sensors

MDPI

Article

Sensor Fusion-Based Approach to Eliminating Moving Objects for SLAM in Dynamic Environments

Xiangwei Dang [1,2,†], Zheng Rong [3,†] and Xingdong Liang [1,2,*]

[1] National Key Laboratory of Microwave Imaging Technology, Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China; dangxiangwei16@mails.ucas.ac.cn
[2] School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China
[3] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; zheng.rong@nlpr.ia.ac.cn
* Correspondence: xdliang@mail.ie.ac.cn
† These authors contributed equally to this work.

Abstract: Accurate localization and reliable mapping is essential for autonomous navigation of robots. As one of the core technologies for autonomous navigation, Simultaneous Localization and Mapping (SLAM) has attracted widespread attention in recent decades. Based on vision or LiDAR sensors, great efforts have been devoted to achieving real-time SLAM that can support a robot's state estimation. However, most of the mature SLAM methods generally work under the assumption that the environment is static, while in dynamic environments they will yield degenerate performance or even fail. In this paper, first we quantitatively evaluate the performance of the state-of-the-art LiDAR-based SLAMs taking into account different pattens of moving objects in the environment. Through semi-physical simulation, we observed that the shape, size, and distribution of moving objects all can impact the performance of SLAM significantly, and obtained instructive investigation results by quantitative comparison between LOAM and LeGO-LOAM. Secondly, based on the above investigation, a novel approach named EMO to eliminating the moving objects for SLAM fusing LiDAR and mmW-radar is proposed, towards improving the accuracy and robustness of state estimation. The method fully uses the advantages of different characteristics of two sensors to realize the fusion of sensor information with two different resolutions. The moving objects can be efficiently detected based on Doppler effect by radar, accurately segmented and localized by LiDAR, then filtered out from the point clouds through data association and accurate synchronized in time and space. Finally, the point clouds representing the static environment are used as the input of SLAM. The proposed approach is evaluated through experiments using both semi-physical simulation and real-world datasets. The results demonstrate the effectiveness of the method at improving SLAM performance in accuracy (decrease by 30% at least in absolute position error) and robustness in dynamic environments.

Keywords: SLAM; dynamic environments; LiDAR; mmW-radar; sensor fusion; moving objects

Citation: Dang, X.; Rong, Z.; Liang, X. Sensor Fusion-Based Approach to Eliminating Moving Objects for SLAM in Dynamic Environments. *Sensors* **2021**, *21*, 230. https://doi.org/10.3390/s21010230

Received: 9 November 2020
Accepted: 27 December 2020
Published: 1 January 2021

1. Introduction

In recent decades, autonomous robots widely used in various fields such as urban warfare, rescue after disaster, autonomous driving and space robotics have attracted more and more attention. A crucial characteristic of an autonomous mobile robot is its ability to determine its whereabouts and make sense of its surrounding environments [1]. Simultaneous Localization and Mapping (SLAM) is a prerequisite for many robotic applications, which involves a system that simultaneously completes the positioning of the mobile robot itself and the map construction of the surrounding environment without any prior information [2]. Therefore, SLAM has been vigorously pursued in the mobile robot research field and various excellent algorithms have emerged. Generally, they can be divided

into vision-based SLAM and LiDAR-based SLAM. Visual SLAM such as MonoSLAM [3], DSO [4], ORB-SLAM [5] and VINS-Mono [6], where the primary sensor is the camera, can get rich information from the environment, but they are sensitive to texture richness and illumination. In contrast, LiDAR-based SLAM is widely used because it can acquire accurate and reliable distance information from the surrounding environment for state estimation. LiDAR-SLAM can be further divided into filter-based and optimization-based method, such as Gmapping [7], Hector SLAM [8], LOAM [9] and Cartographer [10] etc.

However, most of the popular LiDAR-SLAM frameworks deem the environment as motionless and ignore the influence of moving objects, while the assumption of the static environment can be easily violated in the real world, which will result in degradation of localization and mapping performance [11]. Due to the existence of dynamic objects, unexpected features will be extracted and used in the inter-frame matching and back-end optimization, which can introduce estimation error of robots' pose and surrounding map. Taking space robots as an example, in orbital or on planetary environments, the robots may cooperate with other agents and astronauts and encounter floating objects, which forms a challenging dynamic scenario.

To address this issue, SLAM algorithms specially for dynamic scenarios have been proposed. Some of these methods try to implicitly reduce the influence of dynamic objects using filtering or probability-based techniques [12]. The other methods explicitly detect the moving objects through various methods such as optical flow [13], deep learning [14–17], and then eliminate them from the raw sensor measurements to guarantee the accuracy of perception. However, these methods can only deal with the scenario with a few moving objects in good environmental conditions, otherwise the performance will be degraded or even fail.

In this paper, we first propose a semi-physical simulation method to quantitatively analyze the impact of moving objects on LiDAR-based SLAM performance in various dynamic scenarios. This method models the moving objects in the environment and generates echo according to the measurement model of LiDAR, and then integrates the measurements with the data measured in the real scene to achieve semi-physically simulated datasets in dynamic scenarios. We quantitatively evaluate the influence of the shape, quantity, speed, and distribution of the moving objects on the performance of LiDAR-based SLAM , towards providing theoretical background for the further proposed method of eliminating moving objects for SLAM.

Secondly, we propose a novel approach to eliminating the influence of dynamic objects named EMO (eliminating the moving objects) to improve the performance of SLAM by fusing mmW-radar and LiDAR. The LiDAR has a wide field of view (FOV) and can provide accurate distance and angle measurements with high resolution, but it is not straight forward to detect dynamic objects. On the opposite, mmW-radar can easily and directly provide velocity measurement of the objects using the Doppler effect, but it has a narrow FOV and low angular resolution. The fusion of the data from both sensors can thus benefit from their complementary [18]. The proposed method can effectively fuse the information from both sensors and precisely recognize the moving objects. The point clouds from LiDAR are segmented and at the same time the ghost targets in mmW-radar measurements are rejected by verification using LiDAR measurements. Then the segmented point clouds and mmW-radar measurements are associated with yield the precise volume and location of the moving objects. The actually moving objects are finally determined by compensating the Doppler-velocity from mmW-radar with sensor's velocity from real-time SLAM feedback, and eliminated from original point clouds. The filtered point clouds are used as the input of SLAM.

The proposed method is demonstrated by semi-physical simulation and experiments in the real world. The results show that the moving objects can be efficiently and precisely identified and rejected from point clouds in complex environment, which significantly improve the accuracy and robustness of the LiDAR-based SLAM. The main contributions of this paper are: (1) An effective semi-physical simulation method is proposed to analyze

the impact of dynamic scenarios on SLAM performance, taking into account the size, distribution, shape and speed of the objects. The quantitative results provide us a solid theoretical basis for the subsequent proposal of the moving objects elimination method. (2) A moving objects elimination method (EMO) is proposed to improve the performance of SLAM. This method effectively takes advantages of the mmW-radar and LiDAR to realize real-time detection, verification, and rejection of moving objects in environments for the purpose of improving the accuracy and robustness of localization and mapping. (3) The first system containing LiDAR and radar for moving target removal is built. Based on the methods of temporal synchronization and spatial calibration designed by us, the system can fuse the information of the two sensors at the front end to remove the dynamic target in real time, and then uses the filtered point cloud as the input of LiDAR-SLAM to improve the overall performance. (4) The proposed method is fully evaluated using semi-physically simulated datasets and real-world datasets in various scenarios to demonstrate the effectiveness and efficiency of the method.

The rest of the paper is organized as follows: Section 2 reviews related works. Section 3 introduces our method in detail, including semi-physical simulation and moving objects elimination. Section 4 shows the experiments details and results. Finally, Section 5 concludes our work.

2. Related Works

It is well acknowledged that the existence of moving objects in environments can impact the performance of SLAM to some degree. However, there are only a few pieces of research work on the evaluation and investigation of this influence. Pancham et al. [19] evaluate the performance of ORB-SLAM with an RGB-D camera in a dynamic environment including an object moving at a range of specific linear speeds. Experiments show that a moving object at lower speeds degrades the performance of ORB-SLAM, and removing the moving object can improve the performance of ORB-SLAM. Lu Z et al. [20] describe and compare three different approaches of SLAM in dynamic outdoor environments. They proposed a probabilistic SLAM with random sampling consensus(RANSAC) estimation, which shows better performance in the noisy environment to build the occupancy grid maps in their experiment. Roesler O et al. [21] evaluate four different 2D SLAM algorithms available in Robot Operating System (ROS) and find that Hector Mapping achieves the best performance in dynamic scenarios. These studies evaluate the performance of SLAM in dynamic environments, but most of them simply compare some algorithms by building simple scenarios. The correlation and quantitative analysis between dynamic objects and SLAM performance are not given. This is mainly because the data collection of datasets in various dynamic scenarios is difficult, time-consuming and laborious, while semi-physical simulation can solve this problem.

To mitigate the influence of dynamic objects on the SLAM performance, a variety of SLAM algorithms in dynamic environment have been developed. These methods can be generally divided into two categories. The first category of methods does not detect the dynamic objects directly, but implicitly reduces the influence of moving objects using filtering or probability-based techniques. The second category of methods try to explicitly recognize the moving objects using methods such as optical flow, deep learning [14–17], and then eliminates them to guarantee the accuracy of estimation.

Kitt et al. [22] and Tan et al. [23] propose to use RANSAC method to eliminate mismatched information caused by moving objects to improve odometer accuracy. Hhnel D et al. [24] present a new approach that interleaves mapping and localization with probabilistic technique to identify spurious measurements. Bibby C et al. [25] combine the least-squares formulation of SLAM and sliding window optimization together with generalized expectation maximization, to incorporate both dynamic and stationary objects directly into SLAM estimation. Probabilistic grid map [12] is used to decrease the influence of dynamic objects on map construction. Huber norms are also widely used during optimization to mitigate the degradation caused by the dynamic outliers. However, these

methods will fail in the scenario with many dynamic objects because they are not able to explicitly recognize the moving objects.

Some dynamic SLAM algorithms based on dynamic objects detection are also proposed [13,26–29], trying to precisely detect and reject the dynamic objects to improve the performance of SLAM system, while moving objects detection under dynamic background becomes a new challenge. X Zhang et al. [1] incorporate the sensor information of a monocular camera and laser range finder to remove the feature outliers related to dynamic objects to enhances the accuracy of the localization. Wangsiripitak and Murray [30] try to reject moving outliers by tracking known 3D dynamic objects. Moo et al. [31] use two single Gaussian models that can effectively represent the foreground and background to find the moving targets. Sun et al. [32] extended the concept of intensity difference image to identify the boundaries of dynamic objects, and use a motion removal approach as a preprocessing stage to filter out the data associated with moving objects. Zhao H et al. [33] use GPS data and control inputs to diagnose pose error and classify objects with moving object detection and tracking.

Deep learning [14–17]-based methods are also used to detect the object for SLAM recently. Advanced convolutional neural network (CNN) architectures such as YOLO [34], SSD (Single Shot multibox Detector) [35], SegNet [36], and Mask R-CNN [37] can effectively detect the labels of the objects in a scene. Han and Xi [38] propose PSPNet-SLAM (Pyramid Scene Parsing Network-SLAM) to improve ORB-SLAM2, where PSPNet and the optical flow are used to detect dynamic features. DS-SLAM [14] combines semantic segmentation network with moving consistency checking method to reduce the impact of dynamic objects, and thus improve the localization accuracy in dynamic environments. DynaSLAM [15], built on ORB-SLAM2, has the capabilities of dynamic object detection and background inpainting, and the resulting accuracy outperforms the standard visual SLAM baselines in highly dynamic scenarios. Semantic information is also used to detect the dynamic objects. Yang S et al. [16] present a semantic and geometric constrained method SGC-VSLAM, which is built on the RGB-D version of ORB-SLAM2 with the addition of dynamic object detection and static map construction to filter out outliers. Xieyuanli Chen et al. [17] propose SuMa++ to reduce the influence of moving objects. By directly performing semantic segmentation on the 3D point cloud, potential dynamic targets in the environment, such as cars, etc., are recognized and removed to construct a static semantic map. Henein M et al. [39] propose a new feature-based, model-free, object-aware dynamic SLAM algorithm that exploits semantic segmentation to enable robust estimation of motion of robot. However, the above learning-based dynamic detection methods have some drawback. These methods cannot be generalized to all kinds of moving objects in the real world, and the detected dynamic targets are not necessarily moving actually [40]. They also require a lot of training work, and consume considerable computation to achieve real-time performance. Furthermore, most of these methods are based on LiDAR or vision, and will fail in challenging environments such as poor lighting, rain and fog, etc. However, our method (EMO) directly obtains the real moving target in the environment through the measurement of mmW-radar, without semantic cues or prior knowledge and is suitable for any unknown rigid objects. To the best of our knowledge, this is the first dynamic SLAM system that combines of radar and LiDAR for moving targets removal.

3. Methodology

In this section, we first present the method of quantitative analysis to investigating the impact of objects with different dynamic patterns on SLAM performance through semi-physical simulation. Secondly, we propose an effective method to eliminating the influence of moving objects (EMO) on SLAM based on multi-sensor fusion. The investigation result in the first part provides a solid theoretical and empirical background knowledge for the second part of work.

3.1. Semi-Physical Simulation for Evaluation of LiDAR-Based SLAM

To precisely investigate the influencing factors of dynamic environments on the performance of SLAM, datasets in the environments with various dynamic patterns must be collected. Semi-physical simulation is used in this work to precisely control the appearance of moving objects in the environment. First, typical moving objects in the real world are modeled as two types of elements with specified size, shape, quantity and motion model; then according to the measurement model of LiDAR used in the experiment (here is VLP-16 or HDL-64), the echo data of simulated objects is generated and integrated with the data corresponding to the static environment collected in the real world. We use these synthetic datasets to evaluate the state-of-the-art SLAM algorithms and our methods. The semi-physical simulation is depicted in Figure 1.

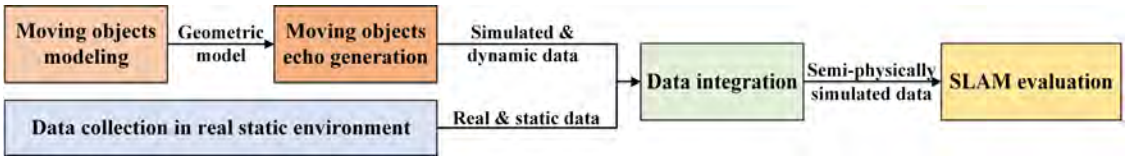

Figure 1. Semi-physical simulation-based LiDAR-SLAM evaluation.

3.1.1. Moving Objects Modeling

In different robot application areas, the type of dynamic objects in the environment is different, but in most scenarios, the moving objects are mainly vehicles and pedestrians. Here, we use cuboids with specified length, width and height to simulate vehicle in the environment and cylinders to simulate pedestrians. The reason we choose two different geometric shapes is that we found different SLAM algorithms are sensitive to different geometric information because they use different methods to do the frame matching. The SLAM algorithms such as LOAM and LeGO-LOAM are feature-based methods, where the feature is extracted by evaluating the curvature of the point and are sensitive to the existence of cuboids. However, algorithm such as Cartographer use grid map-based method, in which all the points are considered. Without losing generality and rationality for the evaluation, we use cuboids and cylinder in our simulation, as shown in Figure 2a.

We denote the world coordinate frame as W, the sensor coordinate frame as L, and the robot body frame as B. We assume that the pose of the cube in the sensor coordinate frame is $P_c^L(x_o, y_o, z_o, \theta_o)$, where θ_o is the heading of the cuboid, i.e., the angle between the x-axis of cuboid and the x-axis of sensor, as shown in Figure 2b. Similarly, the center of the cylinder represented in sensor frame is denoted as $P_h^L(x_o, y_o, z_o)$.

Since the goal of simulation is to get the echo of moving objects measured by 3D LiDAR, for the simplicity of representation and implementation, we analyze the echo of a single-channel LiDAR first. We project the cuboid and cylinder onto the XY plane of the sensor frame, yielding a rectangle and a circle. The length and width of rectangle are l_c and w_c, and its center point is $P_c^L(x_o, y_o, \theta_o)$. The radius of the circle is r_h, and its center is $P_h^L(x_o, y_o)$.

To correctly generate the sensor echo from simulated objects, the boundary of the models in sensor's FOV need to be determined according to their poses in the sensor coordinate frame, as shown in Figure 2b. For a rectangle, according to the specified center point P_c^L and heading angle θ_o, its four vertices $P_{c\_1}^L(x_1, y_1)$, $P_{c\_2}^L(x_2, y_2)$,$P_{c\_3}^L(x_3, y_3)$, $P_{c\_4}^L(x_4, y_4)$ can be determined by Equation (1). Then the angle of each vertex with respect to the sensor frame and their maximum $\varphi_{c\_\max}^L$ and minimum $\varphi_{c\_\min}^L$ can be determined, consequently the corresponding vertices $P_{c\_\min}^L(x_{\min}, y_{\min})$, $P_{c\_\max}^L(x_{\max}, y_{\max})$ can be used to determine the boundary of the object observed by the sensor in the current pose. In particular, if two sides of the object can be seen, we also need to find the middle vertex $P_{c\_mid}^L(x_{mid}, y_{mid})$, and then the echo data can be calculated by Equations (2) and (3). Finally,

according to the vertical resolution and FOV of the LiDAR, the echo data can be easily extended to 3D.

$$\begin{aligned}(x_1, y_1) &= \left(x_o + \frac{w_c}{2}\sin\theta_o + \frac{l_c}{2}\cos\theta_o, y_o + \frac{w_c}{2}\cos\theta_o - \frac{l_c}{2}\sin\theta_o\right)\\ (x_2, y_2) &= \left(x_o - \frac{w_c}{2}\sin\theta_o + \frac{l_c}{2}\cos\theta_o, y_o - \frac{w_c}{2}\cos\theta_o - \frac{l_c}{2}\sin\theta_o\right)\\ (x_3, y_3) &= \left(x_o - \frac{w_c}{2}\sin\theta_o - \frac{l_c}{2}\cos\theta_o, y_o - \frac{w_c}{2}\cos\theta_o + \frac{l_c}{2}\sin\theta_o\right)\\ (x_4, y_4) &= \left(x_o + \frac{w_c}{2}\sin\theta_o - \frac{l_c}{2}\cos\theta_o, y_o + \frac{w_c}{2}\cos\theta_o + \frac{l_c}{2}\sin\theta_o\right)\end{aligned} \tag{1}$$

$$\begin{cases}(y_{\max} - y_{\min})x + (x_{\min} - x_{\max})y + x_{\max}y_{\min} - x_{\min}y_{\max} = 0\\ y = kx, k \in \left(\tan\varphi^L_{c\_\min}, \tan\varphi^L_{c\_\max}\right)\end{cases} \tag{2}$$

$$\begin{cases}x = {}^{(x_{\min}y_{\max} - x_{\max}y_{\min})}/_{(y_{\max} - y_{\min} + kx_{\min} - kx_{\max})}, k \in \left(\tan\varphi^L_{c\_\min}, \tan\varphi^L_{c\_\max}\right)\\ y = kx = {}^{k(x_{\min}y_{\max} - x_{\max}y_{\min})}/_{(y_{\max} - y_{\min} + kx_{\min} - kx_{\max})}\end{cases} \tag{3}$$

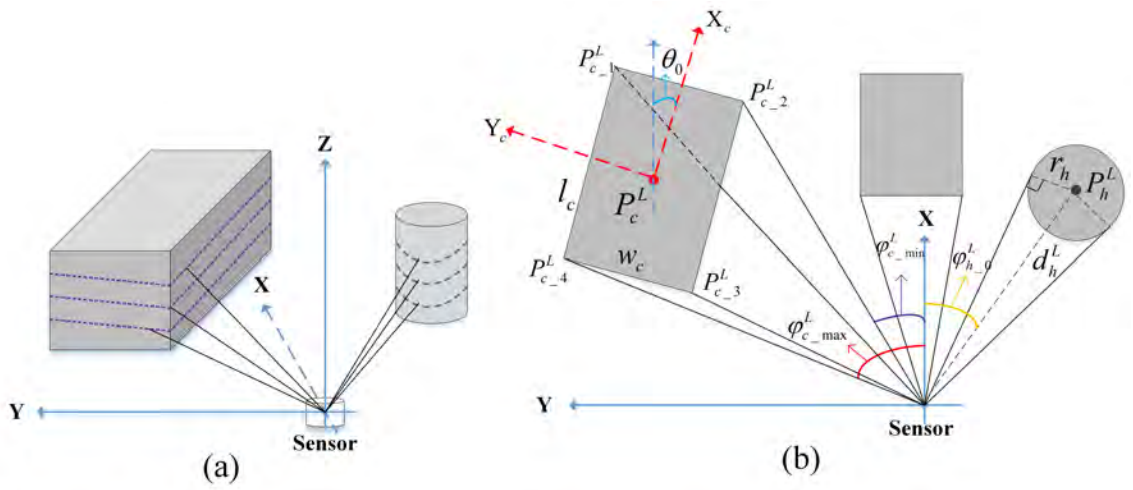

Figure 2. The geometric models used for dynamic objects simulation, cuboids and cylinder, are scanned by LiDAR (**a**). According to their poses $P^L_c(x_o, y_o, \theta_o)$ and $P^L_h(x_o, y_o)$ in the sensor coordinate frame, the LiDAR measurements can be simulated respectively (**b**), which is calculated in XY plane of the sensor frame. We denote the world coordinate frame as W, the sensor coordinate frame as L, and the robot body frame as B. Depending on the position and shape of the object, one or two sides of the object can be detected. By evaluating the boundary of the models in sensor's FOV the echo data can be calculated.

For a circle, according to the coordinates of circle center, the distance d^L_h and angle $\varphi^L_{h\_o}$ of the center point in sensor frame can be determined, and thus the echo occupied by the circle can be calculated using Equations (4) and (5). This equation has two sets of solutions, and we choose the solution set with the smallest Euclidean distance.

$$\begin{cases}(x - x_o)^2 + (y - y_o)^2 = r_h^2\\ y = kx, k \in \left(\tan\left(\varphi^L_{h\_o} - a\tan\frac{r_h}{d^L_h}\right), \tan\left(\varphi^L_{h\_o} + a\tan\frac{r_h}{d^L_h}\right)\right)\end{cases} \tag{4}$$

$$\begin{cases} x = \dfrac{2(x_o + ky_o) \pm \sqrt{4(x_o + ky_o)^2 - 4(1+k^2)\left(x_o^2 + y_o^2 - r_h^2\right)}}{2(1+k^2)}, \\ y = kx = \dfrac{2k(x_o + ky_o) \pm k\sqrt{4(x_o + ky_o)^2 - 4(1+k^2)\left(x_o^2 + y_o^2 - r_h^2\right)}}{2(1+k^2)} \\ k \in \left(\tan\left(\varphi_{h\_o}^L - a\tan\dfrac{r_h}{d_h^L}\right), \tan\left(\varphi_{h\_o}^L + a\tan\dfrac{r_h}{d_h^L}\right)\right) \end{cases} \tag{5}$$

As mmW-radar is used in the elimination stage, the simulation of the radar is also required in the datasets. First, we assume that the radar data has been synchronized and calibrated. Due to the position of the simulated moving object is known when we generate the LiDAR dataset, accordingly we can use the centroid of the object as the echo point from radar, represent with the object's distance, angle and speed. Considering the ranging accuracy of the radar, we also add random noise to the radar data for realistic simulation.

3.1.2. Datasets Generation

After modeling the single moving object, in this subsection we detail the generation of semi-physically simulated datasets by simulating multiple objects with specified motion models and integrating the echo data with real static data.

Based on the knowledge of dynamics and kinematics [41], we specify a motion pattern for each single moving object, including the initial pose, linear and angular velocity, then the pose of each object at any time point can be obtained. In particular, we deal with the problem of mutual occlusion between the multiple moving targets, towards getting the simulated datasets consistent with the actual sensor measurement model. We evaluate the distance and occupied FOV of all the objects in the sensor frame, and try to find the overlaps among the objects. If any coincidence is found, we recalculate the echo data in the sensor FOV with occlusion. To integrate the simulated data with the real measurements, the sensor FOV occupied by the moving objects is calculated, and the real measurements in this area are substituted using the simulated data. By far we complete the semi-physical simulation of one frame. The complete datasets can be synthesized in this manner frame by frame. The process can be depicted as Figure 3.

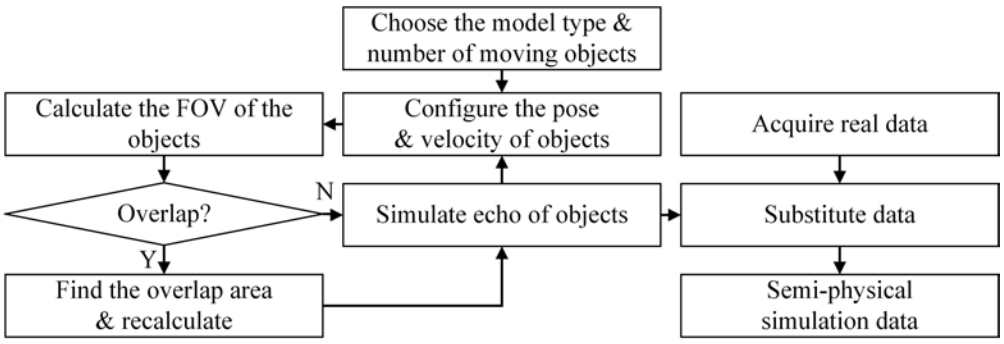

Figure 3. Generation of semi-physically simulated datasets. The real measurement data is partially substituted by the echo data from simulated moving objects.

3.2. Moving Objects Elimination for LiDAR-SLAM

The existence of moving objects in the environment will corrupt the assumption of static environment for SLAM, which will significantly degrade the performance of localization and mapping. We propose a method using multi-sensor fusion to eliminate the influence of dynamic environments on the accuracy and robustness of robot state estimation. The system diagram is depicted in Figure 4.

First, temporal synchronization and spatial calibration [42] between LiDAR and mmW -radar are requisite step for accurate multi-sensor fusion. For timing synchronization,

we use an ARM-based embedded hardware system to precisely synchronize the clock of LiDAR and mmW-radar. The hardware system is based on STM32F103 which uses the Cortex-M3 core, with a maximum CPU speed of 72 MHz. The hardware system will generate Pulse Per Second (PPS) signal to the LiDAR, and at the same time, the processor reads the radar data from the CAN bus to ensure that the LiDAR and radar data are measured simultaneously. For spatial calibration, due to the limited measurement accuracy and low resolution of the mmW-radar, we directly use CAD data to determine the relative translation and rotation between two sensors. Since the angular resolution and ranging accuracy of the radar are far lower than that of the LiDAR, the method based on CAD design and careful mounting can meet the accuracy requirement of sensor fusion of the two sensors.

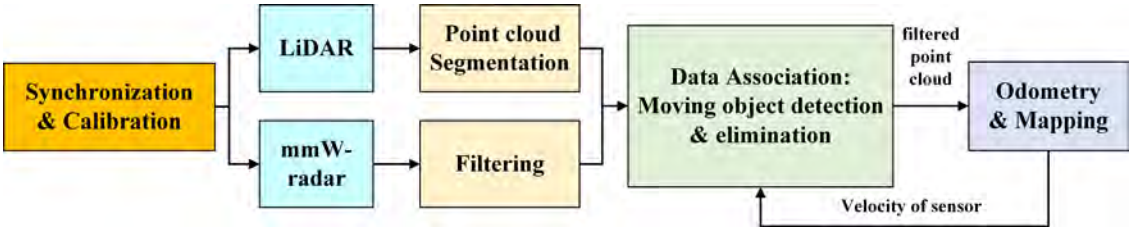

Figure 4. The system overview of moving objects elimination.

After having two kinds of sensor data synchronized in time and space, we preprocess the measurement data from LiDAR and mmW-radar, mainly including the segmentation of LiDAR point clouds to identify objects and filtering of mmW-radar detection to reject ghost targets. Then we associate the preprocessed data from two sensors. Based on the measurement of Doppler effect by radar and velocity feedback from SLAM, we can precisely determine the actually moving objects with reference to the world. At the same time, the precise volume and location of the targets can be obtained by associating the LiDAR measurements. Finally the detected moving objects can be rejected from point clouds and these filtered data are used as the input of odometry and mapping modules in the LiDAR-based SLAM framework.

3.2.1. Point Cloud Segmentation

Since the LiDAR provides dense point clouds representing the surroundings without any texture and clustering information, it is necessary to segment the point clouds first for fusing LiDAR measurements and the targets detected by mmW-radar, i.e., finding the correspondence between point clusters and radar targets.

In the segmentation, we first find the ground plane from point clouds using the similar method with [43]. Then an image-based segmentation method [44] is used to identify and separate the objects in the point clouds. The point clouds are organized into a range image with the size of vertical points number multiplied by LiDAR channel number. Whether a cluster of LiDAR points are corresponded to the same object can be determined by checking the angle between two vectors on the range image, and the points from the same object will be labeled identically.

The point clouds are projected onto a distance image first. As shown in Figure 5, A and B are two adjacent points on the distance image, OA and OB are distance vector directly measured by the LiDAR, and φ is the angle between these two vectors, which can be calculated according to the horizontal and vertical resolution and the position of two points in the image. Then we can calculate the angle β using Equation (6), and determine whether the two points are from the same object: the smaller the angle β and the greater the distance $\|AB\|$between the two points, the less likely the two points are from the same object. By setting the threshold of the angle (10 degrees in our test), the point clouds corresponding to the same object can be identified. For example, in Figure 5b, though the distance in the three sets of evaluated point-pair is similar, we can still correctly cluster the point clouds by furtherly checking the angle β between the two points. This

segmentation approach is computing efficient and easy to implement, compared with the learning-based method. A failure case can be a situation in which the scanned object is planar, such as a wall, and oriented nearly parallel to the laser beams which result in an over-segmentation. Despite this shortcoming, our experiments suggest that the method is effective in practice. The aforementioned scenes occur rarely and if so, it usually results only in an over-segmentation of particularly inclined planar objects.

$$\beta = \text{atan2}(\|BH\|, \|HA\|) = atan2(d_2 \sin\varphi, d_1 - d_2 \cos\varphi) \quad (6)$$

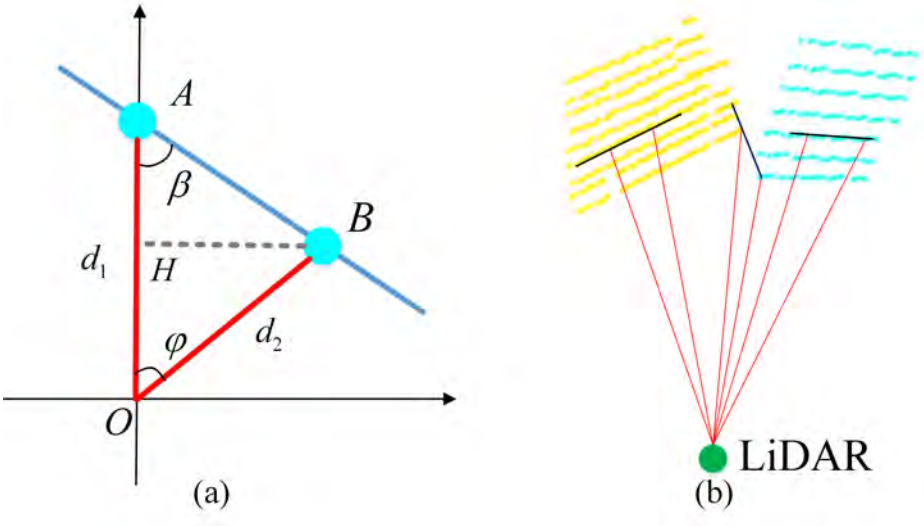

Figure 5. The schematic diagram of segmentation. (**a**) Whether two points are corresponded to the same object can be determined by evaluating the angle β. (**b**) shows an example.

3.2.2. Radar Results Filtering

Due to the propagation characteristics and multipath effect of electromagnetic waves, the detection results from mmW-radar are usually polluted by some non-target noise. To remove the negative impact of the false alarm, a two-stage filtering method is proposed.

In the preliminary filtering state, we use the valid flag included in the data frame of mmW-radar to remove invalid measurements. At the same time, according to the performance characteristics of the radar, some thresholds are set in terms of detection range, angle and speed to eliminate singular values.

However, some ghost targets cannot be removed by the pre-filtering. Benefiting from the accurate measurements of surrounding environment from LiDAR, a verification strategy is furtherly proposed to identify the ghost targets. Based on the sensor calibration, the radar target points can be transformed to the LiDAR coordinate frame. As shown in Figure 6, a KD tree is built from point clouds, and we try to search K nearest neighbors around each radar point within a preset radius (which is determined by ranging accuracy of the radar. 0.5m was used in this paper.) and compute the centroid of all the LiDAR points with the same label from the segmentation results. The radar point is valid only if the neighboring point number is above a threshold and the distance between radar point and the centroid is small enough, otherwise considered to be a ghost target. This method can effectively make use of the advantages of accurate target detection of LiDAR and effectively remove the radar false targets caused by multipath effect.

Through the two-stage filtering, the invalid measurements and false targets from the mmW-radar can be effectively reduced, which provides precise and reliable data for the following data association procedure.

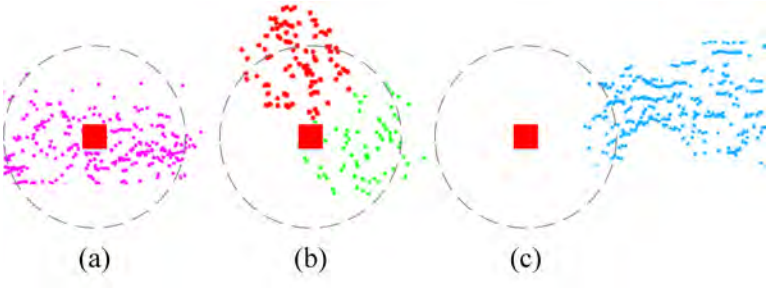

Figure 6. Radar measurement filtering using point cloud-based verification. (**a**) The mmW-radar target (red square) is valid as there are enough LiDAR points around it and the centroid of the segment is close enough to the radar target. (**b**) Two segmented objects are considered corresponding to the radar target. (**c**) Ghost target as the LiDAR point number in the specified radius is small and the point cluster is far from the target.

3.2.3. Data Association

The mmW-radar can directly obtain the velocity information of the detected targets in the environment, while LiDAR can obtain the accurate distance and direction information of the surrounding. By associating the data, dynamic targets in the environment can be accurately detected, identified and removed from point clouds to reduce the impact of moving objects on SLAM.

The velocity measurement of mmw-radar is mainly based on the Doppler effect, i.e., if the detected target is relatively moving with respect to the radar, the frequency of echo wave will be different from the frequency of the emitted wave. By detecting this frequency difference, the relative moving speed of the target can be calculated according to Equation (7).

$$v = \frac{cf_\Delta}{2f_o} \tag{7}$$

where f_o is the working frequency of mmW-radar, c is the speed of light, and f_Δ is the frequency difference measured by radar. Then the moving speed of the target relative to the radar can be obtained as v.

Since the mmW-radar can measure the relative speed of the target only in radial direction with reference to the moving sensors, motion compensation must be performed to get the target velocity in world frame and then identify the actually moving targets.

For each target, the Doppler-velocity v_{tg}^L is measured by mmW-radar. For simplicity of expression, we assume that LiDAR and mmW-radar are in the same coordinate frame based on the aforementioned sensor calibration. By integrating the sensor velocity v_L^W from SLAM feedback, the absolute velocity v_{tg}^{sta} of the target in the current static frame can be estimated as

$$v_{tg}^{sta} = R_W^L v_L^W + v_{tg}^L \tag{8}$$

where R_W^L is inverse of the current orientation of the sensor in world frame. Thus, the actual moving status of the target can be determined.

Due to the limitation of mmW-radar measurements in resolution, density, accuracy, etc., the targets obtained by the radar can only be expressed as a few of sparse points in 2D, whereas the LiDAR can provide a bunch of accurate 3D dense points to represent the target. Thus, finding the accurate correspondence between sparse radar points and LiDAR point clouds is critical for the data fusion. We propose a data association algorithm based on 3D-to-2D projection and K neighborhood searching, as shown in Algorithm 1. In particular, the moving targets in our application mainly move on the ground plane, such as vehicles and pedestrians, so we assume that there can only be one target in any specified x-y 2D position, which makes it reasonable to fuse two-dimensional mmW-radar data with three-dimensional LiDAR data.

Algorithm 1 Data Association

Input P^{seg}, P^{rad}, C, D

Output P^{sta}

P^{seg} is projected onto radar plane $\rightarrow P^{seg'}$
$P^{seg'} \rightarrow$ KD tree
for $p_j^{rad} \in P^{rad}$ **do**
  KNN search($P^{seg'}$, p_j^{rad}, R_{search}) $\rightarrow P^{nb}$
  for $P_k^{nb} \in P^{nb}$ **do**
    Count point number in $P_k^{nb} \rightarrow c$
    Compute centroid of all points in $P_k^{seg'} \rightarrow e_k$
    Compute distance between p_j^{rad} and $e_k \rightarrow d$
    if $c > C$ && $d < D$ **then**
      $P^{seg} = P^{seg} - P_k^{seg}$
    end if
  end for
end for
$P^{seg} \rightarrow P^{sta}$

Let $P^{seg} = \left\{ P_1^{seg}, P_2^{seg}, ..., P_n^{seg} \right\}$ be the segmented points from a LiDAR sweep, where P_i^{seg} is a subset of the point clouds with the same segmentation label i . Let $P^{rad} = \left\{ p_1^{rad}, p_2^{rad}, ..., p_n^{rad} \right\}$ be the detected targets after velocity compensation by mmW-radar. P^{seg} and P^{rad} are the inputs of the algorithm. First, the LiDAR point cloud P^{seg} is projected onto the x-y plane, denoted by $P^{seg'} = \left\{ P_1^{seg'}, P_2^{seg'}, ..., P_n^{seg'} \right\}$, and stored in a KD tree. For each radar target p_j^{rad} , the LiDAR points neighbors around it within radius R_{search} are searched and collected as a point cloud P^{nb}. Then we analyze all the points in P^{nb}. Let P_k^{nb} be the points with label k within P^{nb}. If the number of points within P_k^{nb} is above the threshold C and the distance between centroid of segmentation $P_k^{seg'}$ and the radar point is below the threshold D, then the segmented LiDAR points P_k^{seg} is considered to be some correspondence to the radar target p_j^{rad}, i.e., a moving target, and then all the points labeled with k will be removed from the point clouds. Finally, the algorithm outputs a filtered point cloud P^{sta} corresponding to the environment excluding any moving object.

As shown in Figure 7a, there are a stationary truck and several moving cars on the road. Although both the truck and car are detected as moving targets by radar, by velocity compensation the stationary truck and moving cars can be precisely distinguished, as shown in Figure 7b. In addition, finally the actually moving car is removed from the point clouds as Figure 7d by data association. The proposed data association method can effectively detect the relatively moving objects, and with velocity compensation the absolutely moving objects can be further identified and removed. From the test, the feasibility and accuracy of data association between two sensors with different resolutions are proved.

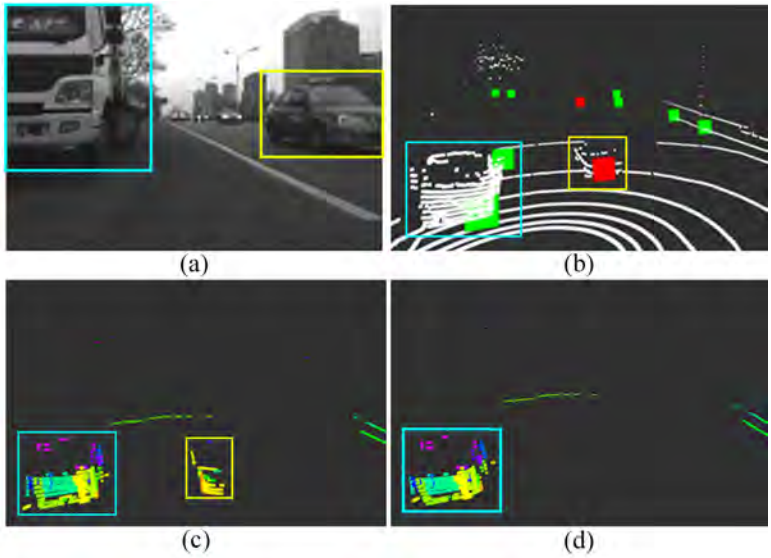

Figure 7. Moving targets detection and removal test. (**a**) Test scenario, two targets are detected by radar. (**b**) Stationary truck (green) and moving cars (red) are distinguished by radar after velocity compensation. (**c**) Segmentation result of point cloud. (**d**) Filtered point cloud.

4. Experiments and Results

The datasets used for the experiments include open-source KITTI [45] datasets and custom datasets collected using a wheeled robot equipped with a LiDAR (VLP-16) and the mmW-radar (Delphi ESR), as shown in Figure 8. VLP-16 is a 16-channel range finder with detection range of 100 m and FOV of $30^\circ \times 360^\circ$. The radar provides two measurement modes simultaneously, including a wide FOV of 90° at mid-range and a small FOV of 20° at long-range, but only outputs 2D position of the targets in horizontal scanning plane. A camera is mounted to record the scenario. All the algorithms are tested on an Intel NUC computer (i7-5557U CPU @ 3.10 GHz and 8.0 GB RAM) with ROS.

We conducted a series of experiments using two open-source algorithm, LOAM [9] and LeGO-LOAM [43]. Both algorithms can realize localization and 3D mapping based on LiDAR solely. We use these algorithms as a baseline to evaluate the influence of moving objects in the performance of LiDAR-SLAM , and demonstrate the evident improvement of the proposed moving object elimination method.

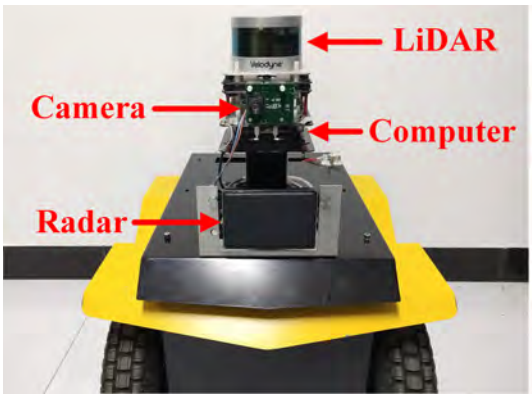

Figure 8. The wheeled robot used in our experiments, equipped with a LiDAR, a mmW-radar, a camera and a computer.

4.1. Evaluation of the State-of-the-Art LiDAR SLAMs in Dynamic Environments

In this section, we evaluate the performance of LiDAR-based SLAM in various scenarios with different dynamic patterns through semi-physical simulation. To quantitatively analyze the results, we calculate the root mean square error (RMSE) of the absolute position

error in meters, representing the accuracy performance of the algorithm. RMSE is defined in Equation (9).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\hat{p}_i - p_i\right)^2} \tag{9}$$

where n is the number of estimates, $\hat{p}_i = \left(\hat{x}_i, \hat{y}_i, \hat{z}_i\right)$ is the ith estimate, and $p_i = (x_i, y_i, z_i)$ is the corresponding ground truth.

To conduct a comprehensive analysis, we simulated five different dynamic patterns based on six real datasets, totally got 30 semi-physical simulated datasets. The first four dynamic patterns are: one car, two cars, four people, four cars moving in the environment but relatively static with the robot. In the fifth pattern four cars are moving in the environment with random speed and relatively moving with the robot. The diagram of the five different dynamic patterns is shown in Figure 9. By comparing (a), (b) and (d), we can analyze the influence of number of dynamic objects on performance of SLAM. By comparing (b) and (c), we can analyze the impact of shape of the objects on SLAM performance. The influence of distribution on SLAM can be obtained by comparing (d) and (e). Through our analysis, we found that both the quantity and the distribution are closely related to the proportion of the moving objects in the total point cloud. Therefore, we evaluate the SLAM performance directly with respect to the proportion of the moving objects that calculated by Equation (10). The six real datasets used here include two KITTI datasets (sequence 2011_09_26_0095 and 2011_09_30_0027) and four custom datasets (flowerbed, research buildings, campus and road).

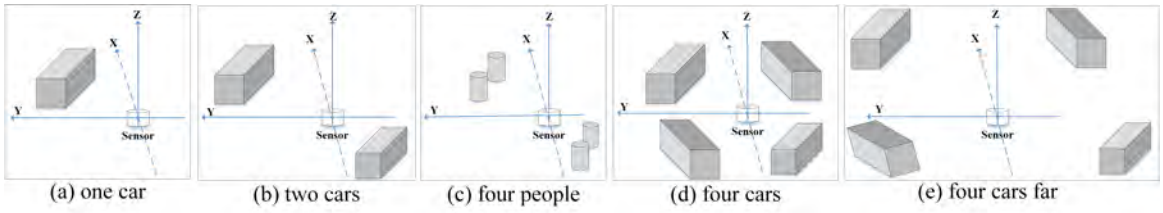

Figure 9. The diagram of the five different dynamic patterns used in semi-physical simulation. By comparing (**a**,**b**,**d**), we can analyze the influence of number of dynamic objects on performance of SLAM. By comparing (**b**,**c**), we can analyze the impact of shape of the objects on SLAM performance. The influence of distribution on SLAM can be obtained by comparing (**d**,**e**).

$$ratio = \frac{1}{n}\sum_{i=1}^{n}\frac{N\_sim}{N\_sum} \tag{10}$$

where n is the number of LiDAR sweep, N_sim is the point number of simulated moving objects and N_sum is total number of points in this sweep. The proportion of moving objects in the five scenarios of six datasets are shown in Figure 10.

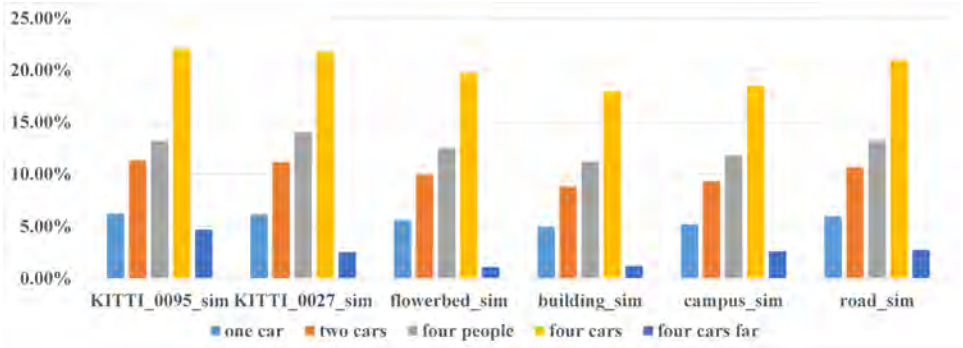

Figure 10. The moving objects proportion of the five scenarios in the six datasets of semi-physical simulation.

4.1.1. Evaluation on KITTI Datasets

We choose two sequences from KITTI datasets, KITTI 2011_09_26_0095 and KITTI 2011_09_30_0027 since there are few moving objects in these original datasets; thus we can simulate the dynamic targets based on the static environment. We test Cartographer, LOAM and LeGO-LOAM on the semi-physically simulated data. The mapping results are shown in Figures 11 and 12 and the odometry results are shown in Figures 13a,b,d,e and 14. Basically, more objects will result in worse estimation result, including large odometry drift and mapping inconsistency. In particular, compared with pedestrian we found that cars have much bigger influence on the mapping performance, and even result in total failure in some scenarios.

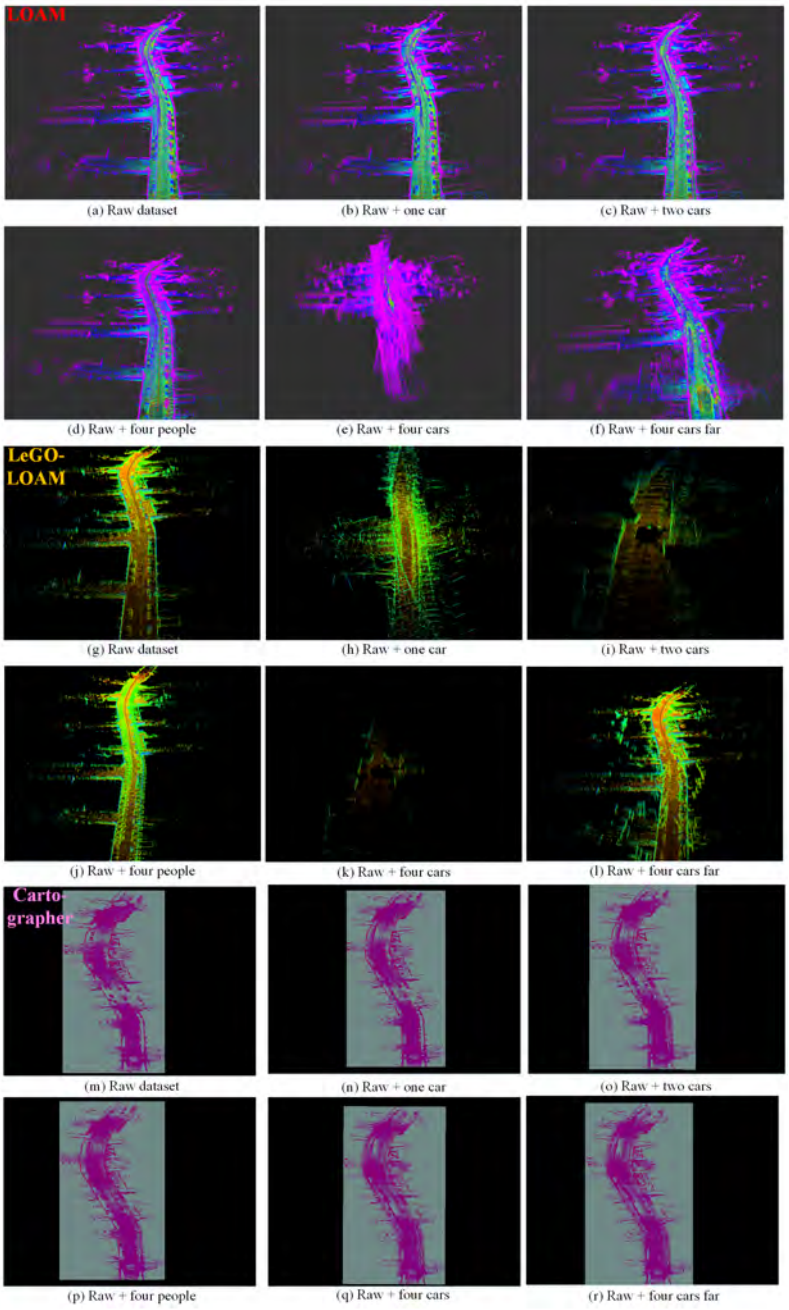

Figure 11. The mapping results (3D view) of LOAM (**a**–**f**), LeGO-LOAM (**g**–**l**) and Cartographer (**m**–**r**) in different simulated scenarios based on KITTI_0095_sim. (**a**,**g**,**m**) are the results on the original static data, used as baseline. More dynamic objects result in worse mapping results, and even total failure in some scenarios such as (**e**,**h**,**i**,**k**). LeGO-LOAM is more sensitive to moving objects in this test.

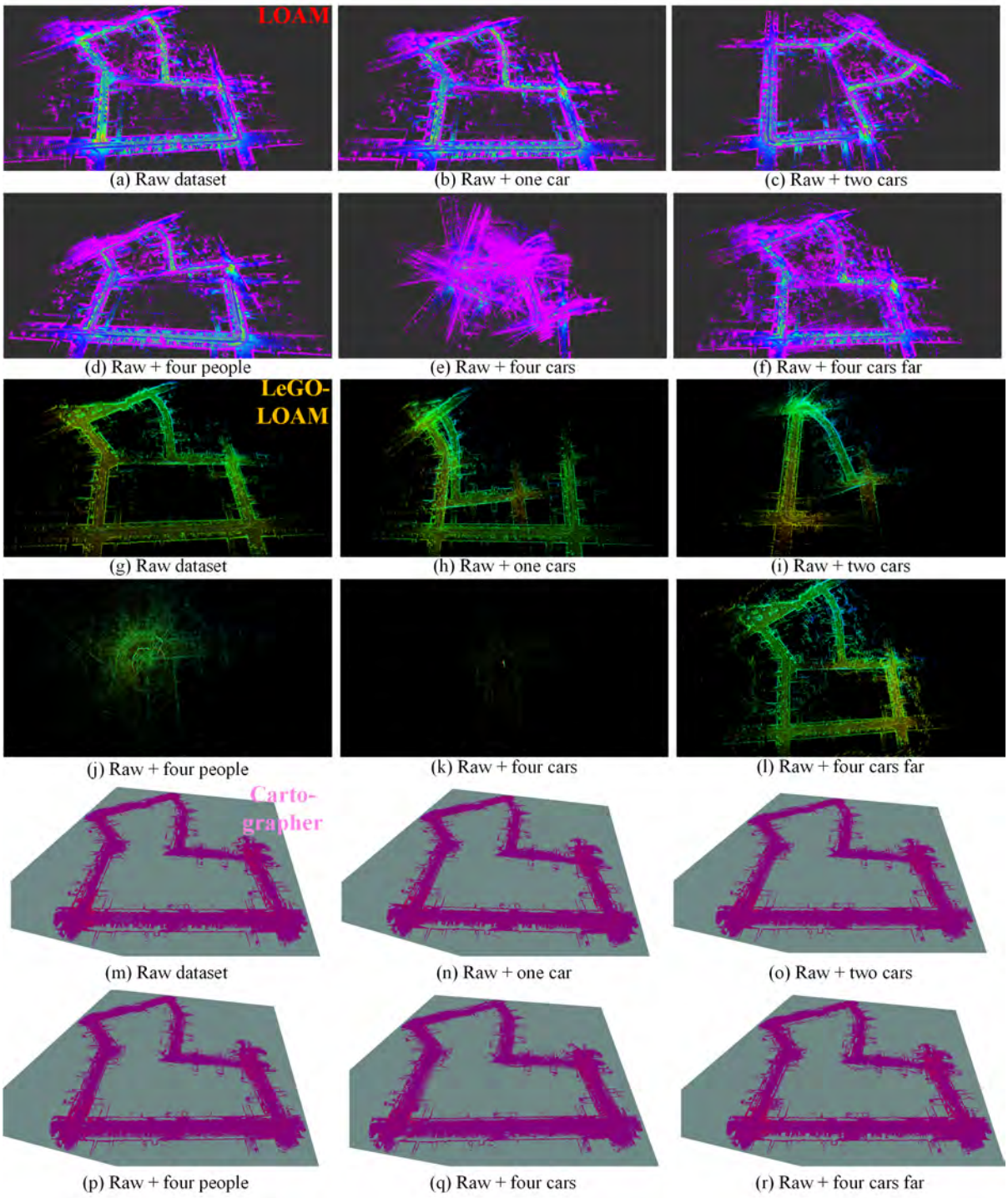

Figure 12. The mapping results (3D view) of LOAM (**a**–**f**), LeGO-LOAM (**g**–**l**) and Cartographer (**m**–**r**) in different simulated scenarios based on KITTI_0027_sim. (**a**,**g**,**m**) are the results on the original static data, used as baseline. Please note that in (**c**) estimation error occurs at the beginning, thus results in the deflected map. More dynamic objects result in worse mapping results, and even total failure in some scenarios such as (**e**,**h**,**i**–**k**). In this test Cartographer shows relatively more robust performance.

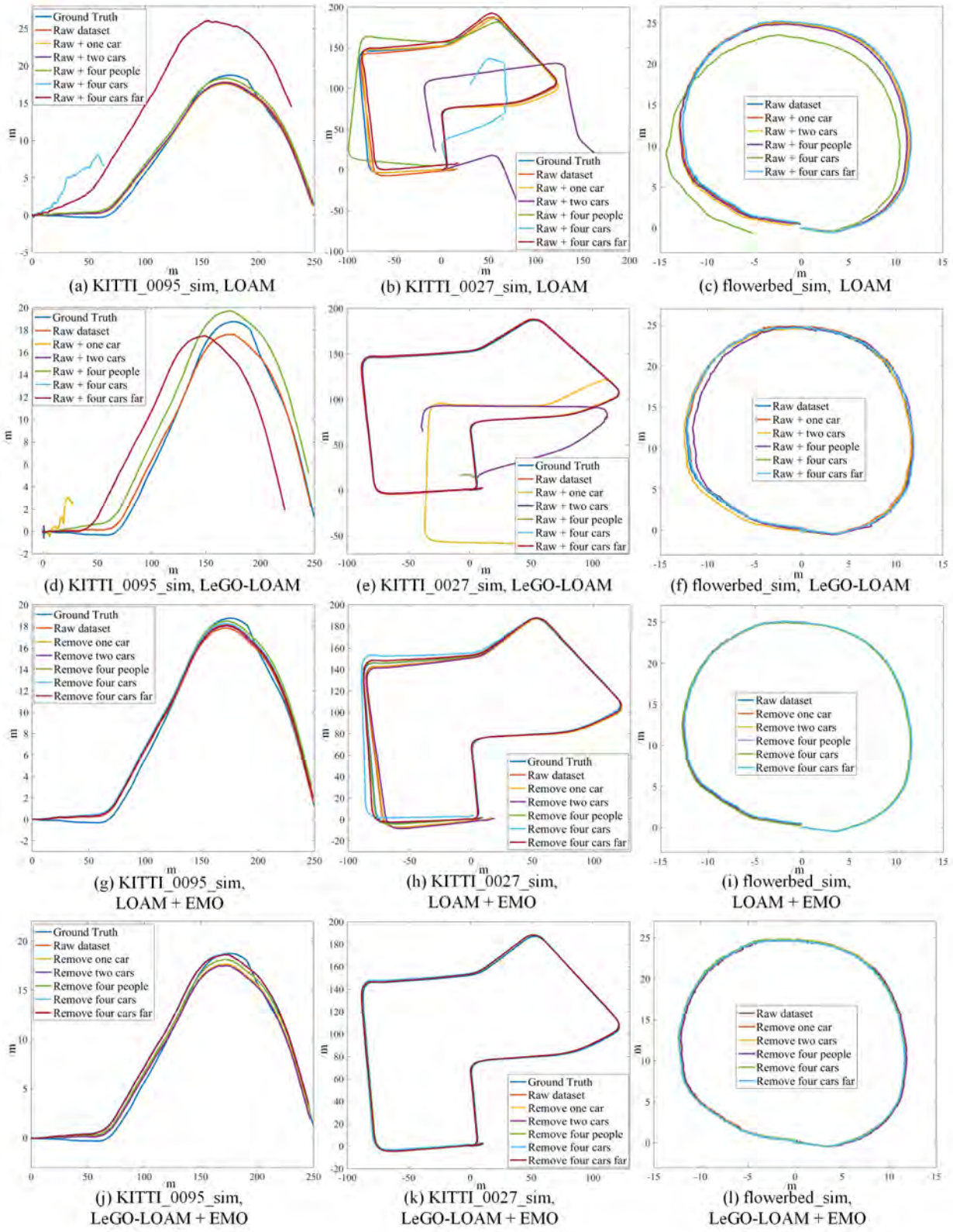

Figure 13. The odometry results of [original LOAM] (**a**–**c**), [original LeGO-LOAM] (**d**–**f**), [LOAM + EMO] (**g**–**i**) and [LeGO-LOAM + EMO] (**j**–**l**) in different simulated scenarios based on KITTI_0095_sim, KITTI_0027_sim and flowerbed_sim. Before removing the dynamic objects, more dynamic objects result in worse odometry drift or even estimation failure (**a**–**f**). After removing the moving objects with our method, the two algorithms yield evidently improved odometry accuracy in all the scenarios (**g**–**l**).

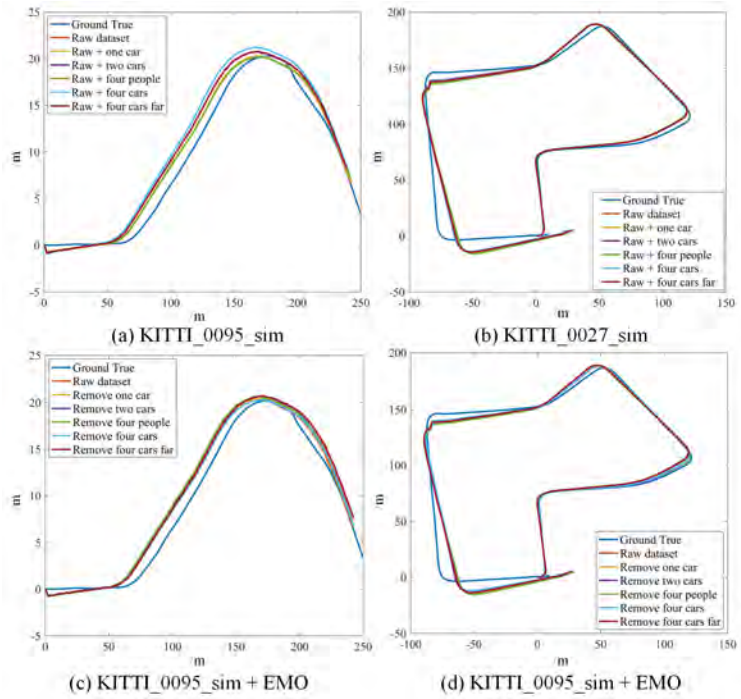

Figure 14. The odometry results of [Cartographer] (**a**,**b**) and [Cartographer + EMO] (**c**,**d**) based on KITTI_0095_sim and KITTI_0027_sim. After removing the moving objects with our method, it yields evidently improved odometry accuracy in all the scenarios.

Furtherly we compute RMSE of the absolute position error for quantitative evaluation of the odometry results, as shown in the first two rows of Table 1. The bold number indicate obvious odometry drift or failure. From the result, we can see that more moving objects will cause worse estimation results. Again, we can notice that different object models have different effects on the performance, i.e., LOAM and LeGO-LOAM are more sensitive to the cars than pedestrian. The reason for this phenomenon is that we use cuboid to model the car and it provides intense edge and plain structure features to the algorithm, which are just the feature information used in LOAM and LeGO-LOAM. On the contrary, the cylinder model for pedestrian perform smaller effect in this aspect.

4.1.2. Evaluation on our Datasets

We collected a set of datasets using wheeled robot in static environment and use the same semi-physical simulation method to generate four datasets with different dynamic scenarios: flowerbed, building, campus, road. The odometry and mapping results of LOAM and LeGO-LOAM on dataset flowerbed are shown in Figures 13c,f and 15. The odometry of both algorithms drift to some degree when moving objects exist in the environment. Due to limited space, the results of datasets building, campus and roads are given quantitatively only, in the last four rows of Table 1.

The quantitative analysis results of RMSE of position error are shown in the last four rows of Table 1. Please note that the ground truth is not available in our datasets, so we use the odometry results of baseline, i.e., the result on the raw data without dynamic objects as ground truth to compute RMSE. The obvious performance degradation or failure is indicated in bold. We can clearly see that more moving objects in the environment will cause worse estimation result and both algorithms are more sensitive to the cars because cuboids provide intense edge and plain features to the algorithms.

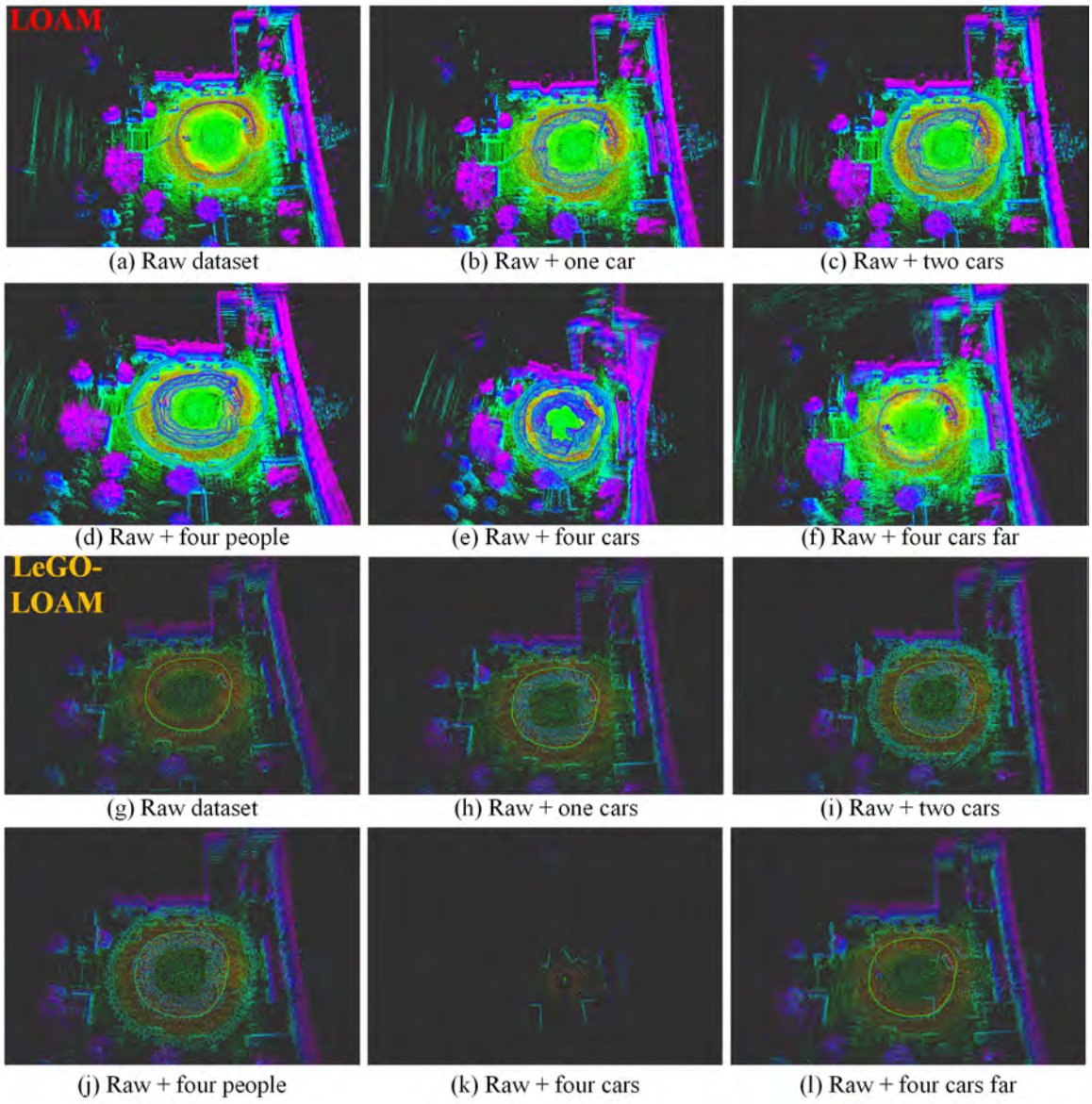

Figure 15. The mapping results (3D view) of LOAM (**a**–**f**) and LeGO-LOAM (**g**–**l**) in different simulated scenarios based on flowerbed_sim. (**a**,**g**) are the results on the original static data, used as baseline. More dynamic objects result in more obvious mapping error such as (**e**), or even total failure in some scenarios such as (**k**).

Through the semi-physical simulation-based experiments with different scenarios in different datasets, we can draw the following conclusions: (1) The presence of dynamic objects does affect the perceptual positioning and mapping performance of robots, and this effect is related to many factors such as the shape, quantity, speed and distribution of moving objects. (2) For the same type of moving objects, the more the proportion of the moving objects in the LiDAR sweep is, the worse the estimation result of odometry and map will be. The proportion is affected by the quantity and distribution (distance) of the objects. (3) For the different types of moving objects, LOAM and LeGO-LOAM are more sensative to the cars that modeled using cuboid, which means moving cars can introduce more estimation error than pedestrian. This is because cuboid provide intense edge and plain features, which is just LOAM and LeGO-LOAM used for scan matching. For Cartographer, we found that cars and pedestrians perform similar influence on the estimation result because Cartographer uses a grid map-based method in which all the points are considered instead of feature points only. (4) The objects' speed can also make a

difference. When the moving objects remain relatively static with the robot, the influence of the dynamic objects is greater.

Table 1. The absolute position RMSE (unit: m) of odometry results from Cartographer, LOAM and LeGO-LOAM in six semi-physically simulated datasets with and without our proposed method (EMO). The bold number indicate obvious odometry drift or failure.

Datasets	Algorithm	One Car	Two Cars	Four People	Four Cars	Four Cars Far
KITTI_0095_sim	LOAM	0.5047	0.8424	0.7843	**108.8954**	**19.7693**
	LOAM + EMO	0.3430	0.3618	0.2680	0.4940	0.2551
	LeGO-LOAM	**126.4614**	**140.5702**	3.0814	**141.1223**	**23.4166**
	LeGO-LOAM + EMO	0.1271	0.1973	0.4182	1.9146	1.7046
	Cartographer	0.9408	2.0061	1.8029	2.7668	1.4945
	Cartographer + EMO	0.8899	0.9732	1.1103	1.2149	1.3603
KITTI_0027_sim	LOAM	4.9643	**145.0990**	19.3609	**109.1039**	6.7535
	LOAM + EMO	2.1389	3.8326	2.7545	5.8555	2.6443
	LeGO-LOAM	**59.9155**	**112.3563**	**117.0891**	**125.8396**	1.3541
	LeGO-LOAM + EMO	0.7182	1.1032	0.8759	1.4460	0.3776
	Cartographer	2.3954	2.4832	3.4827	2.5367	1.1656
	Cartographer + EMO	1.7730	2.3029	2.7699	2.1686	1.1010
flowerbed_sim	LOAM	0.3081	0.6780	0.5914	**3.0507**	0.2593
	LOAM + EMO	0.3079	0.3760	0.1974	0.5848	0.2689
	LeGO-LOAM	1.1677	**2.2230**	1.3041	**16.3797**	0.4063
	LeGO-LOAM + EMO	0.3416	0.4321	0.3888	0.2391	0.0845
building_sim	LOAM	1.2965	1.9941	1.4872	**14.5741**	1.8203
	LOAM + EMO	0.5387	1.2813	1.1456	0.8995	1.3091
	LeGO-LOAM	1.9142	**8.0904**	3.1911	**34.2463**	1.7784
	LeGO-LOAM + EMO	0.5049	0.9626	0.7467	1.7590	0.4377
campus_sim	LOAM	0.5859	0.8080	0.6600	**6.2356**	0.2192
	LOAM + EMO	0.0698	0.1199	0.0790	0.2645	0.0337
	LeGO-LOAM	0.4945	**21.3248**	**21.1645**	**21.7350**	0.1855
	LeGO-LOAM + EMO	0.0812	0.1069	0.1419	0.2419	0.1569
road_sim	LOAM	0.5011	**8.2676**	**7.8464**	**41.0379**	2.1865
	LOAM + EMO	0.4941	0.5275	0.4359	4.7984	0.1683
	LeGO-LOAM	6.8698	**25.1214**	**19.1836**	**44.0201**	5.1829
	LeGO-LOAM + EMO	0.1534	4.9771	4.7344	5.1979	4.9042

4.2. LiDAR-SLAM with Proposed Moving Objects Elimination

In this section, we evaluate and demonstrate the performance improvement of LiDAR-SLAM with our proposed moving objects elimination method, using both semi-physically simulated datasets and real-world datasets.

4.2.1. On Semi-Physically Simulated Datasets

Semi-physically simulated datasets are used here to verify the effectiveness of the proposed method. As mmW-radar is used in the elimination stage, the radar-detected targets are also required in the datasets. Due to the position of the simulated moving object is known when we generate the dataset, accordingly we simulate the mmW-radar targets with specified noise at the same time. Thus, the datasets including point clouds and radar targets are ready for the experiments.

In the proposed method, the point clouds are segmented and moving targets are detected and verified using LiDAR and radar, then the point clouds corresponding to the dynamic objects are removed and filtered point clouds are used as the input of the LOAM and LeGO-LOAM. The odometry results of LOAM and LeGO-LOAM on two KITTI

datasets and one custom dataset are depicted in Figure 13g–l to representatively show the effectiveness of our method. Compared with Figure 13a–f we can clearly see the significant improvement on the accuracy of the state estimation for all the six scenarios in the three datasets. Consequently, the mapping results are also precise, and will not be shown in this paper.

Also, we quantitatively analyzed the position error for all the six datasets, as shown in Table 1. Compared with Table 1, the absolute position error is significantly reduced even in highly dynamic environment, which provides a solid evidence of the effectiveness of the proposed method.

4.2.2. On Real-World Dynamic Datasets

We further demonstrate the proposed method using three real-world datasets collected in dynamic environments with our wheeled robot. In particular, we equipped the robot with a high-precision inertial and GNSS-based navigation system to get the ground truth in these experiments.

In the first test, the robot moves along a side road with some vehicles and pedestrians moving on the road. We compare the odometry and mapping results of LOAM and LeGO-LOAM with and without the moving objects elimination, as shown in Figures 16a and 17a–d. From the results it can be seen that before removing dynamic targets the odometry seriously diverged from the ground truth. The moving objects made an effect on the inter-frame registration, which leads to the positioning error and consequential mapping degradation. However, after removing dynamic objects using our method the estimator yields remarkable improvement including pose accuracy (position error decreased by at least 84.9%) and mapping consistency.

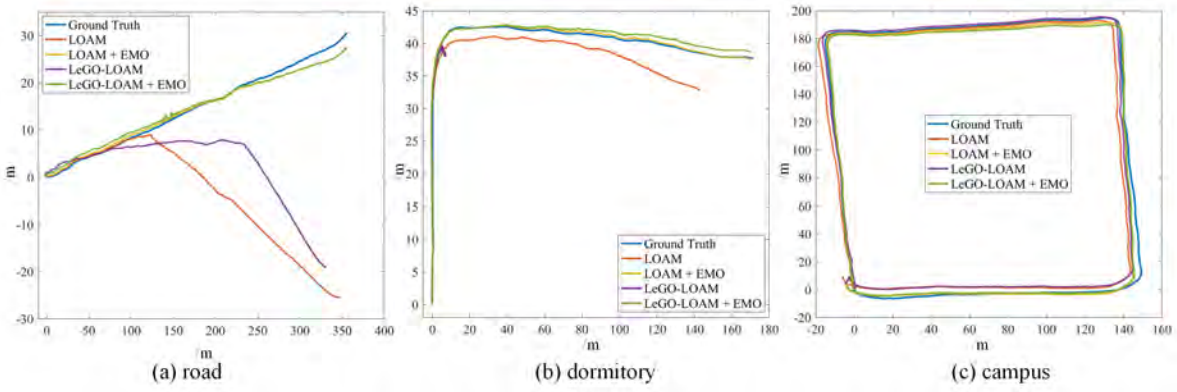

Figure 16. The odometry results of three datasets: straight road (**a**), dormitory park (**b**) and campus (**c**).

In the second test, the robot moves in a dormitory park with several rows of buildings, with some pedestrians walking around. The results of odometry and mapping are depicted in Figures 16b and 17e–h. Before removing dynamic objects, LeGO-LOAM totally failed. With our method, the degradation of state estimation is significantly mitigated (position error decreased by at least 90%) .

In the last test we extend our experiment in a large scenario. The robot moves around a campus with a few of moving targets and returns to the starting point. The odometry result is shown in Figure 16c. By investigating the quantitative result in Table 2, we can still tell the minor improvement of accuracy (position error decreased by at least 30% after using our method, though there only exist a few dynamic objects in the experiment.

The three corresponding quantitative results expressing the position RMSE are shown in Table 2, in which the positioning error significantly decreased (position error decreased by at least 30%) after using our method to remove the dynamic objects.

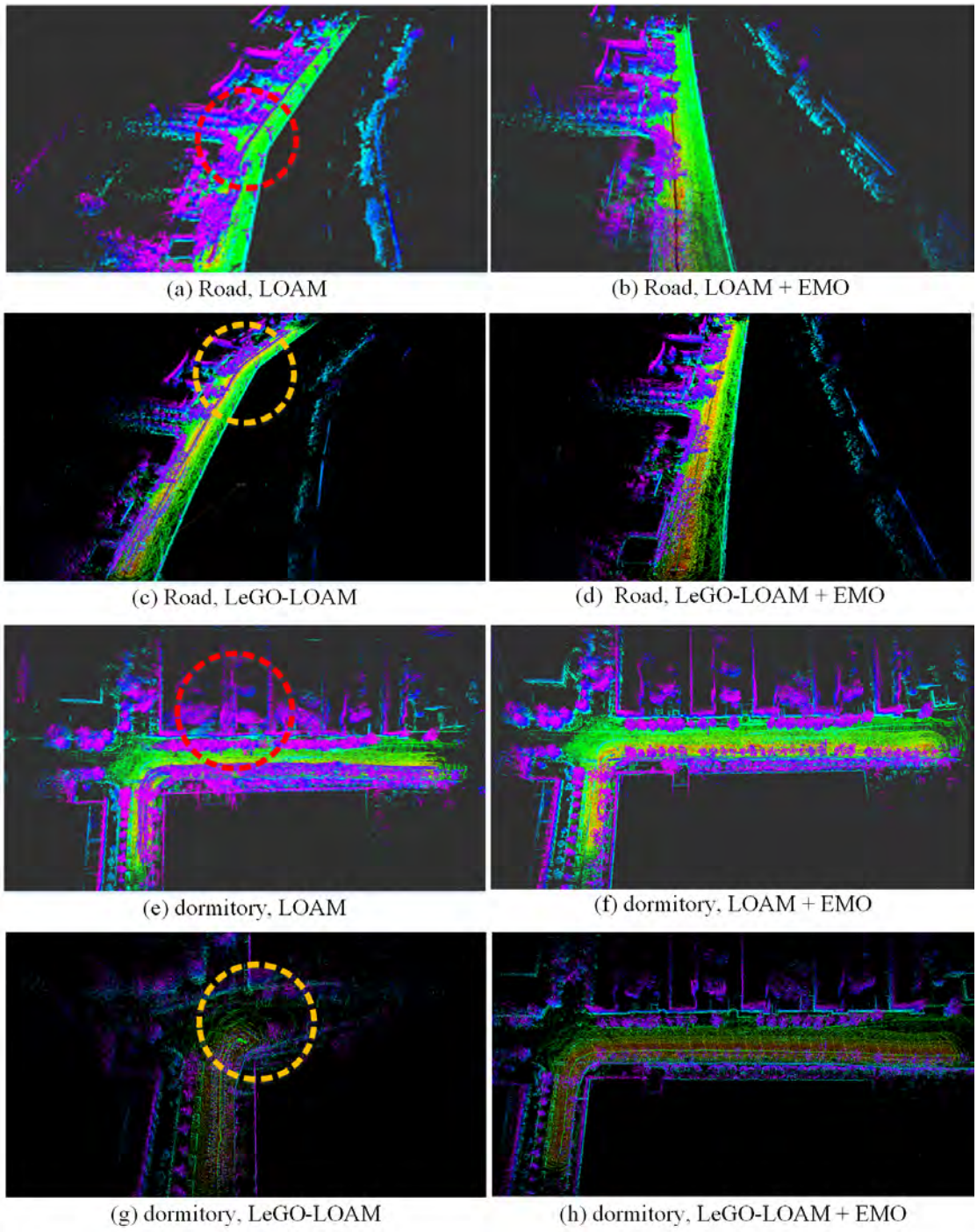

Figure 17. The mapping results (3D view) of dataset road (**a**–**d**) and dormitory (**e**–**h**). (**a**,**c**,**e**,**g**) are the maps built by LOAM and LeGO-LOAM before removing dynamic objects, in which the significant mapping errors are marked using circles. (**b**,**d**,**f**,**h**) are the improved mapping results using our method (EMO) of moving object elimination, showing better map consistency and sharper detail.

From the simulated and real-world experiments, we can clearly see that benefiting from the well-designed sensor fusion and data association, the proposed method can effectively improve the performance of the LiDAR-based SLAM on accuracy and robustness even in highly dynamic environments, without increasing computation significantly.

Table 2. The position RMSE (unit: m) of LOAM and LeGO-LOAM in three real-world dynamic tests with and without the moving objects elimination (EMO). The bold number indicate obvious improvement and smaller error. % indicates the percentage of error reduction.

	LOAM	LOAM + EMO	LeGO-LOAM	LeGO-LOAM + EMO
Road	111.7965	**3.9360** (96.48%)	29.5669	**4.4634** (84.90%)
Dormitory	15.2370	**1.3472** (91.16%)	80.5292	**0.9339** (98.84%)
Campus	5.7304	**3.0851** (46.16%)	4.9835	**3.5135** (29.50%)

5. Conclusions

The existence of moving objects will affect the accuracy of localization and consistency of mapping to varying degrees, and subsequently degrade the reliability of autonomous vehicles. To quantitatively analyze the impact of dynamic objects on the state-of-the-art SLAM performance, we propose a semi-physical simulation method to generate datasets with different dynamic patterns by integrating real-world static data and simulated dynamic data. From the analysis result we found that the impact of dynamic environment is related to the shape, number, speed and distribution of moving objects, which provide a theoretical an empirical background for the following proposed moving object elimination method.

Benefiting from the complementary characteristics of mmW-radar and LiDAR, we propose a method to detect, identify and eliminate efficiently and precisely the moving objects by well-designed sensor fusion and data association, towards enhancing the performance of dynamic SLAM in the real world. At the same time, the temporal synchronization and spatial calibration between the sensors are carefully designed to ensure effective sensor fusion. Through the semi-physical simulation and real-world experiments the effectiveness and performance of the proposed method are demonstrated and evaluated, showing significant improvement in the odometry accuracy (position error decreased by at least 30%), mapping consistency and estimation robustness, without increasing much computing. The proposed method can also be applied to space robots, in orbital or on planetary environments, in which multi-agent robots, astronauts, and other floating objects may exist and form a challenging dynamic scenario.

The focus of this paper is to fuse the information of the two sensors at the front end to remove the dynamic target, and then use the filtered point cloud as the input of LiDAR-SLAM to improve the overall performance, which is a loosely coupled method to fuse the LiDAR and mmW-radar. To fully take the advantage of laser and microwave sensors, our future work involves integrating high-resolution 3D mmW-radar and developing a tightly coupled SLAM using LiDAR and mmW-radar to further improve the robustness and accuracy of localization and mapping. For example, the information of the reflection intensity provided by the radar can be used in frame matching and provide more stable constraints in global optimization to get better results.

Author Contributions: Conceptualization, X.D. and Z.R.; methodology, X.D. and Z.R.; software, X.D.; validation, X.D. and Z.R.; formal analysis, X.D., Z.R. and X.L.; investigation, X.D.; resources, X.L.; data curation, X.D. and Z.R.; writing—original draft preparation, X.D.; writing—review and editing, Z.R, X.L. and X.D.; visualization, X.D.; supervision, Z.R. and X.L.; project administration, Z.R.; funding acquisition, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Nation Natural Science Foundation of China under Grant No. 62002359.

Acknowledgments: The authors thank the editor and anonymous reviewers for their helpful comments and valuable suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, X.; Rad, A.B.; Wong, Y.K. Sensor Fusion of Monocular Cameras and Laser Range finders for Line-Based Simultaneous Localization and Mapping (SLAM) Tasks in Autonomous Mobile Robots. *Sensors* **2012**, *12*, 429–452. [CrossRef] [PubMed]
2. Vidal, A.R.; Rebecq, H.; Horstschaefer, T.; Scaramuzza, D. Ultimate SLAM? Combining Events, Images, and IMU for Robust Visual SLAM in HDR and High-Speed Scenarios. *IEEE Robot. Autom. Lett.* **2018**, *3*, 994–1001. [CrossRef]
3. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067 [CrossRef] [PubMed]
4. Engel, J.; Koltun, V.; Cremers, D. Direct Sparse Odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 611–625. [CrossRef] [PubMed]
5. MurArtal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
6. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
7. Grisetti, G.; Stachniss, C.; Burgard, W. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Trans. Robot.* **2007**, *23*, 34–46. [CrossRef]
8. Kohlbrecher, S.; Stryk, O.V.; Meyer, J.; Klingauf, U. A flexible and scalable SLAM system with full 3D motion estimation. In Proceedings of the 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics, Kyoto, Japan, 1–5 November 2011.
9. Zhang, J.; Singh, S. Low-drift and real-time lidar odometry and mapping. *Auton. Robot.* **2017**, *41*, 401–416. [CrossRef]
10. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-Time Loop Closure in 2D LIDAR SLAM. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
11. Bresson, G.; Alsayed, Z.; Yu, L.; Glaser, S. Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving. *IEEE Trans. Intell. Veh.* **2017**, *2*, 194–220. [CrossRef]
12. Li, L.; Yao, J.; Xie, R.; Tu, J.; Chen, F. Laser-Based Slam with Efficient Occupancy Likelihood Map Learning for Dynamic Indoor Scenes. *Isprs Ann. Photogramm. Remote. Sens. Spat. Inf.* **2016**, *3*, 119–126. [CrossRef]
13. Alcantarilla, P.F.; Yebes, J.J.; Almazán, J.; Bergasa, L.M. On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In Proceedings of the Robotics and Automation (ICRA), 2012 International Conference on IEEE, Saint Paul, MN, USA, 14–18 May 2012.
14. Yu, C.; Liu, Z.; Liu, X.; Xie, F.; Yang, Y.; Wei, Q.; Fei, Q. DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1168–1174.
15. Bescos, B.; Facil, J.M.; Civera, J.; Neira, J. DynaSLAM: Tracking, Mapping, and Inpainting in Dynamic Scenes. *IEEE Robot. Autom. Lett.* **2018**, *3*, 4076–4083 [CrossRef]
16. Yang, S.; Fan, G.; Bai, L.; Li, R.; Li, D. MGC-VSLAM: A Meshing-based and Geometric constraint VSLAM for Dynamic Indoor Environments. *IEEE Access* **2020**, *8*, 81007–81021. [CrossRef]
17. Chen, X.; Milioto, A.; Palazzolo, E.; Giguère, P.; Behley, J.; Stachniss, C. SuMa++: Efficient LiDAR-based Semantic SLAM. In Proceedings of the Intelligent Robots and Systems (IROS), Macau, China, 4–8 November 2019.
18. Göhring, D.; Wang, M.; Schnürmacher, M.; Ganjineh, T. Radar/Lidar sensor fusion for car-following on highways. In Proceedings of the 5th International Conference on Automation, Robotics and Applications, ICARA, Wellington, New Zealand, 6–8 December 2011.
19. Pancham, A.; Withey, D.; Bright, G. *Evaluation of a Simultaneous Localization and Mapping Algorithm in a Dynamic Environment Using a Red Green Blue—Depth Camera*; Springer: Singapore, 2018.
20. Lu, Z.; Hu, Z.; Uchimura, K. Slam Estimation in Dynamic Outdoor Environments. *Int. J. Humanoid Robot.* **2010**, *7*, 315–330. [CrossRef]
21. Roesler, O.; Ravindranath, V.P. Evaluation of SLAM Algorithms for Highly Dynamic Environments. In *Fourth Iberian Robotics Conference*; Springer: Cham, Switzerland, 2020.
22. Kitt, B.; Geiger, A.; Lategahn, H. Visual odometry based on stereo image sequences with RANSAC-based on outlier rejection scheme. In Proceedings of the IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 21–24 June 2010; pp. 486–492.
23. Tan, W.; Liu, H.; Dong, Z.; Zhang, G.; Bao, H. Robust monocular SLAM in dynamic environments. In Proceedings of the 12th IEEE/ACM International Symposium on Mixed and Augmented Reality, Adelaide, Australia, 1–4 October 2013; pp. 209–218.
24. Hahnel, D.; Triebel, R.; Burgard, W.; Thrun, S. Map Building with Mobile Robots in Dynamic Environments. In Proceedings of the 2003 IEEE International Conference on Robotics and Automation, Taipei, Taiwan, 14–19 September 2003.
25. Bibby, C.; Reid, I.D. Simultaneous localization and mapping in dynamic environments (SLAMIDE) with reversible data association. In *Robotics: Science & Systems Iii, June, Georgia Institute of Technology*; DBLP: Atlanta, GA, USA, 2007.
26. Lee, K.H.; Hwang, J.N.; Okopal, G.; Pitton, J. Ground-Moving-Platform-Based Human Tracking Using Visual SLAM and Constrained Multiple Kernels. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 3602–3612. [CrossRef]
27. Bakkay, M.C.; Arafa, M.; Zagrouba, E. Dense 3D SLAM in Dynamic Scenes Using Kinect. In Proceedings of the 7th Iberian Conference IbPRIA, Pattern Recognition and Image Analysis, Lecture Notes in Computer Science, Santiago de Compostela, Spain, 17–19 June 2015.

28. Wang, Y.; Huang, S. Towards dense moving object segmentation based robust dense RGB-D SLAM in dynamic scenarios. In Proceedings of the Control Automation Robotics Vision(ICARCV), 2014 13th International Conference on IEEE, Singapore, 10–12 December 2014.
29. Lin, K.H.; Wang, C.C. Stereo-based simultaneous localization, mapping and moving object tracking. In Proceedings of the Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on IEEE, Taipei, Taiwan, 18–22 October 2010.
30. Wangsiripitak, S.; Murray, D.W. Avoiding moving outliers in visual SLAM by tracking moving objects. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; Institute of Electrical and Electronics Engineers (IEEE): Hoboken, TX, USA, 2009; pp. 375–380.
31. MooYi, K.; Yun, K.; Wan Kim, S.; Jin Chang, H.; Young Choi, J. Detection of moving objects with non-stationary cameras in 5.8 ms: Bringing motion detection to your mobile device. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Portland, OR, USA, 23–28 June 2013; pp. 27–34.
32. Sun, Y.; Liu, M.; Meng, M.Q.-H. Improving RGB-D SLAM in dynamic environments: A motion removal approach. *Robot. Auton. Syst.* **2017**, 89, 110–122. [CrossRef]
33. Zhao, H.; Chiba, M.; Shibasaki, R.; Shao, X.; Cui, J.; Zha, H. SLAM in a dynamic large outdoor environment using a laser scanner. In Proceedings of the IEEE International Conference on Robotics & Automation, Pasadena, CA, USA, 19–23 May 2008.
34. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
35. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 21–37.
36. Badrinarayanan, V.; Badrinarayanan, V.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef] [PubMed]
37. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE international Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
38. Han, S.; Xi, Z. Dynamic Scene Semantics SLAM Based on Semantic Segmentation. *IEEE Access* **2020**, *8*, 43563–43570. [CrossRef]
39. Henein, M.; Zhang, J.; Mahony, R.; Ila, V. Dynamic SLAM: The Need For Speed. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020.
40. Xiao, L.; Wang, J.; Qiu, X.; Rong, Z.; Zou, X. Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment. *Robot. Auton. Syst.* **2019**, *117*, 1–16. [CrossRef]
41. Burgard, W.S.; Fox, D. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*; The MIT Press: Cambridge, MA, USA, 2005.
42. Domhof, J.; Kooij, J.F.P.; Gavrila, D.M. An Extrinsic Calibration Tool for Radar, Camera and Lidar. In Proceedings of the International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019
43. Shan, T.; Englot, B. LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.
44. Bogoslavskyi, I.; Stachniss, C. Efficient Online Segmentation for Sparse 3D Laser Scans. *Photogramm. Fernerkund. Geoinf.* **2017**, *85*, 41–52. [CrossRef]
45. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]