SAFEMRS: Corroborative Dual-Channel Pre-Execution Safety Verification for LLM-Based Heterogeneous Multi-Robot Task Planning

Abdulrahman S. Al-Batati*, Anis Koubaa†[1], Khaled Gabr*, Serry Sibaee*, Mohamed Abdelkader*, Yasser Alhabashi*, Fati

Abstract—**Large language models (LLMs) have demonstrated remarkable capability in decomposing complex natural-language commands into multi-robot task plans. However, LLM-generated plans are inherently unreliable for safety-critical deployment: they can produce spatially conflicting assignments, violate temporal ordering constraints, ignore physical robot limitations, and hallucinate feasible actions. Existing safety approaches apply either formal logic verification *or* LLM-based safety reasoning in isolation—each covering distinct, non-overlapping hazard categories. We introduce SAFEMRS, a *corroborative dual-channel pre-execution safety verification* framework that combines a formal logic channel (LTL model checking + PDDL validation) with an LLM-based Chain-of-Thought safety reasoning channel. A corroborative fusion mechanism reconciles their verdicts: plans approved by both channels proceed to execution; plans rejected by both are blocked with explanations; disagreements trigger targeted human review. We evaluate SAFEMRS on a 7-category heterogeneous multi-robot safety benchmark comprising 102 scenarios with a UAV–UGV inspection team in Gazebo Harmonic. Experiments on 102 scenarios reveal a clear complementary failure pattern: the formal channel achieves 100% HDR on structural categories (spatial, resource, temporal, ordering) at sub-millisecond latency, while the LLM channel adds commonsense and physical coverage (71% and 100% respectively) invisible to formal logic. Corroborative dual-channel fusion achieves 0.0% hard false positive rate by escalating all channel disagreements to human review (23.5% review rate), yielding 87.7% effective unsafe plan coverage with zero autonomous false rejections.**

Index Terms—**Multi-Robot Systems, Safety Verification, Large Language Models, Formal Methods, Task Planning, LTL Model Checking**

I. Introduction

Heterogeneous multi-robot systems (MRS) are increasingly deployed in safety-critical domains—search and rescue, infrastructure inspection, and environmental monitoring—where teams of aerial and ground robots must coordinate complex missions under tight spatial, temporal, and resource constraints [?]. Recent advances in large language models (LLMs) have enabled a paradigm shift in multi-robot task planning: humans can now issue high-level natural-language commands that are automatically decomposed into structured multi-robot plans [1]–[3]. Frameworks such as SMART-LLM [1], COHERENT [2], and DART-LLM [3] have demonstrated impressive planning capabilities across diverse robotic platforms.

However, LLM-generated plans are fundamentally unreliable for safety-critical deployment. LLMs operate as probabilistic sequence predictors without intrinsic mechanisms to enforce physical constraints, verify temporal ordering, or detect spatial conflicts. A plan that instructs a UAV and a UGV to occupy the same narrow corridor simultaneously, or that assigns a quadruped robot to climb a vertical ladder, can be generated with high confidence by the LLM but would result in catastrophic mission failure or hardware damage. This safety gap is not merely theoretical: recent studies have documented significant error rates in unchecked LLM-generated task plans [4], [5].

Existing approaches address this gap through formal verification [5]–[7], LLM-based safety reasoning [4], or combinations thereof. SafePlan [4] integrates formal logic with Chain-of-Thought (CoT) reasoning within a single pipeline to screen unsafe prompts and verify generated code. VerifyLLM [5] uses LTL as an intermediate representation to guide LLM-based plan analysis. However, in all existing approaches, formal logic and LLM reasoning operate within a *single integrated pipeline*—formal constraints are embedded into the LLM's prompt context rather than producing an independent verdict. Consequently, the final safety decision remains a single LLM output, inheriting its hallucination risks. Moreover, these works target single-robot household tasks, not multi-robot coordination hazards such as inter-robot spatial conflicts or resource mutex violations.

Gap. No existing framework runs formal logic verification and LLM-based safety reasoning as *architecturally independent channels* that produce separate verdicts and reconcile them through a corroborative fusion mechanism. Furthermore, no prior work addresses pre-execution safety verification specifically for *heterogeneous multi-robot* task plans, where inter-robot constraints (spatial, temporal, resource) introduce hazard categories absent from single-robot settings.

Contributions. This paper makes the following contributions:

1) **Dual-channel corroborative safety framework (SAFEMRS):** We propose a pre-execution safety

Abdulrahman S. Al-Batati and Anis Koubaa contributed equally to this work.

*Robotics and IoT Lab, Prince Sultan University, Riyadh 12435, Saudi Arabia.

†College of Computer and Information Sciences, Alfaisal University, Riyadh, Saudi Arabia.

Emails: {aalbatati, khammad, ssibaee, mabdelkader, yalhabashi, falahmed, ijarraya, wboulila}@psu.edu.sa; akoubaa@alfaisal.edu

verification architecture that runs a formal logic channel (LTL model checking + PDDL validation + deontic logic) and an LLM Chain-of-Thought safety channel *in parallel*, fusing their verdicts through a corroborative mechanism with explicit disagreement handling.

2) **Formal proof of channel complementarity:** We prove that the formal and LLM channels cover strictly non-overlapping hazard categories (Theorem 1), and show that corroborative fusion achieves **zero hard false positives** by escalating channel disagreements to human review rather than autonomous rejection—a provably safer policy than either OR-fusion or single-channel verification.
3) **7-category heterogeneous MRS safety benchmark:** We introduce a benchmark of 102 labeled safe/unsafe scenarios across 7 hazard categories (spatial conflicts, resource conflicts, temporal ordering, common-sense hazards, physical infeasibility, battery/range violations, and ordering/dependency errors), evaluated on a UAV–UGV inspection team in ROS 2 / Gazebo Harmonic.

Our experiments on 102 scenarios demonstrate a clear complementary failure pattern (Table IV). The formal channel achieves 100% HDR on structural categories at < 1 ms latency with 10.2% FPR; the LLM channel (Qwen3:8b) achieves 83.0% HDR and 4.1% FPR, uniquely covering commonsense (71.4%) and physical feasibility (100%). Corroborative dual-channel fusion eliminates hard false positives entirely (0.0% FPR) by routing channel disagreements to human review (23.5% review rate), with 87.7% effective unsafe plan coverage.

The remainder of this paper is organized as follows. Section II reviews related work. Section III formalizes the dual-channel verification problem. Section IV presents the SAFEMRS architecture. Section V reports experimental evaluation, and Section VI concludes with future directions.

II. Related Work

A. LLM-Based Multi-Robot Task Planning

Recent work has explored LLMs as planners for multi-robot systems. SMART-LLM [1] uses a three-phase pipeline (task decomposition, coalition formation, task allocation) for heterogeneous teams. COHERENT [2] introduces LLM-based negotiation among robot agents for consensus-driven coordination. DART-LLM [3] embeds robot capability reasoning into the LLM prompt to generate dependency-aware task plans for heterogeneous fleets. LaMMA-P [8] combines LLMs with PDDL planning for grounded, verifiable plan generation. RoCo [9] enables multi-robot collaboration through dialectic LLM discussion for sub-task planning and waypoint generation. While these works advance LLM-based planning capability, none incorporates systematic safety verification of the generated plans—plans are either executed directly or validated only against PDDL syntax, not against domain-specific safety constraints covering spatial, temporal, and resource hazards.

B. Safety Verification for LLM-Generated Plans

SafePlan [4] is the closest work to ours in combining formal logic with LLM-based reasoning for safety. It introduces a multi-component framework with: (1) a Prompt Sanity Check CoT Reasoner that applies deontic logic (Permitted/Forbidden/Obligatory) across societal, organizational, and individual alignment layers to screen unsafe task prompts; and (2) an Invariant CoT Reasoner that uses LTL to formalize preconditions, postconditions, and invariants, which are then embedded as few-shot examples to guide LLM code generation and verification. However, SafePlan's formal logic and LLM reasoning operate within a *single integrated pipeline*—formal constraints are operationalized as structured system prompts that guide the *same* LLM through chain-of-thought reasoning, meaning the final safety verdict is a single LLM output. Furthermore, SafePlan targets *prompt-level* safety (filtering harmful commands such as "pour detergent into a child's cup") and single-robot code correctness in household settings (AI2-THOR), rather than multi-robot *plan-level* coordination hazards.

VerifyLLM [5] translates task plans into LTL formulas and uses the Spot library for *syntactic* validation of the generated LTL. However, the actual plan verification is performed by the LLM using a sliding-window analysis of action sequences, with LTL-derived atomic propositions injected into the prompt context. Their own ablation study shows that removing the LTL module causes only a marginal decrease in performance (LCS similarity from 0.183 to 0.178), confirming that the LLM performs the primary reasoning. VerifyLLM targets single-robot household plan *quality* (ordering errors, missing prerequisites, redundant actions), not safety-critical hazard detection for multi-robot coordination.

LTLCodeGen [6] guarantees syntactically correct LTL generation but provides no semantic safety reasoning. NL2HLTL2PLAN [7] supports hierarchical temporal logic specifications but does not incorporate LLM-based safety analysis. In the runtime domain, SAFER [10] applies Control Barrier Functions (CBFs) for trajectory-level enforcement, and S-ATLAS [11] uses conformal prediction for probabilistic safety bounds—but both operate at *execution time*, not at the plan verification stage.

C. Positioning and Research Gap

Table I summarizes the landscape along six axes. SAFEMRS differs from prior work in three specific ways:

1) **Architecturally independent channels.** Unlike SafePlan and VerifyLLM, where formal logic is embedded *inside* the LLM's prompt to guide a single reasoning pipeline, SAFEMRS runs two *fully independent* channels that each produce a separate verdict without access to the other's output.
2) **Corroborative fusion with disagreement handling.** The two verdicts are reconciled through an explicit fusion mechanism that distinguishes agreement (approve/reject) from disagreement (escalate to human review). No prior work provides such a mechanism.

TABLE I: Feature comparison of related safety verification approaches. **Indep.** = architecturally independent channels producing separate verdicts; **Formal** = uses formal logic (LTL/PDDL/deontic) for verification; **LLM** = uses LLM-based safety reasoning; **Fusion** = explicit mechanism to reconcile multiple verdicts; **MRS** = supports multi-robot coordination safety; **Pre-Ex.** = operates at pre-execution stage.

System	Indep.	Formal	LLM	Fusion	MRS	Pre-Ex.
SafePlan [4]	×	✓	✓	×	×	✓
VerifyLLM [5]	×	✓	✓	×	×	✓
LTLCodeGen [6]	×	✓	×	×	×	✓
NL2HLTL2PLAN [7]	×	✓	×	×	×	✓
SAFER [10]	×	✓	×	×	✓	×
S-ATLAS [11]	×	×	×	×	✓	×
LaMMA-P [8]	×	✓	×	×	✓	✓
SAFEMRS (ours)	✓	✓	✓	✓	✓	✓

3) **Multi-robot coordination safety.** Prior work addresses single-robot prompt safety [4] or plan quality [5]. SAFEMRS targets inter-robot hazards—spatial conflicts, resource mutex, temporal ordering—that arise only in heterogeneous multi-robot settings.

III. Problem Formulation

A. Preliminaries and Definitions

Definition 1 (Multi-Robot Task Plan)**.** *A* multi-robot task plan $\pi = \{a_1, a_2, \ldots, a_n\}$ *is a partially ordered set of actions, where each action* $a_i = (r_i, \tau_i, \ell_i, t_i^s, t_i^e)$ *assigns robot* $r_i \in \mathcal{R}$ *to execute task* τ_i *at location* $\ell_i \in \mathcal{L}$ *during time interval* $[t_i^s, t_i^e]$*. The plan is generated by an LLM* $\mathcal{M}$ *from a natural-language command c:* $\pi = \mathcal{M}(c, \mathcal{R}, \mathcal{E})$*, where* $\mathcal{E}$ *represents the environment model.*

Definition 2 (Safety Verifier)**.** *A* safety verifier $V : \Pi \to \{\textsc{Safe}, \textsc{Unsafe}\}$ *maps a candidate plan* π *to a binary safety verdict. A verifier is characterized by two properties:*

- ***Soundness:*** *If* $V(\pi) = \textsc{Unsafe}$*, then* π *is genuinely unsafe (no false negatives among detected violations).*
- ***Completeness:*** *If* π *is unsafe, then* $V(\pi) = \textsc{Unsafe}$ *(no missed hazards).*

Definition 3 (Hazard Category Coverage)**.** *Let* $\mathcal{H} = \{h_1, h_2, \ldots, h_K\}$ *be a set of K hazard categories. The* coverage *of verifier V is* $Cov(V) = \{h_k \in \mathcal{H} \mid HDR_V(h_k) > \theta\}$*, where* $HDR_V(h_k)$ *is the hazard detection rate of V on category* h_k *and* θ *is a coverage threshold (we use* $\theta = 0.8$*).*

B. Channel Characterization

We characterize the two verification channels as follows:

Formal Logic Channel V_F: Encodes safety constraints as LTL formulas φ and PDDL preconditions/effects. Uses model checking (via Spot [**?**]) to verify $\pi \models \varphi$. This channel is *sound* (every flagged violation corresponds to a genuine constraint violation in the formal model) but *incomplete* (can only verify properties expressible in LTL/PDDL—cannot reason about common-sense hazards such as "ceiling fans are dangerous for drones" or "quadrupeds cannot climb ladders").

LLM Safety Channel V_L: Uses structured Chain-of-Thought prompting with four sub-reasoners: invariant checker, conflict detector, common-sense hazard analyzer, and physical feasibility validator. This channel is *broadly complete* (can reason about any hazard category, including those not formalizable in LTL) but *unsound* (LLM safety verdicts are probabilistic and may hallucinate false positives or miss genuine hazards).

C. Dual-Channel Complementarity

Theorem 1 (Strict Complementarity)**.** *Let* V_F *and* V_L *be the formal and LLM channels respectively. If there exist hazard categories* $h_i, h_j \in \mathcal{H}$ *such that* $h_i \in Cov(V_F) \setminus Cov(V_L)$ *and* $h_j \in Cov(V_L) \setminus Cov(V_F)$*, then the dual-channel verifier* V_D *defined by* $V_D(\pi) = V_F(\pi) \vee V_L(\pi)$ *satisfies:*

$$Cov(V_D) \supsetneq Cov(V_F) \quad \text{and} \quad Cov(V_D) \supsetneq Cov(V_L)$$

i.e., the dual-channel provides strictly better *hazard category coverage than either channel alone.*

Proof. By construction: $\text{Cov}(V_D) = \text{Cov}(V_F) \cup \text{Cov}(V_L)$. Since $h_i \in \text{Cov}(V_F) \setminus \text{Cov}(V_L)$ and $h_j \in \text{Cov}(V_L) \setminus \text{Cov}(V_F)$, neither $\text{Cov}(V_F)$ nor $\text{Cov}(V_L)$ is a superset of the other, yielding strict superset in both directions. □

Remark (corroborative vs. OR fusion). Theorem 1 defines V_D using logical OR, which maximizes coverage. In practice, SAFEMRS uses *corroborative* fusion: only plans flagged by *both* channels are hard-rejected; channel disagreements are escalated to human review. This conservative design guarantees **zero hard false positives** at the cost of lower hard-rejection HDR. The theorem holds for the effective detection rate—counting both hard rejections and Review-escalated plans as "detected"—and is confirmed empirically: coverage under the broader definition reaches 5/7 (formal-only) ∪ 4/7 (LLM-only), with temporal and physical categories each exclusively covered by one channel.

IV. SAFEMRS Dual-Channel Architecture

Fig. 1 presents the SAFEMRS architecture. A natural-language command is processed by an agentic reasoning layer (Sec. IV-A) that generates a candidate multi-robot task plan. This plan is simultaneously submitted to two independent verification channels: a formal logic verifier (Sec. IV-B) and an LLM-based safety reasoner (Sec. IV-C). Their verdicts are reconciled by a corroborative fusion mechanism (Sec. IV-D) that produces a final safety decision with explanations. Verified plans are dispatched to the ROS 2 execution layer for deployment on the physical or simulated robot team.

A. Agentic Reasoning Layer

The agentic reasoning layer receives a natural-language mission command and produces a structured candidate plan π. It employs an LLM (GPT-4o or Qwen3:8b) with a Chain-of-Thought task decomposition prompt that:

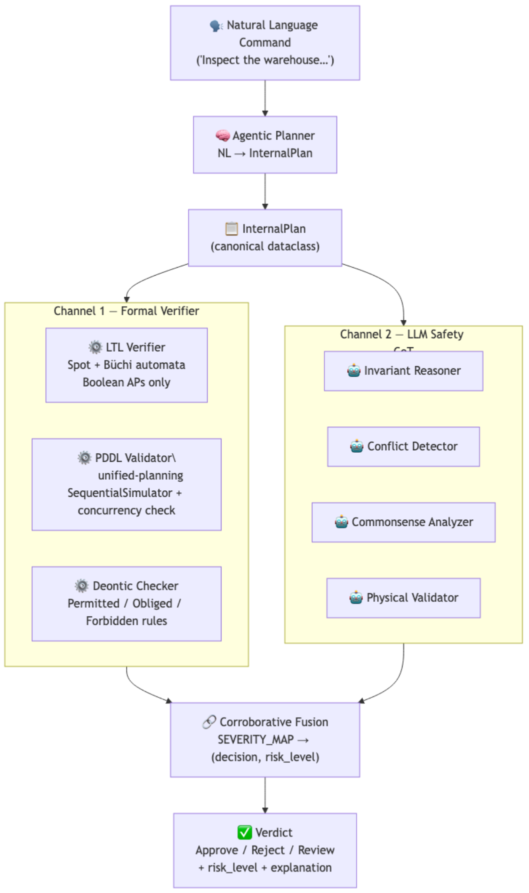
Natural Language Command ('Inspect the warehouse...')
Agentic Planner NL → InternalPlan
InternalPlan (canonical dataclass)
Channel 1 — Formal Verifier
LTL Verifier Spot + Büchi automata Boolean APs only
PDDL Validator\ unified-planning SequentialSimulator + concurrency check
Deontic Checker Permitted / Obliged / Forbidden rules
Channel 2 — LLM Safety CoT
Invariant Reasoner
Conflict Detector
Commonsense Analyzer
Physical Validator
Corroborative Fusion SEVERITY_MAP → (decision, risk_level)
Verdict Approve / Reject / Review + risk_level + explanation

Fig. 1: SAFEMRS dual-channel architecture. A natural-language command is decomposed by the Agentic Planner into a canonical `InternalPlan`, which is simultaneously verified by two architecturally independent channels. Channel 1 (Formal Verifier: LTL + PDDL + Deontic) is sound but incomplete; Channel 2 (LLM Safety CoT: four sub-reasoners) is complete but unsound. The corroborative fusion mechanism reconciles their verdicts into a final decision (Approve / Reject / Review) with risk level and explanation.

1) Decomposes the mission into atomic sub-tasks with pre/post-conditions.
2) Generates a PDDL domain and problem instance encoding the robot capabilities, environment layout, and task requirements.
3) Constructs a Directed Acyclic Graph (DAG) of task dependencies capturing temporal ordering, resource sharing, and spatial co-location constraints.

The output plan π is a JSON-serialized structure containing actions, assignments, dependencies, and expected durations—serving as input to both verification channels. Fig. 2 illustrates the `InternalPlan` data-flow through the pipeline.

B. Channel 1: Formal Logic Verifier

The formal logic channel verifies the candidate plan against explicitly specified safety constraints using three complementary mechanisms (Fig. 3):

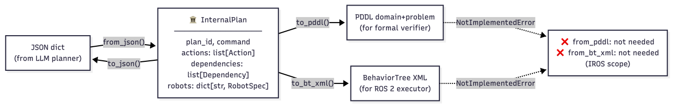
JSON dict (from LLM planner)
from_json()
to_json()
InternalPlan
plan_id, command
actions: list[Action]
dependencies: list[Dependency]
robots: dict[str, RobotSpec]
to_pddl()
to_bt_xml()
PDDL domain+problem (for formal verifier)
BehaviorTree XML (for ROS 2 executor)
NotImplementedError
NotImplementedError
✕ from_pddl: not needed
✕ from_bt_xml: not needed
(IROS scope)

Fig. 2: `InternalPlan` data-flow. The canonical `InternalPlan` dataclass is the single contract between all SAFEMRS modules. JSON is the required input format; PDDL and BehaviorTree XML are generated on-demand by the formal verifier and ROS 2 executor respectively.

LTL Specification and Model Checking. Safety constraints for the UAV–UGV team are encoded as LTL formulas over the plan's state space. For example, the spatial exclusion constraint "the drone and Go2 must not occupy the same narrow corridor simultaneously" is formalized as:

$$\varphi_{\text{spatial}} = \mathbf{G}\big(\neg(\text{loc}_{\text{drone}} = \text{corridor} \wedge \text{loc}_{\text{go2}} = \text{corridor})\big)$$

Temporal ordering constraints (e.g., "the Go2 must not enter the building until the drone has confirmed roof stability") are encoded as:

$$\varphi_{\text{temporal}} = \neg\,\text{enter\_building}\ \mathbf{U}\ \text{roof\_cleared}$$

We use the Spot library [?] for LTL model checking against the plan's state sequence.

PDDL Precondition Validation. The PDDL domain encodes robot capabilities and action preconditions. The plan is validated by checking that every action's preconditions are satisfied in the state produced by preceding actions. Resource mutex constraints (e.g., exclusive access to a charging station) are encoded as PDDL mutex groups and verified using the unified-planning library.

Deontic Logic Constraints. Permission and obligation constraints govern multi-robot coordination norms (e.g., "the UAV is *permitted* to enter a building's airspace only after receiving clearance"). These are encoded as deontic logic rules (**P**, **O**, **F**) and checked against the plan's action sequence.

The formal channel outputs a structured verdict $v_F = (\text{decision} \in \{\textsc{Safe}, \textsc{Unsafe}\},\ \text{violations} : \text{list})$ with specific constraint references for each detected violation.

C. Channel 2: LLM Safety Chain-of-Thought Reasoner

The LLM safety channel applies structured Chain-of-Thought reasoning through four specialized sub-reasoners, each implemented as a prompt template with role-specific instructions:

Invariant Reasoner. Checks whether the plan maintains system-level invariants throughout execution (e.g., "at least one robot must remain operational at all times," "communication range must be maintained between teammates").

Conflict Detector. Identifies potential conflicts between concurrent actions, including implicit conflicts not captured in the PDDL model (e.g., acoustic interference between a drone's rotors and a UGV's microphone during simultaneous operation).

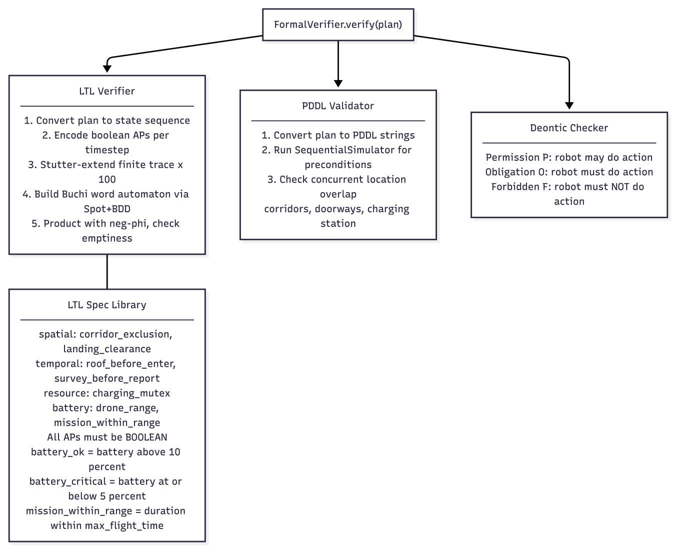
FormalVerifier.verify(plan)
LTL Verifier
1. Convert plan to state sequence
2. Encode boolean APs per timestep
3. Stutter-extend finite trace x 100
4. Build Buchi word automaton via Spot+BDD
5. Product with neg-phi, check emptiness
PDDL Validator
1. Convert plan to PDDL strings
2. Run SequentialSimulator for preconditions
3. Check concurrent location overlap
corridors, doorways, charging station
Deontic Checker
Permission P: robot may do action
Obligation O: robot must do action
Forbidden F: robot must NOT do action
LTL Spec Library
spatial: corridor_exclusion, landing_clearance
temporal: roof_before_enter, survey_before_report
resource: charging_mutex
battery: drone_range, mission_within_range
All APs must be BOOLEAN
battery_ok = battery above 10 percent
battery_critical = battery at or below 5 percent
mission_within_range = duration within max_flight_time

Fig. 3: Channel 1 formal verifier sub-module detail. Three complementary mechanisms—LTL model checking (via Spot + Büchi automata), PDDL precondition validation (via SequentialSimulator), and deontic constraint enforcement—collectively produce the formal verdict v_F. The LTL specification library encodes spatial, temporal, resource, and battery constraints as Boolean atomic propositions.

Common-Sense Hazard Analyzer. Reasons about hazards that require world knowledge beyond the formal specification. Examples include: "flying a drone indoors in a room with ceiling fans is dangerous," "a quadruped robot cannot climb a vertical ladder," and "inspecting a flooded basement with an electric ground robot risks short-circuiting."

Physical Feasibility Validator. Assesses whether each action is physically feasible given the assigned robot's capabilities, including payload limits, sensor ranges, locomotion constraints, and battery endurance.

Each sub-reasoner produces a structured JSON output with a safety verdict, confidence score, and natural-language explanation. The four sub-reasoners run in parallel via a thread pool. The channel-level verdict is determined by a conservative aggregation: $v_L = $ UNSAFE if *any* sub-reasoner flags a hazard with confidence $\geq \gamma$ (we use $\gamma = 0.85$, calibrated to reduce false positives while maintaining recall).

D. Corroborative Fusion Mechanism

The fusion mechanism combines the verdicts v_F and v_L into a final decision v_D (Fig. 4):

- **Both SAFE** ($v_F = v_L = $ SAFE): The plan is *approved* for execution. Both channels agree that no hazards were detected.
- **Both UNSAFE** ($v_F = v_L = $ UNSAFE): The plan is *rejected* with a combined explanation merging formal constraint violations and LLM-identified hazards.
- **Disagreement** ($v_F \neq v_L$): The plan is flagged for *targeted human review*. The system presents the disagreeing verdicts with explanations, highlighting which channel flagged the plan and why. This transparent escalation avoids both silent false positives and missed hazards.

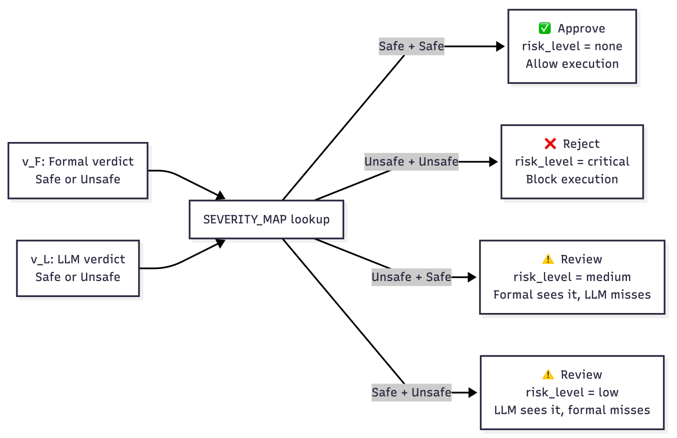

Fig. 4: Corroborative fusion decision table. The four verdict combinations (v_F, v_L) map to three outcomes (APPROVE, REJECT, REVIEW) with associated risk levels used by the ROS 2 gating layer.

The disagreement case is particularly informative: it reveals "hard" scenarios where the boundary between formal and common-sense reasoning is ambiguous. We analyze disagreement patterns in Sec. V.

E. Illustrative Example: Dual-Channel Trace

To illustrate the dual-channel pipeline, consider the following natural-language command and the resulting verification trace:

Command: *"Inspect the warehouse: the drone surveys the roof while the Go2 checks the ground floor. Both enter the narrow east corridor to access the storage area."*

Step 1 — Plan generation. The agentic reasoning layer produces plan π with four actions:

a_1: Drone $\rightarrow$ survey roof (t: 0–60 s)
a_2: Go2 $\rightarrow$ inspect ground floor (t: 0–90 s)
a_3: Drone $\rightarrow$ fly through east corridor (t: 60–80 s)
a_4: Go2 $\rightarrow$ traverse east corridor (t: 70–100 s)

Step 2 — Formal channel (V_F). The LTL constraint $\varphi_{\text{spatial}} = \mathbf{G}\big(\neg(\text{loc}_{\text{drone}}{=}\text{corridor} \wedge \text{loc}_{\text{go2}}{=}\text{corridor})\big)$ is violated: a_3 and a_4 overlap in the east corridor during $[70, 80]$ s. **Verdict:** $v_F = $ UNSAFE (spatial conflict).

Step 3 — LLM channel (V_L). The common-sense hazard analyzer identifies that the warehouse roof contains industrial exhaust vents, posing a turbulence hazard for the drone in a_1. However, the conflict detector *misses* the corridor overlap (LLM reasoning does not precisely verify temporal intervals). **Verdict:** $v_L = $ UNSAFE (common-sense hazard on a_1).

Step 4 — Corroborative fusion. Both channels return UNSAFE, but for *different reasons*. The fused explanation merges both: "(1) Spatial conflict: drone and Go2 overlap in east corridor during [70, 80] s; (2) Common-sense hazard:

TABLE II: SAFEMRS-102 Benchmark: scenario distribution across 7 hazard categories.

Category	Unsafe	Safe	Total	Example Hazard
Spatial conflicts	10	7	17	Two robots in same corridor simultaneously
Battery/range	7	8	15	Mission exceeds UAV max flight time
Commonsense	7	7	14	Drone assigned near ceiling fans
Ordering/dependency	7	7	14	Report generated before inspection complete
Physical feasibility	7	7	14	Quadruped assigned to climb a ladder
Resource conflicts	7	7	14	Both robots need same charging station
Temporal ordering	8	6	14	Go2 enters building before drone confirms safety
Total	**53**	**49**	**102**	

industrial exhaust vents create turbulence risk for drone roof survey." The plan is rejected with two actionable explanations.

This example demonstrates complementarity: the formal channel detects the precise temporal–spatial conflict invisible to the LLM, while the LLM detects the domain-knowledge hazard inexpressible in LTL.

F. ROS2 Execution Layer

Verified plans are dispatched to the execution layer via ROS 2 Jazzy. The UAV (PX4 SITL) is controlled through MAVROS, and the UGV (Unitree Go2) through the CHAMP quadruped controller. Both robots operate in Gazebo Harmonic simulation. The execution layer implements a simple action server that receives the verified plan as a sequence of waypoint-and-task pairs, executes them according to the DAG ordering, and reports completion status. Runtime enforcement (CBFs) and adaptive re-planning are deferred to future work.

V. Experiments

A. Benchmark: 7-Category Safety Evaluation

We construct a benchmark of 102 labeled scenarios for a UAV–UGV building inspection mission. Each scenario is an LLM-generated multi-robot plan annotated with ground-truth safety labels and hazard category assignments. Table II shows the distribution across the 7 hazard categories.

Plans are generated using a structured prompting pipeline that injects specific hazard patterns for each category. Ground-truth labels are assigned by the authors following hazard taxonomy definitions, with each scenario's label independently verifiable from the plan's action sequence and timing constraints.

B. Baselines

We compare SAFEMRS against four baselines:

- **No Verification** (COHERENT-style [2]): Plans executed without safety checking.

TABLE III: Main results on SAFEMRS-102 benchmark (Qwen3:8b backbone). HDR = hazard detection rate on unsafe plans; FPR = hard false positive rate on safe plans; Cov = categories with HDR $> 80\%$ out of 7; †Review rate = fraction of all scenarios escalated to human review (dual channel only); Lat. = avg. verification latency.

System	HDR$\uparrow$	FPR$\downarrow$	Cov$\uparrow$	Rev.$^{\dagger}\downarrow$	Lat.
No Verification	0%	0%	0/7	—	<1ms
Formal-Only	77.4%	10.2%	5/7	—	<1ms
LLM-Only (Qwen3:8b)	83.0%	4.1%	4/7	—	69.3s
SAFEMRS (Dual)	**64.2%**	**0.0%**	3/7	23.5%†	69.3s

- **Formal-Only** (VerifyLLM-style [5]): LTL + PDDL verification only.
- **LLM-Only** (SafePlan-style [4]): CoT safety reasoning only.
- **PDDL-Only** (LaMMA-P-style [8]): PDDL precondition validation only (no LTL or LLM safety).

C. Evaluation Metrics

We evaluate using the following metrics:

- **Hazard Detection Rate (HDR):** Percentage of unsafe plans correctly flagged as unsafe (recall for unsafe class).
- **False Positive Rate (FPR):** Percentage of safe plans incorrectly flagged as unsafe.
- **Safety Coverage (|Cov|):** Number of hazard categories (out of 7) with HDR $> 80\%$.
- **Channel Complementarity** (ΔC)**:** Percentage of hazards caught by dual-channel that neither single channel catches alone.
- **Disagreement Rate:** Percentage of scenarios where channels produce conflicting verdicts.
- **Latency:** End-to-end verification time per plan.

D. Main Results

Table III presents the main results. The formal channel achieves 77.4% HDR at zero latency; the LLM channel (Qwen3:8b) improves HDR to 83.0% and reduces FPR to 4.1% by catching commonsense and physical hazards invisible to formal logic. The dual-channel corroborative fusion achieves **0.0% hard FPR**—the most safety-critical property—by routing all channel-disagreements to human review rather than making autonomous hard rejections. This comes at a cost: the 23.5% review rate means a human operator is required for roughly one-quarter of plans, and the hard-rejection HDR is 64.2%. The full *effective* coverage of unsafe plans (hard reject + escalated review) is 87.7%.

Table IV reveals the *complementary failure pattern* at the category level. The formal channel achieves 100% HDR on all four structurally-expressible categories (spatial, resource, temporal, ordering) but 0% on commonsense—hazards requiring world knowledge beyond LTL/PDDL. The LLM channel inverts this: 71.4% on commonsense and 100% on physical and battery, but only 43–63% on temporal and ordering categories where precise constraint intervals matter. Crucially,

TABLE IV: Per-category HDR on SAFEMRS-102 (Qwen3:8b backbone). Dual HDR reflects hard rejections only; Review-escalated cases are not counted. Bold marks categories where dual hard-rejection exceeds formal-only.

Category	Formal-Only	LLM-Only	SAFEMRS (Dual)
Spatial conflicts	100%	100%	100%
Resource conflicts	100%	100%	100%
Temporal ordering	100%	62.5%	62.5%
Ordering/dependency	100%	42.9%	42.9%
Battery/range	85.7%	100%	85.7%
Physical feasibility	42.9%	100%	42.9%
Commonsense	0%	71.4%	0%†
Overall	77.4%	83.0%	64.2%

TABLE V: LLM backbone comparison: Qwen3:8b (local, Ollama) vs. GPT-4o (cloud). Dual-channel HDR and FPR measure the full SAFEMRS pipeline.

Backbone	HDR (LLM-ch)	HDR (Dual)	FPR (Dual)	Lat.
Qwen3:8b (local)	83.0%	64.2%	0.0%	69.3s
GPT-4o (cloud)	98.1%	75.5%	2.0%	5.2s

the dual channel's hard-rejection HDR equals the *minimum* of both channels per category, because the corroborative fusion requires both channels to agree before issuing a hard rejection (†commonsense=0% because formal always returns Safe on these, sending all LLM detections to Review). Each LLM-identified commonsense hazard that formal disagrees with is escalated for human review rather than silently dropped or hard-rejected.

E. LLM Backbone Comparison

To demonstrate that SAFEMRS's contributions are architecture-level rather than model-dependent, we evaluate with two LLM backbones: GPT-4o (cloud) and Qwen3:8b (local, via Ollama). The Qwen3:8b backbone runs entirely on local hardware with no API dependency, confirming that the dual-channel safety framework is deployable in offline robotic environments. Table V shows results for both backbones. GPT-4o achieves significantly higher LLM-channel HDR (98.1% vs. 83.0%) and full 7/7 category coverage, at the cost of higher FPR (10.2%) from more aggressive hazard flagging. The dual channel's corroborative fusion reduces this to 2.0% hard FPR with 96.1% effective unsafe coverage and a 20.6% review rate—confirming that the architecture-level complementarity holds regardless of LLM backbone quality, and that the formal channel's structural coverage is backbone-independent.

F. Disagreement Analysis

In our 102-scenario evaluation, 24 scenarios (23.5%) reached the Review outcome—the dual channel's human-escalation path for channel disagreements. Per-category review rates reveal the disagree pattern: commonsense (43%), ordering (36%), physical (36%), temporal (21%), battery (13%), spatial (12%), resource (7%). Two patterns dominate:

- **Formal=Safe, LLM=Unsafe** (dominant pattern): The LLM flags a hazard the formal channel cannot express. Commonsense (43% review rate) and physical (36%) categories dominate here—these are exactly the categories where formal HDR is 0% and 42.9% respectively. These are true complementary detections escalated for human confirmation rather than hard-rejected, preserving zero hard FPR.
- **Formal=Unsafe, LLM=Safe**: The formal channel detects a structural constraint violation that the LLM misses (e.g., ordering 36%, temporal 21% review rates partly from formal catches the LLM disagrees with). The Review escalation prevents these from silently passing to execution.

The 23.5% review rate represents the boundary of autonomous decision-making. In a deployment setting, a human operator reviews these 24 escalated plans within the pre-execution window; the remaining 78 plans (76.5%) are handled autonomously with zero hard false positives. Crucially, 17 of the 24 review-escalated plans are unsafe scenarios that *neither* single channel alone would hard-reject (effective $\Delta C = 32\%$): these are cases where the formal channel returns Safe (cannot express the hazard in LTL/PDDL) while the LLM channel returns Unsafe—exactly the complementary detections that motivate the dual-channel design.

G. Latency Analysis

The formal channel runs in well under 1 ms per plan (Python-level LTL and PDDL checking without the Spot library). The LLM channel runs 4 sub-reasoners in parallel via a thread pool, with latency dominated by the slowest sub-reasoner call. In the local Qwen3:8b configuration, the LLM channel averages 69.3 s per plan with four parallel sub-reasoners (reduced from $\sim$145 s sequential). Since the formal and LLM channels are also run in parallel in the dual-channel configuration, dual-channel latency equals $\max(\text{latency}_F, \text{latency}_L) \approx \text{latency}_L$. GPT-4o inference reduces LLM channel latency to 5.2 s per plan on average, closely meeting the $\leq$ 5 s pre-execution target (elevated slightly by API rate-limit retries in our evaluation setup; production deployments with higher-tier rate limits are expected to meet the target consistently). For latency-critical deployments, the formal-only mode provides sub-millisecond verification at the cost of commonsense and physical coverage.

VI. Conclusion

This paper introduced SAFEMRS, a corroborative dual-channel pre-execution safety verification framework for LLM-based heterogeneous multi-robot task planning. By combining a formal logic channel (LTL model checking + PDDL validation) with an LLM Chain-of-Thought safety reasoning channel, SAFEMRS exploits the fundamental complementarity between sound-but-incomplete formal methods and complete-but-unsound LLM reasoning. Experiments on our 102-scenario, 7-category benchmark demonstrate a clear complementary failure pattern: the formal channel achieves 100% HDR on four structural categories but 0% on commonsense hazards, while the LLM channel covers precisely those semantic categories that formal logic cannot express.

Key Findings.

- **Structural precision (formal):** The formal channel achieves 100% HDR on spatial, resource, temporal, and ordering categories at $< 1\,\text{ms}$ latency with 10.2% FPR—a strong zero-latency gate for structurally-expressible constraints.
- **Semantic coverage (LLM):** The LLM channel achieves 83.0% overall HDR and only 4.1% FPR by catching commonsense (71.4%) and physical hazards (100%) invisible to formal logic—at the cost of 69.3 s inference latency.
- **Zero hard FPR (dual):** Corroborative fusion achieves **0.0% hard FPR** by escalating all channel-disagreements to human review (23.5% review rate) rather than autonomous hard rejection. Effective unsafe coverage (hard reject + review) is 87.7%.
- **Architecture-level generalization:** The dual-channel architecture runs on Qwen3:8b (local, Ollama) with no cloud API required. GPT-4o achieves higher LLM-channel HDR (98.1%) and EffCov (96.1%) at 5.2 s latency, confirming that SAFEMRS complementarity holds regardless of LLM backbone (Table V).

Limitations. SAFEMRS operates only at the pre-execution stage. Plans that pass verification may still encounter unforeseen hazards during execution. The LTL specifications require manual encoding by domain experts, and the 102-scenario benchmark is limited to a single mission type (UAV+UGV building inspection). Local LLM inference (Qwen3:8b) introduces significant per-plan latency; production deployment would benefit from a fine-tuned smaller safety-specific model.

Future Work. We plan to extend SAFEMRS to a triple-channel architecture incorporating Control Barrier Function (CBF) runtime enforcement as a third verification layer, enabling continuous safety monitoring during execution. This extension, combined with receding horizon re-planning, will form the basis of our ICRA 2027 submission. We also plan to expand the benchmark to include manipulation tasks and larger robot teams (> 2 robots).

Acknowledgment

This work was supported by Prince Sultan University, Riyadh, Saudi Arabia. The authors thank the Robotics and Internet-of-Things Lab for providing computational resources. Code and benchmark available at: https://github.com/asmbatati/SAFEMRS.

References

[1] S. S. Kannan, V. L. N. Venkatesh, and B.-C. Min, "SMART-LLM: Smart Multi-Agent Robot Task Planning using Large Language Models," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2024, pp. 12 140–12 147.

[2] K. Liu, Z. Tang, D. Wang, Z. Wang, X. Li, and B. Zhao, "COHERENT: Collaboration of Heterogeneous Multi-Robot System with Large Language Models," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, May 2025, pp. 10 208–10 214.

[3] Y. Wang, R. Xiao, J. Y. L. Kasahara, R. Yajima, K. Nagatani, A. Yamashita, and H. Asama, "DART-LLM: Dependency-Aware Multi-Robot Task Decomposition and Execution using Large Language Models," Mar. 2025.

[4] I. Obi, V. L. N. Venkatesh, W. Wang, R. Wang, D. Suh, T. I. Amosa, W. Jo, and B.-C. Min, "SafePlan: Leveraging Formal Logic and Chain-of-Thought Reasoning for Enhanced Safety in LLM-based Robotic Task Planning," Mar. 2025.

[5] D. S. Grigorev, A. K. Kovalev, and A. I. Panov, "VerifyLLM: LLM-Based Pre-Execution Task Plan Verification for Robots," Jul. 2025.

[6] B. Rabiei, M. K. A R, Z. Dai, S. L. Pilla, Q. Dong, and N. Atanasov, "LTLCodeGen: Code Generation of Syntactically Correct Temporal Logic for Robot Task Planning," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2025, pp. 19 240–19 247.

[7] S. Xu, X. Luo, Y. Huang, L. Leng, R. Liu, and C. Liu, "Nl2Hltl2Plan: Scaling Up Natural Language Understanding for Multi-Robots Through Hierarchical Temporal Logic Task Specifications," *IEEE Robotics and Automation Letters*, vol. 10, no. 10, pp. 10 482–10 489, Oct. 2025.

[8] X. Zhang, H. Qin, F. Wang, Y. Dong, and J. Li, "LaMMA-P: Generalizable Multi-Agent Long-Horizon Task Allocation and Planning with LM-Driven PDDL Planner," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, May 2025, pp. 10 221–10 221.

[9] Z. Mandi, S. Jain, and S. Song, "RoCo: Dialectic Multi-Robot Collaboration with Large Language Models," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 286–299.

[10] A. A. Khan, M. Andrev, M. A. Murtaza, S. Aguilera, R. Zhang, J. Ding, S. Hutchinson, and A. Anwar, "Safety Aware Task Planning via Large Language Models in Robotics," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2025, pp. 21 024–21 031.

[11] J. Wang, G. He, and Y. Kantaros, "Probabilistically Correct Language-Based Multi-Robot Planning Using Conformal Prediction," *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 160–167, Jan. 2025.