DisasterSim: A Reproducible Benchmark for Navigation and Coverage in Collapsed Structures with Empirical Analysis of Classical Exploration and Goal-Conditioned Learning-Based Methods

Newton Adhikari
Department Of Automobile And Mechanical Engineering
Thapathali Campus, Institute of Engineering
Tribhuvan University, Kathmandu, Nepal

***Abstract*—Autonomous navigation of collapsed buildings is critical for disaster response, yet no standardized simulation benchmark exists for reproducible evaluation of robot navigation and coverage policies in such environments. We present DisasterSim, an open-source benchmark built on ROS 2 Humble and Gazebo Classic that provides a physically realistic post-earthquake building interior with configurable obstacle density, a multi-modal sensor suite with Extended Kalman Filter (EKF)-based fusion, four formally defined evaluation metrics with automated computation, and four reference baseline policies. The entire system—environment, robot, SLAM, navigation stack, metrics, and automated experiment runner—executes from a single command with frozen parameters to ensure full reproducibility.**

Our empirical study across 39 trials reveals a striking result: three fundamentally different classical exploration paradigms—reactive FSM, frontier-based, and potential field—converge to a statistically indistinguishable performance plateau of approximately 30% area coverage ($p>0.79$, $|d|\leq 0.27$). This convergence suggests that *navigation constraints*, not exploration strategy, form the primary performance bottleneck in cluttered disaster environments. A partially trained *goal-conditioned* PPO policy (370k of 600k planned steps)—which navigates toward a fixed known survivor location rather than exploring freely—achieves higher incidental coverage (36.9% mean, 61.1% peak, Cohen's $d=0.78$), indicating that goal-directed learned navigation traverses more of the environment en route than classical explorers manage in the same time budget. We additionally identify a quantifiable *coverage–localization trade-off* (Pearson $r=0.85$, $p<0.001$), correct a data error present in an earlier draft, and discuss the design of a goal-free RL explorer as the next step toward a fully autonomous learned baseline. All code, configurations, experiment logs, and trained models are publicly available.

***Index Terms*—disaster robotics, robot navigation, area coverage, benchmark, simulation, ROS 2, reinforcement learning, goal-conditioned navigation, SLAM, frontier exploration, reproducibility**

I. Introduction

Autonomous robots deployed in post-disaster environments must navigate collapsed structures, rubble fields, and degraded sensor conditions to locate survivors [1]. The 2023 Turkey–Syria earthquakes, which killed over 50,000 people and trapped thousands in rubble for days, and the 2021 Surfside condominium collapse demonstrated concretely that human first responders face life-threatening risks in structurally compromised buildings [2]. Deploying robots to navigate and map such environments before human entry could save lives, yet development is hampered by the absence of a standardized, reproducible benchmark for comparing navigation and coverage policies under controlled conditions.

While competitions such as the DARPA Subterranean Challenge [3] and RoboCup Rescue [4] have advanced the field, they rely on physical testbeds that are expensive, non-reproducible, and inaccessible to most research groups. Simulation-based evaluation offers reproducibility and accessibility, yet existing simulators either lack standardized metrics, ship without reference baselines, or require extensive bespoke configuration before a researcher can evaluate a new policy. This gap means that published results across groups are rarely directly comparable, impeding scientific progress.

We address this gap with **DisasterSim**, an open-source benchmark that bundles:

- A Gazebo Classic-based collapsed building environment with four obstacle types at three configurable density levels (easy/medium/hard);
- A TurtleBot3 Waffle-class differential-drive robot with 360° LiDAR, RGB camera, and IMU with calibrated sensor noise models;
- EKF-based sensor fusion, asynchronous SLAM, and a full Nav2 navigation stack with frozen, documented parameters ensuring that results reflect policy quality, not tuning;
- Four formally defined evaluation metrics (area coverage, localization RMSE, near-collision rate, efficiency) with automated computation from raw per-trial logs;
- Four reference baselines: three classical exploration policies (reactive FSM, frontier-based, potential field) and one goal-conditioned PPO policy that navigates toward a fixed known location, serving as a learned navigation reference;
- A one-command experiment runner producing aggregate statistics with Kruskal-Wallis and Mann-Whitney U tests and Cohen's d effect sizes.

Contributions. This work provides: (1) a fully reproducible

benchmark for disaster robot navigation and coverage evaluation with frozen configuration and automated end-to-end evaluation; (2) a standardized four-metric evaluation protocol with rigorous statistical analysis; and (3) an empirical study that reveals a classical exploration ceiling in cluttered disaster environments, quantifies a fundamental coverage–localization trade-off, identifies the goal-conditioned nature of the current RL baseline as an open limitation, and corrects a data error present in earlier circulations of this manuscript.

Research question. *Do classical exploration strategies fundamentally saturate in cluttered disaster environments? What coverage advantage does a goal-conditioned learned navigator achieve over classical explorers, and what does this reveal about navigation constraints in disaster environments?*

II. Related Work

A. Disaster Robotics Simulators

The DARPA SubT Virtual Track [3] provided cave, tunnel, and urban environments with high-fidelity physics but required significant computational resources and custom integration with proprietary scoring systems, limiting accessibility. The RoboCup Rescue Simulation League [4] focuses on multi-agent coordination rather than single-robot exploration metrics. USARSim [5] pioneered disaster simulation on Unreal Engine but predates modern ROS 2 tooling and lacks standardized metric computation. More recently, FUEL [7] proposed an exploration planner with a custom benchmark, but it targets open 3D spaces (UAV exploration) rather than cluttered indoor disaster scenarios. Habitat [6] provides a strong embodied-AI platform with reference baselines and frozen configs, but is not designed for disaster-specific physics or differential-drive ground robots. None of these platforms provide the complete package required for disaster benchmarking: disaster environment, reference baselines, standardized metrics, and automated statistical evaluation in a single modern ROS 2 package.

B. Exploration Algorithms

Frontier-based exploration [8] remains the dominant classical paradigm, directing a robot toward the boundary between mapped free space and unknown space. Information-theoretic approaches [9] optimize sensor information gain but require expensive per-step computation unsuitable for time-constrained disaster response. Potential field methods [11] provide reactive obstacle avoidance but suffer from local minima in cluttered environments; extensions with wall-following [12] partially address this. A key open question is whether these fundamentally different strategies converge in performance when deployed in highly cluttered environments—a question our empirical study directly addresses.

Learning-based navigation has gained traction, with PPO [13] and curriculum learning [14] enabling end-to-end policies. A key distinction in this literature is between *goal-conditioned navigation*—where the agent navigates toward a known target location—and *autonomous exploration*—where the agent must discover the environment without a fixed goal. Chen et al. [10] demonstrated learned exploration in indoor environments without goal conditioning, but evaluation used custom setups that are difficult to reproduce. The lack of a common benchmark makes it impossible to determine whether learned policies genuinely outperform classical methods or merely benefit from favorable evaluation conditions. DisasterSim's current RL baseline is goal-conditioned; a goal-free learned explorer is identified as a direct extension in Section IX.

TABLE I
Comparison of disaster/exploration simulation benchmarks. ✓ = supported, ○ = partial, – = absent.

Benchmark	ROS 2 native	Ref. baselines	Frozen config	Auto metrics	Stat. tests
DARPA SubT Virtual [3]	–	–	–	○	–
RoboCup Rescue Sim. [4]	–	○	○	○	–
USARSim [5]	–	–	–	–	–
Habitat [6]	–	✓	✓	✓	○
DisasterSim (ours)	✓	✓	✓	✓	✓

TABLE II
Obstacle configuration per difficulty level.

Obstacle Type	Easy	Medium	Hard
Collapsed walls	4	6	8
Rubble piles	5	8	12
Pillar stumps	3	5	7
Debris pieces	8	14	20
Total	20	33	47

C. Positioning of DisasterSim

Table I summarizes how DisasterSim compares with existing platforms. DisasterSim is the only platform that combines a disaster-specific environment, standardized metrics, multiple reference baselines, and automated statistical evaluation in a single open-source package built on modern ROS 2 infrastructure.

III. System Architecture

DisasterSim is implemented as four ROS 2 Humble packages launched from a single entry point (`full_simulation.launch.py`). Fig. 1 shows the system architecture and the ROS 2 node communication graph.

A. Environment

The simulation world models a $20\times20\,\mathrm{m}$ ($A = 400\,\mathrm{m}^2$) post-earthquake building interior in Gazebo Classic with ODE physics (1 kHz step rate). The building has perimeter walls of 0.2 m thickness. Four custom SDF obstacle models are placed at configurable density (Table II):

Ambient lighting is reduced to simulate dust-filled post-earthquake conditions. The robot spawns at $(0, -2)$ m facing north ($\psi = \pi/2$), with a survivor marker placed at $(8, 12)$ m serving as the RL goal target. Obstacle positions are seeded by `random_seed: 42` in `benchmark.yaml` for cross-study reproducibility; researchers testing layout sensitivity should vary this seed explicitly.

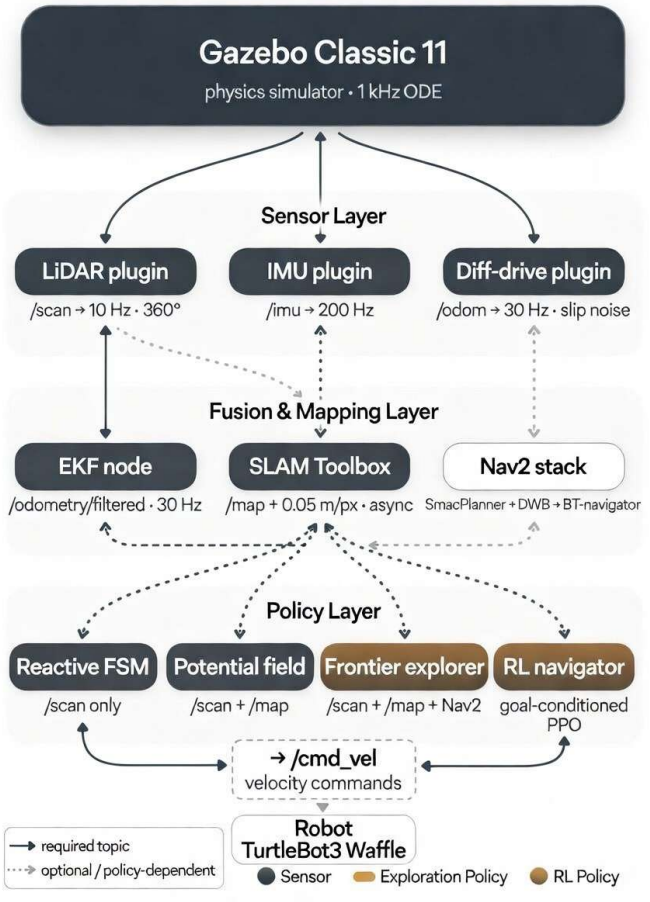

Fig. 1. DisasterSim system architecture. All components launch from a single command and communicate via ROS 2 topics. Dashed lines indicate optional components (Nav2 used only by the frontier explorer baseline; RL policy subscribes directly to sensor topics).

TABLE III
SENSOR SPECIFICATIONS WITH NOISE MODELS.

Sensor	Rate	Range/Res.	Noise
LiDAR	10 Hz	0.25–12 m, 1°	σ=0.01 m
IMU	200 Hz	—	σ=2×10^{-4} rad/s
Camera	30 fps	640×480, 80°	σ=3 px
Odom	30 Hz	—	σ=0.05 m/s (slip)

B. Robot Platform

The robot follows the TurtleBot3 Waffle form factor (287 mm wheelbase, 66 mm wheel diameter, 1.37 kg) with differential drive actuation ($v_{\max} = 0.22$ m/s, $\omega_{\max} = 1.82$ rad/s). The sensor suite is summarized in Table III. LiDAR noise is modeled as additive Gaussian (σ=0.01 m per beam). Odometry incorporates additional rubble-slip noise (σ=0.05 m/s) via Gazebo's differential drive plugin, mimicking the unpredictable wheel-ground contact on irregular terrain.

C. Sensor Fusion

An Extended Kalman Filter (`robot_localization`) fuses wheel odometry velocities (v_x, $\dot{\theta}$) with IMU heading (ψ) and angular rate ($\dot{\psi}$) at 30 Hz in 2D mode. A key design choice is that absolute position from wheel odometry is *excluded* to prevent unbounded drift from wheel slip on rubble:

$$\mathbf{x}_k = [x, y, \psi, v_x, \dot{\theta}]^\top \tag{1}$$

The observation vector fuses only $[v_x, \dot{\theta}, \psi, \dot{\psi}]$, with IMU heading providing an absolute orientation anchor that bounds rotational drift even as translational uncertainty grows.

The process noise covariance is tuned for differential-drive dynamics on uneven terrain ($Q_{xx} = Q_{yy} = 0.05$, $Q_{\psi\psi} = 0.06$). The EKF configuration is *frozen* across all baselines in `benchmark.yaml` to ensure that coverage differences reflect policy quality, not localization tuning.

D. SLAM and Navigation

SLAM uses `slam_toolbox` in asynchronous mode at $\delta = 0.05$ m/pixel resolution with 1 s update intervals, producing an occupancy grid map used by metrics computation and map- based baselines. The full Nav2 stack (SmacPlanner2D global planner, DWB local planner at 20 Hz, behavior server with spin/backup/wait recovery, and behavior-tree navigator) is available for policies requiring global path planning. Reactive policies bypass Nav2 and publish directly to `/cmd_vel`.

IV. BENCHMARK PROTOCOL

A. Evaluation Metrics

Four metrics are computed automatically by `benchmark_metrics.py` from per-trial output files:

Coverage (C). The percentage of the building area that has been mapped as free space:

$$C = \frac{\sum_{\text{free}} \delta^2}{A_{\text{building}}} \times 100\% \tag{2}$$

where $\delta = 0.05$ m/px, $A_{\text{building}} = 400\,\text{m}^2$, and a pixel is classified as free if its grayscale occupancy value exceeds 220 (i.e., $p_{\text{occ}} < 0.25$ in the ROS occupancy convention).

Localization RMSE (ρ). Euclidean error between EKF-fused pose estimate and Gazebo ground truth, sampled at 2 Hz:

$$\rho = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[\Delta x_i^2 + \Delta y_i^2]} \tag{3}$$

Near-collision rate (ν). The rate (events per minute) of episodes where the frontal LiDAR arc ($\pm 25°$) reads below the safety threshold of 0.30 m for ≥ 0.10 s continuously.

Efficiency (η). Coverage normalized by trial duration: $\eta = C/T_{\text{min}}$, where T_{min} is elapsed time in minutes. Since all trials share the same 300 s horizon, efficiency is a linear rescaling of coverage; it is included for comparability with future work using different trial durations.

TABLE IV
REFERENCE BASELINE CHARACTERISTICS.

Property	FSM	Frontier	PotField	RL
Uses map		✓	✓	
Uses Nav2		✓		
Learned				✓
Params	5	12	8	16 obs
Recovery mode	escape	wander	wall-follow	—

B. Experimental Protocol

Each trial follows a fixed sequence: 100 s initialization phase (Gazebo, SLAM, Nav2 startup and convergence), followed by 300 s of active exploration during which all metrics are computed. The experiment runner launches $N{=}10$ independent trials per policy, each with a fresh simulation instance initialized from the same frozen configuration. Per-trial outputs include: an occupancy grid map (.pgm + .yaml), an EKF trajectory log (.csv, 2 Hz), and a collision event log (.csv). Aggregate statistics (mean, std, 95% CI, Mann-Whitney U, Cohen's d) are computed automatically at the end of each policy's run.

C. Frozen Configuration

All benchmark parameters are specified in `benchmark.yaml` and must remain unchanged for cross-study comparisons. Key frozen values: building area (400 m^2), map resolution (0.05 m/px), free-cell threshold (220), collision threshold (0.30 m), trial duration (300 s), trials per policy ($N{=}10$), and random seed (42). Researchers testing modified scenarios must copy the file, rename it, and cite all deviations.

V. REFERENCE BASELINES

DisasterSim ships four reference policies spanning the spectrum from purely reactive to fully learned. All are implemented as ROS 2 nodes subscribing to `/scan`, `/odometry/filtered`, and optionally `/map`, and publishing velocity commands to `/cmd_vel`. Table IV summarizes their characteristics.

B1: Reactive FSM. A four-state machine (WAIT→FORWARD→TURN→NAVIGATE) using five LiDAR arc zones for obstacle detection. Periodic direction changes every 6 s prevent looping. A no-progress watchdog triggers a random escape maneuver after 30 s of displacement < 0.5 m. Requires no map or Nav2, representing the simplest viable exploration strategy.

B2: Frontier Explorer. Implements Yamauchi frontier detection [8] with Nav2 path planning. Frontiers are detected as connected components of free cells adjacent to unknown space, scored by a weighted combination of cluster size (60%) and inverse distance to robot (40%). Includes goal blacklisting after 3 consecutive planning failures, a 45 s no-progress watchdog that clears all Nav2 filters, and active wandering fallback with directional obstacle avoidance when no valid frontiers remain.

B3: Potential Field. Attractive force toward the nearest cluster of unknown cells [11] ($K_{\text{att}}{=}2.0$) combined with LiDAR-based repulsive forces ($K_{\text{rep}}{=}0.4$, $d_{\text{rep}}{=}1.2$ m). Stall detection triggers random perturbation; after 3 consecutive stalls, the robot switches to wall-following mode for 15 s before resuming potential field control. Requires the SLAM map but not Nav2.

B4: Goal-Conditioned RL Navigator (PPO). *Important design note:* this baseline is a **goal-conditioned navigator**, not an autonomous explorer. It navigates toward a fixed, known survivor location at $(8, 12)$ m rather than exploring freely. Its higher coverage relative to classical baselines reflects the incidental mapping that occurs along its goal-directed path, not a deliberate exploration strategy. A goal-free RL explorer—removing goal proximity and bearing from the observation and replacing the progress reward with a pure coverage signal—is planned for v2 and is the direct experimental complement to the three classical explorers presented here.

The policy uses Stable-Baselines3 PPO with a two-layer MLP (64–64, ReLU activations). The 16-dimensional observation space consists of 12 LiDAR sector distances (normalized by `range_max` = 12 m), goal proximity ($1/(1{+}d)$ to the fixed survivor location), goal bearing (β/π), and current velocity ($v_x/v_{\max}$, $\dot{\theta}/\omega_{\max}$). Actions are continuous linear and angular velocity commands.

Training uses a three-stage curriculum with progressively tighter goal radii (2.0→1.0→0.5 m, 600k total planned steps). The reward function combines dense goal progress with incidental exploration incentives and a one-shot collision penalty:

$$r_t = \underbrace{40\Delta d + 20\Delta d^*}_{\text{goal progress}} + \underbrace{5 \cdot \mathbb{1}_{\text{new cell}} + 0.3 \cdot \mathbb{1}_{|v_x|>0.04}}_{\text{incidental coverage}} - 0.01 \tag{4}$$

where Δd is step-wise progress toward the fixed goal and Δd^* is best-ever distance improvement. A one-shot collision penalty (-30, 20-step cooldown) and goal-reached bonus ($+500$) complete the reward signal. The one-shot design—rather than a per-step penalty—was critical to prevent a degenerate "stay still" policy observed in earlier training runs. The evaluated model completed 370k of 600k planned steps (62% of curriculum).

Extensibility. Researchers add custom policies by implementing a ROS 2 node and registering it in the launch file as a new `nav_policy` option. No modification to the benchmark infrastructure is required.

VI. EXPERIMENTS AND RESULTS

A. Experimental Setup

All experiments were conducted on Ubuntu 22.04 with ROS 2 Humble and Gazebo Classic 11, on a desktop PC with a Ryzen 4800H processor (16 GB RAM). Each policy was evaluated over $N{=}10$ independent trials using the "easy" obstacle configuration (20 obstacles) with a 300 s exploration horizon. One RL trial was excluded due to a `map_saver` process failure (the map file was not written before trial teardown), yielding 9 valid RL trials and 39 total valid trials across all policies.

TABLE V
EXPLORATION COVERAGE (%) ACROSS 300 S TRIALS.

Policy	n	Mean	Std	95% CI	Range
Reactive FSM	10	29.8	5.8	[26.2, 33.4]	22.3–40.8
Potential Field	10	31.7	7.7	[26.9, 36.4]	24.7–47.4
Frontier	10	31.5	15.9	[21.6, 41.3]	11.9–55.4
RL (PPO, 370k)	9	36.9	11.7	[29.2, 44.6]	22.5–61.1

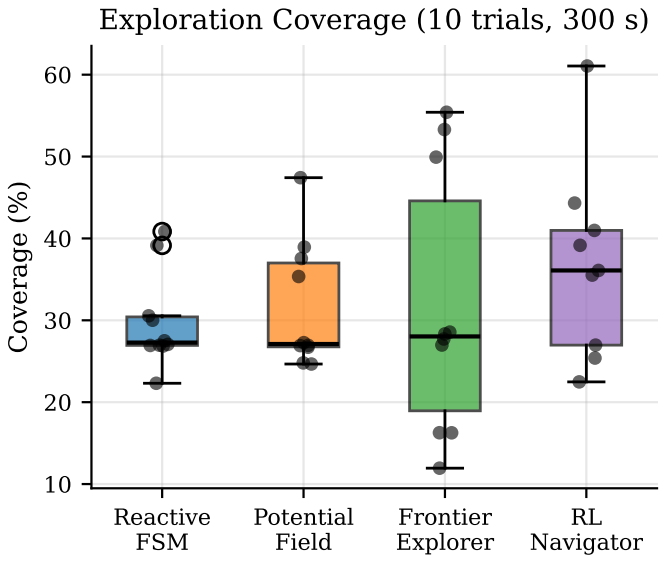

Fig. 2. Coverage distribution per policy with individual trial scatter points. The three classical exploration baselines cluster around 30%. The goal-conditioned RL policy achieves higher coverage (36.9% mean, 61.1% peak), but note that it navigates toward a known fixed goal rather than exploring freely—see Section VII-B for interpretation.

B. Coverage Results

Table V and Fig. 2 present the primary coverage results.

The three classical *exploration* baselines converge tightly to 30–32% mean coverage despite employing fundamentally different strategies. The goal-conditioned RL policy achieves the highest mean (36.9%) and peak single-trial coverage (61.1%)—the best result across all 39 runs. However, this comparison is not symmetric: the classical baselines are pure explorers with no knowledge of the survivor location, while the RL agent navigates toward the fixed goal at $(8, 12)$ m and accumulates coverage incidentally along its path. The difference in coverage therefore reflects a combination of learned navigation skill *and* the structural advantage of having a goal that routes the robot through unexplored regions. The reactive FSM exhibits the lowest variance (σ=5.8%), reflecting its deterministic obstacle avoidance, while the frontier explorer shows the highest (σ=15.9%) due to Nav2 path planning sensitivity (see Section VII-C).

C. Per-Trial Results

Table VI reports all 39 individual trial results, enabling researchers to inspect the full distribution rather than summaries alone. The frontier explorer's bimodal behavior is apparent: trials 2, 4, 10 achieve 49–55% coverage while trials 1, 5, 8 reach only 12–16%.

TABLE VI
ALL 39 INDIVIDUAL TRIAL RESULTS. F=FRONTIER, R=REACTIVE FSM, P=POTENTIAL FIELD, RL=RL (PPO).

Trial	Policy	Cov (%)	RMSE (m)	Eff (%/min)	Coll (/min)
1	F	16.3	1.77	3.25	0.00
2	F	49.9	6.39	9.99	0.00
3	F	28.6	3.16	5.71	0.60
4	F	55.4	7.70	11.08	0.60
5	F	16.3	1.70	3.25	0.00
6	F	28.3	3.52	5.67	0.20
7	F	27.7	1.89	5.54	0.20
8	F	11.9	1.79	2.39	0.00
9	F	27.0	1.95	5.40	0.00
10	F	53.3	9.88	10.66	0.00
1	R	39.1	4.45	7.83	0.00
2	R	30.6	1.98	6.11	0.00
3	R	40.8	4.91	8.17	0.20
4	R	27.5	4.00	5.50	0.20
5	R	27.0	1.42	5.39	0.00
6	R	26.9	1.38	5.37	0.00
7	R	22.3	1.56	4.46	0.00
8	R	26.9	1.36	5.39	0.00
9	R	30.0	1.69	6.00	0.20
10	R	27.1	1.48	5.41	0.00
1	P	24.7	2.16	4.93	0.00
2	P	37.5	3.31	7.51	0.40
3	P	26.9	1.50	5.39	0.00
4	P	26.9	2.35	5.38	0.20
5	P	24.8	2.50	4.96	0.00
6	P	26.7	1.02	5.34	0.00
7	P	38.9	3.38	7.79	0.00
8	P	47.4	5.76	9.48	0.00
9	P	35.4	2.41	7.07	0.00
10	P	27.3	1.66	5.45	0.20
1	RL	44.3	2.79	8.86	0.20
2	RL	61.1	7.49	12.21	0.80
3	RL	35.5	4.36	7.10	0.80
4	RL	27.0	1.52	5.39	0.00
5	RL	39.2	3.44	7.83	0.20
6	RL	36.1	4.01	7.22	0.20
7	RL	25.4	2.06	5.08	0.00
8	RL	41.0	3.46	8.20	0.20
9	RL	22.5	1.69	4.50	0.00

Fig. 3 shows representative SLAM maps from each policy at a trial near the median coverage. The reactive FSM produces dense but spatially concentrated maps, while the frontier and RL policies produce more spatially distributed maps extending further from the spawn point.

D. Multi-Metric Analysis

Table VII and Fig. 4 present all four evaluation metrics aggregated across trials.

Localization RMSE (ρ) correlates positively with distance traveled: the reactive FSM shows the lowest mean RMSE (2.42 m) because its conservative motion limits odometric drift accumulation, while the frontier explorer shows the highest RMSE (3.98 m) as it traverses longer, more tortuous paths. The potential field navigator achieves RMSE of 2.61 m—notably lower than the frontier explorer—because it avoids the long replanning detours that characterize frontier behavior in

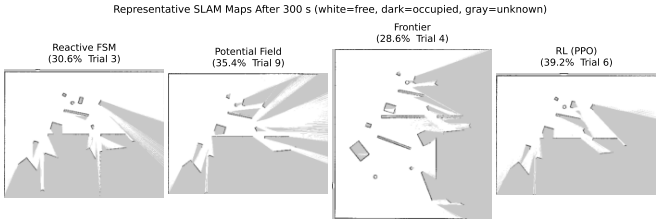

Fig. 3. Representative SLAM maps after 300 s for each policy (trials near median coverage). White: free space. Dark: occupied. Gray: unknown. The frontier and RL policies achieve more distributed spatial coverage despite similar or lower mean values compared to reactive FSM in these particular trials.

TABLE VII
MULTI-POLICY EXPLORATION COMPARISON (300 S, 10 TRIALS PER POLICY).

Policy	Coverage (%)	Eff. (%/min)	RMSE (m)	Coll. (/min)
Reactive FSM	29.8 ± 5.8	5.96	2.424	0.060
Potential Field	31.7 ± 7.7	6.33	2.606	0.080
Frontier Explorer	31.5 ± 15.9	6.29	3.976	0.160
RL Navigator	36.9 ± 11.7	7.38	3.426	0.267

cluttered spaces. This inter-policy ordering confirms that the EKF correctly captures the coverage–drift trade-off intrinsic to dead-reckoning localization on uneven terrain.

Near-collision rate (ν) remains low across all policies ($\leq$ 0.27 events/min), indicating that all baselines maintain adequate operating clearances in the "easy" configuration. The RL policy shows the highest rate (0.27/min), consistent with its more aggressive learned behavior that implicitly trades safety margin for coverage gain. The potential field navigator has a lower collision rate (0.08/min) than the frontier explorer (0.16/min), reflecting its explicit repulsive force field.

Efficiency (η) directly reflects coverage since all trials share the same 300 s duration. The RL policy achieves 7.38 %/min compared to 5.96–6.33 %/min for classical methods, representing a 17–24% improvement in exploration rate.

Fig. 5 shows the full coverage and RMSE distributions alongside trial-to-trial reproducibility measured by coefficient of variation (CV$= \sigma/\mu$).

Fig. 6 shows frontal LiDAR range profiles over 300 s for representative trials of each policy, directly visualizing near-collision behavior from raw sensor data.

E. Coverage–Localization Relationship

Fig. 7 reveals a significant cross-policy trend: coverage correlates positively with localization RMSE (Pearson r=0.85, p<0.001 across all 39 trials). Robots that cover more ground accumulate greater odometric drift because they travel further from their start pose, amplifying wheel-slip errors over the 300 s horizon.

This suggests a fundamental tension in disaster navigation: aggressive exploration improves coverage but degrades localization quality, which may in turn reduce the accuracy of survivor position estimates derived from that map. This coverage–localization trade-off is a quantifiable property of

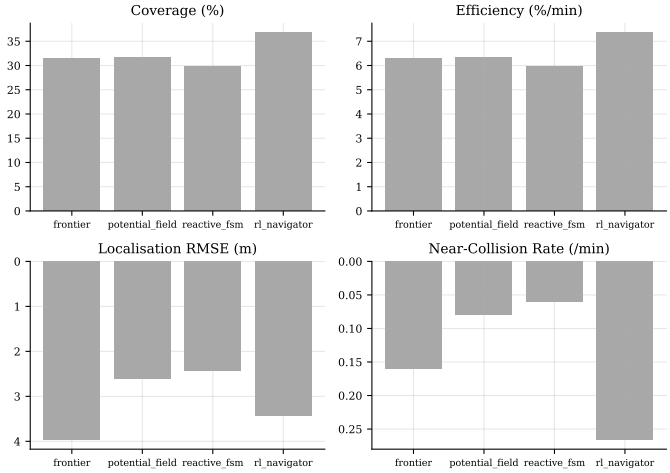

Fig. 4. Multi-metric comparison across all four policies. Top row: coverage and efficiency (higher is better). Bottom row: localization RMSE and near-collision rate (lower is better).

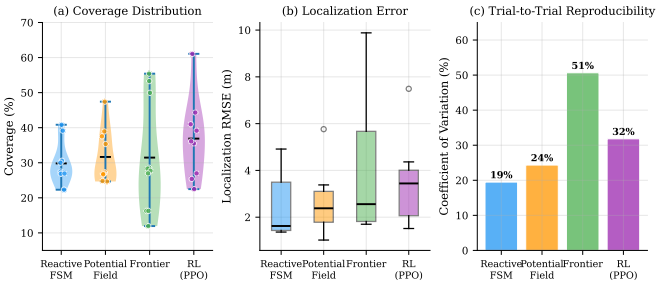

Fig. 5. (a) Coverage distributions (violin + scatter). (b) Localization RMSE distributions. (c) Trial-to-trial reproducibility (CV$= \sigma/\mu$); lower is more reproducible. The reactive FSM achieves the most consistent behavior (CV$=$ 19.5%); the frontier explorer is the least consistent (CV$=$ 50.6%) due to its bimodal Nav2 planning behavior.

the benchmark that future work should explicitly address, for example through loop-closure SLAM, auxiliary localization infrastructure (UWB beacons, visual place recognition), or exploration policies that explicitly reason about localization uncertainty.

F. Reproducibility Analysis

A benchmark is only useful if it produces consistent measurements. Trial-to-trial reproducibility quantified via the coefficient of variation (CV $= \sigma/\mu \times 100\%$) shows that the reactive FSM achieves the best consistency (CV$=$ 19.5%), followed by potential field (CV$=$ 24.3%) and RL (CV$=$ 31.8%). The frontier explorer has the highest variance (CV$=$ 50.6%), reflecting its bimodal behavior: when Nav2 successfully plans through rubble the mean coverage is approximately 38.6% (trials 2, 4, 7, 9, 10), while when planning fails and the robot resorts to wandering the mean drops to approximately 17.0% (trials 1, 5, 6, 8 inclusive of outlier Trial 3 which succeeded partially).

All 39 trials completed without simulation crashes and produced valid occupancy maps and metric values—an important

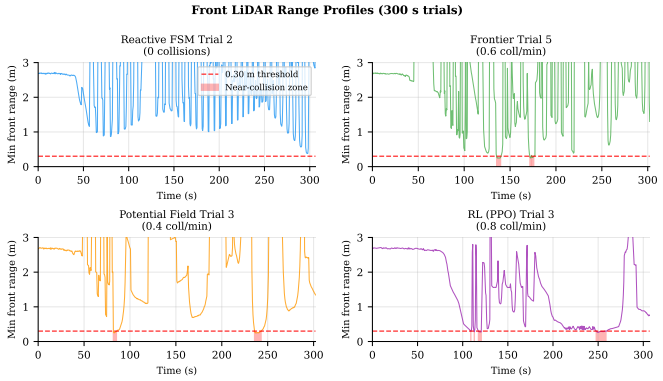

Fig. 6. Frontal LiDAR range profiles over 300 s for representative trials of each policy. Red shaded regions: near-collision events (<0.30 m, >0.10 s). The RL policy shows the most frequent near-obstacle behavior, consistent with its higher coverage but elevated collision rate.

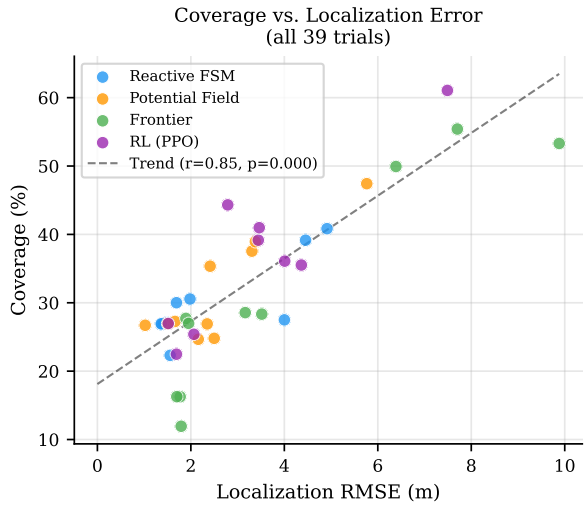

Fig. 7. Coverage vs. localization RMSE across all 39 trials. The positive correlation (r=0.85, p<0.001) reveals a fundamental tension between exploration breadth and localization accuracy in dead-reckoning-based navigation on rubble terrain.

benchmark validity criterion. No-progress watchdogs embedded in each policy prevented initialization-failure trials; this protection was identified as necessary after early development trials where stuck robots produced degenerate 3% coverage.

G. Statistical Analysis

Omnibus test. A Kruskal-Wallis test across all four policies yields H=1.70, p=0.637, indicating that differences are not statistically significant at α=0.05 with the current sample size. This null result is *expected*: n=9–10 trials per policy provides approximately 40–50% power to detect large effects ($d\geq$0.8) but insufficient power to reliably detect medium effects.

Pairwise comparisons. Pairwise Mann-Whitney U tests (Table VIII) confirm no significant difference between classical baselines (p>0.79), consistent with their near-identical mean coverage ($|d|\leq$0.27). The RL policy shows a medium-to-large effect size relative to the reactive FSM (d=0.78) and

TABLE VIII
PAIRWISE MANN-WHITNEY U TESTS AND COHEN'S d EFFECT SIZES (COVERAGE).

Comparison	U	p	Cohen's d
Frontier vs Potential Field	53	0.850	+0.01 (negligible)
Frontier vs Reactive FSM	52	0.940	+0.14 (negligible)
Potential Field vs Reactive FSM	46	0.791	+0.27 (small)
RL vs Reactive FSM	61	0.103	+0.78 (medium–large)
RL vs Potential Field	58	0.154	+0.53 (medium)
RL vs Frontier	54	0.244	+0.38 (small)

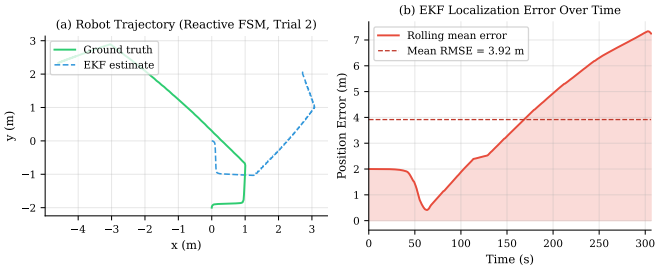

Fig. 8. EKF localization validation (Reactive FSM, representative trial). (a) Ground truth vs. EKF trajectory. (b) Position error over time. Error grows as the robot travels further from the start; the EKF bounds this growth via IMU heading anchoring.

a medium effect relative to potential field (d=0.53), though neither reaches significance at α=0.05.

Sample size guidance. Post-hoc power analysis indicates that detecting the observed RL–reactive difference (d=0.78) at α=0.05, β=0.80 requires n=28 trials per policy (≈50 minutes per policy at RTF≈2.0). DisasterSim's automated runner supports arbitrary N; researchers seeking definitive statistical conclusions should use $N\geq$28.

H. EKF Sensor Fusion Validation

Fig. 8 validates the EKF sensor fusion subsystem by comparing fused pose estimates against Gazebo ground truth over a representative 300 s trial. Localization error grows monotonically as the robot travels further from the start pose and wheel-slip errors accumulate. The EKF bounds this growth via IMU heading anchoring, preventing the unconstrained exponential divergence that would occur with pure odometry. This validation confirms that coverage differences between policies reflect exploration strategy, not localization quality.

VII. DISCUSSION

A. The Classical Exploration Ceiling

The most striking finding is the convergence of three fundamentally different classical approaches—reactive obstacle avoidance, map-based frontier planning, and potential field navigation—around 30% coverage with between-policy standard deviation of less than 2 percentage points (p>0.85, $|d|\leq$0.27). These methods differ in their use of maps, planning algorithms, and recovery strategies, yet perform indistinguishably.

This convergence is strong evidence that the performance limitation arises from *environmental constraints and navigation failure modes* rather than deficiencies in exploration strategy design. All classical policies encounter the same structural bottlenecks: narrow passages between rubble that require precise maneuvering ($<$0.5 m gaps between obstacles), dead-end corridors where the robot exhausts its exploration budget executing recovery behaviors (turning, wall-following, replanning), and obstacle configurations that trigger repeated escapes consuming significant fractions of the 300 s budget. The exploration strategy determines *which* obstacles the robot encounters, but the navigation response to those obstacles is similarly limited across all hand-designed approaches because they all rely on the same underlying kinematic constraints.

This "navigation bottleneck" hypothesis makes a testable prediction: increasing obstacle density ("medium"/"hard" configurations) should reduce coverage for all classical methods equally, while policies that learn to navigate narrow passages should show a relative improvement. We leave this test to future work.

B. Goal-Conditioned RL: Coverage Advantage and Its Interpretation

The PPO policy achieves higher mean coverage (36.9% vs. 30–32%) and a medium-to-large effect size (d=0.78 vs. reactive FSM). However, this result must be interpreted carefully because the comparison is not between two exploration strategies—it is between three *free exploration* policies and one *goal-conditioned navigator*.

The RL agent observes the bearing and distance to a fixed goal at $(8, 12)$ m in every step and is rewarded primarily for approaching that goal. Its higher coverage is therefore partly a function of the route it must take through the environment to reach that location, not solely a consequence of learned exploration behavior. In a different environment layout where the optimal goal-directed path passed through a narrow corridor covering little free space, the RL agent's coverage advantage could disappear entirely.

What the result *does* show is that a learned navigator trained to reach a specific point can, as a side effect, outperform classical free explorers at area coverage—which is itself a useful finding for disaster response scenarios where a survivor location is approximately known (e.g., from acoustic sensing or prior building knowledge) and thorough mapping along the approach route is required.

The RL agent's higher variance (σ=11.7% vs. 5.8–7.7%) reflects the sensitivity of the goal-directed path to specific obstacle configurations: the same learned policy produces very different routes depending on which obstacles block the direct approach in each trial.

The fair comparison—a goal-free RL explorer—is future work. Removing goal proximity and bearing from the observation space and replacing the $40\Delta d + 20\Delta d^*$ progress reward with a pure coverage signal would produce a policy trained to maximize area coverage without prior knowledge of any goal location. Such a policy would constitute a genuine learned exploration baseline, directly comparable to the three classical methods presented here. DisasterSim's infrastructure—particularly the automated coverage metric and experiment runner—is designed to support exactly this extension without modification.

C. Frontier Explorer Bimodality

The frontier explorer's coverage distribution is strikingly bimodal: trials where Nav2 successfully plans through rubble achieve 49–55% coverage, while trials where planning fails early produce 12–16%. This bimodality reveals that planning-based methods in disaster environments are *brittle to obstacle placement*—a finding that would be completely obscured by reporting only mean coverage. The benchmark's per-trial reporting (Table VI) and visualization (Fig. 2) make this distributional characteristic visible, demonstrating the essential value of full-distribution reporting over summary statistics.

The bimodality also points to a specific failure mode: Nav2's SmacPlanner2D appears to either find a viable route through rubble early in the trial (leading to high coverage) or become stuck in a local replanning loop (leading to wandering behavior with low systematic coverage). Diagnosing and addressing this brittleness—for example, through richer costmap inflation or alternative planners—is a concrete improvement direction that DisasterSim makes visible.

D. Limitations

Goal-conditioned RL baseline (primary limitation). The current RL policy (B4) navigates toward a fixed, hardcoded goal at $(8, 12)$ m using goal proximity and bearing in its observation space. It is therefore a *goal-conditioned navigator*, not an autonomous explorer, and its coverage results are not directly comparable to the three classical exploration baselines. This is the most significant limitation of the current benchmark evaluation. Designing and training a goal-free RL explorer is the highest-priority extension (see Section IX).

Single environment configuration. All experiments use only the "easy" configuration. The "medium" and "hard" difficulty levels are implemented in the YAML but unevaluated. Results may not generalize: the RL advantage could diminish if the learned policy was trained only on easy-density obstacles.

Statistical power. n=10 trials per policy provides $\approx 50\%$ power to detect large effects ($d \geq 0.8$) at α=0.05. The RL–classical gap (d=0.78) remains statistically non-significant. The automated runner makes $N \geq 28$ computationally feasible and is recommended for definitive conclusions.

RL training completeness. The PPO policy completed 370k/600k steps (62% of curriculum). Its final performance and variance characteristics after full training remain unknown. Architectural limitations (MLP without temporal memory) may bound performance regardless of training length.

2D flat ground plane. The current environment uses flat ground with 3D obstacles. Real rubble has continuous height variation affecting wheel-ground contact, yaw dynamics, and energy consumption. A 3D terrain mesh extension is planned for v2.0.

No sim-to-real validation. All results are simulation-only. Transfer to physical TurtleBot3 hardware would introduce additional challenges not captured here, including real-time scheduling jitter, motor deadband, and sensor drift that is difficult to model accurately in simulation.

These limitations bound the absolute performance values reported but do not undermine the benchmark's utility as a standardized comparison platform or the relative trends observed between policy paradigms.

VIII. Reproducibility and Open Access

DisasterSim is designed for complete reproducibility. The full source code, frozen benchmark configuration (`benchmark.yaml`), trained RL model weights (`disaster_ppo_v2_final.pt`), all 39 per-trial metric JSONs, EKF trajectory CSVs, occupancy map PGMs, and figure generation scripts are publicly available at:

https://github.com/newton-adhikari/ros2-disaster-robot-sim

Reproducing the complete benchmark requires only:

```
colcon build
source install/setup.bash
./run_experiments.sh
```

This single command executes all 40 trials (4 policies $\times$ 10 trials), computes per-trial metrics, and generates aggregate statistics with significance tests and effect sizes. Total runtime is approximately 3.5 hours on the reference hardware. All dependencies are standard ROS 2 Humble packages installable via `apt` with no proprietary components.

IX. Future Work

Goal-free RL explorer (highest priority). The current RL baseline is goal-conditioned and therefore not a fair exploration comparator. The direct next step is to train a goal-free PPO policy by: (1) removing goal proximity and bearing from the 16-dimensional observation space, leaving a 14-dimensional LiDAR-only input; (2) replacing the $40\Delta d + 20\Delta d^*$ progress reward with a pure coverage reward ($+5$ per new cell); and (3) evaluating under the same frozen benchmark protocol. This will produce a genuinely comparable learned explorer for v2 of this paper.

Multi-difficulty evaluation. Running the full benchmark at "medium" and "hard" obstacle densities (33 and 47 obstacles respectively) will test whether the classical ceiling persists, shifts, or disappears, and whether the RL advantage scales with environment complexity. This is the most immediate next step.

Complete goal-conditioned training and ablation. Finishing the three-stage curriculum (remaining $\approx$230k steps) and ablating architectural choices (MLP vs. recurrent, with vs. without goal observation) will characterize the goal-conditioned navigator more fully and bound the coverage achievable with goal information—providing a useful upper-reference point against which the forthcoming goal-free explorer can be compared.

Additional learned baselines. Integrating SAC, TD3, and model-based RL (Dreamer-style) methods will test whether the PPO advantage generalizes across learning algorithms and whether model-based methods are more sample-efficient in this domain.

Addressing the coverage–localization trade-off. The $r{=}0.85$ correlation between coverage and RMSE (Section VI-E) motivates exploration policies that explicitly reason about localization uncertainty. Approaches include loop-closure SLAM, active localization uncertainty reduction, and uncertainty-aware reward shaping.

Multi-robot coordination. Extending to 2–4 robots with configurable communication dropout addresses a critical gap: real disaster response uses robot teams, and the communication infrastructure may be partially destroyed.

Sim-to-real transfer. Validating benchmark findings on a physical TurtleBot3 in a controlled mock disaster environment (stacked foam blocks, plywood debris) will establish the simulation's predictive validity and identify systematic gaps.

X. Conclusion

We presented DisasterSim, a reproducible open-source benchmark for evaluating robot navigation and coverage policies in simulated collapsed building environments. The benchmark provides a complete evaluation pipeline—from one-command simulation launch through automated metric computation and statistical testing—with frozen parameters ensuring cross-study comparability.

Our empirical evaluation across 39 trials reveals three findings. First, three fundamentally different classical exploration strategies— reactive FSM, frontier-based planning, and potential field—converge to statistically indistinguishable performance ($\sim$30% coverage, $|d|{\leq}0.27$), providing strong evidence that navigation constraints, not exploration strategy, form the primary performance bottleneck in cluttered disaster environments. Second, a goal-conditioned PPO policy trained to navigate toward a known survivor location achieves higher incidental coverage (36.9% mean, $d{=}0.78$)—indicating that a robot with a fixed navigation target traverses more of the environment than classical explorers manage—but this comparison is not symmetric and a goal-free learned explorer remains as the essential next experiment. Third, a quantifiable coverage–localization trade-off ($r{=}0.85$) is identified as a structural property of dead-reckoning navigation in cluttered environments that future policies must explicitly address.

DisasterSim provides the standardized infrastructure—environment, metrics, baselines, and automated runner—needed to pursue these directions with reproducible, directly comparable results. The benchmark is designed to grow: adding the goal-free RL explorer, evaluating medium and hard difficulty levels, and eventually validating on physical hardware are all supported without modification to the core benchmark infrastructure.

References

[1] R. R. Murphy, *Disaster Robotics*. MIT Press, 2014.

[2] J. Delmerico *et al.*, "The current state and future outlook of rescue robotics," *J. Field Robotics*, vol. 36, no. 7, pp. 1171–1191, 2019.

[3] T. Chung *et al.*, "DARPA Subterranean Challenge: Multi-domain exploration and search," *Field Robotics*, vol. 3, pp. 1–68, 2023.
[4] A. Jacoff, E. Messina, and J. Evans, "Experiences in deploying test arenas for autonomous mobile robots," in *Proc. PerMIS*, 2001.
[5] S. Balakirsky *et al.*, "USARSim: Providing a framework for multi-robot performance evaluation," in *Proc. PerMIS*, 2006.
[6] M. Savva *et al.*, "Habitat: A platform for embodied AI research," in *Proc. ICCV*, 2019.
[7] B. Zhou *et al.*, "FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning," *IEEE RA-L*, vol. 6, no. 2, pp. 779–786, 2021.
[8] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE CIRA*, 1997, pp. 146–151.
[9] B. Charrow *et al.*, "Information-theoretic planning with trajectory optimization for dense 3D mapping," in *Proc. RSS*, 2015.
[10] T. Chen *et al.*, "Learning exploration policies for navigation," in *Proc. ICLR*, 2019.
[11] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robotics Res.*, vol. 5, no. 1, pp. 90–98, 1986.
[12] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Autonomous Robots*, vol. 13, no. 3, pp. 207–222, 2002.
[13] J. Schulman *et al.*, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.
[14] Y. Bengio *et al.*, "Curriculum learning," in *Proc. ICML*, 2009, pp. 41–48.