ICML
International Conference
On Machine Learning

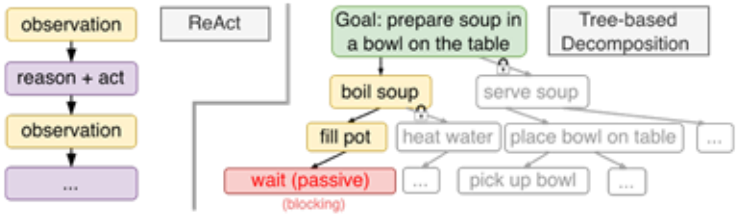
Embodied Task Planning via Graph-Informed Action Generation with Large Language Models

Xiang Li[12], Ning Yan[2], Masood Mortazavi[2] [1]Purdue University [2]Futurewei Technologies

1. Background and Motivation

The Rise of Embodied Agents

- ***Use Cases:*** robots in warehouses and homes aim to execute long-horizon tasks (i.e., cooking, cleaning, etc) .
- ***Challenges:*** sub-goals are rarely independent and environments are dynamically evolving.

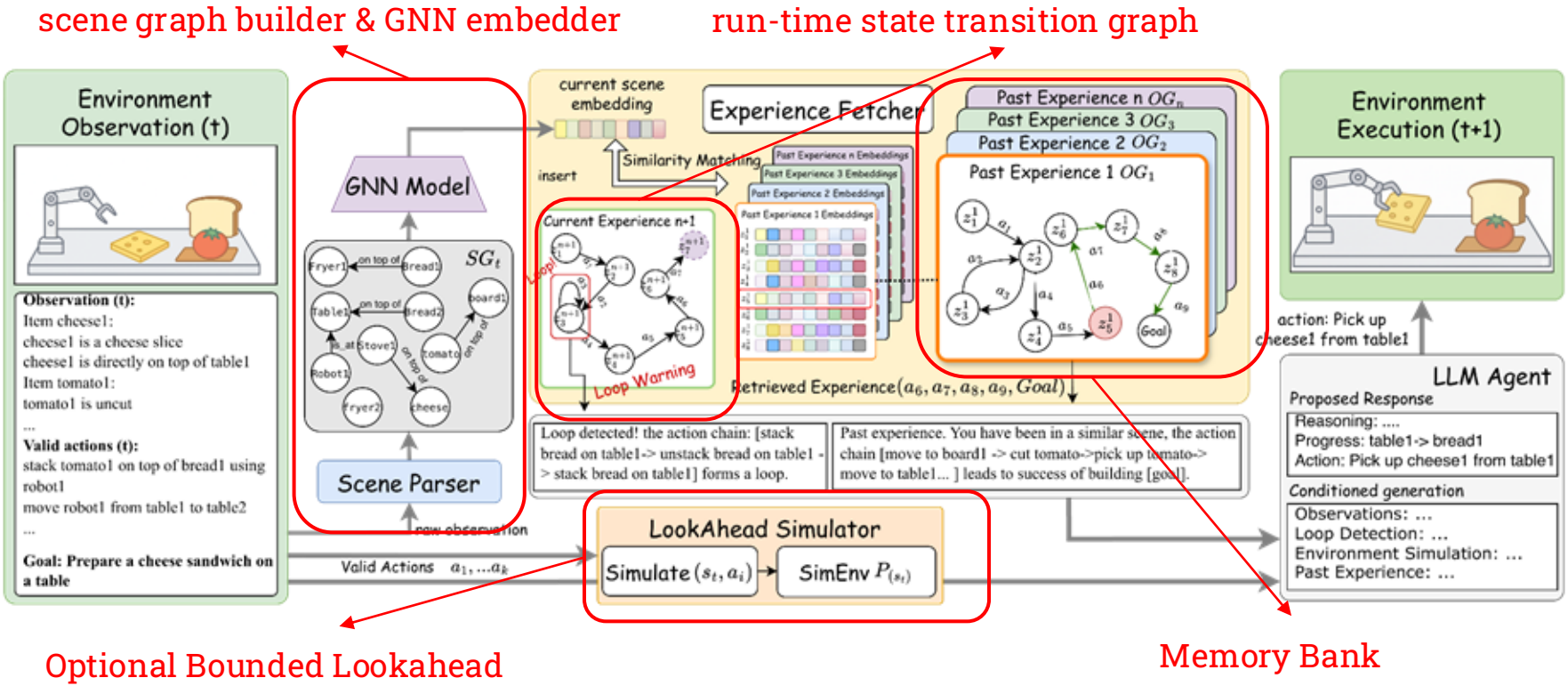


The Limitations of Current LLM Agents

- ***Context Drift:*** reasoning and Act (ReAct[1]) topology leads to a linearly growing history. LLMs lose track of high-level goals.
- ***Artificial Serialization:*** tree-based methods (e.g., ReCAP[2]) force sequential execution, blocking parallel tasks.

[1]Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan,K., and Cao, Y. ReAct: Synergizing reasoning and acting in language models. In International Conference on Learning Representations, ICLR, 2023
[2]Zhang, Z., Chen, T., Xu, W., Pentland, A., and Pei, J. Recap: Recursive context-aware reasoning and planning for large language model agents. In Advances in Neural Information Processing Systems, NeurIPS, 2025

2. Our Solution: Graph-in-Graph Planning Framework



➢ GiG aligns the agent's memory with how the physical world works - *spatially* and *temporally*.

2.1 GNN Encoder Optimization & Experience Retrieval

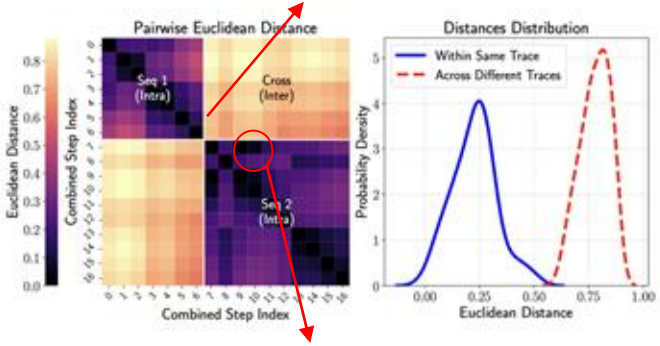
We train the GNN using a composite loss function

$$\mathcal{L} = \mathcal{L}_{\text{triplet}} + \lambda \mathcal{L}_{\text{uniformity}} = \mathbb{E}\left[\max\left(0, \|\mathbf{z}_a - \mathbf{z}_p\|_2^2 - \|\mathbf{z}_a - \mathbf{z}_n\|_2^2 + \alpha\right)\right] + \lambda \left(\frac{1}{N^2} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left(\frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\|\mathbf{z}_i\| \|\mathbf{z}_j\|} \right)^2 \right)$$

- Assumptions of environment coherence: physical states evolve gradually, temporally adjacent representations should be more proximal to each other than to randomly sampled states from disjoint trajectories.

- We match the current scene embedding to the memory bank and retrieve the action sequence that follows the closest historical state.

clear separation between two sequences



dark region indicates exploration loops

3. Evaluation across Benchmarks

3.1 Pass @ 1 Performance

Robotouille Synchronous & Asynchronous

Table 1: Pass@1 Accuracy on Robotouille: Synchronous vs. Asynchronous. Parentheses indicate the absolute percentage gain relative to the most competitive baseline.

Pass@1	Synchronous					Asynchronous				
Model	**GiG**	**GiG(+Exp)**	**ReCAP**	**ReAct**	**CoT**	**GiG**	**GiG(+Exp)**	**ReCAP**	**ReAct**	**CoT**
Qwen3[1]	93 (+19)	97 (+23)	71	74	7	72 (+37)	82 (+47)	35	31	0
DeepSeek[2]	91 (+19)	88 (+16)	72	53	2	59 (+32)	86 (+59)	27	16	0
Gemini[3]	92 (+0)	90 (-2)	89	92	34	66 (+6)	66 (+6)	21	60	4

[1]Qwen3-235B-A22B-Instruct-2507-FP8; [2]DeepSeek-R1; [3]Gemini-2.5-Flash.

ALFWorld

Table 2. Pass@1 Accuracy on ALFWorld

Model	GiG	ReCAP	ExpeL	ReAct
Qwen3[1]	**97**(+6)	89	91	61
DeepSeek[2]	**97**(+15)	82	75	N/A[4]
Gemini[3]	**91**(+2)	86	89	N/A

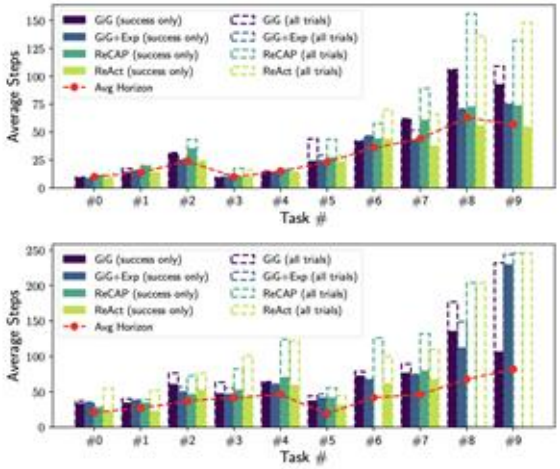


Figure 4. Average steps on Robotouille tasks on Qwen3-235B for synchronous (top) and asynchronous (bottom) settings. Red dots indicate the horizon length of each task type. *success only*: average completion steps of successful attempts. *all trials*: average steps of all attempts, including failure attempts.

[3]Gonzalez-Pumariega, G., Yean, L. S., Sunkara, N., and Choudhury, S. Robotouille: An asynchronous planning benchmark for LLM agents. In The Thirteenth International Conference on Learning Representations, ICLR, 2025
[2]Zhang, Z., Chen, T., Xu, W., Pentland, A., and Pei, J. Recap: Recursive context-aware reasoning and planning for large language model agents. In Advances in Neural Information Processing Systems, NeurIPS, 2025

3.2 Experience Memory Plug-in for Small LLMs

Table 4. Pass@1 Accuracy on Robotouille with small models

Model	GiG	GiG+Exp	ReCAP	ReAct
Qwen3[5]	27	42(+15)	19	28
Gemini[6]	19	26(+7)	20	20

- Significant degradation: Without memory, small models struggle
- The ”Plug-in” Effect: Adding experience memory recovers performance significantly.

3.3 Context Window Efficiency

Table 7. Comparison of average context window consumption/step (mean ± std) across methods.

	Avg. Context Consumption per Step (Prefill)		
	GiG	ReCAP	ReAct
Tokens	12297 (±1840)	40720 (±26655)	57357 (±46860)

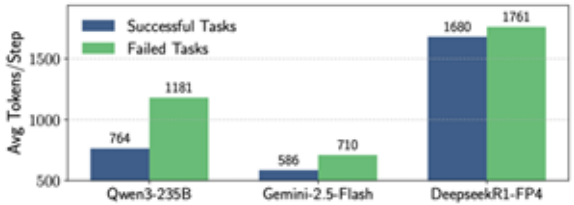
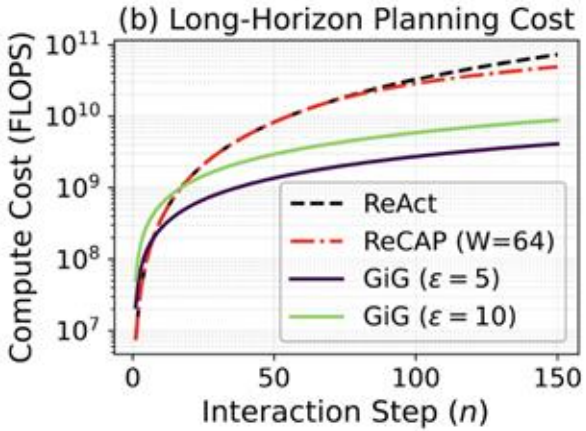


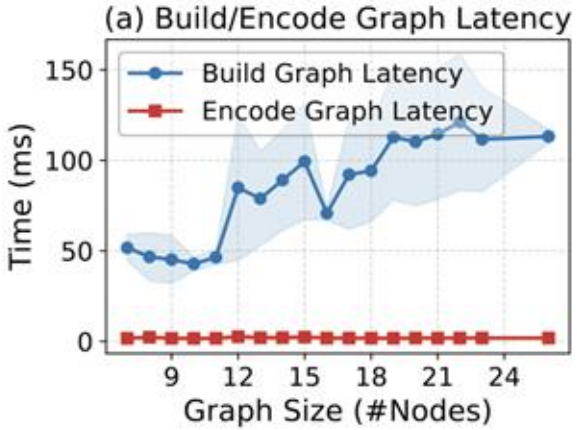
Figure 10. Average tokens (reasoning + output) generated per step. Models put more effort for failed tasks.

- Through dynamically updated scene graph, GiG reduced the average context window consumption to only 30% of ReCAP and 21% of ReAct in long horizon tasks.
- The average decoding token per step varies significantly across different backbone models, but they show a similar trend that failed tasks typically incur more token generation compared to successful tasks, reflecting the additional reasoning effort.

3.4 Cost Analysis: Efficiency & Scalability

Metric: We evaluate FLOPs (Computational Cost) rather than monetary cost to ensure standardized comparison across hardware.





Why GiG is compute efficient?

- Baselines: ReAct/ReCAP process cumulative history, causing cost to explode over time.
  GiG: Uses condensed graph only.
- Result: Orders of magnitude lower FLOPs as task length increases.

Do graphs cause large overhead?

- Graph Building: Scales linearly with node number but remains sub-second (< 150 ms).
- Comparison: LLM decoding takes seconds.
- Conclusion: The overhead of GiG is negligible in practice.

Takeaways:

- ***Structural Necessity:*** Decoupling spatial topology from temporal progression can help mitigate the context drift in long-horizon tasks.
- ***Small Model Enhancement***: Using retrieved structured experience as priors helps improve the performance of less capable LLMs.
- ***Systems Efficiency:*** Maintaining a compact, graph-based context can reduce computational overhead by orders of magnitude.