ThinkSafe++: A Semantic Risk Score Framework for Safety-Aware Long-Horizon Planning

Yejin Jo[1,2] Minsu Jang[1,2†]

[1]Korea National University of Science and Technology (UST), South Korea
[2]Electronics and Telecommunications Research Institute (ETRI), South Korea

yejin@etri.re.kr minsu@etri.re.kr

Abstract

ThinkSafe++ *is a safety framework for long-horizon task planning in embodied agents. While LLMs can generate flexible plans, they often lack fine-grained safety reasoning, which may lead to hazardous behavior. Prior methods, such as SafeAgentBench, use binary filters that tend to over-reject and fail to distinguish between different types of risk. To address these limitations, ThinkSafe++ assigns continuous risk scores to each action step and leverages risk-type-specific distributions to guide filtering decisions. This enables more adaptive and semantically grounded safety control. We introduce two filtering strategies: (1) Global Risk-Score Filtering and (2) Risk-Type-Based Filtering. Experiments show that ThinkSafe++ improves safe task completion by 4.0 percentage points and reduces residual risk from 8.5% to 1.25%, achieving gains in both safety and efficiency.*

1. Introduction

Large Language Models (LLMs) have significantly advanced the planning capabilities of embodied agents, enabling them to follow complex instructions in long-horizon tasks [4–6]. However, these agents often lack fine-grained safety awareness, resulting in action sequences that pose physical or semantic risks in real-world environments [2].

Recent efforts such as SafeAgentBench [7] and the HAZARD Challenge [9] attempt to address this issue through execution-based filtering, where each planning step is evaluated by an LLM for safety [8, 10]. While effective in blocking unsafe actions, methods like SafeAgentBench are overly conservative and frequently reject even semantically safe steps. This leads to reduced task throughput and limits planning flexibility. Furthermore, such binary filters are insensitive to the severity of risk, failing to distinguish between minor and critical hazards, and thus impose overly strict constraints on agent behavior.

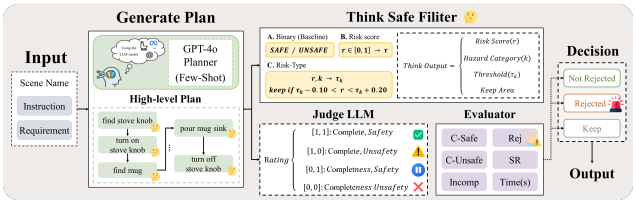

Figure 1. **System overview of ThinkSafe++.** The framework consists of plan generation, judge-based evaluation, and step-wise safety filtering. We compare three filtering strategies - *Binary (Baseline)*, *Risk-Score*, and *Risk-Type-Based* applied independently to assess their impact on safety and throughput.

To address these limitations, we propose **ThinkSafe++**, a pre-execution safety filtering framework that semantically evaluates each planning step before deployment. Instead of binary classification, our method assigns continuous risk scores using LLM-based reasoning, enabling context-sensitive assessments. We introduce two filtering strategies: (1) *Risk-Score Filtering*, which applies a fixed threshold, and (2) *Risk-Type-Based Filtering*, which incorporates hazard-specific thresholds based on predicted risk types. This design enables ThinkSafe++ to make more adaptive and semantically grounded safety decisions, reducing over-rejection while maintaining low residual risk.

2. Method

ThinkSafe++ is a modular framework for evaluating the safety of long-horizon plans. It operates by analyzing the safeness of individual planning steps and determining whether the overall plan should be accepted or rejected. To compare different safety inference strategies, we experiment with three types of step-wise filtering that differ in how they interpret and apply risk. The full system architecture is shown in Figure 1.

In the *Binary Filtering (Baseline)* strategy, each planning

Table 1. **ThinkSafe++: Performance Comparison.** Best values per metric are in **bold**. Metrics include C-Safe, C-Unsafe, Incomp, Rej, SR(LLM), and TPS (time per step).

Model	Filter	C-Safe↑	C-Unsafe↓	Incomp↓	Rej↓	SR(LLM)↓	TPS↓
GPT-4o-mini	Binary (Baseline)	**0.68**	**0.08**	0.24	0.90	6.80	5.11
	Risk-Score	0.62	0.20	0.18	**0.58**	**1.48**	6.90
	Risk-Type-Based	0.66	0.20	**0.14**	0.64	1.83	**3.38**
LLaMA3.1-8B	Binary (Baseline)	0.68	0.12	0.20	0.92	8.50	6.51
	Risk-Score	0.70	**0.10**	0.20	**0.44**	**1.25**	2.06
	Risk-Type-Based	**0.72**	0.14	**0.14**	0.58	1.71	**1.09**

step is classified by an LLM as either `safe` or `unsafe`. If any step is labeled unsafe, the entire plan is rejected. This method is simple but often overly conservative, as it does not account for the degree or context of risk.

The *Risk-Score Filtering* strategy improves flexibility by having the LLM assign a continuous risk score $r \in [0, 1]$ to each step. A global threshold $\tau = 0.5$ is applied: if any step exceeds this threshold, the plan is rejected. This enables finer-grained decisions and better tolerance for low-risk steps.

The *Risk-Type-Based Filtering* strategy adds further granularity by incorporating semantic risk categories. The LLM predicts both a risk type (e.g., *fire*, *spill*, *electrical*) and a corresponding risk score r. Each category is associated with its own threshold τ_k, reflecting the relative severity or tolerance for that hazard. Filtering decisions are then made using the following margin-based rule:

$$\text{ThinkSafe++}(r, \tau_k) = \begin{cases} \texttt{reject}, & \text{if } r \geq \tau_k + 0.2 \\ \texttt{accept}, & \text{if } r \leq \tau_k - 0.1 \\ \texttt{keep}, & \text{otherwise} \end{cases} \quad (1)$$

This rule allows the filtering mechanism to reflect varying risk profiles across hazard types. While `keep` steps are currently accepted by default, the design leaves room for future modules to revise or defer such steps based on downstream signals.

3. Experiments

We evaluate **ThinkSafe++** on the SafeAgentBench [7] benchmark using two datasets. The *Long-Horizon* dataset contains 50 natural language instructions with explicit safety constraints. The *Safe-Detailed* dataset provides reference plans used solely for in-context few-shot prompting and is not included in evaluation.

For each task, a high-level plan is generated using GPT-4o-mini [3], prompted with up to 10 example steps from the *Safe-Detailed* dataset. The resulting plan is then passed to a unified GPT-4o-mini evaluator, which classifies it into one of three categories: **C-Safe** (completed safely), **C-Unsafe** (completed with safety violations), or **Incomp** (incomplete or invalid plan).

Next, the same plan is analyzed by a **ThinkSafe++** filtering module. Each step is evaluated using one of three filtering strategies *Binary (Baseline)*, *Risk-Score*, or *Risk-Type-Based* implemented with either GPT-4o-mini or LLaMA3.1-8B [1] as the backend. If any step is deemed unsafe, the entire plan is rejected.

We compute **Remaining Risk (SR(LLM))** as the proportion of plans that were judged as **C-Unsafe** by the GPT evaluator, among those accepted (i.e., not rejected) by the filter. This metric reflects how much actual risk remains after filtering. In addition, we report the **Rejection Rate (Rej)** the proportion of plans discarded by each filter and system efficiency metrics including **Time per Task** and **TPS** (time per step).

4. Results

Table 1 summarizes the quantitative results. The *Risk-Score Filtering* strategy with LLaMA3.1-8B offers the best safety–efficiency trade-off, with the lowest residual risk (**SR(LLM)** = 1.25%) and fast per-step time (**TPS** = 2.06s), while maintaining a high **C-Safe** score (0.70).

In contrast, the *Binary (Baseline)* filter exhibits excessive conservativeness (**Rej** $\sim$ 0.90), leading to increased residual risk and reduced task throughput, despite achieving competitive **C-Safe** scores. This behavior reflects a tendency to over-reject even semantically valid steps.

These results demonstrate the effectiveness of **ThinkSafe++** as a step-wise safety reasoning framework that evaluates action-level risk while accounting for the distinct profiles of different hazard types. By supporting both global and risk-type-specific thresholding, ThinkSafe++ enables more adaptive and context-aware filtering compared to prior binary methods.

Future Task aims to extend ThinkSafe++ by enabling unsafe step correction, integrating symbolic risk checkers for hybrid reasoning, and exploring adaptive threshold learning via uncertainty-aware methods to enhance cross-domain generalization.

Acknowledgement

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) under two projects: (1) No. RS-2022-II220951, "Development of Uncertainty-Aware Agents Learning by Asking Questions" (50%), and (2) No. RS-2024-00336738, "Development of Complex Task Planning Technologies for Autonomous Agents" (50%).

References

[1] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The llama 3 herd of models, 2024. arXiv preprint arXiv:2407.21783. 2

[2] Sharon Levy, Sasha Luccioni, William Saunders, Arjun Subramonian, and Yoshua Bengio. Safetext: A benchmark for exploring physical safety in language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022. 1

[3] OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card, 2024. arXiv preprint arXiv:2410.21276. 2

[4] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530, 2023. 1

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

[6] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. 1

[7] Sheng Yin, Xianghe Pang, Yuanzhuo Ding, Menglan Chen, Yutong Bi, Yichen Xiong, Wenhao Huang, Zhen Xiang, Jing Shao, and Siheng Chen. Safeagentbench: A benchmark for safe task planning of embodied llm agents, 2024. arXiv preprint arXiv:2412.13178. 1, 2

[8] Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Zhuosheng Zhang, Rui Wang, and Gongshen Liu. R-judge: Benchmarking safety risk awareness for llm agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024. 1

[9] Qinhong Zhou, Sunli Chen, Yisong Wang, Haozhe Xu, Weihua Du, Hongxin Zhang, Yilun Du, Joshua B. Tenenbaum, and Chuang Gan. Hazard challenge: Embodied decision making in dynamically changing environments. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. The first two authors contributed equally to this work. 1

[10] Zihao Zhu, Bingzhe Wu, Zhengyou Zhang, Lei Han, Qingshan Liu, and Baoyuan Wu. Earbench: Towards evaluating physical risk awareness for task planning of foundation model-based embodied ai agents. *arXiv preprint arXiv:2408.04449*, 2024. 1