SC-F-LOAM: A Fast LiDAR SLAM System with Loop Closure

Ezra Altabet, Chien-Lung Chou, Duong Le, Xinhao Sun, Yuhao Wu, Ray Zirui Xu
{ealtabet, lungchou, duongqle, xinhao, wuyuhao, ziruixu}@umich.edu

Abstract— **The light detection and ranging (LiDAR) odometry algorithms, such as Fast Light detection and ranging Odometry And Mapping (F-LOAM) [1], is a key technique in robotic simultaneous localization and mapping (SLAM). However, an estimation error will be accumulated after long-term operation of odometry, and place recognition and pose-graph optimization techniques are needed to calibrate it through loop closure. Also, while some of the state-of-the-art LiDAR SLAM systems, such as SC-A-LOAM [2], are equipped with loop closure and have satisfying accuracy, their efficiency is often compromised by the extra computation for loop closure. To achieve high accuracy as well as fast implementation, in this project, we propose SC-F-LOAM, a LiDAR SLAM system with loop closure for place recognition, by combining F-LOAM with SC-LiDAR-SLAM [2]. SC-LiDAR-SLAM is a modular LiDAR SLAM framework that can be easily implemented with any odometry algorithm. It integrates Scan Context [3] for place recognition and iSAM2 [4] for pose-graph optimization. We evaluate SC-F-LOAM on the KITTI dataset and compare with several existing popular methods, including A-LOAM, F-LOAM, and SC-A-LOAM. The results show that our method achieves a competitive accuracy as well as a real-time implementation. Therefore, SC-F-LOAM is a good balance of accuracy and runtime. Our project is open-source and available on Github: `https://github.com/zirui-xu/ROB-530-Final-Project`.**

I. Introduction

For robots to effectively work in an unknown environment, it is important for them to be able to map the environment and localize themselves in the same time. This is usually formulated as a Simultaneous Localization And Mapping (SLAM) problem [5]. A common LiDAR SLAM system often consists of three modules: LiDAR odometry, place recognition, and pose-graph optimization. As a robot moves in the environment and runs the SLAM system, the odometry algorithm leverages point cloud registration to estimate the robot's pose. A pose-graph that contains the robot's pose information is constructed during this period. However, although odometry itself can obtain a pose estimation and its accuracy has been improved over the past years, the pose estimation error is still inevitably accumulated in long-term operation. This error always causes inconsistency when the same place is revisited (i.e., when the loop is closed). To minimize such inconsistency, place recognition is introduced by adding constraints between such places to the pose-graph. Finally, the system performs pose-graph optimization with the previously added constraints to minimize the accumulated estimation error.

In early 2022, Kim *et al.* created SC-LiDAR SLAM [2], a framework that could integrate place recognition into state-of-the-art odometry models. The authors showed that for small increases in computational time, they could improve the average pose estimate error on A-LOAM, an Advanced version of LOAM. This fusion was known as SC-A-LOAM.

While SC-A-LOAM has shown great promise, its added computational complexity means that some hardware that could run A-LOAM may not be able to run SC-A-LOAM. F-LOAM (Fast LOAM) is a version of A-LOAM that has been optimized for computational efficiency.

Contribution. We have implemented SC-F-LOAM, a new LiDAR SLAM system that combines the computational efficiency of F-LOAM with the improved accuracy of SC-LiDAR-SLAM [2], a modular LiDAR SLAM system using Scan Context (for place recognition) and iSAM2 (for pose-graph optimization). Compared to existing popular LiDAR SLAM systems, SC-F-LOAM achieves competitive accuracy while also maintaining a real-time implementation, and is a more computationally feasible and robust option over SC-A-LOAM for lightweight SLAM systems.

Structure. In Section II we introduce several related methods. We present the proposed methodology in Section III. The results of experiments as well as a discussion of the results are presented in Section IV, and Section V provides the conclusion and future work.

II. Related Work

We introduce several related techniques of odometry, place recognition, and SLAM systems.

A. Odometry

LOAM. The fundamental algorithm of LiDAR Odometry and Mapping (LOAM) is first proposed in [6]. LOAM achieves real-time localization by dividing the SLAM problem into two parts: a high frequency low fidelity odometry estimation and a low frequency high fidelity LiDAR point cloud mapping. The data point cloud is processed to extract feature points: points on edges and points on surfaces. Feature points of the current scan is then transformed back to the previous scan and the distance between feature points and edges and surfaces is calculated. An iterative solver is used to calculate the transform that minimizes the distance between feature points of previous and current scans. Both high frequency and low frequency methods use these feature points. However, for the low frequency method the covariance matrix of multiple points is used to calculate the edge direction and surface norm, which provides a higher accuracy. An advanced implementation of LiDAR Odometry and Mapping (A-LOAM) [7] achieves faster implementation by replacing the customized solver in LOAM with Eigen and Ceres Solver to increase code modularity.

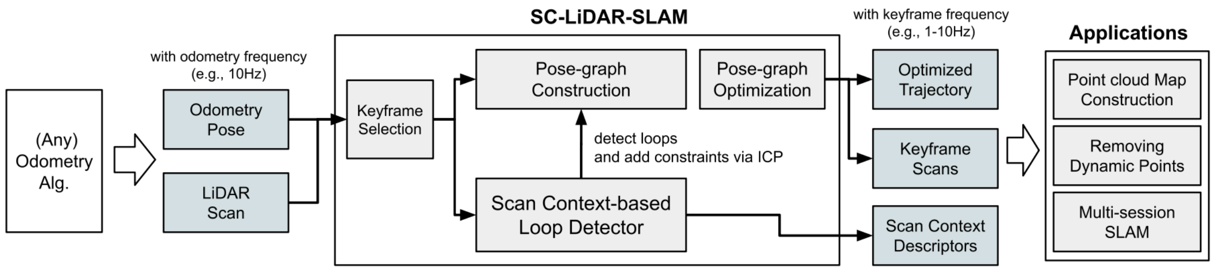


Fig. 1: **Overall pipeline of SC-LiDAR SLAM** (figure from [2]).

F-LOAM. F-LOAM is a version of A-LOAM optimized for computational efficiency. F-LOAM cuts down on computational costs throughout the process while minimizing sacrifices to accuracy. [1]. These improvements include:

- **Sensor Model and Feature Extraction:** In contrast to A-LOAM, rather than utilizing all surface and edge features, F-LOAM discards noisy features and less significant points.
- **Motion Estimation and Distortion Compensation:** F-LOAM takes advantage of the fact that most existing 3D LiDAR systems are able to run at more than 10 Hz, and therefore the time elapsed between two scans is very short. Thus F-LOAM assumes constant angular velocity and linear velocity between frames. In a second stage, distortion is re-computed after pose estimation. These undistorted features are then updated to the final map.
- **Pose Estimation:** It has been observed that edge features with higher local smoothness and planar features with lower smoothness are often consistently extracted over consecutive scans. This is important for matching. A weight function is thus used to further balance the matching process. To reduce computational cost, the local smoothness calculated previously is re-used to determine the weight function.
- **Map Building and Distortion Compensation Update:** The global map is updated using keyframes. Each keyframe is selected when the translational or rotational change is greater than a predefined threshold. The keyframe-based map update reduced computational cost compared to traditional frame-by-frame updates.

B. Place Recognition

Scan Context. Scan Context achieves global localization by constructing an egocentric spatial descriptor based on structural information for place recognition [3]. The structural information is obtained by 3D LiDAR scan. A 2.5D spatial descriptor that uses the LiDAR as the center is built, it maps the azimuthal direction to the x-axis and the radial direction to the y-axis. Then the azimuthal and radial directions are divided into many bins. The height of the highest points in each bin is chosen as the height of the bin. This height map of bins is used as the spatial descriptor. The distance between two spatial descriptor is defined as the minimal cosine distance sum of the radial vectors given all possible azimuthal correspondence. Global localization is achieved by comparing finding the spatial descriptor with minimal distance to the current descriptor in the global map.

Compared to previous place recognition methods, Scan Context does not rely on a histogram, which allows for a modular implementation.

C. SLAM Framework

SC-LiDAR-SLAM. Although the three modules of a SLAM system, i.e., odometry, place recognition, and pose-graph optimization, undertake different tasks as discussed in Section I, their implementation had always been coupled. That makes it hard for practitioners to test the system with a different odometry method, e.g., any change in algorithm would merit changes throughout the entire code structure. To address this issue, a front-end agnostic LiDAR SLAM framework, SC-LiDAR-SLAM, is proposed in [2]. This method leverages the modularity of Scan Context. Any odometry algorithm can be easily combined with SC-LiDAR-SLAM as the front end; the place recognition is conducted by Scan Context [3]; and the pose-graph optimization technique here is iSAM2 [4]. The pipeline of SC-LiDAR-SLAM is shown in Figure 1.

III. Methodology

We present the technical details of Scan Context, F-LOAM, and the proposed SC-F-LOAM.

A. Background

Scan Context [3]. Scan Context partitions whole points into P_{ij}, which is the set of points in the i-th ring and j-th sector,

$$P = \cup P_{ij}. \tag{1}$$

For each P_{ij} the height $z(p)$ of the highest element is defined as the height of the set ϕ,

$$\phi(P_{ij}) = \max_{p \in P_{ij}} z(p). \tag{2}$$

For each scan, a scan context I with N_s columns and N_r rows is defined as

$$I = (a_{ij}) \in R^{N_r \times N_s}, \quad a_{ij} = \phi(P_{ij}). \tag{3}$$

The distance d between two scan context images I^q, I^c is defined as the sum of cosine distance over N_s columns c_i,

$$d(I^q, I^c) = \frac{1}{N_s} \sum_{j=1}^{N_s} (1 - \frac{c_j^q c_j^c}{\|c_j^q\| \|c_j^c\|}). \tag{4}$$

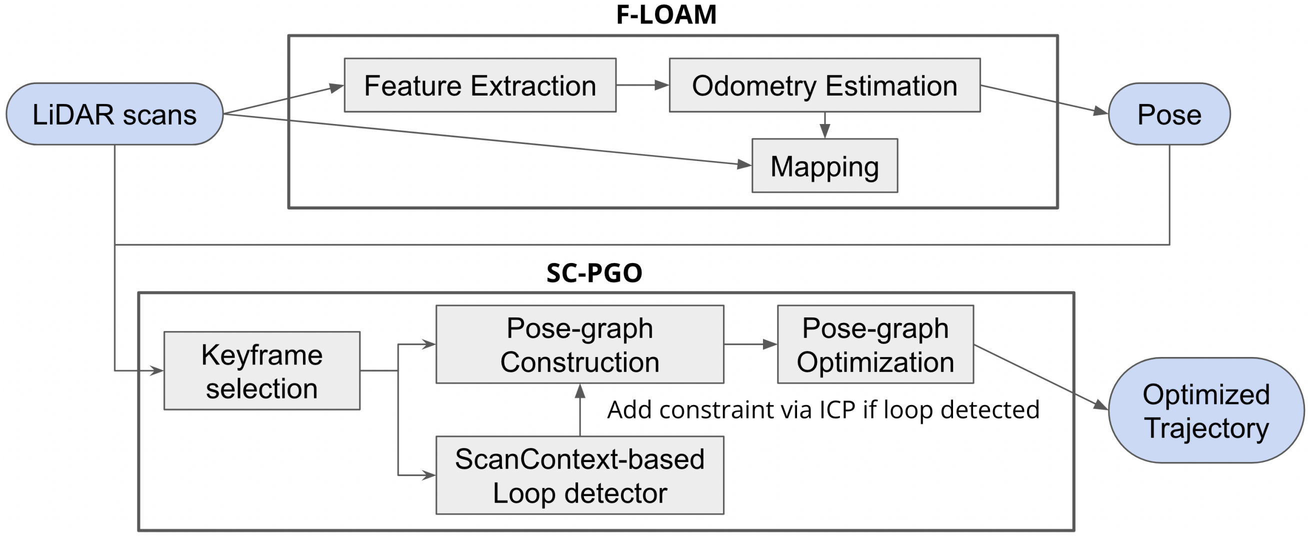


Fig. 2: **Overall pipeline of SC-F-LOAM**.

The distance D between two scan contexts is defined as the minimal distance of all possible shifts I_n^c,

$$D(I^q, I^c) = \min_{n \in [N_s]} d(I^q, I_n^c). \tag{5}$$

Global localization is achieved by comparing finding the spatial descriptor with minimal distance D to the current descriptor in the global map.

F-LOAM [1]. Based on the assumption of LiDAR points are sparse in vertical direction and dense in horizontal direction, the smoothness $c_k^{(m,n)}$ of local surface in frame k is calculated by only considering neighbors in the horizontal direction:

$$c_k^{(m,n)} = \frac{1}{|S_k^{(m,n)}|} \sum_{\mathbf{p}_k^{(m,j)} \in S_k^{(m,n)}} \|\mathbf{p}_k^{(m,j)} - \mathbf{p}_k^{(m,n)}\|, \tag{6}$$

where $S_k^{(m,n)}$ is the adjacent points of $\mathbf{p}_k^{(m,n)}$ and $|S_k^{(m,n)}|$ is the number of points in local point clouds. For a flat surface, the smoothness is small while for corner or edge, the smoothness is large. Hence, for the k-th scan, the feature points can be separated to edge features $\mathcal{E}_k$ and surface features $\mathcal{S}_k$.

Given the assumptions that linear velocity and angular velocity are constant during a short period, the distortion compensation ξ_{k-1}^k can be approximated as follows:

$$\xi_{k-1}^k \approx \xi_{k-2}^{k-1} = \log(T_{k-2}^{-1} T_{k-1}), \tag{7}$$

where $\xi \in \mathfrak{se}(3)$. The transformation $T_k(\delta t)$ of short period δt between consecutive scans can be estimated by linear interpolation:

$$T_k(\delta t) = T_{k-1} \exp(\frac{n}{N} \xi_{k-1}^k), \tag{8}$$

where N is the total movement and n is the movement for a short period δt. Hence, the features can be corrected by

$$\begin{aligned} \hat{\mathcal{E}}_k &= \{T_k(\delta t)\mathbf{p}^{(m,n)} | \mathbf{p}^{(m,n)} \in \mathcal{E}_k\}, \\ \hat{\mathcal{S}}_k &= \{T_k(\delta t)\mathbf{p}^{(m,n)} | \mathbf{p}^{(m,n)} \in \mathcal{S}_k\}, \end{aligned} \tag{9}$$

where $\hat{\mathcal{E}}_k$ is undistorted edge features and $\hat{\mathcal{S}}_k$ is undistorted surface features.

To find the optimized pose, F-LOAM minimized distance between feature points and global edges/planes which is derived by collecting nearby points from the global map. The distance $f_{\mathcal{E}}(\mathbf{p}_{\mathcal{E}})$ between the edges features and global edge is:

$$f_{\mathcal{E}}(\mathbf{p}_{\mathcal{E}}) = \mathbf{p}_n \cdot ((T_k \mathbf{p}_{\mathcal{E}} - \mathbf{p}_{\mathcal{E}}^g) \times \mathbf{n}_{\mathcal{E}}^g), \tag{10}$$

where $\mathbf{p}_n$ is an unit vector of $(T_k \mathbf{p}_{\mathcal{E}} - \mathbf{p}_{\mathcal{E}}^g) \times \mathbf{n}_{\mathcal{E}}^g$, $\mathbf{p}_{\mathcal{E}}^g$ is the center of edge, and $\mathbf{n}_{\mathcal{E}}^g$ is the eigenvector of the edge. The distance $f_{\mathcal{S}}(\mathbf{p}_{\mathcal{S}})$ between planar features and global plane is:

$$f_{\mathcal{S}}(\mathbf{p}_{\mathcal{S}}) = (T_k \mathbf{p}_{\mathcal{S}} - \mathbf{p}_{\mathcal{S}}^g) \cdot \mathbf{n}_{\mathcal{S}}^g, \tag{11}$$

where $\mathbf{p}_{\mathcal{S}}^g$ is the center of the surface and $\mathbf{n}_{\mathcal{S}}^g$ is the normal vector of the surface.

Since the smoothness is important for the feature points, the distance is weighted by the defined weight function as follows:

$$\begin{aligned} W(\mathbf{p}_{\mathcal{E}}) &= \frac{\exp(-c_{\mathcal{E}})}{\sum_{\mathbf{p}^{(i,j)} \in \hat{\mathcal{E}}_k} \exp(-c_{\mathcal{E}}^{(i,j)})}, \\ W(\mathbf{p}_{\mathcal{S}}) &= \frac{\exp(c_{\mathcal{S}})}{\sum_{\mathbf{p}^{(i,j)} \in \hat{\mathcal{S}}_k} \exp(c_{\mathcal{S}}^{(i,j)})}. \end{aligned} \tag{12}$$

By using a Gaussian-Newton solver, the transformation is estimated by minimizing weighted sum of the point-to-edge and point-to-plane distances:

$$\min_{T_k} \sum W(\mathbf{p}_{\mathcal{E}}) f_{\mathcal{E}}(\mathbf{p}_{\mathcal{E}}) + \sum W(\mathbf{p}_{\mathcal{S}}) f_{\mathcal{S}}(\mathbf{p}_{\mathcal{S}}). \tag{13}$$

B. Our Algorithm: SC-F-LOAM

We present SC-F-LOAM, a computationally efficient LiDAR SLAM system. We integrated F-LOAM [1] as the odometry model into the SC-LiDAR SLAM framework, which uses Scan Context [3] for place recognition and iSAM2 [4] for pose-graph optimization. SC-F-LOAM is identical to SC-A-LOAM except that F-LOAM is used as the odometry model instead of A-LOAM. The pipeline is shown in Figure 2.

IV. Experiment

We present the experiment setup, evaluate our algorithm, and discuss the results of the experiments. A link to a presentation of our work on Youtube: `https://youtu.be/rhNIqsH6vB4`.

A. Experiment Setup

We evaluated SC-F-LOAM on the KITTI dataset [8]. The KITTI dataset consists of hours of sensor data obtained by a driving car equipped with Velodyne HDL-64 LiDAR and, as a result, is one of the most popular SLAM evaluation datasets. Most state-of-the-art SLAM methods have been evaluated on this dataset, e.g., ORB-SLAM [9] and V-LOAM [10]. We sampled three sequences (i.e., 00, 02, and 05) in which the car went to the revisited place to demonstrate the loop closure techniques. Furthermore, we tested SC-F-LOAM against other algorithms such as A-LOAM [6], F-LOAM [1], and SC-A-LOAM [2] to demonstrate the improvement of the proposed algorithm. All tests were conducted on the same computer with the same input data to minimize the testing error. These algorithms were tested using Robot Operating System (ROS) Melodic in Ubuntu 18.04, which was installed on a virtual machine (Virtualbox). The virtual machine is configured with seven CPUs and 80 GB RAM on a host machine with an 11th Gen Intel Core i9 with 3.5GHz CPU.

B. Evaluation

We first compared F-LOAM and SC-F-LOAM visually to see how close their estimates are to the ground truth. The trajectory generated by F-LOAM and SC-F-LOAM for each map are shown in Figure 3, with the ground truth marked via a dashed line. Visually, it can be observed from the figure that SC-F-LOAM is closer to the ground truth pose than F-LOAM for all three maps.

We then quantitatively compared the results of all four evaluated SLAM methods using three evaluation metrics: Average Translational Error (ATE) and Average Rotational Error (ARE) and computational time. The results are visually represented in Figure 4. We used the tool evo [11] to calculate all ATE and ARE and recorded them in Table I. Here the ATE is calculated first by,

$$F_i = Q_i^{-1} P_i, \tag{14}$$

where F_i is the deviation for frame i, P_i is the estimated pose at frame i, and Q_i is the pose of the ground truth for the given frame. After calculating F_i, we calculated ATE from the RMSE of translation by,

$$ATE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|trans(F_i)\|^2}, \tag{15}$$

Method	Metric	KITTI 00	KITTI 02	KITTI 05
A-LOAM	ATE (m)	2.486814	107.2454	2.033932
	ARE ($\times$0.01)	0.036044	0.490893	0.024559
F-LOAM	ATE (m)	4.742383	7.35459	2.940284
	ARE ($\times$0.01)	0.043563	0.055032	0.03155
SC-A-LOAM	ATE (m)	1.158887	103.3689	0.712823
	ARE ($\times$0.01)	0.028809	0.476638	0.016656
SC-F-LOAM	ATE (m)	1.465621	3.483975	1.32898
	ARE ($\times$0.01)	0.02746	0.047693	0.018149

TABLE I: **ATE and ARE of all four methods** on KITTI 00, KITTI 02, and KITTI 05. The lower, the better. The red number is the best result for the corresponding KITTI sequence.

where $trans(F_i)$ is the deviation of the translation matrix, n is the total number of frames, and the unit is meter (m) since it is the absolute error.

ARE was also calculated using RMSE between δR_i and identity matrix I,

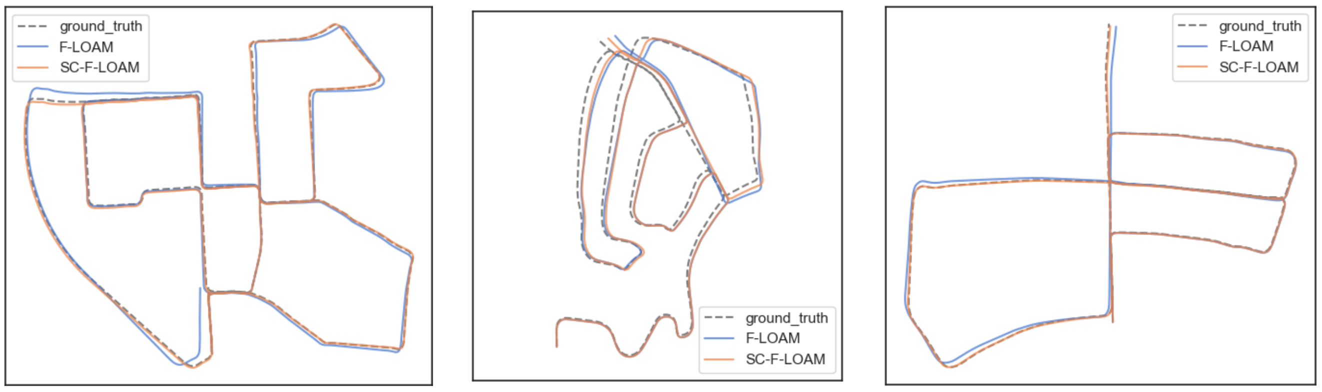
$$ARE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|\delta R_i - I\|^2}, \tag{16}$$

where δR_i is the rotation matrix from the estimated pose to ground truth at time frame i. Thus the result of ARE is a unitless absolute error.

To evaluate the difference in computational time between each method, since we mainly focused on pose estimation, we calculated only the time of odometry estimation and pose-graph optimization. We created a timer to record the computational time of pose estimation, Scan Context descriptor generation, loop closure detection, and pose-graph optimization for SC-A-LOAM and SC-F-LOAM. To obtain A-LOAM and F-LOAM results, we extracted the pose estimates and computation time for A-LOAM and F-LOAM from SC-A-LOAM and SC-F-LOAM odometery modules, respectively. The total execution time of each algorithm is divided by the number of frames in each sequence to obtain the result in Figure 4.

From Figure 4 we can see that overall SC-F-LOAM has much lower error than SC-A-LOAM, but from Table I we can see SC-A-LOAM has marginally lower error than SC-F-LOAM on KITTI 00 and KITTI 05, but has significant error on KITTI 02. This resulted in a much higher average absolute error for SC-A-LOAM. SC-F-LOAM, while having its own increase in error in KITTI 02, showed its robustness by having a significantly smaller increase in error than SC-A-LOAM. Considering that KITTI 02 has the longest map, one theory is that that A-LOAM will have higher accumulated error in a long trajectory, which is the same for SC-A-LOAM.

We can also see that SC-F-LOAM has lower ATE and ARE than F-LOAM, but higher computational time, which reflects the change between SC-A-LOAM and A-LOAM. The addition of Scan Context to F-LOAM clearly shows an increase in average accuracy over F-LOAM, but longer processing time.



KITTI 00 KITTI 02 KITTI 05

Fig. 3: **Trajectory Estimation of F-LOAM vs. SC-F-LOAM** against ground truth. The red line is SC-F-LOAM estimate, the blue line is from F-LOAM, and the dashed line is the ground truth.

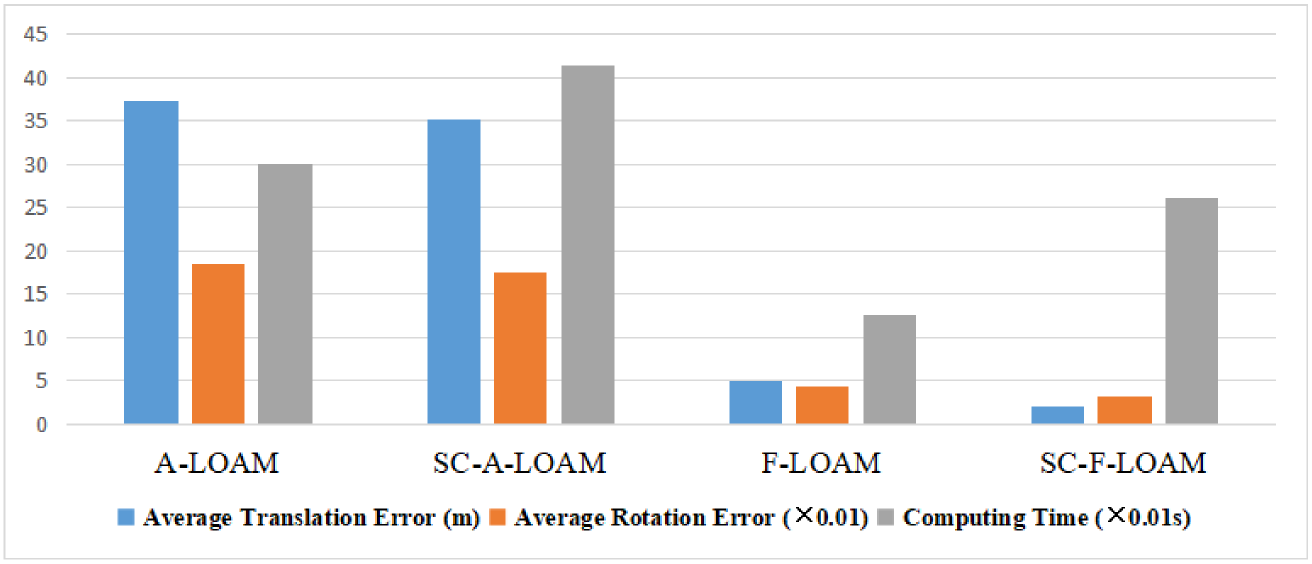


Fig. 4: **Comparison on Accuracy and Time** of A-LOAM, SC-A-LOAM, F-LOAM, and SC-F-LOAM.

C. *Discussion*

A-LOAM has real-time performance in many cases, but our results suggest it would fail due to the accumulated errors over long trajectories. SC-A-LOAM slightly reduces this error but at the cost of added computational time. While this may not be a burden to devices with significant computation power such as autonomous cars, it may prove burdensome on more lightweight robotic devices for real-time SLAM applications. Our proposed method, SC-F-LOAM, demonstrates greater robustness in the aspects of computational time and localization accuracy when compared against A-LOAM and SC-A-LOAM. SC-F-LOAM is only marginally less accurate than SC-A-LOAM in short trajectories and maintains its robustness on long trajectories, demonstrating significantly less error than both A-LOAM and SC-A-LOAM. In all cases, SC-F-LOAM is significantly more accurate than F-LOAM alone. SC-F-LOAM benefits from utilization of the computational efficiency of F-LOAM, primarily via the constant velocity assumption and its reliance on keyframes for map updates, with the added accuracy of place recognition and pose graph optimization supplied by the SC-LiDAR SLAM framework. Hence, SC-F-LOAM can obtain the optimized trajectory more efficiently than A-LOAM and SC-A-LOAM.

V. Conclusion

Summary. In this paper, we proposed and implemented SC-F-LOAM, a computationally efficient LiDAR SLAM system with loop closure. SC-F-LOAM combines F-LOAM for odometry, Scan Context for place recognition, and iSAM2 for pose-graph optimization. We evaluated the efficacy of the system on the KITTI dataset. The results show that, compared to existing popular LiDAR SLAM systems, SC-F-LOAM provides a competitive accuracy as well as maintains a real-time implementation, thus achieving a satisfying trade-off of high accuracy and fast implementation.

Future Work. Per the current experiment results, SC-F-LOAM has a competitive accuracy compared to SC-A-LOAM. However, this observation is obtained from the results of only three KITTI sequences. Therefore, future work will include more experiments with all KITTI sequences to justify whether Scan Context can compensate for the gap of accuracy between A-LOAM and F-LOAM. Also, current results show that SC-A-LOAM produced a significantly large ATE on KITTI 02 compared to KITTI 00 and 05. Further research is needed to investigate the reason for this in order to evaluate the robustness of SC-F-LOAM against SC-A-LOAM.

References

[1] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast lidar odometry and mapping," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4390–4396.

[2] G. Kim, S. Yun, J. Kim, and A. Kim, "SC-LiDAR-SLAM: A front-end agnostic versatile LiDAR SLAM system," *arXiv preprint: 2201.06423*, 2022.

[3] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4802–4809.

[4] 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.

[5] 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 Transactions on robotics (TRO)*, vol. 32, no. 6, pp. 1309–1332, 2016.

[6] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[7] T. Qin and S. Cao, "Advanced implementation of loam," 2019. [Online]. Available: https://github.com/HKUST-Aerial-Robotics/A-LOAM

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[10] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2174–2181.

[11] M. Grupp, "evo: Python package for the evaluation of odometry and slam." 2017. [Online]. Available: https://github.com/MichaelGrupp/evo