LiDAR-Based Object-Level SLAM for Autonomous Vehicles

Bingyi Cao, Ricardo Carrillo Mendoza, Andreas Philipp and Daniel Göhring

***Abstract*— Simultaneous localization and mapping (SLAM) is an essential technique for autonomous driving. Recently, combining image recognition technology to generate semantically meaningful maps has become a new trend in visual SLAM research. However, in the field of LiDAR SLAM, this potential has not been fully explored. We propose a novel object-level SLAM system using 3D LiDARs for autonomous vehicles. We detect and track poles, walls, and parked cars, which are common along urban roads. This paper presents how we process the measurement data of three different shapes of objects to build a graph-based optimization system and facilitate the geometric distribution of poles to detect loops. Experiments were carried out on datasets collected with a test vehicle in city traffic. The results show that our object-level SLAM system can build precise and semantically meaningful maps and produce more accurate pose estimations compared to the state-of-the-art systems on our datasets.**

I. INTRODUCTION

The real-time SLAM technique can help autonomous vehicles build environment models and localize themselves reliably in unknown environments. It is a critical topic in autonomous driving research. Both vision-based and LiDAR-based SLAM approaches have been widely studied, and a series of important methods have been proposed [1]. In recent years, researchers in the VSLAM area have combined image recognition with SLAM to construct object-oriented maps with semantic information [2], [3]. Compared with traditional maps based on feature points or occupancy grids, semantic maps have the advantage of information richness [4]. Robots can better understand their surroundings using this type of map.

In the field of LiDAR SLAM, due to the sparsity of laser scanning points, it is naturally more challenging to obtain semantic information from LiDAR measurements than images. Therefore, research on LiDAR-based semantic SLAM is less explored compared to visual semantic SLAM. Some examples are [5] and [6]. However, with the advantage of having more precise measurements, it has the potential to generate a highly accurate map with semantic meaning, which could benefit robots in many application scenarios.

This paper proposes an object-level SLAM system based on 3D LiDARs. The algorithm detects poles, walls, and cars and fuses this information with the ego-car's wheel odometry. The system performs real-time object-mapping and localization, the visualization of the system is illustrated in Fig.1. The contributions of this paper are as follows:

- We implement a LiDAR SLAM system that can create a semantic map in real-time and provide more accurate vehicle poses than the state-of-the-art systems on our datasets.
- A new loop detection method using the geometric distributions of surrounding poles is introduced.

All authors are with the Department of Mathematics and Computer Science, Freie Universität Berlin, 14195 Berlin, Germany.

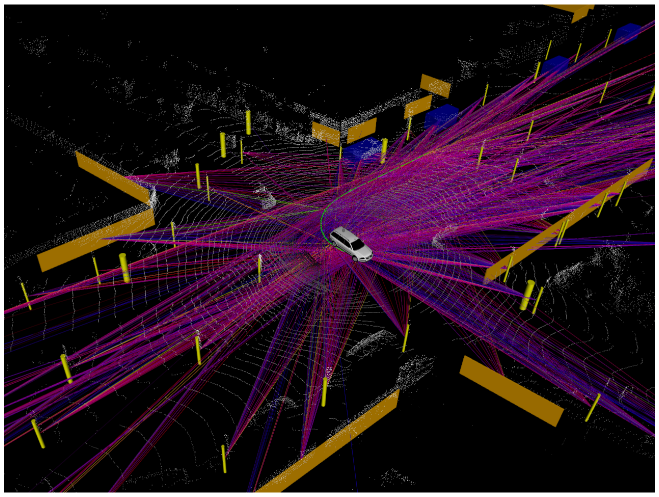
Fig. 1. Run time visualization of our proposed object-level LiDAR SLAM system. The yellow cylinders, the orange planes, and the blue cuboids represent the detected poles, walls, and parked cars, respectively. Colored lines connect each frame to the objects it observes. The green line is the optimized trajectory. The detected and tracked objects provide plenty of constraints across multiple frames for local optimization. This enables our algorithm to achieve robust frame tracking and low drift.

To our knowledge, this is the first object-oriented LiDAR SLAM system that processes circular, line-shaped, and rectangular objects at the same time.

II. RELATED WORK

SLAM is an important topic in robotics research. In the past two decades, many researchers have been involved in exploring solutions to this research problem. Extensive research results have been published and shared in the research community. Especially in vision-based solutions, many feature-points-based [7], direct [8] or semi-direct methods [9] have presented great reliability and accuracy. In the field of LiDAR SLAM, assisted by the high accuracy of LiDAR measurement, many methods have achieved low-drift pose estimation and high-consistent map construction, such as the ICP-based approaches [10], feature-point-based methods [11], and landmark matching [12] methods.

Recently, great progress has been made in vision-based object recognition. Researchers have tried to add semantic information to the SLAM system and build informative environment maps [13], [14], [15]. One category of approaches applied neural network methods to partition the

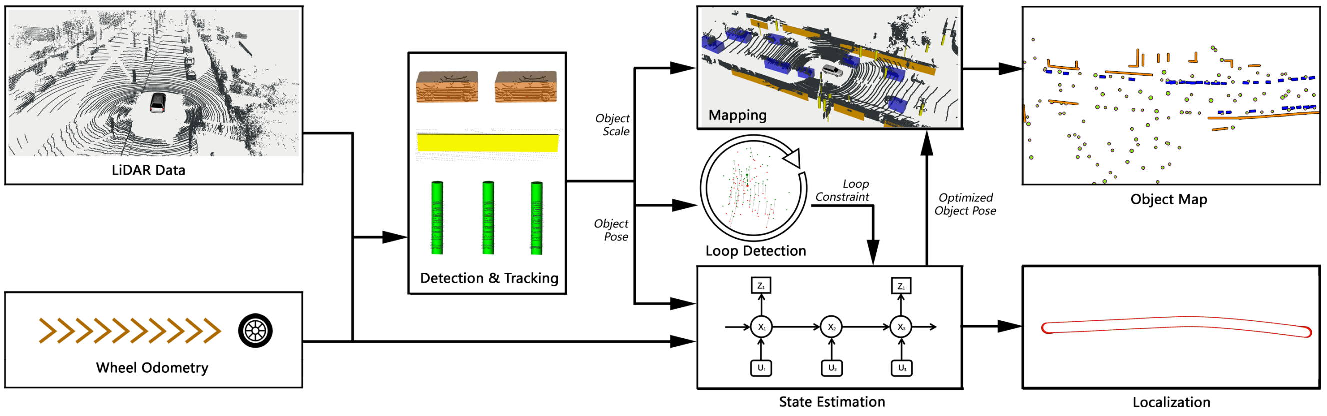

Fig. 2. The pipeline of the proposed object-oriented SLAM system. A detection and tracking module finds vehicles, walls, and poles from the LiDAR data. The measurements of the objects and wheel odometry are integrated to build a factor graph for pose estimation. A loop detection module searches for loops from the detection, and a mapping module produces an object-based map of the environment.

map generated by SLAM systems and assigned semantic labels to the segments [14]. Another type of approach is so-called 'object-oriented SLAM' [16], [17], [18]. These systems recognize objects at run-time and use them to refine poses continually.

Semantic SLAM using information from LiDARs is not a widely explored field. One example is SegMap [19], which divided 3D point clouds into segments with three types of meaningful descriptors (vehicles, buildings, and others). It is shown that object detection assists the data association process and reduces drift. The research in [5] built a surfel-based dense map with semantic labels using the ICP method. The study in [20] proposed an end-to-end method to discover trees and estimate their diameters in the forest. This work extended the previous work from [11] by adding tree and ground labels to the laser points.

The above-introduced LiDAR SLAM systems performed data matching and map building at the point cloud level. Unlike these works, the system proposed in this paper detects and tracks three types of objects that can be commonly found in the urban street, including poles, walls, and stationary vehicles. These objects are represented as 2D shapes and can be processed efficiently.

III. LiDAR Object SLAM System

This section introduces the main parts of our proposed SLAM method for autonomous driving in urban environments. First, we will introduce the selected objects for creating the semantic map and the corresponding detection methods. Then, we will explain the methodology for pose estimation and map update, and finally, the loop detection method will be described. Fig.2 gives the pipeline of the system. The input data are LiDAR range images and the wheel odometry. A detection and tracking module obtains the objects of interest from the input. The measurement data of objects are separated into two categories. The pose-related measurements are integrated with wheel odometry to build a factor graph for state estimation. The scale-related measurements are sent to the mapping module to adjust the size of the mapped objects. Detected poles are also used for loop finding. The output of the system is optimized vehicle poses at 100 Hz, and an object map with semantic meaning.

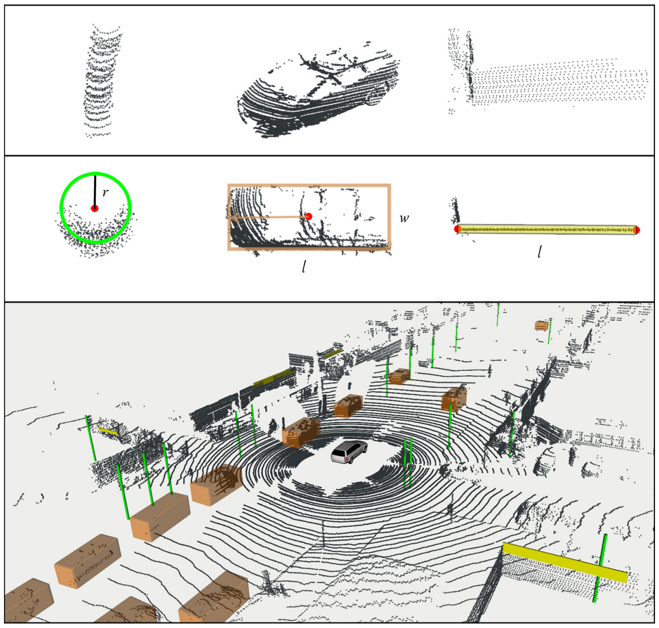

Fig. 3. An illustration of object detection. The top section of the figure shows the targeted objects in 3D space. The middle section shows the measurement data we are interested in after projecting the objects onto the ground plane. The bottom section of the figure presents the detection and tracking results while the algorithm is running on a test dataset.

A. Object Detection

Walls and pole-like structures such as tree trunks, street lights, and traffic signs are long-term stable landmarks. They are considered to be robust features for localization and mapping. To recognize poles and walls, we search for measurement segments from each laser scan line on the range image that could belong to an object according to the measurement points' geometric properties. And then merge the scan segments vertically to obtain the desired objects. The detailed implementation can be found in [21].

To extend the proposed SLAM system's applicable scenarios and enable vehicles to localize themselves in areas with fewer poles and walls, we expanded the objects to parked vehicles on the side of the road, which are also distinguishable in the LiDAR measurements. They can effectively increase both the type and number of landmarks, and therefore improve the robustness and accuracy of a SLAM system.

In order to detect parked cars, we use a multi-object tracker. It implements a complete stack of functions to detect and track static and dynamic objects. The detection is performed on LiDAR range images de-skewed using the wheel odometry. Then we remove the ground pixels, cluster the remaining pixels into objects, and project them onto the ground plane. A bounding box is fitted for each object by the rotating caliper algorithm [22]. The states of the objects are filtered by an Interacting Multiple Model Kalman filter. Then the object class can be evaluated based on the measured length, width, and height, the motion type and, if available, on the roadmap location. The bottom section of Fig. 3 illustrates the results of object detection and tracking.

B. SLAM with LiDAR Objects

The object detector reports measurements of poles, wall segments, and parked vehicles while driving in the urban environment. The detection of objects was conducted in 3D space and then projected them onto the ground. Therefore, we receive circles, line segments, and rectangles as the measurements of poles, walls, and vehicles, respectively, as can be seen in the top and middle section of Fig. 3. The measurements of three types of objects can be denoted as z_p, z_l, and z_v:

$$\begin{aligned} z_{p,i} &= [x_i, y_i, r_i]^T \\ z_{l,i} &= [x_{1i}, y_{1i}, x_{2i}, y_{2i}]^T \\ z_{v,i} &= [x_i, y_i, \theta_i, w_i, l_i]^T, \end{aligned} \tag{1}$$

where x_i, y_i, and θ_i are the geometric center and the orientation of a detected object. r_i is the radius of a pole. For walls, the start and end point of the corresponding line segments are given. w_i and l_i stand for the width and length of a detected parked vehicles. Then we separate position and orientation information from the measurements for back-end optimization and extract the scale information for map update. The motion and measurement error can then be written as follows:

$$\begin{aligned} e_{u,k} &= f(x_{k-1}, u_k) - x_k \\ e_{p,ki} &= \pi_p(z_{p,ki}) - h_p(x_k, m_i) \\ e_{l,ki} &= \pi_l(z_{l,ki}) - h_l(x_k, m_i) \\ e_{v,ki} &= \pi_v(z_{v,ki}) - h_v(x_k, m_i), \end{aligned} \tag{2}$$

for each frame in time k, f is the motion function, h_p, h_l, and h_v are measurement functions of poles, lines, and vehicles, respectively. π_p extracts the first two elements of $z_{p,ki}$, which give the center position of a pole. π_v takes the first three elements of $z_{v,ki}$, which represent the pose of the rectangle. For walls, the function is $\pi_l(z_{l,ki}) = [\rho_{ki}, \phi_{ki}]^T$, where ρ_{ki} is the perpendicular distance from the sensor origin to the line where the line segment is located, and ϕ_{ki} is the angle between the sensor's x-axis and the line. Note that a wall segment is modeled as an infinite line represented by these two parameters in the back-end optimization.

To formulate our object-level SLAM problem, we use $X = \{x_k\}_{k=1}^K$ to represent vehicle poses in $SE(2)$ along K frames. Assume the object map is $M = \{m_i\}_{i=1}^I$ and $M = \{P, L, V\}$, where P, L, and V are sets of poles, lines, and vehicles. The inputs from the odometry are $U = \{u_k\}_{k=1}^K$. $Z^* = \{z_{ki}^*\}_{k=1,i=1}^{K,I}$ are measurements of map elements at each pose, processed through function π_p, π_l, and π_v. We can write the SLAM problem as

$$P(X, M|U, Z^*) \propto \underbrace{\prod_k P(x_k|x_{k-1}, u_k)}_{odometry} \underbrace{\prod_{k,i} P(z_{ki}^*|x_k, m_i)}_{measurement} \tag{3}$$

For data association, i.e. to determine a measurement value z_{ki} is an observation of which map element, we use the method introduced in [21] to match poles and walls. To associate measurements of parked vehicles, we can conveniently take tracking IDs in the measurements to find the corresponding object in the object map. However, vehicle tracking ID is not globally consistent. When the tracker re-discovers a vehicle, a new ID will be assigned to it. In this case, we use the center point of vehicles for nearest neighbor matching to obtain the association between the vehicle and the mapped objects. Newly discovered objects are temporarily kept and will be permanently retained on the map once they are confirmed by adequate observations.

Now we can obtain the optimal estimates of poses and map objects by maximizing the posterior probability of (3), assuming the motion and observation noises are independent and follow Gaussian distribution. Substitute (2) to (3), we can get:

$$\begin{aligned} X^*, M^* &= \underset{X,M}{\text{argmax}} P(X, M|U, Z^*) = \underset{X,M}{\text{argmin}} -logP(X, M|U, Z^*) \\ &= \underset{X,M}{\text{argmin}} \sum_{k=1}^K e_{u,k}^T \Sigma_{u,k}^{-1} e_{u,k} + \sum_{k=1}^K \Bigg[\sum_{i \in P} e_{p,ki}^T \Sigma_{p,ki}^{-1} e_{p,ki} \\ &\quad + \sum_{i \in L} e_{l,ki}^T \Sigma_{l,ki}^{-1} e_{l,ki} + \sum_{i \in V} e_{v,ki}^T \Sigma_{v,ki}^{-1} e_{v,ki} \Bigg], \end{aligned} \tag{4}$$

where Σ_u is the covariance of the input noise. We can estimate its value by using a more precise GNSS/INS navigation system as reference. Σ_p, Σ_l, and Σ_v are the covariance of the three types of observations. They are experimentally chosen in our system. From the above equation, a factor graph representation is constructed and solved using Levenberg-Marquardt algorithm iteratively.

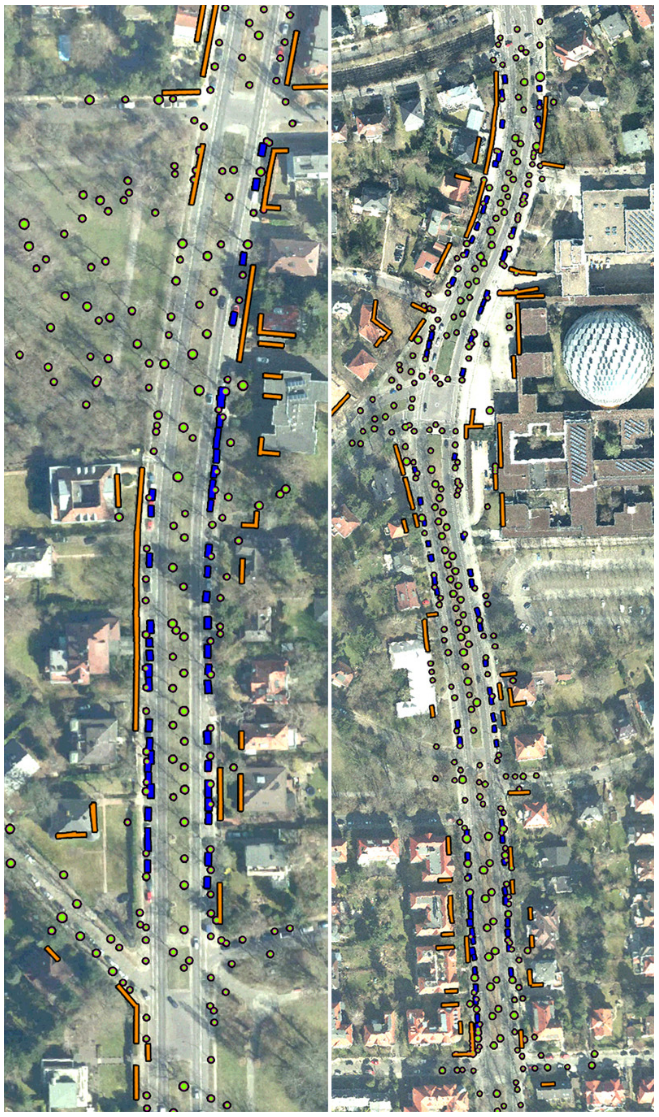

Fig. 4. Generated object map on two test roads, aligned with a satellite map.

C. Map Update

From the measurements of objects in the environment, we obtain the pose-related values to construct the error constraints. Then we add them to the optimization graph to estimate the poses of the vehicle and the map objects. The measured radius of poles, length of wall segments, and width and length of parked vehicles are used to update the scales of the mapped objects. In our object detection algorithm, we estimate an object's size according to the laser points reflected by the object. The accuracy of estimates depends heavily on the number of laser points returned by the object. Due to LiDAR sensors' characteristics, the closer the object is to the sensor, the greater the angle it occupies in the sensor's field of view. Therefore, closer objects reflect more laser points and the scale measurement is more accurate.

We set a distance-dependent *measurement confidence* $mc_i = \exp(-d_i/R)$ for each scale measurement, where d_i is the distance from the sensor's origin to the center of the detected object, and R is the detection range of the sensor. Moreover, measurement consistency is added to the measurement weight to reduce the contribution of noisy measurements. Measurement *consistency level* is computed as $cl_i = \exp(-(s_i-\mu)^2/(2\sigma^2))$, where s_i is the current scale measurement, and μ and σ are the mean and standard deviation of existing measurements of the scale. The weight of a scale measurement is then set as $w_i = mc_i cl_i$. A proportion of the measurements with the highest weights is used to update the map objects' scale value. The weight-based update is not applied to the walls, as the measured length is not the actual length of the wall but usually a fragment of it. The new measurement is merged with the wall in the object map and extends its length.

As the object tracker tracks vehicles and estimates their velocity, the dynamic vehicles can be effectively excluded from the object map.

Fig.4 shows the generated object map aligned with a highly precise satellite map. Green dots are trees and street lamps, orange lines are building walls, and blue rectangles represent parked cars.

D. Loop Detection

We exploit the mapped objects' geographic distribution to implement a three-step loop finding and confirmation method to recognize loops.

Loop frame encoding. First, we choose the loop frame according to the travel interval from the last marked loop frame. Reducing the number of frames involved in searching and comparison can effectively improve computation efficiency. Unlike keyframes, loop frames can be very sparsely chosen. For each loop frame, we generate a ***signature*** for fast candidate selection and a ***local observation*** for loop confirmation.

We extract binary and ternary rotation invariants from the distribution of pole objects around the frame. Set $P = \{p_i\}_{i=1}^{M}$ represents the center points of the detected poles at the current loop frame. We can build two sets:

$$\begin{aligned} D &= \{d(p_i,p_j)) \mid p_i,p_j \in P\ ,\ i \neq j\} \\ T &= \{(d(p_i,p_j),d(p_j,p_k),d(p_i,p_k))|p_i,p_j,p_k \in P, i \neq j \neq k\}, \end{aligned} \tag{5}$$

where $d(.)$ is the distance function. D and T respectively represent all the distances between two detected poles and all the triangles formed by each three of them. We sort the elements in these two sets and the three variables in each element in set T in ascending order. The two ordered sets D' and T' are assigned to the current loop frame as its loop signature.

To avoid false-positive confirmation as much as possible, we collect the detections of a loop frame and its k previous frames. The collected pole positions are transformed into the current loop frame's local coordinate system and merged if a pole is detected multiple times. It is a small-scale local map around the current loop frame. If the size of this map is larger than a threshold o, it can be assigned to the current loop frame as its local observation. Value o needs to be adapted according to the density of poles in the driving environment. For the residential area of the city where our research institute is located, trees and street lights can be commonly found near the road. The threshold is set to 30 for the experiments described in Section IV.

Candidate loop frame finding. Before confirming a loop, we first compare the signature of the loop frame to find the candidate loop frames that are most likely to coincide with the current loop frame. We can match the elements between $D_1^{'}$ and $D_2^{'}$ of two frames if a pair (d_i, d_j) satisfies $\|d_i - d_j\| \leq \varepsilon$, where $d_i \in D_1^{'}, d_j \in D_2^{'}$. The number of matched elements is denoted as n_D. Similarly, the elements between sets $T_1^{'}$ and $T_2^{'}$ can be matched if a pair of elements (t_i, t_j) meet the condition $\|t_{i,(1:3)} - t_{j,(1:3)}\| \leq \xi$, where $t_i \in T_1^{'},\ t_j \in T_2^{'}$. The number of matches is denoted as n_T.

The similarity S between two loop frames is calculated as follows:

$$\begin{aligned} S_D &= \frac{2n_D}{n(D_1^{'}) + n(D_2^{'})},\ S_T = \frac{2n_T}{n(T_1^{'}) + n(T_2^{'})} \\ S &= \beta S_D + (1-\beta) S_T, \end{aligned} \tag{6}$$

where parameter β weights the two type of invariants. The value S_T is more discriminative, so it was given a higher weight in the experiment. The loop frame(s) with highest similarity values are chosen as the candidate loop frame(s).

Loop confirmation. In the field of computer vision, researchers have explored the method and application of matching two sets of rotated, scaled, and translated points [23], [24]. These methods can be applied in fingerprint matching and face recognition, etc. We implemented a simplified algorithm introduced in [21] to match the observations of the current loop frame and the candidate frames. If the points to be matched are sufficient and the spatial distribution of the poles does not have a large proportion of repetitions, such matching can be considered reliable. In practice, we can increase the number of frames that constitute a local observation to expand its size. In our experiments, we set the minimum matching rate to 60% and minimum matching number to 30 to avoid false matching.

IV. Experiments

We collected real-world data using a test vehicle. The data involved in the experiment mainly came from a Velodyne HDL-64E LiDAR, an Applanix POS-LV 510 navigation system, and the Controller Area Network bus. The LiDAR sensor is mounted on the roof of the car. In order to test the accuracy and reliability of our algorithm in different road conditions, we collected four datasets of different lengths and trajectory shapes. The lengths ranged from around 300 meters to 4.8 kilometers. The names and the trajectory lengths of the datasets are listed in Table I.

We first evaluated the proposed loop detection method on Test Road 3. In Fig. 5, we mark each historical loop frame with a different color according to its similarity with the current loop frame. The interval between two loop frames is at least three meters. As can be seen from the figure, most areas of the trajectory are displayed in blue. According to

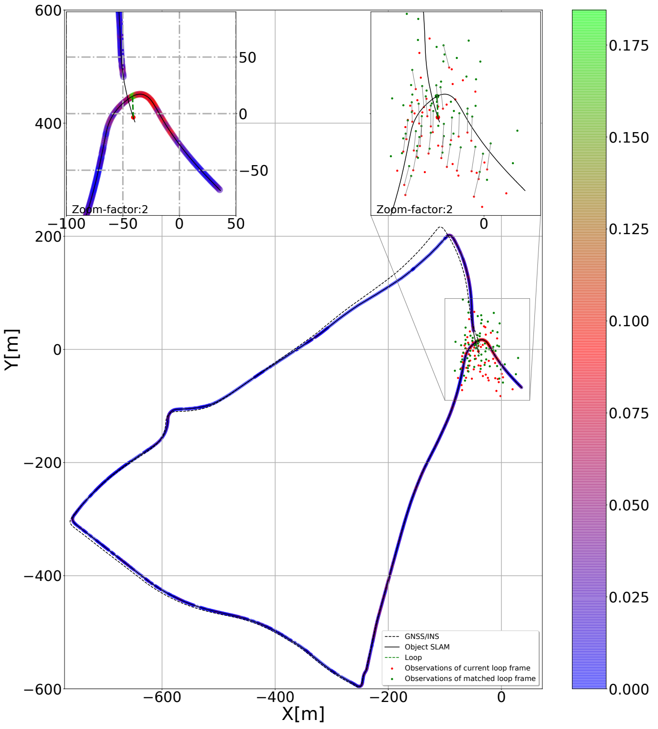

Fig. 5. Loop frames are encoded with different colors according to their similarity with the current loop frame. The true loop is indicated with a dashed green line. The zoomed-in area on the upper right corner shows the matching result of the observations of the current loop frame and the best candidate.

the color coding, this means most loop frames have very low similarity with the current loop frame. Only in the vicinity of the true loop area, as shown in the zoomed-in area in the upper left corner, is the similarity value significantly larger. The green dashed line connects the current loop frame and the best matched historical loop frame. The zoomed-in area in the upper right corner shows the *local observations* of the current loop frame and the best candidate loop frame. They are collected from a loop frame and its 5 previous frames, transformed into the loop frame's local coordinate. The point pattern matching algorithm found a transformation between the two sets of points, which corresponded more than 60% of current observations with the observations of the candidate loop frame. Thus the loop is confirmed. The matched observations are connected with grey lines shown in the upper right area.

To verify the accuracy of poses estimated by our system, we compared our proposed method with A-LOAM, which is an advanced implementation of LOAM [11], ICP Mapping [25], LIO-SAM [26], which fuses LiDAR and inertial sensors. As ground truth, we took the trajectories reported by our GNSS navigation system. Integrated with wheel odometry, our method estimate poses at a frequency of 100 Hz. The GNSS/INS reports the vehicle poses at 200 Hz, and other methods process the LiDAR data at 10 Hz. Note that LIO-SAM uses keyframe technique to reduce the computation; therefore, the poses on its trajectories are sparser. We took the IMU data from the GNSS/INS system and raw LiDAR point cloud as inputs for LIO-SAM, and de-skewed point cloud for all methods. We show the trajectories reported by all methods on our test datasets in Fig. 6 to Fig. 9.

Fig. 6 shows the results of all methods on a short and straight road with no loops. Taking the GNSS/INS result as the reference, the end-to-end translation errors for the proposed method, ICP Mapping, A-LOAM, and LIO-SAM are 0.67, 1.59, 3.14, and 11.80 meters, respectively. They are respectively 0.23%, 0.54%, 1.06%, and 4.01% of the total trajectory length. The output trajectories of various methods on Road 3 are shown in Fig. 7. Since the detection range of the LiDAR covered both sides of the road, when the vehicle drove back through the u-turn, the new data could be matched with the already-generated map. Therefore, all trajectories could return to positions near the starting point. Nevertheless, we can still observe that the path of A-LOAM shows a rotation compared to the ground truth. LIO-SAM reported a path longer than other methods. Furthermore, its trajectory lacked smoothness in the turning area due to the selection of keyframes.

Fig. 4 shows the maps generated on Test Road 2 and another road at different scales. The green circles represent the poles, and their size indicates their radius but is enlarged for better visibility. A closer look will show that the maps are globally consistent: the generated walls conform to the buildings' outline, vehicles are in the parking area, and the poles coincide with the footpoints of the trees and street lamps all along with the maps.

Road 3 and Road 4 have longer trajectory lengths, and they were more challenging compared to the other two test roads. In Fig. 8 we can see that the ICP methods revealed a large drift. The shapes of trajectories generated by other methods are very similar to the reference. Road 4 is a rectangular residential area with a length of 4.8 kilometers. Our proposed system showed a significant advantage on this test road. Compared with other methods, its trajectory shows high consistency with the ground truth. The end translation errors of end points were 3.58, 50.64, 32.49, and 53.79 meters for the proposed method, ICP Mapping, A-LOAM, and LIO-SAM, respectively. We noticed that A-LOAM's end point error was smaller than ICP Mapping and LIO-SAM, but its trajectory had the largest distance from the ground truth. Similar to the result on Road 2, A-LOAM's trajectory also showed a rotation on this test road.

To quantitatively evaluate and compare various methods, we chose three metrics: the Absolute Pose Error (APE), which directly calculates the difference between the estimates and the ground truth; the Relative Pose Error (RPE), which evaluates the local accuracy of the trajectory, i.e. the local displacement in 1 meter compared to the reference [27]; and aligned Absolute Trajectory Error (ATE) [28], which is suitable for evaluating the similarity of trajectory shapes. Table II lists the results of our method, ICP Mapping, A-LOAM, and LIO-SAM on the test datasets. For those methods whose results are expressed in three-dimension, we directly project their results into two-dimension for comparison. The statistics show that our method outperforms others in almost all comparisons. The only exception is on Road 2; after the trajectory alignment, both ICP Mapping and A-LOAM showed smaller errors than our method. This meant that the shapes of their trajectories were more similar to the ground truth. Note that after aligning the results on Test Road 4, A-LOAM gave a better ATE value than ICP Mapping and LIO-SAM, which is explained by its smaller endpoint error.

Road	Trajectory Length [m]	Loop
Test Road 1	294.71	No
Test Road 2	1188.71	Yes
Test Road 3	2455.78	Yes
Test Road 4	4793.36	Yes

TABLE I
DETAILS OF THE DATASETS.

Road	Method	RPE [m]	APE [m]	ATE(aligned) [m]
Road 1	Proposed	**0.04**	**0.36**	**0.18**
	ICP	0.10	1.55	0.35
	A-LOAM	0.11	3.06	0.78
	LIO-SAM	0.15	8.35	4.48
Road 2	Proposed	**0.04**	**0.71**	0.57
	ICP	0.09	**0.71**	**0.38**
	A-LOAM	0.09	5.76	**0.38**
	LIO-SAM	0.16	7.51	7.43
Road 3	Proposed	**0.04**	**2.57**	**1.57**
	ICP	1.18	69.20	64.01
	A-LOAM	0.15	10.42	4.13
	LIO-SAM	0.18	10.76	5.78
Road 4	Proposed	**0.04**	**3.77**	**1.47**
	ICP	0.07	40.53	14.91
	A-LOAM	0.09	98.45	7.93
	LIO-SAM	0.13	34.36	12.74

TABLE II
RESULTS OF PROPOSED METHOD, ICP MAPPING, A-LOAM AND LIO-SAM ON THE TEST DATASETS. TRAJECTORIES ARE SHOWN IN FIG. 6, 7, 8, AND 9

In our test, LIO-SAM showed a relatively large error. As stated by the authors, its performance relies heavily on the accuracy of IMU data. Also, the keyframe-based method dropped LiDAR data information while improving computation efficiency, which produced unsmooth trajectory in the turning area.

Our proposed method chose poles, walls, and parked vehicles for localization and mapping. These objects can be detected in a large number and stably in the test environment. The tracking of these objects naturally helped the algorithm to correctly associate measurements with the map and provided constraints for pose optimization in multiple consecutive frames. Compared with the A-LOAM and ICP methods, we also integrated wheel odometry as extra information. It is integrated into the system in a tightly coupled manner to eliminate motion distortion, assist object tracking and participate in state estimation. Therefore, the proposed system presented high accuracy on our test dataset.

V. CONCLUSION

We proposed an object-level SLAM method based on 3D LiDARs for the application of autonomous vehicles.

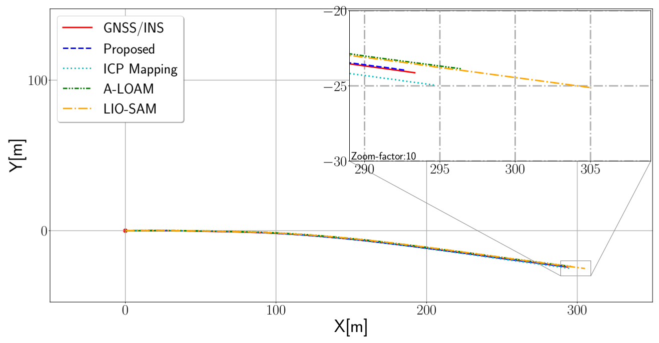

Fig. 6. Trajectories of various methods on a Test Road 1, a short and straight road.

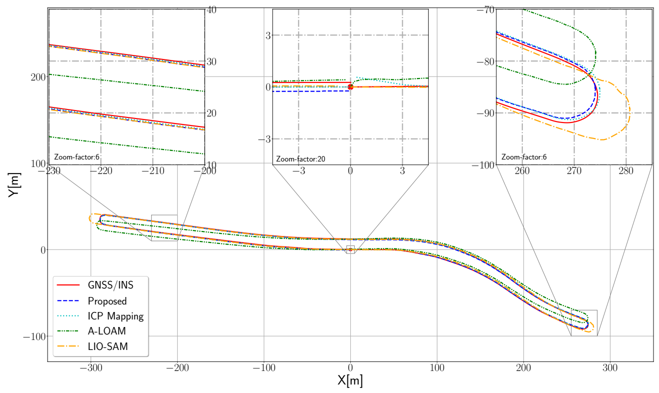

Fig. 7. Trajectory plot on Test Road 2. All methods returned to the starting position.

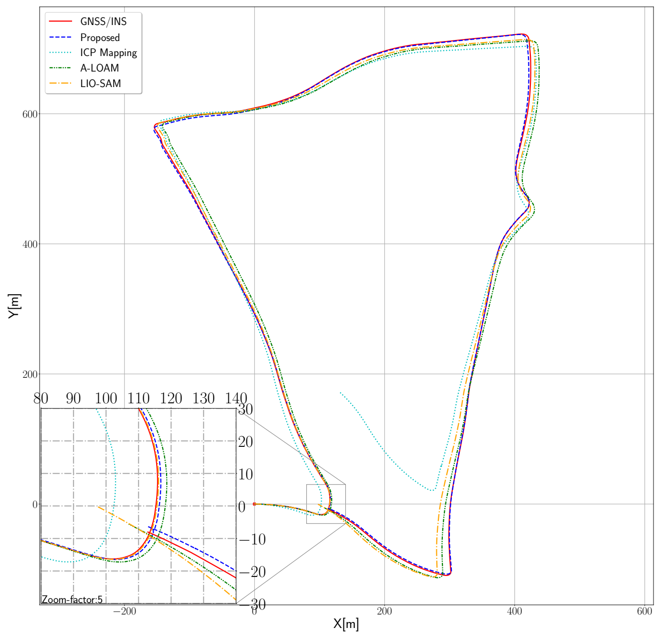

Fig. 8. Results of all methods on Test Road 3.

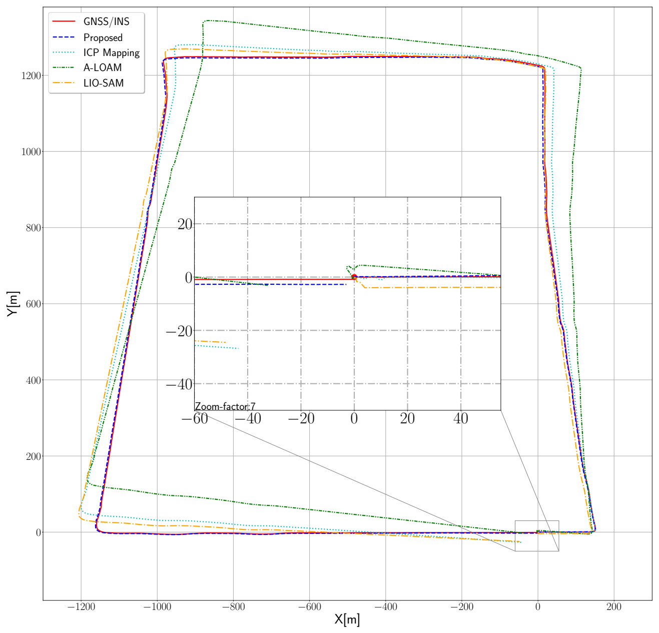

Fig. 9. Results of all methods on Test Road 4.

This method can detect and track poles, walls, and roadside parked vehicles that are common in urban environments. We extract pose-related information from the measurements of the objects, use a graph-based optimization method to estimate the poses of the ego car and map objects, and extract the scale-related information to update the size of the objects in the map. A loop detection method based on the geographical distribution of surrounding poles was introduced. The proposed method was tested on four datasets of different trajectory lengths and compared with other methods. The results showed that our method outperformed other methods in terms of absolute and relative pose accuracy. Besides, the proposed method can generate globally consistent maps with semantic meaning. For future work, we can increase the types of objects involved in the loop detection model to improve its robustness. In addition, other types of objects can be detected and added to the system to enrich the information of the generated map.

References

[1] 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*, vol. 32, no. 6, pp. 1309–1332, Dec 2016.

[2] L. Nicholson, M. Milford, and N. Sünderhauf, “Quadricslam: Constrained dual quadrics from object detections as landmarks in semantic SLAM,” *CoRR*, vol. abs/1804.04011, 2018. [Online]. Available: http://arxiv.org/abs/1804.04011

[3] Z. Qian, K. Patath, J. Fu, and J. Xiao, “Semantic slam with autonomous object-level data association,” 2020.

[4] Y. Nakajima, K. Tateno, F. Tombari, and H. Saito, “Fast and accurate semantic mapping through geometric-based incremental segmentation,” *CoRR*, vol. abs/1803.02784, 2018. [Online]. Available: http://arxiv.org/abs/1803.02784

[5] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, “Suma++: Efficient lidar-based semantic slam,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.

[6] S. W. Chen, G. V. Nardari, E. S. Lee, C. Qu, X. Liu, R. A. F. Romero, and V. Kumar, "SLOAM: semantic lidar odometry and mapping for forest inventory," *CoRR*, vol. abs/1912.12726, 2019. [Online]. Available: http://arxiv.org/abs/1912.12726

[7] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[8] G. Silveira, E. Malis, and P. Rives, "An efficient direct approach to visual slam," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 969–979, 2008.

[9] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[10] 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.

[11] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.

[12] C. Brenner, "Vehicle localization using landmarks obtained by a lidar mobile mapping system," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences:[PCV 2010-Photogrammetric Computer Vision And Image Analysis, Pt I] 38 (2010), Nr. Part 3A*, vol. 38, no. Part 3A, pp. 139–144, 2010.

[13] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid, "Meaningful maps with object-oriented semantic mapping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5079–5085.

[14] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," in *2017 IEEE International Conference on Robotics and automation (ICRA)*. IEEE, 2017, pp. 4628–4635.

[15] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Robust semantic mapping in challenging environments," *Robotica*, vol. 38, no. 2, pp. 256–270, 2020.

[16] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[17] K. Ok, K. Liu, K. Frey, J. P. How, and N. Roy, "Robust object-based slam for high-speed autonomous navigation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 669–675.

[18] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[19] 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," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 339–355, 2020.

[20] S. W. Chen, G. V. Nardari, E. S. Lee, C. Qu, X. Liu, R. A. F. Romero, and V. Kumar, "Sloam: Semantic lidar odometry and mapping for forest inventory," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 612–619, 2020.

[21] B. Cao, C.-N. Ritter, D. Göhring, and R. Rojas, "Accurate localization of autonomous vehicles based on pattern matching and graph-based optimization in urban environments," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[22] G. Toussaint, "The rotating calipers: An efficient, multipurpose, computational tool," 2014.

[23] S.-H. Chang, F.-H. Cheng, W.-H. Hsu, and G.-Z. Wu, "Fast algorithm for point pattern matching: invariant to translations, rotations and scale changes," *Pattern recognition*, vol. 30, no. 2, pp. 311–320, 1997.

[24] A. C. Berg, T. L. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 26–33.

[25] 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, Feb. 2013.

[26] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5135–5142.

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

[28] J. Sturm, W. Burgard, and D. Cremers, "Evaluating egomotion and structure-from-motion approaches using the tum rgb-d benchmark," in *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, 2012.