2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2025)
October 19-25, 2025. Hangzhou, China

2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 979-8-3315-4393-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/IROS60139.2025.11246041

Safety Aware Task Planning via Large Language Models in Robotics

Azal Ahmad Khan[1*], Michael Andrev[1*], Muhammad Ali Murtaza[2], Sergio Aguilera[3], Rui Zhang[1], Jie Ding[1], Seth Hutchinson[4], Ali Anwar[1]

***Abstract*— The integration of large language models (LLMs) into robotic task planning has unlocked better reasoning capabilities for complex, long-horizon workflows. However, ensuring safety in LLM-driven plans remains a critical challenge, as these models often prioritize task completion over risk mitigation. This paper introduces SAFER (Safety-Aware Framework for Execution in Robotics), a multi-LLM framework designed to embed safety awareness into robotic task planning. SAFER employs a Safety Agent that operates alongside the primary task planner, providing safety feedback. Additionally, we introduce LLM-as-a-Judge, a novel metric leveraging LLMs as evaluators to quantify safety violations within generated task plans. Our framework integrates safety feedback at multiple stages of execution, enabling real-time risk assessment, proactive error correction, and transparent safety evaluation. We also integrate a control framework using Control Barrier Functions (CBFs) to ensure safety guarantees within SAFER's task planning. We evaluated SAFER against state-of-the-art LLM planners on complex long-horizon tasks involving heterogeneous robotic agents, demonstrating its effectiveness in reducing safety violations while maintaining task efficiency. We also verify the task planner and safety planner through actual hardware experiments involving multiple robots and a human.**

I. Introduction

Large Language Models (LLMs) have become pivotal in the advancement of robotic task planning, allowing systems to interpret unstructured instructions, reason about multiagent collaborations, and generate context-aware action sequences [1], [2], [3], [4], [5], [6], [7]. Recent frameworks exemplify this progress, where LLMs orchestrate heterogeneous robots [1], [8], [9], [10]. By leveraging LLM's commonsense reasoning and ability to parse natural language, LLMs bridge the gap between high-level human commands and low-level robotic execution, outperforming traditional logic-based planners [11]. These capabilities are particularly transformative for long-horizon tasks that require dynamic coordination between robots with distinct action spaces, such as transporting objects across cluttered environments [12].

Safety Challenges in LLM-based Task Planning. Despite these advances, safety remains a significant challenge in LLM-based systems [13], [14]. Traditional planners rely on rigid constraints and predefined rules to mitigate risks, but LLMs prioritize task efficiency, often overlooking hazards like spatial conflicts, partial action sequences, or unstable object handoffs [15]. For instance, an LLM might direct a quadrotor to land on a checkout counter while a robotic arm is still active, risking collisions, or generate a plan where a robotic arm executes a [grab] action but omits the critical [place] step. These issues are worsen in long-horizon tasks, where the planner's limited context window forces a trade-off between retaining historical steps for coherence and incorporating safety guidelines. Without explicit safeguards, LLMs inherently lack the prioritization of risk mitigation, leaving systems vulnerable to catastrophic failures.

Multi-LLM Conversation in Planning Module. To address this, we propose SAFER a Safety-Aware Framework for Execution in Robotics. SAFER utilizes multi-LLM collaboration for safety-aware planning. Instead of burdening a single LLM with both task planning and safety checks, we decouple these roles. As shown in the Fig. 1 our framework introduces a Safety Planning LLM that operates alongside the central task planner by providing feedback and enforcing generalized safety rules, such as avoidance of conflict in the workspace and validation of the action sequence, without inflating the context window of the task planner. This LLM intervenes during the planning loop, by providing feedback for unsafe actions (e.g., adding prerequisite checks or retiming steps) while preserving task goals. Furthermore, we formalize LLM-as-a-Judge [16], a safety metric where a dedicated LLM evaluates plans against 15 risk criteria, quantifying violations like invalidated handoffs or proximity hazards. This approach not only mitigates risks but also provides interpretable safety reports, enabling continuous system improvement.

On the execution side in robotics, the motion and control policy must uphold the safety paradigm defined by the task planner. This safety paradigm arises from both the physical requirements of the robot—such as joint position, velocity, and torque limits—and the safety considerations associated with the operational and task spaces. This means that the control policy executing the task must prioritize safety over task performance, invoking additional safety protocols only when necessary to address actual safety violations.

In our work, we integrate a Control Barrier Functions (CBFs) based control framework with a SAFER task planner to enforce safety within the control policy of robotic systems. Our motivation for utilizing CBFs stems from their strong theoretical guarantees [17]. To the best of our knowledge, this is the first work that incorporates safety into an LLM-based task planner while integrating CBFs to ensure safety

*Equal contributions
[1]University of Minnesota – Twin Cities Email: {khan1086, zhan@1386, dingj, aanwar}@umn.edu
[2] Institute for Robotics and Intelligent Machines (IRIM), Georgia Institute of Technology, AtlantaEmail: mamurtaza@gatech.edu
[3] Department of Mechanical and Metallurgical Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, ChileEmail: sfaguile@uc.cl
[4] Khoury College of Computer Sciences, Northeastern UniversityEmail: s.hutchinson@northeastern.edu

at the level of the robotic control policy.

Contributions. We validate our framework through extensive experiments, including simulation tests on a heterogeneous robot task planning benchmark and real-world evaluations using two robot arms. Our experimental results demonstrate the practical effectiveness and enhanced safety of our approach. The key contributions of this paper are as follows:

- **C1** We propose a novel framework by incorporating a multi-LLM collaboration in the Planning Module. The framework ensures that safety checks are seamlessly embedded throughout the planning process, enabling more robust and safety-aware task execution.
- **C2** Our approach is evaluated against non-SAFER baselines using multiple reasoning models, both open-source and closed-source. Furthermore, we provide more observations and analysis on the cost and latency of these models in long-horizon robotics tasks.
- **C3** We incorporate Control Barrier Functions (CBFs) within the SAFER framework to enforce a safety-compliant control policy. This integration bridges the gap between high-level task planning and low-level control, ensuring that safety constraints are maintained during execution, even in dynamic environments.

II. Related Works

LLM based Task Planning in Robotics. Traditional task planning frameworks in robotics often struggled with real-world adaptability due to their limited reasoning capabilities [1]. Recently, LLMs have emerged as a promising alternative, demonstrating strong generalization in diverse settings such as household manipulation [18], [19], [20], [21] and navigation [22], [23]. Their ability to execute zero-shot [24], [25] and few-shot [26] reasoning further enhances their applicability. Additionally, LLMs have been leveraged for multi-robot collaboration, breaking down high-level commands into executable steps [27], [28].

However, integrating LLMs into robotic task planning raises safety concerns, as they may generate contextually appropriate but hazardous plans. Recent work [29] has attempted to mitigate this by introducing safety-aware LLM planning pipelines. However, a major challenge in long-horizon tasks is the context-length limitation, which forces a trade-off between retaining task history for consistency and preserving critical safety constraints in prompts. Relying on a single LLM makes this trade-off unavoidable. To address this, we introduce a multi-LLM collaborative framework where different LLMs specialize in task planning and safety feedback. This approach ensures robust safety integration while maintaining long-term coherence, overcoming the limitations of single-LLM methods.

Safety has been a key focus in prior work related to LLM in robotics, such as [30], [31], [32]. For instance, [30] employs Linear Temporal Logic (LTL) to enforce safety constraints, while [31] ensures safety at the level of motion policy. Similarly, [32] addresses safety by accounting for various hazardous scenarios. However, these approaches fall short in addressing dynamically evolving environments or complex human-robot interactions, particularly when multiple humans with distinct roles impose varying safety requirements for others. To overcome these limitations, we propose a novel approach that embeds safety at the control level. This method ensures robust safety enforcement even in the presence of dynamic uncertainty and environmental changes.

The most well-known method for ensuring safety in robotics is Artificial Potential Fields (APF) [33]. APF employs an attraction-repulsion field to enforce safety, but this approach can persistently affect the control output, even when safety constraints are not active. Other notable works include [34], which introduces velocity commands to achieve dynamic safety, and [35], which proposes danger fields to quantify safety risks. However, these methods similarly exert continuous influence on the controller, regardless of whether safety constraints are relevant. In contrast, Control Barrier Functions (CBFs) have emerged as a promising alternative, offering theoretical safety guarantees [17] and a demonstrated connection to APF [36]. CBFs ensure safety by minimally modifying the nominal controller, thereby limiting their impact when safety constraints are inactive. Moreover, CBFs enable real-time safety enforcement and have been successfully applied across various domains including bipedal [37], manipulators [38], [39], swarm robotics [40] and quadrotors [41].

LLM-as-a-Judge. LLM-as-a-Judge has emerged as a novel paradigm for evaluating the safety and feasibility of robotic task plans by leveraging the reasoning capabilities of LLMs [42]. Unlike conventional rule-based safety checks, which rely on predefined constraints, LLM-as-a-Judge introduces a flexible and context-aware evaluation mechanism that can dynamically assess task plans against a set of safety principles. Prior works have explored LLM-based evaluation for programming correctness and factual consistency, but their application in robotic safety assessment is relatively new [43]. By employing LLMs to quantify violations in generated task sequences, this approach provides a structured yet interpretable safety metric. Notably, this method enables real-time risk assessment by evaluating plans against multiple risk criteria, such as action dependencies, spatial constraints, and human-robot interaction risks. The integration of LLM-as-a-Judge within multi-LLM frameworks, as seen in SAFER, represents a promising step toward robust safety-aware task planning, balancing efficiency with error mitigation.

III. SAFER: Safety-Aware Framework for Execution in Robotics

SAFER introduces a multi-LLM collaboration to enhance the safety and robustness of LLM-driven task planning. The framework is composed of four core modules: the Planning Module, Execution Module, and LLM-as-a-Judge, Feedback Module. These components work in collaboration to ensure that safety checks are systematically enforced throughout the task planning and execution pipeline.

Planning Module. As shown in Fig. 1 the Planning Module is responsible for generating structured safety-aware

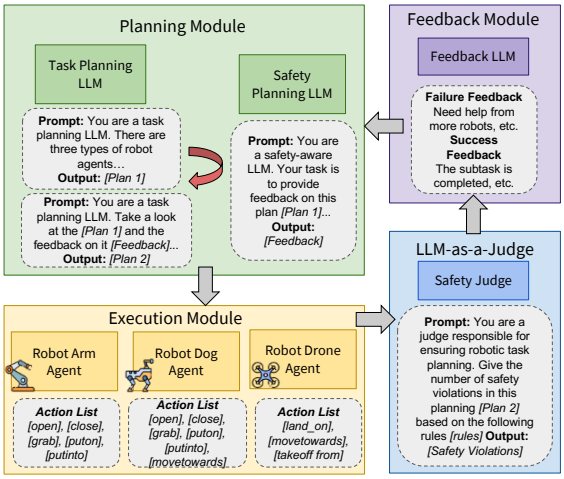

Fig. 1: **Overview of SAFER (Safety-Aware Framework for Execution in Robotics)**. The framework consists of four core modules: (1) Planning Module, where the Task Planning LLM generates initial task sequences while the Safety Planning LLM audits them for potential hazards and iteratively refines the plan; (2) Execution Module, responsible for deploying the task plan through heterogeneous robotic agents, each equipped with an Execution LLM that verifies execution; (3) LLM-as-a-Judge, a dedicated evaluation module that assesses the final task plan for safety violations, quantifying risk factors such as spatial conflicts and incomplete action sequences; and (4) Feedback Module, processes execution outcomes and provides corrective feedback to the Planning Module, ensuring real-time adaptation.

action sequences for heterogeneous robotic agents. Unlike single-agent planning models, SAFER integrates two specialized LLMs to balance task efficiency with safety enforcement. It utilizes a Task Planning LLM that iteratively refines task plans based on external feedback. The Task Planning LLM serves as the primary orchestrator for action sequencing. Given a high-level task description, agent capabilities, and environmental observations, the Task Planning LLM decomposes the goal into executable subtasks. To introduce safety awareness, the Safety Planning LLM works in collaboration with the primary planner. This safety-focused LLM provides safety-related feedback on the generated task sequence to identify potential hazards, including spatial conflicts, invalid action dependencies, and omitted preconditions. The feedback generated by this LLM is integrated into the planning LLM, ensuring that the final plan adheres to generalized safety constraints while remaining efficient.

Why multi-LLM collaboration over Retrieval-Augmented Generation (RAG) or Few-Shot Learning (FSL)? While techniques like RAG and FSL can help retrieve relevant information and integrate it into the prompt of a central task planner, they do not address the context window limitation inherent in LLMs. These approaches still rely on fitting all necessary details within a single LLM's prompt, which constrains long-horizon reasoning and safety enforcement. In contrast, our multi-LLM collaboration approach decouples task planning and safety feedback, allowing one LLM to focus on structured task generation while another LLM specializes in safety assessment and feedback.

Execution Module. The Execution Module is responsible for deploying the refined task plans into actionable commands for robotic agents. It consists of heterogeneous robot agents (robot arms, quadrotor, and robotic dog) that execute the planned actions. Each robot agent is equipped with one specific Robot Execution LLM. As shown in Fig. 1, the robot agents have a list of actions they can perform. Instead of conventional execution pipelines that rigidly follow predefined trajectories, SAFER enables real-time adaptability by verifying the feasibility of actions before execution. Once tasks are assigned to specific robot agents, the Robot Execution LLM predicts whether the task is executable. If a task fails, the Feedback Module analyzes the failure and sends the feedback to the planning module.

Feedback Module. As shown in Figure 1, the feedback module takes the current state input including task goals, robot capabilities, and execution progress. The module suggests corrections after action failures and progress updates after successful executions. The module generated two forms of feedback: *Failure Feedback* and *Success Feedback*. The *failure feedback* is triggered when an action cannot be executed due to environmental constraints, insufficient resources, or incorrect sequencing. Examples include a robotic arm failing to grasp an object due to misalignment or a drone unable to take off due to an obstructed flight path. Then in the next step, the planning module refines the task plan by utilizing the output from the feedback module. The *success feedback* is issued when a subtask is successfully completed, allowing the transition to the next execution phase. This feedback ensures that SAFER maintains coherence across long-horizon tasks.

IV. Safety and Control

To ensure compliance with the safety metrics prescribed by SAFER, we employ Control Barrier Functions (CBFs). CBFs ensure safety by first designing a safe set, then using the CBFs to ensure forward invariance of the given set i.e. if the system starts with the safe set, it remains in the safe set. CBFs provide theoretical guarantees regarding safety and ensure safety by modifying the nominal controller in a minimally invasive manner. Its architecture can be applied to several control strategies and fast solvers ensure real-time deployment. This ensures safety even when the reference trajectory violates safety constraints. We have designed the constraints for the robotics system following approaches similar to those in [38], [39]. Here, we briefly review the CBFs and associated safety constraints referenced from SAFER for safe task planning. For a comprehensive mathematical understanding, readers are directed to the original papers in [17], [38], [39].

A. Control Barrier Functions

For most robotics systems, the safety constraints of the robotics systems can be defined in terms of their state. States space associated with robotics can be defined in terms of its joint, task, or operational space. As CBFs require first defining a safe set, hence the safe set needs to be designed in terms of the associated joint, task, or operational space. If x is the state associated with joint, task, or operational space, and $\mathscr{C}_0$ defines the safe set such that barrier function, $h(x) \geq 0$, ensures safety. Then the safe set can be written as

$$\mathscr{C}_0 = \{x \in \mathcal{T} \mid h(x) \geq 0\}$$

where $\mathcal{T}$ represents the associated joint, task, or operational space. For most of the robotics systems, input appears by taking one or two derivatives of the safe set. For the second-order systems, the equation is given as

$$\ddot{h}(x) = L_f^2 h(x) + L_f L_g h(x) u$$

where $L_f^2 h(x)$ and $L_f L_g h(x)$ are the Lie derivatives of $h(x)$ and u is the input associated with robotics system. The safety associated with the barrier function, $h(x)$, can then be ensured if the following inequality is ensured

$$L_f^2 h(x) + L_f L_g h(x) u + K\eta \geq 0$$

where $\eta = \begin{bmatrix} h(x) \dot{h}(x) \end{bmatrix}$, and $\dot{h}(x)$ is the first derivative of the barrier function $h(x)$. The coefficients of the matrix, K, are chosen by the principles of pole placement in a closed loop linear system and are defined in more detail in [17]. For the first-order constraint, the inequality simplifies to

$$L_f h(x) + L_g h(x) u + \gamma h(x) \geq 0$$

where γ is a positive number. If we have a nominal controller, u^{nom}, we can ensure the safety constraint by minimally modifying the nominal controller by solving a Quadratic Programming (QP) based optimization problem to yield a safe controller as

$$\begin{aligned} u^* = \arg\min_u \quad & \frac{1}{2}\|u - u^{nom}\|^2 \\ \text{subject to :} \quad & \\ & L_f^2 h(x) + L_g L_f h(x) u + K\eta \geq 0 \end{aligned} \tag{1}$$

The flexibility of the QP-based optimization is that it allows additional constraints associated with the robotics system as well. The safety constraints can be broadly classified into two categories, joint and operational space. We will next define the safety constraints associated with the robotics system without delving into the mathematics of each constraint.

B. Joint Safety Constraints

The safety constraints associated with the hardware properties of the robotics systems fall into the category of joint safety constraints. These safety constraints include joint position limits, joint velocity limits, and torque limits associated with motors for each robotics system. They are generally defined by the hardware manufacturer, and any violation of these constraints results in an inoperative state of the robotics system.

C. Task and Operational Safety Constraints

Task and Operational safety constraints associated with robotics systems include properties like obstacle avoidance, limiting the operational space of the robots as well as singularity avoidance associated with robotics systems such that the robots do not lose their ability to control in any direction. It also includes constraints like not colliding with other robots and objects. These constraints ensure the safety of the controller, the safety of the human involved, and the safety of the robots from being damaged.

D. Goals and Constraints Setting

The LLMs define a sequence of instructions and a set of constraints for each robot. At each step on the sequence, we introduce a parser that translates the robot's subtask into a goal, either defining the desired velocity of the base, pose of the end-effector, or the gripper's state, among others. These goals are updated to achieve each of the steps on the sequence. For the set of constraints, they might be global, which are active through the whole sequence, or might be activated at a given step. The constraints are also transformed using a parser that defines barrier functions $h(x)$ that the optimizer will have to consider during control.

V. Experiments and Results

We evaluate SAFER using the COHERENT benchmark for multi-agent robotic task planning [1]. This benchmark encompasses complex long-horizon tasks that require collaboration between heterogeneous robotic agents, making it an ideal testbed for assessing the effectiveness of safety-aware planning.

Benchmark. The COHERENT benchmark consists of various task scenarios requiring multi-agent coordination. Tasks are categorized according to the number of robots required to complete them as mono-type, dual-type, and trio-type tasks. Mono-type tasks require a single robot, dual-type tasks require two robots, and trio-type tasks require three robots to complete the task. Similar to our Non-Safety baseline [1], we use 40 tasks considering the high cost of closed-source model APIs calls. From each scene, we pick 8 tasks with 2 mono-type, 3 dual-type, and 3 trio-type tasks.

Evaluation Metrics. To quantitatively evaluate SAFER's performance, we use steps and safety violations. Steps highlight the number of execution steps required to complete a task, reflecting efficiency. Safety Violation measures the number of detected safety violations per task. A task refers to the overall goal that needs to be achieved, whereas steps represent the individual actions or subtasks that must be executed sequentially to accomplish the task. In Table I, we report the average of the two metrics over a scene. These metrics provide a comprehensive evaluation framework that captures both task execution efficiency and safety awareness.

Results and Discussion. Table I presents a comparative evaluation of SAFER against a non-safety baseline across the COHERENT benchmark, using both GPT-4o and DeepSeek-R1 models. The results indicate that SAFER significantly reduces safety violations in all scenes while maintaining

	S1		S2		S3		S4		S5	
Model	**AS**	**ASV**	**AS**	**ASV**	**AS**	**ASV**	**AS**	**ASV**	**AS**	**ASV**
WS-4o	15.8	32.6	12.1	23.0	22.3	51.5	12.3	26.9	26.0	39.3
S-4o	12.1	17.6 ↓	11.9	17.1 ↓	14.5	16.8 ↓	14.6	20.4 ↓	22.0	19.5 ↓
S-r1	20.5	5.8 ↓	15.4	6.4 ↓	22.6	12.8 ↓	19.3	5.9 ↓	18.6	8.3 ↓

TABLE I: **Comparison of SAFER on COHERENT Benchmark against non-safety baseline for GPT-4o and DeepSeek-r1 models**. Abbreviations: **S1-S5** (Scenes from COHERENT benchmark), **AS** (Average Steps), **ASV** (Average Safety Violations), **WS-4o** (GPT-4o model without SAFER), **S-4o** (SAFER using GPT-4o model), **S-r1** (SAFER using DeepSeek-r1 model). **Arrows indicate change from WS-4o: ↓ indicates improvements.**

competitive task efficiency. Specifically, for SAFER using the DeepSeek-R1 model (S-4o) achieves a reduction of 77.5% in ASV, demonstrating its effectiveness in enforcing safety rules. Similarly, for the GPT-4o model, SAFER reduces ASV by 47% across all scenarios, underscoring its generalization across different LLMs as planners.

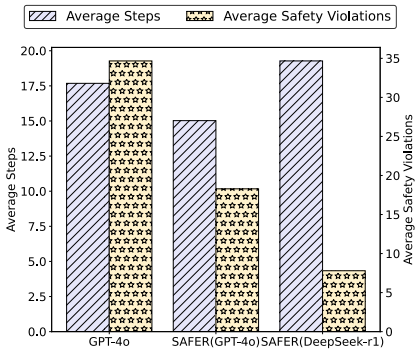

Fig. 2: **Comparative evaluation of SAFER vs. Non-safety baseline.** The figure shows the Average Steps and Average Safety Violations across task scenarios (S1–S5) for baseline and SAFER, highlighting improved safety with minimal impact on efficiency.

Figure 2 highlights the trade-off between task efficiency and safety enforcement in different models. The figure shows the average results across all the scenes in the benchmark. The figure demonstrates a clear reduction in safety violations when using SAFER, validating its ability to mitigate execution risks. However, the non-safety baseline (GPT-4o) exhibits the highest ASV. Moreover, the impact of SAFER's remains consistent between different LLM models, reinforcing its adaptability. This reinforces the central claim that a multi-LLM safety-aware framework can enhance robotic task execution without incurring excessive efficiency trade-offs. Despite better reasoning SAFER (DeepSeek-R1) required more steps compared to SAFER (GPT-4o). This is because R1's improved reasoning significantly reduces safety violations, leading the model to smaller steps to task completion. Furthermore, our results reveal some interesting observations.

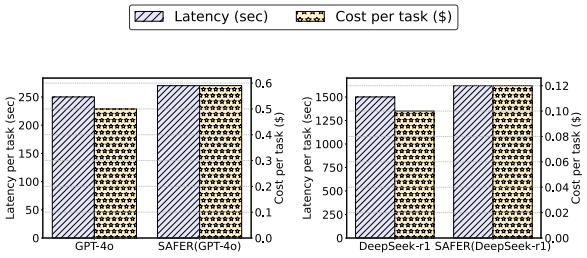

Fig. 3: Latency and cost comparison of SAFER with baseline models. The left plot compares SAFER with GPT-4o, while the right plot compares SAFER with DeepSeek-R1, illustrating low overheads introduced by the safety-aware framework.

Figure 3 highlights the latency and cost overheads introduced by SAFER for GPT-4o and DeepSeek-R1. Although SAFER enhances safety enforcement, it incurs only a minimal increase in computational time and cost. The magnitude of this overhead remains consistent across models. DeepSeek-R1 experiences higher latency due to its sensitivity to load. Furthermore, the consistent overhead is attributed to SAFER introducing two additional API calls per step by the Safety Planning and Safety Judge.

Our evaluation revealed some interesting observations. **OB❶ Stronger reasoning abilities translate to better safety-aware task planning abilities.** We observed that DeepSeek-R1, which has stronger reasoning abilities than GPT-4o, also exhibits significantly fewer safety violations. The reason for this is that more capable reasoners likely generate more structured and foresighted plans, anticipating safety risks in advance rather than reacting to them later because of longer chains that they can generate.

OB❷ LLMs-as-a-Safety-Judge are fair judges rather than answer validators. Previous research has shown that LLMs, when used as answer judges, tend to favor their own responses [44], [45]. However, in our evaluation we do not see this trend when LLMs are used as safety judges. The safety assessment may involve a more explicit rule-based check, where objectivity is required. This shift in task nature may reduce the bias seen in answer validation settings.

OB❸ Balancing efficiency and safety is achievable with a multi-LLM framework. Our results show that SAFER effectively reduces safety violations without introducing excessive steps to complete the tasks. While enforcing safety constraints often leads to longer execution times, the better reasoning capabilities of multi-LLM frameworks help mitigate this trade-off. By leveraging multiple models, SAFER can make informed adjustments that optimize both safety and task efficiency, rather than sacrificing one for the other. This suggests that carefully designed multi-LLM interactions can enable safer task execution without compromising overall performance.

OB❹ LLMs Tend to Prioritize Fast and Efficient Plans Over Detailed Safety Checks. Our experiments show that when LLMs are not given clear instructions to consider safety, they naturally generate plans that are quick

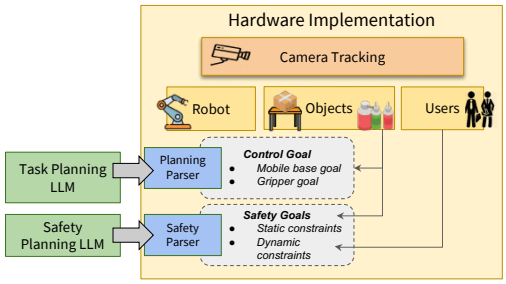

Fig. 4: Overview of the hardware implementation.

and efficient. This means that while the generated plans may require fewer steps, they often miss important safety details. In simpler terms, if safety guidelines aren't explicitly provided, LLMs focus on getting the job done fast rather than carefully checking for potential hazards. This finding highlights the importance of incorporating explicit safety directives when using LLMs in environments where risk management is crucial.

VI. Real Robot Deployment

The outputs generated by LLMs, such as GPT-4o, are expressed in natural language. A representation of the information contained in the prompts is shown in Fig. 5. To convert these outputs into actionable instructions for robotic we propose parsing the natural language descriptions into structured commands, mapping them to robotic control APIs, and ensuring seamless execution. Our approach focuses on creating a robust pipeline that bridges the gap between high-level task specifications and low-level robotic actions while maintaining high accuracy. To accomplish this, we introduce two parsers that consider the robot capabilities, along with objects and users' pose to define the control and safety goals as shown in Fig. 4.

a) Parsing Task Planning LLM Outputs: This LLM outputs a sequence of instructions, such as: `Move Robot 1 to table A, Robot 1 pick the can, Move Robot 1 to table B, Robot 2 pick the box, Robot 2 move box to center of table, Robot 1 place can in box.` These outputs are parsed into structured commands using a custom parser. We separate instructions into their respective robots, identify objects and describe them by their pose in the world, and grasp configuration, finally transform actions into goals. For example, `Move Robot 1 to table A`, table A has position $(0.0, 1.2, 1.0)$ and the action `Move` translate into a new goal pose for the base in $SE(2)$ as $(0.0, 0.6, 90°)$. Other instructions are further split into a subsequence of steps, e.g. `Robot 1 pick the can`, the can has a position $(0.05, 1.1, 1.1)$ and a preconfigured grasp for the gripper to grasp the object. The `pick` instruction is divided into 4 steps: align of gripper over the object, reach for object, close gripper, and lift object.

b) Parsing Safety Planning LLM Outputs: This LLM output a set of constraints for individual robots,

Task Planner - Iteration 1
Input:
System Prompt: You are a task planning LLM ...
Task Goal: Safely transport a cylinder using two collaborating robots while handling human interventions and adjusting speed for non-technical personnel and place cylinder in the box.
Initial Positions: [initial_positions]
Capabilities: [robot_arm_1], [robot_arm_2]
Observations: [robot_arm_1], [robot_arm_2]
Output: [task_planning_1]

Safety Planner
Input:
System Prompt: You are a safety planning LLM ... For the following task goal we recieved a task plan. Your task is to create a safety feedback that the task planning LLM can use to make sure all the safety rules are followed.
Task Goal: [task_goal]
Current Task Planning: [task_planning_1]
Safety Rules: [safety_rules]
Output: [safety_feedback]

Task Planner - Iteration 2
Input:
System Prompt: You are a task planning LLM ...
Task Goal: Modify the following task planning while making sure you incorporate the following safety feedback.
Current Task Planning: [task_planning_1]
Safety Feedback: [safety_feedback]
Output: [task_planning_2]

Fig. 5: Prompts and Output representation of Multi-LLM Collaboration in the Planning Module.

such as: `Robot 1 manipulator must stay away from users, Robot 1 must not collide with table A, Robot 2 must not collide with table B`. These outputs are parsed into structured definitions of control barrier constraints. We can define both dynamic and static constraints. E.g. `not collide with Table A` requires a single query of the table A's position and dimension of the table and we can create a static polygon that encapsulates the object and create a hard constraint for the robot's motion. While `stay away from user` is a dynamic constraint that updates as a user moves and can have different dimensions depending on the user's access level. The robot is constantly checking the user's relative position and the controller ensures that the arm will not violate the user's safe space.

c) Robot Experiment: For our experimental setup, we are using two mobile manipulators composed of two Kuka IIWA LBR 14 robotic arms, which consist of 7 DoF and a clearpath Ridgeback omnidirectional mobile base. Each arm has a two-fingers gripper from Robotiq. For object and robot tracking, we are using a Vicon camera tracking system. For the static objects as tables, we have their global position, for the interacting objects like the spray can and box, we have place markers to track them. To track users, we are using a tracking helmet that gives us the overall position of the person in the workspace. The mobile base uses a PD controller to command the desired twist for the base, wrapped with a CBF. The robotic arm is using the proposed force/torque controller described in IV.

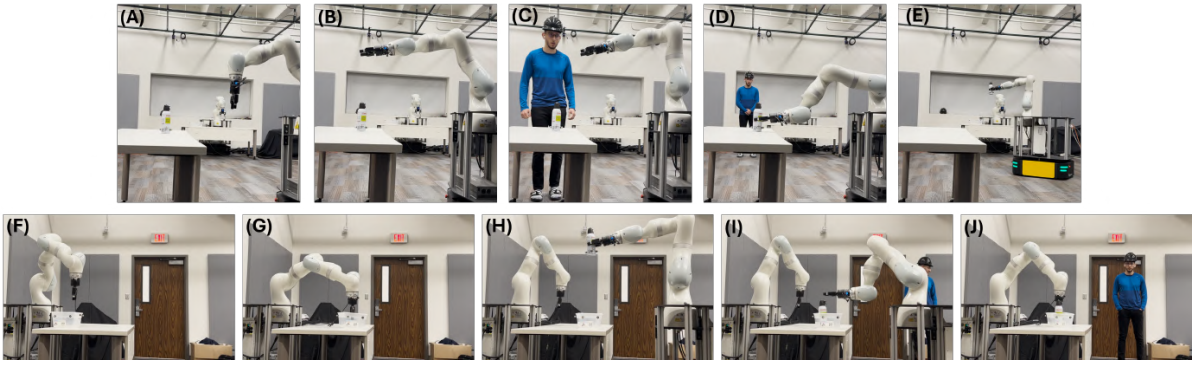

Fig. 6: Steps of the hardware experiment. (A) Robot 1 approaches table A. (B) Robot 1 aligns gripper with spray can. (C) As a person approaches the Robot 1, the manipulator moves back. (D) Robot 1 picks the spray can. (E) Robot 1 moves to table B. (F) Robot 2 aligns gripper with box. (G) Robot 2 picks the box and brings it to the center of the table. (H) Robot 2 leaves box and moves to safe position, while Robot 1 gets closer. (I) Robot 1 places spray can in the box. (J) Robot 1 leaves to the initial position and Robot 2 moves the box to a new location.

For the hardware experiment, we consider two robots, one that can move about the environment, while the second is placed at a fixed point. There are two types of users in the environment that should be considered: i) non-technical personnel, which the robot must keep a distance from, and ii) technical personnel that the robot must slow down its motion when close. We consider the following task: Safely transport a cylinder using two collaborating robots while handling human interventions and adjusting speed for non-technical personnel, and place the cylinder in the box.

The summarized planned task is as follows: Robot Arm 1 navigates to the cylinder using obstacle avoidance, ensures safety, and aligns precisely before lowering and securely grabbing it. It then lifts the cylinder, transports it to the Robot Arm 2 workspace, checks for clearance. Robot Arm 2 verifies safety, aligns, and picks up the box and places it at the center of Table B. Robot Arm 2 releases the box and moves to a safe location. Robot Arm 1, upon arrival at Table B, it checks Robot Arm 2 to be at a safe position, then aligns, lowers, and releases the cylinder safely into the box. Robot Arm 1 moves away from the workspace. Robot Arm 2 verifies that Robot Arm 1 is at a safe position, then aligns, lowers, and picks up the box and places it close to the user. If needed, Robot Arm 1 moves the cylinder cautiously, keeping a distance from non-technical personnel.

We present the hardware execution of the resulting sequence by the task planning LLM in Fig. 6 and in the supporting video*. Along the sequence, the system is modifying the goal for the mobile base to move the Robot 1 around the space and setting the end-effector's goals depending on the task. Along with the basic task allocation, the safety LLM also activates different control barriers that will help maintain safety standards. Over the whole experiment, the robot is able to progress through the sequence, achieving each task and subtasks and successfully finishing the requested task.

*The link to the video: https://www.youtube.com/watch?v=jNTKvqT2Vog

VII. Conclusions

In this work, we introduced SAFER designed to enhance safety in LLM-driven task planning. By employing a multi-LLM collaboration approach, SAFER integrates a dedicated Safety Planning LLM and an LLM-as-a-Judge module to systematically assess and mitigate risks throughout the planning and execution pipeline. Our framework ensures that safety checks are embedded into task plans while maintaining task efficiency. We evaluated SAFER against non-safety baselines in complex multi-robot scenarios, demonstrating a significant reduction in safety violations with minimal impact on execution efficiency. Additionally, we integrate CBFs to enforce safety constraints at the low-level control stage, ensuring real-time safety enforcement. Experimental results, including both simulations and real-world deployments, validate the effectiveness of our approach in reducing hazardous failures while maintaining adaptability across different robotic agents. Future work will explore extending SAFER to more dynamic environments, incorporating adaptive safety feedback, and refining LLM-based evaluation to further enhance transparency and reliability in autonomous robotic systems.

Acknowledgments

The work of Azal Ahmad Khan was supported in part by the Amazon Machine Learning Systems Fellowship and the UMN GAGE Fellowship. Ali Anwar was supported by the Samsung Global Research Outreach Award.

References

[1] K. Liu, Z. Tang, D. Wang, Z. Wang, B. Zhao, and X. Li, "Coherent: Collaboration of heterogeneous multi-robot system with large language models," *arXiv preprint arXiv:2409.15146*, 2024.

[2] H. Zhang, W. Du, J. Shan, Q. Zhou, Y. Du, J. B. Tenenbaum, T. Shu, and C. Gan, "Building cooperative embodied agents modularly with large language models," *arXiv preprint arXiv:2307.02485*, 2023.

[3] J. Duan, W. Pumacay, N. Kumar, Y. R. Wang, S. Tian, W. Yuan, R. Krishna, D. Fox, A. Mandlekar, and Y. Guo, "Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation," *arXiv preprint arXiv:2410.00371*, 2024.

[4] F. I. Dogan, U. Ozyurt, G. Cinar, and H. Gunes, "Grace: Generating socially appropriate robot actions leveraging llms and human explanations," *arXiv preprint arXiv:2409.16879*, 2024.

[5] S. Roychoudhury, A. S. Varde, and W. Wang, "Efficient task organization with commonsense knowledge for human-robot collaborative tasks,"

[6] P. Sikorski, L. Schrader, K. Yu, L. Billadeau, J. Meenakshi, N. Mutharasan, F. Esposito, H. AliAkbarpour, and M. Babaiasl, "Deployment of nlp and llm techniques to control mobile robots at the edge: A case study using gpt-4-turbo and llama 2," *arXiv preprint arXiv:2405.17670*, 2024.

[7] E. Latif, R. Parasuraman, and X. Zhai, "Physicsassistant: An llm-powered interactive learning robot for physics lab investigations," *arXiv preprint arXiv:2403.18721*, 2024.

[8] S. S. Kannan, V. L. Venkatesh, and B.-C. Min, "Smart-llm: Smart multi-agent robot task planning using large language models," *arXiv preprint arXiv:2309.10062*, 2023.

[9] L. Sun, D. K. Jha, C. Hori, S. Jain, R. Corcodel, X. Zhu, M. Tomizuka, and D. Romeres, "Interactive planning using large language models for partially observable robotic tasks," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14054–14061, IEEE, 2024.

[10] Y. Mu, Q. Zhang, M. Hu, W. Wang, M. Ding, J. Jin, B. Wang, J. Dai, Y. Qiao, and P. Luo, "Embodiedgpt: Vision-language pre-training via embodied chain of thought," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[11] Z. Chen, Z. Ji, J. Huo, and Y. Gao, "Scar: Refining skill chaining for long-horizon robotic manipulation via dual regularization," *Advances in Neural Information Processing Systems*, vol. 37, pp. 111679–111714, 2025.

[12] F. Zeng, W. Gan, Y. Wang, N. Liu, and P. S. Yu, "Large language models for robotics: A survey," *arXiv preprint arXiv:2311.07226*, 2023.

[13] X. Wu, S. Chakraborty, R. Xian, J. Liang, T. Guan, F. Liu, B. M. Sadler, D. Manocha, and A. S. Bedi, "Highlighting the safety concerns of deploying llms/vlms in robotics," *arXiv preprint arXiv:2402.10340*, 2024.

[14] Y. Qi, G. Kyebambo, S. Xie, W. Shen, S. Wang, B. Xie, B. He, Z. Wang, and S. Jiang, "Safety control of service robots with llms and embodied knowledge graphs," *arXiv preprint arXiv:2405.17846*, 2024.

[15] Y. Deng, L. Liao, W. Lei, G. Yang, W. Lam, and T.-S. Chua, "Proactive conversational ai: A comprehensive survey of advancements and opportunities," *ACM Transactions on Information Systems*, 2025.

[16] J. Gu, X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, W. Li, Y. Shen, S. Ma, H. Liu, *et al.*, "A survey on llm-as-a-judge," *arXiv preprint arXiv:2411.15594*, 2024.

[17] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*, pp. 3420–3431, Ieee, 2019.

[18] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[19] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg, "Text2motion: From natural language instructions to feasible plans," *Autonomous Robots*, vol. 47, no. 8, pp. 1345–1365, 2023.

[20] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500, IEEE, 2023.

[21] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, *et al.*, "Palm-e: An embodied multimodal language model," *arXiv preprint arXiv:2303.03378*, 2023.

[22] B. Yu, H. Kasaei, and M. Cao, "Co-navgpt: Multi-robot cooperative visual semantic navigation using large language models," *arXiv preprint arXiv:2310.07937*, 2023.

[23] Y. Long, X. Li, W. Cai, and H. Dong, "Discuss before moving: Visual language navigation via multi-expert discussions," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 17380–17387, IEEE, 2024.

[24] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22199–22213, 2022.

[25] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *International conference on machine learning*, pp. 9118–9147, PMLR, 2022.

[26] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[27] H. Sha, Y. Mu, Y. Jiang, G. Zhan, L. Chen, C. Xu, P. Luo, S. E. Li, M. Tomizuka, W. Zhan, *et al.*, "Large language models as decision makers for autonomous driving,"

[28] J. Wang, Z. Wu, Y. Li, H. Jiang, P. Shu, E. Shi, H. Hu, C. Ma, Y. Liu, X. Wang, *et al.*, "Large language models for robotics: Opportunities, challenges, and perspectives," *arXiv preprint arXiv:2401.04334*, 2024.

[29] S. Li, Z. Ma, F. Liu, J. Lu, Q. Xiao, K. Sun, L. Cui, X. Yang, P. Liu, and X. Wang, "Safe planner: Empowering safety awareness in large pre-trained models for robot task planning," *arXiv preprint arXiv:2411.06920*, 2024.

[30] Z. Yang, S. S. Raman, A. Shah, and S. 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, IEEE, 2024.

[31] Z. Wang, Q. Liu, J. Qin, and M. Li, "Ensuring safety in llm-driven robotics: A cross-layer sequence supervision mechanism," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9620–9627, IEEE, 2024.

[32] M. Ni, L. Zhang, Z. Chen, and W. Zuo, "Don't let your robot be harmful: Responsible robotic manipulation," *arXiv preprint arXiv:2411.18289*, 2024.

[33] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[34] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[35] B. Lacevic, P. Rocco, and A. M. Zanchettin, "Safety assessment and control of robotic manipulators using danger field," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1257–1270, 2013.

[36] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, and A. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8129–8136, IEEE, 2021.

[37] S.-C. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," in *2015 American Control Conference (ACC)*, pp. 4542–4548, IEEE, 2015.

[38] M. A. Murtaza, S. Aguilera, M. Waqas, and S. Hutchinson, "Safety compliant control for robotic manipulator with task and input constraints," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10659–10664, 2022.

[39] M. A. Murtaza, S. Aguilera, V. Azimi, and S. Hutchinson, "Real-time safety and control of robotic manipulators with torque saturation in operational space," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 702–708, IEEE, 2021.

[40] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[41] L. Wang, A. D. Ames, and M. Egerstedt, "Safe certificate-based maneuvers for teams of quadrotors using differential flatness," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3293–3298, IEEE, 2017.

[42] S. Yin, X. Pang, Y. Ding, M. Chen, Y. Bi, Y. Xiong, W. Huang, Z. Xiang, J. Shao, and S. Chen, "Safeagentbench: A benchmark for safe task planning of embodied llm agents," *arXiv preprint arXiv:2412.13178*, 2024.

[43] W. Tong and T. Zhang, "Codejudge: Evaluating code generation with large language models," *arXiv preprint arXiv:2410.02184*, 2024.

[44] A. Panickssery, S. Bowman, and S. Feng, "Llm evaluators recognize and favor their own generations," *Advances in Neural Information Processing Systems*, vol. 37, pp. 68772–68802, 2025.

[45] W. Xu, G. Zhu, X. Zhao, L. Pan, L. Li, and W. Y. Wang, "Pride and prejudice: Llm amplifies self-bias in self-refinement," *arXiv preprint arXiv:2402.11436*, 2024.