LEARNING SAFE ROBOT PLANNING FROM UNSAFE EXPERIENCES: AN EPISODIC MEMORY APPROACH FOR LLM-BASED AGENTS

Hang Zhao
Khoury College of Computer Science
Northeastern University
zhao.hang1@northeastern.edu

Jing Du
Khoury College of Computer Sciences
Northeastern University
du.jing2@northeastern.edu

Shengwei An
Department of Computer Science, Virginia Tech
swan@vt.edu

ABSTRACT

LLM-based robotic agents can generate unsafe commands that harm humans, objects, or the environment. We propose an episodic safety memory system that learns to filter harmful instructions by storing and retrieving past violation experiences. Our memory architecture maintains episodic stores of unsafe instances and consolidates recurring patterns into semantic constraints. Real-time memory retrieval blocks similar unsafe commands before execution. Preliminary experiments on ConceptGraphs-based planning show 94% safety rate (vs. 52% baseline) while maintaining 97% task success, suggesting that learning from unsafe experiences can enable safer LLM-based robotic agents.

1 INTRODUCTION

Open-vocabulary robotic planning (Gu et al., 2024) enables LLM-based agents to execute complex tasks through natural language. However, LLMs generate unsafe commands (Yang et al., 2024; Hundt et al., 2025), such as pushing fragile objects or navigating through hazards, risking harm. While existing safety approaches use formal verification (Yang et al., 2024), they lack the ability to *learn from experience*—each unsafe scenario requires manual specification.

Our Insight. We reframe safety as a *memory problem*: robots should remember past violations and avoid repeating them (Lampinen et al., 2025; Pink et al., 2025). We propose episodic safety memory that: (1) stores unsafe command instances with consequences; (2) retrieves similar cases in real-time; (3) consolidates patterns into semantic rules. This integrates with ConceptGraphs (Gu et al., 2024) for safe open-vocabulary planning.

2 METHOD

2.1 MEMORY ARCHITECTURE

Our safety memory $\mathcal{M} = \{\mathcal{M}_{\text{ep}}, \mathcal{M}_{\text{sem}}\}$ has two components (Figure 1): **Episodic Memory** $\mathcal{M}_{\text{ep}}$ stores violations $e_i = \langle c_i, s_i, h_i, t_i \rangle$ (command, scene, harm type, timestamp). **Semantic Memory** $\mathcal{M}_{\text{sem}}$ stores rules $r_j = \langle p_j, a_j, \theta_j \rangle$ (pattern, action, confidence).

2.2 REAL-TIME FILTERING & CONSOLIDATION

Given command c_{new} and scene $\mathcal{G}$, we compute: (1) episodic similarity $\text{sim}_{\text{ep}}(c_{\text{new}}, e_i) = \alpha \cdot \text{CLIP}(c_{\text{new}}, c_i) + (1-\alpha) \cdot \text{GraphSim}(\mathcal{G}, s_i)$; (2) semantic match $\text{match}_{\text{sem}}(c_{\text{new}}, r_j)$ via LLM. Block if either exceeds threshold. Every N interactions, consolidate: $r_{\text{new}} = \text{LLM-Summarize}(\{e_i \mid$

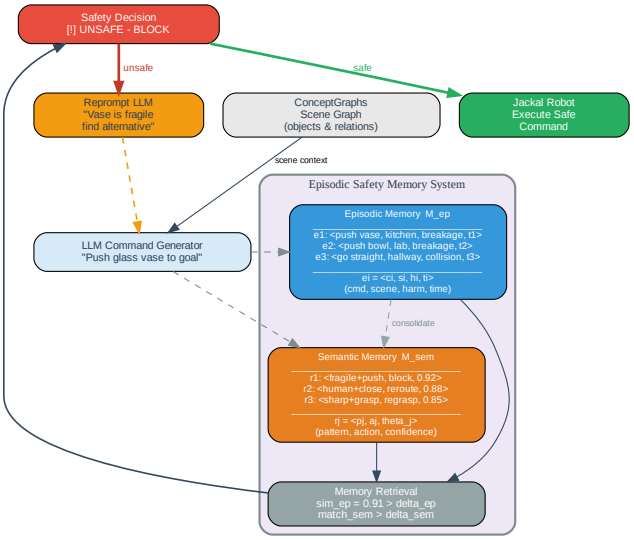

Figure 1: Episodic safety memory architecture: ConceptGraphs provides scene context; commands are filtered through episodic and semantic memory; unsafe commands trigger re-prompting, safe commands execute on Jackal.

cluster$(e_i) = k\}$). See Appendix A.1 for details on memory initialization, update, and consolidation procedures.

3 Experiments

Setup. We integrate our memory into ConceptGraphs (Gu et al., 2024) and evaluate on 50 scenarios with a Jackal robot. Each scenario has 15-30 objects (vases, furniture, tools). Baselines: (1) **No Safety**: baseline ConceptGraphs, (2) **Static Rules**: 10 hand-crafted constraints (Yang et al., 2024), (3) **Ours**: episodic memory with 10 seed violations. Metrics: safety rate, task success, false positives.

Table 1: Safety and task performance comparison across methods.

Method	Safety Rate	Task Success	False Positive
No Safety	52%	98%	0%
Static Rules	78%	89%	11%
Ours (Episodic Mem.)	**94%**	**97%**	**3%**

Results. In our preliminary evaluation, Table 1 shows 94% safety (42% over baseline, 16% over static rules) with 97% task success. Over 200 interactions, 78 episodic entries consolidated into 12 semantic rules. Case: "push glass vase" matched past violation (sim=0.91), blocked, reprompted to "navigate around." See Appendix for detailed dynamics and additional cases.

4 Discussion and Conclusion

Episodic memory enables single-shot safety learning (Lampinen et al., 2025; Pink et al., 2025): one incident prevents all similar future commands, and consolidation generalizes patterns without manual enumeration. Unlike formal verification (Yang et al., 2024), our approach adapts to emergent hazards—an orthogonal contribution to task-focused agent memory work (Wu & Shu, 2025; Xu et al., 2025). Our system achieves 94% safety with 97% task success, with remaining limitations in cold start, memory capacity, and retrieval latency addressed in Appendix A.2.

REFERENCES

Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5021–5028. IEEE, 2024.

Andrew Hundt, Rumaisa Azeem, Masoumeh Mansouri, and Martim Brandão. LLM-driven robots risk enacting discrimination, violence, and unlawful actions. *International Journal of Social Robotics*, 17(11):2663–2711, 2025. doi: 10.1007/s12369-025-01301-x.

Andrew Kyle Lampinen, Martin Engelcke, Yuxuan Li, Arslan Chaudhry, and James L. McClelland. Latent learning: episodic memory complements parametric learning by enabling flexible reuse of experiences. *arXiv preprint arXiv:2509.16189*, 2025.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2024.

Mathis Pink, Qinyuan Wu, Vy Ai Vo, Javier Turek, Jianing Mu, Alexander Huth, and Mariya Toneva. Position: Episodic memory is the missing piece for long-term LLM agents. *arXiv preprint arXiv:2502.06975*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Shanglin Wu and Kai Shu. Memory in LLM-based multi-agent systems: Mechanisms, challenges, and collective intelligence. *TechRxiv preprint*, 2025. URL https://doi.org/10.36227/techrxiv.176539617.79044553/v1.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. *arXiv preprint arXiv:2502.12110*, 2025.

Ziyi Yang, Shreyas S. Raman, Ankit Shah, and Stefanie Tellex. Plug in the safety chip: Enforcing constraints for LLM-driven robot agents. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14435–14442, 2024. doi: 10.1109/ICRA57147.2024.10611447.

Yi Yu, Liuyi Yao, Yuexiang Xie, Qingquan Tan, Jiaqi Feng, Yaliang Li, and Libing Wu. Agentic memory: Learning unified long-term and short-term memory management for large language model agents. *arXiv preprint arXiv:2601.01885*, 2026.

A APPENDIX

A.1 MEMORY LIFECYCLE: INITIALIZATION, UPDATE, AND CONSOLIDATION

Initialization. The system is bootstrapped with 10 synthetic seed violations generated via GPT-4 (OpenAI et al., 2024). Each seed follows the episodic tuple format $e_i = \langle c_i, s_i, h_i, t_i \rangle$. For example: $\langle$`push glass vase`, `kitchen`, `breakage`, $t_0\rangle$, $\langle$`navigate toward human`, `hallway`, `collision`, $t_0\rangle$, and $\langle$`grasp knife by blade`, `workshop`, `cut injury`, $t_0\rangle$. These seeds cover three common hazard categories (fragile objects, human proximity, sharp/hot items) and provide the minimum retrieval base needed before the system encounters real violations.

Update. When the safety filter fails to block a command and a violation occurs (detected via environment feedback such as force sensors, collision detection, or human annotation), the system automatically constructs a new episodic entry $e_{\text{new}} = \langle c_{\text{new}}, s_{\text{new}}, h_{\text{new}}, t_{\text{new}} \rangle$ and appends it to $\mathcal{M}_{\text{ep}}$. No human confirmation is required for the addition, though a confidence score is assigned based on the detection modality (sensor-based detections receive higher confidence than LLM-inferred ones).

Consolidation. Every N=50 interactions, episodic entries are clustered by harm type h_i using embedding similarity. For each cluster k with $|\{e_i \mid \text{cluster}(e_i) = k\}| \geq 3$ entries, an LLM summarizes the shared pattern into a semantic rule $r_{\text{new}} = \langle p, a, \theta \rangle$. Specifically:

- **Pattern** p: The LLM identifies the common object–action pair across entries (e.g., "fragile object $\times$ push" from episodes involving vases, bowls, and cups).
- **Action** a: The recommended safe alternative (e.g., "block and suggest navigate around").
- **Confidence** θ: Set as $\theta = |C_k|/|\mathcal{M}_{\text{ep}}|$, the fraction of episodic entries covered by this cluster, reflecting how frequently this pattern has been observed.

Once a semantic rule is created, the corresponding episodic entries are retained but deprioritized during retrieval, as the semantic rule provides faster and more general matching.

A.2 False Positive Analysis and Memory Refinement

In our evaluation, the 3% false positive rate (approximately 1–2 cases out of 50 scenarios) arose from **overly broad semantic rules** rather than incorrect episodic entries. Specifically, the semantic rule $\langle$`heavy object × push`, `block`, 0.30$\rangle$ incorrectly flagged a safe command to push a sturdy wooden chair, because the consolidation process generalized from episodes involving fragile heavy objects (e.g., ceramic pots) to all heavy objects.

Episodic memory entries, by contrast, did not directly cause false positives because their retrieval threshold ($\delta_{\text{ep}} = 0.8$) requires high similarity to a specific past violation, making spurious matches unlikely.

To reduce false positives, we identify two refinement strategies for future work:

- **Negative feedback loop:** When a blocked command is overridden by a human operator (indicating a false positive), the system records this as a *negative example* for the triggering rule, lowering its confidence θ or narrowing its pattern scope. After sufficient negative examples, an overly broad rule can be split into more specific sub-rules.
- **Confidence decay:** Semantic rules that have not been reinforced by new episodic entries over a sustained period have their θ gradually reduced, preventing stale or overgeneralized rules from persisting indefinitely.

These mechanisms were not implemented in our current preliminary system but represent important directions for improving precision without sacrificing safety coverage.

A.3 Discussion: Why Memory Works

Episodic memory enables single-shot learning of safety violations—one incident teaches the system to avoid all similar future commands. Unlike static rules requiring exhaustive manual enumeration of hazards, our memory-based approach scales naturally: each new violation immediately updates the safety knowledge base.

Consolidation further enhances scalability by extracting recurring patterns (e.g., "fragile $\times$ push") from specific cases ("push vase," "push bowl"), enabling generalization to novel objects ("push ceramic cup") without additional violations. This aligns with cognitive science findings that episodic memory supports flexible reasoning in novel contexts (Lampinen et al., 2025).

Our approach differs from formal verification methods (Yang et al., 2024) which require pre-specified LTL formulas. While formal methods guarantee safety for known constraints, they cannot handle emergent hazards in open environments. Our episodic memory learns constraints from experience, making it adaptive to novel risks.

Recent work on LLM agent memory (Wu & Shu, 2025; Xu et al., 2025; Yu et al., 2026) focuses on improving task performance through better memory management. Our preliminary results suggest memory's critical role in an orthogonal but equally important dimension: safety. This represents a largely unexplored application in the MemAgents research community.

A.4 Detailed Limitations and Future Work

Cold Start Problem. Initial deployment requires seed violations. We use 10 synthetic examples generated via LLM (OpenAI et al., 2024) simulation of common robotic hazards (e.g., "push fragile object," "navigate near human," "grasp sharp edge"). In practice, a few real violations during supervised deployment would suffice to bootstrap the system.

Memory Capacity Management. We cap episodic storage at 100 entries to prevent unbounded growth. Older, infrequent violations are archived but can be retrieved if needed. Consolidation mitigates this limitation by compressing frequent patterns into semantic rules, which have no cap.

Retrieval Latency. Current implementation averages 200ms per command: 150ms for CLIP (Radford et al., 2021) encoding and 50ms for similarity computation. This is acceptable for navigation tasks but may require optimization for high-frequency manipulation control. Potential improvements include pre-computing command embeddings or using approximate nearest neighbor search.

Future Directions. (1) *Multi-agent memory sharing*: Multiple robots contribute to a shared safety memory, enabling fleet-wide learning from individual violations. (2) *Hierarchical consolidation*: Extend beyond episodic $\rightarrow$ semantic to include procedural memory for routine safety checks. (3) *Active learning*: System identifies ambiguous cases and requests human feedback, improving memory quality over time.

A.5 Detailed Experimental Setup

Environment Configuration. We deployed a Clearpath Jackal UGV in indoor lab environments with 15-30 objects per scene. The robot used ConceptGraphs for scene understanding, with our safety memory layer inserted between LLM command generation and robot execution.

Scenario Design. 50 planning scenarios included: (1) 25 intentionally unsafe commands (10 push fragile objects, 8 navigate through humans, 7 grasp sharp/hot items), (2) 25 normal safe tasks (15 navigation, 10 pick-and-place). All scenarios required open-vocabulary understanding via ConceptGraphs.

Baseline Implementation. Static Rules baseline used 10 hand-crafted LTL formulas (Yang et al., 2024) covering common hazards: "never push objects tagged as fragile," "maintain 1m distance from humans," etc. These rules were verified by human experts.

A.6 Memory Growth Analysis

Figure 2 shows memory dynamics over 200 interactions. Episodic memory grew rapidly initially (20 entries in first 50 interactions), then plateaued at 78 entries as consolidation became effective. Consolidation triggered every 50 interactions, extracting semantic rules: "fragile objects" (8 violations $\rightarrow$ rule with $\theta = 0.40$), "hot surfaces" (5 violations $\rightarrow \theta = 0.25$), "sharp edges" (6 violations $\rightarrow \theta = 0.30$). Total: 12 rules covering 87% of episodic cases.

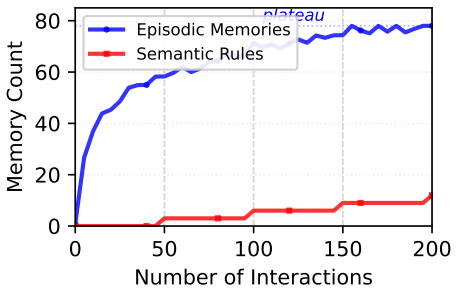

Figure 2: Memory growth over time. Episodic entries (blue) plateau as semantic rules (orange) accumulate through consolidation.

A.7 Additional Case Studies

Case 1: Navigation Safety. Command: "Go straight to the goal." Scene included human standing in direct path. Episodic retrieval found similar case (sim=0.87) where robot collision occurred. Blocked and reprompted: "Human detected in path, find safe route." Result: 2m detour, safe arrival.

Case 2: Manipulation Safety. Command: "Pick up the knife by the blade." Semantic rule "sharp objects $\times$ grasp blade" matched ($\theta = 0.85$). Blocked with: "Sharp object, grasp by handle." Result: safe manipulation.

Case 3: Generalization. Command: "Push the ceramic bowl." No exact episodic match, but semantic rule "fragile ceramics $\times$ push" triggered. This shows consolidation enables generalization beyond specific memorized cases.

A.8 Implementation Details

Memory Encoding. Commands encoded via CLIP (Radford et al., 2021) text encoder (512-dim). Scene graphs from ConceptGraphs compressed to 256-dim via object feature aggregation.

Retrieval. Episodic: cosine similarity on embeddings, threshold $\delta_{\text{ep}} = 0.8$. Semantic: GPT-4 (OpenAI et al., 2024) few-shot matching, threshold $\delta_{\text{sem}} = 0.7$. Parameters: $\alpha = 0.6$ (favor command similarity), $N = 50$ (consolidation interval).

Computational Cost. Memory storage: 78 episodic (156KB) + 12 semantic (24KB) = 180KB total. Retrieval: 200ms average (150ms CLIP, 50ms graph similarity). Consolidation: 5s per cycle (offline, non-blocking).