A Sensor-Aware Phenomenological Framework for LiDAR Degradation Simulation and SLAM Robustness Evaluation

Doumegna Mawuto Koudjo Felix, Xianjia Yu, Zhuo Zou, and Tomi Westerlund, *Senior Member, IEEE*

***Abstract*—Light detection and ranging (LiDAR)-based simultaneous localization and mapping (SLAM) systems are highly sensitive to adverse conditions such as occlusion, noise, and field-of-view (FoV) degradation, yet existing robustness evaluation methods either lack physical grounding or do not capture sensor-specific behavior. This article presents a sensor-aware phenomenological framework for simulating interpretable LiDAR degradations directly on real point clouds, enabling controlled and reproducible SLAM stress testing. Unlike image-derived corruption benchmarks (e.g., SemanticKITTI-C) or simulation-only approaches (e.g., LiDARSim), the proposed system preserves per-point geometry, intensity, and temporal structure while applying structured dropout, FoV reduction, Gaussian noise, occlusion masking, sparsification, and motion distortion. The framework features autonomous topic and sensor detection, a modular configuration with four predefined severity tiers (*light–extreme*), and real-time performance ($<$ 5 ms per frame for solid-state LiDAR and $<$ 20 ms for dense, wide-FoV spinning LiDAR). The implementation is Docker-containerized and compatible with robot operating system (ROS) workflows. Experimental validation across three LiDAR models and five state-of-the-art SLAM systems reveals distinct patterns of robustness shaped by sensor design and environmental context. The open-source implementation provides a practical foundation for benchmarking LiDAR-based SLAM under physically meaningful degradation scenarios.**

***Index Terms*—Adverse environmental simulation, light detection and ranging (LiDAR), simultaneous localization and mapping (SLAM) benchmarking, sensor degradation.**

I. INTRODUCTION

LIGHT detection and ranging (LiDAR) sensors play a vital role in robotic perception tasks, yet their performance degrades under adverse conditions, such as fog, dust, occlusion, and dynamic motion, leading to reduced simultaneous localization and mapping (SLAM) reliability [1]. Prior studies have analyzed LiDAR degradation effects. Bijelic et al. [2] studied fog-induced attenuation, Zhang et al. [3] reviewed degradation taxonomies, and Wu et al. [4] proposed an uncertainty-aware LiDAR–inertial measurement unit (IMU) fusion approach that models measurement noise to improve odometry robustness. Recent surveys [5] further emphasize the need for physically interpretable models grounded in real sensor behavior. However, these works do not provide modular, sensor-specific simulation frameworks tailored for robustness evaluation in SLAM. High-fidelity simulation environments, such as CARLA [6] and LiDARSim [7] simulate LiDAR sensing through full scene reconstruction and ray tracing. While physically detailed, these approaches require complete environment modeling and are less suited for rapid robustness testing on real recorded datasets.

Robustness benchmarks, such as SemanticKITTI-C, nuScenes-C, KITTI-C, and TIERS Dataset [8], [9], [10], [11], evaluate perception under unified 3-D corruptions. While valuable for downstream tasks, these benchmarks apply generic perturbations that do not model the characteristics of different LiDAR hardware types. Augmentation methods, such as PolarMix [12], Real3D-Aug [13], and RealAug [14], target neural network training and often rely on artificial or learned perturbations that do not reflect physical sensor behavior. As a result, they provide limited insights into SLAM or LiDAR-inertial odometry (LIO) performance under realistic LiDAR degradations.

To address this gap, we propose a sensor-aware phenomenological framework that applies physically interpretable degradations directly to raw LiDAR measurements, enabling controlled reproducible evaluation of SLAM robustness. Our design and degradation scenarios build on findings in [2], [3], [4], and [5], emphasizing interpretable modeling of noise, occlusion, field-of-view (FoV) reduction, motion distortion, and other environment-induced artifacts. Unlike augmentation pipelines designed for data-driven training, our objective is to simulate realistic perturbations rather than improve learning-based generalization, enabling structured benchmarking under adverse conditions. This work extends our previous dataset [15], which characterized Livox Avia, Livox Mid-360, and Ouster LiDARs under nominal conditions, by introducing controlled degradation mechanisms applicable across diverse sensor architectures.

Our contributions are summarized as follows.

Received 9 December 2025; revised 15 March 2026; accepted 10 April 2026. Date of publication 16 April 2026; date of current version 11 May 2026. This work was supported by the Research Council of Finland's Digital Waters flagship under Grant 359247. This article was recommended for publication by Associate Editor K. Okada upon evaluation of the reviewers' comments. *(Corresponding author: Xianjia Yu.)*

Doumegna Mawuto Koudjo Felix and Zhuo Zou are with the College of Future Information Technology, Fudan University, Shanghai 200437, China, and also with the Turku Intelligent Embedded and Robotic Systems Lab, University of Turku, 20014 Turku, Finland (e-mail: 23210720352@m.fudan.edu.cn; zhuo@fudan.edu.cn).

Xianjia Yu and Tomi Westerlund are with the Turku Intelligent Embedded and Robotic Systems Lab, University of Turku, 20014 Turku, Finland (e-mail: xianjia.yu@utu.fi; tovewe@utu.fi).

This article has supplementary downloadable material available at https://doi.org/10.1109/RAP.2026.3684773, provided by the authors.

Digital Object Identifier 10.1109/RAP.2026.3684773

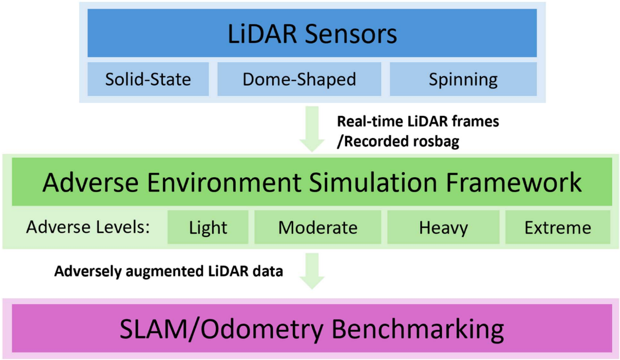

Fig. 1. Architecture of the proposed adverse environment simulation framework for robustness evaluation of SLAM algorithms.

1) *Sensor-aware degradation modeling*: It captures scanning-pattern differences across solid-state (Livox Avia, Mid-360) and spinning (Ouster) LiDARs.
2) *Modular and reproducible configuration*: It parameterizes degradation scenarios with four predefined severity tiers (*light–extreme*), including dropout, FoV reduction, noise, occlusion, sparsification, and motion distortion. Severity tiers can be customized for user-defined degradation scenarios.
3) *Real-time robot operating system (ROS) implementation*: It provides a Docker-containerized C++/Python ROS framework with autonomous topic and sensor detection, integrated visualization, and processing latency below 5 ms for solid-state LiDAR and below 20 ms for dense wide-FoV spinning LiDAR.
4) *Systematic SLAM robustness evaluation*: It enables physically grounded comparison across heterogeneous sensors and state-of-the-art SLAM under adverse conditions.

II. METHODOLOGY

A. *Framework Overview*

Fig. 1 shows the architecture of the proposed sensor-aware phenomenological degradation framework. The system processes multisensor LiDAR streams from both recorded ROS bags and real-time inputs. Incoming point clouds are parsed, time-stamp-aligned, and converted into a unified representation before passing through the degradation pipeline.

Physically interpretable augmentation modules (see Section II-D) are applied according to predefined severity levels (*light–extreme*). Multiple degradation types can be chained or randomized to emulate adverse sensing conditions. For real-time operation, the framework applies a fixed-delay output buffer to the LiDAR and IMU streams prior to republication, thereby preserving chronological consistency. The augmented point clouds are republished on dedicated ROS topics, enabling seamless integration with a wide range of LIO and SLAM systems. Implementation details are provided in our GitHub repository and supplementary material II-F.

TABLE I
DEGRADATION PARAMETERS FOR FOUR SEVERITY TIERS

Parameter	**Light**	**Moderate**	**Heavy**	**Extreme**
Dropout rate r	0%	15%	30%	40%
Gaussian noise σ (m)	0.008	0.020	0.045	0.060
Outlier rate p_{out}	0.3%	1.0%	2.5%	4.0%
Outlier std. σ_{out} (m)	0.15	0.40	0.80	1.20
FoV horiz. reduction h_{red}	0%	8%	15%	20%
FoV vert. reduction v_{red}	0%	12%	25%	30%
Occlusion count N_{occ}	0	0	4	6
Occlusion radius R (m)	—	—	2.5	3.0
Motion distortion	No	No	No	Yes
Linear velocity $\|\mathbf{v}\|$ (m/s)	—	—	—	2.0
Angular velocity $\|\boldsymbol{\omega}\|$ (rad/s)	—	—	—	0.3

B. *Autonomous Sensor Configuration*

The framework includes an autonomous configuration mechanism that removes the need for manual setup across different LiDAR platforms. Sensor characteristics are inferred directly from incoming point cloud fields, allowing automatic identification of each LiDAR stream. Active topics are discovered at runtime, and augmented outputs are published using consistent naming conventions.

C. *Scenario Design and Parameterization*

Degradation scenarios are defined through configuration files that specify parameters for five perturbation types: dropout, FoV reduction, additive noise, occlusion masking, and motion distortion, following prior literature [2], [3], [4], [5]. Parameters are grouped into four severity tiers (*light–extreme*) for consistent SLAM stress testing across diverse LiDAR architectures. Complete configuration templates are provided in the open-source repository, and the parameter values for all tiers are summarized in Table I.

Design philosophy: The framework follows two principles: 1) *physical relevance*, using parameters that reflect measurable sensor or environmental effects, and 2) *comparability*, applying identical severity tiers across sensors and experiments. This tiered design is consistent with recent corruption benchmarks, such as SemanticKITTI-C and nuScenes-C [8], [9], [10].

Modeling approach: We adopt a phenomenological modeling strategy derived from observable LiDAR behavior rather than full radiometric or atmospheric simulation. The degradation modules preserve geometric consistency, temporal ordering, and each sensor's characteristic scanning pattern while remaining suitable for real-time computation in both bag replay and live operation.

Parameter justification: Severity tiers span realistic ranges informed by empirical observation, sensor physics, and prior corruption benchmarks. They are intended not to reproduce a single environmental condition exactly, but to define standardized degradation bands that expose robustness transitions and failure modes in SLAM systems. Dropout ratios range from zero (*light*) to severe point loss (up to 40% at *extreme*), consistent with signal-loss mechanisms such as beam divergence and atmospheric attenuation [16]. Gaussian noise levels ($\sigma \in [0.008, 0.060]$ m, i.e., subcentimetre to 6 cm) reflect the typical range-measurement uncertainty reported for both solid-state and mechanical LiDAR sensors under nominal and adverse conditions [2]. FoV reductions (up to 20% horizontal and 30% vertical of the full angular range) emulate partial angular occlusions caused by sensor housing contamination, road spray, or nearby structures. Occlusion patches remove contiguous local clusters representative of pedestrians, vegetation, or vehicle edges. Scan sparsification ($f_s \in [2, 8]$) reduces point density through systematic subsampling, approximating reduced angular-resolution modes available in configurable LiDAR sensors and undersampling effects observed in degraded scanning conditions. Motion distortion is applied exclusively at the *extreme* tier ($\|\mathbf{v}\| = 2.0$ m/s, $\|\boldsymbol{\omega}\| = 0.3$ rad/s) with per-point temporal offsets, providing a reproducible perturbation that significantly affects geometric alignment.

D. Augmentation (Simulation) Modules

The degradation modules implement the phenomenological parameters defined in the scenario configuration. Let a raw scan be $\mathcal{P} = \{(\mathbf{p}_i, \mathbf{a}_i, t_i)\}_{i=1}^{N}$, where $\mathbf{p}_i \in \mathbb{R}^3$ is a 3-D point, $\mathbf{a}_i$ its attributes, and t_i its time stamp.

1) *Point dropout:* Random point removal is applied through a Bernoulli mask

$$m_i \sim \text{Bernoulli}(1-r), \qquad \mathbf{p}'_i = \begin{cases} \mathbf{p}_i, & m_i = 1 \\ [-3pt]\varnothing, & m_i = 0 \end{cases}$$

where r is the dropout ratio. Structured variants remove contiguous angular or spatial regions to emulate realistic occluders.

2) *Occlusion and FoV reduction:* FoV cropping constrains points to angular limits in azimuth θ_i and elevation ϕ_i, computed as

$$\theta_i = \text{atan2}(y_i, x_i), \qquad \phi_i = \text{asin}(z_i/\|\mathbf{p}_i\|)$$

with cropping applied through bounds $|\theta_i| \le \Theta_{\max}$ and $|\phi_i| \le \Phi_{\max}$. Random occlusion removes K spherical patches of radius s, retaining points satisfying

$$\|\mathbf{p}_i - \mathbf{c}_k\| > s \quad \forall k$$

where s is the patch radius in metres, and occlusion centers $\mathbf{c}_k$ are uniformly sampled within a bounded range.

3) *Noise injection:* In real-time-of-flight LiDAR, beam directions are deterministically fixed by the scanning geometry; only the range is subject to uncertainty. Noise is therefore modeled as a scalar radial perturbation

$$\mathbf{p}'_i = \frac{\max(r_i + \varepsilon_i,\, r_{\min})}{r_i}\,\mathbf{p}_i, \quad r_i = \|\mathbf{p}_i\|$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ and $r_{\min} = 0.1$ m prevents degenerate negative-range points. Optional outlier noise (multipath, mixed-pixel) applies the same radial model with probability q and larger deviation $\sigma_o \gg \sigma$

$$\mathbf{p}''_i = \begin{cases} \frac{\max(r_i + \varepsilon_{o,i},\, r_{\min})}{r_i}\,\mathbf{p}_i, & \text{w.p. } q \\ [4pt]\mathbf{p}'_i, & \text{otherwise} \end{cases}$$

where $\varepsilon_{o,i} \sim \mathcal{N}(0, \sigma_o^2)$.

4) *Motion distortion:* Assume constant linear velocity $\mathbf{v}$ and angular velocity $\boldsymbol{\omega}$ over a scan

$$\mathbf{p}'_i = \mathbf{R}(\Delta t_i)\mathbf{p}_i + \mathbf{v}\Delta t_i, \qquad \Delta t_i = t_i - t_0.$$

The rigid motion follows an SE(3) constant-velocity model, where the rotational component is computed using the SO(3) exponential map approximated by its first-order expansion

$$\mathbf{R}(\Delta t_i) \approx \mathbf{I} + [\boldsymbol{\omega}\Delta t_i]_\times$$

applied for small angles $\|\boldsymbol{\omega}\Delta t_i\| < 0.1$ [17]. Parameters $\mathbf{v}$ and $\boldsymbol{\omega}$ are either estimated from synchronized IMU data, when available, or set to scenario-defined values (see Table I). The adopted constant-velocity model follows the scanwise motion approximation commonly used for LiDAR deskewing in tightly coupled LiDAR-inertial systems, such as FAST-LIO2 [18] and FASTER-LIO [19]. Because a LiDAR scan is acquired over a short interval (tens to hundreds of milliseconds), higher order motion terms, such as acceleration or jerk, provide limited additional realism while increasing model complexity. Therefore, the constant-velocity assumption provides a practical balance between physical interpretability, computational efficiency, and compatibility with real-time SLAM pipelines.

5) *Scan sparsification:* Systematic subsampling retains every f_sth point

$$\mathcal{P}' = \{(\mathbf{p}_i, \mathbf{a}_i, t_i) \mid i \equiv 0 \pmod{f_s}\}$$

modeling structured reductions in effective sampling density. This approximates reduced angular-resolution modes available in configurable LiDAR sensors (e.g., Ouster `lidar_mode`) and related undersampling effects in degraded sensing conditions. Unlike random dropout, sparsification preserves the regular scan acquisition pattern while reducing point density, enabling controlled evaluation of SLAM sensitivity to reduced sampling.

E. Integration and Reproducibility

All modules are parameterized through configuration files, with optional runtime overrides via ROS parameters. Sensor-specific message conversions ensure compatibility with downstream SLAM systems, and IMU measurements are interpolated to estimate velocities used in motion distortion. Deterministic module chaining, fixed random seeds, and diagnostic logging provide full reproducibility across experiments.

F. Implementation and Experimental Setup

The framework is implemented as a hybrid C++/Python ROS package: time-critical augmentation and I/O are handled in C++, while Python utilities support visualization and interactive parameter tuning. Although validated primarily on recorded rosbag data, the ROS-native design supports direct integration with live LiDAR streams.

Experiments were conducted using the multisensor sequences from our publicly available dataset GitHub repository [15],

TABLE II
BASELINE APE (MEAN ± STD, M) ON NONDEGRADED SEQUENCES

SLAM Methods	*IndoorOffice1*			*IndoorOffice2*			*OutdoorRoad*		
	Avia	**Mid360**	**Ouster**	**Avia**	**Mid360**	**Ouster**	**Avia**	**Mid360**	**Ouster**
FAST-LIO2	0.1436 ± 0.1390	0.0451 ± **0.0150**	0.0446 ± **0.0298**	0.3581 ± 0.1334	0.0412 ± 0.0201	0.0438 ± 0.0471	0.3755 ± 0.1527	0.3893 ± 0.1788	0.5845 ± 0.3127
FASTER-LIO	**0.1223** ± **0.0517**	0.0918 ± 0.0423	0.0571 ± 0.0304	0.2692 ± 0.1386	0.0460 ± **0.0147**	0.0549 ± 0.0491	**0.3013** ± **0.0818**	**0.3666** ± 0.1668	**0.4245** ± **0.2273**
S-FAST-LIO	0.1808 ± 0.1840	**0.0427** ± 0.0161	0.0541 ± 0.0367	**0.1019** ± **0.0536**	**0.0388** ± 0.0190	0.0504 ± 0.0529	0.6730 ± 0.3205	0.3721 ± **0.1641**	0.6223 ± 0.3282
GLIM	0.3104 ± 0.2461	0.1760 ± 0.1734	0.0733 ± 0.0395	1.1587 ± 0.6535	0.1559 ± 0.1142	0.0668 ± 0.0559	1.7026 ± 0.2774	0.6867 ± 0.5828	0.8303 ± 0.4314
FAST-LIO-SAM	0.1436 ± 0.1392	0.0457 ± 0.0153	**0.0442** ± 0.0299	0.3581 ± 0.1327	0.0400 ± 0.0190	**0.0431** ± **0.0398**	0.3700 ± 0.1545	0.3873 ± 0.1775	0.5795 ± 0.3137

For each lidar type, the highest accuracy within each data sequence is indicated in bold.

TABLE III
APE (MEAN ± STD, M) ACROSS SEVERITY LEVELS FOR INDOOROFFICE1, INDOOROFFICE2, AND OUTDOORROAD SEQUENCES (L = LIGHT, M = MODERATE, H = HEAVY, X = EXTREME)

SLAM Methods	**Avia_10Hz**				**Mid360_10Hz**				**Ouster_10Hz**			
	L	M	H	X	L	M	H	X	L	M	H	X
IndoorOffice1												
FAST-LIO2	0.2048 ± 0.1916	0.1650 ± 0.1269	0.6808 ± 0.7716	**0.2610** ± **0.1519**	0.0452 ± **0.0166**	0.0454 ± 0.0227	0.0468 ± 0.0233	0.0502 ± 0.0251	**0.0609** ± 0.0424	**0.0651** ± **0.0440**	0.0782 ± 0.0507	0.1486 ± 0.0879
FASTER-LIO	**0.0897** ± **0.0521**	**0.1322** ± **0.0420**	**0.1487** ± **0.0976**	1.2334 ± 0.4576	0.0936 ± 0.0415	0.1037 ± 0.0526	0.0996 ± 0.0422	0.0802 ± 0.0396	0.0828 ± **0.0405**	0.1000 ± 0.0503	0.1244 ± 0.0543	0.1647 ± **0.0632**
S-FAST-LIO	0.2290 ± 0.2891	0.2039 ± 0.1007	0.4159 ± 0.5800	0.7413 ± 0.4395	**0.0411** ± 0.0193	**0.0424** ± **0.0208**	**0.0434** ± 0.0239	0.0475 ± 0.0260	0.0727 ± 0.0502	0.0740 ± 0.0512	0.0762 ± 0.0506	**0.1409** ± 0.0687
GLIM	0.4735 ± 0.4462	0.2603 ± 0.2275	1.2602 ± 0.2980	4.3631 ± 2.4327	0.1124 ± 0.1790	0.1146 ± 0.1807	0.1226 ± 0.1793	0.1178 ± 0.1804	0.0942 ± 0.0536	0.0957 ± 0.0539	0.0950 ± 0.0557	0.1598 ± 0.0841
FAST-LIO-SAM	0.2207 ± 0.1415	0.3857 ± 0.5869	0.4874 ± 0.4540	0.3723 ± 0.4799	0.0434 ± 0.0171	0.0440 ± 0.0213	0.0468 ± **0.0208**	**0.0473** ± **0.0228**	0.0656 ± 0.0435	0.0746 ± 0.0516	**0.0734** ± **0.0465**	0.1538 ± 0.0734
IndoorOffice2												
FAST-LIO2	0.1074 ± 0.0625	**0.0934** ± 0.0988	**0.2608** ± 0.3289	**0.2284** ± **0.2475**	0.0383 ± **0.0171**	**0.0361** ± 0.0243	0.0466 ± 0.0294	0.0496 ± 0.0265	**0.0553** ± **0.0530**	0.0626 ± 0.0571	0.0725 ± 0.0592	0.1479 ± **0.0742**
FASTER-LIO	0.1081 ± 0.0980	0.1629 ± **0.0460**	0.3561 ± **0.0826**	3.3806 ± 1.1507	0.0464 ± 0.0211	0.0543 ± **0.0222**	0.0493 ± 0.0253	0.0526 ± **0.0250**	0.0705 ± 0.0569	0.0774 ± 0.0617	0.0799 ± **0.0584**	0.1651 ± 0.0665
S-FAST-LIO	**0.0972** ± **0.0422**	0.1272 ± 0.1005	0.3670 ± 0.4698	1.0495 ± 0.3538	**0.0379** ± 0.0269	0.0376 ± 0.0245	**0.0438** ± 0.0276	**0.0479** ± 0.0270	0.0655 ± 0.0606	**0.0664** ± 0.0589	0.0708 ± 0.0586	**0.1371** ± 0.0705
GLIM	0.2494 ± 0.1956	0.3329 ± 0.3144	8.3036 ± 3.5148	8.4788 ± 3.5708	0.0846 ± 0.1168	0.0875 ± 0.1287	0.0858 ± 0.1214	0.0936 ± 0.1494	0.0838 ± 0.0701	0.0847 ± 0.0734	0.0857 ± 0.0752	0.1620 ± 0.0868
FAST-LIO-SAM	0.1066 ± 0.1046	0.9495 ± 0.3636	1.0864 ± 0.8129	0.3347 ± 0.3582	0.0417 ± 0.0264	0.0426 ± 0.0261	0.0439 ± **0.0236**	0.0492 ± 0.0288	0.0623 ± 0.0614	0.0674 ± **0.0583**	**0.0680** ± 0.0616	0.1552 ± 0.0712
OutdoorRoad												
FAST-LIO2	0.3398 ± 0.1199	0.4589 ± 0.1957	**0.3186** ± 0.1685	0.6337 ± 0.3142	0.3895 ± 0.1812	**0.3761** ± **0.1694**	0.3697 ± **0.1579**	0.7942 ± 0.3402	0.6197 ± 0.3343	0.5668 ± 0.2975	0.6419 ± 0.3069	0.8968 ± 0.3874
FASTER-LIO	**0.3204** ± **0.1094**	**0.3259** ± **0.0996**	0.4730 ± 0.2607	**0.3313** ± **0.1254**	**0.2957** ± **0.1355**	0.4405 ± 0.1834	0.6490 ± 0.2569	**0.4214** ± **0.2053**	**0.4353** ± **0.2144**	0.5715 ± 0.6647	**0.4667** ± **0.2514**	1.0672 ± 0.4587
S-FAST-LIO	0.3984 ± 0.1752	0.5336 ± 0.2216	0.3536 ± **0.1624**	0.5206 ± 0.2709	0.4763 ± 0.5441	0.5031 ± 0.2178	**0.3682** ± 0.1851	0.6630 ± 0.3181	0.5959 ± 0.3138	0.5735 ± 0.2876	0.6752 ± 0.3162	0.9896 ± 0.3466
GLIM	0.5742 ± 0.2696	0.7187 ± 0.4969	2.5386 ± 0.7586	4.9507 ± 1.3130	0.6295 ± 0.6254	0.8520 ± 0.6880	1.4127 ± 0.9480	2.5318 ± 2.0589	0.8084 ± 0.4374	0.8134 ± 0.4324	0.8889 ± 0.5175	1.5412 ± 0.9320
FAST-LIO-SAM	0.3825 ± 0.1377	0.4419 ± 0.1762	0.5265 ± 0.5818	0.5304 ± 0.2325	0.3431 ± 0.1585	0.3884 ± 0.1711	0.6040 ± 0.2659	0.4343 ± 0.2137	0.6093 ± 0.3508	**0.5326** ± **0.2685**	0.6215 ± 0.3171	**0.7891** ± **0.3395**

which provides synchronized Livox Avia, Livox Mid-360, and Ouster OS0-128 point clouds across indoor and outdoor environments. The system was tested on a workstation running Ubuntu 20.04 with an AMD Ryzen 9 7945HX CPU and 32-GB RAM. Degradation settings were specified in YAML configuration files and automatically applied at runtime by the autonomous sensor configuration module (see Section II-B). To assess robustness under controlled degradations, five representative LiDAR odometry and SLAM systems, FAST-LIO2 [18], FASTER-LIO [19], S-FAST-LIO [20], FAST-LIO-SAM [21], and GLIM [22], were evaluated. These algorithms cover direct, iterative, and mapping-based approaches, enabling evaluation across diverse SLAM paradigms. The two indoor sequences differ in trajectory complexity: *IndoorOffice1* follows a compact loop through corridors and small rooms, while *IndoorOffice2* extends this with additional loops, revisited locations, and varied speeds, imposing greater drift accumulation and relocalization demands. *OutdoorRoad* traverses an open road with building facades, open spaces, and tree-lined segments, providing a geometrically richer but sparser scene context; trajectory ground truth visualizations are available in the dataset repository [15].

III. EXPERIMENTAL RESULTS

The implementation of the augmentation framework also provides a real-time visualization dashboard, described in detail in Section V.

A. *Quantitative SLAM Evaluation*

Quantitative robustness analysis was performed using the `evo_ape` tool[1] to compute absolute pose error (APE).

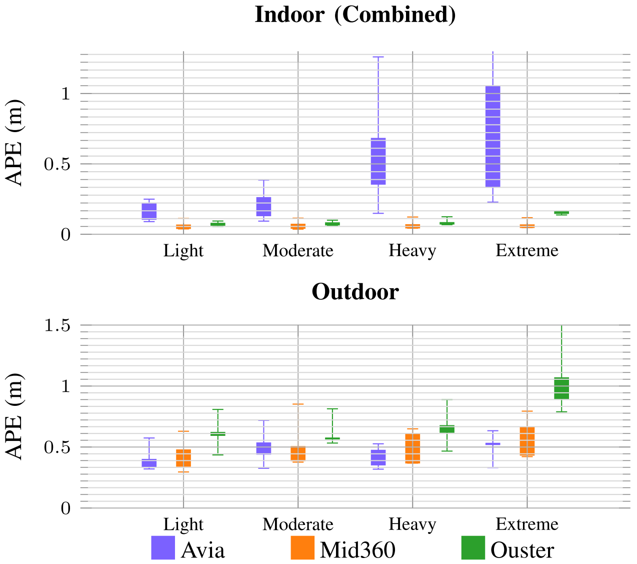

Fig. 2. Distribution of APE across degradation severities (*light–extreme*) for three LiDAR sensors (Avia, Mid360, and Ouster) in both indoor and outdoor settings.

Table II reports baseline APE on nondegraded sequences, while Table III summarizes results across the four severity tiers. The best performing entry per severity tier and sensor modality is highlighted in bold. Occasional cases, where moderate-tier APE is slightly lower than light-tier values, arise from stochastic variability in the SLAM pipeline rather than a systematic trend. Fig. 2 summarizes APE distributions for three LiDAR types

---

[1] [Online]. Available: https://github.com/MichaelGrupp/evo

in both indoor and outdoor environments. Mid-360 maintains consistently low error and variance across all severity levels, confirming strong robustness to the applied degradations across all SLAM back ends. Avia exhibits a threshold-driven failure mode, discussed in detail in Section III-B. Ouster presents intermediate behavior with gradually increasing spread toward higher severities. Overall, the results confirm that the proposed degradations provide controlled stress conditions for systematic SLAM robustness evaluation.

B. Discussion

Sensor-specific robustness patterns: Tables II and III, together with Fig. 2, reveal distinct resilience profiles across LiDAR architectures. Livox Mid-360 exhibits the strongest robustness, maintaining subdecimeter APE across all severities in indoor conditions; its nonrepetitive scanning pattern and uniform spatial coverage preserve feature distributions under dropout and occlusion. Livox Avia shows a *threshold-driven* failure mode: performance remains stable under *light* and *moderate* degradation but deteriorates sharply at *heavy* and *extreme* levels (e.g., FASTER-LIO: 0.09 m at *light* versus 1.23 m at *extreme*; GLIM: 0.47 m at *light* versus 4.36 m at *extreme*, IndoorOffice1), indicating sensitivity to density collapse. Ouster OS0-128 presents intermediate behavior with higher variance at extreme severities, suggesting that structured scanning patterns are more vulnerable to systematic occlusion.

Algorithm–sensor interaction effects: FAST-LIO2 and S-FAST-LIO maintain strong median accuracy but exhibit large worst case errors (up to $\sim$8.5 m at extreme indoor conditions) when dropout exceeds $\sim$35%, indicating reinitialization failures rather than gradual drift. GLIM yields higher baseline error but lower variance across severity tiers, implying that its global scan-to-map consistency is more tolerant to sparsity. FASTER-LIO demonstrates the most uniform behavior across sensors. FAST-LIO-SAM exhibits behavior closely tracking FAST-LIO2 across all severities, with marginally higher error in outdoor sequences due to additional loop-closure overhead under sparse point clouds.

Environmental context dependence: Indoor and outdoor conditions exhibit fundamentally different degradation responses. Indoor environments produce higher failure rates at extreme severity (GLIM/Avia: $\approx 9\times$; Mid-360: $\approx 1.1\times$), reflecting the reduced feature redundancy in confined spaces. Outdoor sequences exhibit higher absolute APE in nominal conditions but degrade more gradually; FAST-LIO2 with Mid-360 increases from 0.39 m (*light*) to 0.79 m (*extreme*), a $\approx 2\times$ increase, consistent with richer geometric structure at the cost of reduced point density at long range. These results emphasize that robustness is not scale-invariant; algorithms tuned for outdoor navigation may fail abruptly in degraded indoor environments.

IV. CONCLUSION

We presented a sensor-aware phenomenological framework for LiDAR degradation, enabling systematic and reproducible robustness evaluation of SLAM algorithms under adverse conditions. Unlike generic corruption benchmarks or simulation engines requiring detailed scene geometry, our approach operates directly on raw LiDAR measurements with real-time performance ($<$ 20 ms per frame), while preserving hardware-specific scan patterns and temporal consistency. The framework integrates autonomous topic and sensor detection with modular degradation models, including dropout, FoV reduction, noise, occlusion, and motion distortion, organized into four severity tiers (*light–extreme*) to provide physically interpretable cross-sensor stress testing.

Experimental evaluation across three LiDAR architectures and five SLAM systems revealed distinct robustness behaviors shaped by both sensor design and environmental context. Mid-360 consistently exhibited strong resilience, Avia showed threshold-driven failures under extreme degradation, and Ouster displayed intermediate sensitivity shaped by its structured scanning pattern. These results demonstrate that the proposed framework provides a practical and generalizable tool for evaluating LiDAR-based SLAM performance. It enables the community to analyze robustness trends, identify failure modes, and design degradation-aware perception systems for real-world deployment.

DESCRIPTION OF SUPPLEMENTARY MATERIALS

This work is accompanied by two supplementary materials: source code and recorded videos. The code is available both as an attachment to this article and in a public GitHub repository available online.[2] It is worth noting that the details of all the supplementary materials and their description can be found in the aforementioned GitHub repository.

A. Code Implementation

The framework is implemented in C++/Python on ROS Noetic and distributed as a Docker container for reproducible deployment. Time-critical components, including augmentation, sensor parsing, and ROS I/O, are implemented in C++ to ensure real-time performance, while Python utilities provide bird's-eye-view (BEV) visualization, online statistics, diagnostic logging, and video generation support. Detailed instructions for building and running the code are available in the GitHub repository.

Configuration and severity tiers: All degradation parameters are defined in the repository under `config/rosbag_test_config.yaml`. Four severity tiers, *light*, *moderate*, *heavy*, and *extreme*, map to the phenomenological models in Section II-D via structured namespaces (e.g., `augmentations.dropout.*`, `augmentations.noise.*`, `augmentations.fov.*`). These settings control dropout ratios, FoV limits, noise levels, occlusion masks, motion distortion velocities, and sparsification, and can be overridden at runtime using ROS parameters.

Integration and reproducibility: The augmentation node automatically discovers available LiDAR and IMU topics and applies sensor-specific parsing rules: Ouster fields (`x,y,z,`

---

[2] [Online]. Available: https://github.com/mawuto/lidar_augmentation_cpp_ws

`intensity, reflectivity, line`) are processed natively, while Livox data are converted between `PointCloud2` and `livox_ros_driver/CustomMsg` when required by downstream SLAM systems. IMU buffers are synchronized using trapezoidal integration to estimate per-point velocities for motion distortion. For real-time use, both IMU messages and augmented LiDAR scans are republished through a configurable fixed-delay buffer (default: 30 ms), ensuring that downstream LIO/SLAM systems receive time-aligned streams despite variable processing latency.

Real-time monitoring and performance: The framework includes a real-time monitoring dashboard. It processes multiple sensors concurrently with augmentation times below 20 ms per frame (Ouster: 19.8 ms; Avia: 5.4 ms; Mid-360: 4.1 ms), ensuring compatibility with 10-Hz operation. The autonomous configuration module eliminates manual setup, enabling scalable multisensor deployments in both offline and real-time settings.

B. Video Description

The supplementary video demonstrates the framework under real-time adverse augmentation, showing the original point cloud in green and the augmented output in red, together with both the BEV visualization and the monitoring dashboard. The dashboard presents processing time, point-count statistics, and the active augmentation settings for each LiDAR stream. The BEV is intended for qualitative inspection and accumulates point clouds before rendering, making it less dynamic than the dashboard display. Although the augmented cloud contains significantly fewer points, the visualization renders a fixed number of samples. Because the remaining points are concentrated within a reduced field of view, they may appear visually denser despite the substantial reduction in the underlying point count. The BEV and dashboard elements in the video are included only to illustrate the system operation; when the package is executed locally, they are displayed at full resolution, and additional screenshots are provided in the GitHub repository. The primary purpose of the video is, therefore, to highlight the visual effect of the LiDAR augmentation on the point cloud itself.

REFERENCES

[1] V. Pritzl, X. Yu, T. Westerlund, P. Štěpán, and M. Saska, "Degradation-aware cooperative multi-modal GNSS-denied localization leveraging LiDAR-based robot detections," 2025, *arXiv:2510.20480*.

[2] M. Bijelic et al., "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11679–11689.

[3] Y. Zhang, A. Carballo, H. Yang, and K. Takeda, "Perception and sensing for autonomous vehicles under adverse weather conditions: A survey," *ISPRS J. Photogrammetry Remote Sens.*, vol. 196, pp. 146–177, 2023.

[4] Q. Wu et al., "UA-LIO: An uncertainty-aware LiDAR-inertial odometry for autonomous driving in urban environments," *IEEE Trans. Instrum. Meas.*, vol. 74, 2025, Art. no. 8502912.

[5] M. Dreissig, D. Scheuble, F. Piewak, and J. Boedecker, "Survey on LiDAR perception in adverse weather conditions," in *Proc. IEEE Intell. Veh. Symp.*, 2023, pp. 1–8.

[6] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 1–16.

[7] S. Manivasagam et al., "LiDARsim: Realistic LiDAR simulation by leveraging the real world," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11167–11176.

[8] S. Li, Z. Wang, F. Juefei-Xu, Q. Guo, X. Li, and L. Ma, "Common corruption robustness of point cloud detectors: Benchmark and enhancement," *IEEE Trans. Multimedia*, vol. 27, pp. 848–859, 2025.

[9] Y. Dong et al., "Benchmarking robustness of 3D object detection to common corruptions in autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 1022–1032.

[10] B. Yang, T. M. T. Pham, and J. Yang, "Evaluating and improving the robustness of LiDAR-based localization and mapping," 2024, *arXiv.2409.10824*.

[11] H. Sier, Q. Li, X. Yu, J. Peña Queralta, Z. Zou, and T. Westerlund, "A benchmark for multi-modal LiDAR SLAM with ground truth in GNSS-denied environments," *Remote Sens.*, vol. 15, no. 13, 2023, Art. no. 3314.

[12] A. Xiao, J. Huang, D. Guan, K. Cui, S. Lu, and L. Shao, "Polarmix: A general data augmentation technique for LiDAR point clouds," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 21605–21618.

[13] P. Šebek, Š. Pokorný, P. Vacek, and T. Svoboda, "Real3D-Aug: Point cloud augmentation by placing real objects with occlusion handling for 3D detection and segmentation," 2023, *arXiv:2206.07634*.

[14] J. Zhan, T. Liu, R. Li, J. Zhang, Z. Zhang, and Y. Chen, "Real-Aug: Realistic scene synthesis for LiDAR augmentation in 3D object detection," 2023, *arXiv:2305.12853*.

[15] D. M. K. Felix, X. Yu, J. Zhang, S. Ha, Z. Zou, and T. Westerlund, "Understanding LiDAR variability: A dataset and comparative study featuring dome-shaped, solid-state, and spinning LiDARs," *IEEE Robot. Automat. Lett.*, vol. 11, no. 1, pp. 570–577, Jan. 2026.

[16] C. Weitkamp *LiDAR: Range-Resolved Optical Remote Sensing of the Atmosphere*, vol. 102, 1st ed. New York, NY, USA: Springer, 2005.

[17] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[18] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[19] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-LIO: Lightweight tightly coupled LiDAR-inertial odometry using parallel sparse incremental voxels," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.

[20] Z. Wang, "S-FAST-LIO: A simplified implementation of FAST-LIO," 2023. Accessed: Nov. 27, 2025. [Online]. Available: https://github.com/zlwang7/S-FAST_LIO

[21] M. L. Eungchang and C. Daniele, "Fast-LIO-SAM: A SLAM implementation combining FAST-LIO2 with pose graph optimization and loop closing based on LIO-SAM," 2023. Accessed: Mar. 7, 2026. [Online]. Available: https://github.com/engcang/FAST-LIO-SAM

[22] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "GLIM: 3D range-inertial localization and mapping with GPU-accelerated scan matching factors," *Robot. Auton. Syst.*, vol. 179, 2024, Art. no. 104750.