
LoHoVLA: A Unified Vision-Language-Action Model for Long-Horizon Embodied Tasks

Yi Yang

Fudan University
21307130076@m.fudan.edu.cn

Jiaxuan Sun

ShanghaiTech University
sunjx2022@shanghaitech.edu.cn

Siqi Kou

Shanghai Jiao Tong University
happy-karry@sjtu.edu.cn

Yihan Wang

Fudan University
23301170011@m.fudan.edu.cn

Zhijie Deng*

Shanghai Jiao Tong University
zhijied@sjtu.edu.cn

Abstract

Real-world embodied agents face long-horizon tasks, characterized by high-level goals demanding multi-step solutions beyond single actions. Successfully navigating these requires both high-level task planning (i.e., decomposing goals into sub-tasks) and low-level motion control (i.e., generating precise robot actions). While existing vision language action (VLA) models and hierarchical architectures offer potential in embodied tasks, the former often falter in planning, and the latter can suffer from coordination issues, both hampering performance. We introduce a new unified VLA framework for long-horizon tasks, dubbed LoHoVLA, to overcome these limitations. LoHoVLA leverages a large pretrained vision language model (VLM) as the backbone to jointly generate language and action tokens for sub-task generation and robot action prediction, respectively. This shared representation promotes better generalization across tasks. Additionally, LoHoVLA embraces a hierarchical closed-loop control mechanism to mitigate errors originating from both high-level planning and low-level control. To train LoHoVLA, we introduce LoHoSet, a dataset built on the Ravens simulator, containing 20 long-horizon tasks, each with 1,000 expert demonstrations composed of visual observations, linguistic goals, sub-tasks, and robot actions. Experimental results show that LoHoVLA significantly surpasses both hierarchical and standard VLA approaches on long-horizon embodied tasks in the Ravens simulator. These findings underscore the promise of unified architectures for advancing generalizable embodied intelligence.

1 Introduction

Embodied agents operating in real world are required to handle tasks that are long-horizon, compositional, and dynamically changing [30, 14, 41, 45]. Unlike short-horizon tasks [47, 21, 49, 38, 55, 22], the long-horizon ones involve high-level goals that cannot be achieved in a single action. Agents must do reasoning, execute movements, and adapt to failures or changes in the environment. This

*Corresponding author.

necessitates both high-level task planning and low-level motion control, with the former decomposing the overall goal into atomic tasks and the latter generating accurate robot actions for execution.

Vision language action (VLA) models have emerged as a dominant paradigm for embodied agents [7, 24, 39]. They usually use large-scale pretrained vision language models (VLMs) as backbones and are fine-tuned on robotic demonstrations to map visual and linguistic inputs to executable robot actions. While VLA models effectively extract key information from observations and instructions, they fall short in effective planning and reasoning, suffering from subpar performance on long-horizon tasks.

Seminal studies on long-horizon embodied tasks usually employ hierarchical architectures [52, 46, 1, 28, 50, 19], including a high-level VLM-based planner to reason about sub-task instructions and a low-level VLA-based controller to convert these instructions into robot actions. While this modular structure provides flexibility, it frequently results in suboptimal coordination and limited generalization [20, 51]. These limitations underscore the need for unified, end-to-end architectures that combine high-level and low-level inference in a single framework.

To bridge the gap, we introduce **LoHoVLA**, a unified **V**ision-**L**anguage-**A**ction model for **L**ong-**H**orizon embodied tasks that integrates both high-level task planning and low-level motion control. LoHoVLA first infers linguistic sub-tasks from the input observations and the specified high-level goal, then uses the inferred sub-tasks as contextual guidance for action prediction. The robot executes the predicted actions to interact with and modify the environment. New observations are subsequently captured to infer the next sub-tasks and the following actions.

We build LoHoVLA upon a large pretrained VLM backbone to leverage its extensive world knowledge and reasoning capabilities [8, 26]. We augment the original language head to generate both linguistic sub-tasks and discrete action tokens. This shared backbone enables learning of generalizable representations across both planning and control. To further enhance robustness, we introduce a hierarchical closed-loop control mechanism: If the execution of a sub-task fails for times exceeding a predefined threshold, the system re-plans the sub-task; otherwise, it updates only the actions based on the new environment state.

Considering the scarcity of long-horizon demonstrations annotated with fine-grained sub-tasks and actions, we synthesize the dataset **LoHoSet** to train LoHoVLA. Specifically, LoHoSet is developed based on the Ravens robot simulator and comprises 20 long-horizon embodied tasks. Each task includes 1,000 expert demonstrations featuring visual observations, linguistic goals, sub-tasks, and actions. The LoHoVLA trained on the dataset can significantly outperform both the hierarchical baseline and a vanilla VLA baseline on both seen and unseen tasks, showing strong reasoning and planning capabilities as well as strong generalization. These findings underscore the potential of unified architectures for real-world embodied intelligence.

2 Related Work

2.1 Vision-Language-Action Models

Vision language action (VLA) models have become a central paradigm in robot learning by bridging perception, language understanding, and control. These models typically repurpose large-scale vision language models (VLMs) [2, 16, 10, 23] pretrained on internet-scale datasets for downstream robotic tasks. Such pretrained VLMs offer rich semantic priors and strong generalization across modalities, making them attractive backbones for visuomotor policies. Recent works [13, 24, 34] fine-tune these backbones on robot demonstration datasets to map image and language inputs to motor actions, showing promising results in generalizing to novel objects, instructions, and environments.

Despite these advances, most existing VLA frameworks are limited in their ability to perform multi-step reasoning or structured task decomposition, which are critical for executing complex or long-horizon tasks. To address this, some approaches introduce external planners [46, 15, 54] or use action experts [44] to handle sub-task planning, though this modularity often incurs coordination overhead and reduces system robustness. Furthermore, while pretrained VLMs offer strong grounding capabilities, their integration into control pipelines remains challenging due to the mismatch between vision-language pretraining objectives and the fine-grained demands of robotic control. Our work contributes to the growing effort to enhance VLA models by exploring architectural designs that support structured reasoning and low-level control within a cohesive framework.

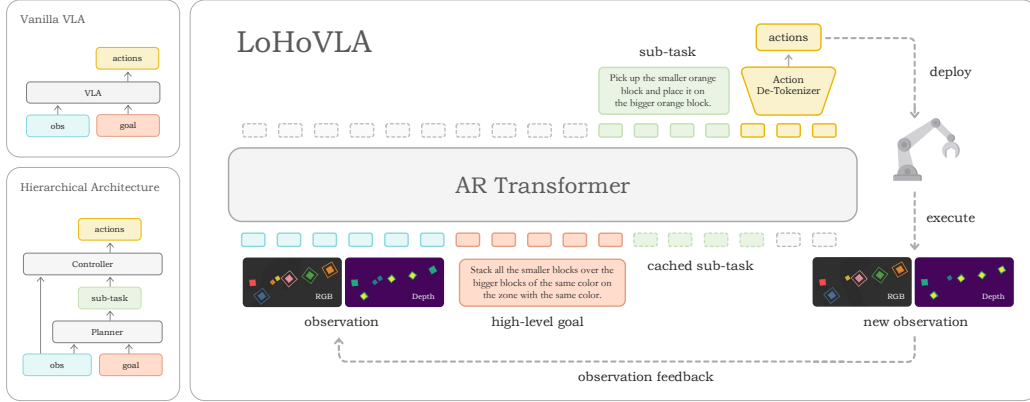


Figure 1: *Left top:* Vanilla VLA directly maps high-level goals and observations to actions. *Left bottom:* The hierarchical architecture separates planning and execution—the planner infers sub-tasks, and the controller executes them. *Right:* LoHoVLA integrates high-level task planning and low-level motion control into a unified model. It uses an auto-regressive (AR) Transformer as its backbone and employs a hierarchical closed-loop control mechanism.

2.2 Long-Horizon Embodied Task Planning

Effective execution of long-horizon tasks in embodied agents necessitates advanced planning capabilities that can decompose high-level goals into coherent sequences of sub-tasks. Recent approaches [1, 4, 9, 11, 12, 18, 25, 27, 33, 37] have leveraged large language models (LLMs) to enhance such planning processes. These methods typically employ LLMs to generate structured plans from natural language instructions, which are then executed by specialized low-level controllers. To improve adaptability, some frameworks [9, 12, 33, 37] incorporate re-planning mechanisms that allow the agent to adjust its strategy in response to environmental changes, thereby enhancing robustness and generalization across diverse tasks and settings. Beyond textual planning, there is a growing interest in integrating chain-of-thought (CoT) reasoning [42] into embodied systems. This paradigm [53, 48, 43] involves generating intermediate reasoning steps that bridge the gap between high-level instructions and low-level actions. By conditioning action prediction on these subgoals, agents can achieve more structured and interpretable behaviors. However, existing approaches often suffer from limitations such as reliance on modular architectures that separate high-level planning and low-level control, leading to suboptimal coordination and limited generalization. Our proposed LoHoVLA model integrates both high-level task planning and low-level motion control within a single unified framework, leveraging a shared vision-language backbone to improve reasoning, planning, and generalization capabilities in long-horizon tasks.

3 LoHoVLA

This section elaborates on LoHoVLA, starting with various modeling configurations, followed by a description of the LoHoSet dataset for training LoHoVLA. We then present the model architecture and outline the training strategies.

3.1 Modeling Configurations

We address the problem of learning a policy π_θ that produces robot actions \mathbf{a}_t based on a visual observation \mathbf{o}_t and a high-level language goal g . The visual observation \mathbf{o}_t consists of images of the scene captured by the robot’s cameras. The goal g defines a high-level instruction for a long-horizon task (e.g., “Clean the desk”), which implicitly incurs the sequential execution of multiple sub-tasks $[\hat{g}_1, \hat{g}_2, \dots, \hat{g}_N]$ (e.g., “Put the pen in the pen holder” \rightarrow “Close the laptop” \rightarrow “Put the book on the bookshelf” \rightarrow etc.). For simplicity, we assume that each sub-task can be completed within a single time step. The action \mathbf{a}_t represents low-level robot actions, such as specifying the Cartesian position of the robot’s end-effector. These commands are executed by the robot to interact with the environment. New observations \mathbf{o}_{t+1} are then captured to determine subsequent actions \mathbf{a}_{t+1} .

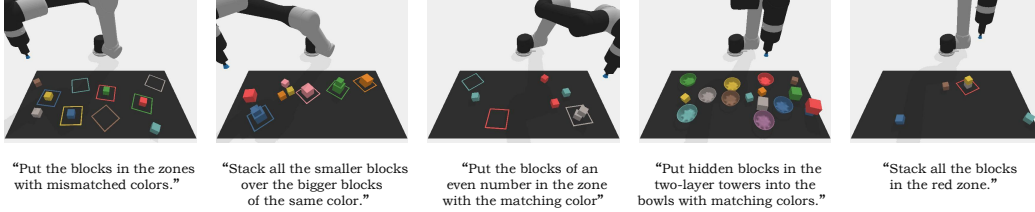


Figure 2: An example of the long-horizon LoHoSet. Object attributes like size, color, quantity, and position vary across cases.

One typical trajectory for this task can be represented as $\{g, \gamma_1, \gamma_2, \dots, \gamma_N\}$, where g denotes the overall goal and $\gamma_t = (\mathbf{o}_t, \hat{g}_t, \mathbf{a}_t)$ denotes a triplet of observation, sub-task, and robot action at time step t . During inference, the policy is only provided with the overall goal and does not have access to the sub-tasks. Consequently, it must rely on implicit or explicit planning to infer the sub-tasks.

Vanilla VLA The vanilla VLA model (Figure 1, left top) is limited to action generation and cannot produce language outputs. Thus, in fact, it implicitly performs sub-task planning, with only a resultant action \mathbf{a}_t yielded based on the high-level goal g and the current observation \mathbf{o}_t :

$$\pi_\theta(\mathbf{o}_t, g) \rightarrow \mathbf{a}_t. \quad (1)$$

The Hierarchical Architecture The implicit sub-task of VLA lacks interpretability and reliability. To address this, the hierarchical architecture (Figure 1, left bottom) proposes to use an external high-level planner to explicitly infer the next atomic sub-task based on the current observation and the high-level goal. Then the low-level controller generates actions to execute this sub-task:

$$\pi_\theta^{\text{planner}}(\mathbf{o}_t, g) \rightarrow \hat{g}_t, \pi_\theta^{\text{controller}}(\mathbf{o}_t, \hat{g}_t) \rightarrow \mathbf{a}_t. \quad (2)$$

LoHoVLA Rather than employing disjoint modules that may suffer from suboptimal coordination and modeling redundancy, we advocate a unified paradigm that integrates high-level task planning and low-level motion control into a model (Figure 1 right). In formal, there is:

$$\pi_\theta(\mathbf{a}_t, \hat{g}_t \mid \mathbf{o}_t, g) = \pi_\theta(\mathbf{a}_t \mid \mathbf{o}_t, g, \hat{g}_t) \cdot \pi_\theta(\hat{g}_t \mid \mathbf{o}_t, g). \quad (3)$$

As the equation implies, LoHoVLA first infers the next atomic sub-task and then uses it as contextual guidance to predict the robot’s action. High-level task planning corresponds to modeling $\pi_\theta(\hat{g}_t \mid \mathbf{o}_t, g)$, while low-level motion control corresponds to modeling $\pi_\theta(\mathbf{a}_t \mid \mathbf{o}_t, g, \hat{g}_t)$, both distributions are represented within a single unified model.

3.2 A Synthetic Dataset for Long-horizon Embodied Tasks: LoHoSet

The training of LoHoVLA relies on a collection of demonstrations (g, γ_t) with $\gamma_t = (\mathbf{o}_t, \hat{g}_t, \mathbf{a}_t)$. The primary challenge is that sub-task annotation for real-world long-horizon tasks can rarely be obtained in a scalable manner without human intervention. To address this, we opt for a simulator-based approach following prior works [38, 50, 32]. Concretely, we construct the LoHoSet dataset based on the Ravens robot simulator [49]. The simulation environment includes a UR5e robotic arm with a suction gripper and several objects placed on a table. The environment provides a reward signal only when the predicted action is both semantically correct and successfully executed. To simulate real-world uncertainties, the simulator adds observation noise and introduces a dropping probability p for the end-effector to drop the picked block every second. The visual observation $\mathbf{o} = (\mathcal{I}_{\text{color}}, \mathcal{I}_{\text{depth}})$ comprises RGB and depth top-down orthographic reconstructions of the scene. The goal instruction g mainly focuses on rearranging the objects into the desired configuration (e.g., “Stack blocks in alternate colors on the green zone”).

We collect decomposed sub-tasks based on manually designed rules, thanks to the availability of complete information about the scenes from the simulator. In particular, we randomly initialize the scene, and we directly estimate the action $\mathbf{a} = (\mathcal{T}_{\text{pick}}, \mathcal{T}_{\text{place}})$ based on the positions of target objects. For sub-tasks without dependency constraints, their execution order is randomized; otherwise, we generate them with pre-defined logics. Each object is assigned a textual label, which is inserted into

Algorithm 1: LoHoVLA Test-time Closed-Loop Control

Data: LoHoVLA policy π_θ , initial observation \mathbf{o}_0 , high-level goal g , reward r , failure count k , failure threshold K , done flag *done*

$t \leftarrow 0, k \leftarrow 0, r \leftarrow 0, done \leftarrow false$

while *not done* **do**

if $t = 0$ **or** $r > 0$ **or** $k > K$ **then**

$\hat{g}_t \sim \pi_\theta(\hat{g}_t \mid \mathbf{o}_t, g)$;

$k \leftarrow 0$;

end

$\mathbf{a}_t \sim \pi_\theta(\mathbf{a}_t \mid \mathbf{o}_t, g, \hat{g}_t)$;

 execute \mathbf{a}_t ;

 get $\mathbf{o}_{t+1}, r, done$ from environment;

if $r = 0$ **then**

$k \leftarrow k + 1$;

end

$t \leftarrow t + 1$;

end

sub-task templates to generate sub-task descriptions (e.g., “Pick up the green block and place it in the green zone” \rightarrow “Pick up the blue block and place it on the green block” \rightarrow etc.). This process enables the generation of a large and diverse set of demonstrations.

The resultant LoHoSet finally includes three types of objects: blocks, bowls, and zones, available in 11 distinct colors. The blocks are of two sizes, large and small. Following LoHoRavens [50], we adopt the 10 long-horizon tasks (detailed in Table 1) and 3 pick-and-place primitive tasks to facilitate comparison with the baselines. We include 10 additional long-horizon tasks in LoHoSet to improve the generalization capacities of the trained model (see Figure 3 left). An example of these tasks is shown in Figure 2. For more details on LoHoSet, please refer to Appendix A.

3.3 Model Architecture

LoHoVLA employs a large pretrained vision language model (VLM) as its backbone to generate the next tokens, with specialized de-tokenizers to translate them as linguistic sub-tasks and actions, respectively. To address errors in both high-level planning and low-level control, it incorporates a hierarchical closed-loop control mechanism. An overview of LoHoVLA is presented in Figure 1.

The Base Vision Language Model. We select PaliGemma [5] as the backbone for our model due to its proven effectiveness in prior studies [6, 20]. PaliGemma is a multi-modal foundation model processing both images and textual prompts for text generation. It integrates a SigLIP-based image encoder, a Gemma-2B decoder-only language model, and a linear projection layer that maps visual features into the language model’s token space.

The Action De-Tokenizer. Following prior work [1, 7, 24], we represent robot actions as discrete tokens to enable joint training with vision–language datasets. Specifically, we discretize the normalized action values into 1,024 uniform bins. During inference, robot actions are recovered by de-tokenizing and de-normalization.

Hierarchical closed-loop control mechanism. Compared to closed-loop control for atomic tasks, managing long-horizon tasks is more complex. Execution failures may arise from sub-task planning errors, inaccurate action predictions, or external disturbances. More formally, the three error types are: (1) sub-task planning error, (2) correct sub-task planning but incorrect action prediction, and (3) correct planning and prediction, with failures caused by external disturbances.

LoHoVLA embraces a hierarchical closed-loop control strategy that re-predicts actions more frequently than it re-plans sub-tasks. Specifically, if the number of failures during the current sub-task exceeds a predefined threshold K , the system triggers sub-task re-planning; otherwise, it only re-predicts the action. The control procedure used during evaluation is outlined in Algorithm 1. We assume that the robot receives a positive reward r upon completing a sub-task. The *done* flag is used to determine whether the overall goal has been achieved.

Table 1: The seen and unseen tasks from LoHoRavens benchmark. Note that we also add the pick-and-place primitive task to the seen tasks.

	Tasks	Instruction
Seen Tasks	A. pick-and-place-primitive	“Put the [OBJ] on the [OBJ / POS].”
	B. put-block-into-matching-bowl	“Put the blocks in the bowls with matching colors.”
	C. stack-smaller-over-bigger-with-same-color	“Stack smaller blocks over bigger blocks of the same color.”
	D. stack-block-in-absolute-area	“Stack all the blocks in the [ABS POS] area.”
	E. put-even-blocks-in-same-color-zone	“Move all blocks of a color that occur in even numbers.”
Unseen Tasks	F. put-block-into-mismatching-bowl	“Put the blocks in the bowls with mismatching colors.”
	G. stack-blocks-of-same-size	“Stack blocks of the same size.”
	H. stack-blocks-with-alternate-color	“Stack blocks in alternate colors.”
	I. stack-smaller-over-bigger-with-same-color-in-same-color-zone	“Stack blocks of the same color in the zone with same color, with the bigger blocks underneath.”
	J. move-blocks-between-absolute-positions	“Move all the blocks in the [ABS POS] area to the [ABS POS] area.”
	K. stack-blocks-of-same-size	“Stack blocks of the same color.”

This design avoids unnecessary sub-task re-planning in cases (2) and (3). Our experiments demonstrate that the hierarchical closed-loop control scheme effectively mitigates errors in long-horizon tasks while avoiding redundant inference steps.

3.4 Training Configurations

During LoHoVLA’s training, we optimize the language model backbone while keeping the image encoder and the linear projection layer fixed. The training objective consists of two components: sub-task generation and action prediction. Both outputs are produced by the language model head and optimized using cross-entropy loss. The total loss is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{text}} + \mathcal{L}_{\text{action}}. \quad (4)$$

We adopt a two-stage training strategy. In the first stage, we fine-tune PaliGemma on long-horizon tasks, optimizing only the text loss to improve high-level task planning. In the second stage, we augment the dataset with pick-and-place primitive tasks and optimize both text and action losses to enhance action prediction.

4 Experiments

Our experimental evaluations aim to assess LoHoVLA’s capabilities in high-level task planning, low-level motion control, and generalization to novel tasks unseen during training. Specifically, we address the following questions:

- How does LoHoVLA perform compared to hierarchical architecture baselines and standard VLA models in terms of performance and generalization on long-horizon tasks?
- How effective are hierarchical closed-loop control schemes at mitigating errors in long-horizon tasks, where failures can arise from both high-level task planning and low-level motion control?
- Can dataset expansion and a two-stage training strategy improve overall model performance?

4.1 Experimental Setup

Baselines. We use LoHoRavens [50] as the baseline with a hierarchical architecture. It consists of a Planner for high-level sub-task planning, an Actor for low-level motion control, and a Reporter for feedback-driven closed-loop control. It supports two types of Reporter-based feedback: (1) Explicit feedback, where the Planner (LLaMA 2 13B [40]) generates sub-tasks, the Actor (CLIPort [38]) executes them, and the Reporter (OpenFlamingo [3]) provides outcome captions to refine future plans; and (2) Implicit feedback, where the Reporter encodes visual observations using a frozen CLIP model [35], projects them via an MLP, and sends the embeddings to the Planner through LLaVA [29] for continuous plan adjustment.

Table 2: Comparison of the average award (%) and success rate (%) on LoHoRavens benchmark. Bold entries indicate the highest success rates, underlined entries indicate the second-highest.

Tasks		Vanilla VLA	LoHoRavens		LoHoVLA
			Explicit feedback	Implicit feedback	
Seen Tasks	A	79.0 / 79.0	67.3 / -	67.3 / -	<u>77.5</u> / 77.5
	B	14.9 / 0.0	31.4 / -	<u>37.0</u> / -	97.8 / 91.5
	C	<u>26.8</u> / 0.5	18.0 / -	22.1 / -	34.9 / 22.5
	D	32.3 / 3.0	30.4 / -	<u>33.2</u> / -	35.8 / 11.5
	E	<u>22.1</u> / 3.5	9.6 / -	8.2 / -	85.1 / 81.0
Unseen Tasks	F	<u>52.1</u> / 9.0	28.5 / -	21.1 / -	86.1 / 41.0
	G	6.8 / 0.0	<u>21.9</u> / -	14.7 / -	40.1 / 25.0
	H	7.3 / 0.0	<u>13.2</u> / -	5.2 / -	16.7 / 7.5
	I	<u>43.1</u> / 1.5	12.8 / -	11.7 / -	77.2 / 52.0
	J	<u>38.6</u> / 10.5	27.4 / -	27.2 / -	43.6 / 22.0
	K	<u>58.2</u> / 33.0	4.0 / -	6.8 / -	73.8 / 54.5

As a secondary baseline, we train a vanilla VLA model to isolate the effect of explicit sub-task prediction. This model directly predicts low-level robot actions without producing intermediate language outputs, relying solely on implicit sub-task inference. Our goal is to test whether it can learn sub-task reasoning without explicit planning.

Training Details. LoHoVLA uses the PaliGemma-3b-mix-224 model as its backbone and is trained in two stages. In the first stage, we fine-tune PaliGemma on 14 long-horizon tasks—comprising 4 seen tasks from LoHoRavens and 10 additional tasks we designed—each with 1,000 demonstrations. We optimize only the text loss to improve high-level task planning. Unlike LoHoRavens, which uses prompt engineering to preserve generalization, our model requires fine-tuning, which leads to overfitting on the limited set of 4 tasks. To address this, we include 10 additional tasks to enhance generalization. In the second stage, we augment the dataset with 10,000 demonstrations for each pick-and-place primitive and optimize both text and action losses to improve low-level motion control.

Training is performed using the AdamW optimizer [31] with weight decay and gradient clipping. We utilize DeepSpeed [36] for efficient distributed training. In training stage one, we train with 8 NVIDIA 4090 GPUs (24GB VRAM), a per-device batch size of 2, a learning rate of $5e-5$, and for 3 epochs. We apply LoRA [17] with a rank of 16 targeting all linear layers. In training stage two, we use a learning rate of $1e-5$ for 1 epoch, keeping other settings unchanged. For comparison, we train the standard VLA model on the same dataset without sub-task labels using a learning rate of $1e-5$, for 5 epochs, under the same hardware and LoRA configuration.

Evaluation Metrics. There are two evaluation methods for determining whether object states match the ground-truth, depending on the task category. The first is pose match, which requires the object’s position and orientation to exactly align with the ground-truth. The second is zone match, where the overlap area between the predicted and ground-truth object must exceed a predefined threshold. We evaluate each task instance using a score from 0 (failure) to 100 (success), based on the proportion of correctly completed pick-and-place steps. For example, if a task requires ten steps and the model completes eight, the score would be 80%. In our evaluations, we report both the average score and the success rate on the test set.

4.2 Main Results

We evaluate LoHoVLA against the vanilla VLA and LoHoRavens on both seen and unseen tasks from LoHoRavens (Table 1). LoHoVLA and Vanilla VLA are trained on five seen tasks and ten additional tasks introduced in this work. Evaluation results, including those for LoHoRavens with both explicit and implicit feedback mechanisms, are shown in Table 2.

Table 3: Comparison of the average reward (%), success rate (%), and number of sub-task planning across three strategies. Strategy (a) performs the worst. Strategies (b) and (c) are comparable overall, but (c) requires fewer high-level planning steps.

Tasks		Strategy (a)		Strategy (b)		Strategy (c)	
Seen Tasks	A	77.5 / 77.5	1.3	77.5 / 77.5	1.3	77.5 / 77.5	1.3
	B	89.5 / 74.0	5.7	96.4 / 88.5	6.4	97.8 / 91.5	6.2
	C	30.3 / 8.5	3.7	35.1 / 24.0	7.9	34.9 / 22.5	7.8
	D	20.1 / 2.5	2.2	31.3 / 9.0	7.6	35.8 / 11.5	7.3
	E	63.1 / 59.0	5.9	86.4 / 84.0	6.6	85.1 / 81.0	6.2
Unseen Tasks	F	62.3 / 29.0	3.6	86.2 / 42.0	6.8	86.1 / 41.0	6.7
	G	32.1 / 19.5	5.2	39.9 / 24.5	8.2	40.1 / 25.0	7.9
	H	14.1 / 4.0	4.5	15.9 / 6.0	8.1	16.7 / 7.5	6.4
	I	66.4 / 41.0	3.7	78.2 / 56.0	6.9	77.2 / 52.0	6.8
	J	31.3 / 14.5	4.2	44.5 / 23.5	7.4	43.6 / 22.0	7.1
	K	52.9 / 34.0	4.7	71.1 / 52.0	7.2	73.8 / 54.5	6.9

LoHoVLA achieves the highest average score and success rate across nearly all tasks. On the *put-block-into-matching-bowl* task, it attains near-perfect accuracy. On the most challenging reasoning task *put-even-blocks-in-same-color-zone*, which requires integrating color recognition, counting, spatial reasoning, and logic, LoHoVLA achieves a score of 85.1 and a success rate of 81.0, while all baselines perform poorly. Notably, LoHoVLA demonstrates strong generalization to unseen tasks, consistently outperforming all baselines despite having no prior exposure.

Interestingly, LoHoVLA occasionally outperforms on long-horizon tasks compared to *pick-and-place-primitive* tasks. This is largely due to differences in evaluation criteria: zone-match tasks (e.g., involving bowls or colored zones) tolerate minor spatial inaccuracies, which LoHoVLA handles effectively. In contrast, pose-match tasks (e.g., block stacking) require precise alignment, where occasional sub-optimal motor trajectories can slightly reduce performance. Nevertheless, LoHoVLA remains robust across both task types.

Vanilla VLA performs the worst among all models, with zero success rates on several tasks. Our qualitative analysis reveals that this is primarily due to the absence of sub-task supervision, which leads the model to overfit to frequent patterns in the training data. For instance, in the *put-block-into-matching-bowl* task, it often places blocks in incorrect bowls, disregarding the goal condition.

4.3 Closed-Loop Strategies Comparison

To assess the effectiveness of our specialized closed-loop control mechanism in addressing task execution failures, we compare the following three control strategies.

- **Strategy (a).** On failure, the system re-predicts the action only without re-planning the sub-task.
- **Strategy (b).** The system re-plans the sub-task and then re-predicts the action after every failure.
- **Strategy (c).** The hierarchical closed-loop control strategy: if the number of failures within the current sub-task exceeds a predefined threshold K , the system initiates sub-task re-planning; otherwise, it only re-predicts the action. In our experiments, we set $K = 2$.

In addition to measuring the average reward and success rate, we track the number of high-level sub-task planning. All three strategies are evaluated using the fine-tuned LoHoVLA model. Results are reported in Table 3.

As expected, strategy (a) yields the poorest performance. When failures stem from incorrect sub-task planning, this approach continues executing flawed plans, potentially leading to deadlocks. Strategies (b) and (c) perform comparably in overall metrics; however, strategy (c) requires fewer high-level

sub-task planning steps. This is because many failures result from low-level prediction errors or external disturbances, where re-planning the sub-task is unnecessary.

Furthermore, task characteristics influence strategy performance. For reasoning-heavy tasks (e.g., *put-even-blocks-in-same-color-zone*), strategy (b) excels by promptly correcting sub-task assignments. In contrast, for tasks demanding precise motor control (e.g., *stack-block-in-absolute-area*), strategy (c) performs better, as repeated low-level attempts improve the likelihood of success.

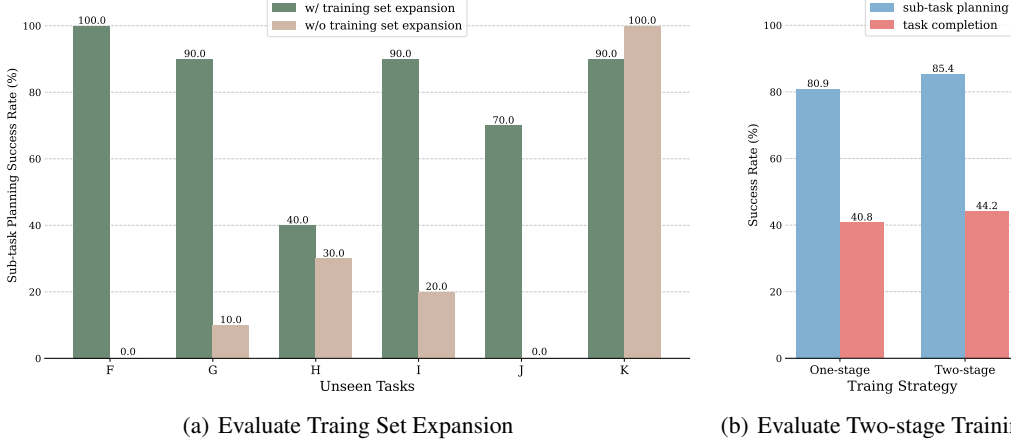


Figure 3: (a) Comparison of performance on unseen tasks between training with and without dataset expansion, evaluated using the sub-task planning success rate (%). (b) Comparison of performance between one-stage and two-stage training strategies, evaluated using both the sub-task planning success rate (%) and task completion success rate (%).

4.4 Ablation Studies

We examine the effects of training set expansion and the two-stage training method on model performance. To better evaluate planning ability, we introduce a new metric: sub-task planning success rate. Unlike sub-task execution success, this metric isolates planning performance by removing the influence of low-level action prediction and external disturbances. Specifically, for each task, we randomly sample 10 timesteps from all episodes and manually enumerate all valid subsequent sub-tasks. An LLM is then used to assess whether the model’s predicted sub-task at each timestep is semantically equivalent to any of the enumerated ground-truth options. Further details are provided in Appendix B.

Evaluate Training Set Expansion. We trained LoHoVLA with and without training set expansion (i.e., the extra 10 tasks used for training mentioned in Section 3.2), and evaluated the generalization ability on the unseen tasks of the LoHoRavens Benchmark. The results are shown in Figure 3 left. It can be observed that the model trained without extra data has poor generalization ability. This is due to severe overfitting to the seen tasks. For example, the success rate of the task *put-block-into-mismatching-bowl* is 0, because its scene is similar to that of *put-block-into-matching-bowl*, which led the model to overfit to the latter and ignore the language goal, placing the block into the matching-colored bowl instead. The expanded dataset alleviates such overfitting issues.

Evaluate Two-stage Training. We conduct both one-stage and two-stage training experiments on LoHoVLA using identical configurations. In both settings, the model is trained for 5 epochs. The key difference is that in the two-stage approach, the primitive tasks and action labels are introduced only after the first 3 epochs. The results—average sub-task planning success rate (%) and average task success rate (%)—are presented in Figure 3 right. The one-stage training setup yields a lower sub-task planning success rate, which consequently results in reduced task success. This indicates that introducing action labels and primitive tasks too early hinders the effective optimization of high-level task planning.

5 Conclusion

For long-horizon embodied tasks requiring both high-level planning and low-level control, existing VLA models and hierarchical approaches struggle with planning and coordination. To address this, we propose LoHoVLA, a unified VLA framework that uses a large pretrained vision-language model to jointly generate sub-tasks and robot actions. It incorporates a hierarchical closed-loop control mechanism to correct errors at both levels. Empirical results show that LoHoVLA outperforms prior VLA and hierarchical methods by decent margins and demonstrates strong generalization.

Limitations. The limitations stem primarily from the limited precision of robot actions due to their discrete structure. Additionally, our assumption that a sub-task can be completed within a single timestep may not be practical in real-time applications.

References

- [1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.
- [2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35: 23716–23736, 2022.
- [3] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*, 2023.
- [4] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.
- [5] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.
- [6] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0 : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.
- [8] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.
- [9] Hongyi Chen, Yunchao Yao, Ruixuan Liu, Changliu Liu, and Jeffrey Ichnowski. Automating robot failure recovery using vision-language models with optimized prompts. *arXiv preprint arXiv:2409.03966*, 2024.
- [10] Xi Chen, Xiao Wang, Lucas Beyer, Alexander Kolesnikov, Jialin Wu, Paul Voigtlaender, Basil Mustafa, Sebastian Goodman, Ibrahim Alabdulmohsin, Piotr Padlewski, et al. Pali-3 vision language models: Smaller, faster, stronger. *arXiv preprint arXiv:2310.09199*, 2023.
- [11] An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.
- [12] Yinpei Dai, Jayjun Lee, Nima Fazeli, and Joyce Chai. Racer: Rich language-guided failure recovery policies for imitation learning. *arXiv preprint arXiv:2409.14674*, 2024.
- [13] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. Palm-e: An embodied multimodal language model. 2023.
- [14] Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.
- [15] Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*, 2025.

- [16] Ziyu Guo, Renrui Zhang, Xiangyang Zhu, Chengzhuo Tong, Peng Gao, Chunyuan Li, and Pheng-Ann Heng. Sam2point: Segment any 3d as videos in zero-shot and promptable manners. *arXiv preprint arXiv:2408.16768*, 2024.
- [17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- [18] Yingdong Hu, Fanqi Lin, Tong Zhang, Li Yi, and Yang Gao. Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning. *arXiv preprint arXiv:2311.17842*, 2023.
- [19] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.
- [20] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.
- [21] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020.
- [22] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2(3):6, 2022.
- [23] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Forty-first International Conference on Machine Learning*, 2024.
- [24] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [25] Boyi Li, Philipp Wu, Pieter Abbeel, and Jitendra Malik. Interactive task planning with language models. *arXiv preprint arXiv:2310.10645*, 2023.
- [26] Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2024.
- [27] Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, et al. Hamster: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025.
- [28] 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 *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.
- [29] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.
- [30] Yang Liu, Weixing Chen, Yongjie Bai, Xiaodan Liang, Guanbin Li, Wen Gao, and Liang Lin. Aligning cyber space with physical world: A comprehensive survey on embodied ai. *arXiv preprint arXiv:2407.06886*, 2024.
- [31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

- [32] Yuan Meng, Xiangtong Yao, Haihui Ye, Yirui Zhou, Shengqiang Zhang, Zhenshan Bing, and Alois Knoll. Data-agnostic robotic long-horizon manipulation with vision-language-guided closed-loop feedback. *arXiv preprint arXiv:2503.21969*, 2025.
- [33] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.
- [34] Abby O’Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.
- [35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.
- [36] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3505–3506, 2020.
- [37] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.
- [38] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR, 2022.
- [39] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.
- [40] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [41] Zixuan Wang, Bo Yu, Junzhe Zhao, Wenhao Sun, Sai Hou, Shuai Liang, Xing Hu, Yinhe Han, and Yiming Gan. Karma: Augmenting embodied ai agents with long-and-short term memory systems. *arXiv preprint arXiv:2409.14908*, 2024.
- [42] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [43] Junjie Wen, Minjie Zhu, Yichen Zhu, Zhibin Tang, Jinming Li, Zhongyi Zhou, Chengmeng Li, Xiaoyu Liu, Yaxin Peng, Chaomin Shen, et al. Diffusion-vla: Scaling robot foundation models via unified diffusion and autoregression. *arXiv preprint arXiv:2412.03293*, 2024.
- [44] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.
- [45] Zhenyu Wu, Ziwei Wang, Xiuwei Xu, Jiwen Lu, and Haibin Yan. Embodied task planning with large language models. *arXiv preprint arXiv:2307.01848*, 2023.
- [46] Zhutian Yang, Caelan Garrett, Dieter Fox, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Guiding long-horizon task and motion planning with vision language models. *arXiv preprint arXiv:2410.02193*, 2024.

- [47] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.
- [48] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.
- [49] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.
- [50] Shengqiang Zhang, Philipp Wicke, Lütfi Kerem Şenel, Luis Figueredo, Abdeldjallil Naceri, Sami Haddadin, Barbara Plank, and Hinrich Schütze. Lohoravens: A long-horizon language-conditioned benchmark for robotic tabletop manipulation. *arXiv preprint arXiv:2310.12020*, 2023.
- [51] Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. *arXiv preprint arXiv:2412.18194*, 2024.
- [52] Chao Zhao, Shuai Yuan, Chunli Jiang, Junhao Cai, Hongyu Yu, Michael Yu Wang, and Qifeng Chen. Erra: An embodied representation and reasoning architecture for long-horizon language-conditioned manipulation tasks. *IEEE Robotics and Automation Letters*, 8(6):3230–3237, 2023.
- [53] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025.
- [54] Zhehua Zhou, Jiayang Song, Kunpeng Yao, Zhan Shu, and Lei Ma. Isr-llm: Iterative self-refined large language model for long-horizon sequential task planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2081–2088. IEEE, 2024.
- [55] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

A LoHoSet



Figure 4: Visual examples of all tasks in the LoHoSet dataset, including 3 pick-and-place primitives and 20 long-horizon tasks, showcasing the diversity and complexity of the task scenarios.

LoHoSet consists of 3 pick-and-place primitive tasks and 20 long-horizon tasks. Among them, 10 long-horizon tasks and all 3 primitive tasks are adapted from the LoHoRavens benchmark to facilitate comparison with existing baselines. The other 10 long-horizon tasks are introduced to improve the generalization ability of the trained model. A complete list of tasks is provided in Table 4, and examples of all tasks are illustrated in Figure 4.

Table 4: Summary of all tasks in LoHoSet, including 3 primitive pick-and-place tasks and 20 long-horizon tasks, with distinctions between those adopted from LoHoRavens and those introduced for improved generalization.

Tasks		Instruction
LoHoRavens Seen Tasks	pick-and-place-primitive	“Put the [OBJ] on the [OBJ].”
	pick-and-place-primitive-with-size	“Put the [SIZE] [OBJ] on the [SIZE] [OBJ].”
	pick-and-place-primitive-with-absolute-position	“Put the [OBJ] on the [ABS_POS].”
	put-block-into-matching-bowl	“Put the blocks in the bowls with matching colors.”
	stack-smaller-over-bigger-with-same-color	“Stack smaller blocks over bigger blocks of the same color.”
	stack-block-in-absolute-area	“Stack all the blocks in the [ABS_POS] area.”
LoHoRavens Unseen Tasks	put-even-blocks-in-same-color-zone	“Move all blocks of a color that occur in even numbers.”
	put-block-into-mismatching-bowl	“Put the blocks in the bowls with mismatching colors.”
	stack-blocks-of-same-size	“Stack blocks of the same size.”
	stack-blocks-with-alternate-color	“Stack blocks in alternate colors.”
	stack-smaller-over-bigger-with-same-color-in-same-color-zone	“Stack blocks of the same color in the zone with same color, with the bigger blocks underneath.”
	move-blocks-between-absolute-positions	“Move all the blocks in [ABS_POS] area to [ABS_POS] area.”
Additional Tasks	stack-blocks-of-same-color	“Stack blocks of the same color.”
	put-block-into-mismatching-zone	“Put the blocks in the zones with mismatching colors.”
	put-hidden-blocks-in-two-layer-towers-into-matching-bowls	“Put all the hidden blocks in the two-layer stacked towers into the bowls with matching colors.”
	put-hidden-blocks-in-two-layer-towers-into-mismatching-bowls	“Put all the hidden blocks in the two-layer stacked towers into the bowls with mismatching colors.”
	put-hidden-blocks-in-three-layer-towers-into-matching-bowls	“Put all the hidden blocks in the three-layer stacked towers into the bowls with matching colors.”
	put-hidden-blocks-in-pyramid-into-matching-bowls	“Put all the hidden blocks on the first layer of the pyramid into the bowls with matching colors.”
	stack-bigger-over-smaller-with-same-color-in-same-color-zone	“Stack blocks of the same color in the zone with same color, with the smaller blocks underneath.”
	stack-all-blocks-on-a-zone	“Stack all the blocks on the [COLOR] zone.”
	stack-blocks-by-relative-position	“Stack all the blocks on the [REL_POS] of the [COLOR] block on the [COLOR] zone.”
	move-blocks-between-absolute-positions-by-size	“Move all the [SIZE] blocks in the [ABS_POS] area to the [ABS_POS] area.”
	move-blocks-between-absolute-positions-by-color	“Move all the [COLOR] blocks in the [ABS_POS] area to the [ABS_POS] area.”

B Ablation Studies Details

We investigate the effects of training set expansion and a two-stage training strategy on model performance. To more accurately evaluate planning capabilities, we introduce a new metric: sub-task planning success rate. Unlike sub-task execution success, this metric isolates high-level planning by excluding the influence of low-level action prediction and external noise. For each task, we randomly sample 10 timesteps across all episodes and manually enumerate valid next sub-tasks. A large language model (LLM) then assesses whether the model’s predicted sub-task at each timestep is semantically equivalent to any of the ground-truth options. The evaluation prompt is as follows:

Evaluation Prompt

You are given the current sub-task in a sequential task and a predicted next sub-task generated by a model. Additionally, you are provided with a list of valid ground-truth next sub-tasks. Determine whether the predicted sub-task is semantically equivalent to any of the ground-truth options. Focus solely on semantic similarity in intent and meaning, ignoring differences in wording or phrasing. Respond with 'Yes' if the prediction is semantically equivalent to at least one ground-truth sub-task; otherwise, respond with 'No'.