

# SC-LiDAR-SLAM: a Front-end Agnostic Versatile LiDAR SLAM System

1<sup>st</sup> Giseop Kim  
*Dept. of Civil and Envtl. Eng.*  
*KAIST*  
 Daejeon, South Korea  
 paulgkim@kaist.ac.kr

2<sup>nd</sup> Seungsang Yun  
*Robotics Program*  
*KAIST*  
 Daejeon, South Korea  
 seungsang@kaist.ac.kr

3<sup>rd</sup> Jeongyun Kim  
*Dept. of Civil and Envtl. Eng.*  
*KAIST*  
 Daejeon, South Korea  
 jungyun0609@kaist.ac.kr

4<sup>th</sup> Ayoung Kim\*  
*Dept. of Mechanical Eng.*  
*Seoul National University*  
 Seoul, South Korea  
 ayoungk@snu.ac.kr

**Abstract**—Accurate 3D point cloud map generation is a core task for various robot missions or even for data-driven urban analysis. To do so, light detection and ranging (LiDAR) sensor-based simultaneous localization and mapping (SLAM) technology have been elaborated. To compose a full SLAM system, many odometry and place recognition methods have independently been proposed in academia. However, they have hardly been integrated or too tightly combined so that exchanging (upgrading) either single odometry or place recognition module is very effort demanding. Recently, the performance of each module has been improved a lot, so it is necessary to build a SLAM system that can effectively integrate them and easily replace them with the latest one. In this paper, we release such a front-end agnostic LiDAR SLAM system, named SC-LiDAR-SLAM. We built a complete SLAM system by designing it modular, and successfully integrating it with Scan Context++ and diverse existing open-source LiDAR odometry methods to generate an accurate point cloud map.

**Index Terms**—LiDAR, Point cloud, SLAM

## I. INTRODUCTION

Point cloud [1] is a very primitive yet direct representation that could accurately render 3D data such as objects or surrounding structural environments [2]. Large-scale reconstruction of a 3D real-world environment represented in a point cloud, in particular, is required for a wide range of applications from autonomous robot navigation [3] to data-driven urban analysis [4].

Recent advances in LiDAR sensor-based robotic mapping technology have made it possible to create a dense and precise 3D point cloud map [5] (e.g., Fig. 1). Its general pipeline, typically known as SLAM [6], is constituted of three independent modules: odometry [7, 8, 9], place recognition [10, 11, 12], and pose-graph optimization [13, 14, 15]. Seamless integration of the aforementioned three modules is particularly critical for map accuracy because standalone odometry is prone to drifts in motion estimation. Many improved LiDAR odometry approaches have recently been proposed [16, 17, 18, 19] to overcome this motion estimation error, but it remains unavoidable.

† This research was supported by a grant(21TSRD-B151228-03) from Urban Declining Area Regenerative Capacity-Enhancing Technology Research Program funded by Ministry of Land, Infrastructure and Transport of Korean government.

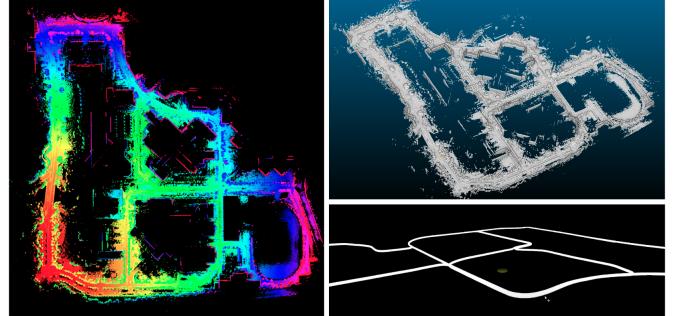


Fig. 1: An example result of the proposed LiDAR SLAM system on KAIST sequence of MulRan dataset. We can see the point cloud map (left: top-view, right-up: side-view) is qualitatively precise and its optimized trajectory (right-bottom) is well-aligned without drifts particularly on the z-axis.

However, there are few open-source LiDAR SLAM projects that have integrated the entire modules, especially the lack of global place recognition (i.e., recognizing previously visited places without an initial information, also called kidnapped robot problem). Thus, current publicly available LiDAR SLAM methods are still vulnerable to accumulated drifts and noisy map generation (e.g., the before case in Fig. 6(c)). To resolve the accumulated errors, a set of long-term data associations made by place recognition is vital.

In this paper, we propose a front-end agnostic LiDAR SLAM system named SC-LiDAR-SLAM and release it as an open-source publicly available. SC-LiDAR-SLAM is *front-end agnostic*, which means it can build a precise 3D point cloud map using any LiDAR odometry method an user prefers with absolutely no additional efforts. Based on our modular implementation design, Scan Context [11, 20] is chosen for global place recognition and internally incorporated with the generic input data from any given odometry method. Finally, SC-LiDAR-SLAM is bundled with useful utilities such as data saver, which stores SLAM results that could be plugged into other robot missions.

Our contributions are threefolds.

- We open a LiDAR SLAM system called SC-LiDAR-SLAM to the public. To the best of our knowledge, this is

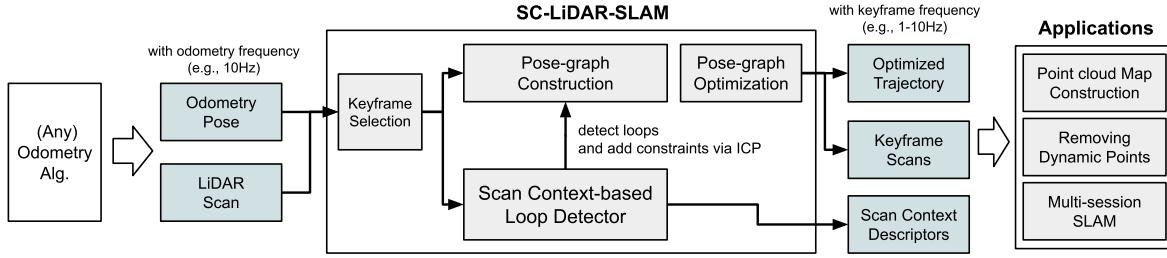


Fig. 2: The overall pipeline of the proposed robotic mapping system, named SC-LiDAR SLAM.

the first front-end agnostic LiDAR SLAM project made available to the academic community.

- To verify the versatility of SC-LiDAR-SLAM, we demonstrate many integrated results in various LiDAR odometry methods and multiple environments.
- We provide multiple tools (e.g., to save file-based outputs) to support a set of plug-and-play-based other robot applications such as dynamic point removal or multi-session map building that take our SLAM outputs without additional data conversion efforts, in addition to our robust and versatile LiDAR SLAM system.

## II. BACKGROUNDS

A simple yet complete form of a modern SLAM system can be defined as [6]:

$$\begin{aligned} \text{SLAM} = & \text{Odometry} \\ & + \text{Place Recognition} \\ & + \text{Pose-graph Optimization} . \end{aligned} \quad (1)$$

We present here existing works of each component based on the aforementioned notion of SLAM.

First, odometry or ego-motion estimation is a backbone of a SLAM system, which estimates consecutive relative motions while a robot moves in space. Raw (or downsampled) point cloud registration famously known as Iterated Closest Point (ICP) [21, 22] or feature-based methods represented by LOAM [23, 24] have been contributed to many recent advanced LiDAR odometry methods showing more accurate results [25, 7, 16, 17, 18]. Despite these gains in accuracy, it is still laborious to avoid the accumulation of the motion estimation errors. This accumulated error causes inconsistency between point cloud maps when a robot revisits the same place. Thus, we need to minimize the error by linking temporally distant places and adding a constraint between them. This procedure is called loop closing and it is carried out using place recognition.

Driven by this need, diverse LiDAR sensor-based place recognition methods have been proposed [10, 26, 27, 11, 12]. Recently, in our previous work, we proposed a rotation and lateral invariant point cloud descriptor named Scan Context++ [20]. It summarizes a single LiDAR scan into two perpendicular axes which are invariant to and variant to robot motions (e.g., lane change or reversed heading) in urban road environments. Scan Context++ takes highest  $z$  value of points

to capture an ego-centric context of a point cloud this encoding function is inspired by the ubran analysis concept Isovor [4]

Despite the progress in each odometry and place recognition field, however, their improvements have commonly been isolated and rarely been integrated to construct a state-of-the-art SLAM system following the definition of (1). Alternatively, in the case of the exisiting LiDAR SLAM system (e.g., [26, 28]), odometry and place recognition modules are tightly entangled, making it difficult to replace or update one improved module at a low cost. To overcome these limitations, in this paper, we propose a front-end (especially odometry) agnostic modular LiDAR SLAM system, SC-LiDAR-SLAM. The details of our framework are followed in the next section and in Fig. 2.

## III. SC-LIDAR-SLAM

### A. Overview

The overall pipeline is depicted in Fig. 2. First, an initial pose-graph is being constructed using a stream of inputs (i.e., relative motion estimation and raw LiDAR scan) from any existing LiDAR odometry method and any LiDAR sensor. Second, thanks to the powerful revisit detection performance of Scan Context, a set of constraints between temporally apart locations could be added to the pose-graph. Finally, an efficient pose-graph optimization [14] is followed, and the optimized poses with of places' data are saved. Those optimized poses and the placewise data such as LiDAR scan acquired a a place or a place descriptor (i.e., Scan Context Descriptor) would be effective materials for next robot missions (e.g., §IV-C and §IV-C).

### B. Pose-graph Construction

Based on the SLAM definition (1), the robot poses are managed by a pose-graph [29, 30, 31, 32, 33] without optimizing landmarks. Before optimizing the pose variables in a pose-graph, pose-graph construction is preceded. The pose-graph construction can be classified in two types.

1) *Short-term Data Association*: Short-term data association is a process of initializing a new pose variable and connecting it with its consecutive previous variable. This sequential constraint is conducted by any LiDAR odometry methods. However, this set of relative motion estimations is prone to have errors, which need to be resolved by long-term data association, called loop closing.

2) *Long-term Data Association*: Long-term data association entails creating a virtual observation constraint between two poses  $X_j$  and  $X_k$  that are separated in time. The necessity of this association is recognized by the global place recognition module, here, Scan Context++ [20]. Scan Context++ summarizes a raw point cloud of a keyframe into a fixed-size of descriptor for fast retrieval. It is rotation and lateral invariant, making it suitable for robotic mapping in urban areas with changing revisit conditions.

After the previously visited place  $j$  is detected, a physical 6D constraint  $z_{jk}$  (i.e., the relative translation and orientation between the two poses) is made via ICP by registering a query LiDAR scan at  $X_k$  to a submap point cloud surrounding the matched pose  $X_j$ . Finally, this constraint is added to the pose-graph. It is required to minimize the accumulated odometry errors over a traverse. This long-term data association is generally called *loop closing* because it is made when a robot detects previously visited places when it forms its trajectory as a loop, and the robot "closes" the errors to make a complete and well-aligned loop.

3) *Pose-graph Optimization*: Pose-graph optimization step finds the best solution of the set of poses based on the short-term or the long-term constraints in a pose-graph. This optimization is a MAP (maximum a posteriori) inference [34] that minimizes a sum of nonlinear least-square errors. For example, the cost function can be written as

$$X^{\text{MAP}} = \underset{X}{\operatorname{argmin}} \sum_i \|h(X_i, X_{i+1}) - z_i\|_{\Sigma_o}^2 + \sum_{(j,k)} \|s_{jk}(h(X_j, X_k) - z_{jk})\|_{\Sigma_l}^2, \quad (2)$$

where  $i$  is an index of a pose.  $j$  and  $k$  is a pair of indexes of a detected loop by Scan Context++.  $s_{jk}$  is a scaling factor [35] for a loop factor between the pose  $j$  and  $k$ . It is used only for loop factors for minimizing the effect of spurious loops (i.e., falsely recognized places), which may trigger catastrophic map collapses. Commonly used M-estimators such as Cauchy or Huber kernels [36, 37] can be also adopted for this purpose instead of the dynamic scaling.  $h(\cdot)$  is a relative transformation between the two argument poses.  $z_i$  and  $z_{jk}$  are measured by odometry and ICP after the place recognition.  $\Sigma_o$  and  $\Sigma_l$  are covariance matrices for odometry and loop constraints, which determine the uncertainties of each measurement.

To solve the above problem, a traditional Gauss-Newton style iterative update [38, 39] or iSAM-based [40, 14] solvers are used.

### C. Open-sources and Utility

We support to save SLAM outputs in compatible formats to our other robot application softwares such as dynamic point removal [41] or multi-session map merging [42]. The stored file structure is visualized in Fig. 3. Also, we prepare a set of tools such as 3D map maker as in Fig. 4 using the optimized poses and LiDAR scans saved as files.

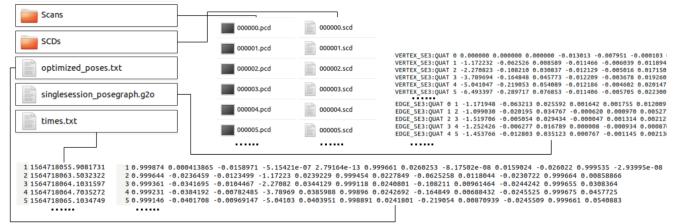


Fig. 3: A data structure of SC-LiDAR-SLAM's saved outputs are visualized. These data (i.e., LiDAR point cloud scans, the optimized trajectory, and place descriptors) are stored place-wise.

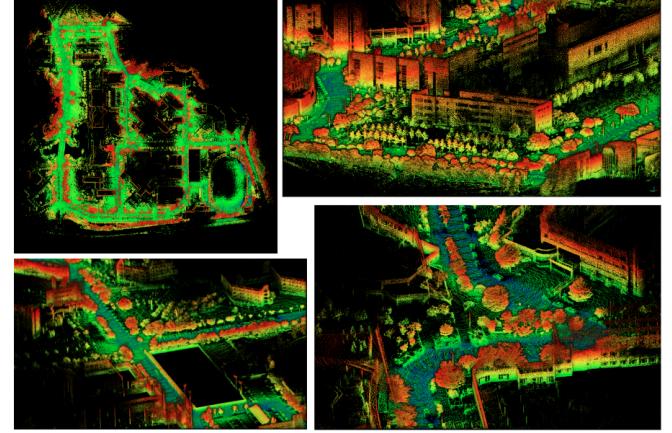


Fig. 4: Using the optimized poses, the SC-LiDAR-SLAM map making tool created a 3D point cloud map (KAIST sequence of MulRan dataset).

## IV. RESULT

### A. Experimental Setup

1) *Implementations*: SC-LiDAR-SLAM's main core is called SC-PGO. SC-PGO has been integrated with existing open-source LiDAR odometry methods, and the integration of the full LiDAR-SLAM systems are available via independent repositories for the ease of use: LeGO-LOAM [7]<sup>1</sup>, LIO-SAM [16]<sup>2</sup>, A-LOAM<sup>3</sup>, or FAST-LIO [18]<sup>4</sup>. The implementations are in C++ and heavily relied on GTSAM [43] for the solver. The place recognition codes are borrowed from our previous work<sup>5</sup>.

2) *Datasets*: Two datasets, KITTI dataset [44] and MulRan dataset [5]<sup>6</sup>, are selected to verify the effectiveness of SC-LiDAR-SLAM. KITTI dataset is a famous benchmark dataset, which is proper to compare existing methods, and MulRan dataset is particularly targeted to develop an algorithm that works for complex urban sites.

<sup>1</sup><https://github.com/irapkaiast/SC-LeGO-LOAM>

<sup>2</sup><https://github.com/gisbi-kim/SC-LIO-SAM>

<sup>3</sup><https://github.com/gisbi-kim/SC-A-LOAM>

<sup>4</sup>[https://github.com/gisbi-kim/FAST\\_LIO\\_SLAM](https://github.com/gisbi-kim/FAST_LIO_SLAM)

<sup>5</sup><https://github.com/irapkaiast/scancode>

<sup>6</sup><https://sites.google.com/view/mulran-pr>

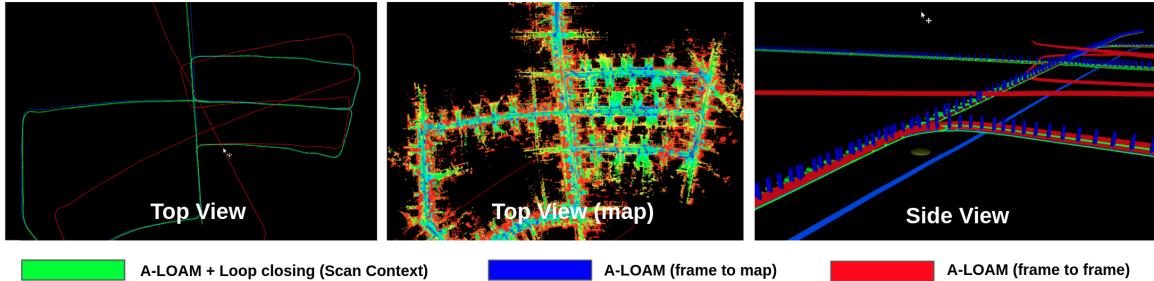
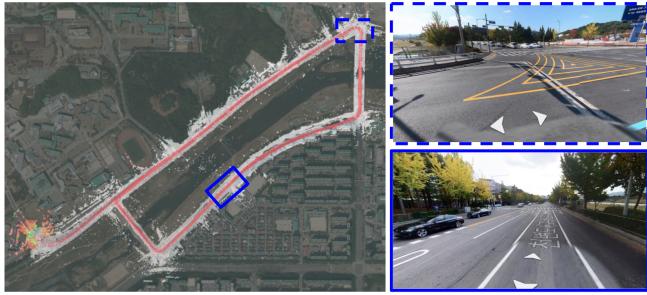
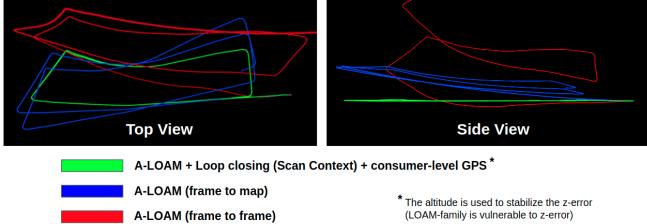


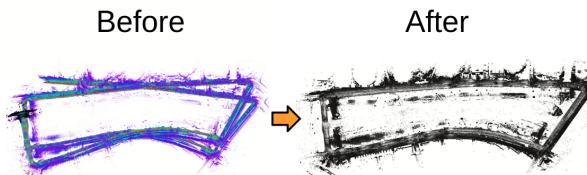
Fig. 5: Result visualizations of SC-LiDAR-SLAM integrated with A-LOAM. The odometry only method without being equipped with global place recognition (i.e., A-LOAM, the blue lines) inevitably contained accumulated drifts, which is particularly noticeable on the z-axis (see the right plot, a zoomed side-view), from errors of relative motion estimation at every step.



(a) Riverside sequence's wide urban roads



(b) Before-and-after trajectories of SC-LiDAR-SLAM (integrated with A-LOAM)



(c) Before-and-after point cloud maps of SC-LiDAR-SLAM (integrated with FAST-LIO)

Fig. 6: Tests of our robustness and versatility on a highly complex (i.e., many dynamic objects exist) urban road environment, Riverside sequence of MulRan dataset. (b) and (c) A-LOAM and FAST-LIO [18] are integrated, respectively.

### B. Pose-graph Optimization and Point cloud Maps

**KITTI dataset.** In Fig. 5, from the 05 sequence of KITTI dataset, we can see the optimized trajectory of SC-LiDAR-SLAM successfully eliminate the z-axis errors.

**MulRan dataset.** KAIST 03 sequence's results are pro-

vided in Fig. 1 and Fig. 4. The resulting point cloud map and the structures such as buildings in the map could clearly be recognized without noise from falsely registered submaps. This map correction by using the optimized loop-closed trajectory is more distinctly found in Riverside sequences in Fig. 6. This environment has wide roads with multiple fast-moving cars. Particularly, there are less structured features such as buildings. The dominant unstructured things side of the road such as trees would provoke more drifts<sup>7</sup> of odometry methods as in the left in Fig. 6(c). SC-LiDAR-SLAM successfully resolved the odometry errors and easily integrated with various odometry methods (i.e., with A-LOAM<sup>8</sup> in Fig. 6(b) and with FAST-LIO Fig. 6(c)).

### C. Applications

As mentioned in §III-C, SC-LiDAR-SLAM can be used for diverse useful robotic applications. In this subsection, we introduce some examples.

1) *Dynamic points removal:* In our other work, we proposed dynamic point removal [41]<sup>9</sup> only using SLAM results, not manual human intervention or labels. This framework *Removert*, takes the files saved from SC-LiDAR-SLAM and remove dynamic points such as moving cars in a map as in Fig. 7.

2) *Multi-session Map Merging:* In our other work [42], we can stitch multiple LiDAR maps acquired at different timestamps by using the results of SC-LiDAR SLAM. In Fig. 8, the two maps (trajectories are only visualized for clearness) covered the same region are shown. They were from different datasets, KITTI dataset [44] and KITTI-360 dataset [45] logged at different times, 2012 and 2013 respectively. As the left of Fig. 8, the two maps were originally not aligned because their starting pose is different but they considered their own starting pose is always the zero (identity). Using the Scan Context descriptors and the initially optimized trajectories by SC-LiAR-SLAM, LT-SLAM [42]<sup>10</sup> is capable to add inter-session loop closing factors and the two maps can

<sup>7</sup><https://youtu.be/94mC05PesvQ>

<sup>8</sup><https://github.com/HKUST-Aerial-Robotics/A-LOAM>

<sup>9</sup><https://github.com/irapkaist/removert>

<sup>10</sup><https://github.com/gisbi-kim/lm-mapper>

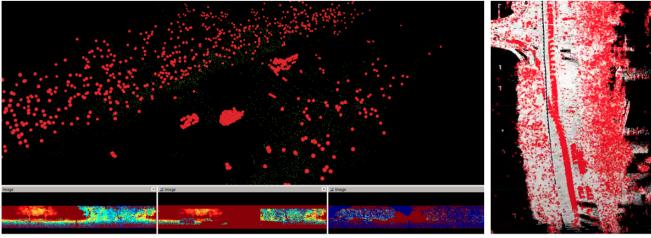


Fig. 7: A dynamic 3D point removal example on the Riverside sequence. The optimized poses and saved LiDAR scans from the results in Fig. 6(b) are used; those are the only ingredients for the removal without any manual inspection or training. The vivid red-colored dragged items came from a near fast-moving automobile and were well removed.

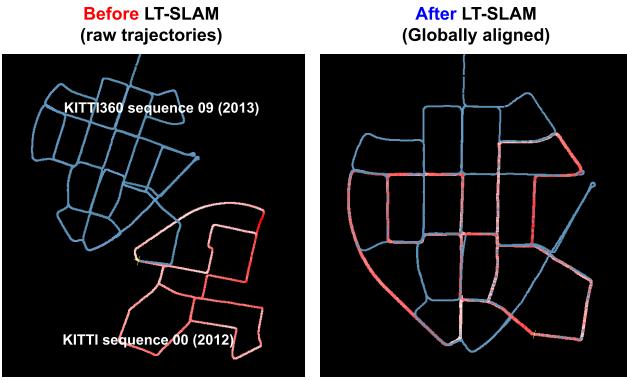


Fig. 8: A multi-session trajectory alignment result. A set of inputs (e.g., optimized pose-graphs and the saved place data such as scans and Scan Context Descriptors) to enable this multi-session SLAM are provided by SC-LiDAR-SLAM.

be successfully aligned in a shared coordinate. This is a key for long-term map management or spatial expanding of point cloud maps from multiple distributed fleet robots.

3) *Radar SLAM*: SC-LiDAR-SLAM's generic pose-graph optimization can be also applied to radar sensors. As in Fig. 9, we consider the radar data like 2D point cloud whose  $z$  values are fixed. Thanks to our front-end agnostic design, the radar SLAM, and accurate radar map can be easily made without extra effort<sup>11</sup>.

## V. CONCLUSION

In this paper, we proposed a front-end agnostic LiDAR SLAM system named SC-LiDAR-SLAM, and the diverse qualitative results were provided. Due to our modular design and the powerful loop closing capabilities of Scan Context++, we were able to easily integrate with various LiDAR odometry (or even radar odometry) methods and generate accurate point cloud maps. In future work, we would add a set of quantitative performance analyses of the proposed LiDAR SLAM system.

<sup>11</sup><https://github.com/gisbi-kim/navtech-radar-slam>

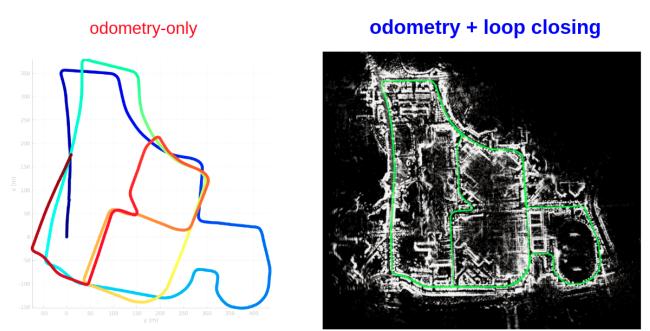


Fig. 9: Before-and-after of SC-LiDAR-SLAM for radar sensor. In this example, Yeti radar odometry [46] is integrated.

## REFERENCES

- [1] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 1–4.
- [2] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [3] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, “Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments,” *Journal of Field Robotics*, vol. 34, no. 5, pp. 940–984, 2017.
- [4] G. KIM, A. KIM, and Y. KIM, “A new 3d space syntax metric based on 3d isovist capture in urban space using remote sensing technology,” *Computers, Environment and Urban Systems*, vol. 74, pp. 74 – 87, 2019.
- [5] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, “MulRan: Multimodal Range Dataset for Urban Place Recognition,” in *Proc. IEEE Int'l. Conf. on Robot. and Automat.*, 2020.
- [6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age,” *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, 2016.
- [7] T. Shan and B. Englot, “LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2018, pp. 4758–4765.
- [8] J. Lin and F. Zhang, “Loam\_livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2019.
- [9] Y. Cho, G. Kim, and A. Kim, “Unsupervised geometry-aware deep lidar odometry,” in *Proc. IEEE Int'l. Conf. on Robot. and Automat.* IEEE, 2020, pp. 2145–2152.
- [10] L. He, X. Wang, and H. Zhang, “M2DP: a novel 3D point cloud descriptor and its application in loop closure detection,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2016, pp. 231–237.
- [11] G. Kim and A. Kim, “Scan Context: Egocentric spatial descriptor for place recognition within 3D point cloud map,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2018, pp. 4802–4809.
- [12] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss, “OverlapNet: Loop Closing for LiDAR-based SLAM,” in *Proc. Robot.: Science & Sys. Conf.*, 2019.
- [13] M. Kaess, A. Ranganathan, and F. Dellaert, “isam: Incremental smoothing and mapping,” *Int. J. Robot. Res.*, vol. 29, no. 11, pp. 1424–1444, 2010.

- tal smoothing and mapping,” *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.
- [14] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “iSAM2: Incremental smoothing and mapping using the Bayes tree,” *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
- [15] N. Sünderhauf and P. Protzel, “Switchable Constraints for Robust Pose Graph SLAM,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2012, pp. 1879–1884.
- [16] 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. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2020.
- [17] K. Li, M. Li, and U. D. Hanebeck, “Towards high-performance solid-state-lidar-inertial odometry and mapping,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5167–5174, 2021.
- [18] 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*, vol. 6, no. 2, pp. 3317–3324, 2021.
- [19] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, “Clins: Continuous-time trajectory estimation for lidar-inertial system,” 2021.
- [20] G. Kim, S. Choi, and A. Kim, “Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments,” *IEEE Transactions on Robotics*, 2021, accepted. To appear.
- [21] A. Segal, D. Haehnel, and S. Thrun, “Generalized-icp,” in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.
- [22] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, “Comparing icp variants on real-world data sets,” *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.
- [23] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time.” in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.
- [24] ———, “Low-drift and real-time lidar odometry and mapping,” *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017.
- [25] J. Behley and C. Stachniss, “Efficient surfel-based slam using 3d laser range data in urban environments.” in *Robotics: Science and Systems*, 2018.
- [26] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, “SegMatch: Segment based place recognition in 3D point clouds,” in *Proc. IEEE Intl. Conf. on Robot. and Automat.*, 2017, pp. 5266–5272.
- [27] M. A. Uy and G. H. Lee, “PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition,” in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2018, pp. 4470–4479.
- [28] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, “SegMap: Segment-based mapping and localization using data-driven descriptors,” *Intl. J. of Robot. Research*, vol. 39, no. 2–3, pp. 339–355, 2020.
- [29] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, “A tutorial on graph-based slam,” *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.
- [30] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller, “Multiple relative pose graphs for robust cooperative mapping,” in *Proc. IEEE Intl. Conf. on Robot. and Automat.*, 2010, pp. 3185–3192.
- [31] L. Carbone, R. Tron, K. Daniilidis, and F. Dellaert, “Initialization techniques for 3d slam: a survey on rotation estimation and its use in pose graph optimization,” in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 4597–4604.
- [32] E. Mendes, P. Koch, and S. Lacroix, “Icp-based pose-graph slam,” in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 195–200.
- [33] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, “Dynamic pose graph slam: Long-term mapping in low dynamic environments,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1871–1878.
- [34] F. Dellaert, M. Kaess *et al.*, “Factor graphs for robot perception,” *Foundations and Trends® in Robotics*, vol. 6, no. 1–2, pp. 1–139, 2017.
- [35] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard, “Robust Map Optimization using Dynamic Covariance Scaling,” in *Proc. IEEE Intl. Conf. on Robot. and Automat.*, 2013, pp. 62–69.
- [36] Z. Zhang, “Parameter estimation techniques: A tutorial with application to conic fitting,” *Image and vision Computing*, vol. 15, no. 1, pp. 59–76, 1997.
- [37] N. Chebrolu, T. Läbe, O. Vysotska, J. Behley, and C. Stachniss, “Adaptive robust kernels for non-linear least squares problems,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2240–2247, 2021.
- [38] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g 2 o: A general framework for graph optimization,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3607–3613.
- [39] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel, “Least squares optimization: from theory to practice,” *Robotics*, vol. 9, no. 3, p. 51, 2020.
- [40] M. Kaess, H. Johannsson, and J. Leonard, “Open source implementation of iSAM,” [people.csail.mit.edu/kaess/isam](http://people.csail.mit.edu/kaess/isam), 2010.
- [41] G. Kim and A. Kim, “Remove, then Revert: Static Point cloud Map Construction using Multiresolution Range Images ,” in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, Las Vegas, Oct. 2020.
- [42] ———, “Lt-mapper: A modular framework for lidar-based lifelong mapping,” *arXiv preprint arXiv:2107.07712*, 2021.
- [43] F. Dellaert, “Factor graphs and GTSAM: A hands-on introduction,” Georgia Institute of Technology, Tech. Rep., 2012.
- [44] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the KITTI vision benchmark suite,” in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2012, pp. 3354–3361.
- [45] Y. Liao, J. Xie, and A. Geiger, “KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d,” *arXiv.org*, vol. 2109.13410, 2021.
- [46] T. D. B. Keenan Burnett, Angela P. Schoellig, “Do we need to compensate for motion distortion and doppler effects in spinning radar navigation?” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 771–778, 2021.