Robotica (2024), **42**, pp. 891–910
doi:10.1017/S0263574723001868

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Semantic geometric fusion multi-object tracking and lidar odometry in dynamic environment

Tingchen Ma[1,2], Guolai Jiang[1], Yongsheng Ou[3] and Sheng Xu[4]

[1]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China, [2]Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen, 518055, China, [3]Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian, 116024, China, and [4]Guangdong Provincial Key Laboratory of Robotics and Intelligent System, Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, 518055, China
Corresponding authors: Yongsheng Ou, Sheng Xu; Emails: yoo2023@dlut.edu.cn, sheng.xu@siat.ac.cn

Received: 14 March 2023; **Revised:** 1 November 2023; **Accepted:** 13 December 2023;
First published online: 11 January 2024

Keywords: mobile robots; navigation; lidar SLAM; multi-object tracking; dynamic object detection

Abstract

Simultaneous localization and mapping systems based on rigid scene assumptions cannot achieve reliable positioning and mapping in a complex environment with many moving objects. To solve this problem, this paper proposes a novel dynamic multi-object lidar odometry (MLO) system based on semantic object recognition technology. The proposed system enables the reliable localization of robots and semantic objects and the generation of long-term static maps in complex dynamic scenes. For ego-motion estimation, the proposed system extracts environmental features that take into account both semantic and geometric consistency constraints. Then, the filtered features can be robust to the semantic movable and unknown dynamic objects. In addition, we propose a new least-squares estimator that uses geometric object points and semantic box planes to realize the multi-object tracking (SGF-MOT) task robustly and precisely. In the mapping module, we implement dynamic semantic object detection using the absolute trajectory tracking list. By using static semantic objects and environmental features, the system eliminates accumulated localization errors and produces a purely static map. Experiments on the public KITTI dataset show that the proposed MLO system provides more accurate and robust object tracking performance and better real-time localization accuracy in complex scenes compared to existing technologies.

1. Introduction

In recent years, artificial intelligence technologies such as service robots, autonomous driving, and AR/VR have rapidly developed and demonstrated broad applications. A key commonality among these applications is the need for high-precision modeling of scenes, including semantic objects such as cars and people. To achieve the above functions, a fundamental and important technology – simultaneous localization and mapping (SLAM) – is indispensable. On the other hand, most SLAM [1–4] systems based on the assumption of rigid scenes are not applicable in dynamic scenes with moving objects. Firstly, during the localization and mapping process, the framework designed for static scenes will introduce disturbances to the estimation problem due to the emergence of dynamic objects. The system may even crash as a result. Secondly, the classic SLAM system can only construct 3D maps with static backgrounds. They cannot provide the foreground semantic object information required for upper-level planning and control tasks. The robots may appear clumsy and unreliable in practical work. Therefore, it is necessary to focus on dynamic SLAM technology to improve the adaptability of navigation systems in real scenarios.

Currently, some studies have used geometric constraints or prior semantic methods to process dynamic objects within the perception range of the sensor. Methods with geometric constraints

[5–8] use spatial clustering, motion segmentation, background model, or reprojection error checking to separate abnormal data associations before state estimation. For learning-based methods, refs. [9, 10] and [11] directly identify moving objects in a single scan or image through the trained network model. Refs. [12] and [13] use a semantic segmentation network [14] to obtain point-level semantic labels in the lidar scan and realize 3D map construction with scene semantics (roads, buildings). Undoubtedly, compared to schemes that check abnormal geometric constraints, SLAM systems which correctly introduce and use semantic information have more potential for application. Firstly, the semantic and geometric information can both be considered to ensure that invalid measurements are recognized. Meanwhile, the semantic map constructed by SLAM systems can provide richer decision-making information for the navigation system.

On the other hand, although the methods mentioned above can improve the robustness of the system in dynamic scenes, semantic SLAM systems such as refs. [12, 13] focus more on extracting and processing background semantic information. They cannot stably track and represent dynamic objects in the map coordinate, which is also necessary for many actual robot systems. In this work, we will solve this problem by using a lidar sensor that provides highly accurate point cloud data. Noting that moving objects are often only observed from a local perspective, the geometric point cloud constraints may sometimes fail to estimate the 6-DOF object motion. We further consider using the information obtained from the perception module to balance the accuracy and robustness of object tracking.

Based on the above analysis, a least-square estimator is proposed that takes into account both semantic and geometric information to perform reliable multi-object tracking (SGF-MOT). Combined with the estimation results of ego odometry, the absolute localization for each object can be calculated. The mapping module uses the absolute trajectory tracking list for dynamic semantic object detection. Then, the accumulated error of ego and object odometry is eliminated. Finally, we realize the construction of a 4D scene map. Figure 1 shows the scene map created by the MLO system. The main contributions of this paper are as follows:

- A complete multi-object lidar odometry system (MLO) can provide the absolute localization of both robot and semantic objects and build a 4D scene map (the robot, semantic object localization, and long-term static map).
- A least-square estimator considering the bounding box planes and the geometric point cloud is used for object state updating in the multi-object tracking (MOT) module. The semantic bounding boxes ensure the correct optimization direction. The relative motion model eliminates the point cloud distortion caused by robot and object motion. Using point clouds with direct measurement further improves the estimation accuracy.
- A dynamic object detection method based on the absolute trajectory tracking list that can identify slow-moving semantic objects.

The rest of this paper is structured as follows. First, in Section 2, we discuss related works. Then, the proposed MLO system is described in Section 3. The experimental setup, results, and discussion will be presented in Section 4. Finally, Section 5 is the conclusion.

2. Related works

2.1. MOT

For the MOT method based on point cloud data, object recognition and tracking are the two main steps of most solutions. In the beginning, traditional methods focused on object extraction from spatial distribution information of point clouds. Then, object tracking is realized by constructing geometric error constraints and a corresponding estimator. Moosmann *et al.* [6] proposed a local convexity criterion for geometric object detection. Then, the MOT task was implemented by the point-to-plane ICP [15] algorithm. Dewan *et al.* [16] used the RANSAC [17] algorithm to compute the motion model sequentially. Then, a Bayesian approach was proposed for matching the object point cloud with the corresponding motion model. Ferri *et al.* [18] proposed a rigid scene flow algorithm for tracking non-rigid objects that

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

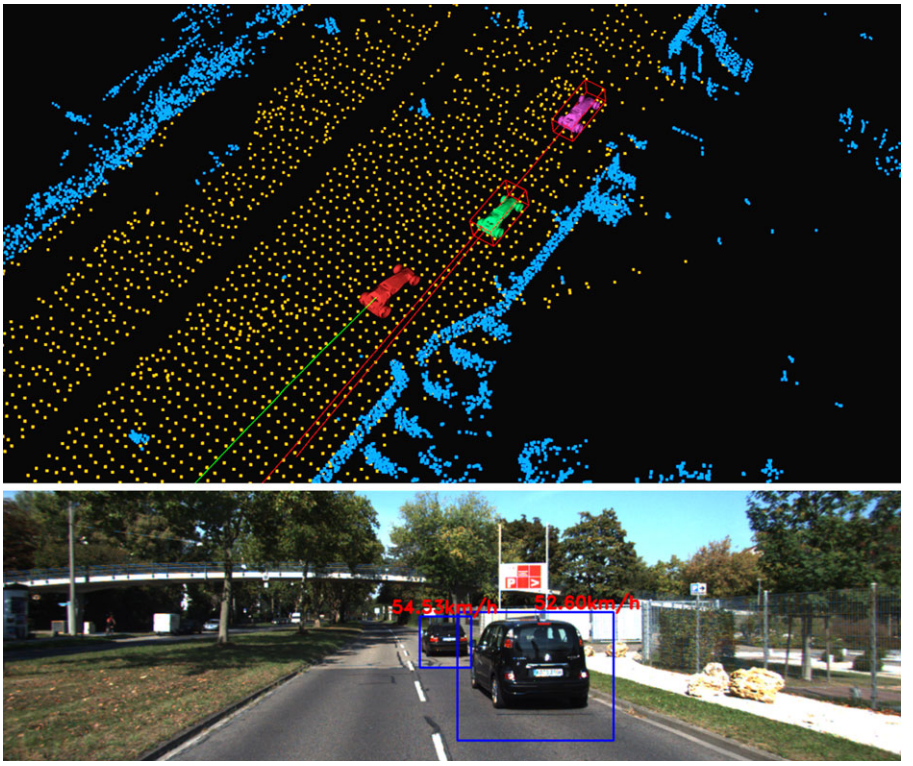

Figure 1. The proposed MLO system testing on sequence 0926-0013 in the KITTI-raw dataset. *The upper part of the picture shows the point cloud map, tracking trajectories of the robot and semantic objects generated by the MLO system. The lower part of the picture (for visualization only) gives the semantic bounding box and absolute motion speed of each object successfully tracked in the synchronized image.*

can also handle the pedestrian. Sualeh *et al.* [19] used a cluster point cloud for fitting 3D bounding boxes (3D-BB). The boxes were tracked by the Kalman filter [20] algorithm.

Recently, the learning-based recognition method on point clouds has brought new solutions to the MOT problem. They can provide richer and more stable recognition information, such as the bounding box with object category and motion direction. Weng *et al.* [21] detected semantic objects by PointRCNN [22] network model and tracked the 3D-BB with a Kalman filter. Kim *et al.* [23] used 2D/3D-BB detected in image and point clouds for filter tracking, achieving better tracking accuracy. Wang *et al.* [24] and Huang *et al.* [25] designed an end-to-end MOT framework using different deep-learning models to complete object detection and data association tasks.

In summary, optimization-based tracking methods that rely on the geometric constraints of point clouds require a reliable initial guess. If the estimator's initial value is poor, many point-level data associations may provide the wrong direction at the early stage of the calculation. Extracting bounding boxes from the point cloud (directly fitting or using a network model) and tracking them by the filter-based methods [26] is more stable but sacrifices high-precision measured point clouds. In our SGF-MOT module, we use semantic bounding box planes and geometric point cloud distributions simultaneously in a least-squares estimator, achieving a better balance between tracking accuracy and robustness.

2.2. Dynamic aware 3D lidar SLAM

At present, 3D lidar SLAM systems are mainly divided into two categories: feature-based and matching-based. Zhang *et al.* [1] and Shan *et al.* [2] extracted stable environment structure features (edge and

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

surface) based on scan line roughness for reliable robot localization and lidar mapping. Meanwhile, Shan *et al.* [2] also considered ground constraints to improve the accuracy and efficiency of estimation. Liu *et al.* [27] proposed a novel lidar bundle adjustment (BA) [28] method, which depends only on the robot poses. The feature landmarks and robot poses can be optimized for high-precision lidar mapping by minimizing the eigenvalue. In the matching-based method, Behley *et al.* [4] used surfel to model the environment and designed a projection data association method for ego localization. Dellenbach *et al.* [29] proposed a continuous-time ICP algorithm for lidar odometry estimation. Their method considered both the constraint of inter-frame discontinuity and the correction of in-scan motion distortion. The above systems rely on the assumption of the rigid scene and cannot work stably in a highly dynamic scene.

To solve the above problems, some studies proposed geometric methods such as background model [7], local convexity [6], and point cloud clustering [5] to detect and eliminate unreliable information. Meanwhile, with the development of deep-learning technology, some semantic-based lidar SLAM can also process dynamic objects within the field of view. Pfreundschuh *et al.* [10] designed a method to automatically label dynamic object points when the 3D-MinNet detection model was trained. The static point cloud after detection will be input into the LOAM [1] framework to enable reliable mapping. Sualeh *et al.* [30] further considered the space-time constraints of sequence frames when training the dynamic object detection model. Both refs. [12] and [13] using semantic segmentation networks realize semantic scene mapping. In addition, they detect dynamic objects through point-by-point checks from the object point cloud.

In contrast to these approaches, our proposed system, MLO, integrates the results of robust ego localization and the SGF-MOT module to achieve accurate and reliable localization of instance objects in the map coordinate. Additionally, our system uses an absolute trajectory tracking list to detect dynamic objects in the scene, creating a 4D scene map that includes both robot and semantic object positions as well as the static environment.

2.3. Robust SLAM with multi-sensor fusion

To overcome the shortcomings of a single sensor, some researchers are considering combining two or three types of sensors to improve the adaptability of SLAM systems in complex scenarios. Unlike lidar, the camera can provide images with rich color information. Thus, visual SLAM has also received widespread attention. According to the different front-end methods, visual SLAM can mainly be divided into feature-based methods [31, 32], semi-direct methods [33], or direct methods [34, 35]. Among them, the feature-based method is more robust to illumination and rotational motion, while the direct method has more application potential because it can construct dense or semi-dense maps. In the field of visual-lidar fusion SLAM, ref. [36] used a monocular camera with higher frequencies for initial pose estimation, providing more accurate initial values for lidar odometry and constructing point cloud maps. Meanwhile, the author also used point cloud information to solve the scale uncertainty problem in monocular visual odometry. In refs. [37, 38], depth information measured by lidar was integrated into the feature-based [3] or direct [39] visual SLAM framework for 3D pixel or feature landmark initialization. The system further improves the local mapping accuracy in the back-end by the BA algorithm.

The Inertial Measurement Unit (IMU), as an internal high-frequency sensor that is not affected by the external environment, has significant advantages in dealing with problems such as intense motion and complex scenes. In refs. [1, 40], the IMU measurement was used to eliminate the motion distortion of lidar point clouds and provided reasonable initial values for the lidar odometry. In addition, ref. [40] further used GPS to reduce the accumulative error generated by lidar odometry. Refs. [41, 42] tracked feature points extracted from images and combined IMU pre-integration technology to construct the tightly coupled visual-inertial SLAM system. They can achieve high-precision sparse feature mapping through the BA algorithm. Recently, ref. [43] proposed a visual-inertial, lidar-inertial odometry fusion

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

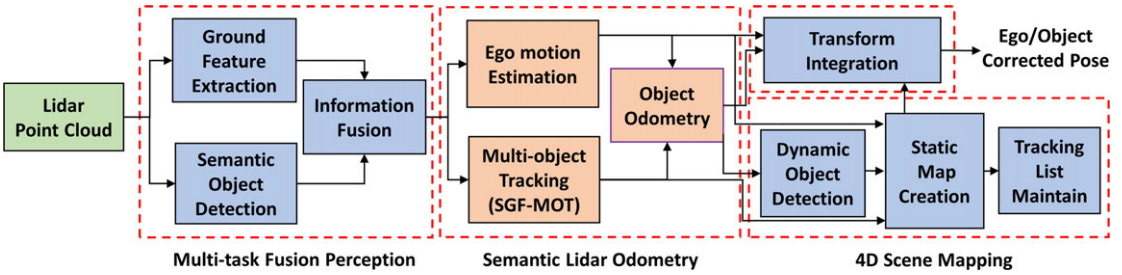

Figure 2. The proposed MLO system framework. *The orange box is the key part of this paper; it includes the semantic-geometric fusion multi-object tracking method (SGF-MOT), which will be discussed in detail. A 4D scene mapping module is also introduced for maintaining the long-term static map and movable object map separately, which can be used to further navigation applications.*

system, which can effectively work when either odometry fails and cope with textureless scenes such as white walls.

Undoubtedly, by combining multiple sensors, the fusion system can utilize richer measurement information to enhance the robustness of robots working in complex motion states or low-texture scenes. However, when the fusion system operates without targeted images or point cloud data processing in a dynamic environment, dynamic measurement information can still lead to significant estimation errors in the localization and mapping processes. In this paper, we compare the proposed semantic lidar odometry (MLO) with other advanced multi-sensor fusion SLAM frameworks to fully analyze the advantages and disadvantages of the MLO system in complex dynamic scenarios.

3. Method

The MLO system proposed in this paper can be divided into three modules: multi-task fusion perception, semantic lidar odometry, and 4D scene mapping. Figure 2 shows the flowchart of the MLO system. Firstly, the ground and objects are detected and refined by information fusion. Then, the proposed SGF-MOT tracker is integrated into the odometry module to achieve robust and accurate object tracking. After dynamic object detection, the mapping module uses both static objects and environment feature sets [1] to eliminate accumulated errors.

3.1. Notation definition

The notation and coordinates used in this paper are defined as follows: In the odometry coordinate o and frame i, the robot pose is $\mathbf{T}_{ol}^{i} \in \mathrm{SE}(3)$. Meanwhile, the observed semantic object has pose $\mathbf{T}_{ot}^{i} \in \mathrm{SE}(3)$. In the mapping module, the corrected robot pose in the map coordinate m is $\mathbf{T}_{ml}^{i} \in \mathrm{SE}(3)$, and the correction matrix which transforms the object from the odometry coordinate o to the map coordinate m is $\mathbf{T}_{mo}^{i} \in \mathrm{SE}(3)$.

3.2. Multi-task fusion perception

For point cloud $\mathbf{\Phi}^{i}$ in frame i, we run the parallel object detection algorithm and ground feature extraction algorithm to obtain the original ground and object point cloud. Then, we detect and eliminate the intersection point in the ground feature set and foreground object by the marked labels and perform background feature extraction. Figure 3 shows the detection results of the multi-task fusion perception module.

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

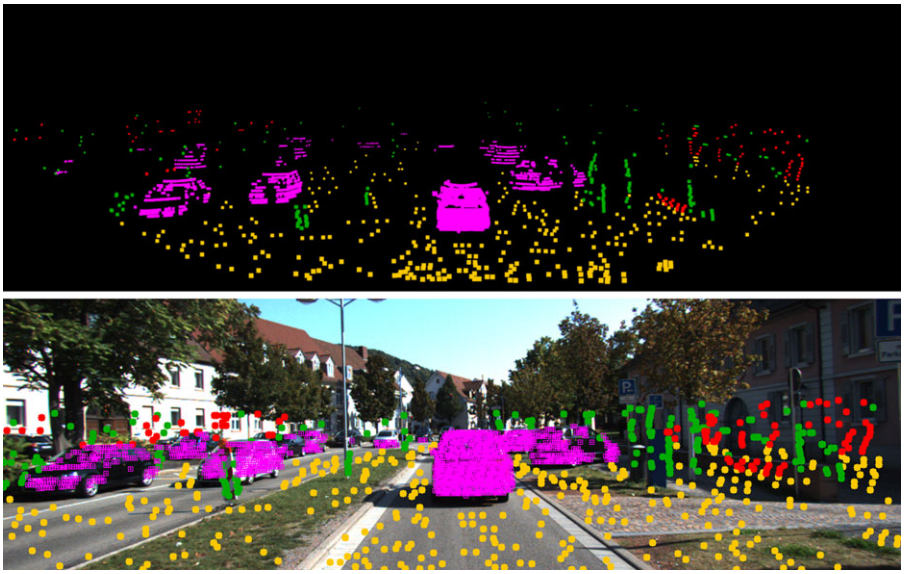

Figure 3. The multi-task fusion perception module testing on sequence 0926-0056 in the KITTI-raw dataset. *The semantic object point clouds are composed of pink points. Yellow points form the ground feature set. Green and red points form the background edge and surface feature sets, respectively. The image projection results with the same timestamp are used for visualization only.*

3.2.1. Ground feature extraction

First, we adjust ground parameters in the point cloud using the iterative PCA algorithm proposed in ref. [44] and mark the corresponding points near the ground in the point cloud $\mathbf{\Phi}_g^i$ (**1:** ground point, **0:** background point).

Assume t^{i-1} and t^i are the beginning and end time of current frame i respectively. Meanwhile, the sampling time of the k-th raw point ${}^l\mathbf{p}_{i,k}$ in the lidar coordinate l is t_k, and the relative timestamp ${}^r t_k$ used to correct motion distortion can be calculated as follows:

$$ {}^r t_k = \frac{t_k - t^i}{t^{i-1} - t^i} \tag{1} $$

Then, the current point cloud is uniformly segmented according to the scan line. Through the point-by-point verification of ground markings in the point cloud $\mathbf{\Phi}_g^i$, it is possible to record whether each segment contains the ground point or background point labels. For segments containing ground or background points, the segment head and tail point ID pairs are stored in the ground container $\mathbf{V}_g$ or background container $\mathbf{V}_b$, respectively.

Based on the calculated roughness (the calculation formula can be found in ref. [1]) and the segmented point ID pairs contained in the ground container $\mathbf{V}_g$, ground feature set $\boldsymbol{\chi}_f^i$ can be extracted from point cloud segments containing ground points. The roughness prior R_a used for subsequent inspection of background surface feature points is obtained as follows:

$$ R_a = \sum_{o=1}^{N} c_o \Big/ N_c \tag{2} $$

where c_o is the o-th ground point roughness. $\boldsymbol{\chi}_c^i$ represents the candidate ground feature set obtained by downsampling, and N_c is the corresponding number of ground points.

For non-ground point cloud segments that are searched by ID pairs in the background container $\mathbf{V}_b$, only point roughness is calculated and sorted here. After that, the information fusion module will neatly select the background feature sets.

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

3.2.2. Semantic object detection

To identify the semantic objects and their corresponding 3D-BB in the point cloud, we use 3DSSD [45], a point-based single-stage object detection model, to perform these tasks. Assuming that k-th 3D-BB in frame i is:

$$\boldsymbol{\tau}^{k,i} = \left[x^{k,i}, y^{k,i}, z^{k,i}, l^{k,i}, w^{k,i}, h^{k,i}, yaw^{k,i}\right]^T \tag{3}$$

where the object position in the lidar coordinate l is (x, y, z). The length, width, and height of 3D-BB are (l, w, h), and *yaw* is the yaw angle for the object in the lidar coordinate l. Assuming $\boldsymbol{\gamma}^{k,i}$ denotes the object point cloud enclosed by 3D-BB and each 3D point is marked with an independent object ID in $\boldsymbol{\Phi}_s^i$. Object semantic confidence is represented as $\mu^{k,i}$.

3.2.3. Information fusion

The 3D-BB, obtained by the semantic detection module, usually encloses both object and ground points. Meanwhile, the ground feature set computed by Section 3.2.1 may also have points belonging to objects. If the object is in motion, the static ground points in the object point cloud and the moving object points in the ground feature set will lead to errors in the pose estimation of the object and the robot, respectively. Therefore, we need to eliminate ambiguous intersection information through a mutual correction step.

Specifically, the object points in the ground feature sets $\boldsymbol{\chi}_f^i$ and $\boldsymbol{\chi}_c^i$ are deleted by the marked object ID in $\boldsymbol{\Phi}_s^i$. The ground point in object point cloud $\boldsymbol{\gamma}^{k,i}$ is removed through the label in $\boldsymbol{\Phi}_g^i$. This ambiguous information is not considered in the subsequent SLAM and MOT tasks.

Based on the ID pairs contained in $\mathbf{V}_b$ and computed point roughness, we perform two feature extractions: The candidate object edge feature set $\boldsymbol{\gamma}_c^{k,i}$ is extracted from the point clouds belonging to objects, which is used in the mapping module to eliminate accumulated errors. Subsequently, the background edge feature set $\boldsymbol{\psi}_f^i$, surface feature set $\boldsymbol{\zeta}_f^i$, and the corresponding candidate sets $\boldsymbol{\psi}_c^i$, $\boldsymbol{\zeta}_c^i$ are extracted from the non-semantic and non-ground point clouds, and they will participate in both localization and mapping modules.

Finally, we define the point-to-line error term $\mathbf{e}_{e_k}$ and the point-to-surface error term $\mathbf{e}_{s_k}$ in the lidar coordinate l, which are used for ego pose estimation in Section 3.3.1:

$$\mathbf{e}_{e_k} = \frac{\left|({}^l\mathbf{p}_{i,k}^e - {}^l\bar{\mathbf{p}}_{i-1,m}^e) \times ({}^l\mathbf{p}_{i,k}^e - {}^l\bar{\mathbf{p}}_{i-1,n}^e)\right|}{\left|{}^l\bar{\mathbf{p}}_{i-1,m}^e - {}^l\bar{\mathbf{p}}_{i-1,n}^e\right|} \tag{4}$$

$$\mathbf{e}_{s_k} = \frac{\left|\begin{array}{c}\left({}^l\mathbf{p}_{i,k}^s - {}^l\bar{\mathbf{p}}_{i-1,m}^s\right) \\ ({}^l\bar{\mathbf{p}}_{i-1,m}^s - {}^l\bar{\mathbf{p}}_{i-1,n}^s) \times ({}^l\bar{\mathbf{p}}_{i-1,m}^s - {}^l\bar{\mathbf{p}}_{i-1,o}^s)\end{array}\right|}{\left|({}^l\bar{\mathbf{p}}_{i-1,m}^s - {}^l\bar{\mathbf{p}}_{i-1,n}^s) \times ({}^l\bar{\mathbf{p}}_{i-1,m}^s - {}^l\bar{\mathbf{p}}_{i-1,o}^s)\right|} \tag{5}$$

where k represents the edge or surface point indices in the current frame. m, n, o represent the point indices of matching edge line or planar patch in the last frame. For the raw edge point ${}^l\mathbf{p}_{i,k}^e$ in the set $\boldsymbol{\psi}_f^i$, ${}^l\bar{\mathbf{p}}_{i-1,m}^e$ and ${}^l\bar{\mathbf{p}}_{i-1,n}^e$ are the matching points in the set $\bar{\boldsymbol{\psi}}_c^{i-1}$ that make up the edge line in the last frame without motion distortion. Similarly, ${}^l\mathbf{p}_{i,k}^s$ is the raw surface point in the set $\boldsymbol{\zeta}_f^i$. ${}^l\bar{\mathbf{p}}_{i-1,m}^s$, ${}^l\bar{\mathbf{p}}_{i-1,n}^s$, and ${}^l\bar{\mathbf{p}}_{i-1,o}^s$ are the points that consist of the corresponding planar patch in the set $\bar{\boldsymbol{\chi}}_c^{i-1}$ or $\bar{\boldsymbol{\zeta}}_c^{i-1}$.

3.3. Semantic lidar odometry

The semantic lidar odometry module simultaneously estimates the absolute motion increment of the robot and the relative motion increment of the semantic object. Then, the absolute motion of each object in the odometry coordinate o can be calculated. Figure 4 shows the transformation of objects and the robot in the odometry coordinate.

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

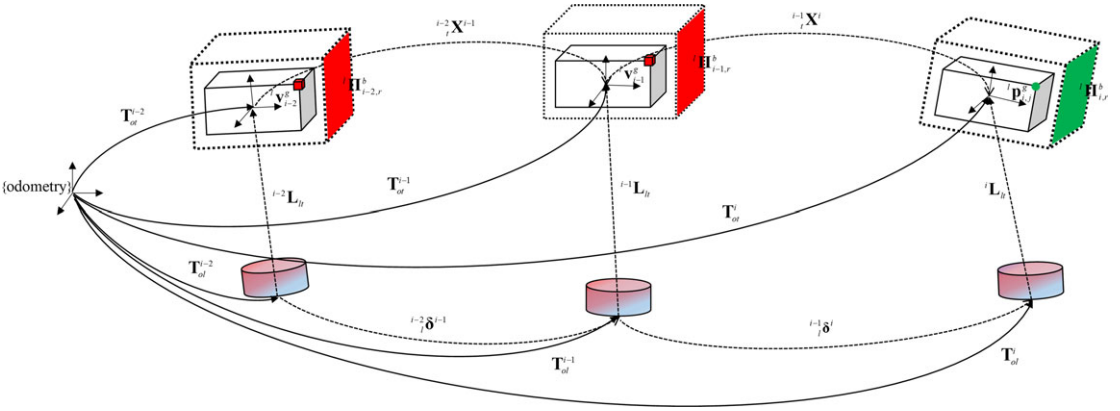

Figure 4. Notation and coordinates. *White solid cuboids and dashed cuboids are the detected objects and corresponding 3D-BB. Colored cylinders are the robot with lidar. Robot and object pose in the odometry coordinate are the solid lines. Object motion increments in the body-fixed coordinate are the dashed lines. Red parallelograms and cubes are the plane features generated by 3D-BB and matched voxels. Green parallelogram and dot are the semantic plane feature and geometric object point in current frame.*

3.3.1. Robust ego odometry

When the semantic object detection module makes the missing detection problem, the accuracy of the ego-motion estimation will be reduced by some unknown moving object points included in the ground or background feature sets. Accordingly, we solve this problem by using Algorithm 1 to perform a geometric consistency checking on current feature sets ($\boldsymbol{\chi}_f^i$, $\boldsymbol{\psi}_f^i$, $\boldsymbol{\zeta}_f^i$). The checked feature sets are defined as ($\tilde{\boldsymbol{\chi}}_f^i$, $\tilde{\boldsymbol{\psi}}_f^i$, $\tilde{\boldsymbol{\zeta}}_f^i$).

Considering the different degrees of freedom (DOF) constraints provided by the ground and background feature sets in estimating the robot pose and the computational efficiency problem, a two-step algorithm proposed in ref. [2] is used to estimate the ego-motion. First, we compute 3-DOF increment $[t_z, \theta_{roll}, \theta_{pitch}]$ by the checked ground feature set $\tilde{\boldsymbol{\chi}}_f^i$ and the candidate feature set $\bar{\boldsymbol{\chi}}_c^{i-1}$ that corrected the motion distortion. Then, another 3-DOF increment $[t_x, t_y, \theta_{yaw}]$ is estimated by surface feature points in $\tilde{\boldsymbol{\zeta}}_f^i$ with smaller roughness than R_a, edge feature set $\tilde{\boldsymbol{\psi}}_f^i$, and corresponding candidate sets. Finally, we accumulate the 6-DOF increment ${}_l^{i-1}\boldsymbol{\delta}^i = [t_x, t_y, t_z, \theta_{roll}, \theta_{pitch}, \theta_{yaw}]$ to compute the robot pose $\mathbf{T}_{ol}^i$ in the odometry coordinate o.

3.3.2. Semantic-geometric fusion MOT

To track semantic objects robustly and accurately, a least-squares estimator combining semantic box planes and geometric object points is designed for the MOT module. Before entering the increment estimation process, we use the Kuhn-Munkres [46] algorithm to obtain the object data association between frames.

Specifically, we need to construct the association matrix using some error functions that can evaluate the similarity of the object between frames: (1) The position error term $\mathbf{e}_p$ is computed by the distance between the object's predicted position which is inferred by the last motion model and object's detection position. (2) The yaw angle increment of the object forms the direction error term $\mathbf{e}_d$. (3) The difference in the semantic 3D-BB size between the objects constructs the scale error term $\mathbf{e}_s$. Further details on error calculation can be found in ref. [47].

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

Algorithm 1. Geometric consistency feature checking.

Input: Current feature sets: $(\chi_f^i, \psi_f^i, \zeta_f^i)$, candidate feature sets: $(\bar{\chi}_c^{i-1}, \bar{\psi}_c^{i-1}, \bar{\zeta}_c^{i-1})$, transformation ${}_l^{i-2}\delta^{i-1}$ from last loop, thresholds: $(th_{ed}, th_r, th_g, th_b)$, association update interval λ

Output: Consistent feature sets: $(\tilde{\chi}_f^i, \tilde{\psi}_f^i, \tilde{\zeta}_f^i)$

The point number in feature sets $(\chi_f^i, \psi_f^i, \zeta_f^i)$ is recorded as N_f. Let ${}_l^{i-1}\delta_{best}^i = {}_l^{i-2}\delta^{i-1}$;

for $i = 1 : N_{iter}$ **do**

  if $i \,\%\, \lambda = 0$ **then**

    The corrected feature sets $(\mathbf{m}^b_{\chi_f^i}, \mathbf{m}^b_{\psi_f^i}, \mathbf{m}^b_{\zeta_f^i})$ at the beginning of the frame are obtained by $(\chi_f^i, \psi_f^i, \zeta_f^i)$ with the relative timestamp and transformation ${}_l^{i-1}\delta_{best}^i$.

    Search the nearest point of the corrected feature sets $(\mathbf{m}^b_{\chi_f^i}, \mathbf{m}^b_{\psi_f^i}, \mathbf{m}^b_{\zeta_f^i})$ in $(\bar{\chi}_c^{i-1}, \bar{\psi}_c^{i-1}, \bar{\zeta}_c^{i-1})$. Calculate the corrected feature sets $(\mathbf{m}^e_{\chi_f^i}, \mathbf{m}^e_{\psi_f^i}, \mathbf{m}^e_{\zeta_f^i})$ at the end of the frame;

  end

  Random select matched pairs from $\mathbf{m}^e_{\chi_f^i}, \mathbf{m}^e_{\psi_f^i}$ and their nearest points in $(\bar{\chi}_c^{i-1}, \bar{\psi}_c^{i-1})$;

  Compute transformation ${}_l^{i-1}\delta^i$ based on ICP algorithm;

  Transform $(\chi_f^i, \psi_f^i, \zeta_f^i)$ to the starting time of the current frame with ${}_l^{i-1}\delta^i$. Find their nearest points in $(\bar{\chi}_c^{i-1}, \bar{\psi}_c^{i-1}, \bar{\zeta}_c^{i-1})$ and perform a Euclidean distance check with threshold th_{ed};

  Record the feature point number which passes the checking as $(\eta_{\chi_f}, \eta_{\psi_f}, \eta_{\zeta_f})$. Let $\eta_n = \eta_{\chi_f} + \eta_{\psi_f} + \eta_{\zeta_f}$;

  if $(\eta_n / N_f \geq th_r) \wedge (\eta_{\chi_f} \geq th_g) \wedge (\eta_{\psi_f} + \eta_{\zeta_f} \geq th_b) \wedge (\eta_n > \eta_{best})$ **then**

    Record feature sets $(\tilde{\chi}_f^i, \tilde{\psi}_f^i, \tilde{\zeta}_f^i)$;

    Let $\eta_{best} = \eta_n$, ${}_l^{i-1}\delta_{best}^i = {}_l^{i-1}\delta^i$;

  end

end

For objects that are tracked in the last frame but no matching relationship is currently found, we use the last calculated motion model to infer their positions in the subsequent two frames. If a successful match is still not found, they are removed.

Considering that the object point cloud $\boldsymbol{\gamma}^i$ (object ID k is omitted) obtained by direct measurement has motion distortion when the robot and object are in motion, we use the VG-ICP [48] algorithm and the relative motion model based on linear interpolation to construct the geometric cost function.

Assuming the j-th raw object point in the lidar coordinate l is ${}^l\mathbf{p}_{i,j}^g$, and it has a relative timestamp ${}^r t_j$. ${}_l^{i-1}\mathbf{H}^i \in \mathrm{SE}(3)$ is the transformation between interval $\left[t^{i-1}, t^i\right]$. Then, the distortion-corrected transformation ${}_l^{i-1}\mathbf{H}_j^i$ for each raw object point can be calculated:

$$ {}_l^{i-1}\mathbf{H}_j^i = {}^r t_j \mathbf{H}^i \tag{6} $$

We use ${}_l^{i-1}\mathbf{R}^i$ and ${}_l^{i-1}\mathbf{t}^i$ to represent the rotational and translational components of the increment ${}_l^{i-1}\mathbf{H}^i$. The geometric error term $\mathbf{e}_{g_j}$ from the distribution to the multi-distribution can be defined as follows:

$$ \mathbf{e}_{g_j} = \frac{\sum {}^l\bar{\mathbf{p}}_{i-1,m}^g}{N_j} - {}_l^{i-1}\mathbf{H}_j^{il}\mathbf{p}_{i,j}^g \tag{7} $$

For the matched voxel ${}^l\bar{\mathbf{v}}_{i-1}^g$ of the current object point ${}^l\mathbf{p}_{i,j}^g$ at the end time of the last frame, it encloses N_j object points ${}^l\bar{\mathbf{p}}_{i-1,m}^g$ that have corrected the motion distortion. The covariance matrix $\boldsymbol{\Omega}_j$ corresponding

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

to the error term $\mathbf{e}_{g_j}$ is as follows:

$$\mathbf{\Omega}_j = \frac{\sum C^g_{i-1,m}}{N_j} - {}^{i-1}_{l}\mathbf{H}^i_j C^g_{i,jl} {}^{i-1}_{l}\mathbf{H}^{iT}_j \tag{8}$$

$C^g_{i-1,m}$ and $C^g_{i,j}$ denote the covariance matrix. They describe the surrounding shape distribution of the point in the matched voxel and point ${}^l\mathbf{p}^g_{i,j}$.

Note that objects detected in a single frame always have less constrained information. Meanwhile, the inaccurate initial value of the estimator at the beginning of the optimization may also lead to poor matchings. Eq. (7) sometimes fails to constrain the 6-DOF motion estimation. Therefore, while using geometric constraints of the point cloud, this work also introduces semantic constraints on increment estimation to ensure the correct direction of convergence.

Specifically, we model a 3D-BB $\boldsymbol{\tau}^i$ as a cuboid containing six planes. Each plane is represented by the normal vector $\mathbf{n}$ ($\|\mathbf{n}\|_2 = 1$) and the distance d from the origin to the plane, that is $\boldsymbol{\pi} = [\mathbf{n}, d]$. Unlike the geometric point cloud error with iterative matching for estimation, the matching relationships for 3D-BB planes are directly available. Then, using the closest point $\mathbf{\Pi} = \mathbf{n}d$ proposed in ref. [49], we can construct the plane-to-plane semantic error term $\mathbf{e}_{b_r}$:

$$\mathbf{e}_{b_r} = {}^l\mathbf{\Pi}^b_{i-1,r} - {}^l\hat{\mathbf{\Pi}}^b_{i-1,r} \tag{9}$$

${}^l\mathbf{\Pi}^b_{i-1,r}$ and ${}^l\hat{\mathbf{\Pi}}^b_{i-1,r}$ are the r-th measured and estimated planes on 3D-BB in frame i in the lidar coordinate l. As described in ref. [50], the motion transformation of a plane is defined as:

$$\begin{bmatrix} {}^l\mathbf{n}^b_{i-1,r} \\ {}^ld^b_{i-1,r} \end{bmatrix} = \begin{bmatrix} {}^{i-1}_{l}\mathbf{R}^i & \mathbf{0} \\ -({}^{i-1}_{l}\mathbf{R}^{iT} {}^{i-1}_{l}\mathbf{t}^i)^T & 1 \end{bmatrix} \begin{bmatrix} {}^l\mathbf{n}^b_{i,r} \\ {}^ld^b_{i,r} \end{bmatrix} \tag{10}$$

Taking Eq. (10) into ${}^l\hat{\mathbf{\Pi}}^b_{i-1,r}$, we can further obtain:

$${}^l\hat{\mathbf{\Pi}}^b_{i-1,r} = \left({}^{i-1}_{l}\mathbf{R}^{il}_j \mathbf{n}^b_{i,r}\right)\left(-{}^{i-1}_{l}\mathbf{t}^{iT}_j {}^{i-1}_{l}\mathbf{R}^{il}_j \mathbf{n}^b_{i,r} + d\right) \tag{11}$$

When using the least-squares method to estimate ${}^{i-1}_{l}\mathbf{H}^i$, it is necessary to obtain the derivative of $\mathbf{e}_{b_r}$ with respect to ${}^{i-1}_{l}\mathbf{H}^i$:

$$\frac{\partial \mathbf{e}_{b_r}}{\partial \delta {}^{i-1}_{l}\boldsymbol{\theta}^i_j} = -\left[\mathbf{F}^{i-1}_{l}\mathbf{t}^{iT}_j \mathbf{F}\right]_\times - \mathbf{F}^{i-1}_{l}\mathbf{t}^{iT}_j [\mathbf{F}]_\times + [\mathbf{F}d]_\times \tag{12}$$

$$\frac{\partial \mathbf{e}_{b_r}}{\partial \delta {}^{i-1}_{l}\mathbf{t}^i_j} = \mathbf{F}\mathbf{F}^T \tag{13}$$

where $\mathbf{F} = {}^{i-1}_{l}\mathbf{R}^{il}_j \mathbf{n}^b_{i,r}$, $\frac{\partial \mathbf{e}_{b_r}}{\partial \delta {}^{i-1}_{l}\boldsymbol{\theta}^i_j}$, and $\frac{\partial \mathbf{e}_{b_r}}{\partial \delta {}^{i-1}_{l}\mathbf{t}^i_j}$ represent the Jacobians of the error term $\mathbf{e}_{b_r}$ w.r.t. the rotation and translation components. The detailed derivation of Eqs. (12) and (13) can be found in Appendix.

Finally, Eq. (14) is the comprehensive cost function for the estimation of ${}^{i-1}_{l}\mathbf{H}^i$ considering the semantic confidence μ^i:

$$\underset{{}^{i-1}_{l}\mathbf{H}^i}{\arg\min} \frac{2-\mu^i}{2N_g} \sum_{j=1}^{N_g} \left(N_j \mathbf{e}^T_{g_j} \mathbf{\Omega}^{-1}_j \mathbf{e}_{g_j}\right) + \frac{\mu^i}{2N_b} \sum_{r=1}^{N_b} \left(\mathbf{e}^T_{b_r} \mathbf{e}_{b_r}\right) \tag{14}$$

where N_g is the point number in the object point cloud. N_b is the plane number on 3D-BB.

In this paper, we use the Levenberg-Marquardt [51] algorithm to solve Eq. (14) and the estimated relative increment to get the motion-distorted object point cloud $\bar{\boldsymbol{\gamma}}^i$ and object edge feature set $\bar{\boldsymbol{\gamma}}^i_c$. $\bar{\boldsymbol{\gamma}}^i$ and $\bar{\boldsymbol{\gamma}}^i_c$ will be used for object tracking in the next frame and accumulated error elimination in the mapping module.

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

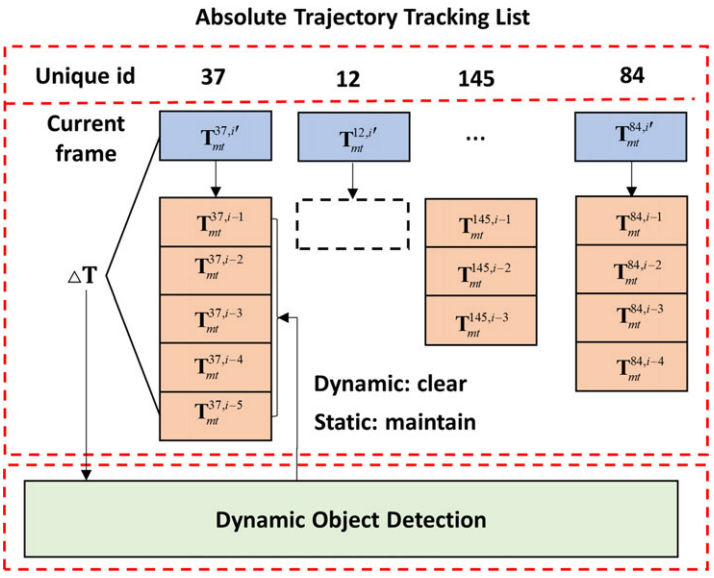

Figure 5. Dynamic object detection based on ATTL. *The blue boxes represent the object's predicted pose, and orange boxes represent their tracking trajectories that have corrected the accumulated error. Objects with IDs 37 and 84 are in a normal tracking state. The object with ID 12 will be initialized as a static object, and the object with ID 145 will be deleted because the current frame does not observe it.* $\Delta \boldsymbol{T}$ *is used to determine the current motion state of each object.*

3.3.3. Object odometry

The aim of object odometry is to get each object pose in odometry coordinate o. First, using the results calculated above, we can compute the object motion increment ${}^{i}_{l}\mathbf{X}^{i-1}$ in the lidar coordinate l:

$$ {}^{i}_{l}\mathbf{X}^{i-1} = {}^{i-1}_{l}\boldsymbol{\delta}^{ii}_{l}\mathbf{H}^{i-1} \tag{15} $$

Suppose the transformation from object to robot in the frame $i-1$ is ${}^{i-1}\mathbf{L}_{lt}$. The absolute increment of object motion in the body-fixed coordinate [52] can be expressed as:

$$ {}^{i-1}_{t}\mathbf{X}^{i} = {}^{i-1}\mathbf{L}^{-1}_{lt}\,{}^{i}_{l}\mathbf{X}^{i-1}\,{}^{i-1}\mathbf{L}_{lt} \tag{16} $$

Finally, we accumulate the increment ${}^{i-1}_{t}\mathbf{X}^{i}$ to obtain the object pose $\mathbf{T}^{i}_{ot}$ in the odometry coordinate o. For an object detected at the first frame, we compute its initial pose $\mathbf{T}_{ot}$ in the odometry coordinate o based on the current robot pose $\mathbf{T}_{ol}$ and the transformation $\mathbf{L}_{lt}$ inferred from the semantic detection module.

3.4. 4D scene mapping

To generate a long-term static map that can be reused, we maintain an absolute trajectory tracking list (ATTL) for processing semantic objects in the field of view. The list contains the unique id, continuous tracking trajectory, accumulated static map, and motion state of each object.

3.4.1. Dynamic object detection

As shown in Fig. 5, the ATTL maintained in the mapping module is used to analyze the motion state of each instance object. First, each object in the current frame is matched with the unique ID generated by the SGF-MOT module and maintained in ATTL. Then, we use the current error correction matrix $\mathbf{T}^{i-1}_{mo}$

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

to predict the object pose $\mathbf{T}_{mt}^{i\,'}$ in the map coordinate m:

$$\mathbf{T}_{mt}^{i\,'} = \mathbf{T}_{mo}^{i-1}\mathbf{T}_{ot}^{i} \tag{17}$$

By calculating the motion increment of the predicted object pose and the static oldest object pose in ATTL, we can determine whether each object is in motion. Note that the detected object in the first frame is set to be static by default.

3.4.2. Static map creation

By matching the static object feature set $\bar{\boldsymbol{\gamma}}_c^i$ in the current frame with the accumulated object map, ground feature set $\bar{\boldsymbol{\chi}}_c^i$, background edge feature set $\bar{\boldsymbol{\psi}}_c^i$, and surface feature set $\bar{\boldsymbol{\zeta}}_c^i$ with their accumulated feature map, it is possible to obtain the corrected robot pose $\mathbf{T}_{ml}^i$ in the map coordinate m. Then, we align the non-semantic feature sets to the map coordinate m to create a long-term static map. Readers can refer to ref. [1] for more details about feature mapping and optimization.

Using the ego pose $\mathbf{T}_{ol}^i$ in the odometry coordinate o and the corrected pose $\mathbf{T}_{ml}^i$ in the map coordinate m, the updated accumulated error correction matrix $\mathbf{T}_{mo}^i$ can be computed.

3.4.3. Tracking list maintain

The ATTL can be updated based on the motion state of each object. If the object is moving, the object map and tracking trajectory should be cleared first. Otherwise, they will remain unchanged. Then, we use $\mathbf{T}_{mo}^i$ and $\mathbf{T}_{ml}^i$ to convert the object odometry pose $\mathbf{T}_{ot}^i$ and extracted object feature set $\bar{\boldsymbol{\gamma}}_c^{k,i}$ into the map coordinate m, which will participate in the following mapping steps.

4. Results

In this section, we validate the proposed MLO system and SGF-MOT module using the public KITTI [53] dataset. First, the ego-motion trajectory generated by the MLO system is compared with other feature-based 3D lidar SLAM systems (A-LOAM [54], Lego-LOAM [2], F-LOAM [55]) on the KITTI-odometry benchmark. In addition, we compared the MLO system with the visual SLAM (DSO [56]) and multi-sensor fusion systems (V-LOAM [36], LIO-SAM [40]) under the KITTI-raw benchmark to analyze its performance in dynamic scenarios. The geometric consistency checking algorithm and the dynamic object detection method are verified. Then, the SGF-MOT module is compared in the KITTI-tracking benchmark with another two efficient MOT methods (AB3DMOT [21], PC3T [57]), which also perform on CPU and use point clouds as input. Meanwhile, we use the absolute object trajectory generated by MLO to evaluate the advantages of the SGF-MOT method compared with semantic-only and geometric-only methods under the KITTI-raw benchmark.

4.1. Experiment setup and evaluation metrics

All experiments in this paper were performed with AMD® Ryzen 7 5800h (8 cores @3.2 GHz), 16 GB RAM, ROS Melodic, and NVIDIA GeForce RTX 3070. The absolute trajectory error [58] (ATE) is used to evaluate localization accuracy. The relative rotation error (RRE) and relative translation error (RTE) [53] can evaluate the drift of odometry estimation. The CLEAR MOT metric [59] denotes the estimated accuracy and consistent tracking of object configurations over time for the MOT module.

Note that our MOT module is performed in the lidar coordinate. Therefore, the ability to track the relative motion of the object is evaluated. On the other hand, object trajectories in the map coordinate m evaluate the overall accuracy of the MLO system.

Considering that the KITTI-raw benchmark does not provide ground-truth trajectories that can be used directly for localization evaluation, we process the raw GPS/IMU data by the Kalman filter

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

[60] algorithm to get the ground truth of robot pose in the map coordinate m. Meanwhile, based on the artificial annotation in the KITTI-tracking dataset, we can further compute the absolute tracking trajectory of each object.

The frameworks used in the comparison experiments, SLAM and MOT, both keep a default parameter configuration. The parameter configuration of each module in the MLO system is as follows:

(1) Geometric consistency feature checking algorithm (3.3.1): Total iteration times $N_{iter,gc} = 100$; Threshold for Euclidean distance $th_{ed} = 0.5$ m; Proportional threshold $th_r = 0.4$; Threshold for point number $th_g = 20$, $th_b = 30$; Association update interval $\lambda = 5$.

(2) Kuhn-Munkres algorithm (3.3.2): Position error weight $w_p = 0.7$; Direction error weight $w_d = 0.25$; Size error weight $w_s = 0.1$; Total error threshold $th_{km} = 5$.

(3) Semantic-geometric fusion least-squares estimator (3.3.2): Number of kd-tree nearest points used in computing covariance $N_{kd} = 10$; Voxel size in VG-ICP $v_s = 0.2$ m; Maximum iteration times $N_{iter,ls} = 5$; Angle increment threshold $th_{a,ls} = 0.1°$, Translation increment threshold $th_{t,ls} = 1e^{-2}$ m (convergence conditions of the estimator).

(4) Absolute trajectory tracking list (3.4.1): Yaw angle increment threshold $th_{yaw,attl} = 2°$; Translation increment threshold $th_{t,attl} = 0.2$ m (static object judgment conditions).

4.2. Ego odometry localization accuracy experiment

The KITTI-odometry benchmark contains many static objects with long sequences. The KITTI-raw dataset, on the other hand, contains more challenging scenes, such as the city sequence with unknown dynamic semantic objects (trains). We evaluate the MLO system in these two datasets by comparing it with other state-of-the-art SLAM systems. Since the KITTI dataset labels semantic objects within the camera's field of view, only the ROI-cutting lidar data is input into the fusion perception module. Figure 6 shows the long-term static map created by the MLO system.

For the experimental 3D lidar SLAM system, A-LOAM [54] is a simplified version of LOAM [1] with removing the use of IMU. It also extracts edge and surface features by the roughness of the scan line to perform frame-to-frame matching for odometry estimation and frame-to-map matching for lidar mapping. Lego-LOAM [2] takes into account ground constraints. Use the ground feature set and background edge feature set to estimate the increment $[t_z, \theta_{roll}, \theta_{pitch}]$ and $[t_x, t_y, \theta_{yaw}]$ to improve efficiency. F-LOAM [55] proposes a two-step algorithm for motion distortion in the point cloud, and then, localization and mapping are performed directly through frame-to-map matching. To fully validate the system performance, we also selected a purely visual SLAM and two multi-sensor fusion systems for comparative experiments. DSO-stereo [56] is a direct sparse stereo visual odometry based on the assumption of constant luminosity. LIO-SAM [40] is a tightly coupled lidar-inertial odometry that uses high-frequency imu-predicted pose as the initial estimation and can effectively correct point cloud distortion even when the robot is in vigorous motion. V-LOAM [36] is a visual-lidar fusion framework that fixes the scale uncertainty problem in monocular visual odometry by point cloud and uses high-frequency visual odometry as a prior value for high-precision lidar mapping.

As shown in Table I, our method successfully performs all sequences under the KITTI-odometry benchmark and achieves the best results in most cases. When estimating $[t_x, t_y, \theta_{yaw}]$ based on the background edge feature set, the Lego-LOAM [2] system shows an obvious drift in many sequences because it uses only point clouds from the camera perspective. Our system also uses the two-step estimation algorithm to ensure computational efficiency. However, using the background surface feature set based on roughness prior R_a and the geometric consistency check algorithm improves the robustness of the MLO system.

In addition, detecting dynamic objects by ATTL and adding static object constraints for static map creation can better eliminate accumulated error compared to the MLO-attl system, which does not use the dynamic object detection method in the mapping module. In sequences 00, 02, 03, 05, and 07 containing many static objects, the MLO system achieved better localization accuracy.

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

Table I. *Ego localization accuracy under KITTI-odometry benchmark.*

Absolute Motion Trajectory RMSE [m]					
Seq	**A-LOAM**	**Lego-LOAM**	**F-LOAM**	**MLO-attl**	**MLO**
00	12.12	41.99	14.71	8.23	**7.52**
01	15.81	**fail**	17.61	**12.11**	12.42
02	**fail**	**fail**	**fail**	18.38	**17.65**
03	0.57	14.19	0.90	0.52	**0.50**
04	0.45	**fail**	0.36	0.29	**0.27**
05	5.72	8.12	8.06	2.74	**1.92**
06	**0.44**	**fail**	1.04	0.45	0.46
07	2.73	3.75	2.49	2.33	**1.48**
08	**4.42**	**fail**	6.02	5.18	4.86
09	**3.48**	**fail**	21.53	4.66	4.72
10	**1.18**	10.57	1.54	1.80	1.90
Mean	4.69	**fail**	7.43	3.88	**3.66**

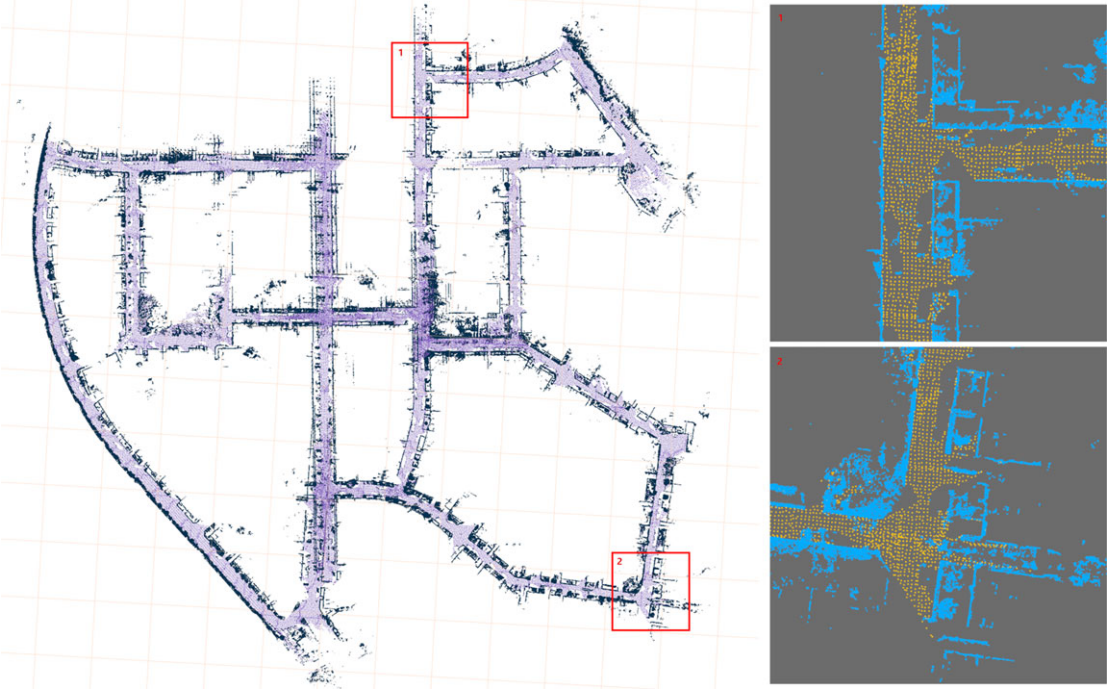

Figure 6. Point cloud map created by sequence 00 in the KITTI-odometry dataset. *As shown in the magnified part on the right, the resulting map does not contain semantic objects that may change over time by maintaining semantic objects and the static map separately.*

Table II shows the localization accuracy of the proposed MLO system and other SLAM frameworks in complex scenes with dynamic objects. Although the DSO-stereo system avoids the scale uncertainty problem of monocular visual SLAM, it is found that the overall accuracy of lidar-based SLAM systems using point clouds is better. Compared to the A-LOAM, LIO-SAM, and V-LOAM systems, the proposed MLO system has higher robustness for localization in dynamic scenes. V-LOAM uses visual odometry as a prior, but fails several times in dynamic scenes. The reason is that the system does not process the

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

Table II. *Evaluation for ego localization accuracy under KITTI-raw City and Residential sequences.*

	Absolute Motion Trajectory RMSE [m]					
Sequence	**A-LOAM**	**DSO-stereo**	**V-LOAM**	**LIO-SAM**	**MLO-gc**	**MLO**
0926-0011	0.13	0.33	**0.11**	0.12	0.13	0.13
0926-0014	0.22	**fail**	0.29	0.35	**0.12**	**0.12**
0926-0019	1.64	1.03	**fail**	0.25	**0.13**	0.16
0926-0022	0.66	2.4	2.02	0.67	0.62	**0.59**
0926-0039	0.23	0.86	**fail**	**0.13**	0.18	0.18
0926-0056	0.22	0.38	**fail**	0.41	**fail**	**0.12**
0926-0059	**0.08**	0.28	0.18	0.14	0.13	0.12
0926-0064	0.43	2.90	1.43	0.67	0.44	**0.41**
0929-0071	0.88	0.86	**0.8**	0.89	0.90	0.89
Mean	0.50	1.13	0.81	0.40	0.33	**0.28**

Table III. *Evaluation for multi-object tracking module under the KITTI-tracking benchmark.*

	MOTA$_{2d}$	**MOTP**$_{2d}$	**Time** [ms]
AB3DMOT	68.83	**87.23**	4.82
PC3T	**85.33**	87.18	**4.5**
SGF-MOT	81.10	87.12	4.71

dynamic objects appearing in the image and could not achieve the ideal situation of providing a better prior value. Meanwhile, the LIO-SAM uses the measurement of IMU, which is not affected by external environments, thus improving the robustness and accuracy of the system compared with V-LOAM. However, the IMU measurement can only estimate the ego localization of the robot, so it cannot correct the relative motion distortion of extracted features on dynamic objects, which may also lead to estimation errors in the system.

Taking sequence 0926-0056 as an example, there is a train that the object detection module cannot detect. The MLO-gc system without Algorithm 1 to check geometric consistency (When Algorithm 1 is not used, the amount of dynamic data association for pose estimation will be significant) and the V-LOAM system, which uses visual odometry as a prior, cannot deal with this situation. A-LOAM and LIO-SAM can work stably in dynamic scenarios. However, their localization accuracy is slightly lower because they do not filter the extracted features and directly use all features for localization and mapping. Compared with the above framework, the MLO system, which considers static and dynamic object processing, achieved the best localization results.

4.3. Object tracking accuracy and robustness experiment

First, we compare the SGF-MOT module proposed in this paper with AB3DMOT [21] and PC3T [57] trackers under the KITTI-tracking benchmark. The tracking results are evaluated in 2D image, which can be computed by the KITTI calibration matrix. AB3DMOT [21] only takes the point cloud as input and uses the Kalman filter for 3D-BB tracking in the lidar coordinate. PC3T [57] uses the ground truth of ego localization provided by the KITTI dataset to transform the detected object into the map coordinate m. Then, it performs absolute motion tracking based on the kinetic model of the object.

As shown in Table III, thanks to the accurate point cloud, the three methods have little difference in the MOTP metric, which illustrates the estimation accuracy of the proposed tracker. Since the object is

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

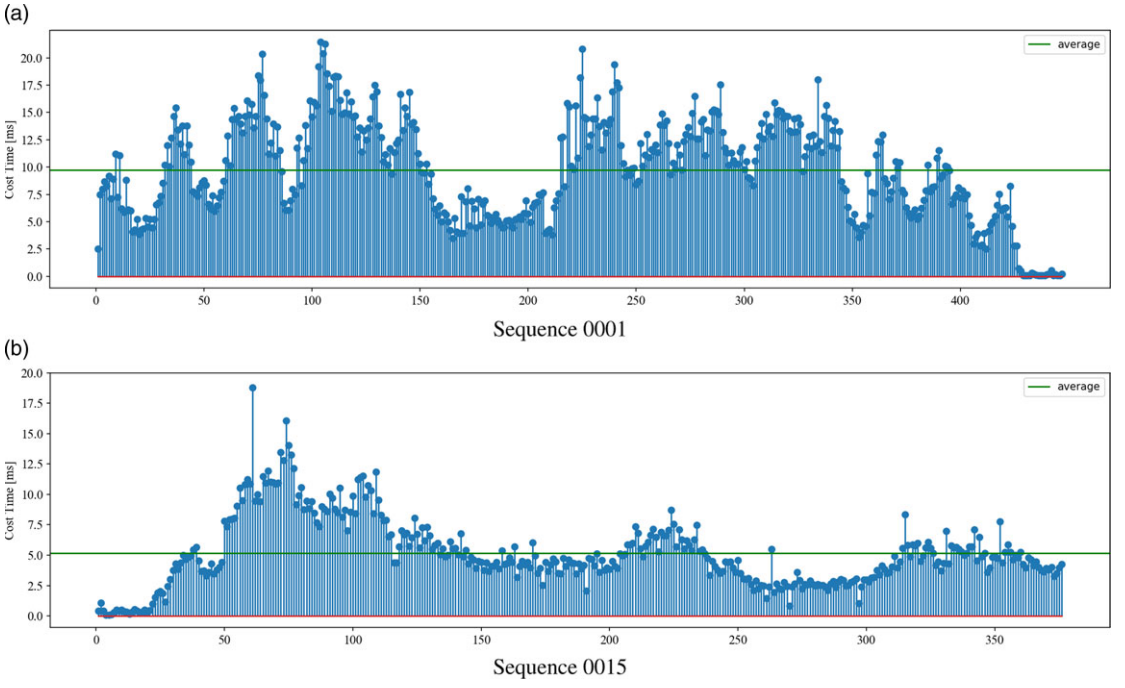

Figure 7. *The running time of SGF-MOT module in sequences filled with objects under the KITTI-tracking dataset.*

Table IV. *Evaluation for absolute object trajectory accuracy under KITTI-raw City and Road sequences.*

		Geometry only			**Semantic only**			**Fusion method**		
Sequence	**id**	**ATE [m]**	**RTE [m/m]**	**RRE [deg/m]**	**ATE [m]**	**RTE [m/m]**	**RRE [deg/m]**	**ATE [m]**	**RTE [m/m]**	**RRE [deg/m]**
0926-0009	87	**0.11**	0.11	0.35	0.25	0.33	2.10	0.17	0.13	0.36
	89	**0.06**	0.06	0.32	0.16	0.16	1.55	0.14	0.11	0.86
0926-0013	1	**0.21**	0.15	0.93	0.39	0.29	3.85	0.40	0.26	2.76
	2	2.54	1.03	9.80	**0.22**	0.18	1.50	0.23	0.16	1.30
0926-0018	6	0.11	0.12	1.77	0.12	0.13	2.53	**0.08**	0.12	1.09
	11	**0.08**	0.10	1.39	0.15	0.16	1.90	**0.11**	0.12	1.02
0926-0051	21	2.28	1.86	2.83	0.20	0.18	1.73	**0.18**	0.13	1.22
	26	**0.07**	0.12	0.27	0.14	1.54	1.21	0.09	0.09	1.13
0926-0084	14	4.81	1.11	7.11	0.27	0.38	6.25	**0.23**	0.35	6.14
0926-0015	32	**0.14**	0.25	1.67	0.19	0.21	2.56	0.16	0.16	1.96
	35	3.29	1.07	2.28	0.30	0.28	5.44	**0.27**	0.27	7.15
0926-0028	1	**fail**	**fail**	**fail**	0.31	0.22	2.22	**0.31**	0.21	2.06
0926-0032	15	0.18	0.16	0.43	0.20	0.21	1.06	**0.17**	0.18	1.31
Mean		1.16	0.52	2.43	0.22	0.34	2.64	**0.19**	**0.17**	**2.19**

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

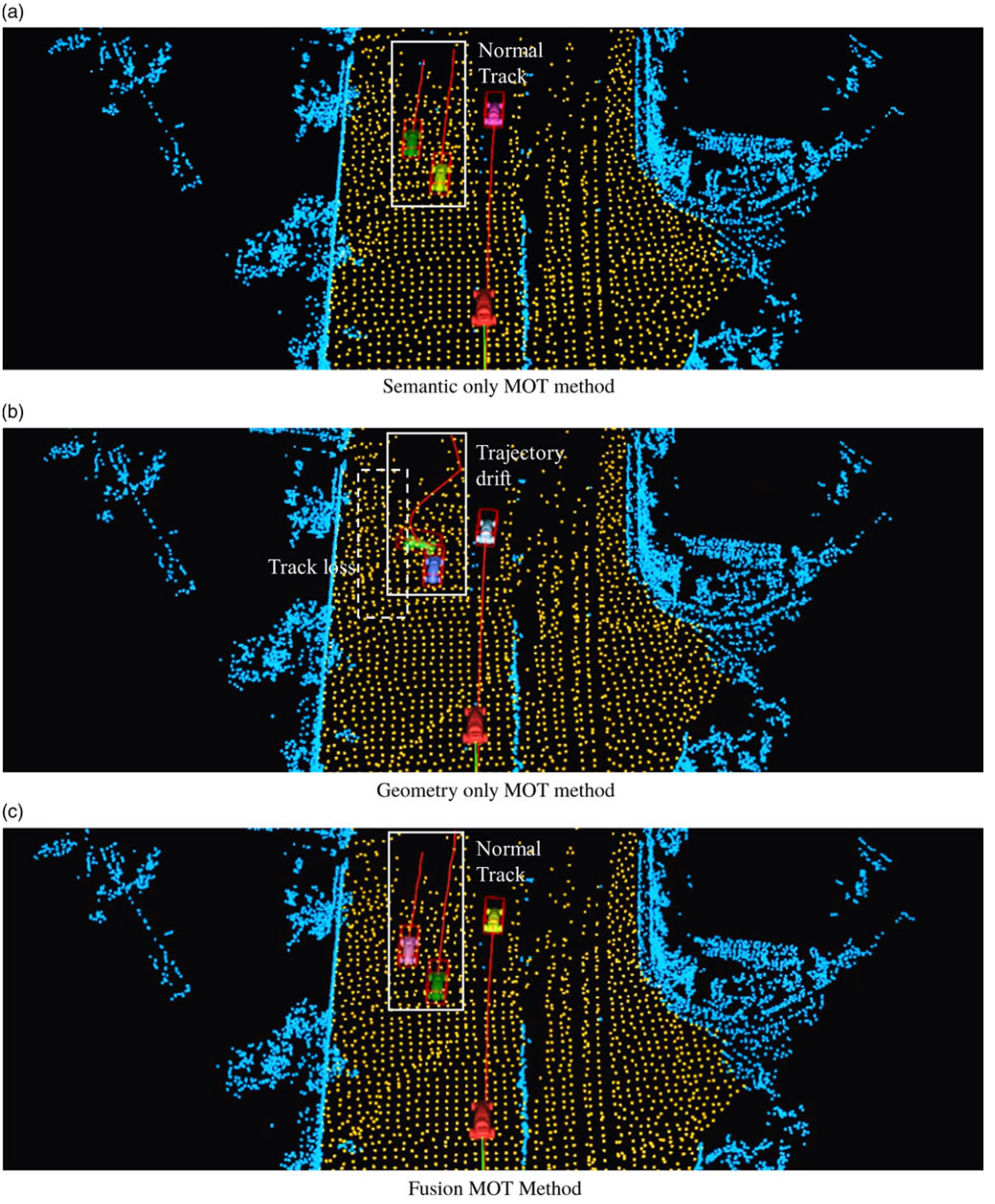

Figure 8. *Comparison of different tracking methods in sequence 0926-0056 under the KITTI-tracking dataset. Both semantic-only and fusion methods can track objects robustly, but geometric-only method has the problems of object tracking loss and trajectory drift.*

https://doi.org/10.1017/S0263574723001868 Published online by Cambridge University Press

transformed into the map coordinate m using the prior ego ground truth, PC3T [57] tracker only needs to consider the uncertainty caused by absolute object motion between frames. Moreover, it achieves the best results on the MOTA metric. The proposed SGF-MOT tracker does not use ego localization information but also improves tracking robustness by introducing a least-squares estimator fusing geometric and semantic constraints. So, we achieve better MOTA results than the AB3DMOT [21] tracker.

Like the other two methods, our method runs in real-time on the CPU. The running time of the SGF-MOT module executed on sequences 0001 and 0015 is shown in Fig. 7. On the KITTI-raw City and Road datasets, which are full of moving objects, we used the absolute tracking trajectory of objects generated by the MLO system to verify the semantic-geometric fusion tracking method (Fusion Method). The fusion method is compared to trackers that use only geometric object points (Geometry only) and semantic box planes (Semantic only) for motion estimation.

As shown in Table IV, the object localization accuracy of the geometry-only method exhibits a distinct "bipolar" distribution. There are two main reasons why the geometry-only method cannot reliably track the object: (1) The estimator requires iterative matching to obtain the point-level data association and places high demands on the initial value of the estimation. (2) Semantic objects are sometimes observed only from a specific perspective within the lidar perception range, for example, when the vehicle is directly in front of or next to the robot. In this case, the lidar sensor can only obtain a single planar point cloud with 3-DOF constraints on the motion of the object. On the other hand, we also find that the geometry-only tracking method can work better than the semantic-only tracking method in terms of accuracy when the estimator is successfully convergent.

Therefore, semantic box constraints, which can provide complete motion constraints, should be considered together with high-precision measurement point clouds in the MOT task. In this way, the tracker can well balance the accuracy of estimation and the robustness of tracking and achieve better overall performance. The experimental results in Table IV support this conclusion. The object motion trajectories of different tracking methods under sequence 0926-0056 in Fig. 8 can also confirm this qualitatively.

5. Conclusions

In this paper, we propose a novel MLO system to solve the problem of simultaneous localization, mapping, and MOT only using a lidar sensor. Specifically, we propose a fused least-squares estimator that uses the semantic bounding box and geometric point cloud for updating the object SGF-MOT. In the mapping module, dynamic semantic objects are detected based on the applied absolute trajectory tracking list, and the system can achieve reliable mapping in highly dynamic scenarios. Experiments with open datasets show that the SGF-MOT module can achieve better overall performance compared to the pure semantic and geometry tracking methods and other advanced MOT frameworks. Compared with other lidar, visual, and multi-sensor fusion SLAM systems, the MLO system with the SGF-MOT tracker can provide more stable and accurate robot and object localization in complex scenes. On the other hand, the high-frequency IMU, which is unaffected by the external environment, can provide good guidance in extracting static feature points and effectively cope with the intense movements of the robot. Therefore, the correct integration of the IMU measurement into the MLO and the building of a fusion SLAM system that can adapt to dynamic scenes as well as to the complex motions of the robot are meaningful problems.

Supplementary material. The supplementary material for this article can be found at http://doi.org/10.1017/S0263574723001868.

Author contributions. Tingchen Ma conceived and designed the study. Guolai Jiang and Sheng Xu completed the experimental verification. Tingchen Ma, Guolai Jiang, Yongsheng Ou, and Sheng Xu wrote the article.

Financial support. This research was funded by the National Key Research and Development Program of China under Grant 2018AAA0103001; in part by the National Natural Science Foundation of China (Grants No. 62173319, 62063006);

in part by the Guangdong Basic and Applied Basic Research Foundation (2022B1515120067); in part by the Guangdong Provincial Science and Technology Program (2022A0505050057); and in part by the Shenzhen Fundamental Research Program (JCYJ20200109115610172).

Competing interests. The authors declare no competing interests exist.

Ethical approval. Not applicable.

References

[1] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," *Rob. Sci. Syst.* **2**(9), 1–9 (2014).
[2] T. Shan and B. Englot, "Lego-loam: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018) pp. 4758–4765.
[3] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular slam system," *IEEE Trans. Rob.* **31**(5), 1147–1163 (2015).
[4] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments," *Rob. Sci. Syst.* **2018**(1), 59 (2018).
[5] J.-E. Deschaud, "IMLS-SLAM: Scan-to-Model Matching Based on 3d Data," *2018 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2018) pp. 2480–2485.
[6] F. Moosmann and T. Fraichard, "Motion Estimation from Range Images in Dynamic Outdoor Scenes," *2010 IEEE International Conference on Robotics and Automation* (IEEE, 2010) pp. 142–147.
[7] J. Park, Y. Cho and Y.-S. Shin, "Nonparametric background model-based lidar slam in highly dynamic urban environments," *IEEE Trans. Intell. Transp.* **23**(12), 24190–24205 (2022).
[8] W. Tan, H. Liu, Z. Dong, G. Zhang and H. Bao, "Robust Monocular SLAM in Dynamic Environments," *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (IEEE, 2013) pp. 209–218.
[9] P. Ruchti and W. Burgard, "Mapping with Dynamic-Object Probabilities Calculated from Single 3d Range Scans," *2018 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2018) pp. 6331–6336.
[10] P. Pfreundschuh, H. F. Hendrikx, V. Reijgwart, R. Dubé, R. Siegwart and A. Cramariuc, "Dynamic Object Aware Lidar SLAM Based on Automatic Generation of Training Data," *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021) pp. 11641–11647.
[11] J. Cheng, Y. Sun and M. Q.-H. Meng, "Robust semantic mapping in challenging environments," *Robotica* **38**(2), 256–270 (2020).
[12] G. Chen, B. Wang, X. Wang, H. Deng, B. Wang and S. Zhang, "PSF-LO: Parameterized Semantic Features Based Lidar Odometry," *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021) pp. 5056–5062.
[13] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley and C. Stachniss, "Suma++: Efficient Lidar-based Semantic SLAM," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2019) pp. 4530–4537.
[14] A. Milioto, I. Vizzo, J. Behley and C. Stachniss, "Rangenet++: Fast and Accurate Lidar Semantic Segmentation," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2019) pp. 4213–4220.
[15] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vision Comput.* **10**(3), 145–155 (1992).
[16] A. Dewan, T. Caselitz, G. D. Tipaldi and W. Burgard, "Motion-based Detection and Tracking in 3d Lidar Scans," *2016 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2016) pp. 4508–4513.
[17] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM* **24**(6), 381–395 (1981).
[18] A. Dewan, T. Caselitz, G. D. Tipaldi and W. Burgard, "Rigid Scene Flow for 3d Lidar Scans," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2016) pp. 1765–1770.
[19] M. Sualeh and G.-W. Kim, "Semantics aware dynamic slam based on 3d modt," *Ah S. Sens.* **21**(19), 6355 (2021).
[20] R. E. Kalman, A new approach to linear filtering and prediction problems (1960).
[21] X. Weng, J. Wang, D. Held and K. Kitani, "3d Multi-Object Tracking: A Baseline and New evaluation Metrics," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2020) pp. 10359–10366.
[22] S. Shi, X. Wang and H. Li, "Pointrcnn: 3d Object Proposal Generation and Detection from Point Cloud," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019) pp. 770–779.
[23] A. Kim, A. Ošep and L. Leal-Taixé, "Eagermot: 3d Multi-Object Tracking via Sensor Fusion," *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021) pp. 11315–11321.
[24] S. Wang, P. Cai, L. Wang and M. Liu, "Ditnet: End-to-end 3d object detection and track id assignment in spatio-temporal world," *IEEE Rob. Autom. Lett.* **6**(2), 3397–3404 (2021).
[25] K. Huang and Q. Hao, "Joint Multi-Object Detection and Tracking with Camera-Lidar Fusion for Autonomous Driving," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2021) pp. 6983–6989.
[26] Á. Llamazares, E. J. Molinos and M. Ocaña, "Detection and tracking of moving obstacles (datmo): A review," *Robotica* **38**(5), 761–774 (2020).
[27] Z. Liu and F. Zhang, "Balm: Bundle adjustment for lidar mapping," *IEEE Rob. Autom. Lett.* **6**(2), 3184–3191 (2021).
[28] B. Triggs, P. F. McLauchlan, R. I. Hartley and A. W. Fitzgibbon, "Bundle Adjustment–a Modern Synthesis," **In:** *International Workshop On Vision Algorithms* (Springer, 1999) pp. 298–372.

[29] P. Dellenbach, J.-E. Deschaud, B. Jacquet and F. Goulette, "CT-ICP: Real-Time Elastic Lidar Odometry with Loop Closure," *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022) pp. 5580–5586.
[30] W. Liu, W. Sun and Y. Liu, "Dloam: Real-time and robust lidar slam system based on CNN in dynamic urban environments," *IEEE Open J. Intell. Transp. Syst.* (2021).
[31] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality* (IEEE, 2007) pp. 225–234.
[32] T. Pire, R. Baravalle, A. D'alessandro and J. Civera, "Real-time dense map fusion for stereo slam," *Robotica* **36**(10), 1510–1526 (2018).
[33] C. Forster, M. Pizzoli and D. Scaramuzza, "SVO: Fast Semi-Direct Monocular Visual Odometry," *2014 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2014) pp. 15–22.
[34] R. A. Newcombe, S. J. Lovegrove and A. J. Davison, "DTAM: Dense Tracking and Mapping in Real-Time," *2011 International Conference on Computer Vision* (IEEE, 2011) pp. 2320–2327.
[35] J. Engel, T. Schöps and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," *European Conference on Computer Vision* (Springer, 2014) pp. 834–849.
[36] J. Zhang and S. Singh, "Visual-Lidar Odometry and Mapping: Low-Drift, Robust, and Fast," *2015 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2015) pp. 2174–2181.
[37] Y.-S. Shin, Y. S. Park and A. Kim, "DVL-SLAM: Sparse depth enhanced direct visual-lidar SLAM," *Auton. Rob.* **44**(2), 115–130 (2020).
[38] S.-S. Huang, Z.-Y. Ma, T.-J. Mu, H. Fu and S.-M. Hu, "Lidar-Monocular Visual Odometry Using Point and Line Features," *2020 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2020) pp. 1091–1097.
[39] J. Engel, V. Koltun and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal.* **40**(3), 611–625 (2017).
[40] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti and D. Rus, "LIO-SAM: Tightly-Coupled Lidar Inertial Odometry via Smoothing and Mapping," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2020) pp. 5135–5142.
[41] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Rob.* **37**(6), 1874–1890 (2021).
[42] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige and R. Siegwart, "Keyframe-based Visual-Inertial SLAM Using Nonlinear Optimization," *Proceedings of Robotis Science and Systems (RSS) 2013* (2013).
[43] T. Shan, B. Englot, C. Ratti and D. Rus, "LVI-SAM: Tightly-Coupled Lidar-Visual-Inertial Odometry via Smoothing and Mapping," *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021) pp. 5692–5698.
[44] D. Zermas, I. Izzat and N. Papanikolopoulos, "Fast Segmentation of 3d Point Clouds: A Paradigm on Lidar Data for Autonomous Vehicle Applications," *2017 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2017) pp. 5067–5073.
[45] Z. Yang, Y. Sun, S. Liu and J. Jia, "3DSSD: Point-based 3d Single Stage Object Detector," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020) pp. 11040–11048.
[46] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logist. Q.* **2**(1-2), 83–97 (1955).
[47] Baidu, Apolloauto (2022) https://github.com/ApolloAuto/apollo.
[48] K. Koide, M. Yokozuka, S. Oishi and A. Banno, "Voxelized GICP for Fast and Accurate 3d Point Cloud Registration," *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021) pp. 11054–11059.
[49] P. Geneva, K. Eckenhoff, Y. Yang and G. Huang, "Lips: Lidar-Inertial 3d Plane SLAM," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018) pp. 123–130.
[50] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision* (Cambridge University Press, UK, 2003).
[51] K. Madsen, H. B. Nielsen and O. Tingleff, Methods for non-linear least squares problems (2004).
[52] M. Henein, J. Zhang, R. Mahony and V. Ila, "Dynamic SLAM: The Need for Speed," *2020 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2020) pp. 2123–2129.
[53] A. Geiger, P. Lenz and R. Urtasun, "Are We Ready for Autonomous Driving? the Kitti Vision Benchmark Suite," *2012 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2012) pp. 3354–3361.
[54] T. Qin and S. Cao, A-loam: A lidar odometry and mapping (2019) https://github.com/HKUST-Aerial-Robotics/A-LOAM.
[55] H. Wang, C. Wang, C.-L. Chen and L. Xie, "F-loam: Fast Lidar Odometry and Mapping," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2021) pp. 4390–4396.
[56] R. Wang, M. Schworer and D. Cremers, "Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras," *Proceedings of the IEEE International Conference on Computer Vision* (2017) pp. 3903–3911.
[57] 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.* **23** (6), 5668–5677 (2021).
[58] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2012) pp. 573–580.
[59] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP J. Image Video Process.* **2008**(1), 1–10 (2008).
[60] T. Moore and D. Stouch, "A Generalized Extended Kalman Filter Implementation for the Robot Operating System," **In:** *Intelligent Autonomous Systems 13* (Springer, 2016) pp. 335–348.

Cite this article: T. Ma, G. Jiang, Y. Ou and S. Xu (2024). "Semantic geometric fusion multi-object tracking and lidar odometry in dynamic environment", Robotica **42**, 891–910. https://doi.org/10.1017/S0263574723001868