KISS-SLAM: A Simple, Robust, and Accurate 3D LiDAR SLAM System With Enhanced Generalization Capabilities

Tiziano Guadagnino* Benedikt Mersch* Saurabh Gupta Ignacio Vizzo Giorgio Grisetti Cyrill Stachniss

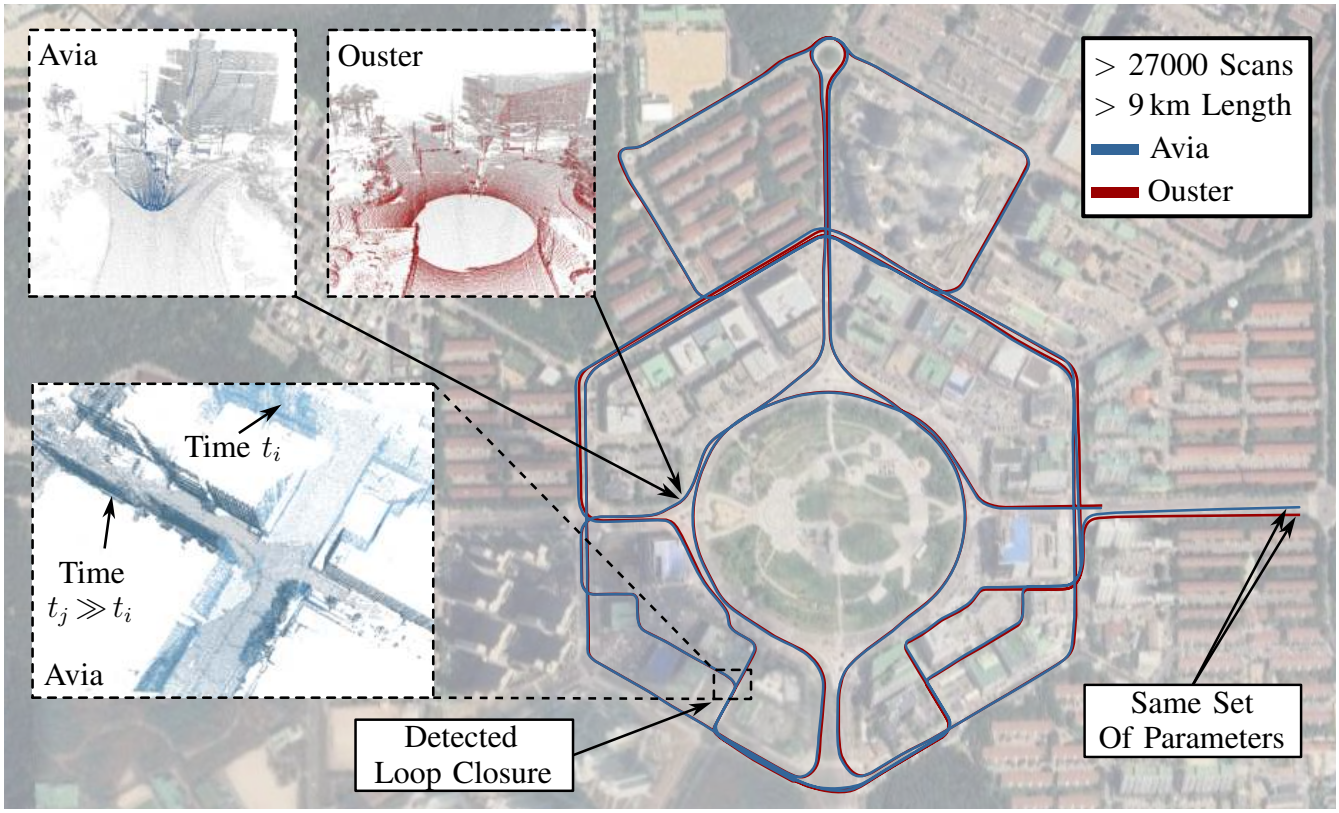

Fig. 1: Our proposed SLAM system can accurately estimate the trajectory for a drive of more than 9 km length in real-time. Using the same parameter configuration, we achieve a similar output for two different LiDAR sensors with different scan patterns and resolutions from the HeLiPR dataset [19]. In both cases, our system can compute the odometry, successfully find loop closures across large time spans, and output a globally consistent trajectory.

***Abstract*— Robust and accurate localization and mapping of an environment using laser scanners, so-called LiDAR SLAM, is essential to many robotic applications. Early 3D LiDAR SLAM methods often exploited additional information from IMU or GNSS sensors to enhance localization accuracy and mitigate drift. Later, advanced systems further improved the estimation at the cost of a higher runtime and complexity. This paper explores the limits of what can be achieved with a LiDAR-only SLAM approach while following the "Keep It Small and Simple" (KISS) principle. By leveraging this minimalist design principle, our system, KISS-SLAM, achieves state-of-the-art performance in pose accuracy while requiring little to no parameter tuning for deployment across diverse environments, sensors, and motion profiles. We follow best practices in graph-based SLAM and build upon LiDAR odometry to compute the relative motion between scans and construct local maps of the environment. To correct drift, we match local maps and optimize the trajectory in a pose graph optimization step. The experimental results demonstrate that this design achieves competitive performance while reducing complexity and reliance on additional sensor modalities. By prioritizing simplicity, this work provides a new strong baseline for LiDAR-only SLAM and a high-performing starting point for future research. Furthermore, our pipeline builds consistent maps that can be used directly for downstream tasks like navigation. Our open-source system operates faster than the sensor frame rate in all presented datasets and is designed for real-world scenarios.**

* Authors contributed equally

Tiziano Guadagnino, Benedikt Mersch, Saurabh Gupta, Ignacio Vizzo, and Cyrill Stachniss are with the University of Bonn, Center for Robotics. Giorgio Grisetti is with Sapienza, University of Rome, Italy. Cyrill Stachniss is additionally with the Lamarr Institute for Machine Learning and Artificial Intelligence.

This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – PhenoRob, by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under STA 1051/5-1 within the FOR 5351 (AID4Crops), and by the German Federal Ministry of Education and Research (BMBF) in the project "Robotics Institute Germany", grant No. 16ME0999.

I. Introduction

Simultaneous Localization and Mapping (SLAM) is an essential building block for any mobile robot autonomously navigating in unknown environments [9], [12]. Several 3D LiDAR SLAM approaches combine multiple sensor sources [27], [29], [38], fusing 3D LiDAR readings with inertial measurement units or GNSS data [39], [40], [41], [42]. This sensor fusion approach helps reduce tracking errors and enhances the precision of pose estimation. However, managing multiple sensors in a sensor fusion pipeline can be challenging due to the need for accurate inter-sensor calibration and time synchronization [34]. Besides that, several recent SLAM systems make use of neural map representations [20], [25], [36], [37] or complex architectures [3], [5], [6], [22]. These pipelines often overfit to the environment and the specific sensor configuration or motion profile in use, requiring extensive parameter tuning or training steps to enhance performance in unseen scenarios.

In this paper, we present KISS-SLAM, a 3D LiDAR-only SLAM pipeline that follows the "Keep It Small and Simple" principle [14], [33]. The main goal is to provide a highly performant SLAM system while minimizing the number of components and parameters. We aim to reduce the system complexity to enhance the generalization of our SLAM system to different environments, sensor resolutions, and motion profiles. Our approach challenges existing geometric SLAM systems and modern deep learning-based solutions. The same system parameters work in various challenging scenarios, such as highway driving of robot cars, handheld devices, and Segways. Furthermore, our method can generalize to different scanning patterns and resolutions.

The main contribution of this paper is a simple yet highly effective approach to LiDAR SLAM that can accurately compute a robot's pose and the corresponding map online while navigating through an environment. We identify and integrate the core components of a SLAM system using established modules and conduct a comprehensive evaluation of the resulting architecture. We show that we obtain highly accurate globally consistent pose estimates while minimizing the number of parameters that require tuning. In sum, we make three key claims: Our "Keep It Small and Simple" SLAM approach (i) is on par or better than state-of-the-art SLAM systems in terms of pose accuracy, (ii) can accurately compute the robot's pose and map in a large variety of environments, sensor characteristics, and motion profiles with the same system configuration, and (iii) we can use its mapping output for robot navigation. The paper and our experimental evaluation back up these claims. We provide an open-source implementation at: `https://github.com/PRBonn/kiss-slam` that precisely follows the description of this paper.

II. Related Work

The development of 3D LiDAR-only SLAM emerged to reduce hardware complexity and enhance applicability in GPS-denied environments [4], [18], [29], [30], [35]. SuMa [1] is a pipeline designed for rotating LiDAR scanners based on rendered views from a surfel map of the environment for data association and loop closure detection. Other approaches like MULLS [24] classify points in a scan based on their geometric features like ground, facades, or pillars and optimize the pose using different loss functions. Direct SLAM pipelines like MD-SLAM unify LiDAR and RGB-D sensor processing through dense photometric alignment, bypassing geometric assumptions and feature extraction [5].

CT-ICP [3] demonstrates that LiDAR-centric SLAM can rival fused systems by refining temporal continuity and motion prediction without inertial data. It integrates loop closures based on features extracted on an elevation grid. Although accurate, CT-ICP requires quite some parameter tuning to operate in unseen environments. Moreover, the system configuration must be adjusted to match the specific motion profile of the platform in use. In contrast, our proposed system does not require parameter tuning based on the platform motion profile.

Loop closing is an essential functionality for 3D LiDAR SLAM. In our system, we leverage the approach of Gupta et al. [15] for loop closing, which utilizes ORB descriptors [28] on a 2D density-based projection of local maps to effectively re-localize the robot. This approach achieves state-of-the-art performances, requiring little to no parameter tuning for unseen scenarios.

Recent advances in neural representations include PIN-SLAM [25], which integrates LiDAR odometry with a probability-based implicit neural mapping framework. This approach achieves accurate 3D reconstruction and loop closure without pose graph optimization by leveraging signed distance fields and uncertainty-aware ray casting. These types of systems have a high computational load and require powerful GPUs to run. As such, these approaches currently do not operate at the sensor frame rate when considering the typical hardware settings available on a real robot. Our system aims to deliver globally accurate pose estimates while operating at a higher frequency than the sensor frame rate.

KISS-ICP [33] achieves accurate 3D LiDAR odometry through minimalistic point-to-point ICP, eliminating the need for IMU integration while achieving real-time performance through adaptive motion compensation. The system performs impressively with little to no parameter tuning required when deployed in an unseen environment. Although KISS-ICP performs well in terms of pose error while tracking [33], the lack of a loop closing and pose graph optimization module limits its performance. This work aims to extend KISS-ICP to a complete SLAM pipeline. Following the "Keep It Small and Simple" principle, we design a system that requires low parameter tuning, delivers robust and accurate performances, and has real-time capabilities.

III. Our Approach to LiDAR-based SLAM

This section presents the main components of KISS-SLAM, our proposed LiDAR-based SLAM system. Given a new LiDAR scan, we first estimate the odometry of the platform based on sensor data in Sec. III-A. We then combine the scan measurement and the ego-motion information into

our local mapping module in Sec. III-B. Upon completing a map, we check for loop closures with previous local maps in Sec. III-C. If a closure is detected, we perform a pose graph optimization of the local map reference frames. Finally, we output the resulting map as a 3D occupancy grid in Sec. III-D. Fig. 2 illustrates our pipeline.

A. LiDAR Odometry Using KISS-ICP

To obtain the pose $\mathsf{T}_t \in \mathbb{SE}(3)$ of the LiDAR sensor in the odometry frame at time t, we first pre-process the incoming point cloud $\mathcal{P} = \{\boldsymbol{p}_i \,|\, \boldsymbol{p}_i \in \mathbb{R}^3\}$ expressed in the sensor frame by de-skewing and voxel downsampling resulting in $\mathcal{S} = \{\boldsymbol{s}_i \,|\, \boldsymbol{s}_i \in \mathbb{R}^3\}$, see [33]. Given the previous estimate of the LiDAR pose T_{t-1} and a constant velocity motion model prediction $\Delta\mathsf{T}_t \in \mathbb{SE}(3)$, we compute an initial guess for the current LiDAR pose as

$$\hat{\mathsf{T}}_t = \mathsf{T}_{t-1}\,\Delta\mathsf{T}_t. \tag{1}$$

We then refine this estimate by using the point-to-point iterative closest point (ICP) algorithm. At each iteration, we obtain a set of correspondences between the source $\mathcal{S}$ and our local map points $\mathcal{Q} = \{\boldsymbol{q}_i \,|\, \boldsymbol{q}_i \in \mathbb{R}^3\}$ which are stored in a voxel grid as in KISS-ICP [33] and updated after registration. We define the residual $\boldsymbol{r}$ between the point $\boldsymbol{q}$ and the point $\boldsymbol{s}$ transformed by T as

$$\boldsymbol{r}(\mathsf{T}) = \mathsf{T}\boldsymbol{s} - \boldsymbol{q}. \tag{2}$$

We then define our point-to-point cost function as:

$$\chi(\hat{\mathsf{T}}_t) = \sum_{(\boldsymbol{s},\boldsymbol{q})\in\mathcal{C}} \left\| \boldsymbol{r}\left(\hat{\mathsf{T}}_t\right) \right\|_2^2, \tag{3}$$

where $\mathcal{C}$ is the set of nearest neighbor correspondences. We can then minimize Eq. (3) in a least squares fashion as:

$$\Delta\boldsymbol{\omega} = \underset{\Delta\boldsymbol{\omega}}{\operatorname{argmin}}\, \chi(\hat{\mathsf{T}}_t \boxplus \Delta\boldsymbol{\omega}), \tag{4}$$

where $\Delta\boldsymbol{\omega} \in \mathbb{R}^6$ is the correction vector, and $\boxplus$ applies the correction vector to the current pose estimate. We repeat this process, including nearest neighbor correspondence search and least squares optimization, until convergence, resulting in the new pose estimate T_t. After convergence, we update the local map with a downsampled version of the registered scan.

B. Local Mapping and Pose Graph Construction

Like several existing methods, our approach avoids maintaining a single global map through a local map-splitting strategy. This design exploits the inherent local consistency of our odometry estimates, which yields precise short-term trajectories within individual map segments. In our system, each local map $\mathcal{M}_k$ contains a keypose reference frame $\mathsf{T}_k \in \mathbb{SE}(3)$, an odometry-derived local trajectory $\mathcal{T}_k \in \{\mathsf{T}_{t\to t+1} \,|\, t_s \le t < t_e\}$, and a keypose-centered voxel grid $\mathcal{V}_k$ containing the local map points. Here, t_s and t_e denote the beginning and end times of the local map's construction.

We first integrate subsampled, deskewed scans into the current local map using the odometry motion estimation presented in Sec. III-A. Next, we evaluate the total distance from the sensor's estimated pose with respect to the keypose T_k of the current local map $\mathcal{M}_k$. We create a new local map when the traveled distance exceeds a certain threshold β. The keypose of this new local map is initialized with the current global pose estimate, which is a combination of the previous keypose and the current odometry estimate. Then, we reset the odometry frame and initialize the new voxel grid with a spatially cropped version of the previous local map.

The system maintains a pose graph where nodes represent keypose-anchored local maps connected through odometric edge constraints. Subsequent sections detail our loop closure detection methodology and explain how we integrate them into the pose graph to enforce global consistency.

C. Loop Closing

After splitting the local maps, we search for loop closures between our last local map and all previous local maps generated. Such loop closures provide additional constraints for the pose graph to correct the drift from odometry toward global consistency.

To search for a closure, we use the approach by Gupta et al. [15], [16]. First, we identify the ground points in the local map and align the point cloud with the xy-plane of the keypose reference frame. Next, we project the local map into a bird's eye view representation by computing the density of projected 3D LiDAR points for each 2D grid cell. We compute binary ORB [28] feature descriptors from the 2D density image and search for matches in a database of descriptors from all previous local map density images. When we find a loop closure candidate, we perform a RANSAC-based geometric validation that provides a 2D alignment of the density images. We combine it with the initial ground alignment to obtain an initial alignment of the matched local maps $\hat{\mathsf{T}}_{i\to j}$.

To ensure that we can effectively use the detected loop closure to correct for odometry drift, we perform a validation step based on the 3D information stored in the local maps. To this end, for each voxel in a local map, we compute the mean $\boldsymbol{\mu}_i$ of the point coordinates and a per-voxel normal vector $\boldsymbol{n}_i$ based on a principal component analysis within the voxel points, resulting in a point cloud $\mathcal{N}_k = \{\{\boldsymbol{\mu}_i, \mathbf{n}_i\} \,|\, \boldsymbol{\mu}_i, \mathbf{n}_i \in \mathbb{R}^3\}$. We apply the initial guess $\hat{\mathsf{T}}_{i\to j}$ and perform a point cloud registration step between the voxel-based point clouds $\mathcal{N}_i$ and $\mathcal{N}_j$ that correspond to the local maps involved in the closure, resulting in a transformation $\mathsf{T}_{i\to j}$. We then compute the Szymkiewicz-Simpson overlap coefficient [32] between the point cloud as:

$$\boldsymbol{\Gamma}(\mathcal{N}_i, \mathcal{N}_j, \mathsf{T}_{i\to j}) = \frac{|\mathcal{N}_j \cap \mathsf{T}_{i\to j} \oplus \mathcal{N}_i|}{\min(|\mathcal{N}_j|, |\mathcal{N}_i|)}, \tag{5}$$

where $\mathsf{T} \oplus \mathcal{N}$ applies the transformation T to the point cloud $\mathcal{N}$, $\cap$ indicates the intersection between the point clouds based on the voxel size, and $|\mathcal{N}|$ is the size of the point cloud. We accept a loop closure in the optimization if

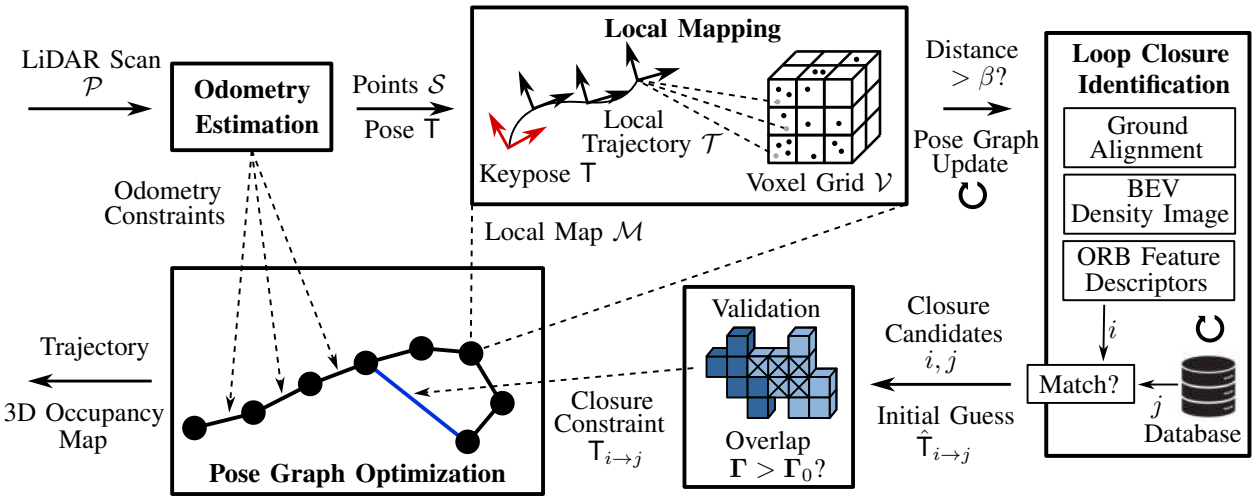

Fig. 2: Overview of KISS-SLAM. Our pipeline processes each LiDAR scan and first computes the odometry of the scanner, see Sec. III-A. Successively, the system integrates the LiDAR point cloud with the motion estimate into a local map, as described in Sec. III-B. When the LiDAR moton exceeds β meters, the system updates the pose graph and searches for loop closures, see Sec. III-C. If they are positively validated based on overlap criteria, we perform a pose graph optimization step.

the value of $\mathbf{\Gamma}$ is above a certain threshold $\mathbf{\Gamma}_0$, which we set fixed to 40 %. In that case, we add a loop closure edge based on $\mathsf{T}_{i\to j}$ and optimize the pose graph to obtain the most up-to-date estimate of the keyposes.

D. Fine-Grained Pose Graph Optimization

After processing all scans, we perform fine-grained pose graph optimization by fixing the nodes corresponding to the keyposes and adding new nodes and edges using the local trajectories stored in the local maps. This way, we can optimize the scan poses within the local maps, allowing us to redistribute minor drift errors among the poses in a local chunk of trajectory. This procedure can be seen as an offline implementation of the approach of Grisetti et al. [10].

IV. Experimental Evaluation

The main focus of this paper is a simple yet highly effective approach for LiDAR SLAM that can accurately estimate a robot's pose and the corresponding map of the environment while the platform navigates through it. The experiments reported here support our key claims, which are that our KISS-SLAM approach is (i) on par or better than state-of-the-art SLAM systems in terms of pose accuracy, (ii) can accurately compute the robot's pose and map in a large variety of environments, sensor characteristics, and motion profiles with the same system configuration, and (iii) we demonstrate how to use the mapping output of the system for a robotics navigation task.

A. Experimental Setup

We employ various datasets and established evaluation methodologies to assess our system's performance. To examine our approach's effectiveness across different autonomous driving datasets utilizing diverse sensors, we evaluate it on the MulRan [21], HeLiPR [19], and Apollo [17] datasets. HeLiPR, in particular, presents four different LiDAR sensors with different ranging technologies and scanning patterns.

Method	2012-01-08
PIN-SLAM	**1.69 / 0.30**
SuMa	316.48 / 50.41
CT-ICP	- / -
MULLS	6.44 / <u>0.59</u>
Ours	<u>3.00</u> / 0.61

TABLE I: Quantitative results on the NCLT dataset, we report the absolute trajectory error (ATE) in meters and the relative KITTI odometry metric in percentage as [m] / [%]. The best and second best performing methods are reported in **bold** and <u>underline</u>, respectively.

Method	DCC	KAIST	Riv	Sej
PIN-SLAM	**3.57 / 1.37**	**2.42** / <u>0.32</u>	**7.55** / 0.46	782.96 / 7.68
SuMa	39.41 / 4.15	26.65 / 1.34	- / -	- / -
CT-ICP	4.03 / 1.43	<u>2.76</u> / 0.50	8.37 / 0.58	- / -
MULLS	27.30 / 1.72	34.21 / 0.60	66.60 / 2.27	1082.55 / 2.53
KISS-ICP	10.72 / <u>1.38</u>	14.00 / **0.31**	37.39 / **0.29**	<u>316.20</u> / <u>0.27</u>
Ours	<u>3.67</u> / 1.44	2.98 / 0.34	<u>7.96</u> / <u>0.40</u>	**178.88 / 0.25**

TABLE II: Quantitative results on the MulRan dataset, we report the ATE in meters and the relative KITTI odometry metric in percentage as [m] / [%]. The best and second best performing methods are reported in **bold** and <u>underline</u>, respectively.

We exclude HeLiPR sequences recorded with the Velodyne VLP-16 because of its self-occlusion with surrounding sensors. Furthermore, we demonstrate our method's versatility by applying it to highly dynamic motion profiles, such as those presented in the NCLT [2] dataset, which uses a Segway platform, and the Newer College dataset [26], captured with a handheld device.

B. Pose Accuracy Evaluation

The first experiment evaluates the performance of our approach. It also supports our first claim, namely that our system reports state-of-the-art performance in pose accuracy. We use the standard ATE to measure absolute pose estimation performances. We use the KITTI metric [7] to assess the relative error. We use the open-source implementation from

Method	Bri Aeva	Bri Avia	Bri Ouster	Rou Aeva	Rou Avia	Rou Ouster	To Aeva	To Avia	To Ouster
PIN-SLAM	-/-	365.72/2.15	**19.23**/**0.13**	-/-	7.02/0.51	1.47/**0.11**	41.19/1.03	**11.40**/0.64	2.55/**0.12**
SuMa	-/-	-/-	140.01/1.08	-/-	-/-	14.88/0.46	132.17/3.16	-/-	-/-
CT-ICP	-/-	**41.47**/**0.36**	579.62/0.89	10.04/0.72	**3.36**/0.25	1.81/0.16	65.29/1.37	63.72/2.02	-/-
MULLS	356.06/9.17	321.87/4.06	52.65/0.68	19.08/1.43	16.39/1.06	2.65/0.24	39.82/2.85	14.93/1.35	4.45/0.27
Ours	**98.61**/**1.57**	148.88/1.82	19.47/0.28	**6.06**/**0.47**	3.84/**0.23**	**1.18**/0.17	**14.44**/**0.85**	12.01/**0.46**	**1.99**/0.21

TABLE III: Quantitative results on the Bridge (Bri), Roundabout (Rou) and Town (To) sequences of the HeLiPR dataset, we report the ATE in meters and the relative KITTI odometry metric in percentage as [m] / [%]. The best and second best performing methods are reported in **bold** and underline, respectively.

Method	BTS 2018-10-12	CP 2018-10-11	H237 2018-10-12	MAVE 2018-10-12	SB 2018-10-03	SJD 2018-10-11 1	SJD 2018-10-11 2
PIN-SLAM	6.10/0.24	**0.64**/**0.16**	97.11/1.66	**7.06**/0.20	**2.23**/**0.25**	1.86/0.22	1.15/0.18
SuMa	181.19/3.55	-/-	366.66/9.51	79.68/0.43	-/-	-/-	-/-
CT-ICP	10.81/0.27	0.97/0.30	258.87/0.62	61.39/0.33	667.28/1.60	**0.78**/**0.16**	1.09/**0.17**
MULLS	104.14/0.27	47.03/0.90	354.21/**0.39**	182.59/**0.16**	-/-	13.38/0.32	11.41/0.56
Ours	**3.74**/**0.23**	0.80/0.22	**30.16**/0.40	9.08/0.26	2.51/**0.25**	2.00/0.17	**0.66**/**0.17**

TABLE IV: Quantitative results on the Apollo dataset, we report the ATE in meters and the relative KITTI odometry metric in percentage as [m] / [%]. The best and second best performing methods are reported in **bold** and underline, respectively.

Method	2020 01-short	2020 02-long	2021 cloister	2021 math e	2021 quad e	2021 stairs	2021 underground e
PIN-SLAM	0.42/0.33	**0.31**/**0.22**	0.15/0.62	**0.08**/0.29	**0.09**/0.12	**0.06**/n.a.	**0.07**/n.a.
SuMa	2.06/9.11	5.77/3.04	0.17/0.79	0.16/0.32	0.21/0.66	1.85/n.a.	0.11/n.a.
CT-ICP	0.63/0.42	25.06/3.01	0.17/0.31	0.09/0.14	0.19/0.14	-/n.a.	0.15/n.a.
MULLS	0.47/**0.17**	8.47/4633.62	**0.13**/**0.17**	0.13/**0.02**	0.16/0.34	1.82/n.a.	0.69/n.a.
Ours	**0.30**/0.26	1.58/2.14	0.40/1.18	0.15/0.61	0.16/**0.03**	3.58/n.a.	0.12/n.a.

TABLE V: Quantitative results on the Newer College dataset, we report the ATE in meters and the relative KITTI odometry metric in percentage as [m] / [%]. The best and second best performing methods are reported in **bold** and underline, respectively. Note that in some cases, the relative KITTI metric is not applicable (n.a.) due to the short trajectory length.

the *evo* package [13] instead of a custom implementation for both metrics. We do this to standardize the experimental evaluation of the results and simplify reproducibility. For the MulRan and HeLiPR datasets, we report the average value of the metrics over each scene, as for both datasets, three different runs are performed per scene.

To assess the performance of our proposed SLAM system, we compare the results to various state-of-the-art open-source LiDAR-only SLAM systems such as SuMa [1], MULLS [24], CT-ICP [3], and PIN-SLAM [25]. We report the results in Tab. I to Tab. V. Note that we consider a run failed (-) if the errors exceed a sequence-specific threshold. The results show that our system consistently delivers state-of-the-art performance in terms of pose accuracy, regardless of the motion profile, scanning pattern, and sensor resolution, often ranking first or second. Furthermore, KISS-SLAM is the only approach that can effectively compute the poses in all the reported scenarios. For example, in Tab. III, we can see how our SLAM pipeline can successfully run on the challenging Aeva sensor in HeLiPR, showcasing the robustness and versatility of our approach, even compared with more sophisticated neural SLAM techniques like PIN-SLAM. Additionally, Tab. II compares our approach with KISS-ICP [33], showing the benefits of a full SLAM pipeline over odometry-only methods in terms of pose estimation accuracy. Note that we obtain all our results with the same system configuration, with no parameter tuning applied.

Furthermore, KISS-SLAM can compute the pose and map estimate above the sensor frame rate. In contrast, the closest baseline regarding pose accuracy, PIN-SLAM, needs to balance runtime and performance. As investigated by Pan et al. [25], the PIN-SLAM configuration needed to achieve the level of pose accuracy that we report in our analysis does not allow the pipeline to run at sensor frame rate on a single NVIDIA A4000 GPU.

C. Ablation on Parameter Change

This experiment supports our second claim that our system can accurately estimate the robot's pose and map in many environments, sensor characteristics, and motion profiles with the same configuration. In contrast, existing SLAM approaches usually require fine-tuning parameters to work successfully with different setups.

Note that this section does not evaluate the total number of parameters because this quantity is not straightforward to measure. In fact, the number of parameters exposed in a configuration file may not include all parameters, as some are set as constants in the implementation.

In this experiment, we count the parameters that had to be changed between different runs to achieve the results reported in Tab. III and Tab. V, because they impose challenging changes in the data. For HeLiPR, we can compare multiple sensor resolutions and scanning patterns using the Avia, Aeva, and Ouster data recorded with the same motion

	HeLiPR Ae ↕ HeLiPR Av	HeLiPR Ae ↕ HeLiPR O	HeLiPR Av ↕ HeLiPR O	HeLiPR O ↕ NCD-2020
PIN-SLAM	12	11	13	23
SuMa	7	9	9	19
CT-ICP	16	16	16	16
MULLS	-	-	**0**	25
Ours	**0**	**0**	**0**	**0**

TABLE VI: Number of changed parameters between configurations for different sensors and motion profiles. For HeLiPR, "Ae" denotes the Aeva sensor, "Av" is Avia, and "O" is the Ouster scanner. Note that "-" means that there is no working set of parameters for at least one of the sequences. The lower the number of changed parameters, the easier it is to run the approach on new data. The lowest number of changed parameters is in **bold**.

profile in the same environment. We include data recorded with a substantially different motion profile with the Newer College handheld dataset. Note that we also consider data-dependent parameters because, typically, these values influence other parameters in the pipeline [25], [33].

We can see in Tab. VI that multiple parameters must be modified for all baselines when running the approaches on the different sensors of the HeLiPR dataset. We also evaluate the number of parameters that must be adjusted when running the approach on data acquired by an autonomous vehicle versus a dataset recorded with a handheld device. Again, all baselines require a substantial change in the configuration, whereas our approach works out of the box. This experiment illustrates the robust generalization capabilities of our SLAM method, which maintains high performance across diverse sensor configurations, motion profiles, and environments without requiring manual parameter tuning.

D. Navigation Experiment on Real Robot

Finally, our last experiment supports our third claim that we can use the output of KISS-SLAM to perform robotic navigation tasks. To illustrate the usability of a global map computed using the pose estimates from our approach, we build a 2D occupancy grid map of an office-like environment and perform a global localization experiment. This demonstrates the general applicability of our approach when using different platforms for mapping and localization.

We start by recording a sequence of 3D LiDAR scans using a Hesai XT-32 scanner mounted on a Clearpath Husky robot. We run KISS-SLAM on this sequence and generate a 3D occupancy grid map of the environment, with a resolution of $0.05\,\text{m}$ per voxel, following a standard occupancy grid mapping algorithm [31]. For this experiment, we reduce the maximum range of the points that KISS-SLAM processes from $100\,\text{meters}$ to $50\,\text{meters}$, as the office environment is relatively small compared to an outdoor scene. Other than this value, we leave the pipeline configuration untouched.

We would like to point out that KISS-SLAM could run faster than the sensor frame rate on the robot's onboard computer, an Intel NUC equipped with an Intel i7 processor and $32\,\text{GB}$ of RAM. Additionally, we run the grid mapping pipeline on the same computer.

To create a 2D occupancy grid, we process a horizontal slice of the 3D occupancy map within a specific height range, $[z_{min}, z_{max}]$. We assign the 2D cells occupancy values based on the maximum occupancy within the corresponding vertical column of the horizontal slice. Once we create a map this way, we aim to globally localize and track the pose of a second robot, the Clearpath Dingo, equipped with a SICK TiM781S 2D LiDAR. Since the 2D LiDAR is mounted at $0.16\,\text{m}$ above the ground, we initially set $z_{min} = 0.1\,\text{m}$ and $z_{max} = 0.2\,\text{m}$ for the 2D occupancy grid construction. The resulting maps are depicted in Fig. 3. The final 2D map is shown in Fig. 3. We use a Monte-Carlo localization (MCL) implementation from the Sapienza Robots Vision and Perception group (RVP) [8], which we abbreviate here as RVP-Loc, to perform localization with the Dingo equipped with a 2D LiDAR. To measure the quality of our occupancy map, we compare the localization performances of RVP-Loc obtained using our map against an occupancy grid generated with GMapping [11]. This second map is based on data recorded in the same environment but with the Dingo robot and its 2D LiDAR.

For our evaluation, we assess global localization performance and pose-tracking accuracy after convergence. To achieve this, we initialize RVP-Loc with 10,000 particles uniformly distributed across the free space in the occupancy map. We define the convergence criterion based on the localizer's reported pose covariance. Specifically, we require the standard deviation of the largest principal component of the translational part to be below $0.2\,\text{m}$ and the standard deviation of the heading angle to be under $10°$. We measure the convergence time that is required for the estimate to converge. After achieving convergence, we evaluate pose-tracking accuracy using ground truth poses derived from AprilTags [23] placed on the ceiling of the office, captured with an upward-looking camera. To ensure robustness, we run RVP-Loc 10 times on each sequence, each time with a randomly initialized seed. We provide the results, including the mean and standard deviation over these 10 runs for each sequence in Tab. VII. The results show no significant difference between the two map representations, showcasing that our system can effectively generate a map suitable for global localization and tracking, validating the real-world applicability of KISS-SLAM.

V. Conclusion

This paper presents KISS-SLAM, a simple yet highly effective approach to LiDAR SLAM. Our approach operates solely on LiDAR scans and does not require additional sensors to compute the robot's trajectory and map of the environment. Our approach exploits a minimalist design that can be employed in different challenging environments, such as highway driving, handheld devices, and Segways. Moreover, the system is not tailored to specific range-sensing technologies or scanning patterns. We only assume that point clouds are generated sequentially as the robot moves through the environment. We implemented and evaluated our approach on different datasets, provided comparisons to other existing techniques, supported all claims made in

Sequence	SLAM Method	Global Localization		Pose Tracking: Absolute Trajectory Error (ATE)			
		Convergence Time [s] ↓	Success Rate [%] ↑	ATE translation [cm] RMS ↓	ATE translation [cm] Max ↓	ATE rotation [°] RMS ↓	ATE rotation [°] Max ↓
Static Sequence 1	GMapping	11.27 ± 5.68	100	**9.48** ± 0.05	16.97 ± **0.11**	**1.89** ± **0.01**	**6.49** ± **0.07**
	Ours	**10.34** ± **4.14**	100	9.71 ± **0.02**	**16.01** ± 0.13	1.97 ± **0.01**	7.81 ± **0.07**
Static Sequence 2	GMapping	**5.68** ± 1.37	100	9.88 ± **0.02**	**15.70** ± **0.20**	**1.77** ± **0.01**	**5.33** ± **0.06**
	Ours	6.19 ± **0.88**	100	**9.26** ± 0.05	17.85 ± 2.45	1.90 ± **0.01**	7.61 ± **0.06**
Static Sequence 3	GMapping	**10.51** ± 1.93	100	9.68 ± **0.13**	16.42 ± **0.11**	1.80 ± **0.01**	**4.99** ± **0.07**
	Ours	11.73 ± **1.73**	100	**8.51** ± 0.20	**14.23** ± 0.43	**1.79** ± 0.06	5.55 ± 0.66
Dynamic Sequence 1	GMapping	11.81 ± 5.05	90	**5.71** ± 0.18	**11.06** ± 0.37	**1.81** ± **0.01**	**6.74** ± 0.09
	Ours	**11.25** ± **1.87**	90	9.57 ± **0.12**	17.54 ± **0.08**	1.99 ± **0.01**	6.80 ± **0.07**
Dynamic Sequence 2	GMapping	**14.06** ± **5.52**	100	**7.43** ± 0.10	**17.60** ± 0.36	**2.38** ± **0.03**	7.99 ± **0.08**
	Ours	22.66 ± 7.34	100	10.42 ± **0.05**	23.73 ± **0.17**	2.67 ± 0.05	**7.92** ± 0.11

TABLE VII: Quantitative evaluation of 2D localization performance using our SLAM-generated 2D occupancy grid map and comparison with a GMapping grid map. The evaluation is based on five sequences recorded in an office environment containing static and dynamic scenes. We report localization metrics' mean and standard deviation over 10 runs of RVP-Loc. The best results are in **bold**.

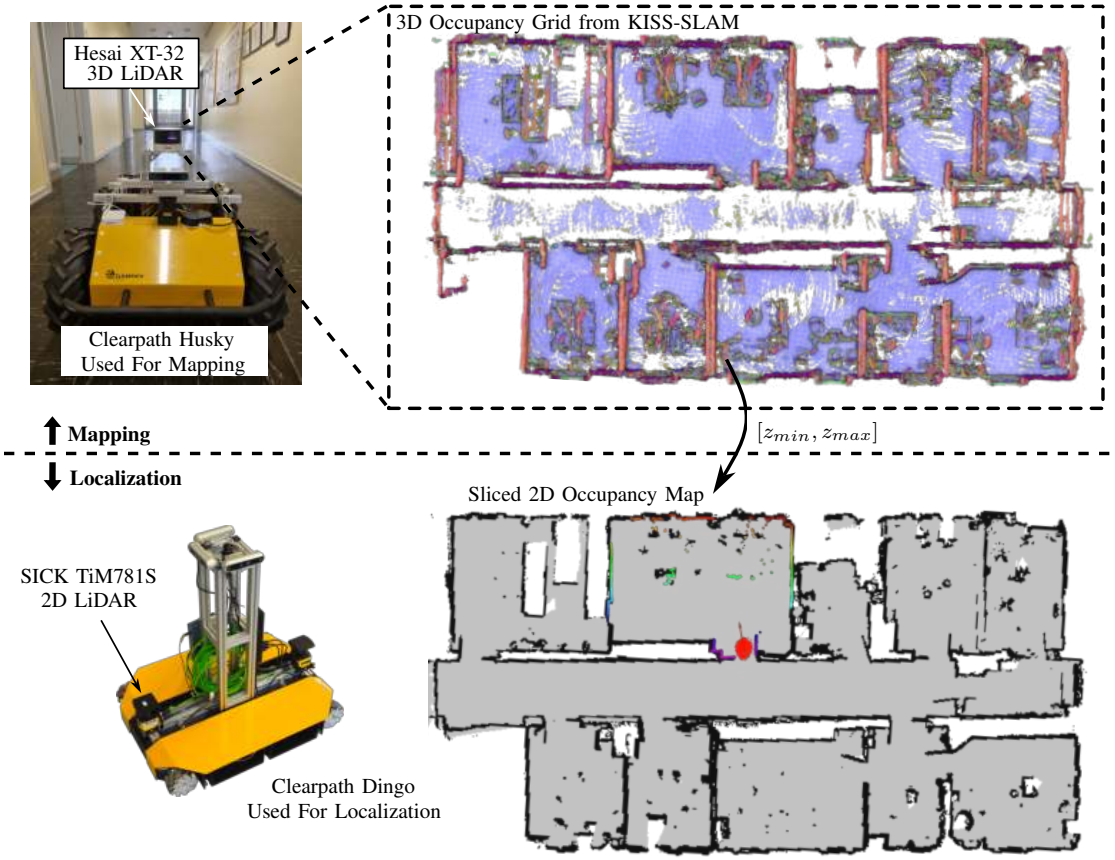

Fig. 3: Occupancy grid maps generated with our approach. The top shows the 3D occupancy grid we generate using KISS-SLAM, after processing all the 3D scans from an Hesai XT-32 with a Clearpath Husky. The bottom displays the corresponding 2D grid generated by slicing the 3D map. We then use this map to perform 2D localization using the approach of Grisetti et al. [8].

this paper, and released our code. The experiments suggest that our approach is on par or better than substantially more sophisticated state-of-the-art LiDAR SLAM systems. Yet, it relies only on a few parameters and performs well on various datasets under different conditions with the same parameter set. Furthermore, the system's output can be effectively used to perform downstream tasks like robot navigation. Finally, our system operates faster than the sensor frame rate in all presented datasets. We believe this work will be a new baseline for future LiDAR SLAM systems and a high-performing starting point for future approaches. Our open-source code is robust and simple, easy to extend, and performs well, pushing the state-of-the-art LiDAR SLAM to its limits and challenging more sophisticated systems.

ACKNOWLEDGMENTS

We thank Luca Lobefaro, Meher Malladi, and Niklas Trekel for fruitful discussions and Jens Behley and Yue Pan for providing the baseline results.

REFERENCES

[1] J. Behley and C. Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[2] N. Carlevaris-Bianco, A. Ushani, and R. Eustice. University of Michigan North Campus long-term vision and lidar dataset. *Intl. Journal of Robotics Research (IJRR)*, 35(9):1023–1035, 2016.

[3] P. Dellenbach, J. Deschaud, B. Jacquet, and F. Goulette. CT-ICP Real-Time Elastic LiDAR Odometry with Loop Closure. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[4] J. Deschaud. IMLS-SLAM: scan-to-model matching based on 3D data. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[5] L. Di Giammarino, L. Brizi, T. Guadagnino, C. Stachniss, and G. Grisetti. Md-slam: Multi-cue direct slam. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[6] L. Di Giammarino, E. Giacomini, L. Brizi, O. Salem, and G. Grisetti. Photometric LiDAR and RGB-D Bundle Adjustment. *IEEE Robotics and Automation Letters (RA-L)*, 8(7):4362–4369, 2023.

[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets Robotics: The KITTI Dataset. *Intl. Journal of Robotics Research (IJRR)*, 32(11):1231–1237, 2013.

[8] G. Grisetti. srrg-localizer2d (1.6.0). `https://gitlab.com/srrg-software/srrg_localizer2d`, 2018.

[9] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Trans. on Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[10] G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg. Hierarchical Optimization on Manifolds for Online 2D and 3D Mapping. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2010.

[11] G. Grisetti, C. Stachniss, and W. Burgard. Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. *IEEE Trans. on Robotics (TRO)*, 23(1):34–46, 2007.

[12] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel. Least Squares Optimization: From Theory to Practice. *Robotics*, 9(3), 2020.

[13] M. Grupp. evo: Python package for the evaluation of odometry and SLAM. `https://github.com/MichaelGrupp/evo`, 2017.

[14] T. Guadagnino, B. Mersch, I. Vizzo, S. Gupta, M. Malladi, L. Lobefaro, G. Doisy, and C. Stachniss. Kinematic-ICP: Enhancing LiDAR Odometry with Kinematic Constraints for Wheeled Mobile Robots Moving on Planar Surfaces. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2025.

[15] S. Gupta, T. Guadagnino, B. Mersch, I. Vizzo, and C. Stachniss. Effectively Detecting Loop Closures using Point Cloud Density Maps. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.

[16] S. Gupta, T. Guadagnino, B. Mersch, N. Trekel, M.V.R. Malladi, and C. Stachniss. Efficiently Closing Loops in LiDAR-Based SLAM Using Point Cloud Density Maps. *arXiv preprint*, arXiv:2501.07399, 2025.

[17] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The ApolloScape Dataset for Autonomous Driving. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, 2018.

[18] B. Jiang and S. Shen. Contour Context Abstract Structural Distribution for 3D LiDAR Loop Detection and Metric Pose Estimation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[19] M. Jung, W. Yang, D. Lee, H. Gil, G. Kim, and A. Kim. HeLiPR: Heterogeneous LiDAR Dataset for inter-LiDAR Place Recognition under Spatiotemporal Variations. *Intl. Journal of Robotics Research (IJRR)*, 12(43):1867—-1883, 2024.

[20] S. Kelly, A. Riccardi, E. Marks, F. Magistri, T. Guadagnino, M. Chli, and C. Stachniss. Target-Aware Implicit Mapping for Agricultural Crop Inspection. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[21] G. Kim, Y. Park, Y. Cho, J. Jeong, and A. Kim. Mulran: Multimodal range dataset for urban place recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[22] Z. Liu and F. Zhang. BALM: Bundle Adjustment for Lidar Mapping. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3184–3191, 2021.

[23] E. Olson. AprilTag: A Robust and Flexible Visual Fiducial System. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2011.

[24] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li. MULLS: Versatile LiDAR SLAM Via Multi-Metric Linear Least Square. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[25] Y. Pan, X. Zhong, L. Wiesmann, T. Posewsky, J. Behley, and C. Stachniss. PIN-SLAM: LiDAR SLAM Using a Point-Based Implicit Neural Representation for Achieving Global Map Consistency. *IEEE Trans. on Robotics (TRO)*, 40:4045–4064, 2024.

[26] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon. The newer college dataset: Handheld lidar, inertial and vision with ground truth. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[27] J. Ruan, B. Li, Y. Wang, and Z. Fang. GP-SLAM+: Real-Time 3D Lidar Slam Based on Improved Regionalized Gaussian Process Map Reconstruction. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[28] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2011.

[29] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[30] T. Shan and B. Englot. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[31] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[32] M. Vijaymeena and K. Kavitha. A Survey on Similarity Measures in Text Mining. *Machine Learning and Applications: An International Journal*, 3(1):19–28, 2016.

[33] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023.

[34] I. Vizzo, B. Mersch, L. Nunes, L. Wiesmann, T. Guadagnino, and C. Stachniss. Toward Reproducible Version-Controlled Perception Platforms: Embracing Simplicity in Autonomous Vehicle Dataset Acquisition. In *Worshop on Building Reliable Ratasets for Autonomous Vehicles, IEEE Intl. Conf. on Intelligent Transportation Systems (ITSC)*, 2023.

[35] X. Wei, J. Lv, J. Sun, E. Dong, and S. Pu. GCLO: Ground Constrained LiDAR Odometry with Low-drifts for GPS-denied Indoor Environments. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[36] L. Wiesmann, T. Guadagnino, I. Vizzo, G. Grisetti, J. Behley, and C. Stachniss. DCPCR: Deep Compressed Point Cloud Registration in Large-Scale Outdoor Environments. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):6327–6334, 2022.

[37] L. Wiesmann, T. Guadagnino, I. Vizzo, N. Zimmerman, Y. Pan, H. Kuang, J. Behley, and C. Stachniss. LocNDF: Neural Distance Field Mapping for Robot Localization. *IEEE Robotics and Automation Letters (RA-L)*, 8(8):4999–5006, 2023.

[38] W. Wu, X. Zhong, D. Wu, B. Chen, X. Zhong, and Q. Liu. LIO-Fusion: Reinforced LiDAR Inertial Odometry by Effective Fusion With GNSS/Relocalization and Wheel Odometry. *IEEE Robotics and Automation Letters (RA-L)*, 8(3):1571–1578, 2023.

[39] Y. Wu, T. Guadagnino, L. Wiesmann, L. Klingbeil, C. Stachniss, and H. Kuhlmann. LIO-EKF: High Frequency LiDAR-Inertial Odometry using Extended Kalman Filters. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.

[40] W. Xu and F. Zhang. FAST-LIO: A Fast, Robust LiDAR-Inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3317–3324, 2021.

[41] C. Zheng, W. Xu, Z. Zou, T. Hua, C. Yuan, D. He, B. Zhou, Z. Liu, J. Lin, F. Zhu, Y. Ren, R. Wang, F. Meng, and F. Zhang. FAST-LIVO2: Fast, Direct LiDAR–Inertial–Visual Odometry. *IEEE Trans. on Robotics (TRO)*, 41:326–346, 2025.

[42] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang. FAST-LIVO: Fast and Tightly-coupled Sparse-Direct LiDAR-Inertial-Visual Odometry. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.