DOI: 10.2195/lj_proc_knitt_en_202510_01

Synthetic Datasets for Data-Driven Localization Monitoring

Markus Knitt, Sean Maroofi, Manav Thakkar, Hendrik Rose, Philipp Braun

Institute of Logistics Engineering
Hamburg University of Technology

Reliable self-localization is fundamental to safe and efficient navigation in autonomous mobile robots and driverless industrial trucks. However, localization failures in highly dynamic or feature-poor environments can lead to safety hazards and costly workflow disruptions. While probabilistic methods such as particle filters mitigate sensing and actuation uncertainties, they lack mechanisms to recognize impending failures. To address this gap, we propose a systematic, physics-based simulation methodology for generating datasets that enable predictive failure detection. The datasets include localization estimates, ground-truth poses, sensor data, and automatically labeled failure cases. By systematically introducing challenging conditions, such as dynamic obstacles, featureless areas, and map ambiguities, we provoke diverse failure modes in a reproducible manner. These datasets establish a scalable foundation for training models that anticipate localization failures, supporting proactive fault detection and enhancing the safety and reliability of autonomous navigation in complex environments.

[Keywords: self-localization, predictive monitoring, data-driven modeling, robotics, machine learning]

1 INTRODUCTION

Localization is a fundamental prerequisite for the navigation of autonomous mobile robots and driverless industrial trucks because all higher-level navigation tasks, e.g., path planning, obstacle avoidance, and mission execution, depend on an accurate estimate of the vehicle's position and orientation in its environment. Without reliable localization, the robot cannot know where it currently is, which makes it impossible to determine how to reach its target safely and efficiently. Different approaches for robot localization exist and are compared in a systematic review in [1]. Self-localization is the process of determining a system's position without relying on external infrastructure. Self-localization is the process of determining a system's position without relying on external infrastructure and is commonly performed using dead reckoning based on odometry sources, such as wheel encoders or an Inertial Measurement Unit (IMU), or by employing range sensors, including Light Detecting and Ranging (LiDAR) or ultrasonic sensors, as well as camera-based methods. These approaches are common but not exhaustive, as alternative strategies exploiting other sensing modalities or data-driven techniques are also being investigated. During operation, malfunctions in the localization system can affect the overall behavior of the mobile robot, potentially jeopardizing operational safety or workflow continuity. As Thrun et al. note, robotic operating environments are "inherently unpredictable" [2, p. 3]. Probabilistic approaches such as particle filters have become standard in self-localization, as they explicitly account for uncertainties in actuation and sensing.

However, there are particularly challenging scenarios in which localization may fail. These localization failures can occur, for example, due to occlusions in the LiDAR scan or noise caused by rain or challenging lighting conditions [3]. Highly dynamic environments, such as public spaces or inbound and outbound areas of warehouses, can obscure static features and degrade visual odometry. Odometry based on wheel encoders can be inaccurate due to discrepancies between actual and modeled dimensions, incorrect encoder readings, and wheel slip. Furthermore, map ambiguities and featureless areas pose additional challenges.

While uncertainty in automation-compatible environments is typically low, many real-world environments exhibit greater levels of uncertainty that must be addressed to ensure reliable localization. One way of addressing these challenges is by detecting localization failures. To safely operate in such environments, AMRs are required in specified accuracy boundaries. Identifying these boundaries as localization failures allows the robot to perform error correction behavior, for example, disregarding faulty sensor readings or correcting localization estimates [4]. Different approaches exist for monitoring the integrity of localization systems. Especially for detecting these failures before they occur, data-driven modeling has shown potential [5]. Using data-driven models to forecast localization failures is not a widely explored research topic, and publicly available datasets or methods for creating datasets are scarce.

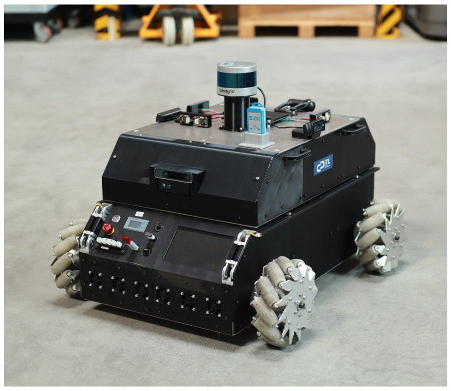
Predictive models, capable of anticipating localization failures before they occur, can improve the safety and reliability of autonomous navigation. However, predictive modeling is hindered by the scarcity of datasets containing failure examples. Existing localization benchmarking datasets often lack such instances, and forced failure scenarios used in some studies are rarely shared or fully documented, limiting reproducibility. Furthermore, many current approaches are restricted to domains like autonomous driving or camera-based systems, and their datasets, methods, and models cannot be directly applied to particle filter-based localization approaches using a LiDAR sensor.

Recording datasets containing localization failures in real-world testbeds presents several challenges. Safety concerns arise when intentionally inducing failure modes in robotic systems, posing risks to both the environment and the robot. Moreover, the manual setup and supervision required for such experiments demand substantial effort, particularly when simulating realistic scenarios involving dynamic obstacles such as pedestrians or moving objects. Access to test environments is often limited due to constraints related to time, space, and concurrent activities at the facility. Furthermore, infrastructure requirements, such as motion capture systems used to provide ground truth localization, can limit the scalability of data collection. Motion capture systems are essential for identifying instances of localization failure, but are costly and not always available.

To address the challenges of detecting and predicting localization failures, this work makes the following contributions: (1) we introduce a systematic process for generating datasets to train localization monitoring models for LiDAR-based ground robots, with open-source code and documentation[1]; (2) we release publicly available, physics-based simulation environments tailored for dataset generation, including a model of the omni-directional robot *MoMo* (cf. Figure 1) [6]; (3) we propose a domain randomization technique that injects artificial odometry drift to provoke diverse failure modes; and (4) we provide an open dataset for developing and evaluating predictive models of localization failure in mobile ground robots [6].

This study focuses on systematically inducing localization failures, collecting corresponding data, and automatically labeling samples. Section 2 reviews related work, Sections 3 and 4 detail the methodology, and Section 5 presents the resulting dataset. Feature extraction, learning algorithm design, and model deployment are beyond the scope of this paper.

[1]https://flowcean.me/examples/robot_localization_failure/


(a) Robot at the real-world testbed

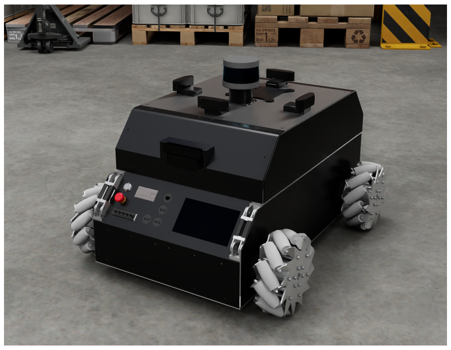

(b) Robot in simulation

Figure 1: Comparison of the MoMo robot in real-world and simulated environments.

2 Related work

Detecting, understanding, and anticipating failures in robot localization systems is critical for ensuring safe and reliable autonomous operation. Over the past years, a variety of approaches have been proposed to detect or mitigate such failures, ranging from traditional model-based techniques to modern data-driven methods. This section reviews existing literature on fault detection and predictive monitoring of localization systems, with a particular focus on the availability of datasets, the applicability of different approaches to ground robots, and the limitations that motivate the present study.

Maharmeh et al. [4] survey integrity concepts such as localization fault detection, which monitor the current state of the localization system. The model-based approaches in the study either detect localization failures based on sensor readings, without accounting for potential issues inherent to the localization algorithm itself, or other approaches included in the study detect localization failures only after they have occurred [7]. At that point, the robot may have already entered an unrecoverable state or become an obstacle to faster-moving agents in its environment.

Another approach is using predictive models that anticipate localization failures before they occur, leveraging modern data-driven techniques[5], [8], [9]. By forecasting potential localization faults in advance, such models have the potential to enhance the reliability and safety of autonomous navigation in complex environments. While several approaches explore data-driven monitoring of localization systems, many are focused on autonomous road vehicles [5], [8], limited to camera-based localization [9], or do not evaluate the predictive performance of their models for future localization states [10].

Developing effective predictive models requires training data that includes examples of localization system failures, i.e., instances in which the estimated pose diverges beyond acceptable error bounds. However, such data is scarce, especially for ground robots. Although datasets exist for benchmarking localization in autonomous driving [11], [12] and RGB-D-based SLAM systems [13], they generally lack annotated failure instances. It could be possible to use the existing benchmarking datasets to test localization algorithms and collect failure cases, but this would likely result in a dataset with very few failures. As a result, studies that address predictive localization monitoring often intentionally induce failures by designing scenarios where the localization system is bound to fail [9], [10]. Although prior research has laid a foundation for fault detection and data-driven monitoring of localization systems, significant gaps remain in the availability of failure-rich datasets and the development of reproducible methods for predictive localization monitoring, particularly for ground robots in diverse environments.

We address these gaps by creating datasets specifically designed for training models that enable the prediction of localization failures in mobile ground robots. Our approach focuses on generating diverse and reproducible failure scenarios in simulation, allowing for controlled data collection without the safety and logistical constraints of real-world experimentation. By inducing failures, systematic domain randomization, and annotating failure instances automatically, we provide a scalable and extensible approach for developing and evaluating data-driven models that forecast failures in robot localization systems. In doing so, this work contributes both the methodology and the data necessary to advance research in proactive fault detection for ground robot localization.

3 SIMULATION SETUP

To generate a diverse and realistic dataset for training and evaluating predictive localization models, a comprehensive simulation setup is created using NVIDIA Isaac Sim. This setup replicates the physical behavior and sensor outputs of a real mobile robot operating in dynamic environments. The simulation framework ensures high fidelity through the use of an accurate robot model, configurable scenarios, and reproducible sensor conditions. The following subsections describe the robot configuration, the simulated environment, and the intentional challenges introduced to provoke localization failures.

3.1 ROBOT SETUP

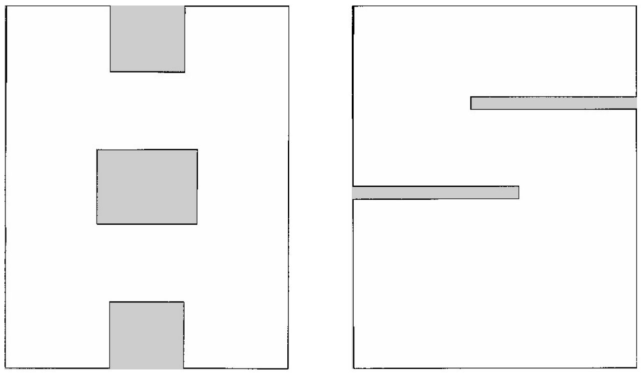
This paper focuses on the commonly used particle filter-based approach Adaptive Monte Carlo Localization (AMCL) [14] in combination with a LiDAR and artificially worsened odometry. The robot used in this study is the open-source omnidirectional mobile robot *MoMo* [15]. For the dataset generation presented in this study, MoMo is equipped with a VLP-16 Velodyne LiDAR, which provides data used for precise localization and state estimation. A digital model of MoMo is created in the Isaac Sim environment, accurately modeling both the robot's physical dimensions and its holonomic kinematics. Figure 1 shows the robot MoMo at the real-world testbed (Fig. 1a) and the simulated robot in Isaac Sim (Fig. 1b).

Following [16], the rollers of the mecanum wheels are modeled to replicate omnidirectional behavior realistically. To interface with standard robotics middleware, LiDAR scans and odometry are published to the ROS ecosystem using NVIDIA's Omnigraph system. This setup ensures consistent behavior between simulated and real-world conditions. While a sim-to-real gap is inevitable due to sensor noise, unmodeled dynamics, and environmental variation, we aim to minimize it by generating diverse datasets across a range of challenging scenarios. To support reproducibility and further research, the Universal Scene Description (USD) file of the simulation is publicly provided.

3.2 SIMULATION ENVIRONMENT

Simulation is used in this study as a safe, flexible, and reproducible method for generating data to train and evaluate predictive models of localization failures. Collecting real-world data that intentionally includes such failures is often impractical due to safety concerns, resource constraints, and the scarcity of datasets containing a sufficient number of failure cases. Simulated environments enable controlled experimentation under a wide range of conditions, including the introduction of artificial noise, dynamic and static obstacles, and various forms of odometry drift. This capability allows for the systematic creation of diverse failure scenarios, which are essential for developing robust, data-driven forecasting models for robot localization systems.

The simulation experiments are conducted in six simple, small-scale environments and one larger, prebuilt warehouse environment provided by Isaac Sim. The warehouse setting is selected for its realistic representation of a typical warehouse, complete with aisles, storage racks, boxes, and common handling equipment. Its symmetric layout and



(a) Symmetric map *(b) Asymmetric map*

Figure 2: Examples of symmetric and asymmetric occupancy grid maps.

large open spaces mirror characteristics found in real-world warehouses, where the repetitive structure of aisles and shelves can cause ambiguity in pose estimation and pose a challenge for standard localization methods. This symmetry is deliberately exploited to induce localization confusion, making the environment particularly well-suited for generating data aimed at predicting self-localization failures.

To broaden the range of symmetric layouts, three of the simple environments are designed with symmetric structures (Figure 2a), while the remaining three are asymmetric (Figure 2b). Using these standardized environments ensures reproducibility, allowing other researchers to replicate the presented experiments and extend the work without requiring extensive manual setup of the environment.

3.3 Environmental challenges

To capture realistic scenarios that are prone to localization failures, several environmental challenges are intentionally introduced into the simulation environment. Dynamic obstacles are introduced using spherical objects that represent typical moving objects, such as humans, robots, or forklifts, encountered in warehouse environments (cf. Figure 3). Since localization failures primarily arise from occlusions in the laser scan rather than the exact geometry of the obstructing object, the specific shape of the obstacles is not critical. Spheres are used due to their computational efficiency and the simplicity of collision handling, while different sizes allow for approximating a variety of moving objects. The spheres follow randomized trajectories at speeds ranging from 0.0 m/s to 5.0 m/s, simulating unpredictable motion patterns. For the current dataset, 35 such spheres are deployed. To ensure safe interaction during collisions, the spheres are assigned a low density, allowing them to be deflected upon contact without impacting the motion of the robot. Despite minimal physical impact, the spheres remain fully detectable by the LiDAR sensor, introducing

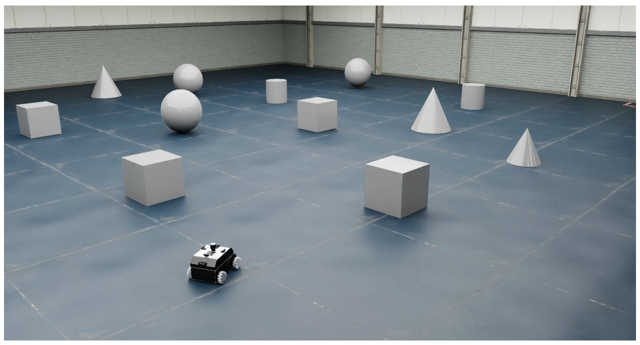


Figure 3: Robot and obstacles in a warehouse environment.

perception and localization complexity. Additionally, several static obstacles, cubes, cylinders, cones, and spheres of varying sizes, are manually placed throughout the environment. The obstacles are not a part of the navigational map of the robot, simulating scenarios where unexpected obstructions occur in operational environments. Their presence challenges the robot's perception and localization systems, making the generated data particularly valuable for training predictive failure models.

Real-world odometry estimates contain drift due to sensor inaccuracies, mechanical imperfections, and dynamic movements. The amount of drift in each direction differs from robot to robot. Using the odometry of only one robot for dataset generation could lead to poor generalizability when deploying on a different robot, as the model may learn patterns in the odometry that are specific to that robot. For this reason, a domain randomization approach for the odometry was followed. To include realistic behavior of as many robots as possible, the ground truth odometry provided by the Isaac Sim was systematically altered by introducing controlled drift. The drift settings are randomized throughout the experiments. By generating imperfect odometry data closely resembling real-world sensor outputs, the datasets enable robust evaluation and enhancement of predictive models intended for early detection of localization failures.

In contrast, the LiDAR sensor data is not randomized in this simulation. This decision allows us to isolate the effects of environmental challenges and odometry drift on localization failures, without introducing additional variability from LiDAR-specific noise, e.g., range inaccuracies or point cloud distortions. Real-world LiDAR sensors like the VLP-16 exhibit relatively low and consistent noise levels compared to odometry sources, and the primary failure modes in this study stem from occlusions and map-related issues rather than intrinsic sensor errors. Future extensions could incorporate LiDAR noise randomization to further narrow the sim-to-real gap.

Figure 4 illustrates the artificial odometry drift. To com-

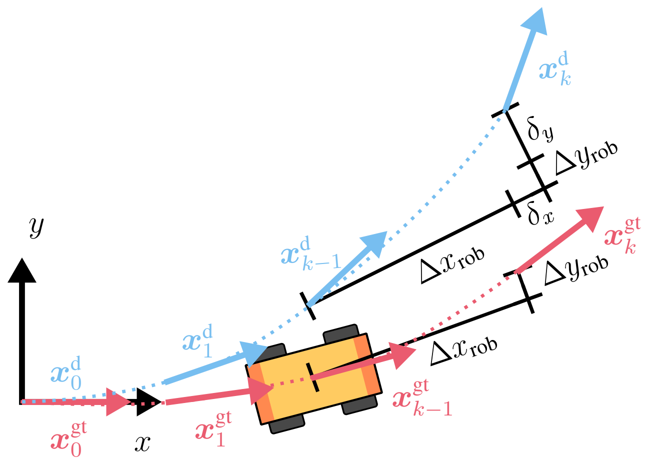

y
x
$\boldsymbol{x}_0^{\mathrm{d}}$
$\boldsymbol{x}_1^{\mathrm{d}}$
$\boldsymbol{x}_{k-1}^{\mathrm{d}}$
$\boldsymbol{x}_k^{\mathrm{d}}$
δ_y
Δy_{rob}
δ_x
Δx_{rob}
$\boldsymbol{x}_k^{\mathrm{gt}}$
Δy_{rob}
Δx_{rob}
$\boldsymbol{x}_{k-1}^{\mathrm{gt}}$
$\boldsymbol{x}_0^{\mathrm{gt}}$
$\boldsymbol{x}_1^{\mathrm{gt}}$

Figure 4: Artificial Odometry drift.

pute the drifted pose $\mathbf{x}_k^{\mathrm{d}} = \begin{bmatrix} x_k^{\mathrm{d}} & y_k^{\mathrm{d}} & \theta_k^{\mathrm{d}} \end{bmatrix}^T$ at time step k, we start with the ground truth poses $\mathbf{x}_{k-1}^{\mathrm{gt}}$ and $\mathbf{x}_k^{\mathrm{gt}}$, and the previous drifted pose $\mathbf{x}_{k-1}^{\mathrm{d}}$. The drift parameters are ϵ_x, ϵ_y, and ϵ_θ, representing the drift per meter in the local x-direction, y-direction, and the change in yaw angle per meter traveled in the local x-direction, respectively. The rotation matrix for transforming coordinates is defined as:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \tag{1}$$

where $\mathbf{R}(\theta)^T$ converts from global to local coordinates. The drifted pose is computed through the following steps: First, the displacement in the global frame is calculated as the difference between consecutive ground truth positions:

$$\Delta\mathbf{p}_{\mathrm{glob}}^{\mathrm{gt}} = \begin{bmatrix} x_k^{\mathrm{gt}} - x_{k-1}^{\mathrm{gt}} \\ y_k^{\mathrm{gt}} - y_{k-1}^{\mathrm{gt}} \end{bmatrix}. \tag{2}$$

This global displacement is transformed into the local frame of the previous ground truth orientation $\theta_{k-1}^{\mathrm{gt}}$ to align with the robot's coordinate system:

$$\Delta\mathbf{p}_{\mathrm{rob}}^{\mathrm{gt}} = \mathbf{R}(\theta_{k-1}^{\mathrm{gt}})^T \Delta\mathbf{p}_{\mathrm{glob}}^{\mathrm{gt}} = \begin{bmatrix} \Delta x_{\mathrm{rob}} \\ \Delta y_{\mathrm{rob}} \end{bmatrix}. \tag{3}$$

Note that $\Delta y_{\mathrm{rob}} = 0$ for non-holonomic robots. The drift amounts δ_x, δ_y, δ_θ are defined proportional to the local displacements Δx_{rob} and Δy_{rob}, scaled by the drift parameters in x direction ϵ_x, ϵ_y, ϵ_θ and the drift parameters in y direction ζ_x, ζ_y, ζ_θ:

$$\delta_x = \epsilon_x \cdot \Delta x_{\mathrm{rob}} + \zeta_x \cdot \Delta y_{\mathrm{rob}}, \tag{4}$$

$$\delta_y = \epsilon_y \cdot \Delta x_{\mathrm{rob}} + \zeta_y \cdot \Delta y_{\mathrm{rob}}, \tag{5}$$

$$\delta_\theta = \epsilon_\theta \cdot \Delta x_{\mathrm{rob}} + \zeta_\theta \cdot \Delta y_{\mathrm{rob}}. \tag{6}$$

The local displacement is then adjusted by applying the drift, scaling the displacements to simulate sensor inaccuracies:

$$\Delta\mathbf{p}_{\mathrm{rob}}^{\mathrm{d}} = \begin{bmatrix} \Delta x_{\mathrm{rob}} + \delta_x \\ \Delta y_{\mathrm{rob}} + \delta_y \end{bmatrix}. \tag{7}$$

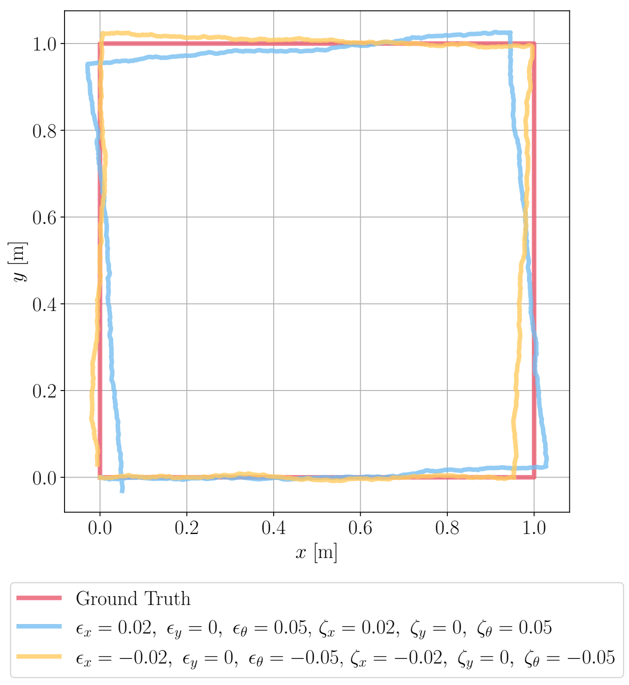


Figure 5: Comparison of the ground truth trajectory (red) with artificially drifted odometry data. Two drift models are shown: one with positive drift parameters (blue) and one with negative drift parameters (yellow).

The drifted local displacement is transformed back to the global frame using the previous drifted orientation $\theta_{k-1}^{\mathrm{d}}$:

$$\Delta\mathbf{p}_{\mathrm{glob}}^{\mathrm{d}} = \mathbf{R}(\theta_{k-1}^{\mathrm{d}})\Delta\mathbf{p}_{\mathrm{rob}}^{\mathrm{d}}. \tag{8}$$

The drifted position is updated by adding the drifted global displacement to the previous drifted position:

$$\begin{bmatrix} x_k^{\mathrm{d}} \\ y_k^{\mathrm{d}} \end{bmatrix} = \begin{bmatrix} x_{k-1}^{\mathrm{d}} \\ y_{k-1}^{\mathrm{d}} \end{bmatrix} + \Delta\mathbf{p}_{\mathrm{glob}}^{\mathrm{d}}. \tag{9}$$

The change in ground truth orientation is computed as the difference between consecutive yaw angles:

$$\Delta\theta^{\mathrm{gt}} = \theta_k^{\mathrm{gt}} - \theta_{k-1}^{\mathrm{gt}}. \tag{10}$$

The drifted orientation is updated by combining the ground truth orientation change with the drift:

$$\theta_k^{\mathrm{d}} = \theta_{k-1}^{\mathrm{d}} + \Delta\theta^{\mathrm{gt}} + \delta_\theta. \tag{11}$$

For initialization $k = 0$ (e.g., when it is the first message), the ground truth equals the drifted pose:

$$\mathbf{x}_0^{\mathrm{d}} = \mathbf{x}_0^{\mathrm{gt}}. \tag{12}$$

Additionally, Gaussian noise is added to the drifted trajectories. Figure 5 shows the effect of adding drift to a stream of ground truth poses for different ϵ_x, ϵ_y, and ϵ_θ. In the figure, a Gaussian noise is applied with a mean of $\mu = 0$m and a standard deviation of $\sigma = 0.001$m.

The simulation environment thus provides a scalable and reproducible means of exposing a mobile robot to a diverse set of realistic and challenging conditions that are known to trigger localization errors. By carefully designing scenarios that include symmetry, dynamic obstacles, unexpected occlusions, and imperfect odometry, the simulation ensures that failure cases occur in a controlled yet varied manner. This setup is essential for generating meaningful datasets that support the development of robust, predictive localization monitoring models. The following section details how these datasets are created, structured, and automatically labeled to facilitate data-driven modeling.

4 Dataset creation

To support the development of predictive localization monitoring systems, this work generates a dataset containing both successful and failed localization episodes. The dataset is produced entirely in simulation to ensure safety, reproducibility, and scalability. It includes diverse scenarios with varying dynamic elements and controlled odometry drift. This section outlines the data collection pipeline, the dataset structure, and the method used for automatically labeling localization failures. Both a digital model of the robot and multiple simulation environments are used within Isaac Sim to simulate mobile navigation in cluttered and dynamic scenes. The robot follows a trajectory consisting of an arbitrary number of randomized goals scattered throughout the environment. Localization failures are induced by moving spheres that traverse the warehouse, occluding the robot's sensors during navigation. These spheres are divided into three size categories based on their *effective diameter* d_{eff} at the LiDAR height z (cf. Figure 6). The effective diameter d_{eff} is calculated as

$$d_{\text{eff}}(z) = 2 \cdot \sqrt{\left(\frac{d}{2}\right)^2 - \left(z - \frac{d}{2}\right)^2}, \tag{13}$$

where d is the full sphere diameter and z is the LiDAR height measured from the bottom of the sphere. Small spheres, numbering fifteen, have d_{eff} in the range of 0.2 m to 0.5 m to simulate humans and small robots; medium spheres, also fifteen in number, have d_{eff} between 0.5 m to 1.0 m to represent large robots and pallet trucks; and large spheres, totaling five, have d_{eff} ranging from 1.0 m to 1.5 m to model forklift trucks. This variety of obstacle sizes ensures that the dataset captures a realistic range of occlusion scenarios affecting sensor visibility.

4.1 Data collection process

Recording training datasets that include localization failures requires a structured and repeatable process to ensure consistency, reliability, and meaningful data for supervised learning. Applying the framework introduced in [17], robot navigation and localization quality evaluation are orchestrated using a hierarchical finite state machine. Figure 7 depicts the procedure for collecting training samples. A single sample lasts from the robot's initialization to the point of localization failure. In *generate random goal poses*, the randomized trajectory is set. In the *localization* block, the robot receives an initial ground truth pose provided by Isaac Sim, which AMCL uses to estimate the pose of the robot. After convergence, the robot starts the point-to-point *navigation*. Simultaneously, the quality of the pose estimation of AMCL is evaluated. The data is collected in sequences. If the position or orientation error exceeds the respective reset thresholds, α_{reset} or β_{reset}, the current sequence is terminated. This reset procedure is employed to prevent AMCL from entering a state from which it cannot converge. During the reset, the robot halts and AMCL is reinitialized with an initial pose derived from ground-truth data. In addition, the drift parameters ϵ_x, ϵ_y, ϵ_θ, ζ_x, ζ_y, and ζ_θ (see

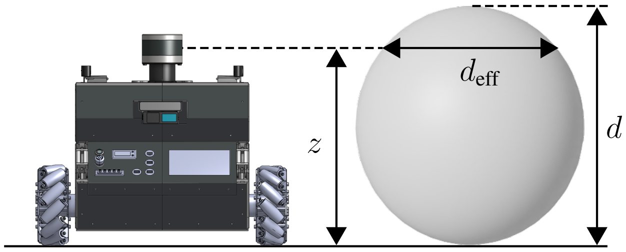


Figure 6: Robot next to a sphere.

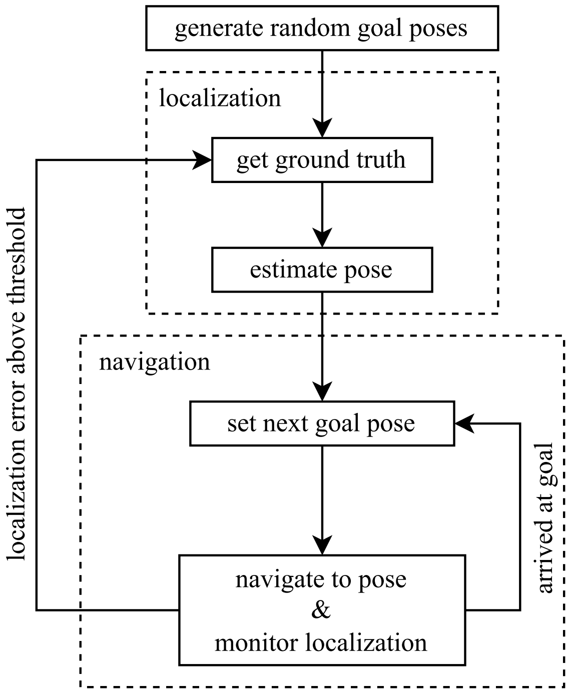

generate random goal poses
localization
get ground truth
estimate pose
navigation
set next goal pose
navigate to pose & monitor localization
localization error above threshold
arrived at goal

Figure 7: Robot process flow during data collection.

Table 1: A list of ROS topics, their message types, and descriptions.

ROS Topic	ROS Message Type	Description
/amcl_pose	geometry_msgs/msg/PoseWithCovarianceStamped	estimated pose by AMCL
/delocalizations	std_msgs/msg/Int16	localization failure counter
/heading_error	std_msgs/msg/Float32	orientation error
/map	nav_msgs/msg/OccupancyGrid	environment map
/momo/pose	geometry_msgs/msg/PoseStamped	ground truth pose
/particle_cloud	nav2_msgs/msg/ParticleCloud	particle set of AMCL
/position_error	std_msgs/msg/Float32	position error

Section 3.3) are randomized to mitigate the sim-to-real gap. The drift parameters are varied by randomly sampling values within the range 0.25 to 0.35. The range is chosen such that, in the absence of obstacles, no localization failures are observed, consistent with the expected behavior of a well-localized robot. Following this procedure, the subsequent training sample is recorded.

A comprehensive overview of all data included in a training sample is provided in Table 1. The experimental data are stored in a *rosbag*, the Robot Operating System (ROS) standard file format for recording and storing time-synchronized message sequences.

To ensure temporal consistency across heterogeneous sensor streams, all recorded messages are aligned to a common time axis using a zero-order hold (ZOH) transformation. In this process, each signal is discretized at the target sampling frequency by carrying forward the most recently observed value until a new message becomes available. This approach preserves the step-wise nature of asynchronous sensor updates while avoiding artificial interpolation of values that may not be physically meaningful. The ZOH transformation guarantees that every training sample is represented as a complete, synchronized feature vector, which is a prerequisite for learning models that operate on tabular time-series data.

In total, the dataset was collected across seven distinct maps: three small symmetric layouts, three small asymmetric layouts, and one large warehouse environment. For each map, experiments were conducted under three obstacle configurations: *static only*, *dynamic only*, and *combined static and dynamic*. This results in $7 \times 3 = 21$ unique experiment runs. Each run lasted 15 min of simulation time, although the corresponding rosbag recordings are shorter due to the difference between simulated and real time, and contained a varying number of resets, corresponding to the number of recorded sequences within that experiment. This variation reflects the influence of both map geometry and obstacle configuration on localization robustness.

4.2 AUTOMATIC LABELING FOR CLASSIFICATION

After experiment data is collected in *rosbag* files, the files need to be loaded into a machine learning pipeline. We are using *Flowcean* [18], a framework that provides a higher-level abstraction to generalize the learning and modeling approach. *Flowcean* includes an interface to load experiment data from rosbag and convert it into a tabular dataset.

Training samples are classified with a label y indicating if the localization is failing or not. The classification is evaluated by comparing the estimated pose $(x_k^{\text{est}}, y_k^{\text{est}}, \theta_k^{\text{est}})$ with the ground truth $(x_k^{\text{gt}}, y_k^{\text{gt}}, \theta_k^{\text{gt}})$ and evaluating the localization quality thresholds α and β, similar to [10]. A localization failure occurs if

$$y = \begin{cases} 0, \text{if } \Delta p < \alpha \wedge \Delta\theta < \beta \\ 1, \text{else.} \end{cases} \tag{14}$$

where Δp and $\Delta\theta$ define the position and orientation error given by

$$\Delta p = \sqrt{(x_k^{\text{gt}} - x_k^{\text{est}})^2 + (y_k^{\text{gt}} - y_k^{\text{est}})^2}, \tag{15}$$

$$\Delta\theta = \|\theta_k^{\text{gt}} - \theta_k^{\text{est}}\|. \tag{16}$$

In our experiments, $\alpha = 0.4\,\text{m}$ and $\beta = 0.4\,\text{rad}$. The choice of error thresholds is use case-specific and can be determined methodically [19]. It should be noted that $\alpha < \alpha_{\text{reset}}$ and $\beta < \beta_{\text{reset}}$. This separation between labeling thresholds and reset thresholds ensures that the system can still accumulate data from borderline states, those where localization is inaccurate but not yet reset. As a result, more samples are labeled as failures before the system intervenes to reset localization. This allows for a better representation of failure cases, avoiding a skewed dataset dominated by successful localization episodes. This ensures a more even distribution of positively (failure) and negatively (non-failure) labeled samples, which is essential for supervised learning algorithms to effectively distinguish between normal and failure states.

In summary, this section has detailed how raw experiment data is transformed into structured, labeled training samples. By automatically identifying failure cases based on pose estimation errors and using conservative thresholds, the dataset provides a balanced foundation for training classification models to detect localization failures.

5 Results

The created datasets consist of synchronized streams of localization estimates, ground-truth poses, sensor measurements, and the corresponding label, either nominal ($y = 0$) or failure ($y = 1$). Multiple challenging navigation scenarios are recorded, including dynamic obstacle interactions, feature-poor zones, and map ambiguities. Table 2 summarizes key statistics for the full collection of experiment runs as well as breakdowns by nominal (no failure) and failure cases.

After aligning the data streams with a ZOH transformation (cf. Section 4.1), the resulting number of samples is summarized in Table 2. Across all maps and experiment variations, the dataset contains a total of 417 185 labeled samples, with 320 870 nominal and 96 315 failure instances, corresponding to an overall failure rate of 23.1 %. Although this distribution of labels is unbalanced (about 1:3.4), this can be handled, e.g., by undersampling the majority-class (y=0) or oversampling the minority-class (y=1) [20, pp. 217-218].

A breakdown by environment shows that the occurrence of failures varies across scenarios, but no clear, consistent trends emerge regarding the impact of map symmetry or obstacle types. In symmetric environments, failure rates range from 9.1 % to 37.0 % in asymmetric layouts, they span 10.1 % to 34.8 %. The warehouse map exhibits intermediate rates (19.1 % to 28.8 %). This variability may be attributed to the domain randomization approach applied to odometry drift parameters (ϵ_x, ϵ_y, ϵ_θ, ζ_x, ζ_y, ζ_θ), which are reset and randomized after each localization failure (as described in Section 3.3). By introducing artificial drift and Gaussian noise in a randomized manner, the simulation ensures diversity in odometry quality, but this also introduces additional noise that can mask potential patterns tied to environmental factors such as geometry, dynamic obstacles, or static clutter.

As described in Section 4.1, the localization is reset once one of the reset thresholds is violated. The number of resets required during experiments provides some insight into scenario difficulty, with higher reset counts often (but not always) aligning with elevated failure rates. For instance, runs in symmetric map 2 and asymmetric map 2 show substantial variation, but these do not form a predictable pattern across all experiments. Overall, the lack of discernible trends underscores the complexity introduced by randomized odometry degradation, which simulates real-world sensor variability but complicates the isolation of specific environmental effects.

Taken together, these results indicate that the dataset captures a diverse set of failure-inducing conditions, even if underlying trends are obscured by odometry randomization. While the class distribution is reasonably balanced for training purposes, careful handling of imbalance (e.g., class weighting or resampling) will be necessary to avoid bias toward the dominant nominal class. Furthermore, the diversity across maps allows for evaluating generalization: models trained in one environment can be tested in structurally different settings to assess robustness.

6 Conclusion

This work investigates the problem of making self-localization in autonomous mobile robots more robust by providing a systematic way to generate failure-rich datasets. The simulation-based methodology shows that under challenging conditions, e.g., dynamic obstacles, ambiguous maps, and randomized odometry drift, the resulting datasets maintain a moderate imbalance between nominal and failure labels. This indicates that the induced localization failures are diverse enough to serve as a basis for data-driven monitoring, while still resembling conditions in which localization systems operate in practice.

The approach is reproducible and scalable: localization failures can be provoked in a controlled manner, ground truth and estimated poses can be recorded, and the system can be reset automatically. This enables the collection of large datasets without safety concerns or extensive manual effort. Several aspects could be improved, including the use of more realistic maps, the inclusion of sim-to-real sensor noise, and ultimately the collection of real-world data to close the gap between simulation and deployment.

Although the present study focuses on LiDAR-based particle filter localization, the methodology is in principle generalizable. Forcing localization failures and recording labeled recovery sequences can also be applied to other localization methods, such as visual odometry or multi-sensor fusion. Extending the simulation setup with additional sensing modalities, for example, cameras or depth sensors, would make it possible to evaluate alternative localization strategies. The dataset presented here, however, is primarily applicable to LiDAR-based systems.

The contributions establish a scalable groundwork for anticipating localization failures and for improving the safety, reliability, and efficiency of autonomous navigation in dynamic environments. The open-sourcing of code, documentation, simulation environments (including the omnidirectional robot model), and datasets enables the robotics

Table 2: Key Dataset Statistics

Exp.	Map	Dynamic	Static	Resets	Samples	Nominal (y=0)	Failure (y=1)	Failure Rate
1	warehouse	✓	✓	10	15 708	11 171	4537	28.8 %
2	warehouse	✓	✗	9	17 188	13 899	3289	19.1 %
3	warehouse	✗	✓	15	18 693	13 570	5123	27.4 %
4	symmetric 1	✓	✓	6	14 891	12 713	2178	14.6 %
5	symmetric 1	✓	✗	5	14 729	12 770	1959	13.3 %
6	symmetric 1	✗	✓	6	18 723	15 947	2776	14.8 %
7	symmetric 2	✓	✓	8	22 909	17 342	5567	24.3 %
8	symmetric 2	✓	✗	12	22 381	14 106	8275	37.0 %
9	symmetric 2	✗	✓	6	18 364	16 690	1674	9.1 %
10	symmetric 3	✓	✓	13	22 893	18 033	4860	21.2 %
11	symmetric 3	✓	✗	15	23 754	17 572	6182	26.0 %
12	symmetric 3	✗	✓	10	19 394	15 625	3769	19.4 %
13	asymmetric 1	✓	✓	11	22 623	16 889	5734	25.3 %
14	asymmetric 1	✓	✗	10	24 621	17 315	7306	30.0 %
15	asymmetric 1	✗	✓	12	23 350	19 560	3790	16.2 %
16	asymmetric 2	✓	✓	9	21 165	14 652	6513	30.8 %
17	asymmetric 2	✓	✗	4	14 548	13 071	1477	10.1 %
18	asymmetric 2	✗	✓	16	24 468	15 963	8505	34.8 %
19	asymmetric 3	✓	✓	10	18 788	13 435	5353	28.5 %
20	asymmetric 3	✓	✗	11	18 767	14 662	4105	21.9 %
21	asymmetric 3	✗	✓	10	19 228	15 885	3343	17.4 %
				Total:	417 185	320 870	96 315	23.1 %

community to replicate, extend, and benchmark predictive monitoring approaches.

Future work will address feature engineering from these datasets, the design and validation of predictive algorithms that integrate temporal and semantic sensor data, and their deployment in real-time robotic systems. Such developments would support proactive interventions, including sensor data filtering, pose correction, or adaptive path replanning, to reduce disruptions and improve autonomy in uncertain environments.

ACKNOWLEDGEMENT

This work is funded by BMBF project AGenC no. 01IS22047A.

REFERENCES

[1] J. Huang, S. Junginger, H. Liu, and K. Thurow, "Indoor Positioning Systems of Mobile Robots: A Review," *Robotics*, vol. 12, no. 2, p. 47, Mar. 2023, ISSN: 2218-6581. DOI: 10.3390/robotics12020047.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics* (Intelligent Robotics and Autonomous Agents Series). MIT Press, 2005, ISBN: 978-0-262-20162-9.

[3] H. Yin et al., "A Survey on Global LiDAR Localization: Challenges, Advances and Open Problems," *International Journal of Computer Vision*, Mar. 2024, ISSN: 1573-1405. DOI: 10.1007/s11263-024-02019-5.

[4] E. Maharmeh, Z. Alsayed, and F. Nashashibi, "A Comprehensive Survey on the Integrity of Localization Systems," *Sensors*, vol. 25, no. 2, p. 358, Jan. 2025, ISSN: 1424-8220. DOI: 10.3390/s25020358.

[5] C. Tsuchiya, S. Takei, Y. Takeda, and A. Khiat, "TTF: Time-To-Failure Estimation for ScanMatching-based Localization," in *2020 IEEE*

Intelligent Vehicles Symposium (IV), Oct. 2020, pp. 631–636. DOI: 10.1109/IV47402.2020.9304632.

[6] M. Knitt, M. B. Thakkar, S. Maroofi, H. W. Rose, and P. M. Braun, *Robot Localization Failure Prediction Dataset*, https://doi.org/10.15480/882.15836, Sep. 2025. DOI: 10.15480/882.15836.

[7] N. Akai, L. Y. Morales, and H. Murase, "Simultaneous pose and reliability estimation using convolutional neural network and Rao–Blackwellized particle filter," *Advanced Robotics*, vol. 32, no. 17, pp. 930–944, Sep. 2018, ISSN: 0169-1864. DOI: 10.1080/01691864.2018.1509726.

[8] K. He, H. Ding, N. Xu, and K. Guo, "Wheel Odometry with Deep Learning-Based Error Prediction Model for Vehicle Localization," *Applied Sciences*, vol. 13, no. 9, p. 5588, Jan. 2023, ISSN: 2076-3417. DOI: 10.3390/app13095588.

[9] H. Roggeman, J. Marzat, A. Bernard-Brunel, and G. L. Besnerais, "Autonomous Exploration with Prediction of the Quality of Vision-Based Localization," *IFAC-PapersOnLine*, 20th IFAC World Congress, vol. 50, no. 1, pp. 10 274–10 279, Jul. 2017, ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2017.08.1479.

[10] M. Eder, M. Reip, and G. Steinbauer, "Creating a robot localization monitor using particle filter and machine learning approaches," *Applied Intelligence*, vol. 52, no. 6, pp. 6955–6969, Apr. 2022, ISSN: 0924-669X, 1573-7497. DOI: 10.1007/s10489-020-02157-6.

[11] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research (IJRR)*, vol. 36, no. 1, pp. 3–15, 2017. DOI: 10.1177/0278364916679498.

[12] K. Chen et al., "Womd-lidar: Raw sensor dataset benchmark for motion forecasting," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2024. DOI: 10.48550/arXiv.2304.03834.

[13] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. DOI: 10.1109/IROS.2012.6385773.

[14] D. Fox, "Kld-sampling: Adaptive particle filters," in *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, ser. NIPS'01, Vancouver, British Columbia, Canada: MIT Press, 2001, pp. 713–720.

[15] M. Knitt, M. Thakkar, Y. Elgohary, P. M. Braun, Y. Blank, and H. W. Rose, "Momo: An open-source modular mobile robot research platform," in *25th International Conference on Material Handling, Constructions and Logistics, MHCL 2024*, Technische Universität Wien, 2024. [Online]. Available: https://tore.tuhh.de/entities/publication/26e95c56-8320-42a0-91a5-01432206b62c.

[16] M. Wiedemann, O. Ahmed, A. Dieckhöfer, R. Gasoto, and S. Kerner, "Simulation Modeling of Highly Dynamic Omnidirectional Mobile Robots Based on Real-World Data," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, Japan: IEEE, May 2024, pp. 16 923–16 929, ISBN: 979-8-3503-8457-4. DOI: 10.1109/ICRA57147.2024.10611459.

[17] M. Schrick, J. Hinckeldeyn, M. Thiel, and J. Kreutzfeldt, "A microservice based control architecture for mobile robots in safety-critical applications," *Robotics and Autonomous Systems*, vol. 183, p. 104 795, 2025, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2024.104795.

[18] M. Knitt et al., "Towards the Automatic Generation of Models for Prediction, Monitoring, and Testing of Cyber-Physical Systems," in *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2023, p. 4. DOI: 10.1109/ETFA54631.2023.10275706.

[19] J. Schyga, M. Knitt, J. Hinckeldeyn, and J. Kreutzfeldt, *Method for Specifying Location Data Requirements for Intralogistics Applications*. Apr. 2023. DOI: 10.15488/13429.

[20] M. Kubat, *An Introduction to Machine Learning*. Cham: Springer International Publishing, 2021, ISBN: 978-3-030-81934-7 978-3-030-81935-4. DOI: 10.1007/978-3-030-81935-4.

Markus Knitt (M.Sc.) studied Mechatronics Engineering at Hamburg University of Technology. Since 2023, he has been working as a Research Associate at the Institute of Logistics Engineering. His research focuses on data-driven modeling of robot localization systems. Address: Theodor-Yorck-Straße 8, 21079 Hamburg, Germany, Phone: +49 40 42878-3557, E-Mail: markus.knitt@tuhh.de

Sean Maroofi (M.Sc.) is a Research Associate at the Institute of Logistics Engineering. Prior, he obtained his Mechatronics Master's degree at Hamburg University of Technology. Address: Theodor-Yorck-Straße 8, 21079 Hamburg, Germany, Phone: +49 40 42878-4005, E-Mail: sean.maroofi@tuhh.de

Manav Thakkar (B.Tech.) is currently pursuing M.Sc. in Mechatronics at Hamburg University of Technology. Since 2024, he has been working as a Student Research Assistant at the Institute of Logistics Engineering. Address: Theodor-Yorck-Straße 8, 21079 Hamburg, Germany, E-Mail: manav.thakkar@tuhh.de

DOI: 10.2195/lj_proc_knitt_en_202510_01

Hendrik Rose (M.Sc.) studied Mechanical and Industrial Engineering at Hamburg University of Technology. Since 2021, he has been working as a Research Associate and Chief Engineer (since 2023) at the Institute of Logistics Engineering. Address: Theodor-Yorck-Straße 8, 21079 Hamburg, Germany, Phone: +49 40 42878-3668, E-Mail: hendrik.wilhelm.rose@tuhh.de

Philipp Braun (M.Sc.) studied Logistics & Mobility and International Industrial Engineering and Management at the Hamburg University of Technology. Since 2019, he has been working as a Research Associate and Chief Engineer (since 2023) at the Institute of Logistics Engineering. Address: Theodor-Yorck-Straße 8, 21079 Hamburg, Germany, Phone: +49 40 42878-3556, E-Mail: philipp.braun@tuhh.de