Edge Device-Optimized LiDAR SLAM for Real-Time and Robust Localization in Dynamic Environments

Sai Parimi, and Robert Bensch, *ANavS GmbH*

BIOGRAPHY

Sai Parimi is a computer vision expert at ANavS GmbH. He works mainly on environment perception using LiDARs with a major focus on LiDAR SLAM and 3D object detection. He received his M.Sc. degree in Electrical and Computer Science Engineering from Darmstadt Univerity of Applied Sciences, Germany. He graduated in 2021 with a masters thesis from Fraunhofer-Institut für Produktionstechnik und Automatisierung working on real-time pointcloud segmentation for industrial object detection to improve bin picking robot's sorting algorithm. He has been working at ANavS GmbH since 2022.

Robert Bensch leads the computer vision team at ANavS. The team focus is on precise and robust positioning in GNSS denied or degraded areas using camera and LiDAR sensors, as well as on environment detection and mapping. He received his PhD degree in applied computer science from the Albert-Ludwigs-Universität Freiburg, Germany, with focus on biomedical image analysis and motion pattern analysis. Dr. Bensch joined ANavS in 2017 and has been building up the computer vision team since several years.

ABSTRACT

LiDAR-based Simultaneous Localization and Mapping (SLAM) is a key technology for achieving reliable localization and accurate 3D reconstruction of the environment. In particular, LiDAR SLAM provides robust information when traversing uncharted or GNSS-denied areas, operating reliably under diverse environmental conditions—unlike cameras, which require favorable lighting. This makes LiDAR SLAM central to autonomous navigation, where robots and vehicles demand precise positioning in challenging terrains, and it offers strong benefits when fused with GNSS-based positioning. There has been extensive progress in LiDAR SLAM as well as in algorithms for dynamic object removal. However, relatively little attention has been given to integrating these two aspects into a unified framework capable of real-time operation. In this work, we address this gap by presenting a real-time LiDAR SLAM pipeline that seamlessly incorporates dynamic object removal to achieve robust performance in highly dynamic environments. Our pipeline tightly couples LiDAR pointclouds with Inertial Measurement Unit (IMU) data in the frontend to produce accurate odometry estimates. The estimated trajectory, along with the associated pointclouds, is then optimized in a factor graph-based backend to reduce long-term drift and improve global consistency. To further strengthen performance, we incorporate dynamic object removal, which improves odometry estimation by emphasizing stable environmental features while also producing cleaner global maps. These refined maps enable more reliable global registration and enhance the subsequent re-localization process. Robust re-localization, in turn, allows drift-free traversal within pre-built maps. Finally, we demonstrate how fusing our improved LiDAR-Inertial Odometry with Real-Time Kinematic (RTK) positioning further enhances localization accuracy, particularly in GNSS-denied environments.

I. INTRODUCTION

Autonomous driving has gained significant traction in recent years, driven not only by advancements in compute platforms but by significant improvements in sensors that perceive and interpret the surrounding environment, which is crucial for safe and robust navigation. While every component of an autonomous system is important, knowing the vehicle's precise position stands out as the most critical. Even a small error in localization can lead to unsafe maneuvers or collisions. Significant research is being conducted to achieve precise localization and mapping using optical sensors such as Light Detection and Ranging (LiDAR), cameras and Radio Detection and Ranging (RADAR).

Accurate localization and mapping using optical sensors becomes particularly critical in dense urban environments or indoor parking areas, where GNSS signals are often unreliable. The DREAM (DRiving aids by E-GNSS AI and Machine learning) project, funded under EUSPA's Fundamental Elements program, aims to develop highly precise and dependable positioning algorithms for such challenging scenarios, providing support for advanced driver assistance systems (ADAS) (Morán et al., 2025).

Among these sensors, LiDAR stands out for its ability to provide highly reliable localization. Unlike cameras, which are limited by their field of view and can be heavily affected by lighting conditions, LiDAR sensors offer a full surround view through multiple laser scanners. They produce dense, three-dimensional point clouds that capture detailed geometric information about the environment. This rich spatial data allows autonomous vehicles to map their surroundings accurately and estimate their motion reliably, even in challenging lighting or weather conditions where cameras may struggle. Compared to RADAR, LiDAR provides much higher spatial resolution, capturing fine structural details of the environment that RADAR often cannot detect. This makes LiDAR particularly effective for building precise maps and estimating vehicle pose in complex, dynamic scenarios. Moreover, LiDAR can consistently detect both nearby and distant objects, enabling drift-resistant localization that is critical for safe autonomous navigation. These features make LiDAR an essential component for high-precision SLAM and robust autonomous operation.

In this paper, we present a robust, real-time LiDAR SLAM system that combines artificial intelligence for dynamic object removal with conventional localization and mapping techniques. Both components are optimized to run on embedded automotive platforms with limited computational resources. We further demonstrate that the output of our SLAM pipeline, when integrated into the federated Kalman filter, produces accurate and reliable results even in GNSS-denied conditions, enabling robust and precise positioning in complex urban and indoor environments.

The remainder of the paper is structured as follows. Section II reviews related work. Section III presents the system architecture. Section IV details the methodology. Section V showcases system results and Section VI concludes the paper and discusses future work.

II. RELATED WORK

The underlying principle behind LiDAR odometry estimation is the Iterative Closest Point (ICP) algorithm (Besl and McKay, 1992) and its variants (Chen and Medioni, 1992; Segal et al., 2009), which align successive pointcloud frames by minimizing the distance between corresponding points or surfaces. The resulting transformation between consecutive scans provides an estimate of the vehicle's motion. In theory, this approach should be robust to the vehicle's movement; however, in practice, motion distortion arising from the sequential acquisition of points within a single scan can lead to significant drift in the estimated odometry. LOAM (Zhang and Singh, 2014) and LeGO-LOAM (Shan and Englot, 2018) address this issue by using IMU measurements to de-skew pointclouds, effectively providing a motion prior for scan matching. While this IMU-based correction improves the quality of the pointclouds and handles short-term drift, both systems remain loosely coupled, as the IMU data is not incorporated into the optimization, leaving them susceptible to long-term drift due to sensor bias.

Recent advancements in LiDAR SLAM, such as LIO-SAM (Shan et al., 2020) and LINS (Qin et al., 2019), tightly couple LiDAR-inertial measurements, where the IMU is used not only for pointcloud de-skewing but also in the optimization step. While LIO-SAM uses factor graphs (Kaess et al., 2011) for backend optimization, LINS uses an iterated error-state Kalman filter. These approaches introduce tightly coupled optimization for improved pose estimation, but still accumulate drift over time, as they operate on smaller local maps to maintain real-time capability. Additionally, these methods rely on feature extraction for scan registration, making them computationally more expensive and leading to degraded performance in sparse environments. FAST-LIO2 (Xu et al., 2021) addresses some of these limitations by using a direct scan-to-map registration that does not rely on feature extraction. The method employs an incremental k-d tree (ikd tree) for efficient map representation, which, unlike traditional k-d trees (Skrodzki, 2019) that require a complete reconstruction for updates, supports dynamic point insertion and deletion in logarithmic time. Despite its robustness and efficiency, FAST-LIO2 still has limitations due to its filtering-based methods and lack of global optimization and loop closures to correct long-term drift. To address these issues, we employ FAST-LIO-SAM (Lee, 2022) which combines the robustness and efficiency of FAST-LIO2 in the frontend and a factor graph-based global optimization and loop closure in the backend.

Although these approaches differ in how they estimate odometry, a common underlying principle remains: they all rely on scan registration. The accuracy of this registration, and the subsequent odometry estimation, depends on the quality of the input pointclouds. Removing dynamic objects ensures that only static structures are retained, providing consistent geometry across frames and leading to more reliable odometry estimation. This process is particularly critical in urban scenarios, where the prevalence of dynamic objects can severely degrade registration, resulting in significant drift over time and, in the worst case, threatening the stability of the entire SLAM pipeline.

Several state-of-the-art dynamic object removal algorithms have been proposed, each introducing a unique strategy to improve accuracy and consistency. LMNET (Chen et al., 2021) proposes a learning-based approach for dynamic object segmentation in LiDAR pointclouds by exploiting sequential scans. This method projects 3D pointclouds into range images and uses convolutional neural networks to identify moving objects based on temporal changes across consecutive frames. 4DMOS (Mersch et al., 2022) proposes converting LiDAR scans into 4D pointclouds and applying sparse 4D convolution to extract spatio-temporal features. A receding horizon strategy is employed to update predictions as new scans arrive, and a binary Bayes filter is used to integrate these predictions over time, enhancing robustness. MapMOS (Mersch et al., 2023) extends 4DMOS by fusing the predictions into a probabilistic representation of the dynamic environment using a Bayes filter, resulting in a volumetric belief model that indicates which parts of the environment can be occupied by moving objects.

Although the aforementioned approaches are very accurate and robust for dynamic object segmentation, they lack real-time capability when running on an embedded device optimized for automotive use cases and working with high density pointclouds. To meet this stringent requirement, we implement a multi-stage approach for real-time dynamic object removal. First, we use a real-time capable semantic segmentation algorithm (Reichert et al., 2025) to identify potential dynamic objects. This is then coupled with conventional methods that check for temporal consistency across voxels. Together, these stages help accurately determine dynamic objects in a scene.

III. SYSTEM ARCHITECTURE

The LiDAR SLAM architecture presented here is a tightly coupled frontend-backend system with a pointcloud preprocessing stage. The frontend is FAST-LIO2, where LiDAR pointclouds and IMU measurements from a high resolution LiDAR are tightly integrated to estimate odometry. The estimated odometry is further optimized in a factor graph-based backend. An IMU prior factor is used for initialization, while relative poses and loop closures serve as between factors in the optimization. The de-skewed pointclouds are transformed using the optimized poses and subsequently stitched together to produce a globally consistent map. The factor graph backend can optionally incorporate GNSS factors to align the trajectory to a global frame. In this work, we omit GNSS integration to highlight the robustness of our SLAM pipeline in GNSS-denied environments.

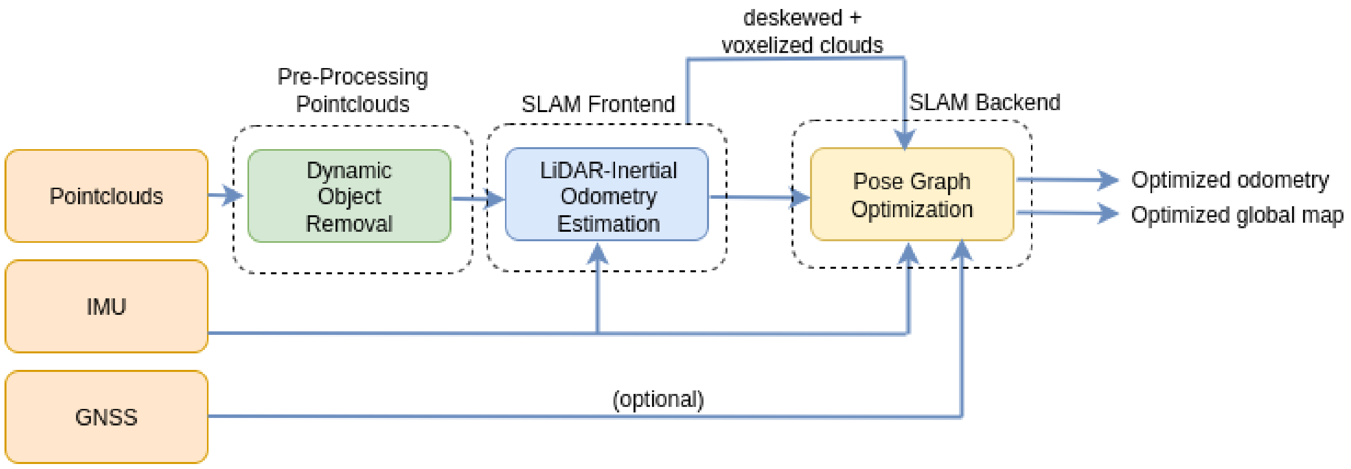


Figure 1: LiDAR SLAM pipeline.

IV. METHODOLOGY

1. AI-aided dynamic object removal

Removing dynamic objects from LiDAR pointclouds is a critical step for achieving robust localization and mapping. The presence of dynamic objects can lead to registration errors, resulting in drifted odometry or failures in localization. To address this challenge, we implement a real-time dynamic object removal algorithm. The algorithm takes raw LiDAR pointclouds as input and produces cleaned pointclouds with dynamic objects removed. These refined pointclouds are subsequently used in the downstream SLAM pipeline to ensure accurate odometry estimation and reliable mapping.

In our algorithm, we subscribe to pointclouds streamed from the sensor. In Figure 2 (top left), we show a raw scan in an urban environment with two dynamic cars, one of which is partially occluded, and a dynamic tram. Static cars are also present in the scene along with other reliable geometries such as buildings, road signs, and vegetation. The pointclouds are dense, with approximately 131.000 points published at $20\,\mathrm{Hz}$, making it challenging to process them within $50\,\mathrm{ms}$ to maintain real-time performance. To improve run time efficiency, we need to downsample the pointclouds.

Downsampling using voxel grids (Rusu and Cousins, 2011) reduces the size of the pointclouds, but at the cost of spatial resolution. To mitigate this trade-off, we mimic downsampling by first semantically segmenting the pointclouds as shown in Figure 2 (bottom left), and only select points likely to be dynamic, such as cars, bicycles, trucks, pedestrians, and motorcycles, while discarding points that are truly static such as buildings, roads, sidewalks, traffic signs, and vegetation. This approach reduces the number of points while preserving the geometry of all objects in the scene. To maintain real-time capability, we optimize the model (Reichert et al., 2025) using TensorRT, achieving a mean IoU of $55.6\,\%$ with a processing time of approximately $12\,\mathrm{ms}$.

After downsampling the pointclouds, we estimate the transformation of the pointclouds in the sliding window. These transformations are then used to bring the pointclouds into a common frame of reference. This step is crucial, as it compensates for ego motion and aligns all frames in the ego view. The aligned pointclouds resemble a virtual stack with respect to the frame of reference. The transformations, as shown in Figure 2 (bottom right), are estimated using small GICP (Koide, 2024), achieving a registration score of 0.2 for a sliding window size of 4 with a processing time of approximately $16\,\mathrm{ms}$.

Now that we have downsampled the pointclouds and transformed them into a common frame of reference, the next step in identifying dynamic points is to check for temporal consistency. We partition the pointclouds into voxel grids in the pointclouds and track whether points remain within the same voxels across all frames in the sliding window. Points that consistently remain in a voxel across all frames are classified as static, while points that move between voxels are deemed dynamic. This binary classification effectively identifies dynamic points in each scene, as shown in Figure 2 (top right), achieving an IoU of $65\,\%$ with a processing time of approximately $20\,\mathrm{ms}$.

Figure 3 shows a side-by-side comparison of the raw pointcloud (left) and the cleaned pointcloud (right) in an urban scenario, where the LiDAR-equipped vehicle is slowly moving at a signal along with surrounding cars. In the cleaned pointcloud, all dynamic cars have been removed, leaving only static points such as containers, buildings, and vegetation.

The performance of the algorithm is evaluated using Intersection over Union (IoU), a metric that computes the ratio between the intersection and union of predicted and ground-truth regions. All evaluations are performed on a Nuvo-10108GC computer.

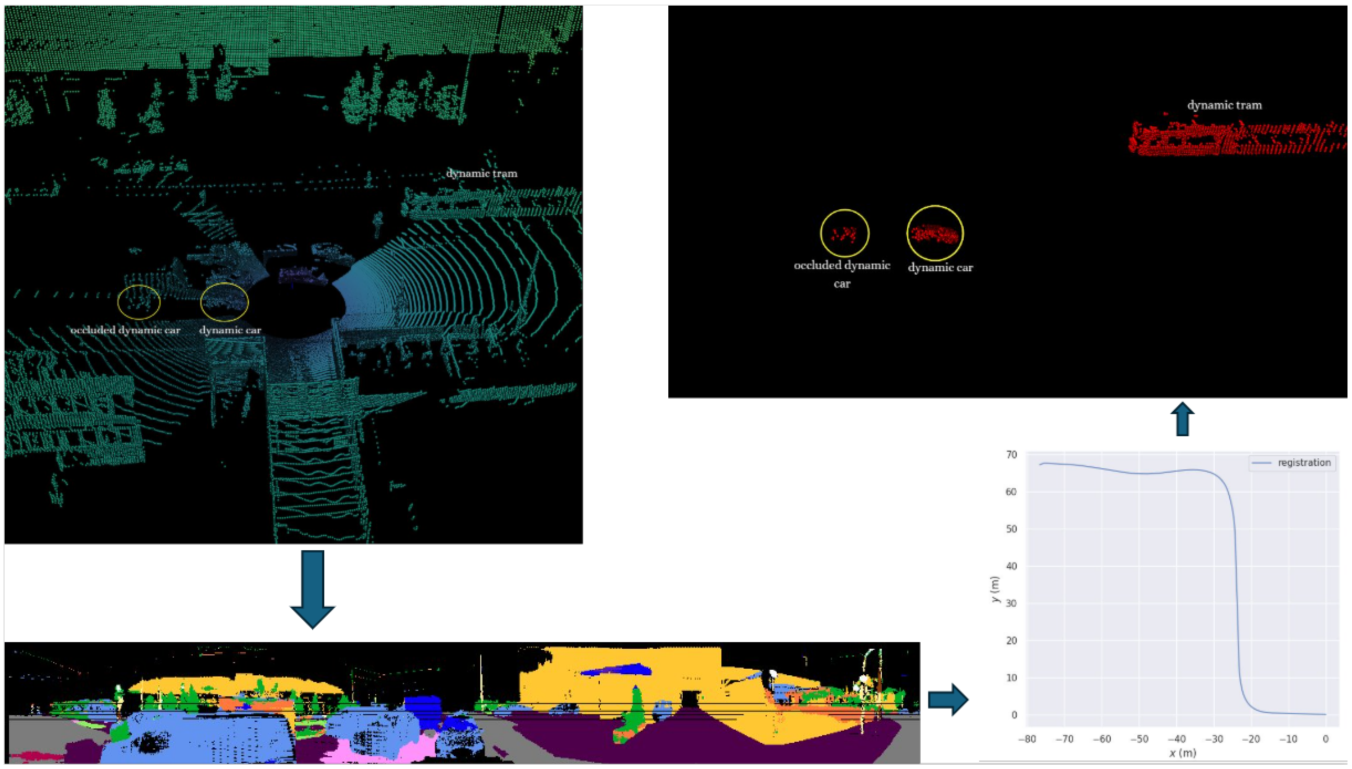


Figure 2: Dynamic Object Removal pipeline. The raw scans (top left) are first semantically segmented (bottom left) to downsample the pointclouds by only filtering possible dynamic classes. Then a registration (bottom right) of the pointclouds is done using small GICP to transform them to a common frame of reference. Finally, a temporal consistency check yields a binary mask (top right) of dynamic objects (red) and static objects (black).

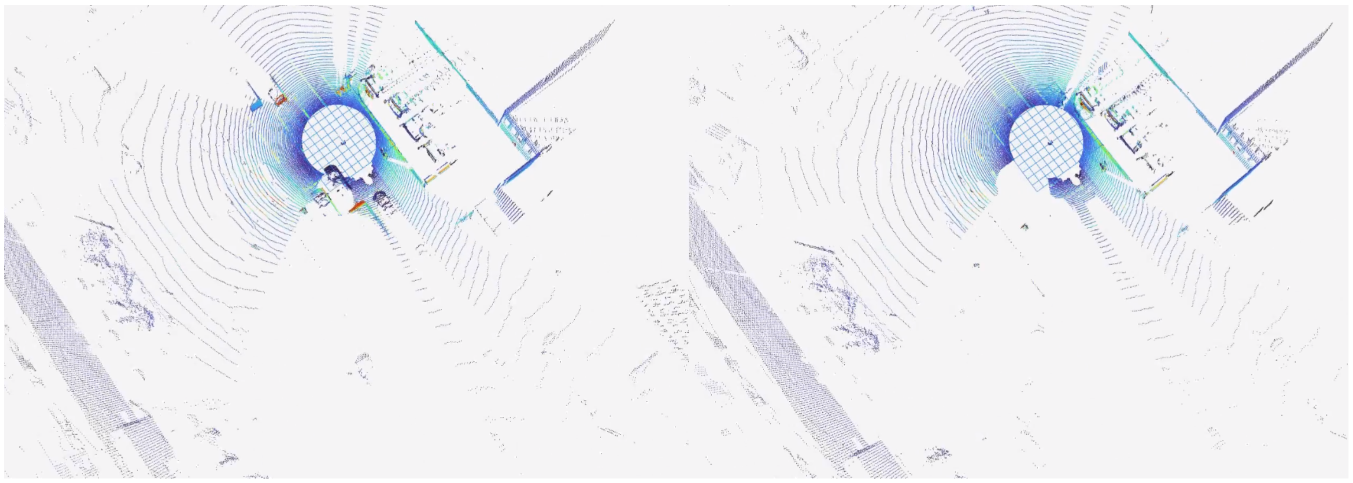


Figure 3: Birds eye view of LiDAR data. Left: Raw pointclouds streamed from the sensor. Right: Cleaned pointclouds with dynamic objects removed.

2. LiDAR SLAM frontend and backend

The frontend of our LiDAR SLAM pipeline is FAST-LIO2, which is robust, compact, and fast, all the qualities that are essential for real-time operation. Figure 4 provides a high-level overview of the algorithm's workflow. The cleaned pointclouds are streamed through our dynamic object removal algorithm along with the IMU measurements from the Ouster LiDAR. The IMU measurements are propagated forward, i.e. they are integrated over time, which becomes the prediction step for the iterated Kalman Filter (iKF) due to their higher frequency and smoother motion estimation. The pointclouds are first de-skewed, i.e. they are motion compensated to create smoother pointclouds. The de-skewed pointcloud is aligned to a local map that is being populated in the ikd-tree using point-to-plane registration. The residual measurements extracted from the point-to-plane distances are used as the update step for the iKF, which, upon converging, generates the odometry. The de-skewed pointcloud is then transformed and inserted into the ikd-tree to extend the local map.

In the backend, we employ a factor graph optimization framework to refine the odometry estimates generated by the frontend. Figure 5 illustrates the fundamental structure of the factor graph approach. The graph consists of variable nodes $(x0 - xN)$ representing the system states to be optimized, including robot poses, velocities, and IMU biases. Factors ($f1 - fN$, LC) encode probabilistic constraints between state variables. These factors encompass various measurement modalities: between-factors representing relative pose constraints, loop closure detections (LC), wheel odometry measurements, GPS observations, landmark associations, and IMU pre-integration terms. An IMU prior factor is used to initialize the graph. Each factor imposes constraints to ensure consistency with the observed measurements.

The factor graph formulation enables joint optimization through nonlinear least squares minimization, where the objective function seeks to minimize the Mahalanobis distance between predicted and observed measurements across all factors. This optimization yields globally consistent pose estimates that account for measurement uncertainties and inter-dependencies. The resulting optimized trajectory, combined with the corresponding de-skewed pointclouds, enables the construction of a globally consistent map.

Figure 6 compares the global maps generated for an indoor parking lot. The map generated from the raw pointclouds (left) exhibits streaks, which are outliers caused by dynamic objects and lead to registration failures. In contrast, the map generated from the cleaned pointclouds (right) accurately captures the static environment with minimal or no outliers, resulting in improved registration.

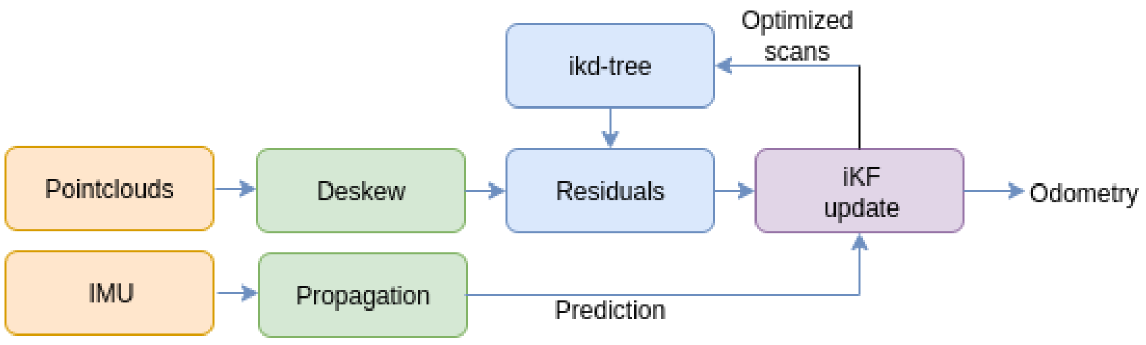


Figure 4: FAST-LIO2 pipeline for odometry estimation.

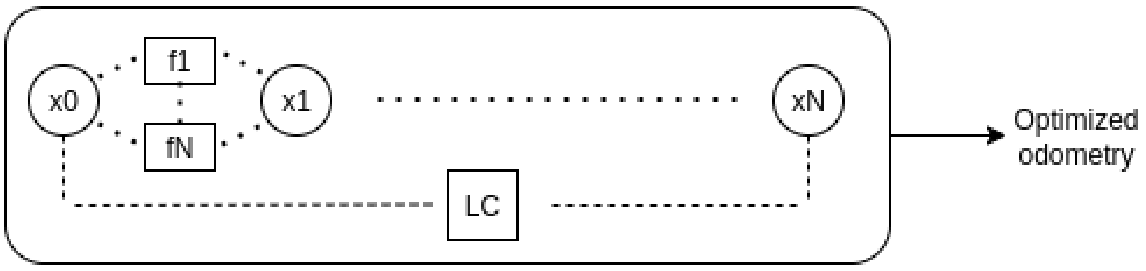


Figure 5: Factor graph structure for optimization.

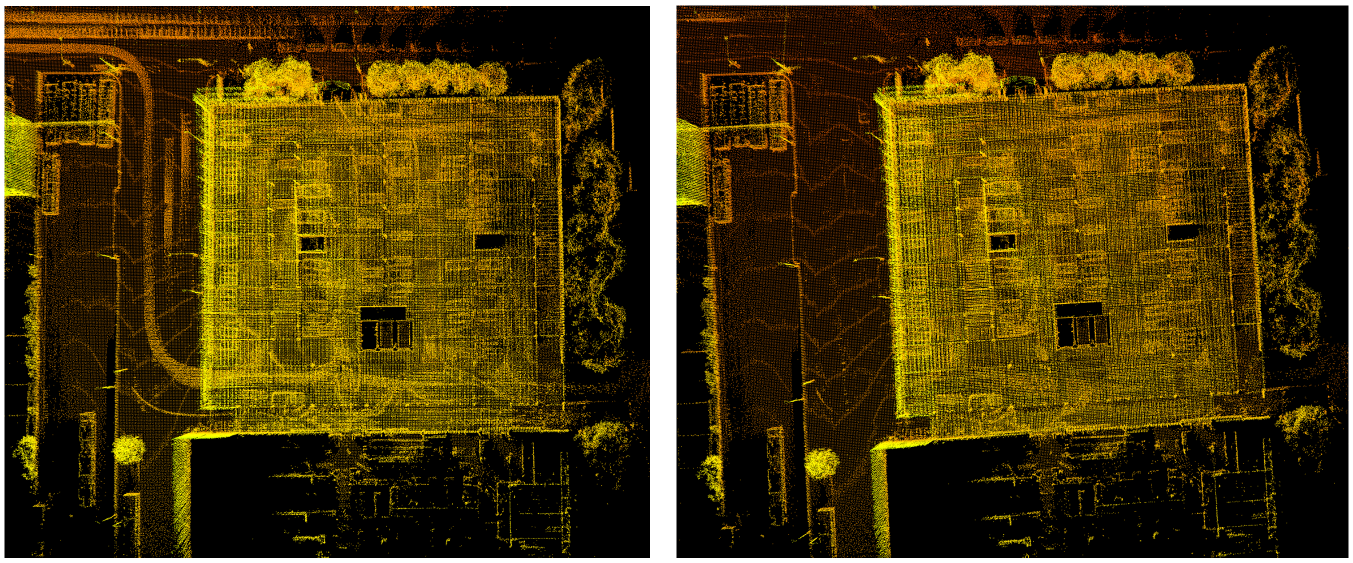


Figure 6: Left: Global map with raw pointclouds. Right: Global map with cleaned pointclouds

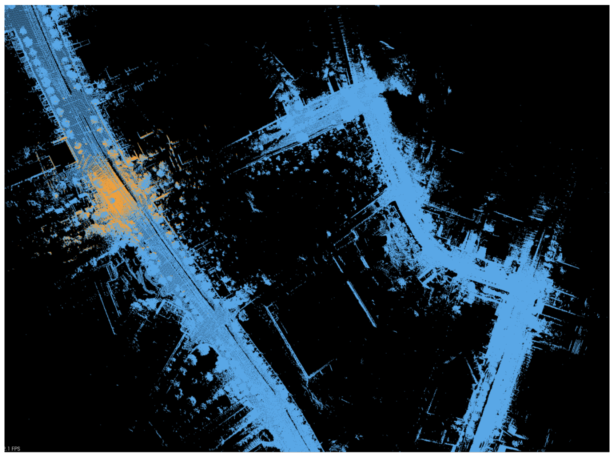
3. Global registration and re-localization

Online LiDAR SLAM systems that estimate odometry from pointclouds and IMU measurements are inherently prone to drift. In the absence of loop closures, this drift accumulates over time, leading to growing discrepancies between the estimated and true trajectories. When a loop closure is detected, the accumulated drift can be corrected by global optimization, significantly improving the overall accuracy of the trajectory. Drift in odometry estimation is inevitable during online SLAM. However, when the robot revisits an area that has already been mapped, it does not need to rely solely on online odometry estimates. By using the global map as reference, loop closure effects can be mimicked, ensuring that the pose is always optimized without accumulating any drift. This process of estimating odometry relative to a pre-optimized map is known as re-localization. Re-localization is a widely used concept in autonomous systems for improving positioning accuracy, thereby enabling more reliable navigation.

A key requirement in re-localization, as in online SLAM, is proper system initialization. Unlike online SLAM, where the initial pose can be set to the identity matrix, re-localization requires a prior on the robot's pose within the global reference frame. Accurate knowledge of this initial pose is essential to align the robot with the pre-optimized global map, which is typically achieved through global registration of the current pointcloud with respect to the map.

We perform global registration using KISS-Matcher (Lim et al., 2025), which combines a novel Faster-PFH feature detector with k-core-based graph pruning to efficiently reject outliers. Fast Point Feature Histogram (FPFH) (Rusu et al., 2009) is a widely used 3D point cloud descriptor that encodes the local geometric relationships between a point and its neighbors, enabling robust point matching. Faster-PFH accelerates this process by optimizing neighborhood searches and reducing redundant computations in the histogram aggregation, achieving similar descriptive power at a fraction of the computational cost. This results in a fast, accurate, and versatile end-to-end registration pipeline. Since the global map typically consists of several million points, registering a pointcloud with only around 100.000 points against such a huge map would be computationally expensive. To speed up the registration, we divide the global map into smaller submaps and perform the registration with each submap in parallel. Once a global initial estimate is obtained, it serves as a coarse drifted odometry. To correct for this drift, the current scan is registered against keyframes within a defined search radius relative to the estimate, thereby refining the pose in a manner analogous to a loop closure. Using this approach, we achieve a processing time of approximately $6\,\mathrm{s}$ for a $4\,\mathrm{km}$-wide map.

Figure 7 shows the global registration results where the yellow pointcloud represents the current input and the blue pointcloud corresponds to the optimized global map. We present registrations for both outdoor(top) and indoor (bottom) environments. The results demonstrate that global registration can be successfully performed even indoors, where GNSS signal is unavailable and the environment is highly symmetric. Once a global initialization is obtained, re-localization is performed by matching the incoming pointclouds against saved keyframes. This effectively creates a continuous loop closure correction of the odometry, providing a drift-free pose estimate relative to the global map.



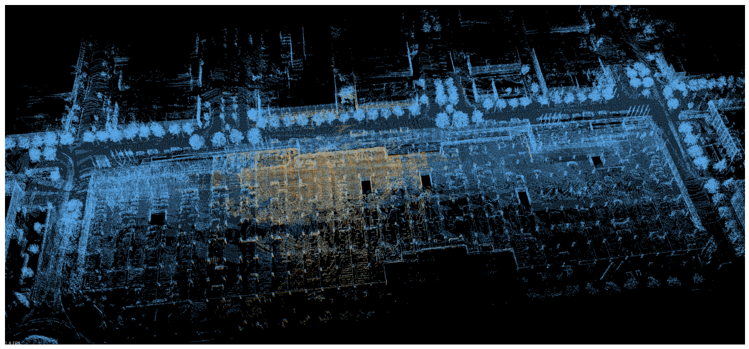


Figure 7: Global registration results for both outdoor (top) and indoor (bottom) environments. The yellow pointcloud represents the current input while the blue pointcloud represents the optimized global map.

V. SYSTEM RESULTS

In this section, we present results from our online LiDAR SLAM, using a high resolution Ouster OS1-128 LiDAR. The evaluation focuses on an indoor trajectory to highlight the robustness of SLAM in GNSS-denied environments. Prior to entering the indoor parking lot, the vehicle traversed approximately $2\,\mathrm{km}$ without loop closures. Inside the parking lot, three loops were completed before exiting.

Figure 8 shows the horizontal error of the LiDAR-Inertial Odometry (LIO) benchmarked against the ground truth. The ground truth was generated using an GNSS/IMU fusion approach with RTK corrections and a high-grade IMU, refined using Rauch–Tung–Striebel smoother to obtain accurate and reliable reference trajectories. We benchmarked two variants of LIO: one using raw, unprocessed pointclouds directly from the sensor, which resulted in an RMSE of approximately $0.7\,\mathrm{m}$, and another using cleaned pointclouds with dynamic objects removed, which achieved an RMSE of approximately $0.5\,\mathrm{m}$. The result plots clearly demonstrate that dynamic object removal improves LIO accuracy, yielding an average reduction in RMSE of about $20\,\mathrm{cm}$.

Figure 9 presents results from our central filter, implemented as a semi-tightly coupled Extended Kalman Filter. When we do not incorporate LIO into the filter and estimate odometry using just the IMU, we observe an RMSE of approximately $1\,\mathrm{m}$. In contrast, incorporating LIO into the fusion significantly improves performance, reducing the RMSE to approximately $0.5\,\mathrm{m}$.

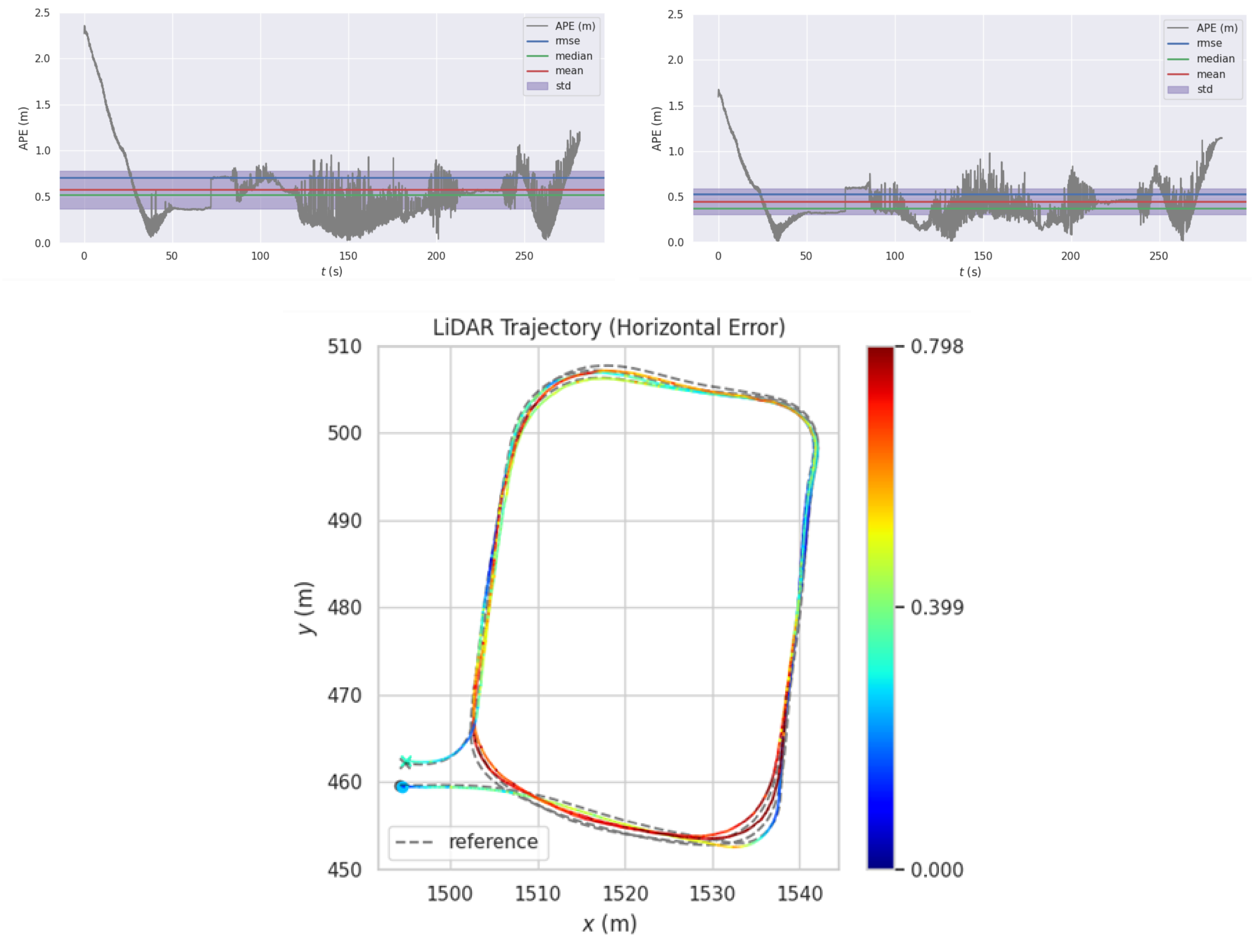


Figure 8: Horizontal error of LiDAR-Inertial Odometry (LIO). Top Left: Horizontal error using unprocessed pointclouds. Top Right: Horizontal error using cleaned pointclouds with any dynamic objects removed. Bottom: Horizontal error of LIO within the indoor parking lot.

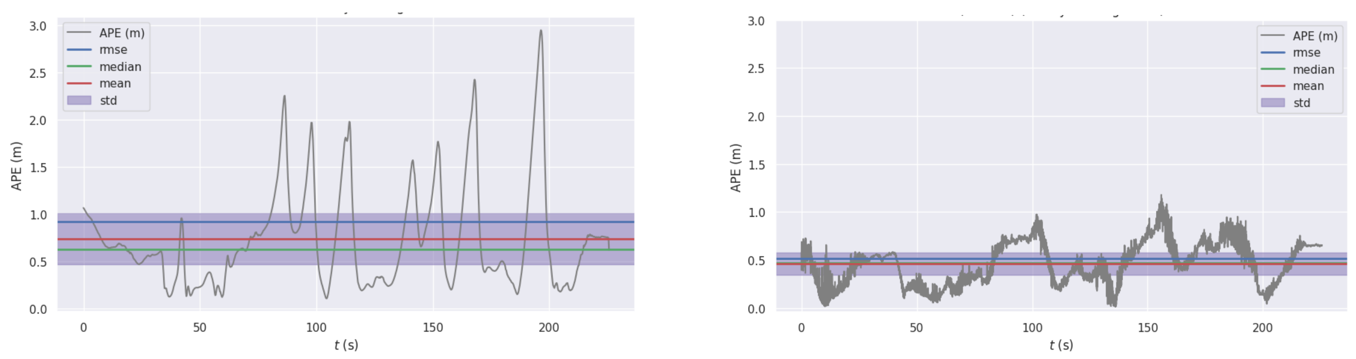


Figure 9: Performance of the central filter. Left: Horizontal error when fusing only IMU data. Right: Horizontal error when additionally fusing LIO measurements.

VI. CONCLUSION AND FUTURE SCOPE

In this paper, we presented a complete LiDAR SLAM pipeline integrating FAST-LIO2 for odometry estimation with a factor graph-based backend for optimization. A key contribution is a real-time dynamic object removal module incorporated into the preprocessing stage, which significantly improves SLAM performance and is particularly crucial in GNSS-denied environments. We further demonstrated that incorporating LiDAR-Inertial Odometry into our central filter enhances sensor fusion accuracy compared to GNSS–IMU integration alone. Robust global registration was achieved using cleaned global maps from the SLAM pipeline, and we extended this capability to re-localization, enabling drift-free odometry relative to pre-generated maps. Finally, we showed that the entire pipeline runs efficiently on edge devices without reliance on external infrastructure. Future work will focus on integrating GNSS to generate geo-referenced maps and further enhance the system's applicability in large-scale deployments. Additionally, we plan to incorporate other factors, such as wheel odometry, to benchmark potential improvements in performance and scalability.

ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to the European Union and the European Union Agency for the Space Programme (EUSPA) for funding the DREAM project under the Fundamental Elements programme, and for their valuable guidance and support throughout the project.

REFERENCES

Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256.

Chen, X., Li, S., Mersch, B., Wiesmann, L., Gall, J., Behley, J., and Stachniss, C. (2021). Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robotics and Automation Letters (RA-L)*, 6:6529–6536.

Chen, Y. and Medioni, G. (1992). Object modelling by registration of multiple range images. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2724–2729.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2011). isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *2011 IEEE International Conference on Robotics and Automation*, pages 3281–3288.

Koide, K. (2024). small_gicp: Efficient and parallel algorithms for point cloud registration. *Journal of Open Source Software*, 9(100):6948.

Lee, E. M. (2022). Fast-lio-sam: Fast-lio with smoothing and mapping. `https://github.com/engcang/FAST-LIO-SAM`.

Lim, H., Kim, D., Shin, G., Shi, J., Vizzo, I., Myung, H., Park, J., and Carlone, L. (2025). KISS-Matcher: Fast and Robust Point Cloud Registration Revisited. In *Proc. IEEE Int. Conf. Robot. Automat.* Accepted. To appear.

Mersch, B., Chen, X., Vizzo, I., Nunes, L., Behley, J., and Stachniss, C. (2022). Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7503–7510.

Mersch, B., Guadagnino, T., Chen, X., Vizzo, I., Behley, J., and Stachniss, C. (2023). Building Volumetric Beliefs for Dynamic Environments Exploiting Map-Based Moving Object Segmentation. *IEEE Robotics and Automation Letters (RA-L)*, 8(8):5180–5187.

Morán, J., Bohlig, P., Bensch, R., Sachsse, L., Parimi, S., Schmid, F., Dey, J., Horokh, O., and Fischer, J. (2025). Ai-assisted multi-sensor fusion for enhanced autonomous vehicle navigation. *ION GNSS+ 2025*.

Qin, C., Ye, H., Pranata, C. E., Han, J., and Liu, M. (2019). LINS: A lidar-inerital state estimator for robust and fast navigation. *CoRR*, abs/1907.02233.

Reichert, H., Serfling, B., Schüssler, E., Turacan, K., Doll, K., and Sick, B. (2025). Real time semantic segmentation of high resolution automotive lidar scans.

Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217.

Rusu, R. B. and Cousins, S. (2011). 3d is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation*, pages 1–4.

Segal, A., Haehnel, D., and Thrun, S. (2009). Generalized-icp. In *Proceedings of Robotics: Science and Systems (RSS)*.

Shan, T. and Englot, B. (2018). Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4758–4765.

Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., and Rus, D. (2020). LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142.

Skrodzki, M. (2019). The k-d tree data structure and a proof for neighborhood computation in expected logarithmic time. *CoRR*, abs/1903.04936.

Xu, W., Cai, Y., He, D., Lin, J., and Zhang, F. (2021). Fast-lio2: Fast direct lidar-inertial odometry.

Zhang, J. and Singh, S. (2014). Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems (RSS)*.