Received: 11 January 2021 | Revised: 8 February 2021 | Accepted: 13 March 2021

DOI: 10.1049/csy2.12019

ORIGINAL RESEARCH PAPER

RGB-D SLAM with moving object tracking in dynamic environments

Weichen Dai[1] | Yu Zhang[1] | Yuxin Zheng[1] | Donglei Sun[2] | Ping Li[1]

[1]State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou, China

[2]Centre for English Language Education, University of Nottingham Ningbo China, Ningbo, China

Correspondence

Yu Zhang, State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China.
Email: zhangyu80@zju.edu.cn

Funding information

The National Natural Science Foundation of China, Grant/Award Number: 62088101; 61673341; Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China, Grant/Award Number: ICT2021A10; China's Double First-class plan; National Key R&D Programme of China, Grant/Award Number: 2016YFD0200701-3

Abstract

Simultaneous localization and mapping methods are fundamental to many robotic applications. In dynamic environments, SLAM methods focus on eliminating the influence of moving objects to construct a static map since they assume a static world. To improve localization robustness in dynamic environments, an RGB-D SLAM method is proposed to build a complete 3D map containing both static and dynamic maps, the latter of which consists of the trajectories and points of the moving objects. Without any prior knowledge of the moving targets, the proposed method uses only the correlation between map points to discriminate between the static scene and the moving objects. After the static points are determined, camera motion estimation is performed only on reliable static map points to eliminate the influence of moving objects. The motion of the moving objects will then be estimated with the obtained camera motion by tracking their dynamic points in subsequent frames. Moreover, multiple groups of dynamic points that belong to the same moving object are fused by a volume overlap checking step. Experimental results are presented to demonstrate the feasibility and performance of the proposed method.

1 | INTRODUCTION

In recent years, simultaneous localization and mapping (SLAM) methods [1–3] have played an essential role in robot navigation, thanks to low-cost and lightweight cameras, especially the advent of RGB-D sensors [4]. In dynamic environments where there are moving objects in front of the static background, most existing SLAM methods still assume a static or mostly static environment. In practice, these methods will underperform or even fail due to the moving objects in the field of view (FOV) [5, 6], for the reason that the map points on the same moving object have the same motion pattern, which influences the robust estimation methods, such as random sample consensus (RANSAC).

To improve the robustness in dynamic environments most SLAM methods treat moving objects as outliers and utilise static scenes to estimate the camera's motion [7, 8]. Nonetheless, some existing SLAM methods show that tracking moving objects and SLAM are mutually beneficial [9]. If the moving objects can be tracked successfully their corresponding image regions can be determined to eliminate their adverse influence on camera motion estimation. Meanwhile, good dynamic point tracking performance relies on the accuracy of camera motion estimation. Most state-of-the-art methods are built upon detection tools that require prior knowledge [10, 11]. Even though these detection tools can directly estimate the boundary or mask of movable objects in the image, they have shortcomings, one of which is that they cannot recognise the objects not included in the training set or without a 3D shape model, limiting the application of these methods to general scenarios.

In this paper, to facilitate robust camera motion estimation, a SLAM method that does not require prior knowledge is proposed to track a moving object in dynamic environments. The map points on the moving objects and the static scene are named dynamic and static points, respectively. To distinguish between these two types of points, point-correlation-based segmentation is used, leveraging the property that the relative

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2021 The Authors. *IET Cyber-systems and Robotics* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology and Zhejiang University Press.

position between static points is consistent over time but inconsistent between dynamic and static points and hence there should be no correlation between the measurements of dynamic and static points. After the segmentation, camera motion estimation is performed using only those reliable static map points, thus ensuring the robustness of the camera motion estimation. With accurate camera motion estimation results the motion of moving objects is then estimated by tracking corresponding dynamic points. Meanwhile, the 3D volume overlap checking is also computed to fuse multiple dynamic points that belong to the same moving object.

The main contribution of this paper is as follows:

- A SLAM method which can track a moving object in dynamic environments is proposed.
- A volume overlap checking method is developed to fuse the dynamic map points that belong to the same moving object.
- The proposed method can track unknown moving objects without prior knowledge.

The rest of the paper is organised as follows. Related work is reviewed in Section 2. Details of the proposed SLAM system are presented in Section 3. Experimental results are discussed in Section 4 and conclusions are drawn in Section 5.

2 | RELATED WORK

Visual SLAM methods use images as input to compute the camera motion and map the static scene simultaneously. These methods can be categorised into filter-based [12, 13] and factor-graph-optimization-based [14, 15] methods. Since the latter are more accurate than the former most of the advanced visual methods fall into this category [16]. The methods in this category can be further divided into two subcategories: direct [17, 18] and indirect (feature-based) methods [19–21]. Since these methods assume that the world is static, so their robustness in highly dynamic environments has been challenged. The key to eliminating the influence of moving objects is to distinguish between them and the static background [22].

Most methods can locate the dynamic points with the same motion pattern on the same moving object. The reprojection difference between consecutive frames is used to determine the dynamic image regions from the static background in most works [23, 24]. Similarly, in study [25], by using the information from the Inertial Measurement Unit (IMU), incorrect visual information from the moving objects can be filtered out directly. Besides locating the whole dynamic image region some methods also seek to extract features on the moving objects [26, 27].

Different from RGB images depth images can reflect the difference between the foreground and background [28, 29]. Hence, if the foreground is treated as a moving object, the dynamic image regions can be determined using image segmentation methods.

Most of the recent frameworks utilise object detection methods. These methods employ deep-learning-based detection or segmentation approaches to provide 2D bounding boxes [10] or masks [30, 31]. Moreover, if the target's 3D object model is known, the object can be detected directly [32, 33]. Based on the prior knowledge, this type of approach works well in environments that only contain known objects but will usually fail when encountering unknown objects.

The above methods can provide notable performance in dynamic environments, but the moving objects' dynamic information is only treated as outliers and not further exploited. Apart from the static information in the environment, moving objects are also essential components that have not been well explored in SLAM. Moreover, most of these methods only apply to RGB-D sensors, which can provide 3D perception directly.

To utilise the information from moving objects, some methods try to track these objects after they are detected. For this purpose two challenges need addressing. On the one hand, the camera motion estimation will be influenced adversely by or even fail due to the unrecognised dynamic image regions associated with the moving objects. On the other hand, to determine these dynamic image regions on the same object from different images is challenging. Therefore, most frameworks use learning-based methods to solve both challenges. With the 2D masks or bounding boxes obtained from these methods, 3D box inference can be further made [10, 11]. The features extracted from the bounding can be used to track the targets [34–36]. Moreover, some end-to-end learning methods can be applied with SLAM to track moving objects directly [37–39].

3 | RGB-D SLAM WITH MOVING OBJECTS TRACKING IN DYNAMIC ENVIRONMENTS

The proposed method is developed based on a classical feature-based SLAM method [20] with added steps to track dynamic points, as depicted in Figure 1. In addition to maintaining a static map the proposed method can also construct object targets. Similar to a static map which contains all camera keyframes and static points for camera motion estimation, each object target consists of its corresponding dynamic points and keyposes to represent a moving object.

The proposed framework consists of front-end tracking and back-end optimization. The tracking component uses consecutive images to estimate camera motion and track object targets based on back-end segmentation results. Similar to the definition of a keyframe, to improve object mapping efficiency, an object pose that significantly differs from the previous one is defined as a keypose. The back-end optimization includes two parts: static scene optimization and object optimization. The former focuses on obtaining the optimal static map with global consistency, where the local mapping thread optimises the static world map using multiple keyframes. If there are moving objects in the static map, the segmentation thread will then partition the corresponding dynamic points from the static points and create a new object target to represent the moving object. In the object optimization, object mapping

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

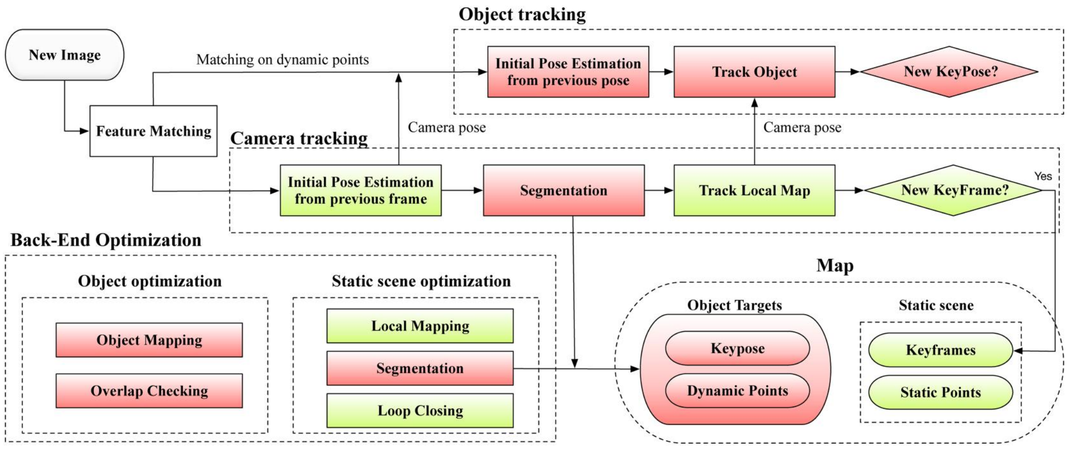

FIGURE 1 System overview. The steps and map resources in red blocks are added to track moving objects. Besides, those in green are implemented using classic feature-based methods [20] for camera motion estimation and static scene mapping

optimises both the keyposes and the structure of the object targets based on dynamic points. Besides, an overlap checking is conducted to determine whether multiple object targets belong to the same moving object.

3.1 | Notations

Before the proposed method is discussed in detail, notations and symbols are defined first, as shown in Figure 2. For the k-th data frame, the pose of the camera frame and the g-th object target are represented as $\mathbf{T}_k^c$ and $\mathbf{T}_k^g$, respectively. $\mathbf{p}_i^s$ is the i-th homogeneous point denoting the position of the i-th static point, and $\mathbf{p}_j^g$ is the j-th homogeneous point associated with the g-th object target. All position states of the static points are denoted by $\mathbf{P}^s$, and all position states of the dynamic points associated with the g-th object target are denoted by $\mathbf{P}^g$. $\mathbf{O}$ denotes the set of all object targets.

The i-th 3D point $\mathbf{p}_i$ in the world frame is projected to the k-th frame coordinates with a transformation $\mathbf{T}_{k,w} \in SE(3)$ using

$$\mathbf{y}_{ik} = \mathbf{g}(\mathbf{T}_k \cdot \mathbf{p}_i) + \mathbf{n}_{ik}, \tag{1}$$

where $\mathbf{y}_{ik}$ represents the measurement result of the i-th point in the k-th frame, $\mathbf{g}(\cdot)$ is the RGB-D camera observation model and $\mathrm{n}_{\mathrm{ik}} \sim (0, \mathrm{C}_{\mathrm{ik}})$ is an additive Gaussian noise. Since the RGB-D sensor can capture 3-D information, we have $\mathbf{y}_{ik} = [u, v, d]^\top$, where (u, v) is the corresponding feature position in the RGB image and d is the corresponding depth measurement.

3.2 | Segmentation and object target creation

Before a moving object can be tracked, the associated dynamic points must be identified in the constructed map. For that purpose, a point-correlation-based segmentation method [40]

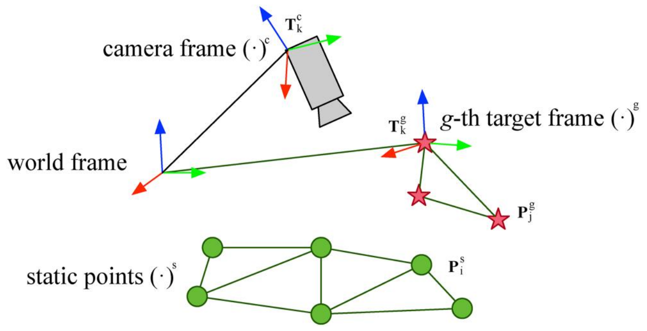

FIGURE 2 Notations of frames and points at time k

is employed to isolate dynamic points from static ones, and the segmented dynamic points with the same motion pattern are treated as a new moving target.

The segmentation runs within the static scene optimization process to differentiate between dynamic and static points. Meanwhile, the segmentation is also applied to camera tracking to ensure that the dynamic points with fast motion can be identified in time. Therefore, the object targets with dynamic points are created in both parts of the system.

As shown in Figure 3, the segmentation method has three steps: (1) a graph is built to represent the correlations of adjacent map points; (2) inconsistent edges without point correlations in the graph are removed in the optimization process; and (3) a search is conducted to check the connectedness of the graph to identify the segmented static and dynamic points. As a result, those points in the same point group will have motion consistency, while those in different groups do not.

3.2.1 | Building a graph

In this step, a graph is built based on the current static points. To avoid the complexity of checking point correlations between all point pairs, the Delaunay Triangulation [41] is applied on $\mathbf{P}^s$ to build a graph $\mathscr{G}$ considering only correlations between adjacent points. In this graph, each edge is indexed as a

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

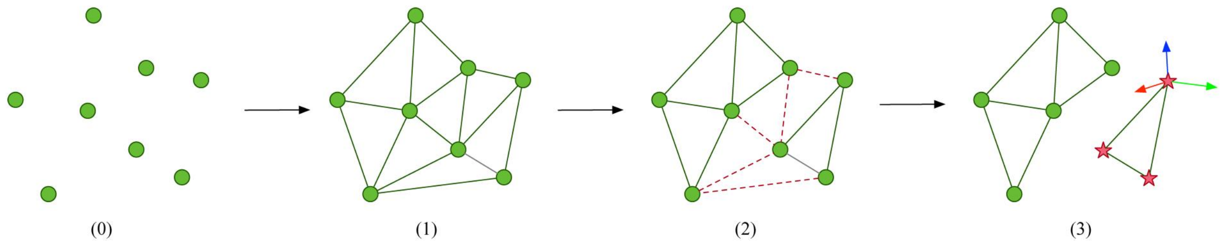

FIGURE 3 Steps to create object targets: (0) existing static points (shown as green dots) are taken as input; (1) based on these static points, a graph is created using Delaunay triangulation to represent potential point correlations; (2) the inconsistent edges in red dotted lines are removed from the graph to separate dynamic points from static points; (3) the connectedness of the graph is checked to determine the connected dynamic points, represented by the red stars, and a new object target is created

state set $\mathbf{l}_{ij}$ to represent the potential correlation between $\mathbf{p}_i^s$ and $\mathbf{p}_j^s$:

$$\mathbf{l}_{ij} = \{\mathbf{p}_i^s, \mathbf{p}_j^s\}. \tag{2}$$

Meanwhile, these edges will be checked in the following step.

3.2.2 | Removing inconsistent edges

In this step, the inconsistent edges will be removed through optimization based on point correlations. The states to be optimised are P^s and all the measurements in K data frames can be denoted by

$$\mathbf{z} = \{\mathbf{z}_{ijk}\}, \tag{3}$$

where $k = 1, ..., K$, and ij corresponds to $\mathbf{l}_{ij} \in \mathscr{G}$. Each measurement of the point correlation $\mathbf{l}_{ij}$ can be calculated by

$$\mathbf{z}_{ijk} = \mathbf{y}_{ik} - \mathbf{y}_{jk}. \tag{4}$$

After all available measurements have been captured an optimization problem can be formulated as

$$\min_{\mathbf{P}^s} \sum_{\mathbf{z}_{ijk} \in \mathbf{z}} \left\| \mathbf{z}_{ijk} - [\mathbf{g}(\mathbf{T}_k^c \mathbf{p}_i^s) - \mathbf{g}(\mathbf{T}_k^c \mathbf{p}_j^s)] \right\|_\delta, \tag{5}$$

where $\|\cdot\|_\delta$ is the Huber norm. This residual is defined as the sum of the error, which is the difference between the reprojected distance and the observed feature distance.

During the optimization iterations the measurements with a large error are considered outliers and removed. When the optimization completes, if all measurements of an edge in $\mathscr{G}$ are removed as outliers, this edge will be considered as an inconsistent edge and removed from $\mathscr{G}$. After all inconsistent edges have been removed, if there are still dynamic points in the map, the remaining graph should be divided into multiple connected components to represent the segmentation result.

3.2.3 | Creating object targets

After $\mathscr{G}$ has been divided, the deep-first-searching algorithm is used to determine the connected components. Among all connected components, the one with the largest volume is kept as static points while the remaining points in all other connected components are classified as dynamic points. Each connected component containing dynamic points will be created as a new object target to represent moving objects and added to $\mathbf{O}$. Meanwhile, the centre of these dynamic points are used to initialise the pose of this object target. The initial pose of the g-th object target is denoted by $\mathbf{T}_0^g$.

Different point groups obtained from the same moving object may represent different object targets temporally and will be integrated by the subsequent volume overlap checking step. Based on the segmentation result, the camera motion estimation is only applied to static points to eliminate the moving objects' influence. Once the correct camera poses are obtained, the poses of the object targets will be estimated by object-level bundle adjustment using the matching results of dynamic points and image features.

3.3 | Camera and object tracking

Figure 4 depicts the two tasks in the front-end tracking: camera tracking and object tracking. The former aims to estimate the camera motion and the latter is to estimate the motion of the object targets. When new images are coming from the camera, feature matching is applied to obtain correspondences from the tracked points of the previous frame. The matching results contain two types of features: the correspondences with static points and with dynamic points, respectively.

3.3.1 | Camera tracking

For camera tracking, only matched static points are used to estimate the camera's initial pose for the current frame. Using

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

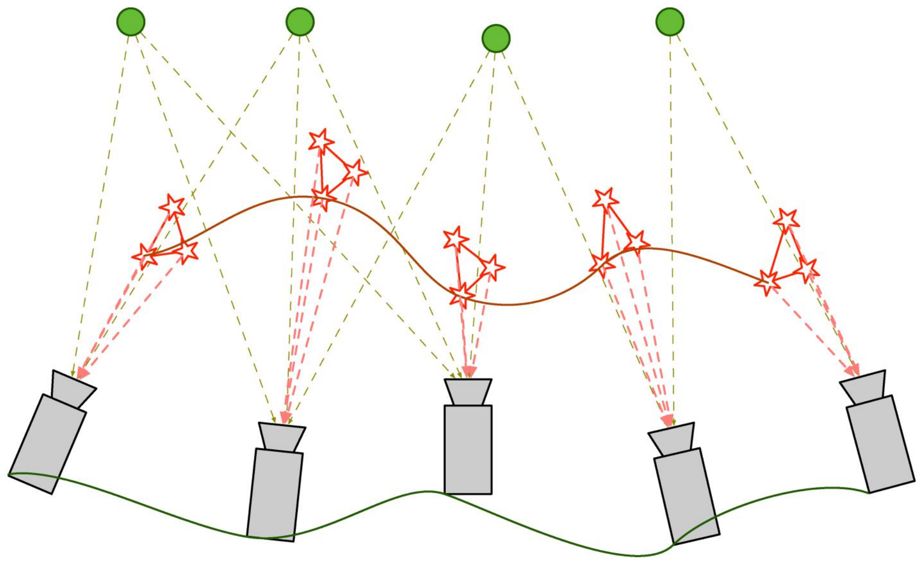

FIGURE 4 Illustrations of object tracking and camera motion estimation. The green line denotes the trajectory of the camera and the red line the trajectory of the object target. Four static points (in green) are used for camera motion estimation, and three dynamic points in the moving object target for tracking

the segmentation method introduced in Section 3.2 with two-frame data, if there are new moving objects, the dynamic points will be determined, and a new object target will be created after the segmentation. Then, camera tracking is conducted to optimise the camera's latest pose with supplementary static points from the local map. Finally, a keyframe check step will decide if this frame should be created as a new keyframe and added to the map. In the camera tracking, the camera pose $\mathbf{T}_k^c$ is estimated with I tracked map points in the k-th data frame using motion-only bundle adjustment, which can be formulated as

$$\mathbf{T}_k^c = \arg\min_{\mathbf{T}_k^c} \sum_i \left\| \mathbf{y}_{ik} - \mathbf{g}(\mathbf{T}_k^c \mathbf{p}_i^s) \right\|_\delta, \tag{6}$$

where $i = 1, ..., I$, and $\| \cdot \|_\delta$ is the Huber norm.

3.3.2 | Object tracking

For object tracking, once the camera's initial pose is obtained, the matched dynamic points will be used to compute their corresponding object target pose. After that, combining the latest camera pose, the object target's pose can be optimised with supplementary dynamic points on the same object target. Finally, if the tracked features change significantly in the current frame, this newest pose will be marked as a keypose and added to the object optimization. Similar to motion-only bundle adjustment, the tracking of the g-th object target with N tracked map points in the k-th frame can be defined as

$$\mathbf{T}_k^g = \arg\min_{\mathbf{T}_k^g} \sum_n \left\| \mathbf{y}_{nk} - \mathbf{g}\left(\mathbf{T}_k^c \left(\mathbf{T}_k^g\right)^{-1} \mathbf{T}_0^g \mathbf{p}_n^g\right) \right\|_\delta, \tag{7}$$

where $n = 1, ..., N$, and $\mathbf{T}_0^g$ is the initial pose of the g-th object target when it is created. This process can be named as motion-only object-level bundle adjustment.

3.4 | Static scene optimization

The static scene optimization estimates the optimal static local map. Since there might be slow-moving objects, the point-correlation-based segmentation method introduced in Section 3.2 is also used here to determine static points. Moreover, local mapping and loop closing are also conducted to keep the global consistency of the static map.

3.4.1 | Local mapping

After the dynamic points have been identified, the remaining static points are used to estimate the camera motion and construct the static scene map. If there are K frames and M points in the map, the states to be estimated can be represented as

$$\mathbf{x} = \{\mathbf{T}_1^c, ..., \mathbf{T}_k^c, \mathbf{p}_1^s, ..., \mathbf{p}_M^s\}. \tag{8}$$

Then the bundle adjustment problem of estimating $\mathbf{x}$ can be formulated as a nonlinear least-square problem:

$$\min_{\mathbf{x}} \sum_{i,k} \left\| \mathbf{y}_{ik} - \mathbf{g}(\mathbf{T}_k^c \mathbf{p}_i^s) \right\|_\delta. \tag{9}$$

The visual residual is defined as the sum of the reprojection error between all the projected 3D map points and the observed 2D features with depth information.

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

In the implementation, full bundle adjustment is used to optimise the local map in the static scene optimization within co-visibility keyframes. The optimization can reduce the drift of estimation and ensure the accuracy of the full system. In the camera tracking, the more efficient motion-only bundle adjustment is used to estimate each frame pose.

3.5 | Object optimization

The shape and trajectory of the moving object are optimised with the full object-level bundle adjustment. Because the FOV of the camera changes, some features on the moving objects are being created repeatedly. Therefore, an overlap checking is proposed to check the collision among different object targets and merge them.

3.5.1 | Object mapping

Since the moving object targets consist of dynamic points, tracking these points can help estimate the motion of these object targets. This process is called object-level bundle adjustment, assuming that the moving object is rigid. It should be noted that the states of the object targets' poses and of the corresponding dynamic points are estimated simultaneously. Since the estimated camera poses are used to estimate the object targets' poses the accuracy of the camera motion also influences that of the object tracking.

For the g-th object target, if there are K frames and N corresponding dynamic points, then alright the states to be estimated are defined as follows:

$$\mathbf{x}^g = \{\mathbf{T}_1^g, \ldots, \mathbf{T}_K^g, \mathbf{p}_1^g, \ldots, \mathbf{p}_N^g\}. \tag{10}$$

With $\mathbf{x}^g$, the full object-level bundle adjustment problem can also be formulated as a nonlinear least-squares problem as

$$\min_{\mathbf{x}^g} \sum_{n,k} \left\| \mathbf{y}_{nk}^g - \mathbf{g}\left(\mathbf{T}_k^c \left(\mathbf{T}_k^g\right)^{-1} \mathbf{T}_0^g \mathbf{p}_n^g\right) \right\|_\delta, \tag{11}$$

where $k = 1, \ldots, K$, $n = 1, \ldots, N$, and $\mathbf{y}_{ik}^g$ is the measurement of the n-th dynamic point in the k-th image frame. $\mathbf{T}_0^g$ is obtained from the reference frame where this object target was created. The residual is still defined as the sum of the reprojection error between the projected 3D map points and the observed 2D features with depth information.

3.5.2 | Overlap checking

The new points on the moving objects are first initialised as static points and then determined as dynamic points in the segmentation discussed in Section 3.2. However, features that are not successfully tracked on the same moving object are repeatedly created as new static points and added to the map. These new map points may be created as new object targets in the subsequent segmentation. Hence, it is essential to propose a mechanism to check whether a new object target belongs to the existing ones since prior knowledge is unavailable. In this process, the object targets that are part of the same object will be merged to achieve object construction and motion estimation.

In practice, the duplicate creation of map points is caused by the change of the object's viewpoint in the camera. Since the image texture at the same 3D position changes, the previously constructed features cannot produce a good matching performance. Consequently, as shown in Figure 5(a), the duplicate object targets that belong to the same object will overlap in the 3D volume. If V_g denotes the 3D volume of the Delaney triangulation using the points of the g-th object target, then the mechanism for checking the overlap of two targets can be defined as

$$d = V_g + V_{g+1} - V_{g,g+1}, \tag{12}$$

where $V_{g,g+1}$ denotes the 3D volume of the Delaney triangulation using the points of both the g-th and the $(g+1)$-th object target. If $d > 0$, there is overlap; otherwise, there is not. Based on this formulation, if a new object target overlaps with an existing one, it will be fused into the existing one to obtain a more integrated object target, as shown in Figure 5(a,b).

4 | EXPERIMENTS

In the following experiments the proposed method will be evaluated in indoor environments. These experiments aim to verify that the proposed algorithm can estimate camera motion under the influence of moving objects. Furthermore, they also demonstrate that the method provides a feasible solution to tracking. Since the proposed method assumes that the moving object is rigid, the TUM RGB-D benchmark [42], which contains persons but does not provide the ground-truth of the person, cannot be used for evaluation. Meanwhile, there are no existing similar formulations or datasets. Therefore, the estimated camera poses will be evaluated against the ground truth obtained from the OptiTrack system. The tracking result of the moving objects will be compared with the ARUCO marker [43].

The RGB-D sensor used here is an Asus Xtion Pro, which can capture both RGB and depth images. Since the feature-based motion estimation needs at least 20 inliers in practice, the object cannot be too far away from the camera. Meanwhile, due to the limited number of OptiTrack cameras available, the camera can only move in a minimal space. In this setup, the object and the camera are moved by a person's two hands.

4.1 | Quantitative evaluation

This experiment is conducted to show the accuracy of the proposed method. In this experiment, the moving object is a

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

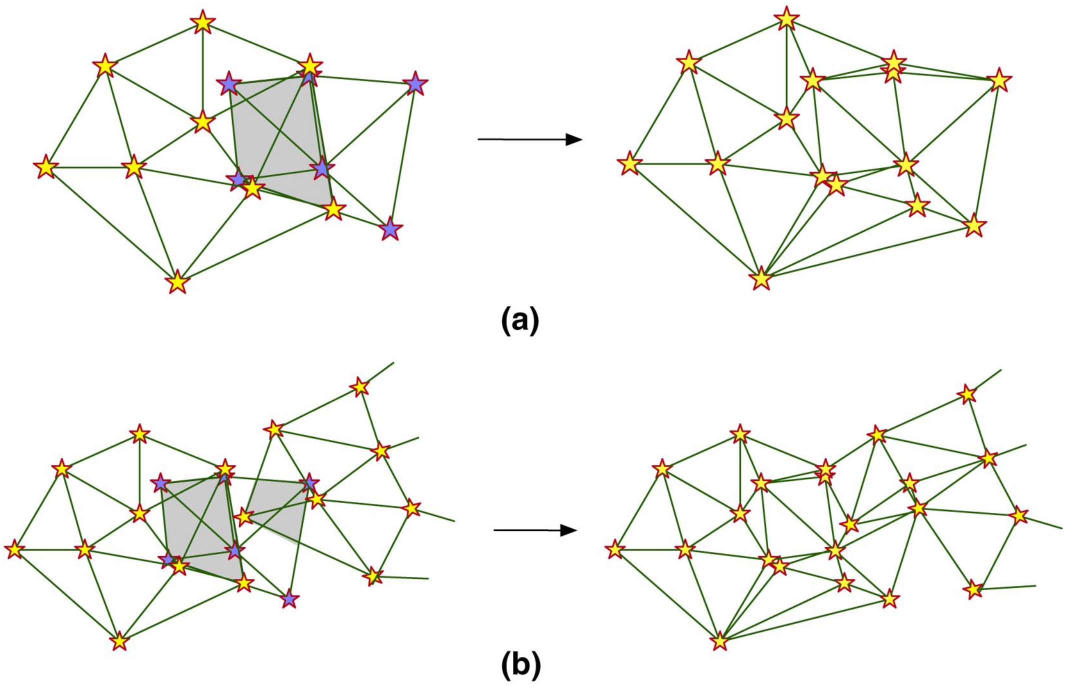

FIGURE 5 Illustration of overlap checking. The purple stars represent new dynamic points while the yellow one the existing dynamic points. The green edges are obtained using Delaunay triangulation. (a) The new dynamic points that belong to the new object target overlap with the existing dynamic points associated to an existing object; (b) Although there is no overlap between two object targets that are far away, an intermediate object target can be used to connect them and thus complete the fusion

marker, as shown in Figure 6. The target and the camera are moving simultaneously.

To compare the tracking results with the ground truth from the OptiTrack system, the estimated camera trajectory is aligned with the ground truth. As shown in Figure 7, the absolute trajectory error (ATE) result is 0.0221 m. For comparison, the ORB-SLAM2 method [20], originally designed for static environments and based on which the current method was developed, is employed as a reference result. The ATE of ORB-SLAM2 is 0.1359 m. The result shows that the proposed method with object tracking can provide robust performance in dynamic environments.

For object tracking, the result of the proposed method is compared with the ground truth obtained from the ARUCO marker with the OptiTrack system, as shown in Figure 8. The ATE result is 0.0333 m. Generally, the accuracy of object tracking depends on the accuracy of camera tracking. Since object tracking is based on camera motion estimation, the proposed 6-DoF object tracking performance is acceptable. Figure 8 also shows that the proposed method produces an unbroken trajectory using overlapping checking, indicating its robustness.

4.2 | Tracking unknown objects

As shown in Figure 9, this experiment is used to prove that the proposed method can track unknown objects in the

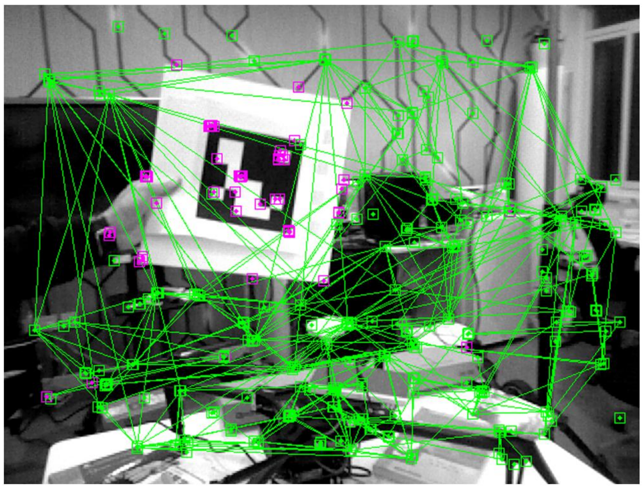

FIGURE 6 Tracking the marker. The edges between static points are shown as green lines. The features on the moving object and in the static scene are shown in purple and green, respectively. In the experiment, the marker has been correctly detected and tracked

environment. In this experiment, a book was randomly chosen. The camera and the book were moved manually in the environment.

This experiment proves that the proposed method does not require any prior knowledge for detection. Meanwhile, the feasibility of the proposed overlap checking mechanism is also demonstrated. As shown in Figure 10, since the new map points are fused into the

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

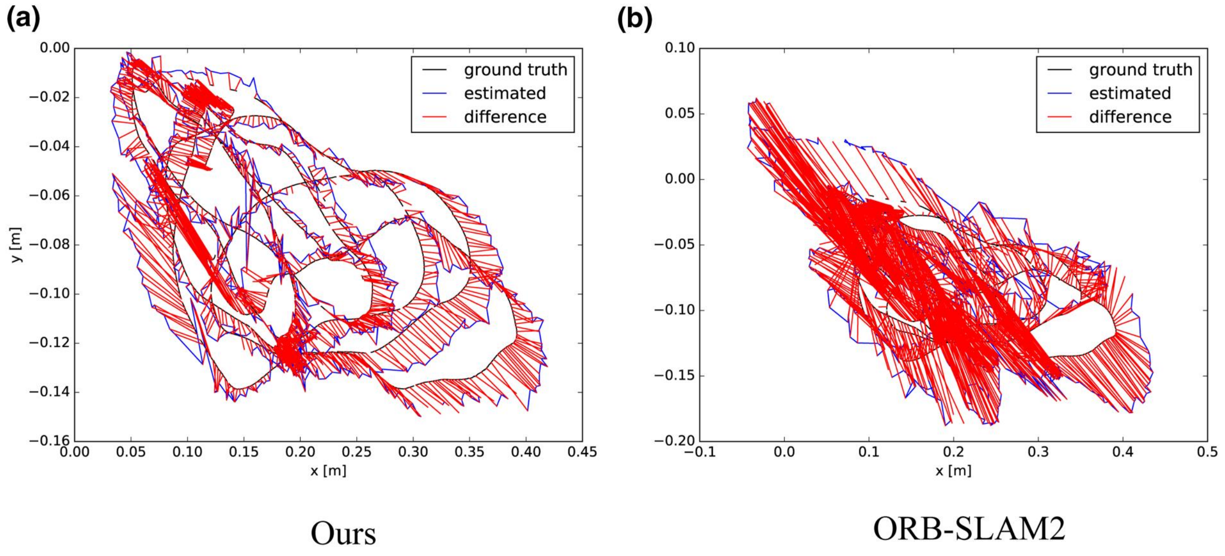

FIGURE 7 Comparison of the estimated camera trajectories with the ground truth: (a) trajectories estimated with the proposed method. The absolute trajectory error (ATE) of the trajectory obtained from the proposed method is 0.022137 m. (b) Trajectories estimated with ORB-SLAM2. The ATE of ORB-SLAM2 is 0.135959 m

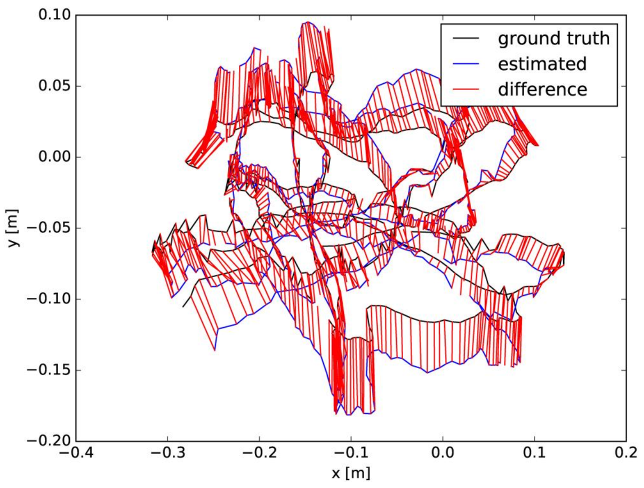

FIGURE 8 Comparison between the estimated trajectory and the trajectory provided by the ARUCO marker

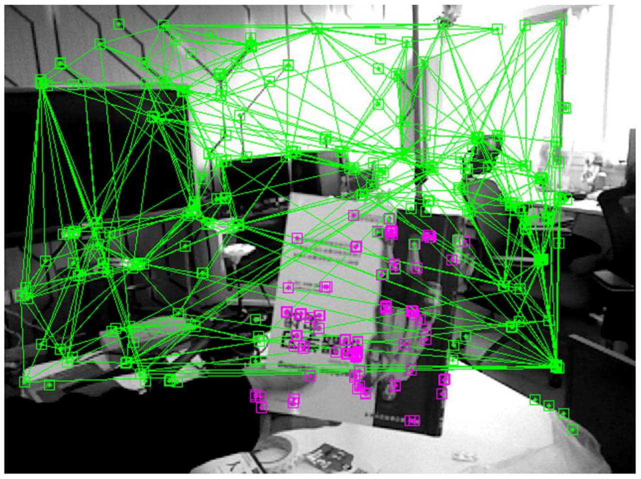

FIGURE 9 Tracking a book. The edges between static points are shown in green while the features on the moving object and in the static scene are shown in purple and green, respectively. Results show that the book has been correctly detected and tracked

existing object in the system, there is only one trajectory in blue in the space.

4.3 | Analysis of the real-time performance

In this section, the statistics of runtime are presented to evaluate the efficiency of the object-level bundle adjustment. The experiment was performed on a desktop computer with Intel Core i9-9900k processor and 32 GB RAM. Since the running time of the bundle adjustment is related to the amount of data, the time required for each bundle adjustment optimization and the number of corresponding map points used in the optimization were collected.

As shown in Table 1, the conventional local bundle adjustment (local mapping), which optimises the static points and camera poses, needs more computational resources compared to the object-level bundle adjustment (object mapping). The reason is that the local bundle adjustment optimises much more variables than the object bundle adjustment. Therefore, to show the real-time performance of these two types of optimization using the same number of map points, we evaluated the efficiency of the object-level bundle adjustment by measuring the time cost concerning the number of map points. As shown in Figure 11, using the same number of points, since the observation model used by object BA is three-dimensional mapping rather than reprojections, the

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

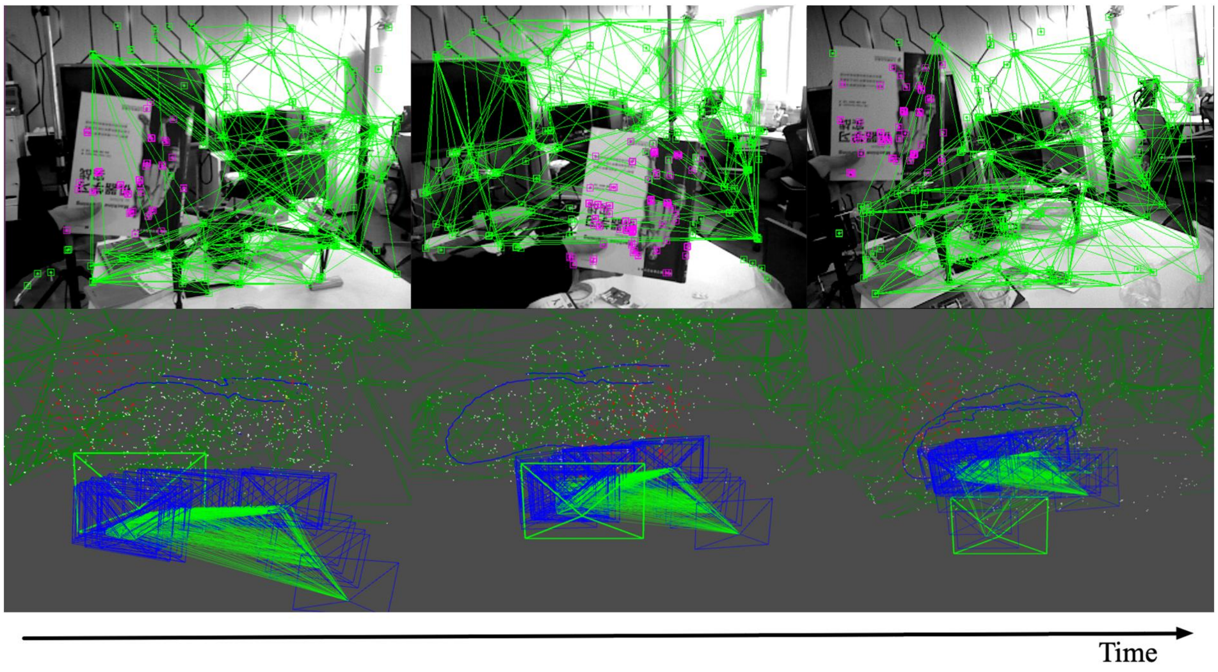

FIGURE 10 The book tracking results over time. The book was picked up randomly from the table and was an unknown object for the method

TABLE 1 Statistical results of real-time performance

Time/Module	Object BA	Local BA
Medium [ms]	6.75	86.71
Mean [ms]	5.07	85.14
Std. [ms]	2.92	53.09

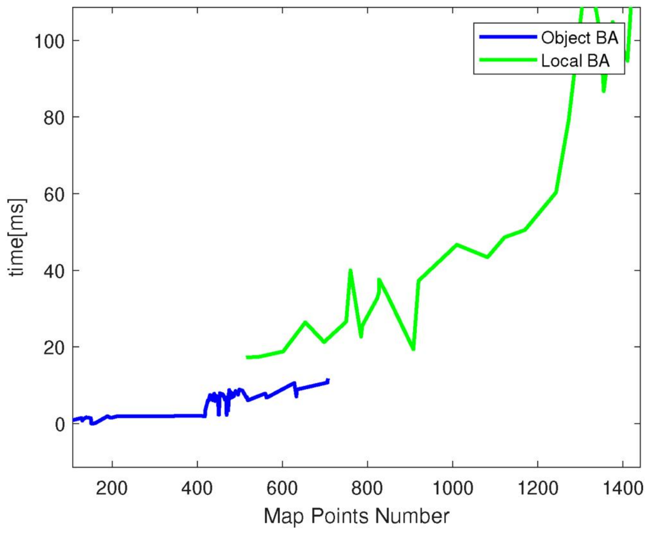

FIGURE 11 Efficiency comparison between local and object-level bundle adjustment using the same number of map points

efficiency of object BA is still better than local bundle adjustment. The figure also shows that the trends of the two lines are close, implying similar real-time performance of the two methods.

5 | CONCLUSIONS

In this paper, a new SLAM method with moving object tracking using RGB-D cameras was proposed. The creation, tracking, and fusion of object targets can be completed without prior knowledge. With the tracking of moving objects, the camera motion estimation will not be influenced by them. Experimental results demonstrated an acceptable estimation accuracy of the proposed method. Besides, tracking an unknown moving object can also be achieved.

However, the proposed method still has limitations: (1) since the proposed method assumes that the moving objects are rigid, it is not easy to apply in the environments where non-rigid objects exist; (2) the proposed method does not build a detailed dynamic point management mechanism or an object-level loop-closing detection method, so it is challenging for the proposed method to track objects for a long time in a large-scale environment.

In future work, object-level loop closure will be implemented to further improve the accuracy and robustness, and a detailed dynamic point maintenance mechanism can be designed to make the system maintain a compact map of moving objects.

ACKNOWLEDGMENTS

This work was supported by NSFC 62088101 Autonomous Intelligent Unmanned Systems, the National Natural Science Foundation of China (Grant No.61673341), National Key R&D Programme of China (2016YFD0200701-3), Double First Class University Plan (CN), the Project of State Key

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

Laboratory of Industrial Control Technology, Zhejiang University, China (No.ICT2021A10).

ORCID

Weichen Dai https://orcid.org/0000-0002-5019-2755
Yu Zhang https://orcid.org/0000-0002-0043-4904
Donglei Sun https://orcid.org/0000-0001-9141-0790

REFERENCES

1. Fuentes-Pacheco, J., Ruiz-Ascencio, J., Rendón-Mancha, J.M.: Visual simultaneous localization and mapping: a survey. Artif. Intell. Rev. 43(1), 55–81 (2015)
2. Stachniss, C., Leonard, J.J., Thrun, S.: Simultaneous localization and mapping. In: Springer Handbook of Robotics, pp. 1153–1176. Springer: Berlin (2016)
3. Martz, J., Al-Sabban, W., Smith, R.N.: Survey of unmanned subterranean exploration, navigation, and localisation. IET cyber-syst. Robot. 2(1), 1–13 (2020)
4. Freedman, B., et al.: Depth Mapping Using Projected Patterns. Google Patents, US Patent 8,150,142 (2012)
5. Cadena, C., et al.: Past, present, and future of simultaneous localization and mapping: towards the robust-perception age. IEEE Trans. Robot. 32(6), 1309–1332 (2016)
6. Saputra, M.R.U., Markham, A., Trigoni, N.: Visual SLAM and structure from motion in dynamic environments. ACM Comput. Surv. 51(2), 1–36 (2018)
7. Bârsan, I.A., et al.: Robust dense mapping for large-scale dynamic environments. In: Proceedings of 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 7510–7517. IEEE, Brisbane Convention & Exhibition Centre, Brisbane (2018)
8. Fan, Y., et al.: Dynamic objects elimination in slam based on image fusion. Pattern Recogn. Lett. 127:191–201 (2019)
9. Wang, C.-C., et al.: Simultaneous localization, mapping and moving object tracking. Int. J. Robot. Res. 26(9), 889–916 (2007)
10. Li, P., Qin, T., et al.: Stereo vision-based semantic 3d object and ego-motion tracking for autonomous driving. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 646–661. Munich (2018)
11. Yang, S., Scherer, S.: Cubeslam: monocular 3-D object slam. IEEE Trans. Robot. 35(4), 925–938 (2019)
12. Chiuso, A., et al.: Structure from motion causally integrated over time. IEEE Trans. Pattern Anal. Machine Intell. 24(4), 523–535 (2002)
13. Davison, A.J., et al.: Monoslam: real-time single camera slam. IEEE Trans. Pattern. Anal. Mach. Intell. 29(6), 1052–1067 (2007)
14. Mouragnon, E., et al.: Real time localization and 3d reconstruction. In: Proceedings of 2006 IEEE Computer Society Conference on vol. 1 Computer Vision and Pattern Recognition, pp. 363–370. IEEE, New York (2006)
15. Klein, G., Murray, D.: Parallel tracking and mapping for small ar workspaces. In: Proceedings of 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234. IEEE, Nara (2007)
16. Strasdat, H., Montiel, J.M.M., Davison, A.J.: Visual slam: why filter? Image Vis. Comput. 30(2), 65–77 (2012)
17. Kerl, C., Sturm, J., Cremers, D.: Dense visual slam for RGB-D cameras. In: Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2100–2106. IEEE, Tokyo (2013)
18. Bergmann, P., Wang, R., Cremers, D.: Online photometric calibration of auto exposure video for realtime visual odometry and slam. IEEE Robot. Automat. Lett. 3(2), 627–634 (2017)
19. Endres, F., et al.: 3-D mapping with an rgb-d camera. IEEE Trans. Robot. 30(1), 177–187 (2013)
20. Mur-Artal, R., Tardós, J.D.: Orb-slam2: an open-source slam system for monocular, stereo, and RGB-D cameras. IEEE Tran. Robot. 33(5), 1255–1262 (2017)
21. Brambley, G., Kim, J.: Unit dual quaternion-based pose optimisation for visual runway observations. IET Cyber-Syst. Robot. 2(4), 181–189 (2020)
22. Cheng, J., Sun, Y., Meng, M.Q.-H.: Improving monocular visual slam in dynamic environments: an optical-flow-based approach. Adv. Robot. 33(12), 576–589 (2019)
23. Kim, D.-H., Kim, J.-H.: Effective background model-based rgb-d dense visual odometry in a dynamic environment. IEEE Trans. Robot. 32(6), 1565–1573 (2016)
24. Alcantarilla, P.F., et al.: On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In: Proceedings of 2012 IEEE International Conference on. Robotics and Automation (ICRA), pp. 1290–1297. IEEE, St Paul (2012)
25. Kim, D.-H., Han, S.-B., Kim, J.-H.: Visual odometry algorithm using an RGB-D sensor and imu in a highly dynamic environment. In: Proceedings on International Conference on Robot Intelligence Technology and Application, pp. 11–26. Beijing (2015)
26. Tan, W., et al.: Robust monocular slam in dynamic environments. In: Proceedings of 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 209–218. IEEE, Adelaide (2013)
27. Li, S., Lee, D.: Rgb-d slam in dynamic environments using static point weighting. IEEE Robot. Autom. Lett. 2(4), 2263–2270 (2017)
28. Azartash, H., Lee, K.R., Nguyen, T.Q.: Visual odometry for RGB-D cameras for dynamic scenes In: Proceedings of 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1280–1284. IEEE, Florence (2014)
29. Sun, Y., Liu, M., Meng, M.Q.-H.: Improving rgb-d slam in dynamic environments: a motion removal approach. Robot Autonom. Syst. 89, 110–122 (2017)
30. He, K., et al.: Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2961–2969. Venice (2017)
31. Ye, L., Duan, T., Zhu, J.: Neural network-based semantic segmentation model for robot perception of driverless vision. IET Cyber-Syst. Robot. 2(4), 190–196 (2020)
32. Lim, J.J., Pirsiavash, H., Torralba, A.: Parsing ikea objects: fine pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2992–2999. Sydney (2013)
33. Zia, M.Z., et al.: Detailed 3D representations for object recognition and modelling. In: IEEE Trans. Pattern. Anal. Mach. Intell. 35(11), 2608–2623 (2013)
34. Qiu, K., et al.: Tracking 3-D motion of dynamic objects using monocular visual-inertial sensing. IEEE Trans. Robot. 35(4), 799–816 (2019)
35. Song, S., Chandraker, M.: Joint sfm and detection cues for monocular 3d localization in road scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3734–3742. Boston (2015)
36. Reddy, N.D., et al.: Dynamic body vslam with semantic constraints In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1897–1904. IEEE, Hamburg (2015)
37. Mousavian, A., et al.: 3d bounding box estimation using deep learning and geometry. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7074–7082. Hawaii (2017)
38. Xiang, Y., et al.: Data-driven 3d voxel patterns for object category recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1903–1911. Boston (2015)
39. Murthy, J.K., et al.: Reconstructing vehicles from a single image: shape priors for road scene understanding In: Proceedings of 2017 IEEE International Conference on Robotics and Automation (ICRA), 724–731. IEEE, Singapore (2017)

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

40. Dai, W., et al.: RGB-D slam in dynamic environments using point correlations. IEEE Trans. Pattern. Anal. Mach. Intell. (2020). https://ieeexplore.ieee.org/document/9145704
41. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.: The quickhull algorithm for convex hulls. ACM Trans. Math. Softw. 22(4), 469–483 (1996)
42. Sturm, J., et al.: A benchmark for the evaluation of RGB-D slam systems In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 573–580. IEEE, Algarve (2012)
43. Garrido-Jurado, S., et al.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recogn. 47(6), 2280–2292 (2014)

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Dai, W., et al.: RGB-D SLAM with moving object tracking in dynamic environments. IET Cyber-Syst. Robot. 3(4), 281–291 (2021). https://doi.org/10.1049/csy2.12019

26316315, 2021, 4, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/csy2.12019 by Readcube-Labtiva, Wiley Online Library on [08/11/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License