Self-Evolving Just-In-Time Memory for Proactive Embodied Safety

Bingrui Sima[1], Lizhong Wang[2], Xiaoya Lu[3], Kun He[1⋆], and Xiao Yang[2⋆]

[1] Huazhong University of Science and Technology, Wuhan, China
[2] Dept. of Comp. Sci. and Tech., Tsinghua University, Beijing, China
[3] Shanghai Jiao Tong University, Shanghai, China
d202481592@hust.edu.cn, yangxiao19@tsinghua.org.cn

Abstract. While Vision-Language Models (VLMs) have empowered embodied agents to execute complex household tasks, they struggle to proactively handle dynamically emerging hazards during closed-loop interactions. Existing safety approaches often rely on runtime guardrails to block unsafe actions or induce excessive caution, which severely stalls task progress instead of actively resolving the underlying risks. To break this safety–progress trade-off, we introduce the Self-Evolving Just-In-Time Memory framework, which reframes embodied safety from progress-stalling guardrails to proactive hazard mitigation. The framework consists of a Risk-Sufficient Topological Belief Graph (RSG) for persistent safety-relevant state tracking under partial observability, an Agency-Grounded Factual Memory for precise hazard anticipation, and an Experience Memory that injects procedural Meta-Skills to guide executable, progress-preserving mitigation. Furthermore, we propose an automated Test-Verify-Write loop, allowing agents to continually refine their mitigation Meta-Skills from execution traces at test time. Experiments on IS-Bench demonstrate that our framework substantially boosts the Safe-Success rate across multiple VLM backbones (e.g., +30.3% on Qwen3-VL-8B), enabling agents to proactively mitigate hazards without stalling task progress. Code is available at `https://github.com/DyMessi/JIT-Memory`.

Keywords: Embodied Planning · Interactive Safety · Agent Memory · Test-Time Evolution

1 Introduction

Recent advances in Vision-Language Models (VLMs) have significantly empowered embodied agents to interact with the complex physical world, enabling them to decompose high-level, natural-language instructions into executable plans through closed-loop interaction [5, 13, 15, 23]. However, while these agents excel at driving toward task completion, they frequently fail to handle dynamically emerging physical risks during these interactions, resulting in accumulated safety hazards [9, 22].

⋆ Corresponding authors.

Current research on embodied interactive safety can be broadly categorized into two paradigms. The first focuses on *malicious-goal safety* [16, 28, 30], where the task objective itself is inherently unsafe (e.g., pouring water on an appliance). For such tasks, the prevailing defense is to deploy runtime guardrails that refuse the command or block execution progress [18, 19]. While effective for preventing explicit hazards, these mechanisms inherently treat safety as task refusal rather than hazard resolution. The second paradigm concerns *benign-goal interactive safety* [9], a more realistic setting where the primary goal is harmless, but hazards emerge dynamically through interaction and environmental state changes. Rather than simply aborting the task, the agent must proactively identify latent risks during closed-loop planning and execute mitigation steps without stalling progress. For instance, if the task goal is "place an apple on a plate" but the plate is dusty, a safe agent should wipe the plate before proceeding instead of abandoning the task. However, existing methods fail to support this proactive, progress-preserving safety.

Achieving this proactive safety requires first formalizing the nature of dynamically emerging hazards. We posit that interactive safety is not an intrinsic property of the environment alone, but a bipartite function of the environmental state and the agent's agency (actionable intents). First, *Situational Risks* act as intent-conditioned negative affordances, where a local state becomes hazardous when coupled with an imminent intent (e.g., a dusty plate is safe until the agent intends to place food on it). Second, *Temporal Risks* manifest as persistent unsafe local states (e.g., a running faucet or a flammable object left near a heat source) that demand appropriate resolution timing before task termination.

However, proactively managing these bipartite risks requires three synergistic capabilities that current Vision-Language Model (VLM) driven agents largely lack. First, agents must maintain *persistent state tracking* across timesteps, as relying solely on immediate visual perception leads to perceptual forgetting of occluded hazardous states under partial observability. Second, agents require *intent-conditioned hazard anticipation* to recognize when an imminent action will transform a currently benign state into a situational risk. Third, agents need *executable mitigation guidance* with precise grounding and timing to resolve hazards without stalling overall task progress.

To equip agents with these capabilities, we propose a Just-In-Time Memory framework that activates safety interventions only when triggered by specific state–intent couplings or persistent risk states. Our framework coordinates three specialized modules corresponding to the aforementioned capabilities: (i) A *Risk-Sufficient Topological Belief Graph (RSG)* serves as the working memory, employing an action-conditioned patch update to persistently track safety-relevant states and relations under partial observability without requiring full-scene reconstruction. (ii) An *Agency-Grounded Factual Memory* compiles abstract human safety norms into triggerable, verifiable rules mapped to the agent's action space, enabling precise, just-in-time hazard anticipation when local RSG states couple with imminent intents. (iii) An *Experience Memory* injects decontextualized procedural *Meta-Skills*, providing actionable guidance on *how* to

mitigate risks (grounding) and *when* to resolve obligations (timing) without sacrificing task progress. Crucially, our system operates within a self-improving *Test-Verify-Write loop*. By programmatically verifying execution traces against factual rules using historical RSG snapshots, the agent continuously mines verified case patches to refine its Meta-Skills at test time, substantially improving proactive safety capabilities without manual annotation.

Our contributions are threefold:

- We introduce a state-agency conditioned formalization of interactive safety, modeling hazards as bipartite functions of environmental state and agent's intent. This transforms abstract human safety norms into triggerable and verifiable rules aligned with the agent's action space, providing a computable foundation for proactive safety.
- We introduce a Just-In-Time Memory framework that integrates a Risk-Sufficient Topological Belief Graph for persistent state tracking, Agency-Grounded Factual Rules for precise hazard triggering, and Procedural Meta-Skills for executable mitigation. This coordinated design enables targeted safety interventions and effectively resolves the trade-off between task efficiency and safety assurance.
- We propose an RSG-driven Test–Verify–Write mechanism for test-time evolution. By programmatically verifying execution traces without manual annotation, this loop distills both successful and failed cases into evolvable Meta-Skills, continuously refining the grounding and timing of mitigation to achieve substantial safe-success gains across multiple VLM backbones.

2 Related Work

2.1 Memory-Augmented Embodied Planning

Recent advancements in Vision-Language Models (VLMs) have substantially advanced the high-level planning capabilities of embodied agents, enabling them to translate natural-language goals into long-horizon, executable action sequences [1, 2, 14, 25, 33]. To improve execution robustness under closed-loop interactions and partial observability (POMDP), recent research has increasingly integrated external memories and structured scene representations into the planning loop [4, 7, 13, 26]. For instance, frameworks like KARMA [21] and RoboMemory [8] employ dual-memory architectures or persistent scene graphs to maintain consistent environmental beliefs, while Voyager [17] builds a continually expanding skill library that stores reusable programs to support long-horizon embodied planning. While these memory-augmented agents heavily optimize for task completion and dense scene reconstruction, they still lack proactive safety awareness to anticipate and mitigate the dynamic physical hazards that emerge iteratively during interactions.

2.2 Safety in VLM-Driven Embodied Agents

As embodied agents become increasingly autonomous, safety evaluation has naturally transitioned from static visual question-answering (VQA) [31, 32] to interactive, simulation-based physical execution [10, 11, 16, 20]. Early embodied safety benchmarks primarily evaluate instruction-level vulnerabilities. For example, BadRobot [30] and SafeAgentBench [28] predominantly focus on the agent's ability to abort tasks when confronted with hazardous instructions, as well as its compliance with explicitly stated temporal safety constraints [29]. Consequently, the prevailing defense mechanisms act as runtime guardrails [19]. Methods such as Agentspec [18] and Plug in the Safety Chip [27] employ Linear Temporal Logic (LTL) or domain-specific languages (DSLs) to dynamically monitor and block unsafe behaviors.

While effective for intercepting explicit hazards, these reactive interventions inherently abort task progress. Crucially, they struggle with the household scenarios evaluated by IS-Bench [9], where task instructions are semantically neutral and risks emerge dynamically from environmental state changes, requiring agents to proactively mitigate hazards and safely complete the task rather than simply halting execution. Addressing this critical gap, our Just-In-Time memory framework empowers the agent with executable, context-aware Meta-Skills to seamlessly anticipate and resolve interactive hazards without stalling overall task progress.

3 Problem Setup & Risk Formulation

To develop a proactive safety mechanism, we formulate a unified state-agency conditioned framework that categorizes interactive hazards into situational risks and temporal risks.

3.1 Closed-loop Embodied Planning under POMDP

We model household embodied tasks as closed-loop interactions under a Partially Observable Markov Decision Process (POMDP). At each step t, the agent observes o_t and executes a parameterized skill action a_t and the environment transitions and returns a new observation o_{t+1}. Since the agent never accesses the full true state s_t, it must maintain a persistent belief state $b_t \approx p(s_t \mid o_{0:t}, a_{0:t-1})$ over historical observations and actions for safe-decision-making. Within our framework, we explicitly instantiate b_t to comprise object entities, their semantic attributes, dynamic unary states, and sparse topological relations.

3.2 State-Agency Conditioned Interactive Hazards

Under semantically neutral household instructions, we argue that interactive safety is not solely an environmental property, but a bipartite function of the environmental state and the agent's agency (actionable intents). Based on this

perspective, we formalize interactive hazards into two distinct computable formulations: Situational Risks (h^{sit}) and Temporal Risks (h^{tem}).

1. Situational Risks (Intent-Conditioned Negative Affordances)
Situational risks emerge when a specific local environmental state is coupled with an imminent action intent. The same local state can be completely benign under one intent but extremely hazardous under another, acting as a "negative affordance" triggered by the agent's agency (e.g., a fragile item resting on a countertop is safe until the agent plans to wipe that surface).

Let $\tilde{a}_t$ denote an imminent actionable intent at step t. The situational risk function evaluates whether the coupling of the concurrent belief state b_t and intent $\tilde{a}_t$ violates any predefined situational safety rule $r \in \mathcal{R}_{\text{sit}}$:

$$h^{\text{sit}}(b_t, \tilde{a}_t) = \mathbf{1}[\exists r \in \mathcal{R}_{\text{sit}}, r(b_t, \tilde{a}_t) = 1]. \tag{1}$$

Consequently, mitigating a situational risk requires proactively transitioning the environment to a safe state prior to executing $\tilde{a}_t$. Instead of blocking execution entirely, the agent must resolve the underlying state conflict to safely proceed with its original subgoal.

2. Temporal Risks (State-Induced Persistent Obligations)
Unlike situational risks that require mitigation before executing a specific intent, temporal risks represent persistent unsafe local states that demand resolution before task termination. These may arise from the agent's past actions (e.g., a faucet left running, a stove left on) or pre-exist in the environment (e.g., a rag on top of the floor that may cause slipping).

Temporal risks do not necessarily demand immediate intervention, as premature mitigation might disrupt ongoing subgoals. However, they impose a persistent obligation that requires long-horizon state tracking to ensure appropriate resolution timing. Let $\mathcal{U}$ denote the set of rules defining persistent unsafe states. The temporal risk function is evaluated as:

$$h^{\text{tem}}(b_t) = \mathbf{1}[\exists u \in \mathcal{U}, u(b_t) = 1]. \tag{2}$$

Summary: This state-agency formulation dictates that an effective safety system must persistently track risk-relevant states in b_t and act as a just-in-time intervention mechanism that triggers exclusively when an intent $\tilde{a}_t$ conflicts with the current state or a persistent temporal risk exists.

4 Method

Proactive interactive safety requires three core capabilities: (i) persistent tracking of risk-relevant states under partial observability, (ii) intent-conditioned hazard anticipation before acting, and (iii) executable mitigation with precise grounding and timing to preserve task progress. To fulfill these requirements without overwhelming the agent, we propose a Just-In-Time Memory framework that coordinates three specialized memory modules (Working Memory, Factual Memory, and Experience Memory) within a Test-Verify-Write evolution loop.

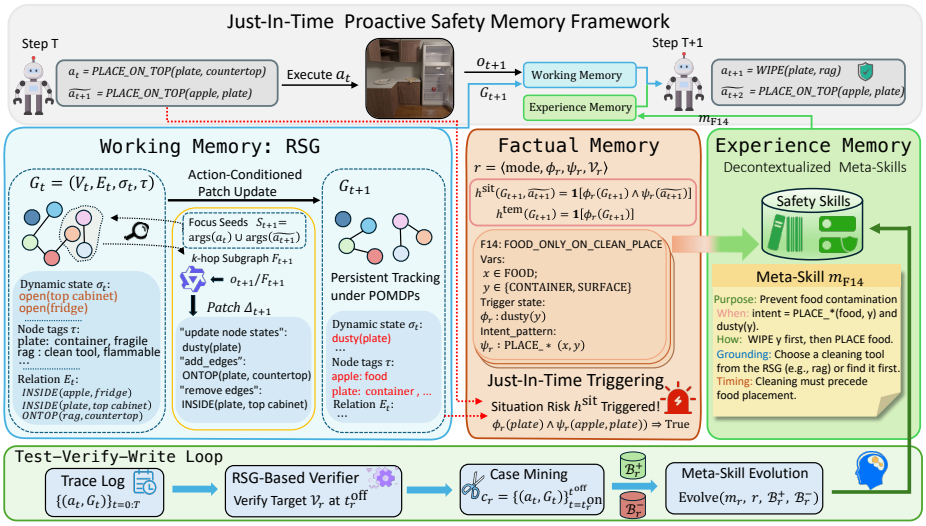

Fig. 1: Architecture of the Just-In-Time Memory framework. The *Working Memory* tracks risk states via action-conditioned RSG patches. The *Factual Memory* triggers interventions for situational and temporal risks based on state-intent couplings or persistent states. When triggered, the *Experience Memory* injects procedural Meta-Skills for progress-preserving mitigation, which self-evolve via a *Test-Verify-Write Loop*.

4.1 Overview: The Just-In-Time Memory Loop

Our Just-In-Time Memory System embeds proactive safety awareness into closed-loop embodied planning under POMDPs through a structured, multi-stage pipeline at each timestep (illustrated in Fig. 1).

State Tracking via Working Memory. At step t, the planner receives observation o_t and produces an immediate action a_t alongside a predictive one-step intent $\tilde{a}_{t+1}$. Upon executing a_t, an auxiliary VLM updates the Risk-Sufficient Topological Belief Graph (RSG, G_{t+1}) via an action-conditioned patch mechanism. This RSG persistently tracks safety- and planning-relevant states under partial observations without the overhead of full-scene reconstruction.

Anticipation and Mitigation via Factual & Experience Memory. Instead of serving as a reactive guardrail that simply blocks execution, this predictive intent $\tilde{a}_{t+1}$ acts as a proactive lookahead probe. Guided by the updated G_{t+1}, our Agency-Grounded Factual Memory explicitly determines whether this intent couples with local states to form situational risks (h^{sit}), or if persistent temporal risk states (h^{tem}) exist. If no risks emerge, the memory remains dormant to prevent over-caution. Conversely, if a rule is triggered, a corresponding procedural Meta-Skill from the Experience Memory is injected into the planner's prompt. This supplies explicit grounding and timing guidance to safely generate the actual executable next action a_{t+1} without aborting task progress.

Test-Time Evolution via Test-Verify-Write Loop. Finally, upon episode completion, execution traces are programmatically verified against triggered factual rules using historical RSG snapshots. This automated verification enables mining of successful mitigation cases to continuously refine Meta-Skills during testing. Collectively, this loop ensures the agent's safety awareness and resolution capabilities systematically evolve over time.

4.2 Working Memory: Risk-Sufficient Topological Belief Graph

Closed-loop planning under partial observability requires persistent tracking of risk-relevant states to prevent agents from forgetting latent hazards. We therefore explicitly instantiate the belief state b_t as a Risk-Sufficient Topological Belief Graph (RSG):

$$G_t = (V_t, E_t, \sigma_t, \tau), \tag{3}$$

where V_t represents task-relevant object nodes and E_t denotes sparse topological relations (e.g., `INSIDE`, `NEXTTO`). Crucially, to abstract away dense visual noise and decouple risks from specific object instances, we define a finite but sufficient vocabulary for the RSG. For each node v, $\sigma_t(v)$ stores dynamic unary states from a finite set (e.g., `open`, `toggled_on`), and $\tau(v)$ assigns static functional/safety tags (e.g., `SAFETY_FLAMMABLE`). By treating physically disparate objects (e.g., a "rag" and a "sponge") simply as a unified `FUNCTION_CLEANING_TOOL`, the graph preserves only the minimal semantic abstractions necessary to evaluate safety. This makes the representation *risk-sufficient* with respect to the evaluated factual rule sets (the complete vocabulary is detailed in Appendix A.1). Furthermore, this finite vocabulary perfectly aligns with our Factual Memory (Sec. 4.3), enabling generalizable hazard detection across open-world environments.

To maintain G_t under POMDPs without perceptual instability and belief inconsistency from full-scene re-parsing, we employ an action-conditioned patch update. Given the executed action a_t and predictive intent $\tilde{a}_{t+1}$, we first define a local focus seed set $S_{t+1} = \mathrm{Args}(a_t) \cup \mathrm{Args}(\tilde{a}_{t+1})$ and extract its k-hop neighborhood subgraph $F_{t+1} = \mathrm{Hop}_k(S_{t+1}; G_t)$. An auxiliary VLM then compares the new observation o_{t+1} with this local context to generate an update patch:

$$\Delta_{t+1} = \mathrm{VLM\_PatchUpdate}(o_{t+1}, a_t, \tilde{a}_{t+1}, G_t[F_{t+1}]). \tag{4}$$

Specifically, the patch Δ_{t+1} comprises localized structural modifications, including the addition of newly revealed nodes and edges within the focus region in o_{t+1}, the updating of dynamic states σ_{t+1}, and the removal of obsolete relations (examples are provided in Appendix A.1). Finally, the patch is programmatically merged into the global graph $G_{t+1} = \mathrm{Merge}(G_t, \Delta_{t+1})$. This incremental update naturally preserves currently occluded hazardous states, providing the robust temporal tracking required for persistent obligations.

4.3 Factual Memory: Agency-Grounded Rules and Just-In-Time Triggering

A core philosophy of our framework is that safety is not an intrinsic property of the environment alone, but a function of the agent's agency. Abstract human safety norms (e.g., "prevent fires") are non-actionable until mapped to the agent's specific affordance boundaries. To bridge this gap, we propose a *Norm-to-Affordance Grounding* mechanism.

Specifically, an automated offline compilation process translates a finite set of general safety principles into a structured, computable rule schema aligned with the agent's action space. Since these rules are defined over the semantic RSG vocabulary rather than specific object instances, they can be automatically instantiated across diverse environments (details and examples are provided in Appendix A.2).

$$r = \langle \text{mode}, \phi_r, \psi_r, \mathcal{V}_r \rangle \tag{5}$$

where mode $\in \{\text{Situational}, \text{Temporal}\}$ defines the risk type. Crucially, these compiled factual rules r directly instantiate the situational and temporal rule sets ($\mathcal{R}_{sit}$ and $\mathcal{U}$) formalized in Sec. 3.2.

Instance-Decoupled Generalization. The component ϕ_r defines the hazardous local state pattern. Crucially, rather than relying on specific object instances, ϕ_r is constructed strictly using the finite tag and predicate vocabulary shared with the RSG. For example, a fire hazard rule is defined abstractly as `NEXTTO(SAFETY_FLAMMABLE, FUNCTION_HEAT_SOURCE)`. This decoupling ensures that a single factual rule automatically generalizes zero-shot to any novel object combinations satisfying the tag conditions in open-world scenarios.

Just-In-Time Triggering via Lookahead. The component ψ_r defines the actionable intent pattern. Unlike traditional memory systems that retrieve dense historical trajectories via noisy semantic similarity, our factual rules remain entirely dormant until deterministically activated by an exact subgraph match on the RSG. Before the planner executes a_{t+1}, the predictive one-step intent $\tilde{a}_{t+1}$ from step t acts as a proactive lookahead probe. A situational risk is triggered only if the current state G_{t+1} and the imminent intent $\tilde{a}_{t+1}$ jointly match the rule ($h^{\text{sit}}(G_{t+1}, \tilde{a}_{t+1}) = 1$). Meanwhile, temporal risks are continuously monitored for persistent hazardous states ($h^{\text{tem}}(G_t) = 1$). This precise triggering effectively prevents "over-caution" in normal operations by intervening only when necessary.

RSG-Based Instantiation and Verification. Once triggered, the component $\mathcal{V}_r$ specifies the exact state constraints required to resolve the risk. The verification target $\mathcal{V}_r$ is formulated as an abstract logical formula over tags and predicates (akin to Linear Temporal Logic). For instance, resolving the previously defined fire hazard requires the condition "$\neg$`NEXTTO(SAFETY_FLAMMABLE, FUNCTION_HEAT_SOURCE)`" to be met before task termination. Each triggered factual rule is dynamically instantiated by binding the tags to concrete objects within the current RSG. Because the RSG incrementally tracks these identical tags and predicates, $\mathcal{V}_r$ translates directly into computable subgraph queries over historical RSG snapshots. This enables the agent to programmatically verify its

own execution traces, forming the deterministic foundation for our self-evolution mechanism (Sec. 4.4).

4.4 Experience Memory: Decontextualized Meta-Skills and RSG-Driven Evolution

While Factual Memory specifies *what* hazards are triggered (via $h^{\text{sit}}/h^{\text{tem}}$) and *when* to verify its resolution (via $\mathcal{V}_r$), it does not instruct the planner *how* to mitigate a risk in an executable, scene-grounded, and progress-preserving manner. Traditional memory-augmented embodied agents typically rely on retrieving raw *episodic* trajectories based on scenario similarity. However, these raw cases lack generalization, forcing the planner to implicitly deduce the underlying mitigation logic and struggle when adapting to novel environments.

To overcome this, our Experience Memory paradigm shifts from episodic stacking to procedural learning. By decontextualizing verified execution traces, we distill raw trajectories into generalizable, evolvable Procedural Meta-Skills that explicitly guide mitigation strategy. For each factual rule r, the experience entry is maintained as

$$\mathcal{E}_r = \langle m_r,\ \mathcal{B}_r^+,\ \mathcal{B}_r^- \rangle, \tag{6}$$

where $\mathcal{B}_r^+$ and $\mathcal{B}_r^-$ are bounded buffers of verified successful and failed case patches, and m_r is the distilled meta-skill. Crucially, m_r structures mitigation along functional dimensions:

$$m_r = \langle \textsc{Purpose},\ \textsc{When},\ \textsc{How},\ \textsc{Grounding},\ \textsc{Timing},\ \textsc{Constraints} \rangle. \tag{7}$$

When a risk is triggered, injecting the highly structured m_r alongside a single verified case c_r (sampled from $\mathcal{B}_r^+$) into the planner prompt provides precise guidance while maintaining a bounded context window. For situational risks, m_r primarily provides How/Grounding, instructing the planner on how to leverage the RSG for safe parameter selection and execute immediate mitigating actions without stalling task progress. For temporal risks, m_r emphasizes Timing, instructing the planner on the optimal moment to resolve a persistent risk state, avoiding both premature task disruption and delayed safety failures.

To continuously refine Meta-Skills at test time, we introduce an automated self-evolution mechanism. Leveraging the deterministic verifiability established in Sec. 4.3, the system automatically evaluates execution traces to determine risk resolution and mine causal case patches. For each triggered rule r, let t_r^{on} denote its trigger timestep and let t_r^{off} be the earliest timestep where $\mathcal{V}_r$ is satisfied. We extract a minimal causal case patch

$$c_r = \{(a_t, G_t)\}_{t=t_r^{\text{on}}}^{t_r^{\text{off}}}. \tag{8}$$

and route it to the respective buffer $\mathcal{B}_r^+$ or $\mathcal{B}_r^-$. (The detailed automated verification logic and causal case mining process are described in Appendix A.3). Once

a buffer reaches capacity (e.g., $|\mathcal{B}_r^+| = K$) or accumulates repeated failures, the agent itself updates the meta-skill by contrasting newly observed evidence:

$$m_r \leftarrow \text{Evolve}(m_r,\ r,\ \mathcal{B}_r^+,\ \mathcal{B}_r^-). \tag{9}$$

Here, Evolve$(\cdot)$ denotes verifier-guided empirical refinement of the stored Experience Memory, grounded in RSG-based verification outcomes. Importantly, this Test-Verify-Write loop ensures the agent does not merely memorize scenarios, but explicitly evolves three core interactive safety capabilities: (i) **Mitigation guidance** (How) becomes more accurate and progress-preserving; (ii) **Contextual grounding** (Grounding) becomes more robust in novel scenes; and (iii) **Resolution scheduling** (Timing) becomes better calibrated to avoid premature task disruption or delayed safety failures.

5 Experiments

5.1 Experiments Setup

Benchmark and Metrics. We evaluate on IS-Bench [9], an interactive safety benchmark built on the OmniGibson simulator, comprising 161 semantically neutral household tasks and 388 latent risks embedded throughout execution, requiring agents to proactively mitigate these hazards during closed-loop planning without task abortion. We adopt the four standard evaluation metrics natively defined in IS-Bench: Task Success (TS), Safe Success (SS) (strict task completion with full risk mitigation), and two risk-specific metrics, Pre and Post, which measure hazard mitigation rates timed before and after designated risk-prone actions respectively. Conceptually, the Pre and Post metrics broadly align with the mitigation of our formalized situational and temporal risks. Formal definitions of the metrics are provided in Appendix C.1.

Evaluation Models and Baselines. We integrate our Just-In-Time Memory across both open-source (Qwen3-VL-8B and 32B [24]) and proprietary Vision-Language Models (GPT-4o [6]). We focus our detailed analysis on Qwen3-VL-8B since it substantially improves closed-loop embodied task success over prior open-source backbones, reducing confounding failures due to insufficient base planning ability. Our primary baseline is Safe-CoT [9], which injects pre-generated global safety tips and prompts step-wise safety reasoning at every planning step. Additionally, GPT-5.2 [12] and Gemini-3.1-Pro-Preview [3] are evaluated under Vanilla and Safe-CoT settings to provide upper-bound proprietary references.

Implementation Details. We utilize Qwen3-VL-32B as the auxiliary VLM for RSG construction and patch updates due to its strong visual perception capabilities in the simulation environment, and the RSG patch update uses a 1-hop local focus region. Initial Meta-Skills for each risk rule are bootstrapped from a single verified safe trajectory drawn from a held-out seed set. Additional implementation details are provided in Appendix B.

Table 1: Main results on IS-Bench. **TS** (Task Success) and **SS** (Safe Success) measure overall performance, while the **Pre** and **Post** metrics broadly align with the mitigation of situational and temporal hazards, respectively.

VLM Backbone	Method	Success Rate (%)		Risk Mitigation (%)	
		TS ↑	SS ↑	Pre ↑	Post ↑
Qwen3-VL-8B	Vanilla	69.3	18.4	17.2	46.2
	Safe-CoT	57.6	30.2	40.0	61.1
	Ours	**71.3**	**48.7**	**46.8**	**89.4**
Qwen3-VL-32B	Vanilla	70.5	25.2	16.7	76.0
	Safe-CoT	71.2	33.3	32.8	63.8
	Ours	**76.9**	**58.7**	**61.5**	**94.1**
GPT-4o	Vanilla	77.7	35.3	15.4	72.0
	Safe-CoT	71.7	31.2	29.5	72.3
	Ours	**81.3**	**66.2**	**70.9**	**96.8**
GPT-5.2	Vanilla	**78.0**	28.0	20.6	71.7
	Safe-CoT	74.7	**45.3**	**39.3**	**77.2**
Gemini-3.1-Pro-Preview	Vanilla	**88.0**	25.8	21.3	61.3
	Safe-CoT	82.0	**46.0**	**38.7**	**81.3**

5.2 Main Results on IS-Bench

As summarized in Table 1, a major challenge in interactive safety is the strict trade-off between task progression and risk aversion. While step-wise reasoning baselines like Safe-CoT increase safety, they induce excessive caution that substantially degrades Task Success (**TS**). In contrast, our Just-In-Time Memory consistently alleviates this trade-off, achieving substantial Safe Success (**SS**) gains without sacrificing task progression (e.g., an absolute improvement of +30.3% over the Vanilla baseline on Qwen3-VL-8B). This demonstrates that embodied interactive safety requires timely hazard mitigation rather than persistent over-caution.

Furthermore, our method significantly improves both **Pre** (situational) and **Post** (temporal) risk metrics across all models. Notably, the **Post** metric increases from 46.2% to 89.4% on Qwen3-VL-8B and from 72.0% to 96.8% on GPT-4o. This highlights that our RSG-based persistent state tracking effectively overcomes the amnesic planning typical of POMDP environments.

Finally, under the proprietary reference setting, Qwen3-VL-8B with our memory framework achieves higher **SS** (48.7%) than GPT-5.2 (45.3%) and Gemini-3.1-Pro-Preview (46.0%) with Safe-CoT, suggesting that structured memory can be more effective than generic step-wise safety reasoning. Importantly, the system's benefits scale with the backbone's foundational capabilities, particularly for mitigating situational risks. Stronger models like Qwen3-VL-32B and GPT-

Table 2: Ablation study of the Just-In-Time Memory System on Qwen3-VL-8B. Each variant removes or replaces a key component of the framework.

Method Variant	Success Rate (%)		Risk Mitigation (%)	
	TS ↑	**SS** ↑	**Pre** ↑	**Post** ↑
Vanilla Qwen3-VL-8B	69.3	18.4	17.2	46.2
A. Triggering Mechanism				
Replace with Dense Retrieval	62.0	24.0	26.8	76.0
B. Working Memory				
Replace with Per-step Re-parse	68.7	40.0	45.2	72.3
C. Experience Format				
Only Episodic Cases	68.0	21.3	28.0	42.0
Only Factual Principles	71.3	28.1	35.6	55.4
Our Full System	**71.3**	**48.7**	**46.8**	**89.4**

4o exhibit substantially larger absolute gains in the **Pre** metric (+44.8 and +55.5 percentage points over Vanilla, compared to +29.6 for the 8B model). This confirms that stronger foundational ability allows the agent to better leverage distilled Meta-Skills, grounding abstract mitigation guidance into context-specific action parameters.

5.3 Ablation Study

To understand the contribution of each component of our method, we conduct ablation studies using the Qwen3-VL-8B backbone. Table 2 summarizes the impact of different design variants on the core metrics.

A. Triggering Mechanism: State–Intent Triggering vs. Dense Retrieval. We replace our deterministic state–intent triggering with a standard dense retrieval baseline, where factual rules are retrieved by computing cosine similarity between global scene descriptions and rule embeddings (baseline implementation detailed in Appendix B.3). As shown in Table 2, this change substantially degrades Safe Success (**SS**) to 24.0%, accompanied by a sharp drop in situational risk mitigation (**Pre**: 26.8%). Dense semantic matching struggles because global scene descriptions contain excessive visual noise, which dilutes localized hazard signals. More importantly, the resulting noisy retrieval distracts the planner, further degrading Task Success (**TS**) to 62.0%.

B. Working Memory: Patch Update vs. Per-step Re-parse. To validate the importance of cross-step state maintenance under POMDP, we replace our action-conditioned patch update with a *Per-step Re-parse* strategy that reconstructs the entire RSG from scratch at each step. This modification reduces **SS** from 48.7% to 40.0%, mainly due to a substantial decline in temporal risk mitigation (**Post**: 89.4% → 72.3%). This degradation arises from two key issues.

First, full re-parsing introduces perceptual amnesia: previously activated hazard states (e.g., a toggled-on faucet) disappear once they fall outside the current view. Second, reconstructing the full graph at every step increases the perceptual load on the VLM, leading to unstable graph predictions and missed states. In contrast, our localized patch update preserves occluded historical states while reducing perception overhead, resulting in a more stable long-horizon belief graph.

C. Experience Format: The Necessity of Procedural Meta-Skills. Finally, we study the representation of experience memory. Once a risk is triggered, supplying *Only Factual* principles yields only modest safety gains, reaching a 28.1% Safe Success rate, as the agent struggles to ground abstract norms into executable actions. Providing raw historical trajectories (*Only Episodic*) performs even worse (21.3% **SS**, 68.0% **TS**), suggesting poor generalization and limited adaptability to novel environments. In contrast, distilling experiences into procedural Meta-Skills achieves optimal performance (48.7% **SS**) by explicitly guiding the agent on *how* to mitigate risks and *when* to schedule their resolution through structured experience.

5.4 Test-Time Evolution Analysis

We evaluate the agent's ability to *self-evolve* via the Test-Verify-Write loop. We partition the 150 IS-Bench episodes into 60 Train and 90 Test episodes. The agent runs sequentially on the Train set. Every 20 episodes, we freeze its current Experience Memory and re-evaluate on the same 90 Test episodes. To avoid cold-start sparsity, memory is initialized with one seed trajectory per risk.

As illustrated in Fig. 2, the system exhibits steady, monotonic improvements. At episode 0, the initial bootstrapped Meta-Skills provide basic hazard awareness (Safe Success (**SS**) rises slightly from the Vanilla 17.8% to 20.0%), but cause a drop in Task Success (**TS**: 63.3% → 60.0%). This initial degradation occurs because the rudimentary Meta-Skills lack accurate mitigation timing; they tend to resolve temporal risk states immediately upon triggering (e.g., turning off a faucet before filling a cup), inadvertently aborting the ongoing task. However, as evolution progresses to episode 60, **SS** robustly climbs to 42.2%, alongside surging situational (**Pre**: 50.0%) and temporal (**Post**: 80.5%) risk mitigation rates. Notably, **TS** recovers and surpasses the baseline (64.4%).

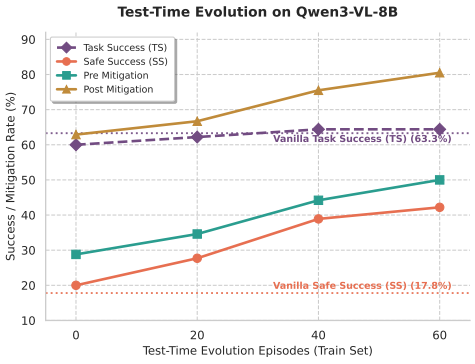

Fig. 2: Test-time evolution on Qwen3-VL-8B. Steady improvements in hazard mitigation demonstrate effective procedural learning.

Crucially, these results indicate that the Test-Verify-Write mechanism does not merely memorize scenes, but executes a continuous cycle of *de-contextualization and restructuring*. By contrasting verified successful causal patches against failure streaks, the agent progressively refines its existing Meta-Skills, distilling episodic experiences into generalized, progress-preserving procedural knowledge (qualitative examples of this evolution are provided in Appendix D.1). Specifically, this self-evolution systematically refines three core cognitive dimensions within the Meta-Skills: (i) **Timing** learns sophisticated dependency scheduling, delaying the resolution of temporal obligations (e.g., waiting for a 'cook' action to finish before toggling off a stove) to avoid premature task disruption; (ii) **Grounding** evolves dynamic, context-aware parameter selection (e.g., explicitly instructing the planner to search for a `FUNCTION_SURFACE` if a preferred `FUNCTION_STORAGE` is unavailable), ensuring robust hazard mitigation across novel floorplans; and (iii) **How** continuously optimizes the structural efficiency of the mitigation action sequence itself. Ultimately, the framework enables the agent to seamlessly transition from being rigidly over-cautious to efficiently safe.

5.5 Further Analysis

To investigate whether enhanced visual perception alone can resolve interactive hazards, we evaluate two baselines (Table 3). First, we test a *Perception-CoT* that forces the agent to explicitly describe objects and spatial relations before planning. This approach yields negligible safety gains (**SS**: 20.0%), as unstructured textual descriptions often overlook latent risks. Second, by injecting the structured RSG directly into the prompt without our experience memories, *RSG-Augmented* slightly improves temporal mitigation (**Post**: 57.5%) but remains ineffective for situational risks (**Pre**: 19.2%). This reveals that even when accurately perceiving a local state, the agent lacks the counterfactual reasoning to anticipate how an imminent action might trigger a hazard. Furthermore, without explicit procedural guidance, the agent frequently neglects temporal risk states once the main task goals are met.

Table 3: Evaluation of perception augmented baselines on Qwen3-VL-8B.

Method	SS ↑	Pre ↑	Post ↑
Vanilla	18.4	17.2	46.2
Perception-CoT	20.0	17.2	42.8
RSG-Augmented	28.0	19.2	57.5
Our Full System	**48.7**	**46.8**	**89.4**

Collectively, the failures of Safe-CoT (being overly cautious), purely perception-augmented approaches (lacking persistent tracking and counterfactual risk awareness), and *Only Factual* rules (lacking execution grounding) demonstrate that proactive safety in embodied scenarios intrinsically requires three core capabilities: (i) persistent tracking of risk-relevant states under partial observability, (ii) intent-conditioned hazard anticipation prior to taking action, and (iii) executable mitigation strategies with precise grounding and timing to preserve overall task progress.

6 Conclusion

In this work, we introduce the Self-Evolving Just-In-Time Memory framework to address the critical challenge of proactive, progress-preserving safety in VLM-driven embodied agents. Building on our formalization of interactive safety as a bipartite function of environmental states and agent intents, our framework shifts the safety paradigm from progress-stalling guardrails to proactive, just-in-time hazard mitigation. Our architecture achieves this by coordinating a Risk-Sufficient Topological Belief Graph (RSG) for persistent state tracking, an Agency-Grounded Factual Memory for precise hazard anticipation, and an Experience Memory that supplies procedural Meta-Skills for executable intervention. Furthermore, our automated Test-Verify-Write loop enables agents to continually refine their Meta-Skills from execution traces at test time. Extensive evaluations demonstrate that our system substantially boosts safe task completion across various VLM backbones. Ultimately, this work provides a scalable, self-improving foundation for deploying more autonomous and safe embodied agents in complex, dynamic environments.

Acknowledgments

We sincerely thank the reviewers for their insightful comments and valuable feedback, which significantly improved the quality of our manuscript. This work is supported by National Natural Science Foundation of China (U22B2017), and International Cooperation Foundation of Hubei Province, China (2024EHA032).

References

1. Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., Julian, R., et al.: Do as i can, not as i say: Grounding language in robotic affordances. In: Conference on robot learning. pp. 287–318. PMLR (2023)
2. Driess, D., Xia, F., Sajjadi, M.S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al.: Palm-e: an embodied multimodal language model. In: Proceedings of the 40th International Conference on Machine Learning. pp. 8469–8488 (2023)
3. Google DeepMind: Model Evaluation – Approach, Methodology & Results: Gemini 3.1 Pro. Tech. rep., Google DeepMind (2026), `https://storage.googleapis.com/deepmind-media/gemini/gemini_3-1_pro_model_evaluation.pdf`, accessed 25 June 2026
4. Hu, Y., Liu, S., Yue, Y., Zhang, G., Liu, B., Zhu, F., Lin, J., Guo, H., Dou, S., Xi, Z., et al.: Memory in the age of ai agents. arXiv preprint arXiv:2512.13564 (2025)
5. Huang, S., Jiang, Z., Dong, H., Qiao, Y., Gao, P., Li, H.: Instruct2act: Mapping multi-modality instructions to robotic actions with large language model. arXiv preprint arXiv:2305.11176 (2023)
6. Hurst, A., Lerer, A., Goucher, A.P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al.: Gpt-4o system card. arXiv preprint arXiv:2410.21276 (2024)

7. Kurenkov, A., Lingelbach, M., Agarwal, T., Jin, E., Li, C., Zhang, R., Fei-Fei, L., Wu, J., Savarese, S., Martın-Martın, R.: Modeling dynamic environments with scene graph memory. In: International Conference on Machine Learning. pp. 17976–17993. PMLR (2023)
8. Lei, M., Cai, H., Cui, Z., Tan, L., Hong, J., Hu, G., Zhu, S., Wu, Y., Jiang, S., Wang, G., et al.: Robomemory: A brain-inspired multi-memory agentic framework for interactive environmental learning in physical embodied systems. arXiv preprint arXiv:2508.01415 (2025)
9. Lu, X., Chen, Z., Hu, X., Zhou, Y., Zhang, W., Liu, D., Sheng, L., Shao, J.: Isbench: Evaluating interactive safety of vlm-driven embodied agents in daily household tasks. arXiv preprint arXiv:2506.16402 (2025)
10. Lu, X., Zhou, Y., Chen, Z., Wang, R., Sima, B., Zhou, E., Sheng, L., Liu, D., Shao, J.: Homeguard: Vlm-based embodied safeguard for identifying contextual risk in household task. arXiv preprint arXiv:2603.14367 (2026)
11. Lu, X., Huang, Z., Li, X., Zhang, C., Xu, W., et al.: Poex: Towards policy executable jailbreak attacks against the llm-based robots. arXiv preprint arXiv:2412.16633 (2024)
12. OpenAI: Update to gpt-5 system card: Gpt-5.2. Tech. rep., OpenAI (2025), `https://cdn.openai.com/pdf/3a4153c8-c748-4b71-8e31-aecbde944f8d/oai_5_2_system-card.pdf`, accessed: 25 June 2026
13. Sarch, G., Wu, Y., Tarr, M., Fragkiadaki, K.: Open-ended instructable embodied agents with memory-augmented large language models. In: Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 3468–3500 (2023)
14. Shi, L.X., Ichter, B., Equi, M.R., Ke, L., Pertsch, K., Vuong, Q., Tanner, J., Walling, A., Wang, H., Fusai, N., et al.: Hi robot: Open-ended instruction following with hierarchical vision-language-action models. In: International Conference on Machine Learning. pp. 54919–54933. PMLR (2025)
15. Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., Garg, A.: Progprompt: Generating situated robot task plans using large language models. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 11523–11530. IEEE (2023)
16. Son, Y., Kim, M., Kim, S., Han, S., Kim, J., Jang, D., Yu, Y., Park, C.Y.: Subtle risks, critical failures: A framework for diagnosing physical safety of llms for embodied decision making. In: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. pp. 25703–25744 (2025)
17. Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., Anandkumar, A.: Voyager: An open-ended embodied agent with large language models. arXiv preprint arXiv:2305.16291 (2023)
18. Wang, H., Poskitt, C.M., Sun, J.: Agentspec: Customizable runtime enforcement for safe and reliable llm agents. arXiv preprint arXiv:2503.18666 (2025)
19. Wang, L., Ying, Z., Yang, X., Zou, Q., Yin, Z., Li, T., Yang, J., Yang, Y., Liu, A., Liu, X.: Robosafe: Safeguarding embodied agents via executable safety logic. arXiv preprint arXiv:2512.21220 (2025)
20. Wang, X., Li, J., Weng, Z., Wang, Y., Gao, Y., Pang, T., Du, C., Teng, Y., Wang, Y., Wu, Z., et al.: Freezevla: Action-freezing attacks against vision-language-action models. arXiv preprint arXiv:2509.19870 (2025)
21. Wang, Z., Yu, B., Zhao, J., Sun, W., Hou, S., Liang, S., Hu, X., Han, Y., Gan, Y.: Karma: Augmenting embodied ai agents with long-and-short term memory systems. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 1–8. IEEE (2025)

22. Xing, W., Li, M., Li, M., Han, M.: Towards robust and secure embodied ai: A survey on vulnerabilities and attacks. arXiv preprint arXiv:2502.13175 (2025)
23. Xu, Z., Wu, K., Wen, J., Li, J., Liu, N., Che, Z., Tang, J.: A survey on robotics with foundation models: toward embodied ai. arXiv preprint arXiv:2402.02385 (2024)
24. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al.: Qwen3 technical report. arXiv preprint arXiv:2505.09388 (2025)
25. Yang, R., Chen, H., Zhang, J., Zhao, M., Qian, C., Wang, K., Wang, Q., Koripella, T.V., Movahedi, M., Li, M., et al.: Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In: International Conference on Machine Learning. pp. 70576–70631. PMLR (2025)
26. Yang, Y., Yang, H., Zhou, J., Chen, P., Zhang, H., Du, Y., Gan, C.: 3d-mem: 3d scene memory for embodied exploration and reasoning. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 17294–17303 (2025)
27. Yang, Z., Raman, S.S., Shah, A., Tellex, S.: 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. IEEE (2024)
28. Yin, S., Pang, X., Ding, Y., Chen, M., Bi, Y., Xiong, Y., Huang, W., Xiang, Z., Shao, J., Chen, S.: Safeagentbench: A benchmark for safe task planning of embodied llm agents. arXiv preprint arXiv:2412.13178 (2024)
29. Ying, Z., Wang, L., Xiao, Y., Wang, J., Ma, Y., Guo, J., Yin, Z., Zhang, M., Liu, A., Liu, X.: Agentsafe: Benchmarking the safety of embodied agents on hazardous instructions. arXiv preprint arXiv:2506.14697 (2025)
30. Zhang, H., Zhu, C., Wang, X., Zhou, Z., Yin, C., Li, M., Xue, L., Wang, Y., Hu, S., Liu, A., et al.: Badrobot: Jailbreaking embodied llms in the physical world. arXiv preprint arXiv:2407.20242 (2024)
31. Zhou, K., Liu, C., Zhao, X., Compalas, A., Song, D., Wang, X.: Multimodal situational safety. In: International Conference on Learning Representations. vol. 2025, pp. 10658–10688 (2025)
32. Zhu, Z., Wu, B., Zhang, Z., Han, L., Liu, Q., Wu, B.: Earbench: Towards evaluating physical risk awareness for task planning of foundation model-based embodied ai agents. arXiv preprint arXiv:2408.04449 (2024)
33. Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., et al.: Rt-2: Vision-language-action models transfer web knowledge to robotic control. In: Conference on Robot Learning. pp. 2165–2183. PMLR (2023)