test-time sampling. In: ICLR (2025)
44. Liu, Z., Zhu, L., Shi, B., Zhang, Z., Lou, Y., Yang, S., Xi, H., Cao, S., Gu, Y., Li, D., et al.: Nvila: Efficient frontier visual language models. In: CVPR (2025)
45. McArthur, M., Moshfeghi, Y., Cashmore, M.: Egoplan: A framework for multi-agent planning using single agent planners. In: The International FLAIRS Conference Proceedings (2022)
46. Meta AI: Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices (2024)

47. NVIDIA Research: GR00T N1.5: An Improved Open Foundation Model for Generalist Humanoid Robots. https://research.nvidia.com/labs/gear/gr00t-n1_5 (2025)
48. Octo Model Team, Ghosh, D., Walke, H., Pertsch, K., Black, K., Mees, O., Dasari, S., Hejna, J., Xu, C., Luo, J., Kreiman, T., Tan, Y., Sanketi, P., Vuong, Q., Xiao, T., Sadigh, D., Finn, C., Levine, S.: Octo: An open-source generalist robot policy. In: RSS (2024)
49. OpenAI: GPT-4o Mini: Advancing Cost-Efficient Intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence (2024)
50. O'Neill, A., Rehman, A., Maddukuri, A., Gupta, A., Padalkar, A., Lee, A., Pooley, A., Gupta, A., Mandlekar, A., Jain, A., et al.: Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In: ICRA (2024)
51. Pertsch, K., Stachowicz, K., Ichter, B., Driess, D., Nair, S., Vuong, Q., Mees, O., Finn, C., Levine, S.: Fast: Efficient action tokenization for vision-language-action models. In: RSS (2025)
52. Qiu, L., Chen, Y., Ge, Y., Ge, Y., Shan, Y., Liu, X.: Egoplan-bench2: A benchmark for multimodal large language model planning in real-world scenarios. arXiv preprint arXiv:2412.04447 (2024)
53. Qu, D., Song, H., Chen, Q., Yao, Y., Ye, X., Ding, Y., Wang, Z., Gu, J., Zhao, B., Wang, D., et al.: Spatialvla: Exploring spatial representations for visual-language-action model. In: RSS (2025)
54. Rana, K., Haviland, J., Garg, S., Abou-Chakra, J., Reid, I., Suenderhauf, N.: Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. arXiv preprint arXiv:2307.06135 (2023)
55. Ross, S., Gordon, G., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics. pp. 627–635. JMLR Workshop and Conference Proceedings (2011)
56. Sermanet, P., Ding, T., Zhao, J., Xia, F., Dwibedi, D., Gopalakrishnan, K., Chan, C., Dulac-Arnold, G., Maddineni, S., Joshi, N.J., et al.: Robovqa: Multimodal long-horizon reasoning for robotics. In: ICRA (2024)
57. Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., Garg, A.: Progprompt: Generating situated robot task plans using large language models. In: ICRA (2023)
58. Tan, H., Zhou, E., Li, Z., Xu, Y., Ji, Y., Chen, X., Chi, C., Wang, P., Jia, H., Ao, Y., et al.: Robobrain 2.5: Depth in sight, time in mind. arXiv preprint arXiv:2601.14352 (2026)
59. Team, B.R.: Robobrain 2.0 technical report. arXiv preprint arXiv:2507.02029 (2025)
60. Torne, M., Pertsch, K., Walke, H., Vedder, K., Nair, S., Ichter, B., Ren, A.Z., Wang, H., Tang, J., Stachowicz, K., et al.: Mem: Multi-scale embodied memory for vision language action models. https://www.pi.website/download/Mem.pdf (2026)
61. Wu, W., Lu, F., Wang, Y., Yang, S., Liu, S., Wang, F., Zhu, Q., Sun, H., Wang, Y., Ma, S., Ren, Y., Zhang, K., Yu, H., Zhao, J., Zhou, S.,

Qiu, Z., Xiong, H., Wang, Z., Wang, Z., Cheng, R., Li, Y., Huang, Y., Zhu, X., Shen, Y., Zheng, K.: A pragmatic vla foundation model. arXiv preprint arXiv:2601.18692 (2026)
62. Yan, R., Fu, J., Yang, S., Paulsen, L., Cheng, X., Wang, X.: Ace-f: A cross embodiment foldable system with force feedback for dexterous teleoperation. arXiv preprint arXiv:2511.20887 (2025)
63. Yang, J., Tan, R., Wu, Q., Zheng, R., Peng, B., Liang, Y., Gu, Y., Cai, M., Ye, S., Jang, J., et al.: Magma: A foundation model for multimodal ai agents. In: CVPR (2025)
64. Yang, R., Chen, H., Zhang, J., Zhao, M., Qian, C., Wang, K., Wang, Q., Koripella, T.V., Movahedi, M., Li, M., et al.: Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In: ICML (2025)
65. Yuan, Y., Cui, H., Chen, Y., Dong, Z., Ni, F., Kou, L., Liu, J., Li, P., Zheng, Y., Hao, J.: From seeing to doing: Bridging reasoning and decision for robotic manipulation. In: ICLR (2026)
66. Yuan, Y., Cui, H., Huang, Y., Chen, Y., Ni, F., Dong, Z., Li, P., Zheng, Y., Hao, J.: Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. In: ICLR (2026)
67. Zhang, S., Xu, Z., Liu, P., Yu, X., Li, Y., Gao, Q., Fei, Z., Yin, Z., Wu, Z., Jiang, Y.G., et al.: Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In: ICCV (2025)
68. Zhao, Q., Lu, Y., Kim, M.J., Fu, Z., Zhang, Z., Wu, Y., Li, Z., Ma, Q., Han, S., Finn, C., et al.: Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In: CVPR (2025)
69. Zheng, R., Liang, Y., Huang, S., Gao, J., III, H.D., Kolobov, A., Huang, F., Yang, J.: TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In: ICLR (2025)
70. Zhu, J., Wang, W., Chen, Z., Liu, Z., Ye, S., Gu, L., Tian, H., Duan, Y., Su, W., Shao, J., et al.: Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. arXiv preprint arXiv:2504.10479 (2025)
71. Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., et al.: Rt-2: Vision-language-action models transfer web knowledge to robotic control. In: CoRL (2023)