Structured Preference Optimization for Vision-Language Long-Horizon Task Planning

Xiwen Liang[1*], **Min Lin**[1*], **Weiqi Ruan**[2], **Rongtao Xu**[3], **Yuecheng Liu**[4], **Jiaqi Chen**[5], **Bingqian Lin**[6], **Yuzheng Zhuang**[4], **Xiaodan Liang**[1†]
[1]Shenzhen Campus of Sun Yat-sen University, [2]The Chinese University of Hong Kong, [3]Spatialtemporal AI, [4]Huawei Noah's Ark Lab, [5]The University of Hong Kong, [6]Shanghai Jiaotong University

Abstract

Existing vision-language planning methods perform well on short-horizon tasks but struggle with long-horizon reasoning in dynamic environments due to the difficulty of training models to generate high-quality reasoning processes. To address this, we propose **S**tructured **P**reference **O**ptimization (**SPO**), a framework that enhances reasoning and action selection for long-horizon task planning through structured evaluation and optimized training. SPO introduces: 1) Structured Preference Evaluation and Optimization, which evaluates reasoning chains across task relevance, historical consistency (as part of textual coherence), and image awareness (alignment with visual observations) to construct high-quality preference pairs; and 2) Curriculum-Guided Progressive Learning, enabling the model to adapt from simple to complex tasks, thereby improving generalization and robustness. To advance research in vision-language long-horizon task planning, we introduce ExtendaBench, a comprehensive benchmark covering 1,509 tasks across VirtualHome and Habitat 2.0, categorized into ultra-short, short, medium, and long tasks. Experimental results demonstrate that SPO significantly improves reasoning quality and final decision accuracy, outperforming prior methods on long-horizon tasks and underscoring the effectiveness of preference-driven optimization in vision-language task planning. Specifically, SPO achieves a +5.98% GCR and +4.68% SR improvement in VirtualHome and a +3.30% GCR and +2.11% SR improvement in Habitat over the best-performing baselines.

1 Introduction

In autonomous systems, there is a growing demand for robots capable of executing complex, real-world tasks in domestic environments. Tasks such as organizing a room, preparing a meal, and cleaning up afterward require not only a diverse set of actions but also sophisticated long-term planning capabilities. However, current approaches struggle with long-horizon tasks due to a lack of learning in long-term planning ability and the fact that most benchmarks (Puig et al., 2018; Liao et al., 2019; Shridhar et al., 2020a,b) focus on short-term discrete tasks. This gap hinders progress toward robots capable of handling the complex, multi-step tasks demanded by real-life scenarios.

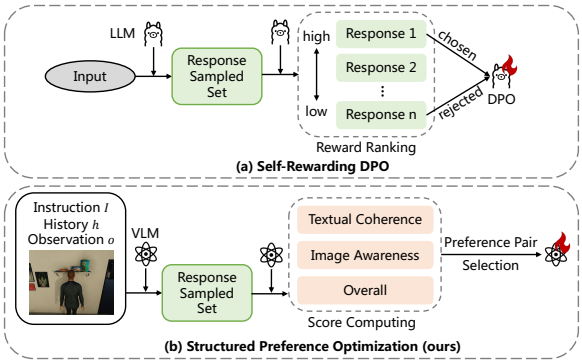

Figure 1: Comparison with existing methods. (a) Self-Rewarding DPO (Yuan et al., 2024) relies on a single reward criterion to rank sampled responses and selects both the highest-ranked (preferred) and lowest-ranked (rejected) responses for DPO training. (b) Structured Preference Optimization (ours) introduces a structured scoring framework with multiple criteria and an adaptive preference selection strategy, enabling more fine-grained and informed optimization.

Existing reasoning-based decision-making methods primarily rely on prompting strategies or environmental feedback to determine actions, often without explicitly modeling the quality of reasoning chains. While recent approaches (Yao et al., 2022; Zhao et al., 2024; Zhi-Xuan et al., 2024) leverage textual inputs for reasoning, they lack a structured mechanism to incorporate multimodal information or refine reasoning processes over extended horizons. Furthermore, prior

[*]Equal contribution.
[†]Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 17490–17515
November 4-9, 2025

optimization frameworks, such as Self-Rewarding DPO (Yuan et al., 2024), rely on a single reward criterion, which may lead to suboptimal preference selection as in Figure 1.

To address these limitations, we propose Structured Preference Optimization (SPO), a novel framework designed to enhance reasoning quality and decision-making in long-horizon task planning through structured preference evaluation and progressive learning. SPO consists of two core components: 1) **Structured Preference Evaluation and Optimization:** SPO systematically evaluates reasoning chains along two key dimensions—Textual Coherence, which assesses task relevance and historical consistency, and Image Awareness, which measures alignment with visual observations. These evaluations are used to construct high-quality preference pairs that explicitly guide the model toward superior reasoning steps, thereby enhancing decision reliability in complex multimodal tasks. 2) **Curriculum-Guided Progressive Learning:** SPO utilizes a progressive curriculum, incrementally increasing task complexity during training. This structured progression helps the model develop robust reasoning strategies and enhances generalization to diverse long-horizon scenarios, ensuring consistent real-world performance.

Finally, to bridge the notable gap in the field regarding the absence of a benchmark tailored for long-horizon tasks, we propose ExtendaBench, a comprehensive benchmark that categorizes the task into four difficulty levels based on the number of steps required for completion, namely ultra-short, short, medium, and long. Leveraging the generative capabilities of GPT-4o (OpenAI, 2024), we create a diverse and extensive collection of tasks. These tasks undergo minimal human refinement to ensure high-quality data while significantly reducing the costs and effort associated with manual data labeling.

Our contributions can be summarized as follows:

- We introduce Structured Preference Optimization (SPO), a framework that enhances long-horizon reasoning through structured preference-based evaluation and curriculum-guided learning, enabling more effective decision-making.
- We propose ExtendaBench, a benchmark with four levels of difficulty and 1,509 tasks across VirtualHome and Habitat 2.0, providing a comprehensive evaluation suite for sustained reasoning in long-horizon task planning.
- We validate SPO through extensive experiments, demonstrating state-of-the-art performance in long-horizon task planning.

2 Related Work

2.1 Multimodal Large Language Models

The emergence of LLMs (Touvron et al., 2023; Chiang et al., 2023) has driven substantial progress in multimodal large language models (MLLMs), which aim to integrate both visual and textual modalities, advancing toward a more generalized form of intelligence. Early works such as BLIP-2 (Jian et al., 2024), MiniGPT-4 (Zhu et al., 2023), LLaVA (Liu et al., 2024), and OpenFlamingo (Awadalla et al., 2023) capitalized on pretrained vision encoders paired with LLMs, demonstrating strong performance in tasks like visual question answering and image captioning. mPLUG-Owl (Ye et al., 2023) introduces a modularized training framework to further refine cross-modal interactions. On the closed-source side, models such as GPT-4V (OpenAI, 2023) and Gemini (Team et al., 2023) pushes the boundaries of multimodal reasoning and interaction capabilities. Unlike general-purpose MLLMs, we repurpose them for structured training in embodied planning tasks.

2.2 LLM Self-improvement

Self-improvement methods enhance LLMs by training on their own generated outputs. These methods often involve supervised fine-tuning (SFT) on high-quality responses generated by the models themselves (Li et al., 2023; Wang et al., 2024b) or preference optimization (Yuan et al., 2024; Rosset et al., 2024; Pang et al., 2024; Prasad et al., 2024; Zhang et al., 2024b; Jiang et al., 2024), where the model is trained to distinguish between better and worse responses. These approaches mostly employ LLM-as-a-Judge prompting (Zheng et al., 2024) or train strong reward models (Xu et al., 2023; Havrilla et al., 2024) to evaluate and filter generated data, thereby guiding the model toward improved performance. Unlike prior methods, we introduce structured preference optimization with targeted pair selection and curriculum learning for long-horizon embodied tasks.

2.3 Embodied Task Planning

Traditional robotics planning methods have relied on search algorithms in predefined domains (Fikes and Nilsson, 1971; Garrett et al., 2020; Jiang et al., 2018), but face scalability challenges in complex environments with high branching factors (Puig et al., 2018; Shridhar et al., 2020a). Heuristics have helped alleviate these limitations, leading to advancements (Baier et al., 2009; Hoffmann, 2001; Helmert, 2006; Bryce and Kambhampati, 2007). More recently, learning-based methods like representation learning and hierarchical strategies have emerged, showing effectiveness in complex decision-making (Eysenbach et al., 2019; Xu et al., 2018, 2019; Srinivas et al., 2018; Kurutach et al., 2018; Nair and Finn, 2019; Jiang et al., 2019). The advent of LLMs has further revolutionized planning by enabling task decomposition and robust reasoning (Li et al., 2022; Huang et al., 2022b; Ahn et al., 2022; Valmeekam et al., 2022; Silver et al., 2022; Song et al., 2023; Rana et al., 2023; Driess et al., 2023; Liu et al., 2023b; Wu et al., 2023; Wake et al., 2023; Chen et al., 2023; Bhat et al., 2024; Zhi-Xuan et al., 2024; Liang et al., 2023b; Zhou et al., 2025; Zhang et al., 2025; Chen et al., 2024a). Other works focus on translating natural language into executable code and formal specifications (Vemprala et al., 2023; Liang et al., 2023a; Silver et al., 2023; Xie et al., 2023; Skreta et al., 2023; Liu et al., 2023a; Zhang and Soh, 2023; Ding et al., 2023b,a; Zhao et al., 2024). Some approaches fine-tune LLMs for better performance (Driess et al., 2023; Qiu et al., 2023; Zhang et al., 2024a), while others opt for few-shot or zero-shot methods (Huang et al., 2022b,a; Singh et al., 2023) to avoid the resource demands of model training. In contrast, our method introduces multimodal preference optimization, fine-grained preference scoring, and curriculum-guided optimization.

3 Preliminaries

Direct Preference Optimization (DPO) (Rafailov et al., 2024) is a reinforcement learning-free approach that optimizes a models policy using preference-labeled data. Instead of relying on an explicit reward model, DPO directly enforces preference ordering by encouraging the model to assign higher probabilities to preferred outputs over less preferred ones.

Given a dataset $D = \{(x, y^+, y^-)\}$, where y^+ is the preferred response and y^- is the less preferred response for input x, DPO optimizes the following contrastive ranking loss:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y^+ \mid x)}{\pi_{\text{ref}}(y^+ \mid x)} - \beta\log\frac{\pi_\theta(y^- \mid x)}{\pi_{\text{ref}}(y^- \mid x)}\right)\right], \quad (1)$$

where σ is the sigmoid function, and β is a scaling factor controlling preference sharpness.

4 Structured Preference Optimization

The Structured Preference Optimization (SPO) framework enhances long-horizon task planning by introducing a structured evaluation mechanism for reasoning quality and a progressive training strategy to improve model generalization. Unlike standard preference optimization, which lacks explicit reasoning quality assessment and task complexity adaptation, SPO systematically refines the models reasoning capabilities through Preference-Based Scoring and Optimization and Curriculum-Guided Training. The overview of our framework is shown in Figure 2.

4.1 Preference-Based Scoring and Optimization

The structured preference-based optimization mechanism evaluates and ranks reasoning chains based on explicit criteria. Unlike standard preference optimization, which treats reasoning as a single scalar preference, SPO decomposes reasoning quality into multiple dimensions and optimizes the models decision-making accordingly.

4.1.1 Structured Preference Evaluation

Instead of relying on external annotations, SPO adopts a self-evaluation approach, where the vision-language model (sLVLM) itself serves as the judge to assess reasoning quality. Given a generated reasoning chain R_i, the model evaluates it based on the task context, which includes: task instruction (I), current image observation (o), and history of executed actions (h). Using this structured input, the model assigns two separate scores to assess different aspects of reasoning quality:

- Textual Coherence (S_{text}): Evaluates the logical consistency of the reasoning chain, ensuring that each step is task-relevant and maintains historical consistency with prior steps. This prevents

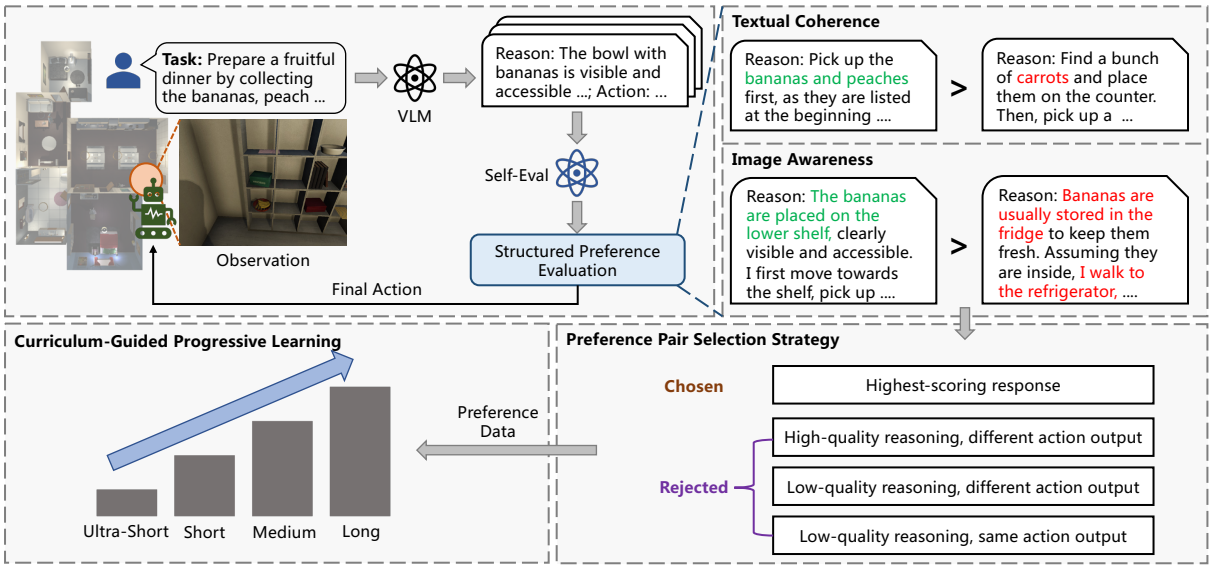

Figure 2: Overview of the Structured Preference Optimization. The SPO consists of three key components: 1) Structured Preference Evaluation, which systematically scores reasoning-action pairs based on textual coherence and image awareness; 2) Preference Pair Selection Strategy, which refines response optimization by selecting the highest-scoring reasoning-action pairs while rejecting low-quality alternatives based on structured criteria; 3) Curriculum-Guided Training, which progressively improves the models capabilities by adapting from ultra-short to long-horizon tasks through a staged optimization process.

reasoning errors such as goal misalignment or contradictions in multi-step plans.

- Image Awareness (S_{image}): Measures whether the reasoning chain sufficiently incorporates relevant information from the visual observations, ensuring that decisions are grounded in the environment rather than relying solely on textual priors.

To obtain these scores, the model is prompted with an evaluation query p, where the model M estimates reasoning quality as follows:

$$S_{\text{text}} = M(p_{\text{text}}, R_i, I, h), \tag{2}$$

$$S_{\text{image}} = M(p_{\text{image}}, R_i, I, o, h), \tag{3}$$

where p_{text} and p_{image} are evaluation prompts designed to assess textual coherence and image awareness, respectively. The overall preference score can then be computed as either a weighted combination:

$$S(R_i) = w_1 \mathcal{S}_{\text{text}} + w_2 \mathcal{S}_{\text{image}}, \tag{4}$$

where w_1 and w_2 are weighting factors that control the relative contribution of textual coherence and image awareness. Alternatively, instead of using a predefined weighted sum, the model directly provides an overall preference score:

$$S(R_i) = M(p_{\text{overall}}, R_i, I, o, h), \tag{5}$$

where p_{overall} is an evaluation prompt requesting a single comprehensive score. Empirically, we found that using the model to generate the overall preference score yields better optimization results compared to manually setting weighting factors.

4.1.2 Preference Pair Selection Strategy

To refine the models reasoning capabilities, SPO constructs structured preference pairs from model-generated samples, ensuring that the optimization process explicitly accounts for both reasoning quality and action selection. Unlike prior methods, which simply select the highest-scoring reasoning chain as the positive sample and the lowest-scoring reasoning chain as the negative sample, SPO introduces a targeted preference selection strategy that prevents the model from over-optimizing reasoning at the cost of decision accuracy.

Given a set of generated reasoning chains $\{R_i\}$ for the same task input (I, o, h), the model self-evaluates each reasoning chain using the scoring mechanism described in Structured Preference Evaluation. The positive sample R^+ is selected as the highest-scoring reasoning chain, and in cases where multiple chains achieve the same highest score, we choose the one where the final action appears most frequently across all generated samples. This ensures that the model prioritizes common and stable action choices, reducing the risk

of selecting an outlier action due to randomness in generation. For the negative sample R^-, instead of always selecting the lowest-scoring reasoning chain, SPO considers different selection strategies to ensure both reasoning quality and action feasibility are optimized. The negative sample is chosen from one of the following categories:

- High-quality reasoning, different action output: Reasoning chain with high preference scores but a different final action from R^+. This discourages optimizing reasoning quality while overlooking action correctness.
- Low-quality reasoning, different action output: Reasoning chain with low preference scores and incorrect final action. This clarifies distinctions between poor reasoning and high-quality thought processes.
- Low-quality reasoning, same action output: Reasoning chain with low preference scores but identical final action to R^+. This prevents the model from focusing solely on reasoning quality without validating decision consistency.

4.1.3 Preference Optimization

Once the structured preference pairs (R^+, R^-) are selected, SPO directly applies DPO to align the models policy with the preferred reasoning chains. The optimization follows the original DPO contrastive ranking loss (referencing Eq. 1, adapted to our task setting with inputs (I, o, h)):

$$\mathcal{L}_{\text{pref}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(I,o,h,R^+,R^-)\sim\mathcal{D}}\Big[\log\sigma\Big(\beta\log\frac{\pi_\theta(R^+ \mid I,o,h)}{\pi_{\text{ref}}(R^+ \mid I,o,h)} - \beta\log\frac{\pi_\theta(R^- \mid I,o,h)}{\pi_{\text{ref}}(R^- \mid I,o,h)}\Big)\Big]. \quad (6)$$

4.2 Curriculum-Guided Progressive Learning

To facilitate structured learning, SPO categorizes tasks into four levels: ultra-short, short, medium, and long-horizon tasks. Instead of training on all task types simultaneously, SPO follows a progressive training strategy to gradually expose the model to increasing task complexity while preventing catastrophic forgetting.

Training is divided into four stages, where the model starts with ultra-short tasks and progressively incorporates more complex tasks in each subsequent stage. At every stage, a certain amount of previously learned tasks is retained to reinforce fundamental reasoning skills and prevent the model from overfitting to newly introduced tasks. This approach ensures that earlier-learned reasoning strategies remain effective as the model learns to handle longer task horizons.

A key challenge in curriculum learning is stabilizing the transition between different difficulty levels without disrupting previously learned decision patterns. To address this, SPO maintains a dynamic balance between newly introduced tasks and previously learned ones. During each training phase, the model is exposed to a mixture of current-stage tasks and replayed tasks from earlier stages, ensuring that it can generalize across task difficulties while refining long-horizon reasoning capabilities.

5 ExtendaBench

The ExtendaBench task corpus is developed using tailored approaches for each simulator, both leveraging GPT-4os advanced generative capabilities. For VirtualHome (Puig et al., 2018), we utilize GPT-4o to directly generate diverse and complex tasks, allowing for a wide range of scenarios. For Habitat 2.0 (Szot et al., 2021), GPT-4o is used to generate pre-defined templates as well as to create specific task instances from these templates, resulting in systematically varied tasks with extended action sequences that are suitable for long-horizon planning.

5.1 VirtualHome

Task Proposal The initial phase begins within the confines of VirtualHome, a simulated environment, where a varied collection of objects sets the stage for a multitude of task scenarios. By employing GPT-4o as a task generator, we design tasks focusing on object manipulation, striving for a wide array of task varieties and complexities. This method ensures an exhaustive representation of scenarios that closely mimic real-world challenges. To facilitate the generator's task creation, we provide prompts that are carefully constructed to inspire a broad range of tasks.

Review In the subsequent phase, GPT-4o undertakes the generation of detailed action plans for the devised tasks, meticulously outlining the steps required for successful task execution. To ensure the feasibility and coherence of these tasks, we introduce an additional examiner of scrutiny, also powered by GPT-4o. This examiner evaluates each task and its associated action plan for clar-

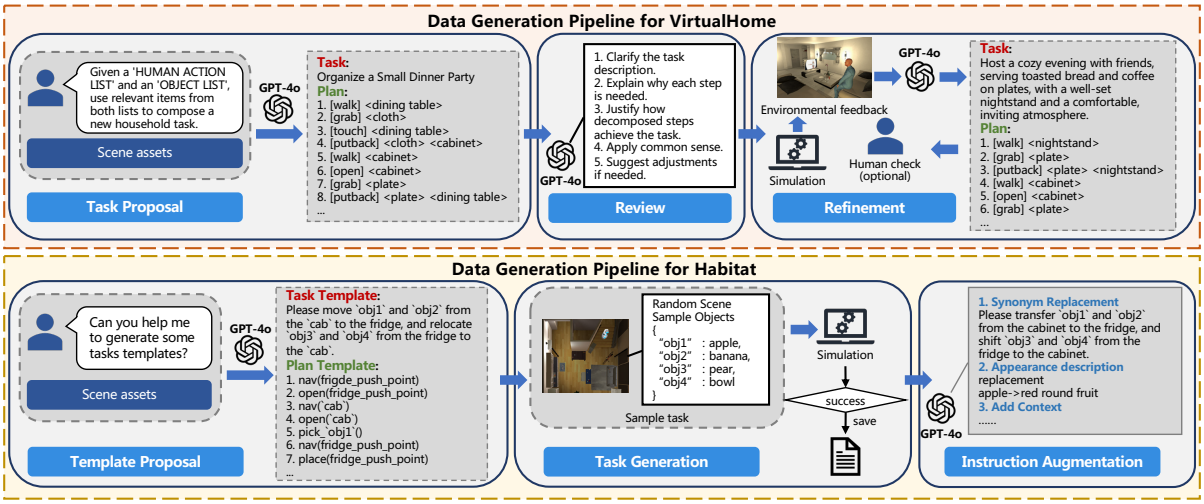

Figure 3: Data generation pipeline for ExtendaBench. (Top) In VirtualHome, GPT-4o generates tasks based on scene assets and a human action list, followed by plan generation, review, and refinement using simulation and feedback. (Bottom) In Habitat, scene objects are sampled and filled into task templates to create executable plans, which are validated in simulation. Instructions are then augmented with synonyms, appearance descriptions, and contextual cues to enhance linguistic diversity.

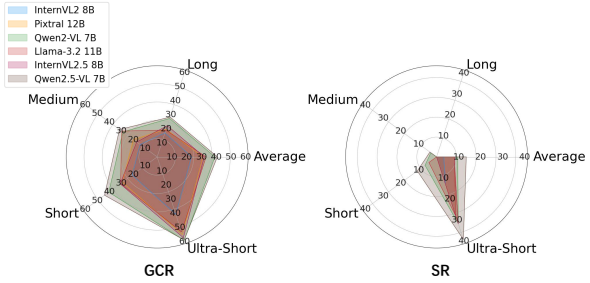

Figure 4: Comparison of various small vision-language models on different sets of our ExtendaBench in VirtualHome.

ity, necessity, and coherence of steps, as well as the relevance and practicality of the actions and items involved, ensuring they belong to the simulated environment VirtualHome. It also assesses each step for common sense applicability, providing constructive feedback for further refinement.

Refinement After undergoing expert scrutiny, the generator refines the tasks and their corresponding action plans. Subsequent simulation of these revised tasks and plans enables further improvements based on simulator feedback. Tasks that are successfully executed within the simulator receive preliminary approval. Nevertheless, to guarantee optimal quality and applicability, we subject each task to a rigorous manual review, evaluating them for practicality and realism. Tasks that do not achieve success in the simulation are minimally modified by human according to the simulator's feedback, focusing on enhancing their realism and feasibility.

The multi-stage process, with minimal human intervention, is designed to ensure the reliability and quality of the tasks and their associated plans. The whole process of generating tasks in benchmark is shown in Figure 3.

5.2 Habitat 2.0

Building on the idea of Language Rearrangement (Szot et al., 2023), we replace its hand-crafted, short-horizon templates with an LLM-driven pipeline (Figure 3) that automatically writes task schemas and expands them into markedly longer action sequences.

Template Proposal GPT-4o generates initial task templates based on scene assets. These templates define general task structures (e.g., moving objects between locations) and serve as the basis for generating varied instructions.

Task Generation Using the task templates, we sample objects within random scenes to generate specific tasks with extended action sequences. This phase results in more complex task plans that evaluate an agent's capacity for long-term planning and adaptability.

Instruction Augmentation To increase task diversity, we apply various transformations to the instructions. These include synonym replacement, appearance description alterations (e.g., apple to red round fruit), and additional contextual details. This augmentation, powered by GPT-4o, allows us to expand the instruction set, testing the agent's understanding and flexibility in interpreting varied language inputs.

Table 1: Comparison with existing methods using Qwen2.5-VL 7B as the baseline on different sets of our ExtendaBench in VirtualHome.

	Ultra-Short		Short		Medium		Long		Average	
Method	GCR	SR	GCR	SR	GCR	SR	GCR	SR	GCR	SR
Baseline	57.32	35.00	42.72	9.62	30.57	3.33	27.47	0	39.52	11.99
CoT (Wei et al., 2022)	68.66	41.67	35.46	3.85	36.36	1.67	20.45	0	40.23	11.80
Self-Rewarding (Yuan et al., 2024)										
Iteration 1	62.13	35.00	42.89	7.69	36.38	3.33	22.55	0	40.99	11.51
Iteration 2	59.15	31.67	44.47	11.54	29.92	3.33	28.80	0	40.58	11.64
Iteration 3	58.93	35.00	48.16	9.62	32.56	3.33	27.46	0	41.78	11.99
Iterative RPO (Pang et al., 2024)										
Iteration 1	67.03	38.33	41.59	7.69	26.97	1.67	26.06	0	40.41	11.92
Iteration 2	72.31	43.33	40.06	1.92	31.66	3.33	22.33	0	41.73	12.15
Iteration 3	59.86	31.67	46.42	11.54	34.02	6.67	29.06	0	42.34	12.47
SPO (1 iteration)	71.53	48.33	48.96	13.46	38.92	3.33	31.41	2.17	**47.71**	**16.83**

5.3 Dataset Statistics

The categorization within ExtendaBench is defined by the length of the action sequence required to accomplish a task, distributed as follows:

- Ultra-Short Tasks: Tasks that can be completed in fewer than 10 actions.
- Short Tasks: Tasks requiring 10 to 20 actions for completion.
- Medium Tasks: Tasks necessitating 20 to 30 actions to finish.
- Long Tasks: Tasks that demand more than 30 actions to complete.

The VirtualHome set includes a total of 605 tasks, with 220 ultra-short tasks, 128 short tasks, 155 medium tasks, and 102 long tasks. Similarly, the Habitat 2.0 set comprises 904 tasks, distributed as 161 ultra-short tasks, 243 short tasks, 190 medium tasks, and 310 long tasks.

6 Experiments

6.1 Experimental Setup

For the VirtualHome set, we designate 218 tasks as the test set, with the remaining tasks serving as the training set. The Habitat 2.0 set also includes 120 test tasks. As our approach is unsupervised, we do not utilize the training set data for model training.

Evaluation Metrics To assess system efficacy, we employ success rate (SR) and goal conditions recall (GCR) (Singh et al., 2023) as our primary metrics. SR measures the proportion of executions where all key goal conditions (changing from the beginning to the end during a demonstration) are satisfied. GCR calculates the discrepancy between the expected and achieved end state conditions, relative to the total number of specific goal conditions needed for a task. A perfect SR score of 100% corresponds to achieving a GCR of 100%.

6.2 Comparison of Vision-Language Models

To assess the capabilities of various small-scale vision-language models (sLVLMs) on long-horizon task planning, we evaluated six models—InternVL2 8B (Chen et al., 2024c), Pixtral 12B (Agrawal et al., 2024), Qwen2-VL 7B (Wang et al., 2024a), Llama-3.2 11B (Dubey et al., 2024), InternVL2.5 8B (Chen et al., 2024b), and Qwen2.5-VL 7B (Team, 2025)—on our ExtendaBench benchmark in VirtualHome, spanning ultra-short to long tasks. Figure 4 presents the comparative performance in terms of GCR and SR across different task horizons. The results indicate that while all models perform well on ultra-short tasks, performance drops sharply as task complexity increases, with SR reaching 0% on long tasks for most models. Among them, Qwen2.5-VL 7B achieves the highest average GCR and SR, demonstrating the best overall performance in long-horizon task planning.

6.3 Comparison with Existing Methods

We compare SPO with existing long-horizon reasoning methods, including Chain-of-Thought (CoT) (Wei et al., 2022), as well as the multimodal VLM-based versions of Self-Rewarding (Yuan et al., 2024) and Iterative RPO (Pang et al., 2024), using Qwen2.5-VL 7B (Team, 2025) as the baseline. The evaluations are conducted on ExtendaBench in VirtualHome (Table 1) and Habitat (Table 2), covering tasks of increasing complexity

Table 2: Results using Qwen2.5-VL 7B as the baseline on different sets of our ExtendaBench in Habitat.

	Ultra-Short		Short		Medium		Long		Average	
Method	GCR	SR	GCR	SR	GCR	SR	GCR	SR	GCR	SR
Baseline	41.67	33.33	14.39	8.57	3.17	0	2.48	0	15.43	10.48
CoT (Wei et al., 2022)	42.36	50.00	9.49	8.57	7.51	0	6.10	0	16.36	14.64
Self-Rewarding (Yuan et al., 2024)										
Iteration 1	43.06	36.11	13.33	8.57	3.32	0	2.48	0	15.55	11.17
Iteration 2	43.06	33.33	14.19	8.57	4.34	0	2.48	0	16.01	10.48
Iteration 3	44.44	36.11	13.59	8.57	3.63	0	2.48	0	16.03	11.17
Iterative RPO (Pang et al., 2024)										
Iteration 1	45.14	36.11	12.90	8.57	3.43	0	2.86	0	16.08	11.17
Iteration 2	33.33	38.89	12.17	11.43	11.35	0	7.62	0	16.12	12.58
Iteration 3	38.54	44.44	15.06	8.57	10.49	0	6.76	0	17.71	13.25
SPO (1 iteration)	52.08	50.00	14.78	11.43	10.51	0	6.67	0	**21.01**	**15.36**

Table 3: Ablation studies of different modules in VirtualHome.

			Ultra-Short		Short		Medium		Long		Average	
Textual	**Image**	**Curriculum**	GCR	SR	GCR	SR	GCR	SR	GCR	SR	GCR	SR
✗	✗	✗	67.25	36.67	34.50	1.92	34.58	3.33	23.09	0	39.86	10.48
✓	✗	✗	71.70	43.33	45.52	9.62	30.57	1.67	16.71	0	41.13	13.65
✗	✓	✗	69.71	40.00	42.11	1.92	25.98	3.33	26.16	0	40.99	11.31
✓	✓	✗	70.52	43.33	43.89	7.69	33.96	3.33	27.90	0	44.07	13.59
✓	✓	✓	71.53	48.33	48.96	13.46	38.92	3.33	31.41	2.17	**47.71**	**16.83**

from Ultra-Short to Long.

ExtendaBench provides a structured evaluation protocol with graded task difficulty, enabling fine-grained analysis of model reasoning under increasing complexity. As shown in Tables 1 and 2, model performance consistently degrades on harder tasks, validating the benchmarks ability to reveal reasoning limitations across different methods.

Across both benchmarks, CoT improves over the baseline in Habitat, particularly on shorter tasks, highlighting the benefits of explicit reasoning in simpler settings. However, its performance drops notably on longer tasks, where it lacks structured planning capabilities. In VirtualHome, CoT offers limited gains and struggles with complex scenarios.

Self-Rewarding and Iterative RPO introduce iterative refinement, leading to moderate improvements on short and medium tasks. However, both methods fail to generalize to long-horizon planning, with SR dropping to 0% in long tasks across environments, indicating difficulties in maintaining coherent reasoning over extended sequences.

In contrast, SPO achieves the best overall performance in both VirtualHome and Habitat, outperforming all baselines. Notably, SPO delivers strong long-horizon reasoning without relying on iterative generation, achieving higher GCR and SR than Iterative RPO and Self-Rewarding across all difficulty levels.

6.4 Ablation Study

To evaluate the contributions of different components in SPO, we conduct an ablation study on VirtualHome, selectively removing textual coherence scoring, image awareness scoring, and curriculum-guided training. The results in Table 3 show that removing textual coherence scoring leads to the most significant performance drop, especially on short, medium, and long tasks, indicating its critical role in maintaining reasoning consistency. Removing image awareness scoring also results in a decline, particularly on long tasks, where integrating visual observations becomes more important. Without curriculum learning, performance on medium and long tasks deteriorates, demonstrating that progressive training helps the model handle more complex task sequences. The full SPO model achieves the highest performance, with 47.71% GCR and 16.83% SR, confirming that structured preference learning and curriculum-guided training together enable more effective long-horizon task planning.

7 Conclusion

We introduce Structured Preference Optimization (SPO), a method for improving long-horizon vision-language task planning through structured preference learning and curriculum-guided training. Unlike existing methods that struggle with multi-step decision-making, SPO systematically evaluates reasoning chains based on textual coherence and image awareness, ensuring high-quality reasoning and action selection. Additionally, curriculum-guided training progressively adapts the model from simpler to more complex tasks, enhancing generalization and robustness in long-horizon scenarios. To support research in this area, ExtendaBench provides a benchmark spanning VirtualHome and Habitat simulators with tasks of increasing difficulty. Experimental results show that SPO outperforms prior methods, particularly in long-horizon task planning, demonstrating improved reasoning consistency and decision-making accuracy.

Limitations

While our proposed Structured Preference Optimization (SPO) framework demonstrates strong performance in long-horizon task planning, it is currently implemented using smaller-scale vision-language models to enable efficient training and extensive experimentation. This design choice allows for faster iteration and detailed analysis but may not fully reflect the potential of SPO when applied to larger, more capable models. Extending the framework to larger-scale models remains an important direction for future work, as it could further enhance reasoning ability and task performance in complex embodied environments.

Acknowledgements

This work is supported by National Key Research and Development Program of China (2024YFE0203100), Scientific Research Innovation Capability Support Project for Young Faculty (No.ZYGXQNJSKYCXNLZCXM-I28), National Natural Science Foundation of China (NSFC) under Grants No.62476293, Nansha Key R&D Program under Grant No.2022ZD014, and General Embodied AI Center of Sun Yat-sen University.

References

Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Devendra Chaplot, Jessica Chudnovsky, Saurabh Garg, Theophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, et al. 2024. Pixtral 12b. *arXiv preprint arXiv:2410.07073*.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. 2023. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*.

Jorge A Baier, Fahiem Bacchus, and Sheila A McIlraith. 2009. A heuristic search approach to planning with temporally extended preferences. *Artificial Intelligence*, 173(5-6):593–618.

Vineet Bhat, Ali Umut Kaypak, Prashanth Krishnamurthy, Ramesh Karri, and Farshad Khorrami. 2024. Grounding llms for robot task planning using closed-loop state feedback. *arXiv preprint arXiv:2402.08546*.

Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. 2023. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pages 287–318. PMLR.

Daniel Bryce and Subbarao Kambhampati. 2007. A tutorial on planning graph based reachability heuristics. *AI Magazine*, 28(1):47–47.

Kehan Chen, Dong An, Yan Huang, Rongtao Xu, Yifei Su, Yonggen Ling, Ian Reid, and Liang Wang. 2024a. Constraint-aware zero-shot vision-language navigation in continuous environments. *arXiv preprint arXiv:2412.10137*.

Siwei Chen, Anxing Xiao, and David Hsu. 2023. Llm-state: Expandable state representation for long-horizon task planning in the open world. *arXiv preprint arXiv:2311.17406*.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. 2024b. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. 2024c. How far are we to gpt-4v? closing the gap to commercial

multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Yan Ding, Xiaohan Zhang, Saeid Amiri, Nieqing Cao, Hao Yang, Andy Kaminski, Chad Esselink, and Shiqi Zhang. 2023a. Integrating action knowledge and llms for task planning and situation handling in open worlds. *arXiv preprint arXiv:2305.17590*.

Yan Ding, Xiaohan Zhang, Chris Paxton, and Shiqi Zhang. 2023b. Task and motion planning with large language models for object rearrangement. *arXiv preprint arXiv:2303.06247*.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ben Eysenbach, Russ R Salakhutdinov, and Sergey Levine. 2019. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in Neural Information Processing Systems*, 32.

Richard E Fikes and Nils J Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208.

Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2020. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448.

Alex Havrilla, Sharath Raparthy, Christoforus Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, and Roberta Raileanu. 2024. Glore: When, where, and how to improve llm reasoning via global and local refinements. *arXiv preprint arXiv:2402.10963*.

Malte Helmert. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246.

Jörg Hoffmann. 2001. Ff: The fast-forward planning system. *AI magazine*, 22(3):57–57.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2022b. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Yiren Jian, Chongyang Gao, and Soroush Vosoughi. 2024. Bootstrapping vision-language learning with decoupled language pre-training. *Advances in Neural Information Processing Systems*, 36.

Songtao Jiang, Yan Zhang, Ruizhe Chen, Yeying Jin, and Zuozhu Liu. 2024. Modality-fair preference optimization for trustworthy mllm alignment. *arXiv preprint arXiv:2410.15334*.

Yiding Jiang, Shixiang Shane Gu, Kevin P Murphy, and Chelsea Finn. 2019. Language as an abstraction for hierarchical deep reinforcement learning. *Advances in Neural Information Processing Systems*, 32.

Yuqian Jiang, Shiqi Zhang, Piyush Khandelwal, and Peter Stone. 2018. Task planning in robotics: an empirical comparison of pddl-based and asp-based systems. *arXiv preprint arXiv:1804.08229*.

Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart J Russell, and Pieter Abbeel. 2018. Learning plannable representations with causal infogan. *Advances in Neural Information Processing Systems*, 31.

Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. 2022. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason Weston, and Mike Lewis. 2023. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023a. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE.

Xiwen Liang, Liang Ma, Shanshan Guo, Jianhua Han, Hang Xu, Shikui Ma, and Xiaodan Liang. 2023b. Cornav: Autonomous agent with self-corrected planning for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2306.10322*.

Yuan-Hong Liao, Xavier Puig, Marko Boben, Antonio Torralba, and Sanja Fidler. 2019. Synthesizing environment-aware activities via activity sketches.

In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6291–6299.

Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. 2023a. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual instruction tuning. *Advances in neural information processing systems*, 36.

Zeyi Liu, Arpit Bahety, and Shuran Song. 2023b. Reflect: Summarizing robot experiences for failure explanation and correction. *arXiv preprint arXiv:2306.15724*.

Xiaofei Lu. 2010. Automatic analysis of syntactic complexity in second language writing. *International journal of corpus linguistics*, 15(4):474–496.

Suraj Nair and Chelsea Finn. 2019. Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation. *arXiv preprint arXiv:1909.05829*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

OpenAI. 2024. Hello gpt-4o. Accessed: 2024-05-26.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *arXiv preprint arXiv:2404.19733*.

Archiki Prasad, Weizhe Yuan, Richard Yuanzhe Pang, Jing Xu, Maryam Fazel-Zarandi, Mohit Bansal, Sainbayar Sukhbaatar, Jason Weston, and Jane Yu. 2024. Self-consistency preference optimization. *arXiv preprint arXiv:2411.04109*.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502.

Jielin Qiu, Mengdi Xu, William Han, Seungwhan Moon, and Ding Zhao. 2023. Embodied executable policy learning with language-based scene summarization. *arXiv preprint arXiv:2306.05696*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. 2023. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. *arXiv preprint arXiv:2307.06135*.

Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. 2024. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020a. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020b. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B Tenenbaum, Leslie Pack Kaelbling, and Michael Katz. 2023. Generalized planning in pddl domains with pretrained large language models. *arXiv preprint arXiv:2305.11014*.

Tom Silver, Varun Hariprasad, Reece S Shuttleworth, Nishanth Kumar, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2022. Pddl planning with pretrained large language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE.

Marta Skreta, Naruki Yoshikawa, Sebastian Arellano-Rubach, Zhi Ji, Lasse Bjørn Kristensen, Kourosh Darvish, Alán Aspuru-Guzik, Florian Shkurti, and Animesh Garg. 2023. Errors are useful prompts: Instruction guided task programming with verifier-assisted iterative prompting. *arXiv preprint arXiv:2303.14100*.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. 2018. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning*, pages 4732–4741. PMLR.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. 2021. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266.

Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. 2023. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Qwen Team. 2025. Qwen2.5-vl.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*.

Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. 2023. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20.

Naoki Wake, Atsushi Kanehira, Kazuhiro Sasabuchi, Jun Takamatsu, and Katsushi Ikeuchi. 2023. Chatgpt empowered long-step robot control in various environments: A case application. *arXiv preprint arXiv:2304.03893*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024a. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. 2024b. Self-taught evaluators. *arXiv preprint arXiv:2408.02666*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. 2023. Tidybot: Personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*.

Yaqi Xie, Chen Yu, Tongyao Zhu, Jinbin Bai, Ze Gong, and Harold Soh. 2023. Translating natural language to planning goals with large-language models. *arXiv preprint arXiv:2302.05128*.

Danfei Xu, Roberto Martín-Martín, De-An Huang, Yuke Zhu, Silvio Savarese, and Li F Fei-Fei. 2019. Regression planning networks. *Advances in Neural Information Processing Systems*, 32.

Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. 2018. Neural task programming: Learning to generalize across hierarchical tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3795–3802. IEEE.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. 2023. Some things are more cringe than others: Preference optimization with the pairwise cringe loss. *arXiv preprint arXiv:2312.16682*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. 2023. mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*.

Bowen Zhang and Harold Soh. 2023. Large language models as zero-shot human models for human-robot interaction. *arXiv preprint arXiv:2303.03548*.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and Wang He. 2024a. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*.

Kaidong Zhang, Rongtao Xu, Pengzhen Ren, Junfan Lin, Hefeng Wu, Liang Lin, and Xiaodan Liang. 2025. Robridge: A hierarchical architecture bridging cognition and execution for general robotic manipulation. *arXiv preprint arXiv:2505.01709*.

Xuan Zhang, Chao Du, Tianyu Pang, Qian Liu, Wei Gao, and Min Lin. 2024b. Chain of preference optimization: Improving chain-of-thought reasoning in llms. *arXiv preprint arXiv:2406.09136*.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2024. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems*, 36.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Tan Zhi-Xuan, Lance Ying, Vikash Mansinghka, and Joshua B Tenenbaum. 2024. Pragmatic instruction following and goal assistance via cooperative language-guided inverse planning. *arXiv preprint arXiv:2402.17930*.

Shengli Zhou, Xiangchen Wang, Jinrui Zhang, Ruozai Tian, Rongtao Xu, and Feng Zheng. 2025. p^3: Toward versatile embodied agents. *arXiv preprint arXiv:2508.07033*.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. 2025. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*.

A More Details for ExtendaBench

A.1 Statistics

A.1.1 Overview

Table 4 provides a summary of key characteristics of the VirtualHome and Habitat datasets in our ExtendaBench, highlighting differences in scene complexity, task variety, and action requirements. The VirtualHome dataset consists of 7 distinct scenes with a total of 390 objects, supporting 294 task types across 605 instructions. In VirtualHome, the simulator provides 16 unique executable actions, enabling a broader range of task interactions. In contrast, the Habitat dataset features 105 scenes with 82 distinct objects, enabling 20 task types across 904 instructions. The Habitat simulator supports 6 unique executable actions.

A.1.2 Data Distribution Across Sets

VirtualHome For the VirtualHome dataset, tasks are categorized into ultra short, short, medium, and long. Each category includes a portion reserved for testing, with the remaining used for training. The distribution is as follows:

Table 4: Overview of scene and task characteristics in VirtualHome and Habitat.

	VirtualHome	Habitat
Scene Number	7	105
Scene Objects	390	82
Task Type	294	20
Instructions	605	904
Action Number	16	6

- Ultra short: This category contains 220 tasks in total, with 46 allocated for testing and 174 for training.
- Short: A total of 128 tasks, with 60 reserved for testing and 68 for training.
- Medium: Comprising 155 tasks, with 52 for testing and 103 for training.
- Long: The most complex category, including 102 tasks in total, with 60 allocated for testing and 42 for training.

Habitat For Habitat, the dataset is similarly divided into four categories based on task length: ultra short, short, medium, and long. For each category, a portion of the tasks is allocated for testing, and the remaining are used for training. The details are as follows:

- Ultra short: This category contains 161 tasks, with 36 reserved for testing and 125 for training.
- Short: There are 243 tasks, of which 35 are for testing and 208 for training.
- Medium: A total of 190 tasks, including 31 for testing and 159 for training.
- Long: The largest category, comprising 310 tasks, with 30 allocated for testing and 280 for training.

A.1.3 Word Frequency Distribution

Figure 5 presents the top 50 most frequent words, excluding prepositions, in the datasets generated for VirtualHome and Habitat environments. Subfigure (a) shows the word frequencies from VirtualHome, highlighting terms associated with common objects and actions, such as “table,” “kitchen,” and “place,” reflecting its simulation of domestic scenarios. Subfigure (b) illustrates the word

frequencies for Habitat, where terms like "from," "counter," and "cup" dominate, indicating tasks involving object interaction and spatial relationships.

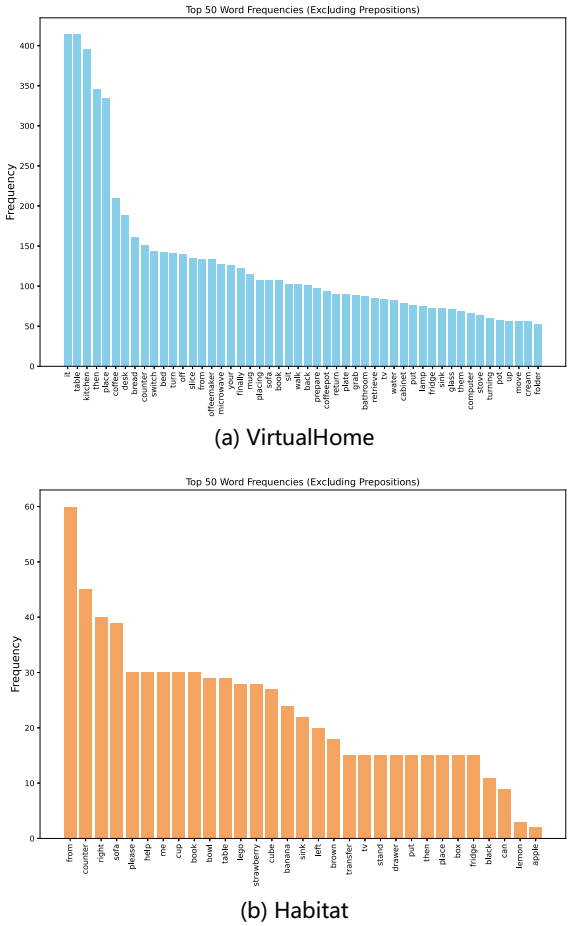

Figure 5: Word frequency analysis for ExtendaBench.

A.1.4 Action Lengths

Figure 6 splots primitive-action lengths and underscores the benchmarks long-horizon nature: in VirtualHome the training split already ranges broadly (mean 14.8 actions) with a heavy tail extending to 58 steps, while an extreme task is held out for testing to enforce horizon extrapolation; Habitat pushes lengths even higher—training tasks centre around 2035 actions (mean 21.0) and the test split, though slightly shorter on average (17.7), still requires multi-dozen-step plans—so across both environments the majority of tasks demand extended, sequential reasoning, and all subsequent results are reported per environment and split to reveal model performance along this length-generalisation axis.

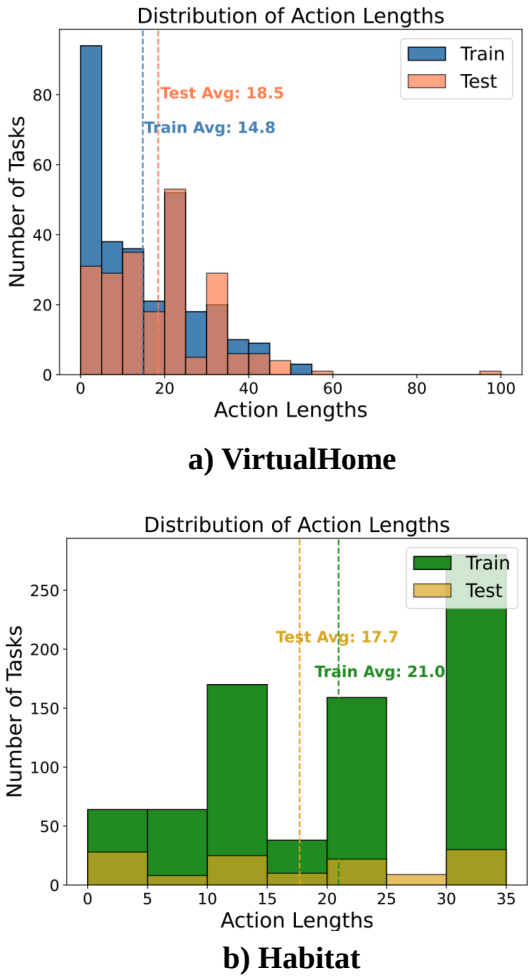

Figure 6: Distribution of action lengths in our benchmark.

A.2 Task Complexity

A.2.1 Correlation between Action Length and Task Complexity

Longer action chains inevitably accumulate execution error: under the all-or-nothing success metric, a single early slip invalidates the entire trajectory, so the chance of finishing a plan falls sharply as its length grows. In the VirtualHome portion of ExtendaBench, the split-wise statistics as in Table 5 confirm this effect: average plan length stretches from 7.48 to 36.11 steps, instruction length from 18.1 to 61.7 tokens, and the syntactic-complexity score from 0.21 to 0.29, indicating deeper clause nesting and a larger set of entities that must be tracked. Empirically, the same Qwen2.5-VL-7B baseline that achieves 42% GCR (33% SR) on ultra-short VirtualHome tasks manages only 2% GCR (0% SR) on long ones, underscoring how these structurally richer instructions translate into far tougher planning problems. Taken together—

error-accumulation theory, the monotonic rise in structural and linguistic complexity, and the observed performance cliff—demonstrate that action length is a well-grounded proxy for task difficulty within ExtendaBench.

A.2.2 Comparison of Task Complexity across Datasets

ExtendaBench demonstrates significant advantages across multiple key metrics as in Table 6. Its average action length (18.60) substantially exceeds that of other datasets (ALFRED: 6.71, LLarp: 4.40, VirtualHome: 9.83), with the long-horizon subset reaching an average of 35.56 steps—highlighting its unique value for long-horizon reasoning tasks. Additionally, ExtendaBench exhibits the highest instruction length (31.15 tokens on average) and syntactic complexity score (0.31) among all compared datasets, indicating that its task instructions are both structurally complex and semantically rich.

B More Details for Experiments

B.1 Experimental Setup

For data generation, we produce $K = 5$ responses per prompt, employing a sampling temperature of 0.7 and a top-p value of 0.95. The generated dataset is then used to train the model for 3 epochs. During training, the learning rate is set to $2e{-}5$. For LoRA, we use a rank value of 16, an alpha parameter of 32, and a dropout rate of 0.05. Unlike the Self-Rewarding framework, which involves iterative training where the trained model is used to re-label data and retrain in a loop, our approach trains the model only once, simplifying the training process while maintaining effectiveness.

B.2 Comparison of Vision-Language Models

In addition to the models discussed in Section 6.2, we also compare InternVL3 8B (Zhu et al., 2025) to assess whether Qwen2.5-VL 7B remains competitive despite having fewer parameters. As shown in Table 7, Qwen2.5-VL 7B achieves the highest average SR on both VirtualHome (11.99% vs. 11.28%) and Habitat (10.48% vs. 9.78%), and matches or outperforms InternVL3-8B on 11 of 16 split-metric pairs. While InternVL3-8B shows notable gains on medium/long tasks, Qwen2.5-VL 7B demonstrates more stable performance across environments and task lengths, confirming its reliability as a backbone model.

B.3 Effect of Preference Pair Selection

To assess the impact of our preference pair selection strategy, we perform an ablation study using the Textual Coherence model (corresponding to row 2 in Table 3). As shown in Table 8, enabling pair selection improves GCR from 40.04% to 41.13% and SR from 11.73% to 13.65%, yielding gains of +1.09 and +1.92 percentage points, respectively. These results indicate that structured preference selection contributes to more accurate decision-making by guiding the model with more informative comparisons.

B.4 Impact of Evaluator Strength

To quantify how the capacity of the external evaluator influences learning, we replaced the default Qwen2.5-VL 7B assessor with GPT-4o and retrained under otherwise identical settings. As reported in Table 9, the stronger evaluator yields consistent gains on the VirtualHome benchmark, improving GCR from 47.71% to 49.62% and SR from 16.83% to 19.26%. These results confirm that higher-quality evaluators provide more informative preference signals, which in turn translate into better long-horizon task performance.

B.5 Score Combination Strategies

As described in Section 4.1.1, we explore two approaches for combining the textual coherence score (S_{text}) and image awareness score (S_{image}). The first, shown in Equation 4, uses a weighted sum with tunable weights (w_1, w_2). The second, described in Equation 5, adopts a direct scoring approach, where the model is guided to first assess task alignment and image utilization independently, and then produce an overall score that integrates both aspects, following the prompt described in Section C.2. This approach avoids manual weighting and achieves better empirical performance. Table 10 compares the two approaches, showing that direct scoring achieves the best performance among the tested settings and does not require manual tuning of combination weights. We hypothesize two main reasons:

- In a weighted sum, a chain that writes a very clear rationale can still receive a good overall score even if it references an object that does not exist in the image—the high text score masks the low image score. The direct-scoring judge, by contrast, looks at text and image evidence together and gives a high

Table 5: Correlation between action length and linguistic complexity across task difficulty levels in the VirtualHome subset of ExtendaBench.

	Avg Action Length↑	Avg Instruction Length↑	Avg Syntax Score (Lu, 2010)↑
Ultra-Short	7.48	18.14	0.21
Short	12.97	28.84	0.26
Medium	22.46	43.96	0.28
Long	36.11	61.71	0.29

Table 6: Comparison of action, instruction, and language complexity across datasets.

Dataset	Avg Action Length↑	Avg Instruction Length↑	Avg Syntax Score (Lu, 2010)↑
ALFRED (Shridhar et al., 2020a)	6.71	9.26	0.19
LLarp (Szot et al., 2023)	4.40	10.59	0.20
VirtualHome (Puig et al., 2018)	9.83	19.11	0.18
ExtendaBench (Ours)	**18.60**	**31.15**	**0.31**

mark only when the reasoning is both logically consistent and visually grounded. This tighter evaluation criterion may explain the performance improvements reported for direct scoring.

- Optimal weights shift with task composition; small changes in w_1, w_2 can flip the ranking (SR spans from 9.17% to 13.88%). Direct scoring removes this hyper-parameter altogether, providing a stable criterion that generalises across lengths and environments.

B.6 Effectiveness of Curriculum Learning

Our curriculum consists of four sequential stages aligned with progressively increasing task complexity. To quantify the contribution of each curriculum stage, we evaluate model performance cumulatively after completing each stage, with each stage initialized from the checkpoint obtained in the previous one. Results in Table 11 show consistent improvements from Stage 1 through Stage 4, with GCR rising from 41.88% to 47.71% and SR improving from 11.80% to 16.83%. These gains highlight the effectiveness of our curriculum strategy in systematically enhancing model capabilities on complex long-horizon tasks.

B.7 Comparison between SPO and Single-negative DPO

We ran standard single-negative DPO with each of our three negative-pair rules in isolation and compared them to SPO, which retains all three rules simultaneously. The results in Table 12 show that every single-rule variant improves over the vanilla model to a similar extent (e.g., low-quality reasoning, same action reaches 44.88 GCR / 15.32 SR), yet none matches the full SPO configuration: combining the three complementary negatives pushes performance to 47.71 GCR / 16.83 SR, a further gain of +2.8 GCR and +1.5 SR over the best single-rule baseline. This demonstrates that the additional negative categories capture distinct failure modes and that their union provides the strongest learning signal, clarifying the specific benefit of our multi-type sampling strategy over conventional single-negative DPO.

B.8 Comparison with Existing LLM-based Methods

Our task formulation assumes that the agent receives only image observations and textual instructions as input, requiring it to infer actions solely from visual context without relying on externally provided ground-truth object identities or positions. In contrast, prior frameworks such as SayCan (Brohan et al., 2023) and ProgPrompt (Singh et al., 2023) depend on explicit object-level annotations, making them less suited to realistic embodied environments. To facilitate meaningful comparison, we adapted both methods by restricting inputs to task instructions and raw visual observations, removing access to ground-truth environment information. Table 13 shows that under these consistent input constraints, our approach significantly outperforms SayCan and ProgPrompt on the VirtualHome benchmark.

The observed performance gap arises due to in-

Table 7: Performance comparison between Qwen2.5-VL 7B and InternVL3 8B.

		Ultra-Short		Short		Medium		Long		Average	
		GCR	SR	GCR	SR	GCR	SR	GCR	SR	GCR	SR
VirtualHome	Qwen2.5-VL 7B	57.32	35.00	42.72	9.62	30.57	3.33	27.47	0	39.52	11.99
	InternVL3 8B	65.83	33.33	41.94	5.77	33.26	4.35	29.20	1.67	42.56	11.28
Habitat	Qwen2.5-VL 7B	41.67	33.33	14.39	8.57	3.17	0	2.48	0	15.43	10.48
	InternVL3 8B	33.68	30.56	12.36	8.57	6.23	0	2.48	0	13.69	9.78

Table 8: Average performance of preference pair selection strategy in VirtualHome.

pair selection	GCR	SR
✗	40.04	11.73
✓	41.13	13.65

Table 9: Compare with methods using different evaluators in VirtualHome environment.

evaluator	GCR	SR
Qwen2.5-VL 7B	47.71	16.83
GPT-4o	49.62	19.26

herent limitations of these approaches when restricted to our realistic input scenario. SayCan, which relies on scoring all potential atomic actions with a language model, struggles with the combinatorial explosion of action-object pairs (over 2,000 possibilities) in our setting, causing inefficiencies and unreliable scoring—particularly with smaller models. ProgPrompt also faces challenges: firstly, it was originally designed to leverage large language models such as GPT3, whereas our setup employs a smaller vision-language model with constrained structured reasoning and code generation capabilities; secondly, smaller models inherently struggle with long-context comprehension, hindering their ability to generate coherent, visually-grounded multi-step programs aligned with historical context.

B.9 Details for High-Quality Reasoning Selection

For each task instance, all reasoning chains are first ranked by their overall scores. We identify the subset of chains that achieve the highest score. If multiple top-scoring responses result in different final actions, we select the one with the most frequently occurring action as the preferred output. The remaining top-scoring variants are treated as high-quality reasoning samples with differing actions.

Table 10: Compare with methods using different combinations in the VirtualHome environment.

		GCR	SR
weighted sum			
w_1=1.0	w_2=1.0	41.66	9.17
w_1=1.0	w_2=0.8	43.13	11.73
w_1=1.0	w_2=0.5	41.59	13.46
w_1=0.8	w_2=1.0	45.25	13.88
w_1=0.5	w_2=1.0	41.81	11.99
direct scoring		47.71	16.83

Table 11: Comparison of different stages in curriculum learning in the VirtualHome environment.

	GCR	SR
stage 1	41.88	11.80
stage 2	43.21	14.78
stage 3	45.98	14.90
stage 4	47.71	16.83

In cases where only a single top-scoring response exists, we also include reasoning chains whose scores fall within a margin of 0.1 points from the maximum and lead to different final actions. These are likewise categorized as high-quality reasoning but are used as negative examples during training, as their final actions deviate from the preferred one. This distinction encourages the model to differentiate between logically coherent reasoning and correct decision-making.

This combination of threshold-based filtering and action consistency selection helps reduce the bias in self-assessment and prevents the model from over-optimizing for reasoning fluency alone.

Table 12: Comparison between SPO and standard single-negative DPO in VirtualHome.

	GCR	SR
High-quality reasoning, different action	44.52	14.01
Low-quality reasoning, different action	44.12	14.07
Low-quality reasoning, same action	44.88	15.32
SPO (three rules together)	47.71	16.83

Table 13: Comparison with SayCan and ProgPrompt in the VirtualHome environment.

	GCR	SR
SayCan (Brohan et al., 2023)	40.80	10.83
ProgPrompt (Singh et al., 2023)	39.64	9.17
Ours	47.71	16.83

B.10 SPO vs. GRPO with Structured Preference Rewards

To keep training cost tractable we ran all GRPO (Shao et al., 2024) baselines with the smaller Qwen2.5-VL 3B backbone on Habitat (as shown in the Table 14). Our SPO (DPO-style) already surpasses vanilla GRPO (+4.38 pp GCR, +1.41 pp SR). When GRPO is augmented with the same Structured Preference rewards, its score rises further to 20.18 / 16.07, narrowing the margin to SPO to $\leq$ 1.4 pp. This result shows (i) Structured Preference signals are the key performance driver—whichever optimisation is used—and (ii) our DPO-based SPO achieves comparable performance with a simpler, resource-efficient training loop.

Table 14: Compare with GRPO-based methods using Qwen2.5-VL 3B as the base model in Habitat environment.

	GCR	SR
baseline	12.91	11.17
SPO	19.83	14.66
GRPO	15.45	13.25
GRPO w/ Structured Preference rewards	20.18	16.07

B.11 Results on other Planning Datasets

We centred our study on ExtendaBench because, unlike prior embodied-planning benchmarks, it spans multiple difficulty tiers—ultra-short to long (up to 60 primitive actions) within a single RGB-grounded suite. This breadth of plan complexity is exactly the setting for which Structured Preference Optimisation (SPO) was designed. To give an additional reference point, we ran a quick check on ALFRED: using only 10% of the ALFRED split, the Qwen2.5-VL-7B baseline reaches 7.21% SR, whereas our SPO model attains 12.02% SR. The gain, achieved with minimal tuning, suggests that our structured-preference signal transfers beyond ExtendaBench.

B.12 Visualization

To showcase the diversity and progressive difficulty of tasks in VirtualHome and Habitat 2.0, we present representative visualizations across four difficulty levels: ultra-short, short, medium, and long (Figures 7-10, 11-14). Tasks are categorized based on action sequence length—a practical proxy for planning complexity.

Figures 7-10 illustrate VirtualHome tasks ranging from simple navigation to complex, multi-step meal preparation. Figures 11-14 show corresponding Habitat tasks that progress from basic object transfers to extensive spatial rearrangements.

To further highlight the challenges of long-horizon reasoning, we include two additional long task examples in Habitat (Figures 15 and 16), featuring diverse object types, complex layouts, and longer planning sequences.

These visualizations confirm that our benchmark enables structured, fine-grained evaluation across a spectrum of embodied reasoning difficulties.

B.13 Case Study

Visual grounding failures occasionally occur due to the limitations of the small VLM backbone (Qwen2.5-VL 7B). Common issues include misidentifying small or partially occluded objects (e.g., mistaking a wrench for a spoon as in Figure 17) or failing to attend to relevant scene regions. Our structured scoring mechanism—based on textual coherence and image awareness—encourages better alignment between reasoning and visual input, which helps mitigate such errors during training.

A common failure in long-horizon tasks is historical inconsistency—where the model fails to consider previously completed steps. As shown in Figure 18, the CoT baseline incorrectly suggests walking to the stove again, ignoring that the salmon has already been baked. In contrast, our method correctly interprets the execution history and proceeds to the next relevant subtask. This

Figure 7: Generated task example in VirtualHome (ultra short).

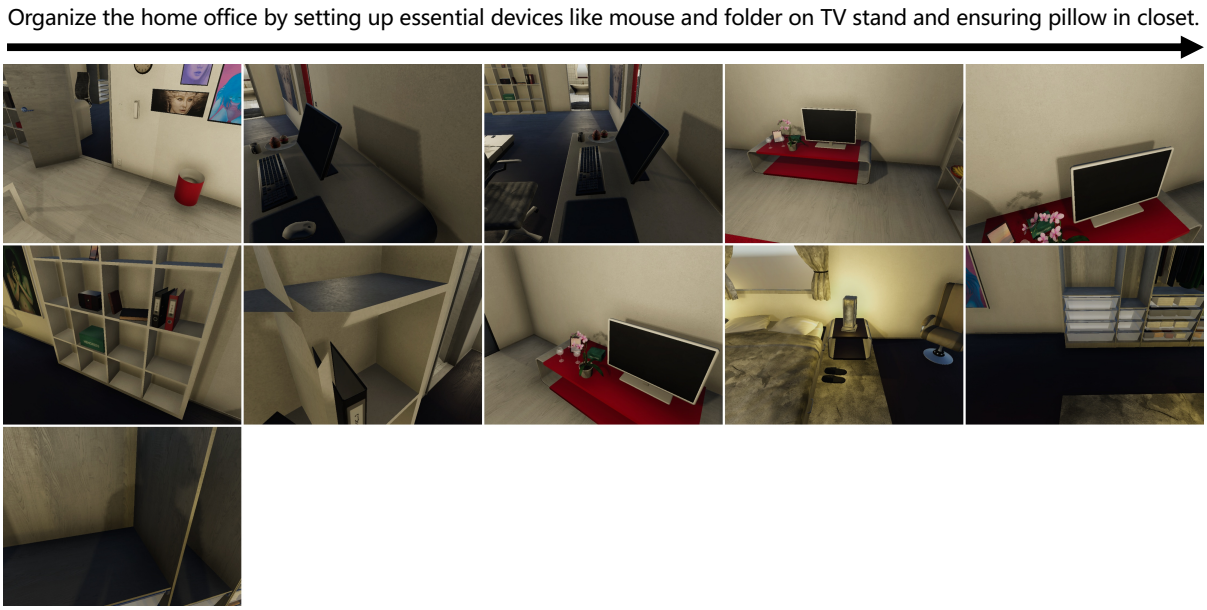

Figure 8: Generated task example in VirtualHome (short).

demonstrates more coherent and context-aware reasoning.

C Prompts

C.1 Prompts for Generating Data

C.1.1 VirtualHome

Task Proposal

Follow these steps to generate your answer:
1. Think about the task generation:
- Design a task with more than 30 sequential steps.
- Use only actions from the "HUMAN ACTION LIST" and objects from the "OBJECT LIST."
- Ensure the task involves at least 12 distinct objects from the "OBJECT LIST."
2. Provide a detailed task description:
- Output a comprehensive description of the task.
- Include all subtasks and the required objects.
3. Decompose the task step by step:
- Break the task into individual steps.
- After completing each step, analyze and output what needs to be done next.
- Include reasoning for each subsequent step before outputting it.
Important rules:
- You have only two hands. Each time you grab an object, one hand becomes unavailable until you put the object back.
- Track the number of free hands after each action. Ensure you have at least one free hand before interacting with any object.
- Use only actions from the "HUMAN ACTION LIST" and objects from the "OBJECT LIST."
- The task must maintain a strong sequential

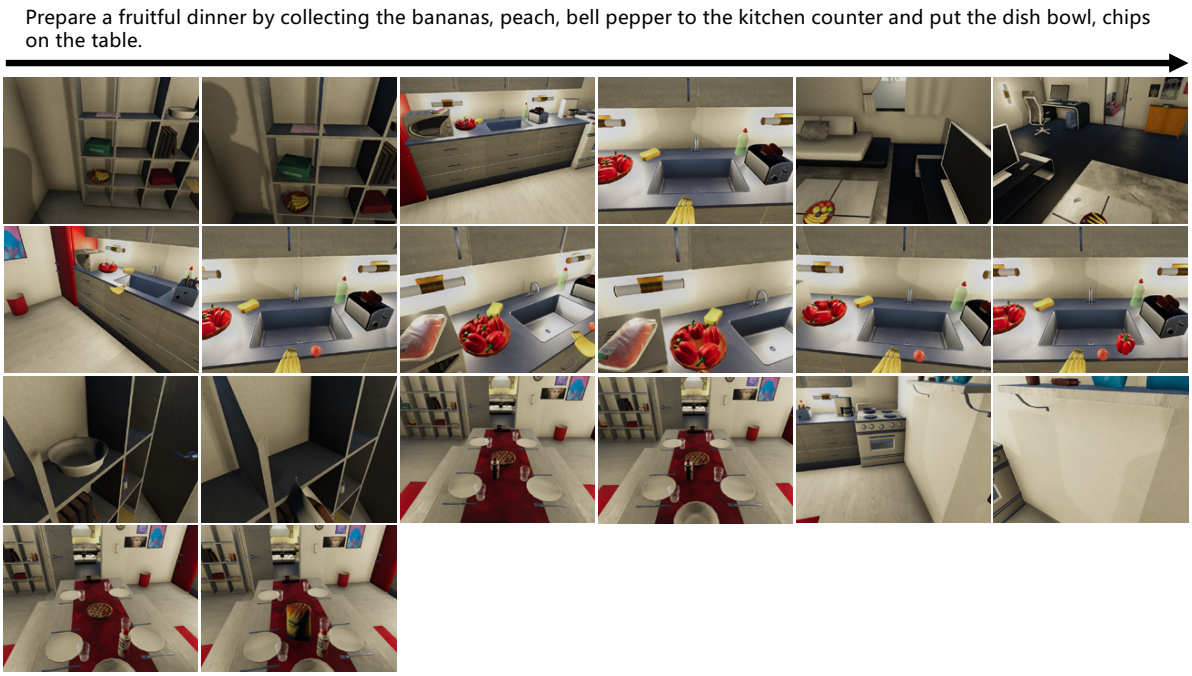

Figure 9: Generated task example in VirtualHome (medium).

relationship between its decomposed steps, ensuring logical and coherent progression.

Review

Follow these steps to verify the given task and decomposed steps step by step.

- Think about whether the task description is detailed enough to make it clear to a household agent what needs to be done, including every objects in decomposed steps. Give your reasons for this as well as your answer, if the answer is no, give a more detailed description of the task.
- Think and output the reasons why each step is necessary to complete the task.
- Think and output that each step is coherent with a necessary back-and-forth relationship between them.
- Think and output the reasons why the decomposed steps accomplish the task.
- verify the actions in decomposed steps only come from "HUMAN ACTION LIST."
- verify the objects in decomposed steps only come from "OBJECT LIST." The inclusion of any additional objects or locations is strictly prohibited.
- Think and output the reasons why each step make common sense.
- verify that each step is compliant with the rule of [walk] object before interacting with it.

If the verification passes, return true, otherwise return false and then give your adjustment.

Refinement

This is the feedback and observation based on your steps that have been executed:
[feedback]
<image>

Please perform the following steps based on the feedback:
1. Please think about and output the reason why the steps failed to execute.
2. Based on the reasons why the steps failed, think about and output the reasons why this task is feasible given the rules, and output yes or no.

Figure 10: Generated task example in VirtualHome (long).

3. if the task is feasible, output your modifications to the failed step.

C.1.2 Habitat

Template Proposal

You are a robot task generator that can generate robot task templates of different lengths based on given robot actions and examples.

The actions you can use include:
1.nav(obj or receptacle) is used by the robot to navigate to the corresponding object or receptacle
2.pick(obj) is used by the robot to grab an object
...

Rules:
1. You need to output five parts, including instructions, task planning, replaceable objects and target states.
2. If the object or receptacle in the instructions and task planning can be replaced, use plus pronouns to replace it.

Figure 11: Generated task example in Habitat (ultra short).

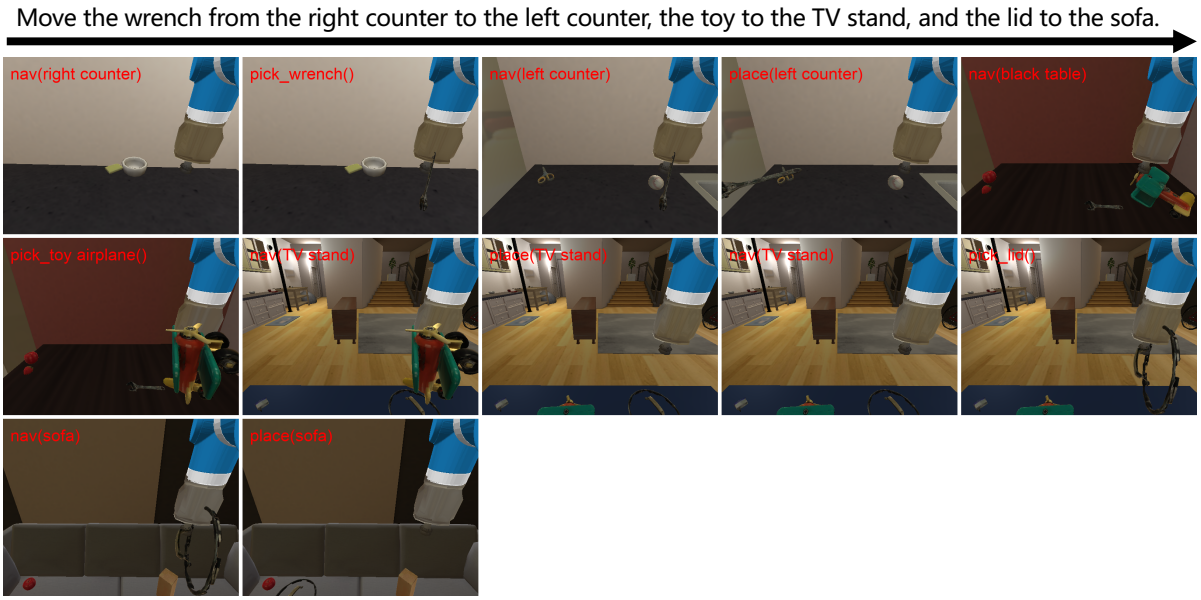

Figure 12: Generated task example in Habitat (short).

Instruction Augmentation

You are a task instruction rewriter, and you can rewrite and expand the robot's task instructions according to the given rewriting rules.

Rules:
1. You can use the verbs of the task instructions Use synonyms to replace, for example, change move to reposition.
2. You can replace the objects used in the task instructions, replace the objects with corresponding colors or appearance descriptions, such as changing apple to a red round Fruit.
3. Add some context descriptions, for example, in "Please put an apple on the table for me," change it to "I want to eat an apple, please put an apple on the table for me" to make the instruction longer.

Now Please help me rewrite the following instructions:

C.2 Prompts for Preference Evaluation

Preference Evaluation

You are an evaluation system designed to assess how well a reasoning chain (CoT) aligns with the task instruction and how effectively it utilizes the current image observation.

Given a task instruction, a reasoning chain (CoT), past execution history, and an RGB image observation, your task is to evaluate:
1. **Task Alignment Score**: How well the reasoning chain follows the task instruc-

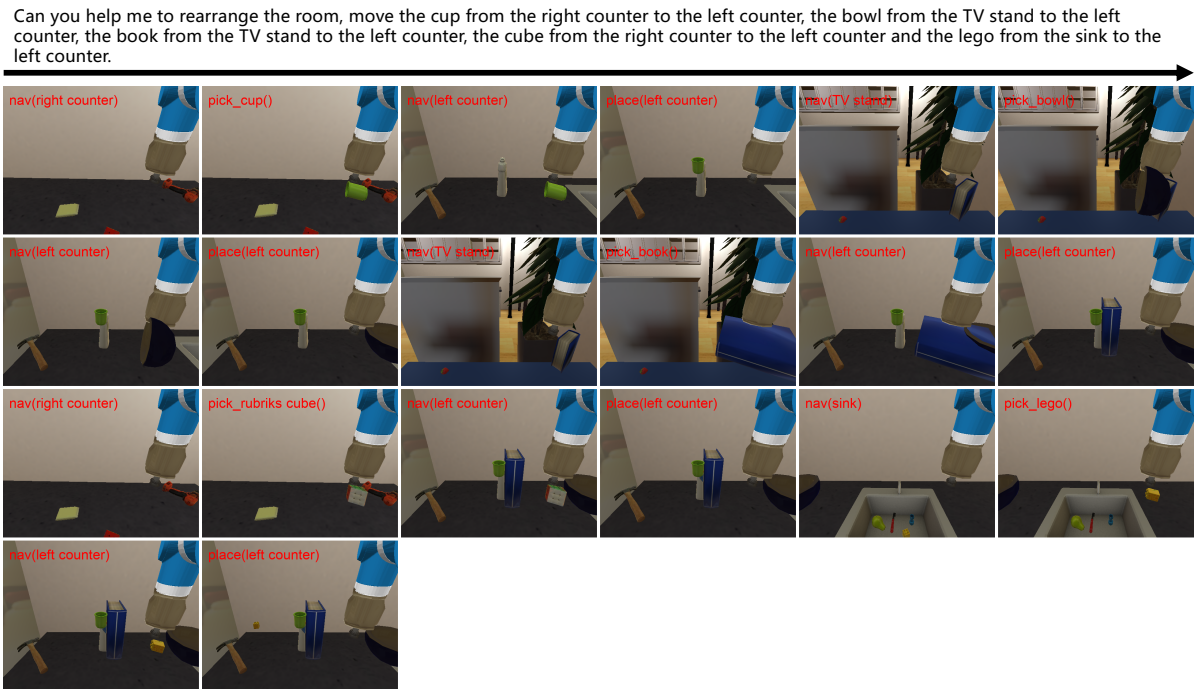

Figure 13: Generated task example in Habitat (medium).

tion and previous history.
2. **Image Utilization Score**: How well the reasoning chain leverages the current image observation to infer the next step.
3. **Overall Score**: A final score that summarizes the overall quality of the reasoning chain, considering both task alignment and image utilization.

Input Data:
- **Task Instruction:** {INSTR}
- **Chain of Thought (CoT):** {REASON}
- **Previous Execution History:** {HISTORY}
- **Current Image Observation (RGB):** <image>

Output Format:
Return the three scores in the following format:
Task Alignment Score: X
Image Utilization Score: Y
Overall Score: Z
Where **X**, **Y**, and **Z** are numbers between **0 and 1**.

D License

The dataset is published under CC BY-NCSA 4.0 license, which means everyone can use this dataset for non-commercial research purposes.

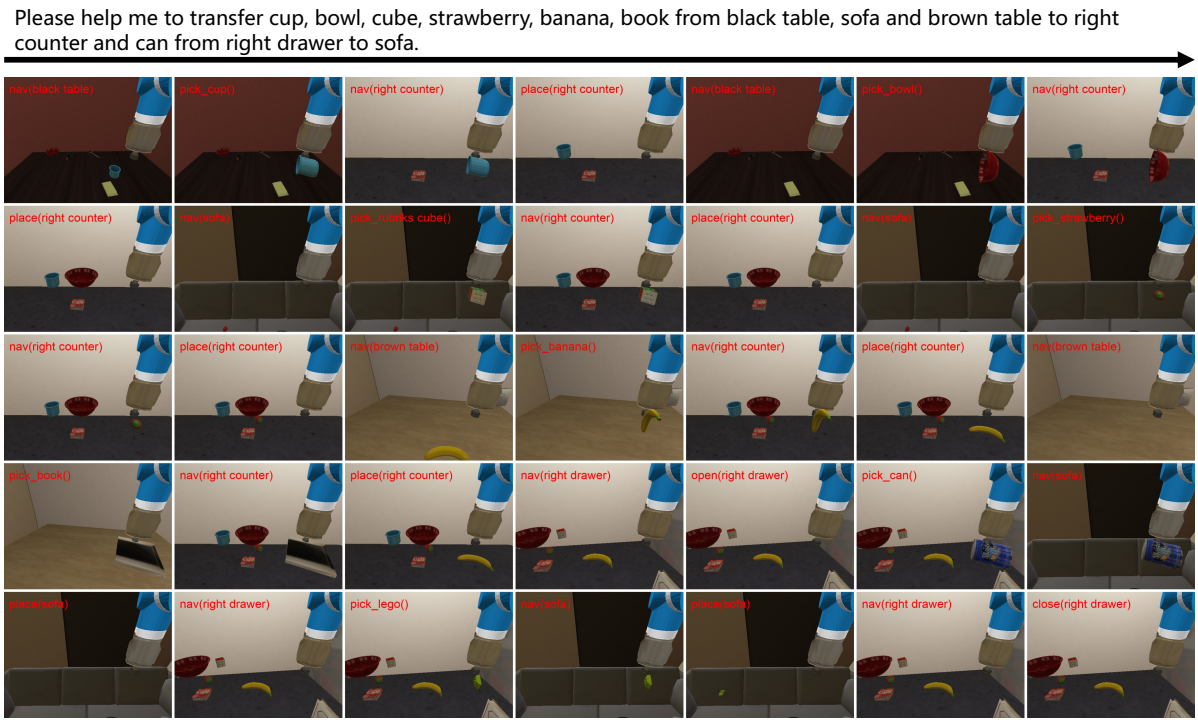

Figure 14: Generated task example in Habitat (long).

Figure 15: Generated task example in Habitat.

Figure 16: Generated task example in Habitat.

Predicted action: pick_spoon() ✖

Figure 17: Example of visual grounding failure due to object misidentification.

Instruction: Prepare salmon by baking in the stove, heat creamy buns in the microwave, then set bananas and a peach on the kitchen table; ...
History: ... [open] <stove> ... [place] <salmon> ... [close] <stove>

CoT:
Reason: The task requires baking salmon in the stove. Therefore, the next step is to walk to the stove.
Next plan: [walk] <stove>

Ours:
Reason: The salmon has been baked as required. I am now in front of the stove. The next step is to find the creamy buns and go to the microwave to heat them.
Next plan: [walk] <buns>

Figure 18: Comparison between baseline CoT and our method on reasoning consistency.