Beyond Needle(s) in the Embodied Haystack: Environment, Architecture, and Training Considerations for Long Context Reasoning

Bosung Kim
University of California, San Diego
bosungkim@ucsd.edu

Prithviraj Ammanabrolu
University of California, San Diego
prithvi@ucsd.edu

Abstract

We introduce ∞-THOR*, a new framework for long-horizon embodied tasks that advances long-context understanding in embodied AI.* ∞-THOR *provides: (1) a generation framework for synthesizing scalable, reproducible, and unlimited long-horizon trajectories; (2) a novel embodied QA task, Needle(s) in the Embodied Haystack, where multiple scattered clues across extended trajectories test agents' long-context reasoning ability; and (3) a long-horizon dataset and benchmark suite featuring complex tasks that span hundreds of environment steps, each paired with ground-truth action sequences. To enable this capability, we explore architectural adaptations, including interleaved Goal-State-Action modeling, context extension techniques, and Context Parallelism, to equip LLM-based agents for extreme long-context reasoning and interaction. Experimental results and analyses highlight the challenges posed by our benchmark and provide insights into training strategies and model behaviors under long-horizon conditions. Our work provides a foundation for the next generation of embodied AI systems capable of robust, long-term reasoning and planning.*

1. Introduction

Real-world embodied reasoning is a sequential decision-making problem requiring long-horizon planning, where task success depends on both memorizing and reasoning over multiple events that occur far apart in time. Using large pre-trained vision-language-action (VLA) models as policies for such tasks requires surpassing the key challenge of *long-context* reasoning. We seek to answer questions pertaining to what design choices matter in terms of environments, model architectures, and training methods when using VLA models for long-horizon embodied tasks. To this end, we develop a new framework for long-horizon tasks designed to push the boundaries of long-context understanding in embodied AI.

We introduce ∞-THOR, a new framework for generation, training, and evaluation of long-horizon embodied tasks. Our benchmark uniquely features tasks with a synthetic final goal, which involves multiple objects that appear at distant time steps, requiring multi-step reasoning across over hundreds of steps. Figure 1 illustrates an example: the agent observes the tomato at an early step (t=17) and the counter top much later (t=560). Then, the final task is given at t=670, which requires the agent to place the tomato on the counter top. This setup highlights the challenge of long-horizon dependency, where key objects and locations must be remembered and acted upon after hundreds of steps.

This long-horizon setup introduces a new challenging task, Needle(s) in the Embodied Haystack (NiEH). Unlike the standard Needle in a Haystack task [19], which focuses on recalling a single clue in text, NiEH poses two main challenges: (1) multiple scattered clues (Needles) and (2) multi-modal inputs that combine visual and linguistic observations from the environment (Embodiment). This task is designed to evaluate the agent's ability to recall and reason about previously encountered environmental details, such as identifying objects and recalling performed actions.

Going beyond static evaluations such as NiEH, ∞-THOR also provides an interactive evaluation, allowing agents to execute policies and complete long-horizon tasks within a dynamic environment. To support this, we release a trajectory dataset for training, with episodes over 400 steps in the training set and more than 600 steps in the dev and test sets. These trajectories can be used for imitation learning, and our experiments show that access to longer context during training leads to significant performance gains, highlighting the importance of our dataset for long-context embodied reasoning.

We further investigate various architectural considerations for embodied agents to operate under extreme sequence lengths. We show that interleaved Goal-State-Action modeling—a multimodal, goal-conditioned VLA architecture that jointly models interleaved sequences of goals, states, and actions using a LLM backbone is the most practical approach for this class of problems. Moreover, since standard LLMs are constrained by fixed con-

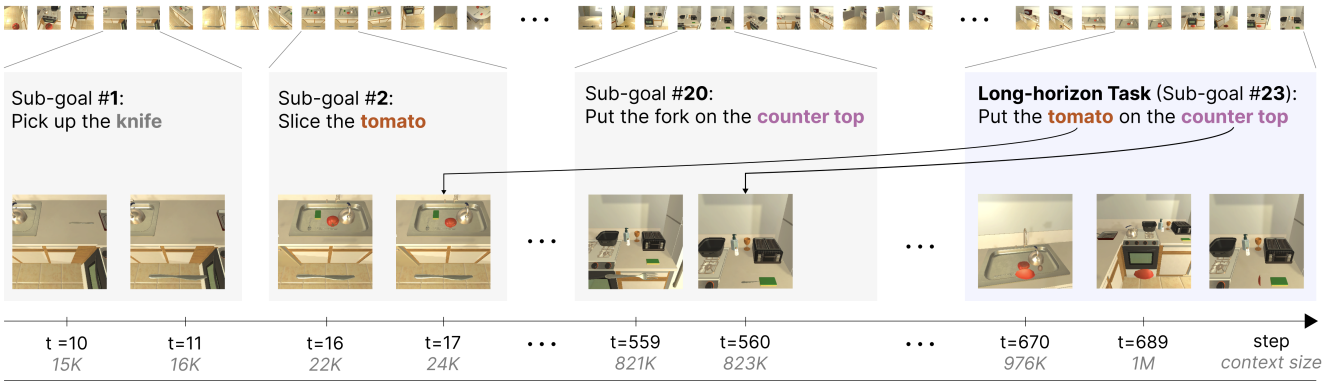


Figure 1. Example of the trajectory and a long-horizon embodied task generated from ∞-THOR. The final goal ("Put the tomato on the counter top" at t=670) requires recalling both the tomato (seen at t=17) and the counter (seen at t=560) to solved the long-horizon task. Context size refers to the input token length when converting the trajectory into the LLM input space.

text windows and cannot natively handle inputs exceeding 1M tokens, we explore long-context extension techniques such as rotary embedding scaling and positional interpolation [6, 10, 22]. Lastly, we demonstrate how to further strengthen long-context reasoning by fine-tuning the model on extended-context inputs using Context Parallelism, a parallel training strategy that allows efficient scaling to very long sequences.

We provide comprehensive experiments and analyses, demonstrating both the challenges posed by our benchmark and the behavior of baseline models under long-horizon settings. We investigate a range of training considerations, including different configurations for fine-tuning and long-context adaptation, and evaluate their impact on model performance.

Our contributions are summarized as follows:

- We introduce ∞-THOR, a new framework for generating, training, and evaluating long-horizon embodied tasks, featuring synthetic final goals that require multi-step reasoning across hundreds of steps.
- We propose a novel embodied QA task, Needle(s) in the Embodied Haystack, requiring agents to recall and reason over multiple scattered clues across extended trajectories.
- We release a large-scale trajectory dataset and an interactive evaluation environment to support both offline imitation learning and online policy execution in long-horizon settings.
- We describe architectural adaptations including interleaved Goal-State-Action modeling, long-context extension and Context Parallelism, tailored for interactive embodied reasoning.
- We present empirical results and analyses, providing insights to the current capabilities and limitations of embodied AI systems on long-horizon tasks.

2. Related Work

Long-horizon Planning in Virtual Environments. AI2THOR [15] provides interactive indoor environments widely used for embodied reasoning research, while ProcTHOR [9] extends these capabilities by procedurally generating scalable environments, potentially facilitating longer trajectories. MineDojo [11] offers an open-ended platform within Minecraft, explicitly geared toward tasks requiring extensive long-term planning. Additionally, platforms such as VirtualHome [23] and Habitat 3.0 [24] have demonstrated suitability for tasks involving long-term interactions and complex activity sequences. However, all of these platforms only provide environments and do not include standardized datasets or benchmark suites to support training and evaluation for long-horizon embodied tasks.

Embodied QA and Multimodal Needle in the Haystack Tasks. Embodied QA tasks, such as EmbodiedQA [8] and MM-EGO [31], require agents to answer questions grounded in visual observations, often demanding spatial and temporal reasoning over the agent's trajectory. While these benchmarks emphasize multimodal understanding, they do not involve active interaction with the environment during evaluation. Another related area to our NiEH task is the multimodal Needle in a Haystack (NiH) problem. Traditional NiH tasks primarily assessed textual recall within long-context inputs [19], while recent multimodal variants extend this idea by incorporating visual components [29, 30]. However, these works operate on relatively short context windows (typically up to 72K tokens) and do not require embodied reasoning or temporal dependencies across a dynamic trajectory.

Datasets and Benchmarks for Long-horizon Embodied Tasks. Recent efforts have pushed toward long-horizon embodied tasks, where agents must complete multi-

Benchmark / Platform	Task Horizon	Interaction w/ env	Dataset			QA set	
			modality	# steps	GT actions	single	multi
ProcTHOR [9]	✗	✓	✗	✗	✗	✗	✗
MineDojo [11]	Long	✓	✗	✗	✗	✗	✗
Habitat 3.0 [24]	Long	✓	✗	✗	✗	✗	✗
VirtualHome [23]	Short	✓	multi	11.6	✓	✗	✗
ALFRED [26]	Medium	✓	multi	50	✓	✗	✗
ALFWorld [27]	Medium	✓	text	50	✓	✗	✗
BEHAVIOR-100 [28]	Med/Long	✓	✗	✗	✗	✗	✗
BALROG [21]	Long	✗	✗	✗	✗	✗	✗
EQA [8]	✗	✗	✗	✗	✗	✓	✗
MM-EGO [31]	✗	✗	✗	✗	✗	✓	✗
∞-THOR	Long	✓	multi	600+	✓	✓	✓

Table 1. Comparison of benchmarks. We use Short ($<$ 50 steps), Medium (50–300 steps), and Long ($>$ 300 steps) to describe task horizon, reflecting the approximate number of environment steps required to complete a task in each benchmark. Single/Multi in the QA set column denotes single- and multi-evidence question type.

step goals with extended temporal dependencies. While ALFRED [26] and ALFWorld [27] introduced multi-step instruction-following tasks with action annotations and textual grounding, their task horizons are relatively short, typically under 50 steps. BEHAVIOR-100 [28] evaluates agent generalization on household activities, some of which require prolonged engagement, but mainly focus on single task. BALROG [21] is a benchmark for testing the agentic capabilities of long-context LLMs, but its scope is limited to game-based environments.

Long-context Benchmarks. Outside embodied AI, general benchmarks have addressed challenges in long-context reasoning. Benchmarks, such as LongBench [4] and RULER [12], focus on retrieval or summarization tasks. GSM-∞ [32] extends GSM-8K [7] to assess mathematical reasoning over extremely long textual inputs.

3. ∞-THOR: An Environment for Generating, Training, and Evaluating Long-horizon Embodied Tasks

∞-THOR features with a generation framework for synthesizing long trajectories to train and evaluate AI agents in long-horizon embodied tasks. We build ∞-THOR upon the AI2-THOR simulator [15], an interactive 3D environment for embodied AI research that supports diverse scenes, objects, and agent actions. ∞-THOR enables the creation of trajectories with no length limit, and provides an evaluation setup where agents can interact dynamically with the environment during both training and testing. This supports both offline learning by producing large-scale datasets, and online learning through direct agent-environment interaction.

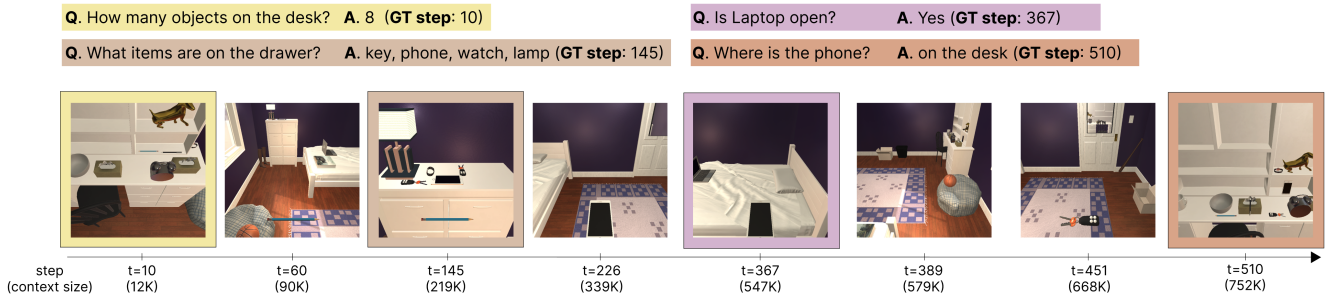
Each trajectory generated by ∞-THOR consists of multiple task goals, such as "Put a clean sponge on a metal rack" and "Pick up the apple and place it on the microwave", requiring grounded understanding and action to achieve the goal. This enables the agent to explore and interact with the environment over extended episodes. At the end of each trajectory, the agent is assigned a synthetic long-horizon task that requires reasoning over entities encountered at distant time steps. For the example in Figure 1, the long-horizon task (Sub-goal #23) at step t=689, "Put the tomato on the counter top", depends on observations made far earlier: the tomato at t=17 and the counter top at t=560. Our generation framework can generate unlimited tasks, the trajectories can be exceptionally long, exceeding 1M context tokens or beyond when the trajectory is processed with LLMs.

Successfully completing this task requires the agent to (1) memorize and integrate key environmental information over hundreds of steps, and (2) plan actions based on dependencies that are separated in time, demonstrating the need for long-context reasoning and robust spatio-temporal memory.

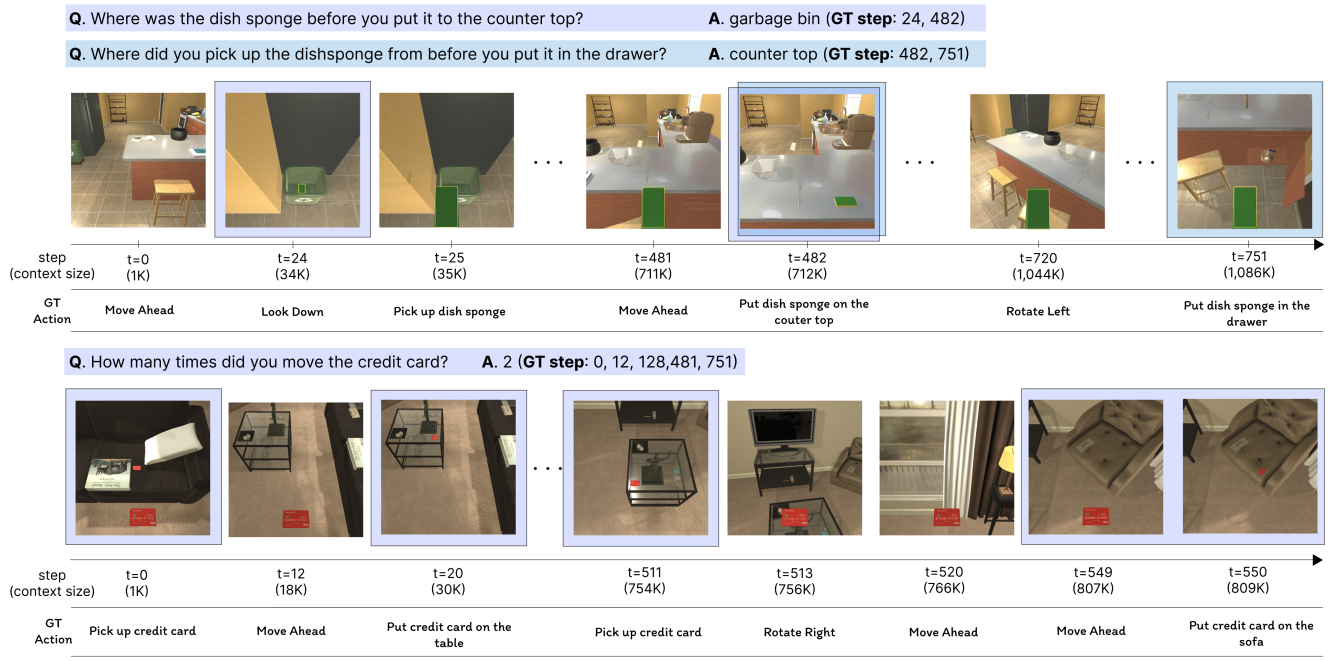
3.1. Static Evaluation: Needle(s) in the Embodied Haystack

We first introduce a novel task in the form of a static evaluation: Needle(s) in the Embodied Haystack (NiEH). NiEH is designed to evaluate an agent's ability to recall and reason about environmental states encountered throughout a trajectory. Unlike traditional embodied QA tasks that focus primarily on visual understanding of a single image, NiEH emphasizes reasoning about environmental changes over time, requiring agents to interpret and integrate sequences of multimodal observations.

Figure 2 presents examples of the two NiEH task types. In the single-evidence setting, a question is answerable based on a single observation step; in the multi-evidence setting, multiple temporally distant steps must be combined



(a) Needle in the Embodied Haystack: Single-evidence question types.



(b) Needles in the Embodied Haystack: Multi-evidence question types.

Figure 2. Example of N(s)iEH task and Ground-truth steps.

to answer the question. The NiEH testset includes diverse question types, such as binary ("yes" or "no"), "what"-, "where"-, and "how many"-style questions. These questions span a broad range of difficulty, from simple memory recall (similar to the Needle in a Haystack paradigm) to complex queries that requiring multi-step reasoning across temporally and spatially distributed evidence.

Testset Construction. We first replay the generated trajectories and collect the agent's egocentric views, along with all objects that interact with the agent throughout the trajectories, such as objects that are picked up, moved, or even simply observed. Based on these interactions, we apply a set of rule-based templates to generate QA pairs, such as "Q. What object did you slice? A. {`object name`}" and "Q. Is {`obeject name`} on the desk? A. Yes/No". Then, we sample questions based on the frequency to ensure diversity across object types, and annotate the GT answer steps using the replay logs.

After generating QA pairs and annotating the GT steps, we cross-validate the answerability of each question with GT images using four different multimodal LLMs: LLaVA-OneVision 7B [16], Qwen2.5-VL 7B [3], Deepseek-VL 7B [20], and Pixtral 12B [1]. Since these models are highly capable at standard visual QA, we filter out the questions that none of the four models successfully answer with GT images. At test time, the entire trajectory is treated as a Haystack and then cropped based on the GT image's depth. Full details on templates, generation rules, and the validation scores of the four models are included in the Appendix.

Challenges in Needle(s) in the Embodied Haystack. The NiEH task introduces two key challenges for current models. First, many questions require reasoning over multiple temporally distant events. As shown in Figure 2(b), the agent moves a `dish sponge` from the garbage bin at $t = 24$, then to the counter top, and later places it into a drawer at $t = 751$. A question such as *"Where was the dish sponge before you put it on the counter top?"* requires the model to recall and chain together multiple actions and

NiEH testset		Single-clue	Multi-clue
# of question-answer pair		829	474
Trajectory	**Train**	**Dev**	**Test**
# trajectory	2,456	125	225
# avg/max sub-goals	12/18	16/24	18/33
# avg steps	405	613	627
# max steps	654	890	952
# avg token length	602K	880K	912K
# max token length	954K	1.2M	1.3M

Table 2. Dataset Statistics

locations across hundreds of steps. Second, some questions demand aggregating sparse and temporally scattered evidence from long trajectories. In the second example in Figure 2(b), answering *"How many times did you move the credit card?"* requires the model to track and count all relevant actions occurring from the beginning to the end of the episode. These challenges highlight the need for models that can perform robust long-horizon reasoning across both time and modalities in complex embodied environments.

3.2. Constructing Long-horizon Trajectories for Interactive Evaluations

With ∞-THOR's generation framework, we can synthesize long-horizon trajectories to construct training, validation, and test sets for offline learning and evaluation. Our approach builds upon a planner-based method [15], in which we sequentially concatenate multiple single-task demonstrations into a extended trajectory, while maintaining consistency in object states and agent interactions throughout.

To generate each trajectory, we first sample a task type from one of seven predefined templates (e.g., pick two objects and place, pick and place with movable receptacle). We then sample objects that are required to perform the task, such as items to be picked up or receptacles to be interacted with. Based on the sampled task and objects, we use a classical task planner that operates on PDDL-defined domains to generate ground-truth action sequences. These plans are executed in a simulator, and only successful rollouts are retained. We then concatenate these successful demonstrations to construct long-horizon sequences that span hundreds of steps. For the final goal, the involved objects are sampled exclusively from those seen during the early 20% and the final 20% of the trajectory. This enforces a long-term temporal dependency between two objects that must be jointly referenced to complete the final task. Through this procedure, we generate 2,456/125/225 trajectories for the training, validation, and test sets, respectively. Details on task types and a pseudo-algorithm for the generation process are provided in the Appendix.

4. Architectures for Long-Horizon Vision-Language-Action Models

Embodied agents must effectively interact with complex, dynamic environments, necessitating capabilities to interpret multimodal inputs (vision, language) and produce coherent sequences of actions. Developing such Vision-Language-Action (VLA) models is particularly challenging due to the need for seamless integration of perceptual understanding, linguistic reasoning, and action prediction. Existing VLA models either use separate encoders for vision, language, and action modules [26], or focus on short-horizon, low-level tasks such as robot arm manipulation in constrained environments [5, 14], where decisions depend only on the most recent observation and a single instruction. While recent multimodal LLMs like LLaVA [17], MiniGPT-4 [33], and Llama 3.2 [2] show strong multimodal reasoning abilities, they primarily operate on static inputs (e.g., single or few images) and lack the dynamic interactivity and memory needed for long-horizon embodied tasks involving continuous vision-language-action sequences. Moreover, many state-of-the-art models are only accessible via proprietary APIs, making them impractical for real-time, controllable embodied settings and managing long-term memory states.

We explore the potential of a multimodal LLM as a unified model for VLA modeling, utilizing an interleaved input structure of goal, state (visual observations), and action tokens, as illustrated in Figure 3. This interleaved multimodal input allows the model to process vision, language, and actions concurrently, thereby facilitating more coherent, real-time interaction modeling. Specifically, our goal-conditioned agent uses a multimodal LLM backbone trained to predict subsequent actions autoregressively, conditioned on sequences of goal and state tokens. At each timestep t, the environment provides a new visual observation s_t, which is encoded as state tokens and appended to the existing token stream. The model then autoregressively predicts the next action a_t conditioned on the full history of goals, states, and previously taken actions. This action is executed in the environment (e.g., "Pick up the book"), which leads to the next observation s_{t+1}, continuing the perception-action loop. This interactive sequence is repeated over hundreds of steps, allowing the model to reason over temporally distant information while maintaining grounded behavior in dynamic settings. By leveraging this interleaved Goal–State–Action modeling, our architecture supports coherent decision-making across long-horizon embodied tasks.

Context Extension. Given the limitations in context length of most LLMs, using off-the-shelf models is insufficient for processing long inputs such as those exceeding 1M tokens. We explore various long-context extension

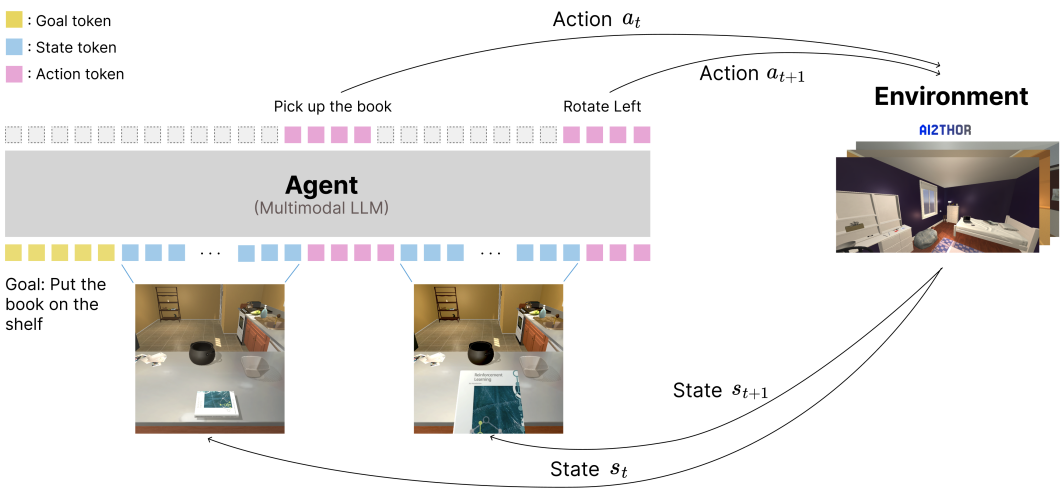


Figure 3. Agent–environment interaction through interleaved Goal-State-Action modeling.

techniques that allows the model to generalize to longer input sequences without retraining from scratch. Specifically, we consider: **Linear Interpolation**[6]: Rescales input positions to fit within the pretrained RoPE range by linearly interpolating positional indices; **Dynamic Scaling**[6]: Adapts RoPE frequencies at runtime based on the input sequence length, using a linear rescaling to maintain consistent positional encoding behavior across varying lengths; **YaRN**[22]: Dynamically interpolates attention frequencies during inference, balancing between pretrained and extrapolated positional regimes; **LongRoPE**[10]: Augments RoPE with specially designed extrapolation functions, enabling robust generalization to long sequences without degrading attention quality. We apply these techniques during fine-tuning, at inference time, or in both settings simultaneously.

Context Parallelism. To further enhance the model's ability to reason over long contexts, it is crucial to fine-tune on extended context inputs. However, the quadratic complexity of the attention mechanism makes it computationally infeasible to train directly over long sequences. To address this challenge, we employ Context Parallelism, a parallel training technique designed for efficient long-context modeling.

Context Parallelism leverages Ring Attention [18], a novel parallel implementation of the attention layer. In Ring Attention, key-value (KV) shards are cyclically shuffled across devices, and partial attention scores are computed iteratively. This process is repeated until all KV shards have been incorporated on each device, ensuring complete attention coverage without the full memory cost of standard attention. By combining Context Parallelism with our dataset of extended long-context inputs, we are able to scale fine-tuning to substantially longer sequences, unlocking improved long-horizon reasoning capabilities.

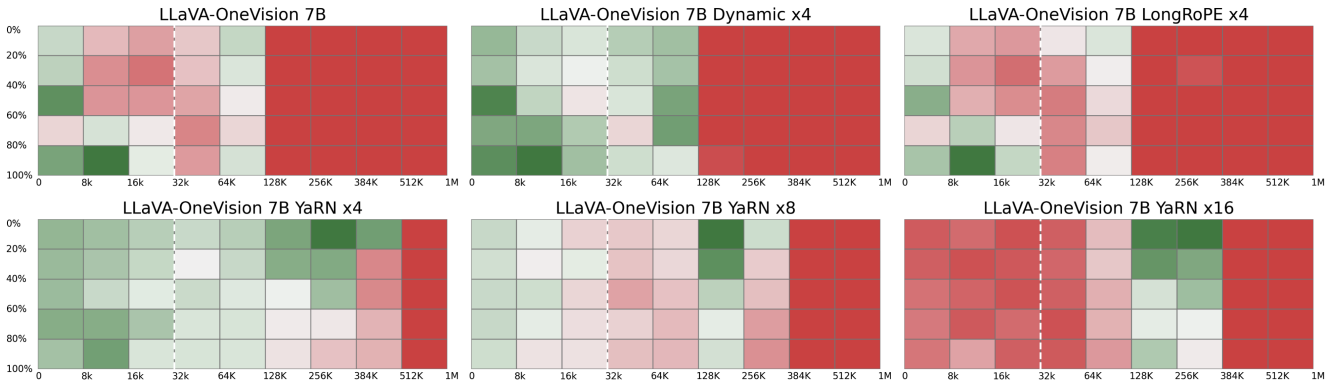
5. Experiments

5.1. Static Evaluation: Needle(s) in the Embodied Haystack

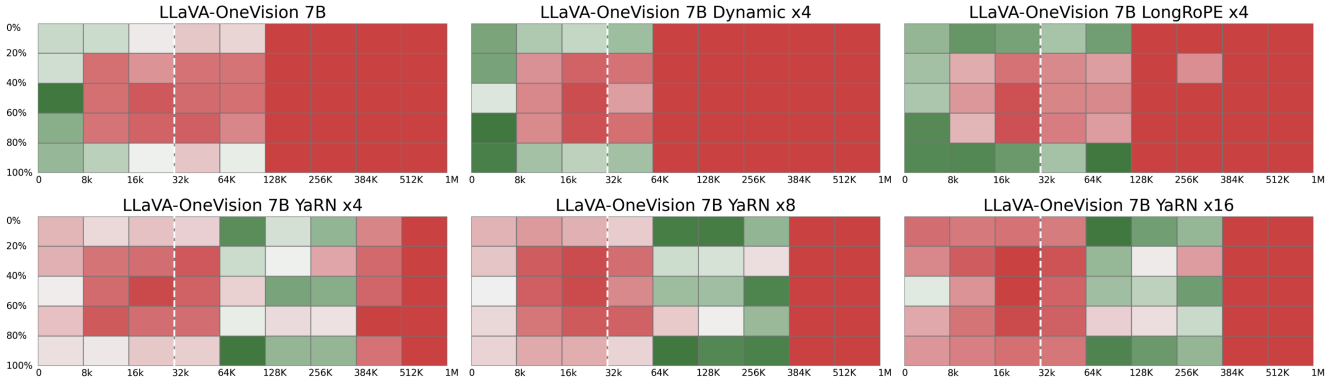
We first evaluate model performance on the Needle in the Embodied Haystack (NiEH) and Needles in the Embodied Haystack (NsiEH) tasks, which test an agent's ability to retrieve and reason over sparse evidence scattered throughout long embodied trajectories.

Building a Embodied Haystack. Unlike the traditional Needle in the Haystack setup, which inserts a target sentence into a long text corpus like a book, we use the entire embodied trajectory as the input context. To simulate different reasoning depths, we crop the input sequence either from the beginning or the end based on the GT image's position. In the NsiEH task, where multiple evidences are scattered throughout the trajectory, we fill the context with intermediate steps in between the GT steps keeping their temporal order.

Results. Figure 4 presents the performance of LLaVA-OneVision 7B [16] model across various context extension methods. Linear Interpolation, Dynamic Scaling, and LongRoPE scaling all struggled with very long contexts beyond 128K tokens (the results of Linear Interpolation are excluded from Figure since it fails at all examples). YaRN consistently outperformed other methods across both NiEH and NsiEH, successfully answering questions at context lengths exceeding 384K tokens, likely due to its architectural alignment with LLaVA-OneVision's Qwen2 LM backbone, which employs RoPE and YaRN scaling during pretraining. YaRN performed best at moderate scaling factors (e.g., x4), however, further scaling to x8 and x16 did not yield additional gains. In particular, x16 slightly improved performance in the 256K–384K token range but led to degradation notably at shorter context sizes (¡64K), sug-



(a) Results of Needle in the Embodied Haystack (NiEH).



(b) Results of Needles in the Embodied Haystack (NsiEH).

Figure 4. Results of Needle(s) in the Embodied Haystack with the LLaVA-OneVision 7B model. The white dashed line denotes the maximum input context length of the model. Context Parallelism is applied to all experiments with the context size over 384K.

gesting that excessive scaling may introduce instability and negatively impact performance. Overall, all methods fail beyond 512K tokens. We expect these trends to persist as models are pushed to even larger context windows, highlighting the need for more advanced methods to effectively handle extremely long-context scenarios in the future.

Single vs Multi-evidence Reasoning. Comparing NiEH to the more challenging NsiEH task, we observe a significant performance drop in the multi-evidence setting. This is especially pronounced for mid-depth questions involving sparse or distant evidence (e.g., *"Where was the Mug before you put it on the CounterTop?"*) or questions requiring the aggregation of multiple clues (e.g., *"How many times did you move the Apple?"*), as shown in Figure 2(b). These results demonstrate that our NiEH and NsiEH tasks pose a substantial challenge to current long-context models and success requires both fine-grained temporal memory and multi-evidence reasoning across extended embodied interactions.

5.2. Interactive Evaluation in ∞-THOR

We conduct an interactive online evaluation using the AI2THOR simulator to measure agent performance on our long-horizon test set. Our experiments analyze reward accumulation across various context extension methods, different scaling factors, and multiple fine-tuning context lengths: 32K, 64K, and 130K.

Training. We fine-tune the LLaVA-OneVision 7B model on our training set while freezing the vision encoder. We used 8 H100 GPUs with both tensor parallelism [25] and pipeline parallelism [13] for the 32K context size, while Context Parallelism is employed for training on larger context sizes (64K and 130K). Additional training specifics are available in the Appendix.

Plan-level Evaluation. We evaluate agent performance based on reward accumulation given previous states and actions. Evaluation is performed at the plan-level, where a plan represents a short sequence of actions aimed at a specific intermediate goal. For example, a "Go to location" plan comprises actions like `Move Ahead`, `Rotate Right/Left`, and `Look Up/Down`. Each trajectory is thus composed of multiple sequential plans. At each step, agents are presented with a task goal alongside the previous state-action sequences, predicting subsequent actions and interacting continuously with the environment until the plan completion. Further evaluation details are provided in the Appendix.

Results and Discussion. Figure 5 illustrates the com-

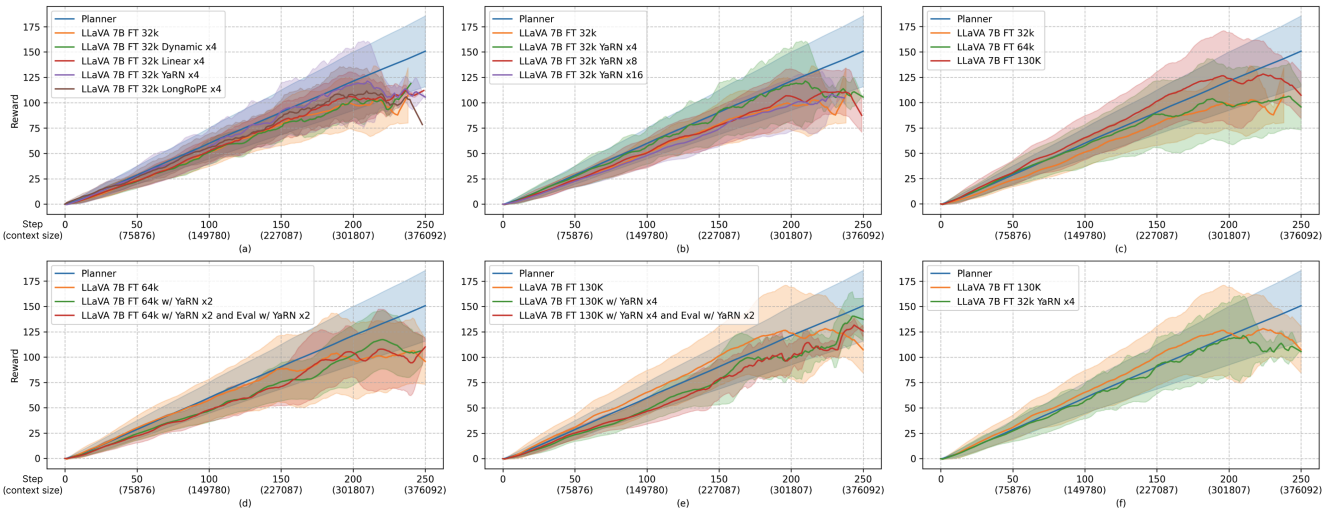


Figure 5. Agent's reward across different experimental configurations. We compare (a) context extension methods at fixed scaling (x4), (b) varying YaRN scaling factors, (c) fine-tuning with different context lengths using Context Parallelism, and (d-e) combinations of scaling during both training and inference. (f) summarizes the most effective strategies, highlighting that exposure to longer contexts during training significantly improves performance. Non-planner models cannot generate valid actions after around 250 steps (≈376K).

parative results across six experimental configurations. The Planner trajectory serves as the performance upper-bound. We address the following key questions through our experimental results:

Q. Which context extension methods perform best? Figure 5(a) compares different context extension methods at a fixed scaling factor of x4. Similar to the NiEH results, YaRN consistently achieves the highest performance showing very close performance to Planner.

Q. Does further scaling enhance performance? Figure 5(b) explores YaRN scaling at different scaling factors (x4, x8, and x16). Interestingly, increasing the scaling factor beyond x4 does not significantly improve performance, indicating a diminishing return for larger scaling factors.

Q. Is fine-tuning on a dataset with long trajectories effective? Figure 5(c) demonstrates the effectiveness of fine-tuning with Context Parallelism, enabling scaling of context lengths up to 64K and 130K tokens. At a 130K context size, the model can learn sequences comprising approximately 86 steps, substantially longer compared to only 22 steps with a 32K context size. This shows that exposure to longer context during training significantly enhances model performance, suggesting that incorporating more long-horizon data by ∞-THOR could further improve model capabilities. We note that context extension methods were not applied in this experiment.

Q. Does combining context extension methods during both training and inference provide additional benefits? Figures 5(d) and 5(e) illustrate experiments with scaling applied at both training and evaluation stages. Results indicate that additional scaling at evaluation after fine-tuning with scaled RoPE provides no further performance improvement and may degrade performance at shorter context lengths ($\leq$300K tokens).

Based on these observations, we can conclude that fine-tuning strategies are most effective when long-trajectory datasets are available. In the absence of extensive training data, employing YaRN scaling at x4 yields performance comparable to the Planner upper-bound, particularly within context lengths under 200K tokens (Figure 5(f)).

6. Conclusion

We presented ∞-THOR, a new framework for long-horizon embodied tasks designed to advance long-context understanding in embodied AI. Our framework enables scalable synthesis of long, complex trajectories paired with GT action sequences, and supports both offline training and online interaction with the environment. As part of this framework, we introduced a novel embodied QA benchmark, Needle(s) in the Embodied Haystack, that challenges agents to reason over sparse, temporally distant visual evidence embedded within extended trajectories. To equip models for this setting, we explored architectural adaptations including interleaved Goal–State–Action modeling, context extension techniques such as YaRN and LongRoPE, and efficient fine-tuning via Context Parallelism. Our experiments demonstrate that exposure to longer contexts during training significantly improves model performance, and the limitation of existing context extension techniques struggle with long-context reasoning. We hope our framework and benchmark encourage further research into models capable of robust long-horizon reasoning under realistic, interactive environments.

References

[1] Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Baptiste Bout, Devendra Chaplot, Jessica Chudnovsky, Diogo Costa, Baudouin De Monicault, Saurabh Garg, Theophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, Albert Q. Jiang, Kartik Khandelwal, Timothée Lacroix, Guillaume Lample, Diego Las Casas, Thibaut Lavril, Teven Le Scao, Andy Lo, William Marshall, Louis Martin, Arthur Mensch, Pavankumar Muddireddy, Valera Nemychnikova, Marie Pellat, Patrick Von Platen, Nikhil Raghuraman, Baptiste Rozière, Alexandre Sablayrolles, Lucile Saulnier, Romain Sauvestre, Wendy Shang, Roman Soletskyi, Lawrence Stewart, Pierre Stock, Joachim Studnia, Sandeep Subramanian, Sagar Vaze, Thomas Wang, and Sophia Yang. Pixtral 12b, 2024. 4, 1

[2] Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024. Accessed: 2025-05-11. 5

[3] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. 4, 1

[4] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand, 2024. Association for Computational Linguistics. 3

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. 5

[6] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation, 2023. 2, 6

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. 3

[8] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3

[9] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Jordi Salvador, Kiana Ehsani, Winson Han, Eric Kolve, Ali Farhadi, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation, 2022. 2, 3

[10] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens, 2024. 2, 6

[11] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge, 2022. 2, 3

[12] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. 3

[13] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Mia Xu Chen, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism, 2019. 7

[14] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 5

[15] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 2, 3, 5

[16] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024. 4, 6, 1

[17] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, pages 34892–34916. Curran Associates, Inc., 2023. 5

[18] Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023. 6

[19] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12: 157–173, 2024. 1, 2

[20] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, Jingxiang Sun, Tongzheng Ren, Zhuoshu Li, Hao Yang, Yaofeng Sun, Chengqi Deng, Hanwei Xu, Zhenda Xie, and Chong Ruan. Deepseek-vl: Towards real-world vision-language understanding, 2024. 4, 1

[21] Davide Paglieri, Bartłomiej Cupiał, Samuel Coward, Ulyana Piterbarg, Maciej Wolczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, Jakob Nicolaus Foerster, Jack Parker-Holder, and Tim Rocktäschel. Balrog: Benchmarking agentic llm/vlm reasoning on games. In *preprint*, 2024. 3

[22] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 6

[23] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs, 2018. 2, 3

[24] Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, Vladimír Vondruš, Theophile Gervet, Vincent-Pierre Berges, John M. Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Jitendra Malik, Devendra Singh Chaplot, Unnat Jain, Dhruv Batra, Akshara Rai, and Roozbeh Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023. 2, 3

[25] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020. 7

[26] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Xinyi Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10740–10749, 2020. 3, 5, 2

[27] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 3

[28] Sanjana Srivastava, Chengshu Li, Michael Lingelbach, Roberto Martín-Martín, Fei Xia, Kent Vainio, Zheng Lian, Cem Gokmen, Shyamal Buch, C. Karen Liu, Silvio Savarese, Hyowon Gweon, Jiajun Wu, and Li Fei-Fei. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments, 2021. 3

[29] Hengyi Wang, Haizhou Shi, Shiwei Tan, Weiyi Qin, Wenyuan Wang, Tunyu Zhang, Akshay Nambi, Tanuja Ganu, and Hao Wang. Multimodal needle in a haystack: Benchmarking long-context capability of multimodal large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3221–3241, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. 2

[30] Weiyun Wang, Shuibo Zhang, Yiming Ren, Yuchen Duan, Tiantong Li, Shuo Liu, Mengkang Hu, Zhe Chen, Kaipeng Zhang, Lewei Lu, Xizhou Zhu, Ping Luo, Yu Qiao, Jifeng Dai, Wenqi Shao, and Wenhai Wang. Needle in a multimodal haystack, 2024. 2

[31] Hanrong Ye, Haotian Zhang, Erik Daxberger, Lin Chen, Zongyu Lin, Yanghao Li, Bowen Zhang, Haoxuan You, Dan Xu, Zhe Gan, Jiasen Lu, and Yinfei Yang. MMEgo: Towards building egocentric multimodal LLMs for video QA. In *The Thirteenth International Conference on Learning Representations*, 2025. 2, 3

[32] Yang Zhou, Hongyi Liu, Zhuoming Chen, Yuandong Tian, and Beidi Chen. Gsm-infinite: How do your llms behave over infinitely increasing context length and reasoning complexity?, 2025. 3

[33] Deyao Zhu, Kan Chen He, Junnan Zhao, Wayne Wu, and Xinchao Chen. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023. 5

Beyond Needle(s) in the Embodied Haystack: Environment, Architecture, and Training Considerations for Long Context Reasoning

Supplementary Material

A. Dataset Construction

A.1. Building the Needle(s) in the Embodied Haystack Benchmark

We construct the Needle(s) in the Embodied Haystack benchmark in three stages: 1) Trajectory Replay and Metadata Collection; 2) Rule-Based QA Generation; and 3) Cross-validation with Multimodal LLMs. The following sections provide detailed descriptions of each step.

A.1.1. Trajectory Replay and Metadata Collection

We first replay 225 test trajectories generated by ∞-THOR, logging both visual observations (agent's egocentric views) and structured metadata at each timestep. For every step, we store the list of visible objects, agent-inventory items, openable containers, and their contents from the simulator. This produces a fine-grained interaction log that captures grounded scene dynamics over time.

An example of the collected metadata at a single timestep is shown below:

Example of metadata entry

```
{
    "img_idx": 2,
    "img_filename": "000000002.png",
    "step": 1,
    "object_log": {
        "visible": ["Shelf", "Vase", "Book"],
        "pickupable": ["Vase", "Book"],
        "isOpen": [],
        "inven_obj": [],
        "receptacles": ["Shelf"],
        "recep_objs": {
            "Shelf": ["Vase", "Book"]
        }
    }
}
```

Each metadata entry corresponds to a low-level action step and provides the semantic state of the scene, enabling the construction of temporally grounded QA instances in later stages.

A.1.2. Rule-Based QA Generation

To construct the QA set, we apply rule-based generation templates to each trajectory using its sequence of low-level actions and associated metadata. The QA generation process involves parsing the agent's interactions with objects, containers, and the environment, and applying a set of handcrafted rules to synthesize grounded questions.

Our QA generation logic covers a diverse range of question types, including object presence, object state, location tracking, slicing actions, container content reasoning, and action counting. For instance, if an object is seen for the first time at a particular step, a presence question such as "Is there any `apple` in this room?" is generated. Similarly, after a `PutObject` action, location-based questions like "Where was the `apple` before you put it to the `microwave`?" are produced. When slicing actions happen, we create questions about the object being sliced and other nearby items (e.g., "What objects were in the `Fridge` when you sliced the `apple`?"). Then, we sample questions based on the frequency to ensure diversity across object types, and annotate the GT answer steps using the replay logs. Table 3 summarizes the types of questions generated, and corresponding trigger conditions and example templates.

A.1.3. Cross-validation with Multimodal LLMs

To ensure the answerability and clarity of the generated QA pairs, we perform cross-validation using four powerful multimodal LLMs: LLaVA-OneVision 7B [16], Qwen2.5-VL 7B [3], Deepseek-VL 7B [20], and Pixtral 12B [1]. Each model is prompted with the GT images corresponding to the annotated QA steps and asked to answer the associated questions. Given their strong performance on standard visual QA tasks, we use these models to assess whether a question can be correctly answered or not. We keep only the QA pairs that are correctly answered by at least one of the four models, and discard those that fail across all models. This helps improve dataset quality and filtering out ambiguous or visually ungroundable questions. Table 4 shows the accuracy of each model on the finalized QA set when evaluated with GT images. Notably, even with access to GT images, all models struggle with questions requiring reasoning over three or more evidence steps. To maintain the benchmark's difficulty and support evaluation of more capable models in future, we manually inspect the multi-clue questions and include those that are answerable.

A.2. Constructing Long-Horizon Trajectories

To synthesize long-horizon trajectories, we construct each trajectory by sequentially chaining successful sub-tasks sampled from a predefined set of task templates. This process is illustrated in Algorithm 1. We begin by sampling a task template from a fixed task pool, which includes goal types such as `pick and place simple`, `pick two obj and place`, and `pick and place with`

Table 3. QA types, trigger conditions, and corresponding question templates used in rule-based generation.

QA Type	Trigger Condition	Example Template(s)
object presence (Yes/No)	object appears visibly in the trajectory	`Is there any {obj} in this room?` `Have you seen a/an {obj}?`
open state questions	container marked as open in metadata	`Was {container} open?`

object location tracing	sequences of `Pickup` and `PutObject` actions	`Where was {obj} before you put it to {container}?`
		`Where did you move the {obj} from the {container}?`
		`Where is {obj} now?`

slicing-based questions	`SliceObject` action detected in trajectory	`What did you slice?` `What objects were in/on {container} when you slice {obj}?`
container content	container visibility with non-empty contents	`What objects were in/on the {container}?` `What object did you put in/on the {container}?`
put action questions	unique `PutObject` action for a container	`What object did you put in/on the {container}?`

final object state	final location of an object at episode end	`Is {obj} in/on the {container}?`
		`What objects are in/on the {container}?`
		`How many objects were in/on the {container}?`

movement counting	object picked up more than once	`How many times did you move {obj}?`

Table 4. QA accuracy (%) of multimodal LLMs on ground-truth images.

Model	Size	# of clues (GT steps)			Total
		1	2	≥3	
LLaVA-OneVision	7B	86.61	68.55	23.74	71.15
Qwen2.5-VL	7B	85.94	89.83	64.40	82.20
Deepseek-VL	7B	81.56	39.14	22.57	62.88
Pixtral	12B	91.34	39.60	58.56	76.25

`movable recep`. Each sampled template requires relevant objects in the scene (e.g., pickupable items, target receptacles), which are then used to define the goal for that task.

We use a classical task planner, which operates over PDDL-defined domains [26], to generate a low-level action sequence for the sampled goal, and simulate this plan in an interactive environment. If the rollout fails (e.g., due to collisions, object occlusions, or unreachable conditions), we discard the sequence and re-sample from the task pool. Otherwise, the successful rollout is retained and appended to the ongoing trajectory.

This sampling-execution loop is repeated until a long trajectory with a desired number of sub-goals is formed. The resulting synthetic long-horizon trajectory consists of multiple sub-goals concatenated into a continuous sequence. To induce long-term temporal dependencies, the final sub-task is constrained to involve only objects that appear in the early 20% and late 20% of the overall trajectory, requiring the agent to integrate temporally distant evidence to answer associated questions.

Algorithm 1 Construct Long-horizon Trajectory

Input: Task Pool $\mathcal{T}$, max sub goals N
Output: Long-horizon trajectory τ
Initialize empty trajectory $\tau \leftarrow []$
while len(τ) $< N$ **do**
  Sample task template $g \sim \mathcal{T}$ and objects
  Plan action sequence π_g by planner
  if Simulate(π_g) is successful **then**
    Append π_g to trajectory: $\tau \leftarrow \tau \parallel \pi_g$
  else
    Discard and re-sample
  end if
end while
// Final sub-task with long-term object dependency
Sample $g_{\text{final}} \sim \mathcal{T}$ and objects in early 20% and late 20%
Plan and simulate π_{final} using restricted objects
if Simulate(π_{final}) is successful **then**
  Append π_{final} to trajectory: $\tau \leftarrow \tau \parallel \pi_{\text{final}}$
else
  Repeat sampling until success
end if
return τ

B. Training and Evaluation Details

B.1. Interactive Evaluation in ∞-THOR

Training. We fine-tune the LLaVA-OneVision 7B model on our training set while freezing the vision encoder. The model is trained using a next-action prediction objective, where only the action tokens are optimized, conditioned on the goal and state tokens. Table 5 summarizes the training specifications for different context lengths. For 32K

Table 5. Training specifications for different context lengths.

Context Length	Parallelism	# GPUs	Training Time
32K	Tensor (4) + Pipeline (2)	8	160 hrs
64K	Context (8)	8	120 hrs
130K	Context (16)	16	134 hrs

training, we apply tensor parallelism with a degree of 4 and pipeline parallelism with a degree of 2, utilizing 8 H100 GPUs in total. Since pipeline parallelism requires the batch size to match the pipeline degree, we set the batch size to 2. For longer context lengths, we use context parallelism: 8-way for 64K (on 8 GPUs) and 16-way for 130K (on 16 GPUs). All models are fine-tuned for approximately 3 epochs with a learning rate of 1e-5, using the AdamW optimizer and a linear learning rate schedule with a 0.03 warmup ratio.

Plan-Level Evaluation. We evaluate agent performance using a plan-level framework, where each plan corresponds to a short sequence of actions aimed at achieving a specific intermediate sub-goal (e.g., navigating to an object, placing an item). A trajectory is composed of multiple such plans, executed sequentially. For the interactive evaluation, the agent is presented with the current plan's goal along with the history of previous GT states and actions. Using this context, the agent predicts the next action and interacts step-by-step with the environment. The interaction continues until the current plan is either successfully completed or terminated due to failure (e.g., collisions or deadlocks). After each plan, the context is reset to include the GT actions and states from the completed portion of the trajectory, and the agent proceeds to the next plan. This ensures that each plan is evaluated independently, conditioned only on the correct prior history. The agent's performance is measured via cumulative reward across all plans in the trajectory. Pseudocode for this evaluation procedure is provided in Algorithm 2.

Algorithm 2 Plan-Level Evaluation

Input: Trajectory $T = \{P_1, P_2, \ldots, P_N\}$, Agent policy π, Environment $\mathcal{E}$
Initialize: Reward $R \leftarrow 0$
Initialize state and history with initial observation
for each plan P_i in T **do**
    Initialize context with GT actions up to P_{i-1}
    while not *done* and not *failure* **do**
        $a_t \leftarrow \pi(\text{context})$
        $s_{t+1}, r_t, \textit{done}, \textit{failure} \leftarrow \mathcal{E}.\text{step}(a_t)$
        Append (a_t, s_{t+1}) to context
        $R \leftarrow R + r_t$
    end while
    if *failure* **then**
        Break evaluation
    end if
end for
return Total accumulated reward R