applied sciences

MDPI

Article

Object-Level Semantic Map Construction for Dynamic Scenes

Xujie Kang [1,2], Jing Li [3,*], Xiangtao Fan [1], Hongdeng Jian [1] and Chen Xu [1,2]

[1] Key Laboratory of Digital Earth Science, Aerospace Information Research Institute (AIR), Chinese Academy of Sciences, Beijing 100094, China; kangxj@radi.ac.cn (X.K.); fanxt@radi.ac.cn (X.F.); jianhd@radi.ac.cn (H.J.); xuchen20@mails.ucas.ac.cn (C.X.)
[2] University of Chinese Academy of Sciences, No.19 Yuquan Road, Shijingshan District, Beijing 100049, China
[3] National Engineering Research Center for Geoinformatics, Aerospace Information Research Institute (AIR), Chinese Academy of Sciences, No. 20A, Datun Road, Chaoyang District, Beijing 100101, China
* Correspondence: lijing202115@aircas.ac.cn

Featured Application: Robotics, augmented reality, and dense reconstruction.

Abstract: Visual simultaneous localization and mapping (SLAM) is challenging in dynamic environments as moving objects can impair camera pose tracking and mapping. This paper introduces a method for robust dense bject-level SLAM in dynamic environments that takes a live stream of RGB-D frame data as input, detects moving objects, and segments the scene into different objects while simultaneously tracking and reconstructing their 3D structures. This approach provides a new method of dynamic object detection, which integrates prior knowledge of the object model database constructed, object-oriented 3D tracking against the camera pose, and the association between the instance segmentation results on the current frame data and an object database to find dynamic objects in the current frame. By leveraging the 3D static model for frame-to-model alignment, as well as dynamic object culling, the camera motion estimation reduced the overall drift. According to the camera pose accuracy and instance segmentation results, an object-level semantic map representation was constructed for the world map. The experimental results obtained using the TUM RGB-D dataset, which compares the proposed method to the related state-of-the-art approaches, demonstrating that our method achieves similar performance in static scenes and improved accuracy and robustness in dynamic scenes.

Keywords: object tracking; instance segmentation; dynamic simultaneous localization and mapping; camera pose tracking

Citation: Kang, X.; Li, J.; Fan, X.; Jian, H.; Xu, C. Object-Level Semantic Map Construction for Dynamic Scenes. *Appl. Sci.* **2021**, *11*, 645. https://doi.org/10.3390/app11020645

Received: 21 November 2020
Accepted: 4 January 2021
Published: 11 January 2021

1. Introduction

Nowadays, intelligent robots are not only widely used in industry, but also have extensive application prospects in various environments coexisting with humans. For decades, visual simultaneous localization and mapping (SLAM) systems have been focused on jointly solving the tasks of tracking the position of a camera as it explores unknown locations and creating a 3D map of the environment. With the rapid development of computer vision, visual SLAM has attracted increasing attention due to its diverse application scenarios, low cost, and ability to extract semantic information. The real-time capability of SLAM methods has made them the cornerstone of ambitious applications, such as autonomous driving, robot navigation, and augmented/virtual reality. Meanwhile, the wide availability of affordable structured light, time-of-flight depth sensors, and computer hardware upgrades have had enormous impacts on both the democratization of the acquisition of 3D models in real-time from hand-held cameras and providing robots with powerful but low-cost 3D sensing capabilities, and triggered extensive research aimed at real-time 3D scanning. Tracking camera motion while maintaining a dense representation of the 3D geometry of its environment in real-time has become more important than ever [1–4].

Dense camera pose tracking does not rely on the geometric features of the environment, but rather fully utilizes the geometric structure and texture information of the environment. Compared with the traditional SLAM method based on geometric features, dense camera pose tracking is much more robust in complex environments. State-of-the-art dense visual SLAM methods can produce impressive reconstructions of large indoor scenes. However, few mainstream algorithms for dense visual SLAM utilize object-level semantic information. Thinking of a scene as a representation of objects, we argue that this approach is a natural and efficient way to represent the things that are the most important for robotic scene understanding, mission planning, and physical interaction. It is also highly suitable as the basis for human–robot communication. Therefore, object-level SLAM has attracted considerable attention in recent years, in addition to being the development trend of visual SLAM. However, most such methods have difficulty obtaining better performance in dynamic environments, because moving objects can significantly interfere with positioning, scene segmentation, and map construction. If not actively segmented and detected, dynamic objects can be fused into the map, which can lead to irreversible corruption. To design robots that can interact with dynamic scenes smoothly, it is crucial to equip them with the abilities to (i) discover static and dynamic objects in a scene via segmentation and motion detection and (ii) track and estimate the 3D geometry of each object independently in a dynamic environment. These high-level, object-based semantic scene representations would greatly enhance the perception and physical interaction capabilities of robots.

In this work, we developed an object-oriented dense online SLAM system constructed based on ElasticFusion [5] with a focus on indoor scene understanding using RGB-D data. The semantic RGB-D visual SLAM architecture is integrated with a deep learning network with enhanced semantic instance segmentation and object motion detection for location and mapping in complex dynamic environments. Our objectives were to take a live stream of RGB-D frame data as input, to obtain the accurate camera pose, and to segment the scene into different objects while simultaneously tracking and reconstructing the 3D shapes of the objects in a dynamic environment. According to prior knowledge of previously constructed object model databases, the camera pose can be estimated roughly. Then, the motion information of each object in the database will be identified through object-oriented 3D tracking against the camera pose, and its motion status will be updated in the current frame. By creating associations between the results of fusion segmentation combining the semantic and geometric cues and the object database, the dynamic part in the current frame will be found and eliminated, so as to obtain the camera pose accurately. Based on the camera pose accuracy and instance segmentation results, we produced semantically-labelled surfel (surface element) reconstructions of object instances without strong a priori knowledge of the object types present in a scene and integrated object attribute information with the system to improve the system efficiency and accuracy. Our underlying assumption is that objects of interest can be detected and segmented in real time using efficient segmentation algorithms. We used a Mask RCNN [6] to provide 2D instance mask predictions, utilized geometric information in the depth images to enhance the object segmentation, and then fused these masks online into the surfel reconstruction along with a 3D "voxel mask" to fuse the instance foreground. This approach enables the problems in dense and accurate camera pose tracking, semantic instance segmentation, object dynamic detection, and map construction to be solved in a dynamic environment. The main contributions of this work can be summarized as follows.

1. An object motion detection method involving object-oriented 3D tracking against the camera pose based on prior knowledge of a previously constructed object database is provided.
2. This approach enables dynamic objects to be found in the current frame by creating associations between the results of the instance segmentation algorithm combining the semantic and geometric cues and the object database updated by object motion detection.
3. The camera motion estimation method reduces the overall drift by leveraging the 3D static model for frame-to-model alignment, as well as dynamic object culling in the current frame.

4. To achieve an accurate 3D object-level semantic map representation of a scene, conduct object motion detection, and perform accurate instance segmentation and camera pose estimation, ElasticFusion is utilized, creating a complete and novel dynamic RGB-D SLAM framework.

The remainder of this paper is organized as follows: Section 2 presents the related work conducted in the past few years on dynamic object-level semantic RGB-D SLAM. Section 3 describes our proposed approach. The experimental results obtained by testing the method on the public TUM datasets are given in Section 4. Section 5 provides a discussion and, finally, Section 6 summarizes the conclusions.

2. Related Work

2.1. Dense SLAM

The arrival of the Microsoft Kinect device and the sudden availability of inexpensive depth sensors to consumers triggered extensive research aimed at real-time 3D scanning. Systems such as KinectFusion [1] first enabled mapping of the 3D geometry of arbitrary indoor scenes accurately and in real-time, by fusing the images acquired by a depth camera simply by moving a sensor around the environment. Successors to KinectFusion have quickly addressed some of its shortcomings. Ongoing research has produced systems with impressive performance, ranging from scalable extensions [7] and loop closure capabilities [2,8] to methods that consider runtime limitations [9,10] in order to enable 3D scanning and mapping of large indoor scenes for maximum robustness and ease of use. Although some researchers have focused on extending the capabilities of such systems to handle very large scenes or to include loop closure [3], others have made the tracking more robust [2] or improved the memory and scale efficiency by using point-based instead of volumetric representations [7], leading to increased 3D reconstruction quality [11]. KinectFusion [12] proved that a truncated signed distance function (TSDF)-based map representation can achieve fast and robust mapping and tracking in small environments. Subsequent work [13,14] showed that the same principles are applicable to large-scale environments by choosing appropriate data structures. In addition, many research results have been obtained through the expression and organization of dense maps. Surface elements (surfels) have a long history in computer graphics and have found many applications in computer vision [5]. Recently, surfel-based map representations were introduced into the domain of RGBD-SLAM. A map of surfels is similar to a point cloud with the difference that each element encodes local surface properties—Typically a radius and normal in addition to the location. In contrast to TSDF-based maps, surfel clouds are naturally memory efficient and avoid the overhead due to switching representations between mapping and tracking that is typical of TSDF-based fusion methods. Whelan et al. [3] presented a surfel-based RGBD-SLAM system for large environments with local and global loop closure. With the advent of artificial intelligence, robots are required to interact well and intelligently with the environment. Although the above methods have achieved good results in camera tracking and map construction, they are assumed to run in static environments and do not add semantic information that can understand the environment, so they cannot meet the requirements of embedded unmanned platform intelligence. Because no semantic information is added, many robots cannot complete complex operations, such as road kinematics planning, obstacle avoidance, and human-computer interaction.

2.2. Object Semantic SLAM

The computer graphics and vision communities have devoted substantial efforts to object and scene segmentation. Segmented data can broaden the functionality of visual tracking and mapping systems, for instance, by enabling robots to detect objects. The task of target detection is applied to find all the targets or objects of interest in the image and to determine their positions and sizes, which is one of the core issues in machine vision. The traditional target detection process often utilizes a sliding window for region selection and then applies SIFT, HOG, or another method for feature extraction. Finally, the support vector machine approach and Adaboost are adopted for category judgment. However, the traditional method has poor robustness, making it incapable of reflecting the illumination changes, background diversity, etc. In addition, the regional selection is not targeted, the time complexity is high, and the window is redundant. Some researchers have proposed segmenting RGBD data based on the geometric properties of surface normals, mainly by assuming that objects are convex. Moosmann et al. [15] successfully segmented 3D laser data based on this assumption. The same principle has also been used by other authors, in order to segment objects in RGBD frames [16–20]. The significant recent progress in deep learning can provide solutions involving the utilization of semantic spatial information. From the RCNN to the Fast RCNN and Faster RCNN, deep learning methods represented by RCNNs in target detection have sharply improved the target detection accuracy. Subsequently, He et al. added a fully convolutional network to generate the corresponding MASK branch based on the original Faster RCNN algorithm. The Mask RCNN method can be used to accomplish various tasks such as target classification, target detection, and semantic segmentation. Deep learning neural networks are used to detect and segment objects in visual SLAM and build object-level semantic maps of scenes, achieving excellent results. Whelan et al. proposed a method of constructing a dense 3D semantic map called semantic fusion based on a convolutional neural network, which relied on the elastic fusion SLAM algorithm to provide indoor RGB-D video interframe for position-pose estimation, utilizing a convolutional neural network to predict the pixel-level object category label and ultimately combining the Bayesian upgrade strategy with the conditional random field model. This approach enabled the probability escalation of CNN prediction values from different perspectives and finally generated a dense 3D semantic map containing semantic information. Engel et al. [21] proposed a monocular and semi-dense 3Dsemantic mapping construction method based on a CNN and large scale direct (LSD) SLAM, where the LSD-SLAM method was used instead of the elastic fusion method to estimate the camera pose. Sunderhauf et al. [22] proposed an object-oriented mapping system composed of instances using bounding box detection from a CNN and an unsupervised geometric segmentation algorithm using RGB-D data. SLAM++ by Salas-Moreno et al. [23] was an early RGB-D object-oriented mapping system. They used point pair features for object detection and a pose graph for global optimization. The drawback was the requirement that the full set of object instances, with their very detailed geometric shapes, had to be known beforehand and preprocessed in an offline stage before running the algorithm. Stuckler and Behnke [24] also previously tracked object models learned beforehand by registering them to a multi-resolution surfel map. Tateno et al. [25] used a pre-trained database of objects to generate descriptors, but they used a KinectFusion [1] TSDF to segment regions of a reconstructed TSDF volume incrementally and to match 3D descriptors directly against those of other objects in the database. In RGB-only SLAM for object detection, Pillai and Leonard [26] used ORB-SLAM [27] to assist object recognition. They use a semi-dense map to produce object proposals and aggregate detection evidence across multiple views for object detection and classification. MO-SLAM by Dharmasiri et al. [28] focused on object discovery through duplicates. The authors used ORB [29] descriptors to search for sets of landmarks, which could be grouped by single rigid body transformation.

The above instance detection and segmentation methods based on deep learning do not fully utilize the scene information obtained by an RGB-D camera, and it is difficult to obtain accurate object segmentation using only RGB or depth data, or offline processing is required to obtain prior knowledge of scene objects. In contrast, our system combines RGB image semantic segmentation based on a deep learning network and depth image geometric segmentation results to obtain accurate object boundary and semantic category information without prior knowledge. It also offers methods of fusing labelled image data into segmented 3D maps, achieving real-time 3D object recognition, further improved tracking performance, and higher-level, object-based semantic scene descriptions while introducing the possibility of virtual or even real interaction with the scene.

2.3. Dynamic SLAM

Despite this progress, much work is still based on the fundamental assumption of a static environment, within which points in the 3D world always maintain the same spatial positions in the global world, with the only moving object being the camera. Most odometry methods perform registration between the current image and a previous reference, and the estimated transformation between these images is assumed to be due to the camera motion. Traditional SLAM algorithms are based on this assumption and achieving performance results, whereas robots tend to be artificial dynamic environments and are more concerned with dynamic objects, such as people, animals, and vehicles. Therefore, the recognition and 3D tracking of dynamic objects are the core issues of modern SLAM systems. Since it became possible to obtain better visual SLAM results in static environments, some studies have addressed visual SLAM in dynamic environments. There are three ways to deal with dynamic objects in visual SLAM. One deforms the whole world in a non-rigid manner in order to include a deformable/moving object [4]. The second specifically aims at building a single static background model while ignoring all possibly moving objects and, thus, improving the camera tracking accuracy [30–33]. For example, DynaSLAM [33] uses a CNN to identify dynamic objects and remove them from the image to avoid dynamic object interference. Zhao et al. [34] proposed a method of detecting potentially dynamic objects using semantic information and then using a contour refinement algorithm to detect objects more accurately. Sun et al. [35,36] proposed a motion removal approach to improve RGB-D SLAM in dynamic environments, as well as a background modeling method to segment foreground from background. This approach was mostly demonstrated on driving scenarios and was limited to sparse reconstructions. StaticFusion [31] performs segmentation by coupling camera motion residuals, depth inconsistency, and a regularization term. In the implicit handling of dynamic elements, it is common to use a robust cost function within the visual odometry front end, which penalizes the contributions of high-residual points and implicitly increases the robustness of pose estimation against un-modelled effects. Meanwhile, Kerl et al. [10,13] demonstrated the robustness against the presence of small moving objects in the scene. However, these solutions are insufficient and fail when moving parts occupy a significant portion of the image.

The third means of dealing with dynamic objects in visual SLAM models the dynamic components by creating sub-maps for every possibly rigidly moving object in the scene while fusing corresponding information into these sub-maps [37,38]. The SLAMMOT project [39] represented an important step towards extending the SLAM framework to dynamic environments by incorporating the detection and tracking of moving objects into the SLAM operation. Only very recently has the problem of reconstruction of dense dynamic scenes in real time been addressed. Xu et al. proposed the MID-Fusion [40] system, which is a new multi-instance dynamic RGBD SLAM system using a Mask R-CNN to find semantic instances and an object-level octree-based volumetric representation. Co-fusion [37] is another real-time dense mapping system capable of reconstructing both a static background map and dynamic objects from a scene. The corresponding paper presented a real-time pipeline that can segment and track dynamic objects based on either motion or semantic cues, reconstructing them separately using a surfel-based representation. DynSLAM [33] utilizes a mapping system for autonomous driving applications capable of separately reconstructing both the static environment and moving vehicles. However, the overall system is not real-time and vehicles constitute the only dynamic object class reconstructed, so the functionality of this approach is limited to road scenes. Barsan et al. [41] presented a stereo-based dense mapping algorithm for large-scale dynamic urban environments that can simultaneously reconstruct the static background, moving objects, and potentially moving but currently stationary objects separately. However, this algorithm cannot operate in real-time and is mainly designed for large indoor scenes. As in our method, Runz and Agapito [38] used Mask RCNN predictions to detect object instances. Their objective was to reconstruct and track moving instances densely using an ElasticFusion [3] surfel model for each object, as well as for the background static map. They did not attempt to reconstruct high-quality objects as pose-graph landmarks in room-scale SLAM. Rünz et al. [37] proposed a method of reconstructing and tracking each moving object with a separate 3D model, being one of the first real-time methods to perform dense tracking and fusion of multiple objects to handle dynamics explicitly. These approaches have naturally paved the way towards interaction and dynamic object reasoning.

In addition, motion detection from still cameras has been studied for several years [42,43], where foreground subtraction [44] and the optical flow [45] have been exploited. However, these approaches are not directly usable in many applications that involve moving cameras. Some new research on this problem has appeared in recent years [14,46–48]. These works have mainly adopted motion segmentation approaches to handle motion detection using a moving camera. For example, Tan et al. [49] projected keyframes onto the current frame for appearance and structure comparison, which could reveal changed regions. Keller et al. [7] used input points with no close model correspondence to seed a region-growing procedure to segment the current image into static and dynamic parts. Subsequently, model points matched with dynamic input points are removed from the reconstruction. This approach can only be utilized once a confident reconstruction of the scene is in place. Wangsiripitak and Murray [50] proposed edge-based polyhedral tracking to achieve dynamic object judgment. Further, Jaimez et al. [30] used geometric clustering to divide objects into dynamic and static parts. Nevertheless, in these approaches, no spatial or temporal coherence is enforced among the detected dynamic points between consecutive frames. These methods are more dependent on the shape and color characteristics of the object and do not have sufficient adaptability to the environment.

3. Methodology

3.1. System Overview

Our proposed method, built on the ElasticFusion [5] framework, enables real-time dense multi-model dynamic RGBD SLAM at the object level. In addition to obtaining camera poses accurately, our system can detect, track, and reconstruct multiple objects, including dynamic and static objects, while precisely segmenting each object instance and assigning it a corresponding semantic label. Figure 1 illustrates the proposed method in a detailed flowchart, which involves four steps.

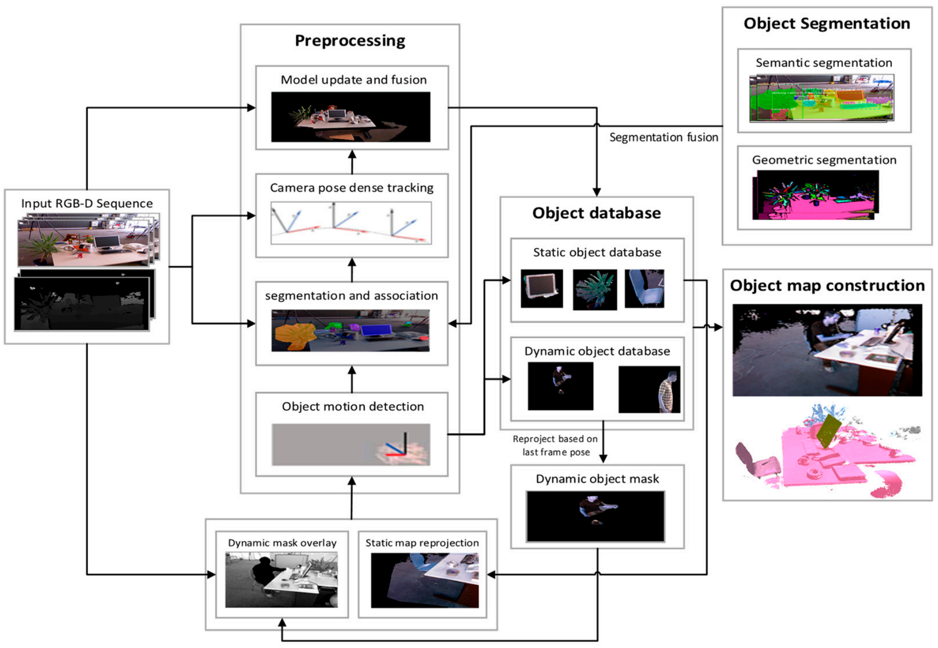


Figure 1. Overview of the system pipeline. Images data captured by an RGB-D camera serve as the input data for the motion detection, segmentation, camera pose tracking, and map construction modules, among others. Then, the precise camera pose and object-level semantic map can be obtained.

(1) Firstly, object motion detection is performed by 3D object tracking based on a priori knowledge of the object model database updated by the previous frame to find the moving object in the object database.
(2) Secondly, object instance accuracy segmentation is achieved using a combination of semantic and geometric cues in the current frame, then associated with the object database to find the dynamic part of the current image data.
(3) Thirdly, the camera pose is obtained precisely through dynamic part elimination in the current frame and each object model pose in the database is updated using the accurate camera pose.
(4) Fourthly, the object model is updated and fused with each frame input, and the 3D object-level semantic map is reconstructed incrementally.

The details of each component of the pipeline will be described below.

3.2. Notation and Preliminaries

We define the image space domain as $\Omega \subset$ N2 and define the 3D back-projection of point $u \in \Omega$ given depth map D as $p(u, D) = K^{-1}ud(u)$, where K is the intrinsic camera matrix and u is the homogeneous pixel coordinate form. We also specify the perspective projection of 3D point $p = [x, y, z]^T$ as $u = \pi(Kp)$, where $\pi(p) = [x/z, y/z]^T$ denotes the dehomogenization operation. The intensity value of pixel $u \in \Omega$ given color image C with color $c(u) = [c_1, c_2, c_3]^T$ is defined as $I(u, C) = (c_1 + c_2 + c_3)/3$. For each input frame at time t, we estimate the global pose of the camera Pt (with respect to a global frame F) by registering the current live depth map and color image captured by the camera with the surfel-splatted predicted depth map and color image of the active static model from the pose estimate corresponding to the last frame. All camera poses are represented by a transformation matrix:

$$T_{cur} = \begin{bmatrix} & R_t & & t_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \in SE_3, \tag{1}$$

where every object model is stored in surfels M^s_m, which have attribute values $(p \in R^3, n \in R^3, c \in R^3, w \in R, r \in R, t \in R^2)$ $\forall$s $< |M_m|$, position $p \in R^3$, normal $n \in R^3$, color $c \in N^3$, weight $w \in R$, radius $r \in R$, initialization timestamp t_0, and last updated timestamp t. The radius r of each surfel is intended to represent the local surface area around a given point while minimizing the visible holes, computed using the method of Salas-Moreno et al. [51]. Additionally, each model is associated with a class ID $C_m \in \{0 \ldots 80\}$ to represent semantic information and an object index ID $I_m = m \, \forall m \in \{0 \ldots N\}$, static indicators $st_m \in$ 0, 1, and a rigid pose $T_m \in SE_3$. We define the object model pose T_m as the product of the inverse of the camera pose corresponding to the frame in which the object is observed times the current frame camera pose:

$$T_m = {T_{obs}}^{-1} T_{cur} \tag{2}$$

where T_{obs} is the precise camera pose corresponding to the frame in which the object is observed and T_{cur} is the accurate camera pose of the current frame. Therefore, the defined object pose is equivalent to the relative conversion matrix of the camera pose between the two frames.

3.3. Object Motion Detection through 3D Tracking Based on Object Model Database

In a dynamic scene, the current frame data captured by an RGB-D sensor include the dynamic object. Generally, camera pose tracking is based on the assumption that the environment is static. However, as dynamic objects move independently of the camera, this assumption will be broken, resulting in inconsistent motion between the dynamic object and other parts of the image. Thus, dynamic objects inevitably affect the camera pose tracking and map construction and need to be eliminated. Considering the continuous motion nature of the moving objects, the method proposed in this paper makes full use of the scene prior knowledge in the dynamic object data and static object database updated by the last image frame to obtain motion information for each object in the database in the current frame. Note that the current frame is not segmented or associated with the object database at this point, so the dynamic part of the current frame is unknown.

First, the OpenGL rendering pipeline is used to project the objects in the dynamic object database based on the camera pose corresponding to the last frame to produce a dynamic object mask. Assuming that moving objects generally undergo continuous motion, the dynamic part in the current frame is roughly eliminated by overlaying this mask and the current frame data. In fact, due to the motion of the camera and that of the moving object itself, the dynamic object mask cannot accurately cover the dynamic part of the current frame image. Therefore, the boundary of the dynamic object mask is expanded by 5 pixels, finally yielding the dynamic object mask $Mask_{dynamic}$, as shown in Figure 2. This processing reduces the number of observations in the current frame data but is nevertheless preferable to adding outliers in camera pose estimation. Meanwhile, the non-dynamic map constructed at the camera pose corresponding to the last frame (including the static object and background models) is rendered. Then, alignment is performed between the current frame and rendered map by minimizing joint geometric error and photometric error functions for rough camera pose estimation. The geometric error and photometric error are defined as follows.

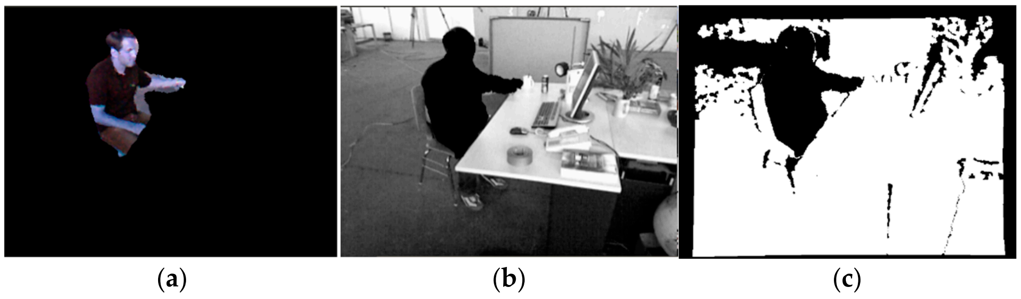


Figure 2. Based on the camera pose corresponding to the previous frame, the dynamic object mask is projected from the dynamic object database, and the dynamic object part in the current frame is removed. (**a**) Final dynamic object mask; (**b**) dynamic part culling in intensity image; (**c**) dynamic part culling in depth image.

3.3.1. Geometric Error Term

For the current frame t, we seek to minimize the cost of the point-to-plane ICP registration error between the 3D back-projected vertices of the current live depth map and the predicted depth map of static model from the last frame $t-1$:

$$E_{icp}^{static} = \begin{cases} \sum_i \left(\left(v^i - T_{rou} v_t^i \right) \cdot n^i \right)^2 & \text{if } i \notin Mask_{dynamic} \\ 0 & \text{if } i \in Mask_{dynamic} \end{cases} \tag{3}$$

where v_t^i is the back-projection of the i-th vertex in the current depth map D_t and v^i and n^i are respectively the back-projection of the i-th vertex of the predicted depth map of the static model from frame $t-1$ and its normal. T_{rou} describes the transformation that aligns the model in frame $t-1$ with that in frame t, $T_{rou}v_t^i$ represents the current frame vertex data v_t^i that is transformed to the camera coordinate system in the previous frame by pose T_{rou} and $Mask_{dynamic}$ is the dynamic object mask rendered by frame $t-1$ in the dynamic object database.

3.3.2. Photometric Error Term

The photometric image registration is defined as minimizing the brightness constancy between the live frame and synthesized view of the 3D model in frame $t-1$. The cost takes the form:

$$E_{rgb}^{static} = \begin{cases} \sum_{u}(I_t(u) - I_{t-1}(\pi(\mathrm{KT}_{rou}\pi^{-1}(u, D_t))))^2 & \text{if } u \notin Mask_{dynamic} \\ 0 & \text{if } u \in Mask_{dynamic} \end{cases} \tag{4}$$

After the current image RGB data approximately eliminates the dynamic objects in the object database, the remaining intensity data $I_t(u)$ and $I_{t-1}(.)$ that come from the predicted intensity map of the static model (including the background model and static objects) from the previous frame $t-1$ are used to construct an error equation to iteratively calculate the current frame camera pose T_{rou} based on the assumption of photometric consistency. Where $\pi^{-1}(u,\ D_t)$ represents that the current frame depth data D_t will be back-projected into 3D space and $T_{rou}\pi^{-1}(u,\ D_t)$ will transform vertex data $\pi^{-1}(u,\ D_t)$ to the camera coordinate system in the previous frame via pose T_{rou}, the pixel coordinates in the predicted intensity image will be obtained through perspective projection $\pi\left(KT_{rou}\pi^{-1}(u,\ D_t)\right)$, and, then the gray intensity value will be obtained.

The final overall alignment error term is:

$$E_{track}^{static} = \underset{T_{rou}}{min}\left(W_{icp}E_{icp}^{static} + E_{rgb}^{static}\right) \tag{5}$$

where W_{icp} is weight of the geometric error term. The cost function is minimized using the Gauss-Newton approach in a three-level coarse-to-fine scheme to calculate the rough camera pose T_{rou} relative to the static part of the environment.

This tracking method fully utilizes the geometric and texture information of the effective observation data in the current frame, increases the constraint conditions of the camera pose estimation, and improves the camera pose accuracy in real-time. Especially for ICP iterations, if the number of vertex data is small, camera pose estimation will be affected.

The relatively precise camera pose T_{rou} obtained in the previous step is used to update each object pose in the dynamic and static object databases according to the following definition:

$$T_m = {T_{obs}}^{-1}T_{rou} \tag{6}$$

Then, 3D tracking for each object is performed using the current frame data, the movement information of the object is updated as the prior knowledge for object motion judgement in the next frame. The vertex position p, normal vector n, and intensity I stored in object m are transferred to the current frame camera coordinate system using the object pose T_m, as follows:

$$p_{local} = {T_m}^{-1}p \tag{7}$$

$$n_{local} = {T_m}^{-1}n \tag{8}$$

$$I_{local}(\pi\left(K{T_m}^{-1}\pi^{-1}(u,\ D_{obs})\right) = I_{obs}(u) \tag{9}$$

where p_{local} and n_{local} are the vertex position and normal vector, respectively, in the camera coordinate system corresponding to the current frame and I_{local} is the intensity value of the vertex in the current image plane. Similar to the method used for camera pose estimation, the pose change T_m^{Δ} of object m against the camera pose T_{rou} is estimated by minimizing an energy that combines geometric error with photometric error between the data in the current frame and m transferred using pose T_m:

$$E_{track}^{m} = \underset{T_m^{\Delta}}{min}\left(W_{icp}E_{icp}^{m} + E_{rgb}^{m}\right) \tag{10}$$

where E_{icp}^{m} and E_{rgb}^{m} are the object geometric error and photometric consistency error, respectively, against the current frame data. T_m^{Δ} can be obtained using the Gauss–Newton approach in a three-level coarse-to-fine scheme. If $T_m^{\Delta} > \varepsilon$ (translation threshold 0.2, rotation angle threshold 0.1), which indicates that object moves independently from the camera in the current frame, then the object is treated as a dynamic object and inserted into the dynamic object database. Otherwise, the object is treated as a static object and inserted into the static object database.

3.4. Object Instance Segmentation and Data Association with Object Database

In this step, the current frame RGB-D data can be accurately segmented into object instances using the algorithm provided in [38], combined with semantic and geometric cues, which provide the foundation for recognition of the dynamic part in the current frame, object model reconstruction, and fusion. Meanwhile, using T_{rou}, the segmentation result is associated with the object database to identify the dynamic object part in the current frame or to create a new object. Figure 3 display the flowchart of the segmentation algorithm.

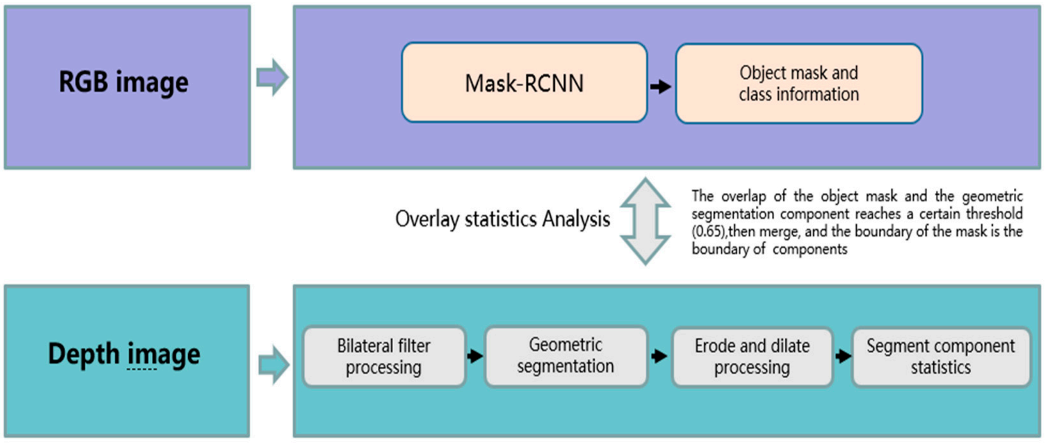

RGB image
Mask-RCNN
Object mask and class information
Overlay statistics Analysis
The overlap of the object mask and the geometric segmentation component reaches a certain threshold (0.65),then merge, and the boundary of the mask is the boundary of components
Depth image
Bilateral filter processing
Geometric segmentation
Erode and dilate processing
Segment component statistics

Figure 3. Flowchart of object instance segmentation based on geometric and semantic cues.

In the segmentation stage, the deep neural network Mask-RCNN [6] is used to provide object masks with semantic labels in semantic cues. Although this algorithm is impressive and provides good object masks, it tends to provide imperfect object boundaries and cannot be executed at the frame rate. Therefore, geometric segmentation, which produces higher-quality instance boundaries rapidly but tends to result in over-segmentation by assuming that the object convexity is taken out, is performed. Each stage of results processing of geometric segmentation on current frame depth data is shown in Figure 4: (a) represents the depth data processed by bilateral filtering; (b) represents the original boundary result generated after geometric segmentation; (c) represents the boundary result after dilation and erode operations that connected boundary inner points and eliminated isolated noise points; and (d) represents the corresponding color display for each component surrounded by the segmentation boundary after statistical analysis. Through statistical overlay analysis shown in Figure 3, we fuse two segmentation results to obtain high-quality segmentation and semantic recognition results in real time, compensating for the shortcomings of both the semantic and geometric cues. The boundary of the semantic segmentation mask becomes the boundary of one or more geometric segmentation components where the overlap rate reaches 65%. The final segmentation result is represented as instance collection $\{C_i\,, L_i|\, \mathrm{i} \rightarrow\ 0 \ldots N\}$, where C_i is the segmentation instance, L_i is the semantic category, and N is the instance number.

Next, the fusion segmentation results are associated with the object database through T_{rou}, as shown in Figure 5. The OpenGL pipeline is used to render the model ID image from the dynamic and static object databases updated by the current frame. The numbers of pixels in the overlapping part between instances C_i in the segmentation results and each ID I_m in the model ID rendering image are counted. If the pixel number is more than $65\% \cdot |C_i|$ and the semantic category information is consistent between C_i and m, then an association is created. If m is dynamic, then C_i is treated as the dynamic part $C_i^{dynamic}$ in the current frame. If C_i is not associated with any object model in the database, a new object model is created.

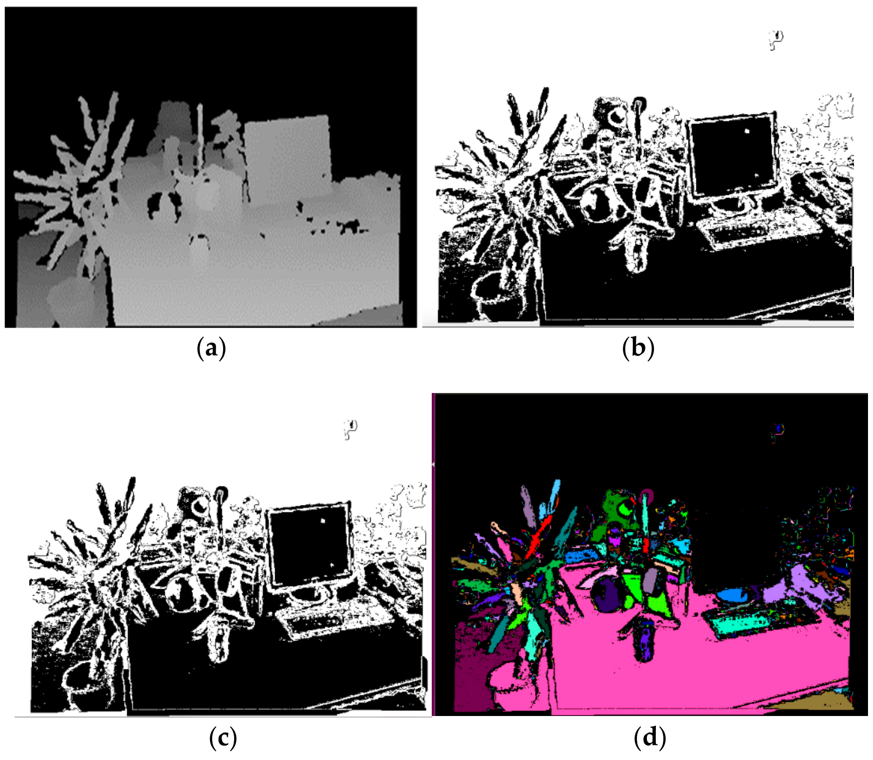


Figure 4. Each stage processing results of geometric segmentation on current frame depth data (**a**) Depth data processed by bilateral filtering; (**b**) raw geometric segmentation results; (**c**) dilation and erode segmentation results; (**d**) final statistical results of geometric components (each color represents a component), showing that the potted plant is over-segmented in multiple components.

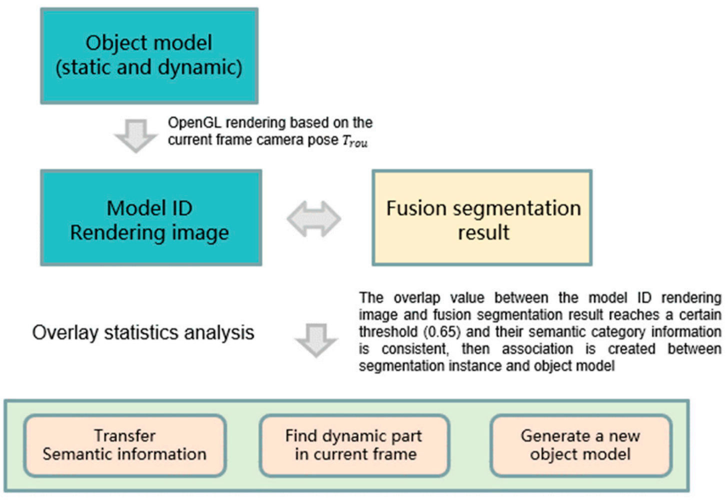

Object model
(static and dynamic)
OpenGL rendering based on the current frame camera pose T_{rou}
Model ID
Rendering image
Fusion segmentation result
Overlay statistics analysis
The overlap value between the model ID rendering image and fusion segmentation result reaches a certain threshold (0.65) and their semantic category information is consistent, then association is created between segmentation instance and object model
Transfer Semantic information
Find dynamic part in current frame
Generate a new object model

Figure 5. Segmentation results are associated with the object model database to find the dynamic part in the current frame, create a new object, and transfer the semantic information.

3.5. Accurate Camera Pose Tracking

Accurate camera pose tracking play a very important role in map construction. In this step, using the culling dynamic part $C_i^{dynamic}$ in the current frame, which was acquired in the previous step, the accurate camera pose T_{cur} can be obtained by performing alignment between the current frame data without the dynamic part and rendering the map data from previous frame $t-1$ by minimizing the geometric and photometric error functions. The geometric and photometric error functions are respectively defined as follows:

$$E_{icp}^{static} = \begin{cases} \sum_i \left(\left(v^i - T_{cur} v_t^i \right) \cdot n^i \right)^2 & \text{if } i \notin C_i^{dynamic} \\ 0 & \text{if } i \in C_i^{dynamic} \end{cases} \tag{11}$$

The geometric error function (11) represents that after the current frame depth data accurately removes the dynamic part, the remaining vertex data v_t^i and the vertex data v^i that come from the predicted depth map of the static model (including the background model and static objects) from the previous frame $t-1$ are used to calculate the precise current frame camera pose T_{cur} based on the point-surface ICP iterative algorithm.

$$E_{rgb}^{static} = \begin{cases} \sum_u \left(I_t(u) - I_{t-1}\left(\pi \left(\mathrm{K} \mathrm{T}_{cur} \pi^{-1}(u, D_t) \right) \right) \right)^2 & \text{if } u \notin C_i^{dynamic} \\ 0 & \text{if } u \in C_i^{dynamic} \end{cases} \tag{12}$$

In photometric error function (12), $I_t(u)$ is the current frame gray intensity map after removing the dynamic part, $I_{t-1}(\cdot)$ provides the color attached to a vertex on the model in frame $t-1$, and $C_i^{dynamic}$ is the dynamic object part in the current frame. After acquiring the accurate camera pose T_{cur}, each object model pose is redefined in the database for object model updating and subsequent fusion. The new object pose is set as 4D identity matrix and T_{obs}=T_{cur}:

$$T_m{}^{dynamic} = T_m \, T_m^{\Delta} \tag{13}$$

$$T_m{}^{static} = T_{obs}{}^{-1} T_{cur} \tag{14}$$

$$T_m{}^{new} \tag{15}$$

where T_m^{Δ} is the amount of change in the object pose relative to the camera movement and $T_m{}^{new}$ is newly created object model pose.

3.6. Object Model Fusion, Semantic Map Optimization, and Construction

When the first RGB-D data frame is input, a background model and object model database are created through image data segmentation and recognition with the camera pose as the identity matrix. Then, an object-level map representation is built for the scene. When the subsequent frame comes in, our proposed method is performed step by step. At first, the object motion information is obtained through 3D tracking against the rough camera pose in the current frame.

Next, the object instance segmentation results produced by the fusion segmentation algorithm combining semantic and geometric cues are used to associate them with the object model database, aiming to find the dynamic part in the current frame, transfer the object semantic category information, and create a new object. Accurate camera pose tracking can be achieved by culling the dynamic part in the current frame data. Each object model pose is updated based on the accurate camera pose. When C_i is associated with m in the database, m will be updated using C_i data. C_i data are transferred to the camera coordinate system in which the object is observed using object pose T_m. Then, the fusing algorithm provided by [3,7] is performed between the data from C_i and the raw data stored in m. Note that dynamic and static objects share the same fusion algorithm, but the dynamic object model, a pedestrian, as a non-rigid object, was created directly using C_i data regardless of the model created previously. The new object model, treated as a static object, is created directly using C_i data, which are not associated with any object model in the database.

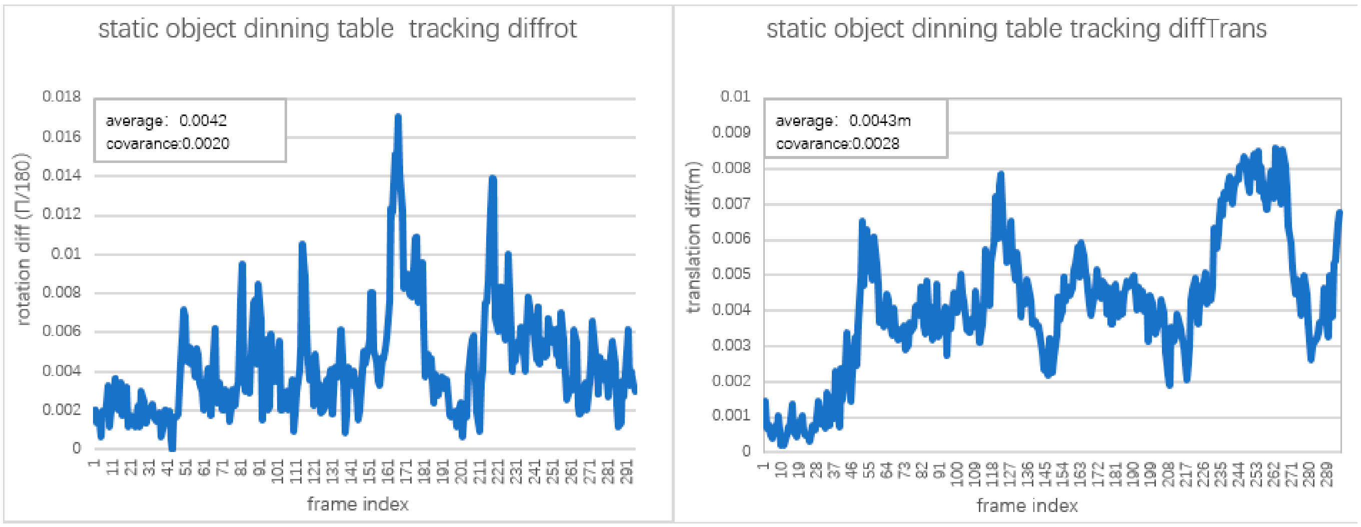
Due to the existence of accumulated camera pose error, we adopted the ElasticFusion [3] strategy to optimize the deformation map generated by static data, including the static object model and background model. Finally, we constructed the object-level semantic map accurately.

4. Results

We evaluated our system on a Linux system using a laptop (Dell Precision 7530 GPU workstation) with a 12 Intel Core i7-8750H CPU, 2.20 GHz frequency, 16 GB RAM memory, 6G NVIDIA corporation GP104GLM. As our workstation only has one GPU, the image semantic instance segmentation was processed offline using a Mask R-CNN on the GPU with the publicly available weights and implementation without fine-tuning. Our input was standard 640 × 480 resolution RGB-D video. Each object constructed was stored into the dynamic or static object database with surfels representation.

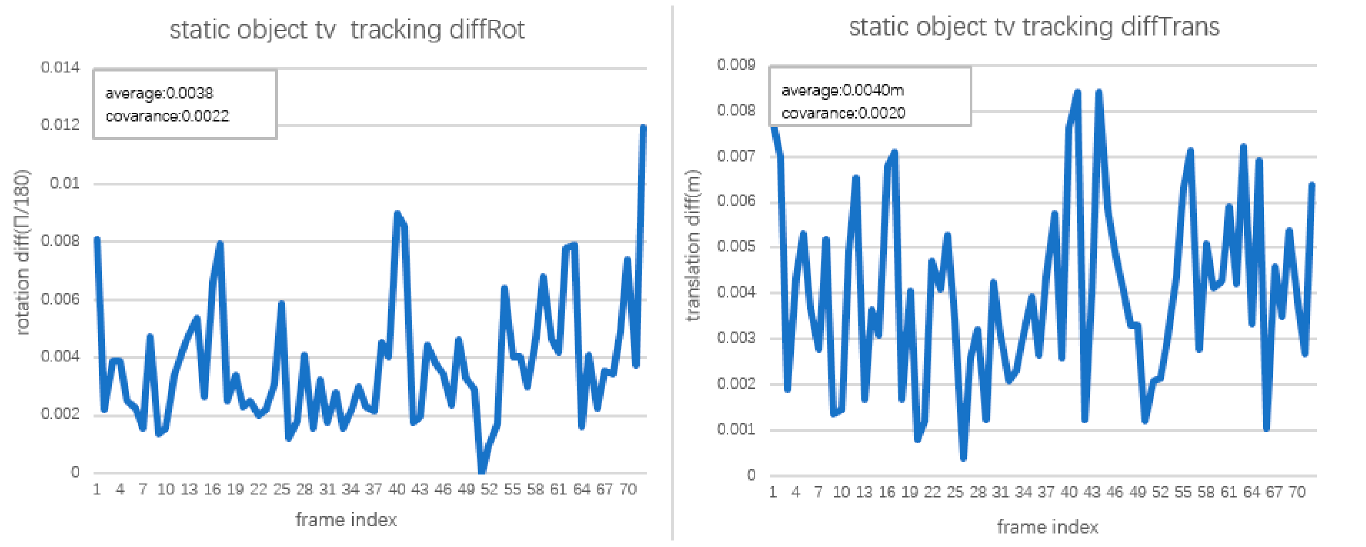
4.1. Object 3D Tracking for Motion Detection

We performed object 3D tracking on TUM RGB-D dataset fr3/sit-xyz and estimated the object motion detection result.fr3/sit-xyz is a low dynamic environment dataset with a frame rate of 30 frames/s, in which the object pedestrian moved slightly. If low dynamic object motion can be detected, then high dynamic object motion can also be detected under the premise of accurate object instance segmentation. Figure 6 presents the evaluation results. The static objects, such as dining tables, TVs, and potted plants, are tracked. The motion tracking results show that the angle change against the current camera pose does not exceed 0.008 rad and that the translation change does not exceed 0.01 m. For dynamic objects, such as a pedestrian walking normally, the average change in angle against the current camera pose is 0.1776 rad, with a variance of 0.0410, and the average change in translation against the current camera pose is 0.2262 m, with a variance of 0.0308. Therefore, to find actual dynamic objects as much as possible, we set the translation threshold to a low value of 0.2 m and the angle threshold to a low value of 0.1 m to judge the dynamic and static objects in the scene.

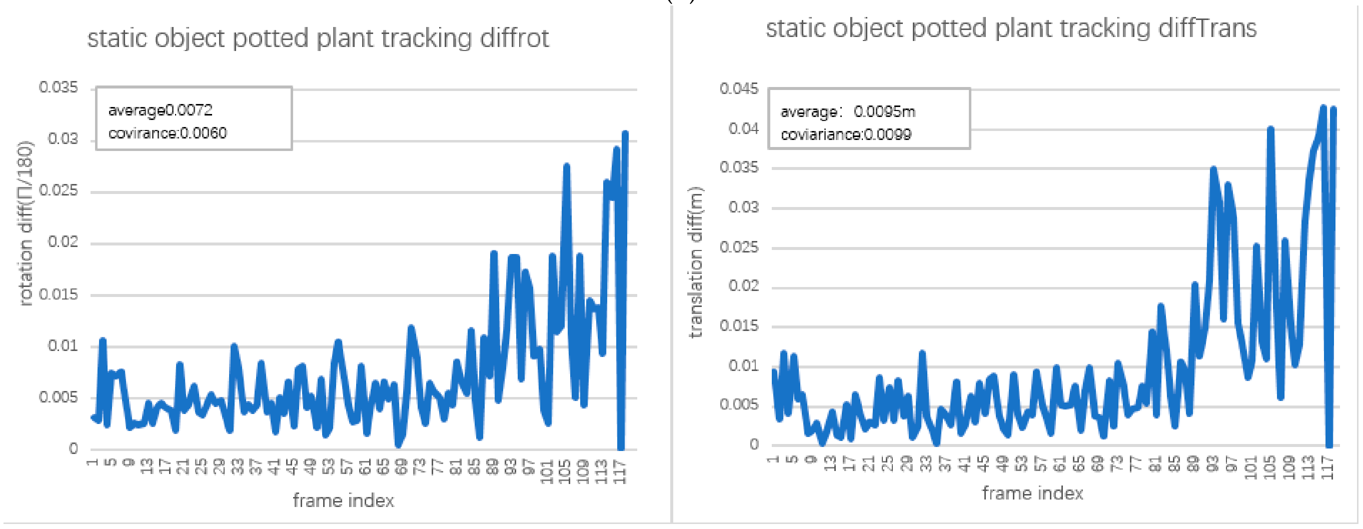


(a)

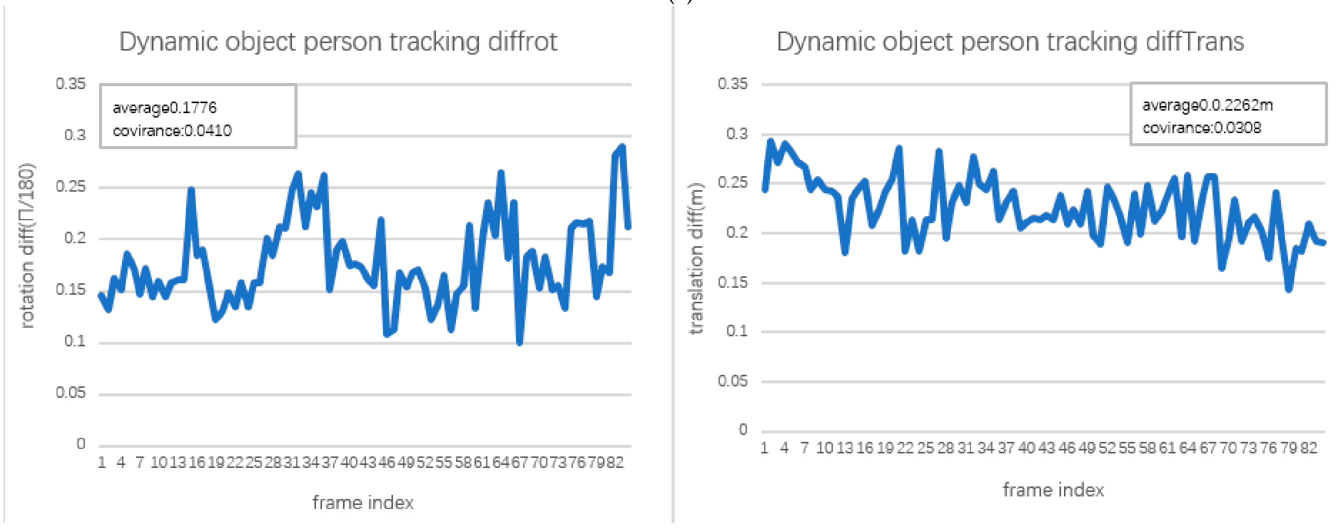
Figure 6. *Cont.*



(**b**)



(**c**)

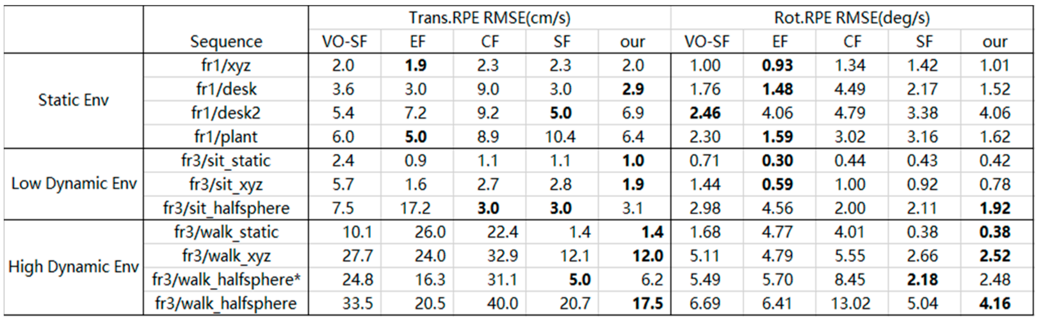


(**d**)

Figure 6. The different objects motion detection result against the current camera pose: (**a**) Static object dining table; (**b**) static object TV; (**c**) static object potted plant; (**d**) dynamic object pedestrian.

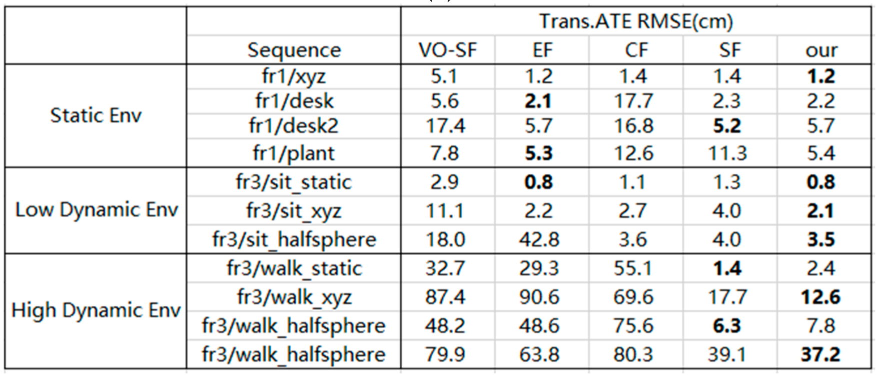
4.2. Camera Pose Estimation Accuracy

We evaluated the camera pose estimation accuracy in static and dynamic environments by employing the widely used TUM RGB-D dataset [52]. The dataset provides RGB-D sequences with ground truth camera trajectories recorded by a motion capture system. We report the commonly used root mean square error (RMSE) of the absolute trajectory error (ATE) and relative pose error. As shown in Figure 7, because our proposed method was developed based on the ElasticFusion framework, the camera pose estimation has accuracy comparable to that of the ElasticFusion [3] method in a static scene. Meanwhile, in a dynamic environment, inspired by the MaskFusion [38] and Co-Fusion [37] methods, our proposed method, which eliminates the dynamic part in the current frame and tracks all static objects models for camera pose estimation, achieves higher accuracy than the other methods. In general, the experimental results, which compare the proposed method to the related state-of-the-art approaches, demonstrate that our method achieves similar performance in static scenes, and improved accuracy and robustness in dynamic scenes.



		Trans.RPE RMSE(cm/s)					Rot.RPE RMSE(deg/s)				
	Sequence	VO-SF	EF	CF	SF	our	VO-SF	EF	CF	SF	our
Static Env	fr1/xyz	2.0	**1.9**	2.3	2.3	2.0	1.00	**0.93**	1.34	1.42	1.01
	fr1/desk	3.6	3.0	9.0	3.0	**2.9**	1.76	**1.48**	4.49	2.17	1.52
	fr1/desk2	5.4	7.2	9.2	**5.0**	6.9	**2.46**	4.06	4.79	3.38	4.06
	fr1/plant	6.0	**5.0**	8.9	10.4	6.4	2.30	**1.59**	3.02	3.16	1.62
Low Dynamic Env	fr3/sit_static	2.4	0.9	1.1	1.1	**1.0**	0.71	**0.30**	0.44	0.43	0.42
	fr3/sit_xyz	5.7	1.6	2.7	2.8	**1.9**	1.44	**0.59**	1.00	0.92	0.78
	fr3/sit_halfsphere	7.5	17.2	**3.0**	**3.0**	3.1	2.98	4.56	2.00	2.11	**1.92**
High Dynamic Env	fr3/walk_static	10.1	26.0	22.4	1.4	**1.4**	1.68	4.77	4.01	0.38	**0.38**
	fr3/walk_xyz	27.7	24.0	32.9	12.1	**12.0**	5.11	4.79	5.55	2.66	**2.52**
	fr3/walk_halfsphere*	24.8	16.3	31.1	**5.0**	6.2	5.49	5.70	8.45	**2.18**	2.48
	fr3/walk_halfsphere	33.5	20.5	40.0	20.7	**17.5**	6.69	6.41	13.02	5.04	**4.16**

(**a**)



		Trans.ATE RMSE(cm)				
	Sequence	VO-SF	EF	CF	SF	our
Static Env	fr1/xyz	5.1	1.2	1.4	1.4	**1.2**
	fr1/desk	5.6	**2.1**	17.7	2.3	2.2
	fr1/desk2	17.4	5.7	16.8	**5.2**	5.7
	fr1/plant	7.8	**5.3**	12.6	11.3	5.4
Low Dynamic Env	fr3/sit_static	2.9	**0.8**	1.1	1.3	**0.8**
	fr3/sit_xyz	11.1	2.2	2.7	4.0	**2.1**
	fr3/sit_halfsphere	18.0	42.8	3.6	4.0	**3.5**
High Dynamic Env	fr3/walk_static	32.7	29.3	55.1	**1.4**	2.4
	fr3/walk_xyz	87.4	90.6	69.6	17.7	**12.6**
	fr3/walk_halfsphere	48.2	48.6	75.6	**6.3**	7.8
	fr3/walk_halfsphere	79.9	63.8	80.3	39.1	**37.2**

(**b**)

Figure 7. Quantitative comparison of camera pose estimation between the proposed method and other methods. (**a**) Relative pose error. (**b**) Absolute trajectory error.

4.3. Map Construction

As the object model constructed has no true value, we qualitatively and intuitively viewed the constructed model, as shown in Figure 8. Figure 8a–c depict static object models constructed in the scene, namely, those of a TV, potted plant, and chair. Figure 8d shows the dynamic object models constructed at different moments. The entire model constructed in the scene is provided in Figure 9. Figure 8a–c are an overlay of the static object and background models, an overlay of the static and dynamic object models, and the semantic label map representation of the scene, respectively. Static object TV model construction with different 3D tracking results is shown in Figure 10.

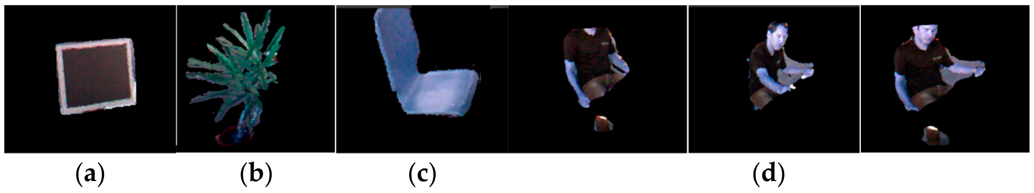


Figure 8. Static object models in the static object database for a (**a**) TV, (**b**) potted plant, and (**c**) chair. (**d**) Dynamic object model in the dynamic object database at different moments.

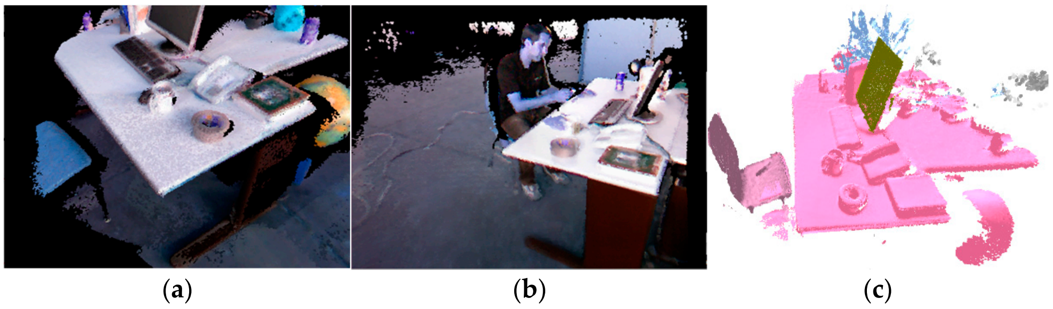


Figure 9. Scene map construction in surfel-based representation: (**a**) Static object model and background model representation; (**b**) overlay of dynamic pedestrian and static object models at a certain moment; (**c**) semantic label map representation of the scene.

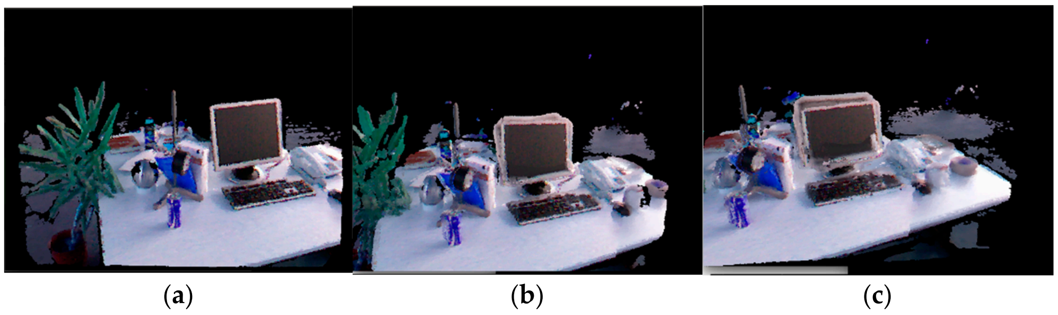


Figure 10. Object TV model fuse construction with different 3D tracking results displayed: (**a**) Accurate 3D tracking result and object model fuse construction; (**b**,**c**) 3D tracking results with errors and object model fuse construction with ghosts.

5. Discussion

5.1. Object Motion Detection

Our object motion detection system is based on the assumption that moving objects generally undergo continuous motion. Using the object database maintained by the last frame data, the dynamic objects in the current frame are removed and the rough camera pose is obtained. This pose can serve as a good initial value of the camera pose for object motion detection, enabling the motion status of each object in the object database to be obtained in current frame. The motion status of each object is updated in real time according to the object 3D tracking results. Note that we use the current frame raw data to perform object 3D tracking. Our method only performs motion detection on large objects with a large number of data stored. Since small objects (such as tea cups and pencils) do not have sufficient data constraints for object 3D tracking, unreliable tracking results may be produced. In addition, small moving objects affect the camera pose estimation less than large objects. Therefore, in order to reduce the memory consumption of the system, smaller objects are ignored in the motion detection and our system only reconstructs and tracks the objects occupying more than 0.5% of pixels in the image. The advantage of our method is that it performs 3D tracking for object motion detection in 3D space while fully utilizing the depth information and gray texture information stored in object. This motion detection strategy finds hidden moving objects, which move perpendicular to the viewpoint of the camera, more easily than methods performed in 2D space with only using epipolar constraints between the previous and current frames. In addition, the accuracy of object segmentation directly affects the object 3D tracking results.

5.2. Object Instance Segmentation and Association

The object instance segmentation provides the data foundation for 3D object tracking and recognition of the dynamic part in the current frame. Inaccurate instance segmentation will lead to object motion detection errors and inaccurate camera pose estimation. For example, as the geometric shapes of potted plants are irregular, the object mask generated by the Mask-RCNN [6] tends to have the overall outline of the periphery of the object, so the interior contains some pixels that are not the potted plants. As the camera perspective changes, the pixels that are not potted plants will change, and the 3D tracking results will inevitably produce errors. Thus, potted plants may be treated as dynamic objects. Therefore, a fusion segmentation algorithm combining geometric and semantic cues is used to segment the object instance precisely in the current frame, which guarantees reliable object motion detection and camera pose estimation.

Our strategy for finding the dynamic part in the current frame involves utilizing the raw data for the current frame raw data and 3D object tracking to update the motion status of each object in object database. Then, the correlation between the object instance segmentation results and object database is established by using T_{rou}. Finally, the dynamic part in the current frame is identified accurately according to the object motion information in object database.

5.3. Camera Pose Estimation Accuracy

The surfel-based map constructed previously can be used to convey the temporal and spatial consistency of the geometric and semantic information. Therefore, the camera pose estimation is more accurate due to its leveraging of the 3D static model for frame-to-model alignment, rather than frame-to-frame alignment. The Co-Fusion [37] and Mask-Fusion [38] approaches use background data except for the previously recognized objects to estimate the camera pose. This method camera pose estimation has the drawback that when the recognized objects occupy most of a frame, the tracking results will be inaccurate as the background data will be less effective for camera pose tracking. However, in our method, all static object model data are used to estimate the camera pose, providing an additional data source for camera pose tracking and improved accuracy.

5.4. Object Model Fusion and Updating

The object model in the object database is updated by the previous frame. The static and dynamic object models share the same fusion method based on the object model pose. As the dynamic object is a non-rigid object, the data fusion between the current frame and model constructed previously will cause model ghosting. Therefore, the pedestrian model constructed previously is cleared, and a new model is generated according to the current frame data.

5.5. System Efficiency

In object 3D tracking, the depth information and gray-scale texture information are both used so the processed data has been massively increased compared with the 2D tracking method. To achieve the real-time or quasi-real-time performance of the algorithm, we perform this task with GPU. If there are 3–4 objects tracking in the scene at the same time, the system test can execute at a rate of approximately 20–25 fps. Nevertheless, our algorithm can handle multiple moving objects in the scene by simply taking up more GPU computing resources. However, if data occlusion occurs due to move overlap on these moving objects, our system will be helpless because the depth data and grayscale texture data of the same object used for 3D tracking would be different at different moments after the objects overlap. As a result, 3D object tracking results would have large errors.

6. Conclusions

This paper presented a new object-level RGB-D dense SLAM approach for accurate motion estimation of an RGB-D camera in the presence of moving objects, while constructing an object-level semantic map of the scene. There are three important innovations in this method. First, object motion detection is performed in 3D space. Compared with the method in 2D space, the 3D object tracking method against the camera pose makes full use of the depth and gray texture information stored in the object and is more robust for detecting potential moving objects following the camera movement. Second, the method of establishing an association between instance segmentation and an object database with motion information could accurately find the dynamic part of the current frame. Third, our method fully utilizes the static data in the current frame as a data source for camera pose tracking, eliminating the interference from the dynamic data and thereby improving the camera pose accuracy. The quantitative and qualitative results demonstrate that our method can accurately detect dynamic objects in a scene and eliminate the effects of dynamic objects on camera pose tracking and map construction. The camera positioning accuracy can reach the level of state-of-the-art techniques when the tested sequences include several moving objects. We found that small objects are potentially difficult to track, and it remains a topic for future work to optimize the method for according data.

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

Funding: This work was funded and supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (grant no. XDA19080101). This work was also partially supported by Key Deployment Program of Chinese Academy of Sciences (no. KFZD-SW-317). This research was funded by the National Key Research and Development Program of China (no. 2019YFC1511304), and the National Natural Science Foundation of China (no. 41101436). This research was also partially funded by the National Natural Science Foundation of China (No. 41901328).

Institutional Review Board: Not applicable.

Informed Consent: Not applicable.

Data Availability Statement: I use public dataset to verify my algorithm, and the public dataset link already writed in reference [52].

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

References

1. Newcombe, R.A.; Izadi, S.; Hilliges, O.; Molyneaux, D.; Kim, D.; Davison, A.J.; Kohli, P.; Shotton, J.; Hodges, S.; Fitzgibbon, A.W. KinectFusion: Real-time Dense Surface Mapping and Tracking. In Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, Basel, Switzerland, 26–29 October 2011; pp. 127–136.
2. Whelan, T.; Kaess, M.; Fallon, M.; Johannsson, H.; Mcdonald, J. Kintinuous: Spatially Extended KinectFusion. *Robot. Auton. Syst.* **2012**, *69*, 3–14. [CrossRef]
3. Whelan, T.; Leutenegger, S.; Moreno, R.; Glocker, B.; Davison, A. ElasticFusion: Dense SLAM Without A Pose Graph. In Proceedings of the Robotics: Science and Systems, Rome, Italy, 13–17 July 2015. [CrossRef]
4. Newcombe, R.A.; Fox, D.; Seitz, S.M. DynamicFusion: Reconstruction and Tracking of Non-rigid Scenes in Real-time. In Proceedings of the Computer Vision & Pattern Recognition, Boston, MA, USA, 7–12 June 2015.
5. Whelan, T.; Salas-Moreno, R.F.; Glocker, B.; Davison, A.J.; Leutenegger, S. ElasticFusion: Real-time dense SLAM and light source estimation. *Int. J. Robot. Res.* **2016**, *35*, 1697–1716. [CrossRef]
6. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*; IEEE: Piscataway, NJ, USA, 2017.

7. Keller, M.; Lefloch, D.; Lambers, M.; Izadi, S.; Weyrich, T.; Kolb, A. Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion. In Proceedings of the International Conference on 3d Vision, Seattle, WA, USA, 29 June–1 July 2013.
8. Henry, P.; Krainin, M.; Herbst, E.; Ren, X.; Fox, D. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *Int. J. Robot. Res.* **2013**, *31*, 647–663. [CrossRef]
9. Steinbrücker, F.; Sturm, J.; Cremers, D. Volumetric 3D Mapping in Real-time on a CPU. In Proceedings of the IEEE International Conference on Robotics and Automation, Hong Kong, China, 31 May–7 June 2014; pp. 2021–2028.
10. Kerl, C.; Sturm, J.; Cremers, D. Dense Visual SLAM for RGB-D Cameras. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 2100–2106.
11. Lefloch, D.; Weyrich, T.; Kolb, A. Anisotropic Point-based Fusion. In Proceedings of the International Conference on Information Fusion, Washington, DC, USA, 6–9 July 2015.
12. Izadi, S.; Kim, D.; Hilliges, O.; Molyneaux, D.; Newcombe, R.A.; Kohli, P.; Shotton, J.; Hodges, S.; Freeman, D.; Davison, A.J. KinectFusion: Real-time 3D Reconstruction and Interaction using a Moving Depth Camera. In Proceedings of the Acm Symposium on User Interface Software & Technology, Santa Barbara, CA, USA, 16–19 October 2011.
13. Kerl, C.; Sturm, J.; Cremers, D. Robust Odometry Estimation for RGB-D Cameras. In Proceedings of the Robotics and Automation (ICRA), 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013.
14. Chen, T.; Lu, S. Object-Level Motion Detection from Moving Cameras. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *27*, 2333–2343. [CrossRef]
15. Moosmann, F.; Pink, O.; Stiller, C. Segmentation of 3D Lidar Data in Non-flat Urban Environments using a Local Convexity Criterion. In Proceedings of the Intelligent Vehicles Symposium, Xi'an, China, 3–5 June 2009.
16. Whelan, T.; Kaess, M.; Finman, R.E.; Leonard, J.J. Efficient Incremental Map Segmentation in Dense RGB-D Maps. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014.
17. Karpathy, A.; Miller, S.; Li, F.F. Object Discovery in 3D Scenes via Shape Analysis. In Proceedings of the IEEE International Conference on Robotics & Automation, Karlsruhe, Germany, 6–10 May 2013.
18. Stein, S.C.; Schoeler, M.; Papon, J.; Bernstein, F.W. Object Partitioning Using Local Convexity. In Proceedings of the Computer Vision & Pattern Recognition, Columbus, OH, USA, 23–28 June 2014.
19. Tateno, K.; Tombari, F.; Navab, N. Real-time and Scalable Incremental Segmentation on Dense SLAM. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, Hamburg, Germany, 28 September–2 October 2015.
20. Uckermann, A.; Elbrechter, C.; Haschke, R.; Ritter, H. 3D Scene Segmentation for Autonomous Robot Grasping. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, Vilamoura, Portugal, 7–12 October 2012.
21. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Computer Vision—ECCV 2014*; Springer: Cham, Switzerland, 2014.
22. Sunderhauf, N.; Pham, T.T.; Latif, Y.; Milford, M.; Reid, I. Meaningful Maps with Object-oriented Semantic Mapping. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017; pp. 5079–5085.
23. Salas-Moreno, R.F.; Newcombe, R.A.; Strasdat, H.; Kelly, P.H.J.; Davison, A.J. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 1352–1359.
24. Stückler, J.; Behnke, S. Model Learning and Real-Time Tracking using Multi-Resolution Surfel Maps. In Proceedings of the Twenty-sixth AAAI Conference on Artificial Intelligence, Toronto, ON, Canada, 22–26 July 2012.
25. Tateno, K.; Tombari, F.; Navab, N. When 2.5D is not Enough: Simultaneous Reconstruction, Segmentation and Recognition on Dense SLAM. In Proceedings of the IEEE the International Conference on Robotics & Automation, Stockholm, Sweden, 16–21 May 2016.
26. Pillai, S.; Leonard, J. Monocular SLAM Supported Object Recognition. In Proceedings of the Robotics: Science and Systems, Rome, Italy, 13–17 July 2015.
27. Mur-Artal, R.; Tardós, J.D. ORB-SLAM: Tracking and Mapping Recognizable Features. In Proceedings of the Workshop on Multi View Geometry in Robotics, Berkeley, CA, USA, 13 July 2014.
28. Dharmasiri, T.; Lui, V.; Drummond, T. MO-SLAM: Multi Object SLAM with Run-time Object Discovery through Duplicates. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, Daejeon, South Korea, 9–14 October 2016.
29. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G.R. ORB: An Efficient Alternative to SIFT or SURF. In Proceedings of the International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011.
30. Jaimez, M.; Kerl, C.; Gonzalez-Jimenez, J.; Cremers, D. Fast Odometry and Scene Flow from RGB-D Cameras based on Geometric Clustering. In Proceedings of the IEEE International Conference on Robotics & Automation, Singapore, Singapore, 29 May–3 June 2017.
31. Scona, R.; Jaimez, M.; Petillot, Y.R.; Fallon, M.; Cremers, D. StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 1–9.
32. Barnes, D.; Maddern, W.; Pascoe, G.; Posner, I. Driven to Distraction: Self-Supervised Distractor Learning for Robust Monocular Visual Odometry in Urban Environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 1894–1900.

33. Berta, B.; Facil, J.M.; Javier, C.; Jose, N. DynaSLAM: Tracking, Mapping and Inpainting in Dynamic Scenes. *IEEE Robot. Autom. Lett.* **2018**, *3*, 4076–4083.
34. Zhao, L.; Liu, Z.; Chen, J.; Cai, W.; Wang, W.; Zeng, L. A Compatible Framework for RGB-D SLAM in Dynamic Scenes. *IEEE Access* **2019**, *7*, 75604–75614. [CrossRef]
35. Sun, Y.; Liu, M.; Meng, Q.H. Active Perception for Foreground Segmentation: An RGB-D Data-Based Background Modeling Method. *IEEE Trans. Autom. Sci. Eng.* **2019**, *16*, 1596–1609. [CrossRef]
36. Sun, Y.; Liu, M.; Meng, M.Q.H. Motion Removal for Reliable RGB-D SLAM in Dynamic Environments. *Robot. Auton. Syst.* **2018**, *108*, 115–128. [CrossRef]
37. Rünz, M.; Agapito, L. Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple Objects. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, Singapore, 29 May–3 June 2017.
38. Rünz, M.; Agapito, L. MaskFusion: Real-Time Recognition, Tracking and Reconstruction of Multiple Moving Objects. In Proceedings of the 2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Munich, Germany, 16–20 October 2018.
39. Wang, C.C.; Thorpe, C.; Hebert, M.; Thrun, S.; Durrant-Whyte, H. Simultaneous Localization, Mapping and Moving Object Tracking. *Int. J. Robot. Res.* **2007**, *26*, 889–916. [CrossRef]
40. Xu, B.; Li, W.; Tzoumanikas, D.; Bloesch, M.; Leutenegger, S. MID-Fusion: Octree-based Object-Level Multi-Instance Dynamic SLAM. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019.
41. Barsan, I.A.; Liu, P.; Pollefeys, M.; Geiger, A. Robust Dense Mapping for Large-Scale Dynamic Environments. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Singapore, Singapore, 29 May–3 June 2017.
42. Derome, M.; Plyer, A.; Sanfourche, M.; Besnerais, G.L. Moving Object Detection in Real-Time Using Stereo from a Mobile Platform. *Unmanned Syst.* **2015**, *3*, 253–266. [CrossRef]
43. Kundu, A.; Krishna, K.M.; Sivaswamy, J. Moving Object Detection by Multi-view Geometric Techniques from a Single Camera Mounted Robot. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, St. Louis, MO, USA, 10–15 October 2009.
44. Bayona, Á.; SanMiguel, J.C.; Martínez, J.M. Stationary Foreground Detection using Background Subtraction and Temporal Difference in Video Surveillance. In Proceedings of the 2010 IEEE International Conference on Image Processing, 26–29 September 2010; pp. 4657–4660.
45. Sun, Y.; Liu, M.; Meng, Q.H. Invisibility: A Moving-object Removal Approach for Dynamic Scene Modelling using RGB-D Camera. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017.
46. Leonardo, S.; André, C.; Luiz, G.A.; Tiago, N. Stairs and Doors Recognition as Natural Landmarks Based on Clouds of 3D Edge-Points from RGB-D Sensors for Mobile Robot Localization. *Sensors* **2017**, *17*, 1–16.
47. Yazdi, M.; Bouwmans, T. New Trends on Moving Object Detection in Video Images Captured by a moving Camera: A Survey. *Comput. Sci. Rev.* **2018**, *28*, 157–177. [CrossRef]
48. Vertens, J.; Valada, A.; Burgard, W. SMSnet: Semantic Motion Segmentation using Deep Convolutional Neural Networks. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2018.
49. Tan, W.; Liu, H.; Dong, Z.; Zhang, G.; Bao, H. Robust Monocular SLAM in Dynamic Environments. In Proceedings of the IEEE International Symposium on Mixed & Augmented Reality, Adelaide, SA, Australia, 1–4 October 2013.
50. 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; pp. 375–380.
51. Salas-Moreno, R.F.; Glocken, B.; Kelly, P.H.J.; Davison, A.J. Dense Planar SLAM. In Proceedings of the 2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Munich, Germany, 10–12 September 2014; pp. 157–164.
52. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A Benchmark for the Evaluation of RGB-D SLAM Systems. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, Vilamoura, Portugal, 7–12 October 2012.