DL-SLOT: Tightly-Coupled Dynamic LiDAR SLAM and 3D Object Tracking Based on Collaborative Graph Optimization

Xuebo Tian, Zhongyang Zhu, Junqiao Zhao, *Member, IEEE*, Gengxuan Tian, and Chen Ye, *Member, IEEE*

***Abstract*—Ego-pose estimation and 3D object tracking are two critical problems that autonomous driving systems must solve. The solutions to these problems are usually based on their respective assumptions, i.e., the static world assumption for simultaneous localization and mapping (SLAM) and the accurate ego-pose assumption in object tracking. However, these assumptions may not hold in dynamic road scenarios, where SLAM and object tracking are closely correlated. Therefore, we propose DL-SLOT, a tightly-coupled dynamic LiDAR SLAM and 3D object tracking method that addresses both problems simultaneously. This method integrates the state estimations of both the autonomous vehicle and the stationary and dynamic objects in the environment in a unified optimization framework. First, we use object detection to identify all points belonging to potentially dynamic objects. Subsequently, LiDAR odometry is performed using the filtered point cloud. Simultaneously, we propose a sliding window-based 3D multi-object tracking method considering the historical trajectories of the tracked objects. The states of the ego-vehicle, stationary objects, and dynamic objects are jointly estimated by the sliding window-based collaborative graph optimization to achieve SLOT. True stationary objects are restored from the potentially dynamic object set to join the optimization. Finally, a global pose graph is implemented to eliminate the accumulated error. Experiments on the KITTI datasets demonstrate that our method achieves better accuracy than SLAM and 3D object tracking baseline methods. This confirms that solving SLAM and object tracking simultaneously is mutually beneficial, and dramatically improves the robustness and accuracy of SLAM and object tracking in dynamic road scenarios.**

***Index Terms*—LiDAR SLAM, multi-object tracking, graph optimization.**

Manuscript received 10 July 2023; accepted 5 September 2023. Date of publication 19 September 2023; date of current version 23 February 2024. This work was supported by the National Key Research and Development Program of China under Grant 2021YFB2501104. *(Xuebo Tian and Zhongyang Zhu contributed equally to this work.) (Corresponding author: Junqiao Zhao.)*

Xuebo Tian, Zhongyang Zhu, Gengxuan Tian, and Chen Ye are with the Department of Computer Science and Technology, School of Electronics and Information Engineering, Tongji University, Shanghai 200070, China, and also with the The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 200070, China (e-mail: 1930773@tongji.edu.cn; 2233057@tongji.edu.cn; 2130791@tongji.edu.cn; yechen@tongji.edu.cn).

Junqiao Zhao is with the Department of Computer Science and Technology, School of Electronics and Information Engineering, Tongji University, Shanghai 200070, China, also with the The Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 200070, China, and also with the Institute of Intelligent Vehicles, Tongji University, Shanghai 200070, China (e-mail: zhaojunqiao@tongji.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIV.2023.3317308.

Digital Object Identifier 10.1109/TIV.2023.3317308

I. Introduction

In RECENT years, LiDAR simultaneous localization and mapping (SLAM) methods have been intensively studied as one of the cornerstones of autonomous driving. Although several advanced LiDAR SLAM methods have been proposed and shown high accuracy [1], [2], they are built on a static world assumption and tend to fail in dynamic road environments.

Conventional methods either filter out dynamic laser points to eliminate the impact of moving objects on LiDAR SLAM [3], [4], [5], [6], [7], or use traditional multi-object tracking methods to track dynamic objects [8], [9]. Although such methods guarantee that the static world assumption holds, the loss of dynamic information can degrade the localization accuracy and cause SLAM to fail in a road environment that constrains many moving objects.

The latest studies on dynamic visual SLAM combine visual SLAM and multi-object tracking [10], [11], [12], [13], [14]. These methods implement a bundle adjustment (BA) with camera poses, object poses, and visual features. However, object detection and tracking are mostly performed in 2D image space, which are less accurate than LiDAR-based methods due to field-of-view (FOV) limitations, depth ambiguity, and greater susceptibility to occlusion.

Inspired by these methods, we propose DL-SLOT, which realizes tightly-coupled LiDAR SLAM and 3D object tracking (SLOT) for dynamic road scenes. The state estimations of the ego-vehicle and the stationary and dynamic objects in the scene are incorporated into a collaborative graph optimization framework.

First, an object detector is employed to identify potentially dynamic objects (PDOs)[1] in each LiDAR frame. Subsequently, a preliminary LiDAR odometry is implemented using the point clouds that have been filtered out of the points belonging to the PDOs. Moreover, we propose using a sliding window with a fixed time interval to perform 3D object association using the historical trajectories of the tracked objects, which enhances the accuracy of the association. Finally, assuming that the dynamic objects move at their respective constant velocities over a short period of time, we propose integrating the constant velocity constraint as well as the states of the stationary and dynamic objects and the ego-vehicle into a sliding window-based

[1]PDOs are determined by their semantics rather than their actual motion state.

Fig. 1. Mapping result for sequence 05 on the KITTI Odometry dataset. The gray shading indicates the height information of the environment. The green dots and red dots indicate the trajectories of the ego-vehicle and those of dynamic objects, respectively.

collaborative graph optimization framework. Therefore, our system can simultaneously estimate the map and the trajectories of the ego-vehicle and the moving objects, as shown in Fig. 1.

The main contributions of this article are 3-fold:

- To the best of the authors' knowledge, this is the first LiDAR SLOT system that has been proposed and intensively tested.
- A sliding window-based 3D multi-object tracking method using historical trajectory information.
- A tightly-coupled optimization framework, simultaneously estimating the map, the states of the ego-vehicle, and the PDOs.

II. Related Works

A. LiDAR SLAM in Dynamic Scenes

Current LiDAR SLAM methods for dynamic environments are mainly implemented through the rejection of dynamic points. These methods can be roughly classified into two categories based on the method of extracting the dynamic points.

The first category directly partitions the point cloud into dynamic and stationary parts. Li et al. [3] applied a neural network to predict a dynamic mask using two consecutive scans. The mask was then used to partition the dynamic and stationary regions, which allowed odometry using only the stationary point clouds. Qian et al. [4] used adaptive multi-resolution range images to detect moving points in a LiDAR scan. After the points were removed from the scan, the scan was matched to the submap to estimate odometry. However, segmenting the dynamic points without considering the semantics and behaviors of dynamic objects is non-trivial and inaccurate.

The second category utilizes object detection or semantic segmentation approaches to obtain the semantic label for each point, and simply removes all points belonging to PDOs. Vaquero et al. [5] adopted a two-stream convolutional neural network (CNN) to segment PDOs from 3D LiDAR point clouds. After removing the points associated with the PDOs, the point clouds were used to implement odometry. Zhao et al. [6] also used a CNN to detect and remove PDOs from each LiDAR scan, and then performed the LiDAR-inertial odometry using an error state Kalman filter. However, points belonging to stationary PDOs, such as parked cars, were also removed, which reduces the information available for SLAM and thus reduces the accuracy of LiDAR SLAM.

B. Coupled LiDAR SLAM and Multi-Object Tracking

The method proposed by Moosmann et al. [8] was one of the first studies to jointly perform LiDAR SLAM and multi-object tracking. This method modeled both problems as a unified tracking-without-detection problem. The aim was to track all the dynamic segments as well as the static scene, so that the ego-motion could be estimated from the tracking result of the static scene. However, the dynamic segments and the static scene were gradually identified and separated during the tracking process. Therefore, because the segmentation and tracking were initially inaccurate, the registration between the tracked segments and the full input point cloud inevitably introduces noise into the SLAM.

Ma et al. [9] fused geometric feature extraction and semantic object detection to segment PDOs and static background features. While estimating ego-motion based on the static background features, multi-object tracking was achieved using the least-squares method by fusing 3D detection and point clouds. In this method, SLAM and multi-object tracking were performed independently with the purpose of obtaining a better understanding of the dynamic scene. However, the accuracy of object tracking depends on accurate ego-pose estimation, and SLAM is likely to fail when it operates independently in complex dynamic environments.

C. Tightly-Coupled Visual SLAM and Multi-Object Tracking

To address the above issues, the more recent visual SLAM methods have proposed tightly-coupled SLAM and multi-object tracking, i.e., jointly optimizing the poses of both the ego-vehicle and the surrounding objects in a unified framework.

Henein et al. [15] first proposed integrating information about the motion of dynamic objects into a SLAM system. However, this method did not track the objects and estimate their poses. Yang et al. [10] represented the objects using 3D bounding boxes (BBoxes) estimated from the detected 2D BBoxes. For dynamic objects, they used a 2D visual object tracking algorithm directly [16]. The ego-poses and the states of the objects were then optimized with stationary features in a BA. However, the proposed 3D object recovery and tracking methods using a single monocular camera are not robust. Zhang et al. [11] did not recover 3D BBoxes of objects, but instead used instance-level semantic segmentation to segment and identify PDOs. Object association was achieved using optical flow based on visual features. Subsequently, a BA was implemented to optimize the camera poses, static features, and object motions according to a novel formulation of dynamic motion models. Bescos et al. [12] also used instance semantic segmentation to detect objects. As was done in [10], the intersection over union (IoU) of 2D

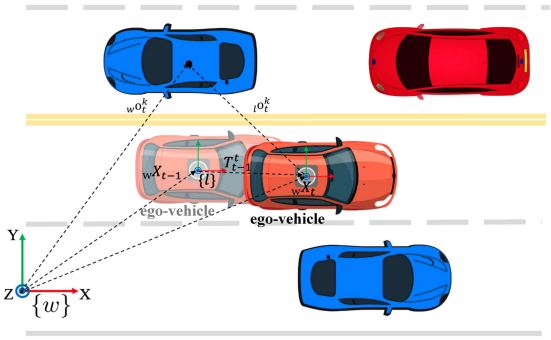


Fig. 2. Pose notations for ego-vehicle and objects. $\{w\}$ is the world coordinate system and $\{l\}$ is the ego-vehicle coordinate system. ${}_wX_t$ is the autonomous vehicle's pose, and T_{t-1}^t is the odometry between the adjacent frames. ${}_wo_t^k$ and ${}_lo_t^k$ are the object's poses in different coordinate systems.

BBoxes was used as the cost function for object association. Then a higher level association was performed based on the ORB feature [17] correspondences. Gonzalez et al. proposed TwistSlAM [13], which first performed the panoptic segmentation of images. Then 3D semantic point clusters were created by fusing multiple 2D observations. However, dynamic data association was still achieved using optical flow estimation in 2D image space. In addition, this method performed a novel constraint BA to jointly estimate the poses of both the moving objects and the camera by defining inter-cluster constraints modeled by mechanical joints to improve the accuracy of the object poses. The same authors recently developed TwistSlAM++ [14], which is an extension of TwistSlAM that improves the accuracy of object poses by injecting LiDAR data. However, the need for multiple sensors reduces this method's versatility.

In summary, existing tightly-coupled visual SLAM and multi-object tracking methods mostly perform object detection and tracking in 2D image space and thus suffer from inaccurate 3D object representation and feature degraded environments, i.e., environments with loss of texture and low illumination. In addition, the pose accuracy of both the ego-vehicle and objects in vision is generally not as good as that of LiDAR-based methods due to the limited FOV and depth ambiguity.

III. Methods

A. Problem Formulation

1) Notation Definition: As shown in Fig. 2, in the world coordinate system $\{w\}$, the pose of the autonomous vehicle at the frame t is denoted as ${}_wX_t \in SE(3)$, and its pose transformation from the previous frame $t-1$ to the current frame t is denoted as $T_{t-1}^t \in SE(3)$. In the local coordinate system $\{l\}$, we represent the k-th object's pose detected at frame t as ${}_lo_t^k \in SE(3)$ and as ${}_wo_t^k \in SE(3)$ in the world coordinate system respectively. ${}_lo_t^k$ can be converted to ${}_wo_t^k$ and vice versa by (1).

$$ {}_wo_t^k = {}_wX_t \cdot {}_lo_t^k \tag{1}$$

The trajectory of the j-th tracked object at frame t is defined as $Tr_t^j = ({}_wo_{t-n+1}^j, {}_wo_{t-n+2}^j, \ldots, {}_wo_t^j)$ in the $\{w\}$ frame, where n is the number of trajectory points. Then the set of tracked objects and their trajectories at frame t can be defined as $O_t = \{Tr_t^j | j \in [1, M]\}$, where M is the number of the tracked objects in the t-th frame. The pose transformation of the j-th object from frame $t-1$ to frame t is defined by the matrix ${}^jc_{t-1}^t \in SE(3)$.

2) Problem Definition: With these established notations, the SLOT problem is defined as follows: Given the pose transformation sequence $T_0^1, T_1^2, \ldots, T_{t-1}^t$ and a sequence of tracked object sets $O_0, O_1, \ldots, O_t$, the system simultaneously estimates the autonomous vehicle's poses and the state of each object.

Specifically, let $\boldsymbol{X}_{ego} = \{{}_wX_i | i \in [0,t]\}$, $\boldsymbol{X}_O = \{{}_wo_j^i | j \in O_i, i \in [0,t]\}$ and let $\boldsymbol{X}_C = \{{}^jc_{i-1}^i | j \in O_i, i \in [1,t]\}$. Note that $j \in O_i$ here represents that j is the index of the tracked object in O_i. Given $\boldsymbol{X} = \boldsymbol{X}_{ego} \cup \boldsymbol{X}_O \cup \boldsymbol{X}_C$, the error terms e_{odo} and e_{obj} are constructed for the ego-vehicle and object poses respectively, and a remaining error term e_{motion} is constructed using the motion model. This problem can then be converted to a least-squares optimization problem as follows:

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{argmin} \Bigg\{ \sum_{i\in[0,t]} \Bigg(\left\| e_{odo}^i \right\|_{\Sigma_{odo}}^2 + \sum_{j\in O_i} \Bigg(\left\| e_{obj}^j \right\|_{\Sigma_{obj}}^2 + \left\| e_{motion}^j \right\|_{\Sigma_{motion}}^2 \Bigg) \Bigg) \Bigg\} \tag{2}$$

B. System Overview

The architecture of the proposed DL-SLOT system is shown in Fig. 3. The system consists of three components: the SLOT front-end, the sliding window-based 3D multi-object tracking, and the SLOT back-end.

The SLOT front-end detects PDOs using a deep convolutional network (DCN)-based detector [18] and filters out LiDAR points in the detected 3D BBoxes. Subsequently, it calculates the frame-to-frame transformations based on the LiDAR odometry (Section III-C) and detects loop closures.

3D object association is performed between the tracked and detected objects using their trajectories within a sliding window, which will be discussed in Section III-D. The SLOT back-end, which includes sliding window-based collaborative optimization and global optimization, jointly estimates the ego-poses and the states of the objects (Section III-E).

C. LiDAR Odometry

The proposed DL-SLOT is based on the popular LiDAR SLAM method LeGO-LOAM [2] but is not limited to any particular LiDAR SLAM method. The LiDAR odometry consists of three main modules: feature extraction, registration, and mapping. Edge feature points and planar feature points are detected based on the local roughness of the point cloud. Next, point-to-edge and point-to-plane scan-matching is conducted between two consecutive scans. Simultaneously, the mapping module registers and merges scans to the global point cloud map. During this process, the poses (i.e., ${}_wX_t$) are further refined. The transformation T_{t-1}^t between two poses can be calculated using (3).

$$T_{t-1}^t = {}_wX_{t-1}{}^{-1} \cdot {}_wX_t \tag{3}$$

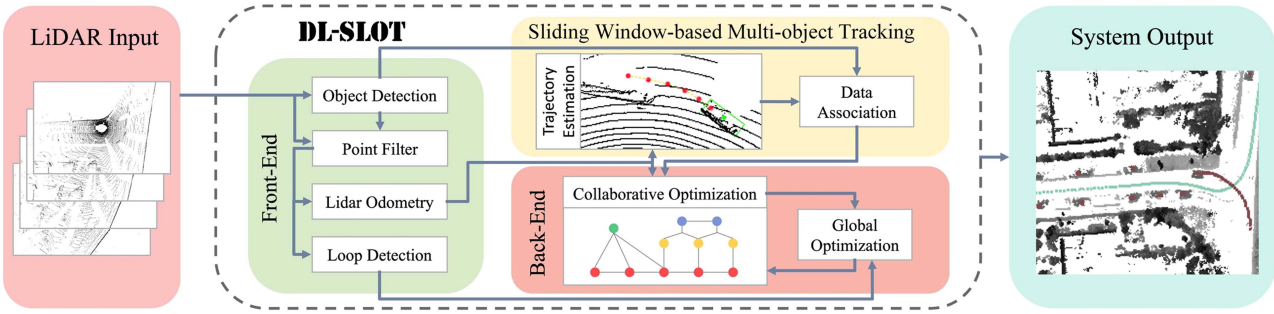


Fig. 3. System architecture of DL-SLOT. The system consists of the SLOT front-end, the sliding window-based 3D multi-object tracking, and the SLOT back-end.

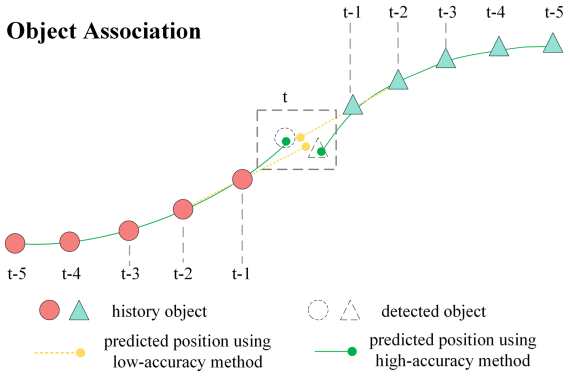


Fig. 4. Schematic for object association. The colored circle and triangle denote two different objects. The solid green line represents the object's trajectory. The dashed circle and triangle represent the detected objects, and the yellow dots and green dots represent the predicted results using the low-accuracy method and the high-accuracy method, respectively.

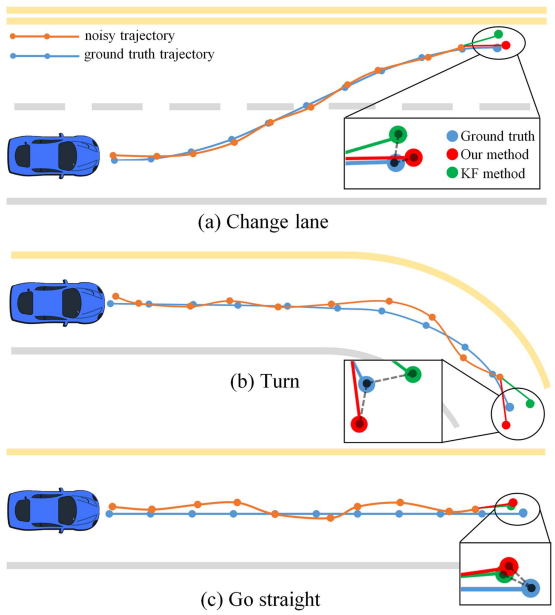


Fig. 5. Results for the proof-of-concept experiment. Three driving situations are shown: changing lanes, turning, and going straight. The blue dots and orange dots show the ground truth trajectory and the noisy trajectory points used for state prediction, respectively. The red dots and green dots are the positions predicted by our method and by the constant velocity-based KF method, respectively.

D. Sliding Window-Based 3D Multi-Object Tracking

3D multi-object tracking involves two key functions: state prediction and object association.

1) State Prediction: Most of the recent real-time 3D multi-object tracking methods, e.g., AB3DMOT [19] and PC3T [20], adopt the Kalman filter (KF) as the state estimator, which is efficient and accurate for well-established motion models. However, the probabilistic filter uses only the previous state to predict the next state based on the motion model. Therefore, it does not make full use of historical states and is limited by the selected motion model. Learning-based multi-object tracking methods [21], [22], [23], [24] can predict complex future states from historical trajectory data through neural networks. However, they are usually several times slower than filter-based methods and are mostly difficult to execute in real-time.

Accurate state prediction is essential for correct object association. As demonstrated in Fig. 4, two objects are moving toward each other. The low-accuracy predicted results are vulnerable to incorrect associations, as illustrated by the yellow dots. The cubic order polynomial curve ((4)) is frequently used in motion planning to optimize or smooth the trajectory of a vehicle [25], [26]. This study proposes a trajectory-based state prediction method using successive trajectory points within a sliding window. The motion intention of an object can be approximated by fitting the trajectory with a cubic order polynomial using the least-squares method.

$$P(t) = \theta_1 \cdot t^3 + \theta_2 \cdot t^2 + \theta_3 \cdot t + \theta_4 \tag{4}$$

where $P(t)$ represents the position at frame t; $\theta_1, \theta_2, \theta_3, \theta_4$ represent the parameters that need to be estimated. We approximate the object's trajectory along the x-axis and y-axis, respectively, to obtain the predicted position of the object at frame t.

To demonstrate that the trajectory-based state prediction method can predict a vehicle's future state more accurately than the constant velocity-based KF model, we conducted a simple proof-of-concept experiment. As shown in Fig. 5, we compared the performance of two methods in predicting the final state using simulated trajectories in three typical driving scenarios, which are changing lanes, turning, and traveling in a straight line. The simulated trajectories are composed of 11 steps at intervals of 0.1 s. Gaussian noise $N(0, 0.2)$ is applied to the first 10 steps of the trajectories, and the average prediction error of the 11th state in 9 repetitions of the experiment is given in Table I. Both

TABLE I
ERROR(M) RESULTS OF DIFFERENT STATE PREDICTION METHODS IN THE SIMULATION EXPERIMENT

Driving scenarios	Velocity-based KF	Ours
Change lane	0.622	0.349
Turn	0.539	0.532
Go straight	0.361	0.347

methods have a similar prediction accuracy when the vehicle is traveling in a straight line or making a turn. However, in a complex lane-changing scenario, trajectory approximation with the use of historical information obtains better results, as shown by the zoomed-in plot in Fig. 5. More importantly, in our proposed SLOT system, the historical trajectories of objects in the sliding window are continuously optimized (see Section III-E), which makes our state prediction method more suitable for our system and leads to better results.

2) Object Association: 3D object association is usually based on the matching score matrix built based on IoU or the distance between the BBoxes of objects [19], [20], [27]. PC3T [20] also introduced the appearance cost and motion cost, but the authors stated that these newly introduced costs contribute little to the tracking performance. Therefore, in our approach, the distance-based cost function is used for generality and computational simplicity.

The matching score matrix S_t is defined as an M by N matrix, where M is the number of tracked objects and N is the number of the detected objects at frame t. The element of S_t indicates the matching score, $s_{j,k}$, between the j-th tracked object and the k-th detected object; the higher the score, the higher the possibility of association between them. $s_{j,k}$ is calculated using (5).

$$s_{j,k} = \begin{cases} \mathbb{I}\frac{\alpha - d_{j,k}}{\alpha} & (n > I_{thres} \cap d_{j,k} < \lambda_a) \\ & \cup\, (n \le I_{thres} \cap d_{j,k} < \lambda_b) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where α is a constant, $\mathbb{I}$ is the indicator function that takes a value of 1 when the detected and tracked objects are of the same class, and $d_{j,k}$ is the distance between the detected object and the predicted position of the tracked object. We define tracked objects with the number of trajectory points n greater than I_{thres} as initialized objects. The position prediction of uninitialized objects has lower accuracy because the objects have only been tracked for a few frames. Therefore, we set a larger association range λ_b for the uninitialized objects and a smaller association range λ_a for initialized objects. This is to increase the probability of association for the uninitialized objects. However, to ensure accuracy, only the pose nodes of the initialized objects are added to the optimization graph (Section III-E). After the matching score matrix is constructed, the continuous shortest path algorithm [28] is used to solve the data association problem.

E. DL-SLOT Back-End

1) Collaborative Optimization: Collaborative optimization is defined based on graph optimization [29]. As shown in Fig. 6, all the states to be estimated are represented by nodes. The red nodes indicate the ego-pose ${}_wX_t$, and the edge between two ego-pose nodes indicates the LiDAR odometric constraint (e_{odo} in (6)).

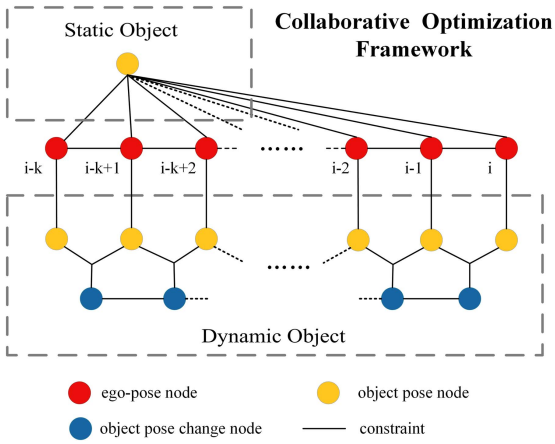


Fig. 6. Collaborative optimization framework. The red node represents the ego-pose, the yellow node represents the detected object pose, and the blue node represents the motion of the associated object. The lines connecting the nodes are the constructed constraints.

$$e_{odo}({}_wX_{t-1}, {}_wX_t) = ({}_wX_{t-1}{}^{-1} \cdot {}_wX_t) \cdot T_{t-1}^{t}{}^{-1} \tag{6}$$

The yellow nodes indicate object poses for two types of objects, dynamic and stationary, which are presented in two dashed blocks. The observation constraint e_{obs} between the object pose and ego-pose is computed using (7).

$$e_{obs}({}_wX_t, {}_wo_t^j) = ({}_wX_t{}^{-1} \cdot {}_wo_t^j) \cdot {}_lo_t^{j\,-1} \tag{7}$$

Furthermore, we add the object motion ${}^jc_{t-1}^t$ into the optimization graph, represented by the blue nodes, whose initial value ${}^jc_{t-1}^t$ is calculated by (8). The constraint between the object pose and object motion e_{mov} is defined using (9).

$${}^jc_{t-1}^t = {}_wo_{t-1}^j{}^{-1} \cdot {}_wo_t^j \tag{8}$$

$$e_{mov}({}_wo_{t-1}^j, {}_wo_t^j, {}^jc_{t-1}^t) = ({}_wo_{t-1}^j{}^{-1} \cdot {}_wo_t^j) \cdot {}^jc_{t-1}^t{}^{-1} \tag{9}$$

We assume a dynamic object moves at a constant velocity over a short time period, and then the constant velocity constraint between the motion node of the associated object e_{cons} is given as (10).

$$e_{cons}({}^jc_{t-2}^{t-1}, {}^jc_{t-1}^t) = {}^jc_{t-2}^{t-1}{}^{-1} \cdot {}^jc_{t-1}^t \tag{10}$$

Finally, our optimization problem is defined using (11).

$$\begin{aligned} \boldsymbol{X}^* = \underset{\boldsymbol{X}}{argmin} \Bigg\{ & \sum_{i\in[t-K+1,t]} \Bigg(\left\| e_{odo}({}_wX_{i-1}, {}_wX_i) \right\|^2_{Q_{odo}} \\ & + \sum_{j\in O_i} \Bigg(\left\| e_{obs}({}_wX_i, {}_wo_i^j) \right\|^2_{Q_{obs}} \\ & + \left\| e_{mov}({}_wo_{i-1}^j, {}_wo_i^j, {}^jc_{i-1}^i) \right\|^2_{Q_{mov}} \\ & + \left\| e_{cons}({}^jc_{i-2}^{i-1}, {}^jc_{i-1}^i) \right\|^2_{Q_{cons}} \Bigg) \Bigg) \Bigg\} \end{aligned} \tag{11}$$

where i represents the i-th frame, j represents the index of the tracked object in O_i, and Q represents the covariance matrix.

We use a sliding window of size K frames to limit the scale of optimization in order to reduce the computational complexity and ensure real-time performance. When the window slides, the Schur complement is used to preserve the constraints that will be removed from the windows.

2) Global Optimization: To further eliminate the accumulated error, we maintain a global pose graph that contains only the LiDAR odometry and loop closure constraints. When a loop closure is detected between the $t-d$-th frame and the t-th frame, its constraint is defined by:

$$e_{loop}({}_wX_{t-d}, {}_w X_t) = ({}_wX_{t-d}{}^{-1}\,{}_wX_t)T_{t-d}^{t}{}^{-1} \tag{12}$$

where T_{t-d}^{t} is the LiDAR odometry between the $t-d$-th frame and the t-th frame. Therefore, the objective function of the global optimization problem is defined using (13).

$$\boldsymbol{X}_L^* = \underset{\boldsymbol{X}_L}{argmin}\left\{ \sum_{i\in[t-d+1,t]} \|e_{odo}({}_wX_{i-1}, {}_w X_i)\|^2_{\Sigma_{odo}} + \|e_{loop}({}_wX_{t-d}, {}_w X_t)\|^2_{\Sigma_{odo}} \right\} \tag{13}$$

F. Implementation Details

The object detector adopted in this study is PointRCNN [18] with the fine-tuned weights on the KITTI dataset. The LiDAR odometry is based on the widely used LeGO-LOAM. The loop is detected based on Scan Context [30]. Finally, the optimization is implemented based on g2o [29].

In object association, there are two main reasons why a tracked object is not associated: either the object is out of the effective sensing range of the LiDAR, or the object detection algorithm misses the object because the sensor is occluded for a short time. To mitigate the latter situation, we set an upper limit σ_{init}^{miss} for the number of unassociated frames for tracked objects. When the number of missed associations is greater than σ_{init}^{miss}, the object is considered to be out of the sensing range. Otherwise, we execute a supplementary detection using the predicted position of the object at frame t and its orientation at frame $t-1$ to facilitate continued tracking, as shown in Fig. 7.

While stationary objects are being tracked, their global pose may not be stable over continuous observations due to noisy detection or localization. Therefore, we add only one object pose node to the collaborative optimization when the object is judged as stationary, i.e., when its velocity is less than a velocity threshold V_{thres}, which ensures the uniqueness of the stationary object's global position.

However, there may be objects that are neither constantly stationary nor dynamic. When a stationary object is judged to be dynamic, our system will add pose and motion nodes for it and establish constraints. Conversely, when an object changes from dynamic to stationary, our system will stop adding new pose nodes for it, thereby maintaining its unique global position.

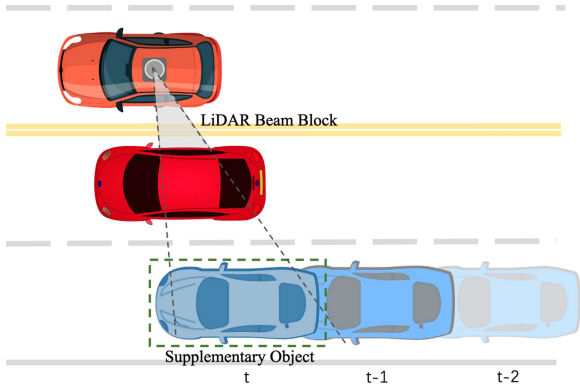


Fig. 7. Supplementary detection construction. Because the LiDAR is blocked by the red vehicle, the blue vehicle cannot be detected at time t. In this case, the system supplements the missed objects with the predicted position to increase the robustness of collaborative optimization.

IV. Experimental Results

We conducted experiments to evaluate the performance of the proposed DL-SLOT method using the public dataset KITTI [31]. The KITTI Tracking dataset was used to evaluate the effectiveness of collaborative optimization of the SLOT system from the perspective of trajectory accuracy and 3D multi-object tracking, respectively. This dataset was collected in urban areas and highways and contains raw point clouds, GPS data, and ground truth labels that can be used to evaluate tracking. The longest sequences, which are 04, 07, 08, 09, 11, 15, 18, and 19, were chosen as the valuation set (except for sequence 20) because longer sequences allow for a better evaluation of the SLOT system. Sequence 20 was excluded because its scenario is a highway, which is a degraded scenario for LiDAR odometry. The KITTI Odometry dataset was used to evaluate global optimization, as it contains loops.

For all the experiments, we used the settings $K = 10$, $\alpha = 100$, $V_{thres} = 0.1$, and $\sigma_{init}^{miss} = 1$. Additionally, the thresholds λ_a, λ_b, and I_{thres} for object association were set to 2 m, 3.5 m, and 5, respectively. To reduce the impact of inaccurate detection due to sparse point scanning, objects located from 3 to 45 m in front of the ego-vehicle were considered in the experiments. The experiments were carried out on a PC running Ubuntu 20.04 and equipped with an Intel Core i7-9800 3.8 GHz processor and 32 G RAM.

A. Ego-Trajectory Evaluation

We adopted the root-mean-squared error (RMSE) of absolute trajectory error (ATE) to assess the accuracy of trajectories. For comparison, the widely used LiDAR odometry method LeGO-LOAM was chosen as the baseline method. Because LeGO-LOAM does not consider the influence of dynamic objects, we filtered out the points belonging to PDOs to generate the LeGO-LOAM* results. Our method is referred to as DL-SLOT w/o loop when only collaborative optimization is performed, and DL-SLOT when global optimization is performed. As none of the sequences in the KITTI Tracking dataset contain loop closures, only the results of DL-SLOT w/o loop are presented here.

TABLE II
RMSE(M)/RMSE(RAD) RESULTS OF LEGO-LOAM, LEGO-LOAM*, AND OUR METHOD ON THE KITTI TRACKING DATASET

Seq	LeGO-LOAM	LeGO-LOAM*	DL-SLOT w/o loop
04	1.342/0.273	1.352/0.268	**1.151/0.257**
07	**0.934**/0.274	1.286/**0.273**	1.002/0.274
08	1.786/**0.305**	1.785/0.315	**1.765**/0.319
09	3.739/1.122	3.822/1.125	**3.548/1.121**
11	0.225/**0.206**	0.279/0.289	**0.218**/0.266
15	0.352/1.485	0.322/0.516	**0.300/0.374**
18	0.713/0.173	**0.669**/0.175	0.676/**0.172**
19	1.784/1.181	1.775/1.306	**1.552/0.564**
mean	1.359/0.627	1.411/0.533	**1.277/0.418**

TABLE III
EVALUATION RESULTS FOR OBJECT DETECTION AND OBJECT TRACKING ON THE KITTI TRACKING DATASET

Seq	Detected Object		Tracked Object		
	Recall	Precision	MOTA	Recall	Precision
04	0.9031	0.9824	0.8432	0.9153	0.9460
07	0.9834	0.9280	0.8923	0.9853	0.9251
08	0.9304	0.9899	0.8762	0.9317	0.9671
09	0.9393	0.9335	0.8448	0.9494	0.9205
11	0.8913	0.9881	0.8571	0.9027	0.9696
15	0.9442	0.9694	0.9103	0.9558	0.9671
18	0.8953	0.9574	0.8448	0.9036	0.9466
19	0.9855	0.9262	0.8718	0.9879	0.9221
mean	0.9341	0.9594	0.8647	0.9415	0.9455

The results are presented in Table II. For half of the sequences, the localization accuracy of LeGO-LOAM* is lower than that of LeGO-LOAM. This is because although filtering out PDOs prevents the selection of unreliable feature points on moving objects, the stationary objects in the PDOs are also removed, which results in feature sparsity and reduces localization accuracy. The localization accuracy of our method is higher than that of LeGO-LOAM and LeGO-LOAM* on most sequences, indicating that collaborative optimization can effectively improve localization accuracy, i.e., an autonomous vehicle can optimize ego-poses through continuous observation of the dynamic objects' motion. In addition, the error maps of the ego-trajectory of sequences 04 and 09 are shown in Fig. 11(a).

B. Object Tracking Evaluation

To evaluate the object tracking performance of the proposed method, we used the Precision-Recall metrics and the multi-object tracking accuracy (MOTA) metric [32].

Table III shows the result of object detection and tracking. The IoU with ground truth label for each tracked object should be above a threshold IoU_{thres} to be considered a successful match. We set IoU_{thres} to 0.5 for this evaluation. It can be observed that the Recall for object tracking is higher than that for object detection, which indicates the effectiveness of the proposed object association and supplementary detection methods. However, the precision of the tracking results is decreased, and we find that this is mainly because some objects in the dataset miss ground truth labels. As shown in Fig. 8(a), the object with ID 6 is being tracked by our method. In the subsequent frame, the object is too close to the ego-vehicle to be detected (Fig. 8(b)). However, the proposed supplementary detection method correctly generates

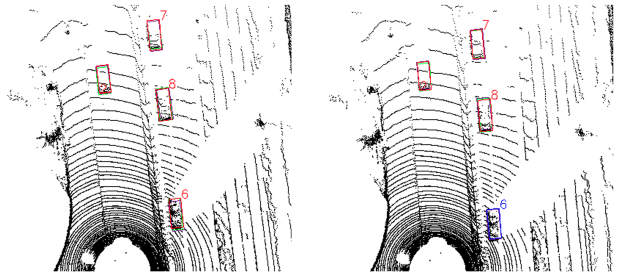


Fig. 8. False detection due to lack of the ground truth labeling. The red, green, and blue bounding boxes indicate the detection, ground truth, and supplementary detection results, respectively.

TABLE IV
COMPARISON OF DIFFERENT MULTI-OBJECT TRACKING ALGORITHMS ON THE KITTI TRACKING DATASET

Method	MOTA		
	IoU_{thres} = 0.25	IoU_{thres} = 0.5	IoU_{thres} = 0.7
AB3DMOT	0.8643	0.8480	0.6087
PC3T	0.8836	0.8773	0.7103
PC3T*	0.8815	0.8596	0.6830
Ours	0.8741	0.8647	0.6844

the BBox for this object and the tracking continues, as shown by the blue box in Fig. 8(b). However, due to the missing label in the dataset, this supplementary object is misjudged as a false detection in this frame.

We further compared the tracking results of our SLOT system with two popular LiDAR-based real-time 3D multi-object tracking methods, AB3DMOT [19] and PC3T [20]. AB3DMOT first proposed to evaluate 3D multi-object tracking directly in 3D space, which is more suitable for LiDAR-based methods, and fully evaluated MOTA with different 3D IoU thresholds. PC3T is the state-of-the-art (SOTA) LiDAR-based real-time 3D multi-object tracking method. It performs tracking in the world coordinate system, so it requires the ground truth ego-poses as input, whereas AB3DMOT performs tracking in the local coordinate system. Therefore, we also evaluated the results of PC3T using the poses generated by LeGO-LOAM (referred to as PC3T*). We conducted the experiment with the same object detector, PointRCNN [18], for all these methods.

The results are shown in Table IV. As can be seen, PC3T and our method performed better than AB3DMOT in terms of MOTA for all IoU thresholds. This is because PC3T additionally considers the confidence of object tracking, and sets a larger association range for objects with low confidence. Our results are similar to those of PC3T*. Specifically, the results of our method are better than those of PC3T* when a higher IoU threshold is used for the evaluation, indicating that our method obtains more accurate object poses. In summary, our SLOT system is comparable to SOTA 3D multi-object tracking methods in terms of tracking, whereas the proposed system simultaneously estimates the ego-poses.

The estimated velocity, pose, and trajectory of the tracked objects were then examined to further investigate the accuracy of object state estimation through collaborative optimization.

TABLE V
EVALUATION RESULTS OF OBJECT POSE ACCURACY AND AVERAGE VELOCITY ON THE KITTI TRACKING DATASET

Seq	Obj_id	Tracked frame length	RMSE(m)	RMSE(rad)	v_{true}(km/h)	$v_{estimate}$(km/h)
04	2	88	1.311	0.224	42.125	42.264
07	55	45	1.111	0.407	13.608	13.736
08	8	184	0.468	0.054	46.435	46.442
09	9	30	0.332	0.048	27.391	27.749
11	0	372	0.321	0.062	22.661	22.771
15	18	169	0.212	0.039	9.938	11.435
18	3	257	0.384	0.550	15.799	16.088
19	63	131	0.484	0.317	10.613	10.746

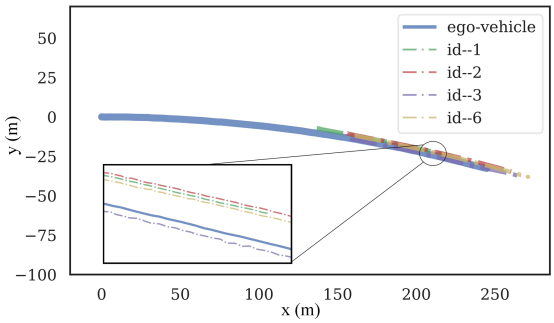


Fig. 9. Trajectories of the four main tracked objects and the ego-vehicle in sequence 18.

TABLE VI
THE TRACKED STATIONARY OBJECT INFORMATION

Seq	Obj_id	v_{true} (km/h)	Tracked frame length
09	8	1.833	64
09	16	1.909	29
11	28	1.149	128
11	29	1.359	128
11	30	1.036	118
11	35	1.192	111
15	2	0.648	349
19	42	1.386	204

1) Evaluation of the Object Pose and Average Velocity: We selected the dynamic object with the longest tracking length in each sequence and evaluated its trajectory and average velocity. The ground truth and the estimated average velocity of the object are marked as v_{true}[2] and $v_{estimate}$, respectively. The results are shown in Table V. It can be observed that the trajectory accuracy of the tracked objects is acceptable, and that the estimated average velocity is close to the ground truth. In addition, a qualitative experimental result for the trajectories of the main tracked objects in sequence 18 is shown in Fig. 9.

Additionally, we analyzed the velocities of the tracked objects that are considered stationary in the above sequences and that have tracking lengths longer than 50 frames, as shown in Table VI. These objects have a globally unique pose and zero velocity in our collaborative optimization, providing high-quality observation constraints for ego-poses optimization. However, these stationary objects all have a small ground truth average velocity (v_{true}), which is due to the noisy annotations in the dataset.

[2]v_{true} is obtained by dividing the length of the ground truth trajectory during tracking by the length of the tracking time.

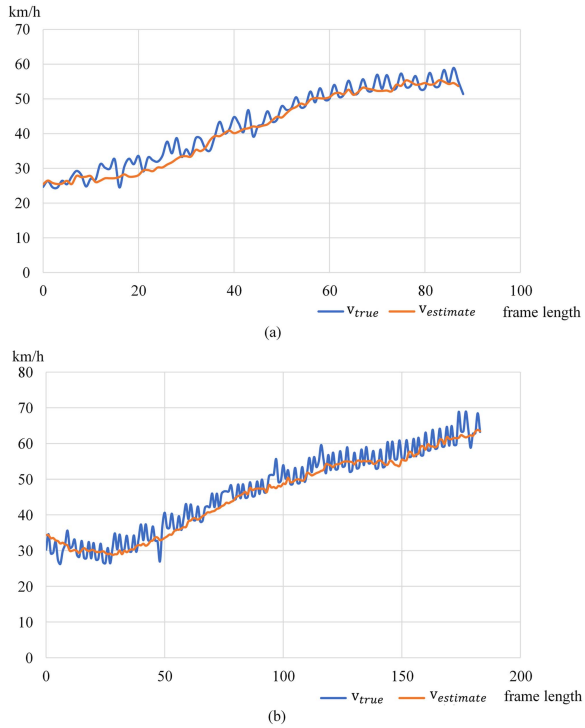
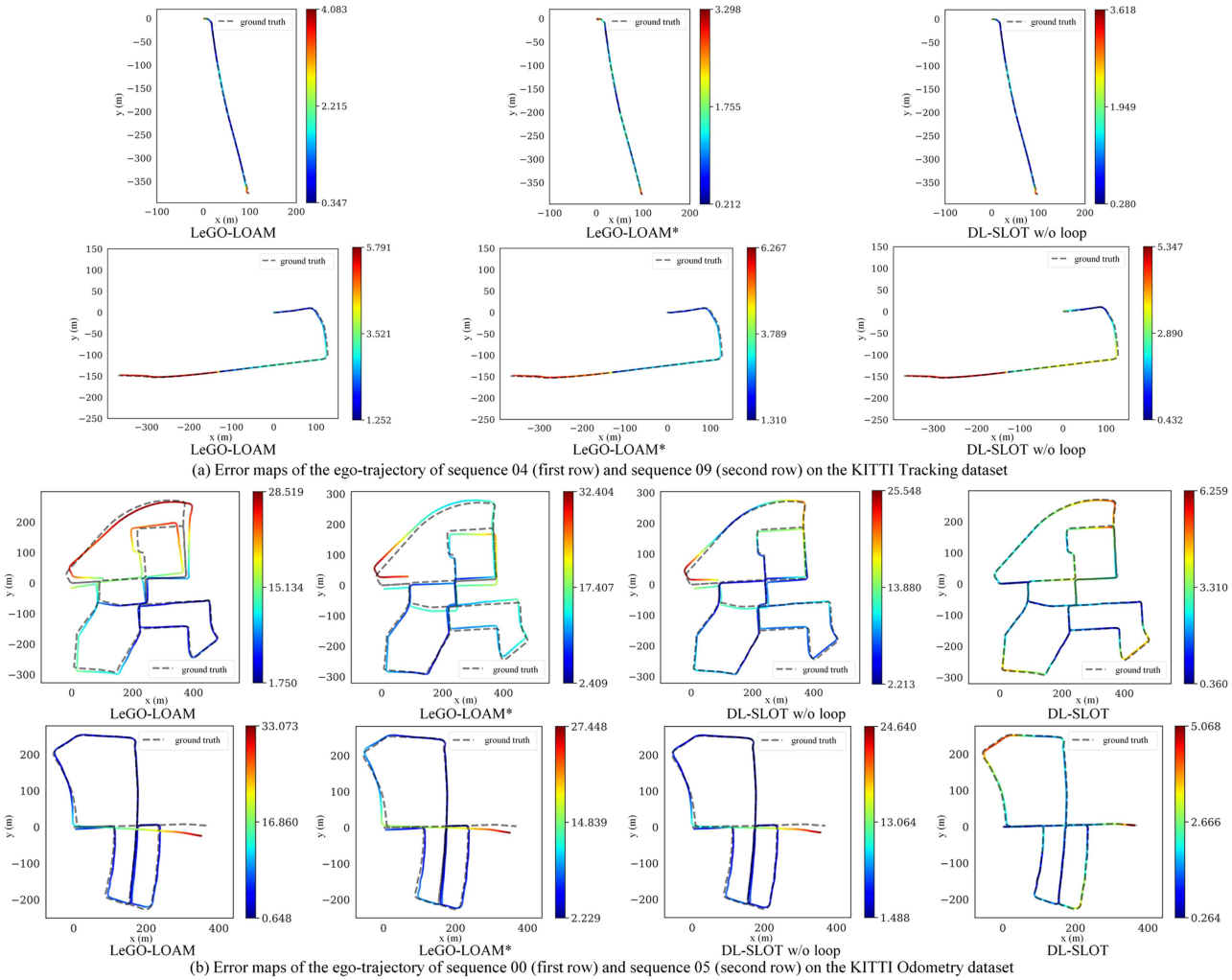


Fig. 10. Comparison of the ground truth and the estimated instantaneous velocity. (a) and (b) show the results of the object with ID 2 in sequence 04 and the object with ID 8 in sequence 08, respectively.

2) Evaluation of the Instantaneous Velocity: It is crucial to accurately estimate the instantaneous velocities of dynamic objects. We selected two objects with the highest velocities in Table V for evaluation. The blue line in Fig. 10, represents the ground truth instantaneous velocity of the object, and the orange line represents the estimated instantaneous velocity obtained by collaborative optimization. It can be observed that the ground truth instantaneous velocity fluctuates greatly, which is again caused by the low accuracy of the object annotation in the dataset. However, the estimated instantaneous velocities are smooth and fall within the middle of the ground truth velocity fluctuation range. This result indicates that it is feasible to add the constant velocity constraint in collaborative optimization.

C. Global Optimization Evaluation

Finally, we conducted experiments on the entire DL-SLOT system that integrates loop closure and global optimization. The experiments were performed on the KITTI Odometry datasets



(a) Error maps of the ego-trajectory of sequence 04 (first row) and sequence 09 (second row) on the KITTI Tracking dataset

(b) Error maps of the ego-trajectory of sequence 00 (first row) and sequence 05 (second row) on the KITTI Odometry dataset

Fig. 11. Comparison of ego-trajectory error maps for different methods on the KITTI Tracking and Odometry datasets. The gray dashed line represents the ground truth.

TABLE VII
RMSE(M)/RMSE(RAD) RESULTS OF LEGO-LOAM, LEGO-LOAM*, DL-SLOT W/O LOOP AND DL-SLOT ON THE KITTI ODOMETRY DATASET

Seq	LeGO-LOAM	LeGO-LOAM*	DL-SLOT w/o loop	DL-SLOT
00	15.190 / 0.687	14.700 / 0.681	10.849 / 0.684	**2.976 / 0.032**
05	8.894 / 0.532	9.283 / 0.531	7.012 / 0.530	**2.030 / 0.022**
07	2.581 / 0.774	2.501 / 0.775	2.620 / 0.776	**1.428 / 0.021**
08	10.425 / 0.609	18.507 / 0.615	10.829 / 0.609	**8.303 / 0.053**
09	5.813 / 0.659	6.769 / 0.666	5.685 / 0.658	**3.009 / 0.037**
mean	8.581 / 0.652	10.352 / 0.654	7.399 / 0.651	**3.549 / 0.033**

TABLE VIII
AVERAGE TIME-CONSUMING OF THE MAIN FUNCTIONAL MODULES FOR PROCESSING ONE SCAN

Module	Average Runtime (ms)
LiDAR Odometry	33.6
Multi-object Tracking	9.3
Collaborative Optimization	13.5

that contained loops. The evaluation metric is the RMSE of ATE, and the compared methods are LeGO-LOAM, LeGO-LOAM*, DL-SLOT w/o loop, and DL-SLOT. The experimental results are shown in Table VII. It can be observed that the localization accuracy of LeGO-LOAM is significantly higher than that of LeGO-LOAM*, because the Odometry dataset includes more static objects than the Tracking dataset, and filtering out all the PDOs reduces the localization accuracy. Additionally, DL-SLOT w/o loop shows higher localization accuracy than LeGO-LOAM and LeGO-LOAM*, and DL-SLOT has the highest localization accuracy. These results suggest that collaborative optimization improves ego-pose accuracy, and global optimization eliminates the accumulated errors generated by odometry and improves the overall accuracy. Additionally, the error maps of the ego-trajectory of sequences 00 and 05 are shown in Fig. 11(b).

D. Performance Evaluation

We calculate the average time-consumption of the main functional modules except for object detection, which can be accelerated by GPU. As shown in Table VIII, the method proposed in this study enables real-time performance.

V. Conclusion

This article presented an effective and robust LiDAR-based SLOT method that can be used on dynamic road scenes. The method integrates the state estimation of PDOs and the ego-vehicle into a unified collaborative optimization framework. As a result, SLAM and object tracking can be performed simultaneously to mutually benefit. Additionally, we proposed an effective sliding window-based association method, which can accurately predict the tracked object's position using its trajectory. The results indicate that DL-SLOT can considerably improve localization accuracy and accurately track dynamic and stationary objects, which makes our method widely applicable to dynamic road scenes.

In addition, we found that the pose error of the moving objects mainly originates from the rotation error of the ego-vehicle. Therefore, in the future, we will investigate a more efficient and robust collaborative optimization framework and tightly integrate measurement data from an inertial measurement unit to further improve system performance, especially the pose accuracy of moving objects.

References

[1] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst. Conf.*, 2014, pp. 1–9.

[2] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.

[3] Q. Li et al., "LO-Net: Deep real-time lidar odometry," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8473–8482.

[4] C. Qian, Z. Xiang, Z. Wu, and H. Sun, "RF-LIO: Removal-first tightly-coupled lidar inertial odometry in high dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 4421–4428.

[5] V. Vaquero, K. Fischer, F. Moreno-Noguer, A. Sanfeliu, and S. Milz, "Improving map re-localization with deep movable objects segmentation on 3D liDAR point clouds," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 942–949.

[6] S. Zhao, Z. Fang, H. Li, and S. Scherer, "A robust laser-inertial odometry and mapping method for large-scale highway environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1285–1292.

[7] X. Chen, A. Milioto, E. Palazzolo, P. Giguére, J. Behley, and C. Stachniss, "SuMa : Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4530–4537.

[8] F. Moosmann and C. Stiller, "Joint self-localization and tracking of generic objects in 3D range data," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1146–1152.

[9] T. Ma and Y. Ou, "MLO: Multi-object tracking and lidar odometry in dynamic envirnoment," 2022, *arXiv:2204.11621*.

[10] S. Yang and S. Scherer, "CubeSLAM: Monocular 3-D object SLAM," *IEEE Trans. Robot.*, vol. 35, no. 4, pp. 925–938, Aug. 2019.

[11] J. Zhang, M. Henein, R. Mahony, and V. Ila, "VDO-SLAM: A visual dynamic object-aware slam system," 2020, *arXiv:2005.11052*.

[12] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "DynaSLAM II: Tightly-coupled multi-object tracking and slam," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5191–5198, Jul. 2021.

[13] M. Gonzalez, E. Marchand, A. Kacete, and J. Royan, "TwistSLAM: Constrained slam in dynamic environment," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6846–6853, Jul. 2022.

[14] M. Gonzalez, E. Marchand, A. Kacete, and J. Royan, "TwistSLAM: Fusing multiple modalities for accurate dynamic semantic slam," 2022, *arXiv:2209.07888*.

[15] M. Henein, G. Kennedy, R. Mahony, and V. Ila, "Exploiting rigid body motion for slam in dynamic environments," *Environments*, vol. 18, 2018, Art. no. 19.

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 2564–2571.

[18] S. Shi, X. Wang, and H. Li, "PointrCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 770–779.

[19] X. Weng, J. Wang, D. Held, and K. Kitani, "3D multi-object tracking: A baseline and new evaluation metrics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 10359–10366.

[20] H. Wu, W. Han, C. Wen, X. Li, and C. Wang, "3D multi-object tracking in point clouds based on prediction confidence-guided data association," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5668–5677, Jun. 2022.

[21] C. Kim, F. Li, and J. M. Rehg, "Multi-object tracking with neural gating using bilinear LSTM," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 200–215.

[22] H.-N. Hu et al., "Joint monocular 3D vehicle detection and tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 5390–5399.

[23] X. Farhodov, K.-S. Moon, S.-H. Lee, and K.-R. Kwon, "LSTM network with tracking association for multi-object tracking," *J. Korea Multimedia Soc.*, vol. 23, no. 10, pp. 1236–1249, 2020.

[24] H. Wu, Q. Li, C. Wen, X. Li, X. Fan, and C. Wang, "Tracklet proposal network for multi-object tracking on point clouds," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 1165–1171.

[25] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 4889–4895.

[26] P. Petrov and F. Nashashibi, "Modeling and nonlinear adaptive control for autonomous vehicle overtaking," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 4, pp. 1643–1656, Aug. 2014.

[27] K. Burnett, S. Samavi, S. Waslander, T. Barfoot, and A. Schoellig, "aUToTrack: A lightweight object detection and tracking system for the SAE autodrive challenge," in *Proc. IEEE 16th Conf. Comput. Robot Vis.*, 2019, pp. 209–216.

[28] R. Ahuja, T. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms and Applications*. Upper Saddle River, NJ, USA: Prentice Hall, 1993.

[29] G. Grisetti, R. Kümmerle, H. Strasdat, and K. Konolige, "g2o: A general framework for (hyper) graph optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 9–13.

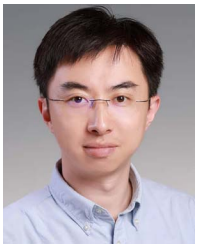
[30] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3D point cloud map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4802–4809.

[31] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[32] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," *EURASIP J. Image Video Process.*, vol. 2008, pp. 1–10, 2008.

Xuebo Tian was born in 1997. He received the B.S. degree in computer science and technology from Zhejiang University of Technology, Hangzhou, China, in 2019, and the M.S. degree in computer science and technology from Tongji University, Shanghai, China, in 2022. His research interests include state estimation, object tracking, and graph optimization in simultaneous localization and mapping for autonomous driving.

Zhongyang Zhu was born in 1999. He received the B.S. degree in vehicle engineering in 2022 from Tongji University, Shanghai, China, where he is currently working toward the M.S. degree in computer science. His research interests include multi-object tracking, sensor fusion-based, and other sensors-aided LiDAR-inertial simultaneous localization and mapping for autonomous driving.



Junqiao Zhao (Member, IEEE) was born in 1983. He received the B.S. degree in GIS technology from Southeast University, Nanjing, China, in 2006 and the Ph.D. degree in photogrammetry and remote sensing from the State Key Laboratory of Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, China, in 2011. In 2012, he joined as a Postdoctoral Researcher with the GIS Technology Group, Delft University of Technology, Delft, The Netherlands. In 2014, he became an Assistant Professor with the Department of Computer Science and Technology, Tongji University, Shanghai, China, and an Associate Professor in 2019. His research interests include the simultaneous localization and mapping for autonomous driving, intelligent planning, and simulation.

Gengxuan Tian received the B.S. degree in computer science in 2021 from Tongji University, Shanghai, China, where he is currently working toward the M.S. degree in computer science. His research interests include LiDAR-Based place recognition and simultaneous localization and mapping for autonomous driving.

Chen Ye (Member, IEEE) received the B.S. degree in automation and the Ph.D. degree in computer science from Tongji University, Shanghai, China, in 2003 and 2015, respectively. He is currently a Researcher with the College of Electronic and Information Engineering, Tongji University. His research interests include autonomous driving, machine learning, image processing, Big Data analysis, and its application in the field of industrial intelligence.