SLAM algorithm based on bounding box and deep continuity in dynamic scene

Baofu Fang* and Xiumeng Han

School of Computer Science and Information Engineering,
Intelligent Interconnected Systems Laboratory of Anhui Province,
Hefei University of Technology,
Hefei, Anhui 230009, China
Email: fangbf@hfut.edu.cn
Email: hanxiumengness@163.com
*Corresponding author

Zaijun Wang

Key Laboratory of Flight Techniques and Flight Safety,
Civil Aviation Flight University of China,
Guanghan, Sichuan 618307, China
Email: zaijunwang@cafuc.edu.cn

Xiaohui Yuan

Department of Computer Science and Engineering,
University of North Texas,
Denton, Texas 311277, USA
Email: xhyuan@unt.edu

Abstract: To solve the problems of low positioning accuracy and poor robustness caused by the inability of Simultaneous Localisation and Mapping (SLAM) algorithm to deal with dynamic targets in dynamic scenes, a SLAM algorithm based on bounding box and depth continuity in dynamic scenes is proposed. Firstly, the pixel random search filling process is carried out in combination with the depth and the bounding box to obtain the pixel-level segmentation result of the prior dynamic target. Then, the dynamic features are screened to eliminate the influence of dynamic targets. According to the screening results, relative static features which can be used for posture optimisation are selected. Posture optimisation is carried out by combining the static features to obtain the optimised camera posture. All experimental results show that the proposed algorithm significantly improves the positioning accuracy and real-time performance of SLAM algorithm in complex dynamic scenes.

Keywords: SLAM; simultaneous localisation and mapping; visual sensor; RGB-D perception; YOLO; object detection; dynamic scene; semantic information; depth continuity; data association; mapping.

Reference to this paper should be made as follows: Fang, B., Han, X., Wang, Z. and Yuan, X. (2021) 'SLAM algorithm based on bounding box and deep continuity in dynamic scene', *Int. J. Wireless and Mobile Computing*, Vol. 21, No. 4, pp.349–364.

Biographical notes: Baofu Fang received PhD degree in Computer Application Technology from Harbin Institute of Technology (HIT), China in 2013. He joined Department of Computer Science and Technology, School of Computer and Information, Hefei University of Technology in 2000, and an Associate Professor in 2010, and Master's Supervisor in 2011. His current research interests include multi robot/agent system, emotion/self-interest robot and machine learning. He is Technology Chair of Anhui Robot Competition, Member of Standing Committee of China Association of Artificial Intelligence (CAAI) Young Committee, Member of Standing Committee of China Association of Artificial Intelligence (CAAI) Robot and Culture Committee, He have many funds sponsored by National High Technology Research and Development Program of China (863), Natural Science Foundation of Anhui Province, Fundamental Research Funds for the Central Universities, Hefei University of Technology and some enterprise and company. He has published about 70 more research papers in journals and conference from 2002.

Xiumeng Han graduated from Chengdu University of Technology with a bachelor degree in computer science and technology. He is currently pursuing his ME in Computer Technology in the School of Computer and Information, Hefei University of Technology, Hefei, China. He won the first-class scholarship for graduate students in 2020 and 2021. His current research interests are SLAM, robotics, computer vision, etc.

Zaijun Wang received her MEng degree in Materials Science and Engineering from the Xihua University, China, in 2008. She is a Professor of the CAAC Academy of Flight Technology and Safety, Civil Aviation Flight University of China. Her research interests include multi robot task allocation, artificial intelligence and data mining.

Xiaohui Yuan received the BS degree in Electrical Engineering from Hefei University of Technology, China in 1996 and PhD degree in Computer Science from Tulane University in 2004. After his graduation, he worked at the National Institutes of Health on Medical Imaging and Analysis till 2006. He joined the University of North Texas (UNT) as an Assistant Professor in 2006 and was promoted to Associate Professor with tenure in 2012. His research interests include computer vision, data mining, machine learning, and artificial intelligence. He served as PI and co-PI in projects supported by Air Force Laboratory, National Science Foundation (NSF), Texas Advanced Research Program, Oak Ridge Associated Universities, and UNT. His research findings are reported in over 70 peer-reviewed papers. He is a recipient of Ralph E. Powe Junior Faculty Enhancement award in 2008 and the Air Force Summer Faculty Fellowship in 2011, 2012, and 2013. He is a member of IEEE and SPIE.

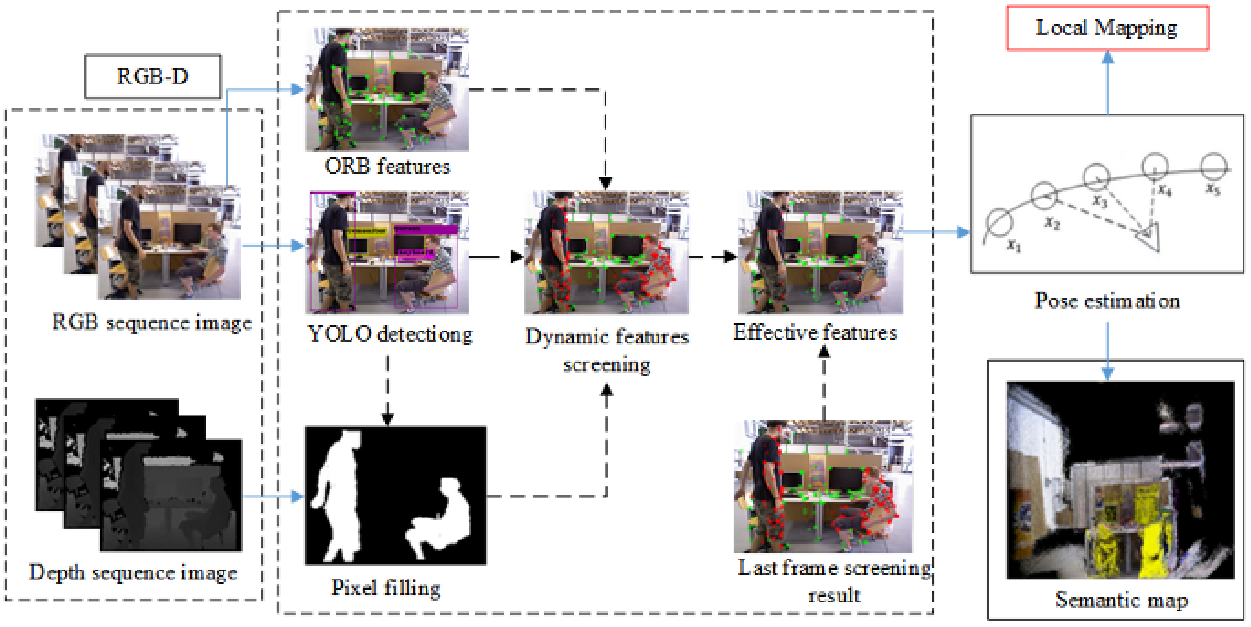
1 Introduction

In recent years, with the rapid development of mobile robots, autonomous driving, unmanned aerial vehicles and other technologies in the field of artificial intelligence, Simultaneous Localisation and Mapping (SLAM) technology, as one of the prerequisites for intelligent robots to complete advanced tasks, has been widely concerned by Du et al. (2018) and Fang et al. (2019).With the further reduction of camera manufacturing cost and the continuous enrichment of collected information, SLAM technology based on vision sensors has become a hot research topic (Fuentes-Pacheco et al., 2015)).

Visual SLAM obtains data from cameras (monocular, binocular, RGB-D cameras, etc.), and can obtain abundant visual texture information from the environment, to locate itself and perceive the environment. At present, mature visual SLAM algorithms include Oriented FAST and rotated BRIEF SLAM (ORB-SLAM, Mur-Artal and Tards, 2017), Direct Sparse Odometry (DSO, Engel et al., 2018), Large-Scale Direct-SLAM (LSD-SLAM, Engel et al., 2014), and so on. Although these systems have good results in pose estimation, they are all assumed to be in a static environment. However, the existence of moving objects will obviously reduce the accuracy of pose estimation of these visual SLAM algorithms, and even fail to locate them (Mur-Artal and Tards, 2017; Engel et al., 2018, 2014). Therefore, how to accurately and stably realise autonomous positioning of robots in dynamic scenes has gradually become an important research direction in SLAM field.

Some visual SLAM algorithms proposed at present (Sun et al., 2016; Kim and Kim, 2016; Wang and Huang, 2014) can be applied to dynamic scenes, but the traditional visual SLAM algorithms have slightly insufficient perception ability for dynamic objects and cannot obtain environmental semantic information. Because the cost of obtaining semantic information has been reduced in recent years, the SLAM algorithm combined with semantic information (Wang et al., 2021; Fang et al., 2021) has better adaptability to dynamic scenes, but its real-time performance cannot be guaranteed.

Figure 1 An overview of our system



To solve the above problems, we propose a SLAM algorithm based on bounding box and depth continuity in the dynamic scene. The flowchart of this algorithm is shown in Figure 1. It shows the operation process of our system: The original RGB image is used for the YOLO detection and ORB features extraction, and the YOLO detection result is used for dynamic object segmentation together with the original depth image. The segmentation results are used for dynamic feature elimination, and then effective dynamic feature detection is carried out. The detection results and static features are finally used for pose estimation. The main contributions are as follows:

- To deal with dynamic objects in the environment effectively and in real-time, we propose an online segmentation method of dynamic objects based on bounding box and depth continuity to obtain pixel-level segmentation results of prior dynamic objects, which can well deal with the influence of dynamic targets.
- To maximise the use of all the effective information in the field of vision to improve the accuracy of pose estimation, we propose a relatively static feature detection method based on feature Data association, which selects effective dynamic features which can be used for pose optimisation. Based on these works, a dense semantic map is constructed in the dynamic environment, which eliminates prior dynamic targets.

Section 2 introduces the related work, Section 3 introduces our main work in detail, Section 4 shows and analyses the experimental results and Section 5 concludes the paper with a summary.

2 Related works

The key of the SLAM algorithm in dynamic scene lies in whether the dynamic target in the visual field can be detected correctly, and how to deal with the dynamic target subsequently.

Yang et al. (2020) eliminated dynamic features by grouping pixel regions of keyframes and detecting motion consistency. Dai et al. (2020) used the correlation between map points to separate points belonging to static scenes and points belonging to different moving targets into different clusters to eliminate the influence of dynamic targets.

These methods are all pure vision SLAM. Although they perform well in processing speed and dynamic scene pose estimation, the positioning accuracy decreases slightly in a complex and highly dynamic environment, and the semantic information of the environment cannot be understood.

With the innovation of deep learning algorithms in the field of computer vision, there are many excellent algorithms in target detection and semantic segmentation technology, such as YOLO (Redmon and Farhadi, 2018), Mask R-CNN (He et al., 2017), etc. Therefore, the SLAM algorithm combined with semantics also shows good performance in a dynamic environment.

Bescos et al. (2018) obtained the semantic information of the scene through the Mask R-CNN instance segmentation network and combined it with the multi-view geometry method to detect and reject the dynamic target. However, the multi-view geometry method is easily affected by the current pose estimation results. In the dynamic scene, the pose estimation accuracy of this method is excellent, but the processing speed is a little slow, which cannot meet the real-time requirements. MaskFusion (Runz et al., 2018) recognises, tracks and reconstructs multiple targets based on ElasticFusion (Whelan et al., 2016). Mask R-CNN is also used to recognise and segment the pixel level of dynamic targets. Combining with geometric segmentation based on depth discontinuity and non-concavity, the interference of dynamic targets is eliminated. Finally, the camera pose is calculated by minimising spatial geometric error and photometric error. This algorithm achieves good results in a dynamic environment. But it needs dual GPUs to run, which consumes a lot of computing resources. Yu et al. (2018) used SegNet semantic segmentation network to obtain the segmentation results of objects in the environment, combined with motion consistency detection method to eliminate the interference of dynamic objects, and built an octree map with semantic tags based on a probability model. This method can meet the real-time requirements, but the accuracy of pose estimation is insufficient.

SLAM algorithm combined with semantic information still has some problems in the above dynamic scenarios, either it can't effectively deal with the impact of dynamic targets, which leads to insufficient positioning accuracy or it can't guarantee real-time. Therefore, this paper uses a dynamic target online segmentation method based on semantic bounding box and depth continuity, combined with relatively static feature detection, which performs well in a dynamic environment and improves real-time. On this basis, an environment map with semantics is established.

3 Framework and process of the SLAM algorithm

The datas flow diagram is shown in Figure 2. It shows the structure of our system. In this paper, four modules have been added to the visual odometer: semantic information acquisition module, dynamic target segmentation module, dynamic feature screening module and effective dynamic feature reuse module.

- In the dynamic target segmentation thread, the semantic information acquisition module obtains the bounding box results of target detection in the environment.
- The dynamic object segmentation module obtains pixel-level segmentation results of prior dynamic objects based on bounding boxes.
- In the tracking thread, after extracting the ORB features of the image, the dynamic feature screening module completes the feature screening task and screens the features into static and dynamic features.

- The effective dynamic feature multiplexing module detects the relative static features and obtains the effective dynamic features which can be used for posture optimisation.

Finally, the back-end mapping thread uses the pose estimation results and the dynamic target segmentation results to build a semantic environment map.

3.1 Semantic information acquisition

At present, most semantic SLAM algorithms use semantic segmentation networks to obtain semantic information, such as Mask R-CNN (He et al., 2017), DeepLabv3 (Chen et al., 2017), SegNet (Badrinarayanan et al., 2017) and so on. Although the semantic segmentation algorithm is excellent in accuracy, it can get accurate classification to pixel level, but it is slow and unable to be real-time. These defects have a great impact on the real application of the SLAM system in robots. And the target detection algorithm can meet the basic application requirements, so we choose to use the target detection algorithm instead of the semantic segmentation algorithm. In this paper, YOLOv3 (Redmon et al., 2018) is chosen, and YOLO takes target detection as a regression problem, which has extremely fast recognition speed and high recognition accuracy, and has low-performance requirements for hardware devices. We train YOLOv3 on COCO data sets (Lin et al., 2014) and it can identify 80 categories.

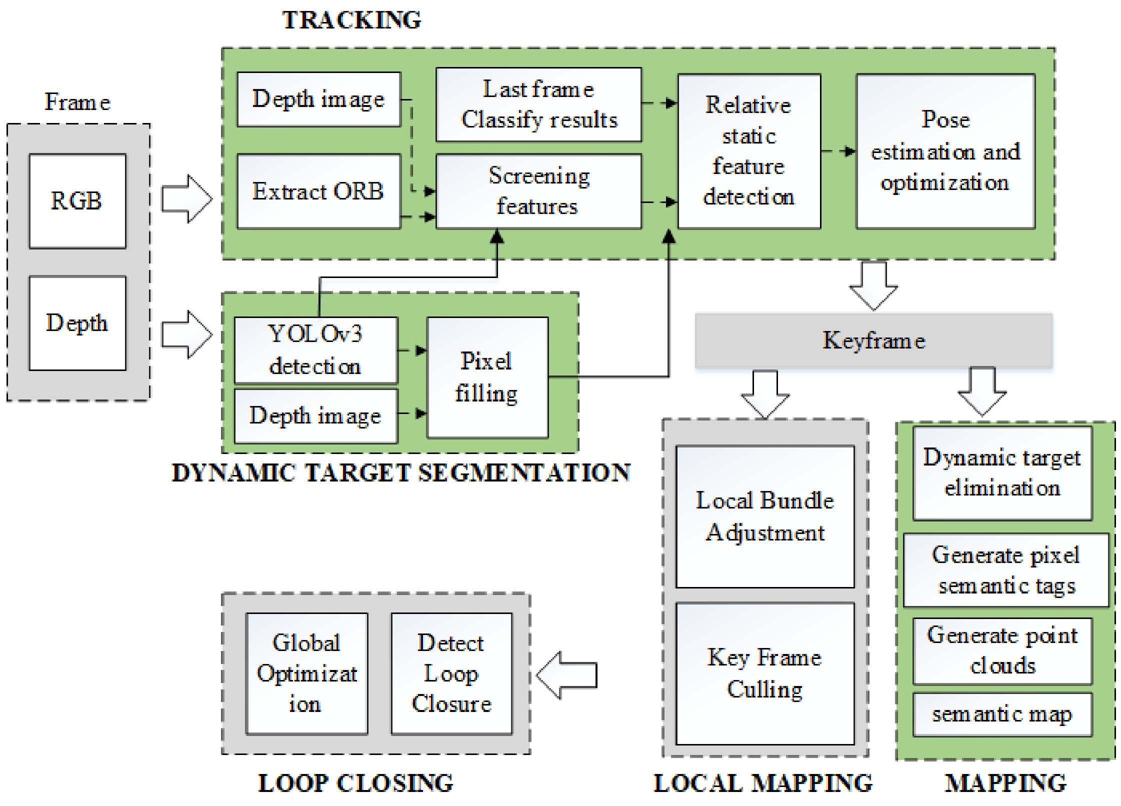
In this paper, these 80 categories are divided into two categories, and the targets with the possibility of autonomous movement are defined as priority dynamic targets, named PDT, such as people, cats, dogs, etc. Targets without the possibility of autonomous motion are defined as priority non-dynamic targets, named PNDT, such as seats, cups, books, etc. The PNDT is divided into two categories. Targets with a high probability of involuntary movement are defined as high probability passive moving targets, named H-PNDT, such as chairs, books, cups, etc. A target with a low probability of involuntary movement is defined as a low probability passive moving target, named L-PNDT, such as a table, monitor, sofa, etc.

The returned results of YOLO contain information such as target category, bounding box, confidence level, etc. The corresponding object bounding box set $\mathbb{B}_{\text{bounding\_box}}$ box of YOLO detection result set of a certain frame is shown in:

$$\mathbb{B}_{\text{bounding\_box}} = \left\{ \left(b^i_{id,u}, b^i_{id,v}, b^i_{id,w}, b^i_{id,h} \right) \middle| 0 \le i \le N \right\} \tag{1}$$

where N represents the total number of targets detected in a certain frame. $b^i_{id,\text{u}}$ and $b^i_{id,\text{v}}$ represent the coordinate information of the upper left corner of the i-th target bounding box. $b^i_{id,\text{w}}$ and $b^i_{id,\text{h}}$, respectively represent the width and height of the i-th target bounding box.

Figure 2 Datas flow of our algorithmic. Our system is based on ORB-SLAM2, on which a dynamic object segmentation thread is added and the tracking thread is modified. A semantic mapping thread is added in the back-end part



3.2 Dynamic target segmentation

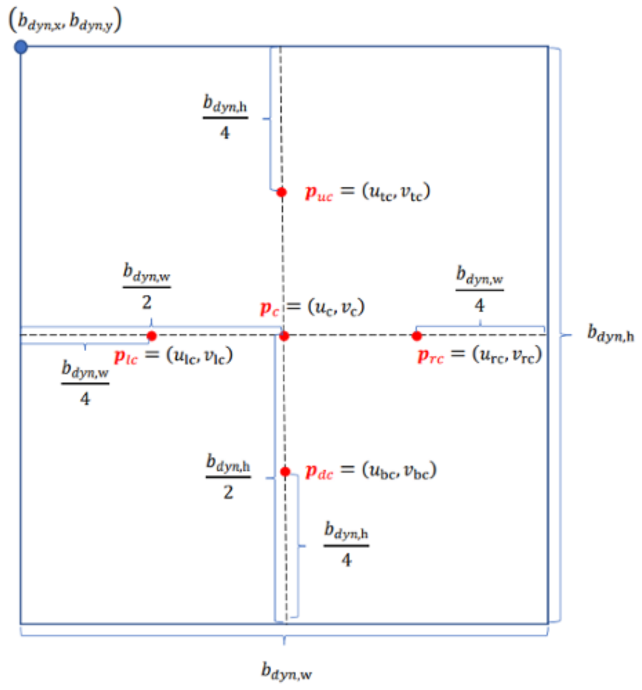
In the real environment, the movement of a PNDT is usually associated with a PDT, so the first concern is the PDT in the field of vision.

YOLOv3 can't get the classification results at the pixel level, so it can't accurately get which pixels belong to the PDT. Because of the wrapping property of bounding boxes, if the features in the bounding boxes of the PDT are simply removed, too much effective information in the field of vision will be lost, which will have a bad influence on SLAM.

To solve this problem, based on bounding boxes, we design a pixel segmentation method to find pixels belonging to the PDT in the current image frame. Because the PDT in the bounding box should occupy the main body, and the pixels belonging to the same target have depth continuity, while the pixel depth at the edge of the contour is discontinuous, a pixel random search filling process is designed by using the difference between the pixel depth values to realise region segmentation. The segmentation process is as follows:

Assuming that the bounding box of a PDT in the current frame is $B_{\text{dynamic,bounding\_box}}$, it is particularly important to find the Seed point of random search in order to achieve good and sufficient pixel segmentation results. Firstly, the centre point $p_c = (u_c, v_c)$ of $B_{\text{dynamic,bounding\_box}}$ box is selected as the first random search initial seed point. Then, based on this centre point, the two central axes of the bounding box are divided into four line segments, and the midpoint of these four line segments $p_{uc} = (u_{tc}, v_{tc})$, $p_{dc} = (u_{bc}, v_{bc})$, $p_{tc} = (u_{tc}, v_{tc})$, $p_{lc} = (u_{lc}, v_{lc})$ are selected as the initial seed point of random search, as shown in Figure 3. In Figure 3, five initial search points are preset in a bounding box. Firstly, p_c is selected for random search filling, and if the filling result is insufficient, other initial search points are selected sequentially.

Figure 3 The initial search point in bounding box



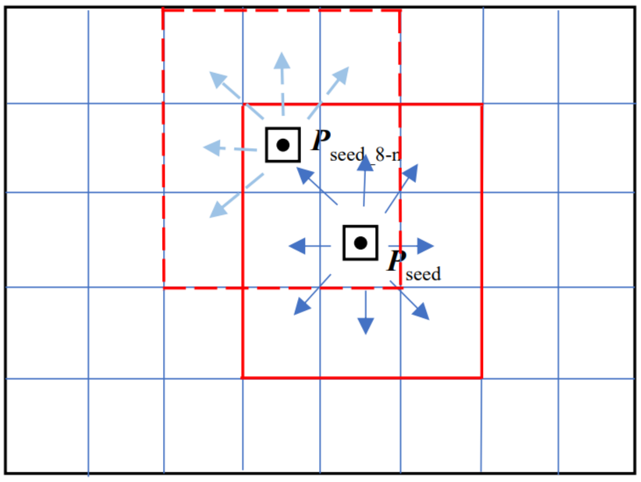
Then, we randomly search and fill pixels in the bounding box according to the depth:

- Step 1, the first seed point above is selected, $p_{\text{seed}} = p_c$.
- Step 2, for each pixel point $p_{\text{seed\_8-n}}$ in the 8 neighbourhood of the seed point p_{seed}, assume that the depth value of $p_{\text{seed\_8-n}}$ is $d_{\text{seed\_8-n}}$. If

$$| d_{\text{seed}} - d_{\text{seed\_8-n}} | < \omega_{\text{pixel}} \tag{2}$$

  where d_{seed} is the depth value of p_{seed}. ω_{pixel} is a gradient limit standard of the pixel random search. Then add the pixel point $\mathbf{p}_{\text{seed\_8-n}}$ to the segmentation result if it located in the bounding box.
- Step 3, taking the newly added pixel point $p_{\text{seed\_8-n}}$ in the segmentation result, let $p_{\text{seed}} = p_{\text{seed\_8-n}}$. Then returning to step 2 and continue to execute. Until no new pixel point is added, step 4 will be executed in sequence. The pixel search process is shown in Figure 4. As can be seen from Figure 4: The random search process is similar to BFS, and the initial point first traverses all points in its 8 neighbourhood. The so-called 8 neighbourhood refers to the adjacent points in the 3*3 area centred on the initial point.
- Step 4, judging the validity of the segmentation result, taking the pixel $\mathbf{p}_{\text{vmin}} = (u_{\text{vmin}}, v_{\text{vmin}})$ with the smallest ordinate and the pixel $p_{\text{vmax}} = (u_{\text{vmax}}, v_{\text{vmax}})$ with the largest ordinate in the segmentation result, and if the difference between v_{vmax} and v_{vmin} is greater than 0.75 times the bounding box height $b_{\text{dyn,h}}$, the segmentation is completed. Otherwise, returning to step 1, select the remaining initial seed points and continue to execute until all five initial seed points are executed.

Figure 4 Random search diagram



After the above steps, depth-based pixel filling is completed. However, because the initial seed point selection strategy tends to overfill, the segmentation result will contain some pixel noise points that do not belong to the dynamic target, so it is necessary to filter the segmentation result.

Firstly, the average depth value d_{ave} of the segmentation area is calculated as:

$$d_{\text{ave}} = \left(\sum_{\mathbf{p}_i} d_i \right) / N_p \tag{3}$$

where d_i represents the depth value of pixel point p_i in the segmentation area, and N_p represents the number of pixel points in the segmentation area.

Then, for each pixel point p_l in the segmentation result, its relative depth difference D_i is calculated as:

$$D_i = \alpha \cdot d_{\text{ave}} - d_i \tag{4}$$

where α is the preset parameter, which indicates the criteria of pixel screening according to depth.

Assuming that the pixel coordinate is $p_i = (u_i, v_i)$, the *X*-axis minimum relative distance g_{x} of this pixel relative to the four sides of the bounding box $B_{\text{dynamic,bounding\_box}}$ is calculated as:

$$g_{\text{x}} = \min\left(u_i - b_{\text{dyn,u}}, b_{\text{dyn,u}} + b_{\text{dyn,w}} - u_i\right) \tag{5}$$

and the *Y*-axis minimum relative distance g_y is calculated as:

$$g_{\text{y}} = \min\left(v_i - b_{\text{dyn,v}}, b_{\text{dyn,v}} + b_{\text{dyn,h}} - v_i\right) \tag{6}$$

The minimum relative distance G_i is calculated from g_x and g_y:

$$G_i = \min\left(g_x, g_y\right) \tag{7}$$

If the weighted sum of the relative depth difference D_i and the minimum relative distance G_i of the pixel p_i is less than zero, p_i is removed from the segmentation result:

$$D_i + \omega \cdot G_i < 0 \tag{8}$$

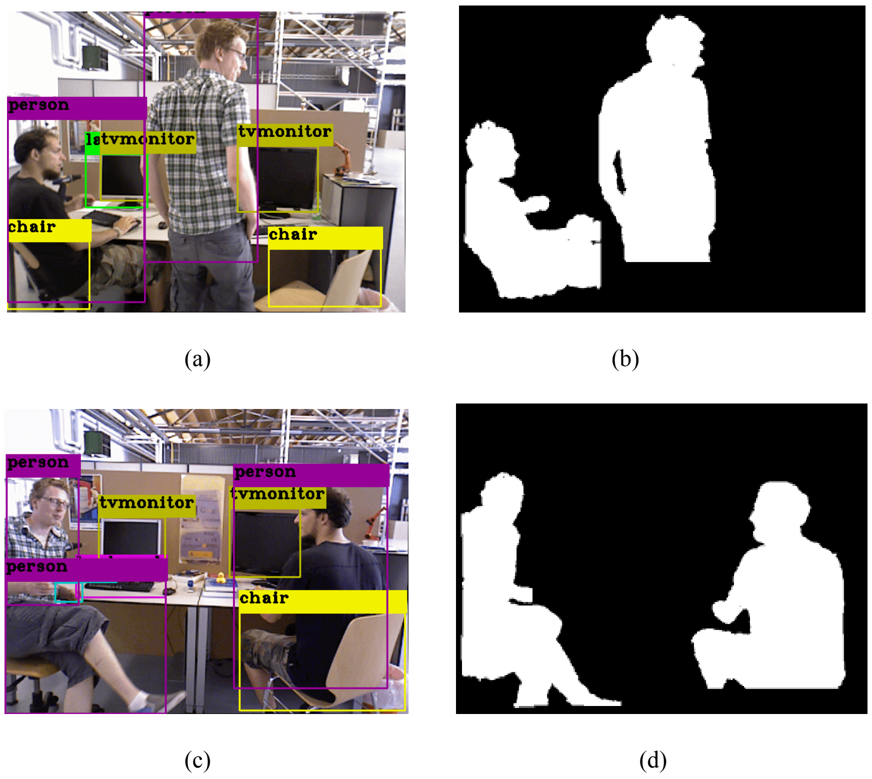
where ω represents the weight of the minimum relative distance. The greater the weight, the greater the influence of the relative position of pixels on the screening results, and the less the influence of the depth of pixels.

Up to now, the segmentation step based on the bounding box jointing depth is finished, and the pixel-level PDT segmentation results are obtained. The pseudo code of the above algorithm is shown in Algorithm 1. The following pseudo-code in Algorithm 1 contains the process of segmentation and result optimisation. For each bounding box of PDT, the Algorithm 1 steps are carried out. Figure 5 shows the segmentation results of the algorithm. YOLO detection results on the left and segmentation results on the right. It can be seen that the segmentation method proposed can get pixel-level segmentation results, and the boundary of the results is relatively accurate.

Algorithm 1: Dynamic target segmentation algorithm

```
Input:  Target detection result set of current frame,
            $\mathbb{O}_{\text{object}}$, $\mathbb{B}_{\text{bounding\_box}}$
Output: Pixel-level dynamic target segmentation result,
            Mask
1:  Mask[w][h] ← {0,0,0…,0}
2:  For $\mathbb{O}^i_{\text{object}} \in \mathbb{O}_{\text{object}}$ do
3:    If $\mathbb{O}^i_{\text{object}} \in \mathbb{O}_{\text{dyn}}$ then
4:      visitStack ← {,,,, $p_c$ }
5:      Mask[Pix.u][Pix.v]=1, Pix in visitStack
6:      While visitStack is not empty do
7:        $p_{seed}$ =Pix, Pix is visitStack.top
8:        visitStack.pop
9:        For ∈ s 8-neighbor do
10:         If Mask[.u][.v] == 0 then
11:           If $| d_{\text{seed}} - d_{\text{seed\_8-n}} | < \omega_{\text{pixel}}$ and
                  $\mathbf{p}_{\text{seed\_8-n}} \in B_{\text{dynamic,bounding\_box}}$ then
12:             Mask[.u][.v]=1
13:             Push  into visitStack
14:           End if
15:         End if
16:       End for
17:     End while
18:     For Pix∈
              and Mask[Pix.u][ Pix.v] == 1 do
19:       If Pix meet formula (8) then
20:         Mask[Pix.u][ Pix.v]=0
21:       End if
22:     End for
23:   End if
24: End for
25: Return Mask
```

Figure 5 Segmentation results



3.3 *Dynamic feature screening*

After extracting ORB features, the features belonging to PDT can be obtained according to the segmentation results of PDT, and the features belonging to PDT are classified as dynamic features in this process.

However, in practical application, the PNDT may be moved, e.g., as shown in Figure 6, a man was dragging a chair. At this time, the chair is also dynamic, and the features belonging to the chair need to be screened as dynamic features. After the above steps, only the features belonging to people can be set as dynamic features, but in fact, the chair is moving with people, but the features belonging to chairs cannot be set as dynamic features at this time. Aiming at similar situations (such as people holding books and cups, etc.), we design a dynamic feature screening method.

For the case that the PNDT follows the PDT, the motion probability p_{pas} of the PNDT is first set. For H-PNDT, the p_{pas} is set to 0.8 and the p_{pas} of L-PNDT is set to 0.25.

Assuming that a bounding box $B_{\mathrm{passive,bounding\_box}}$ of PNDT has an area of A_{pas}, judge whether it intersects with any bounding box of PDT. Assuming that the area of the bounding box of PDT which is intersecting with $B_{\mathrm{passive,bounding\_box}}$ is A_{act}, then we calculate whether the ratio of A_{pas} to the intersection area $A_{\mathrm{pas}} \cap A_{\mathrm{act}}$ is less than the threshold value ω:

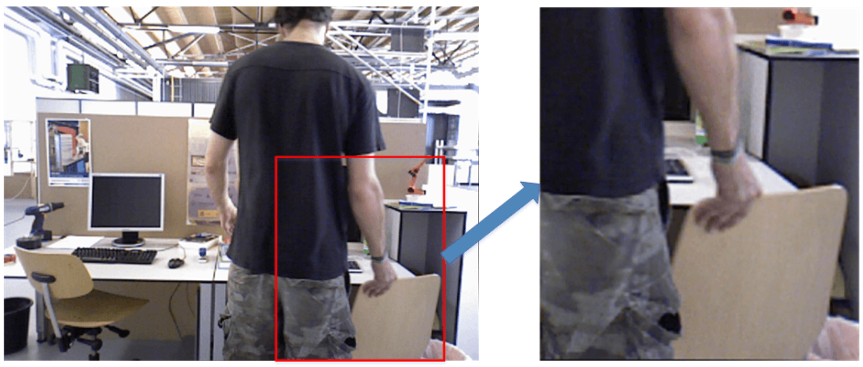
$$\frac{A_{\mathrm{pas}}}{A_{\mathrm{pas}} \cap A_{\mathrm{act}}} \le \omega \tag{9}$$

the threshold ω in (9) is calculated as:

$$\omega = K \cdot p_{\mathrm{pas}} \tag{10}$$

where K is a constant. If $B_{\mathrm{passive,bounding\_box}}$ conforms to (10), search whether there is a pixel set belonging to the PDT in this bounding box (3.2 segmentation result).

Figure 6 Man was dragging a chair



After the search is successful, the pixel set belonging to the PDT is taken as the initial seed point. The pixel random search filling process is carried out in $B_{\text{passive,bounding\_box}}$, which is similar to Sub-section 3.2. The only difference is that the gradient limit standard ω_{pixel} of pixel random search should be appropriately increased at this time, to find the pixel set of object which is in contact with the PDT.

After the pixel random search filling process of the previous step, the number N_{pas} of pixels belonging to this object in $B_{\text{passive,bounding\_box}}$ is obtained, and whether the ratio of N_{pas} and the total number N_{pas} of pixels in $B_{\text{passive,bounding\_box}}$ is less than the threshold value ω is calculated:

$$\frac{N_{\text{box}}}{N_{\text{pas}}} \leq \omega \tag{11}$$

where ω represents the threshold, and the calculation equation of it is the same as (10). Since we will select effective dynamic features which can be used for pose optimisation from the dynamic features, if $B_{\text{passive,bounding\_box}}$ conforms to (11), the features in $B_{\text{passive,bounding\_box}}$ are set as dynamic features. The pseudo code of the above algorithm is shown in Algorithm 2. It shows the specific operation steps in detail, and finally obtains the static feature set and the dynamic feature set.

Algorithm 2: Dynamic feature point screening algorithm

```
Input:  Target detection result set of current frame,
        $\mathbb{O}_{\text{object}}$, $\mathbb{B}_{\text{bounding\_box}}$ 、dynamic target segmentation
        result, Mask and the set of ORB features,
        ORBs;
Output: the set of static features, STAs and the set of
        dynamic features, DYNs;
1:  For ORB $\in$ ORBs do
2:    If Mask[ORB.u][ORB.v]==1 then
3:      DYNs add(ORB)
4:    End if
5:  End for
6:  For $\mathbb{O}^i_{\text{object}} \in \mathbb{O}_{\text{object}}$ do
7:    If $\mathbb{O}^i_{\text{object}} \in \mathbb{O}_{\text{passive}}$ then
8:    If $B_{\text{passive,bounding\_box}}$ meet formula (9) then
9:        $N_{\text{box}} \leftarrow 0$, $N_{\text{pas}} \leftarrow 0$
10:       For Pix $\in$ do
11:         $N_{\text{box}}$ ++
12:       If Mask[Pix.u][Pix.v]==1 then
13:           Push Pix into visitStack
14:         End if
15:       End for
16:       While visitStack is not empty do
17:       Pixel random searching filling and $N_{\text{pas}}$ ++
18:     End while
19:     If $N_{\text{box}} / N_{\text{pas}} \leq \omega$ then
20:         For ORB $\in$ ORBs $\cap$ $B_{\text{passive,bounding\_box}}$ do
21:           DYNs add(ORB)
22:         End for
23:       End if
24:     End if
25:   End if
26: End for
27: For ORB $\in$ ORBs do
28:   If ORB not in DYNs then
29:     STAs add(ORB)
30:   End if
31: End for
```

Figure 7 shows the results of dynamic feature screening. Figure 7(a) is the YOLO detection result of this frame, Figure 7(b) is the PDT segmentation result of this frame, Figure 7(c) shows the result of chair pixel filling and Figure 7(d) shows the final classification effect of features, with red as dynamic features and green as static features.

3.4 *Effective dynamic feature multiplexing*

After filtering the dynamic features, the features will be divided into static and dynamic features. At this time, if the dynamic features are simply eliminated, many available information may be ignored in some scenes, resulting in a decrease in the number of features that can be used for pose optimisation, and a decrease in pose estimation accuracy, and even a failure in positioning. For example, as shown in Figure 8, the person who is in a sitting posture may only slightly move his limbs and heads, so his body still contains many effective features that can be used for pose estimation. Like this kind of situation, the dynamic features may still contain a lot of valid information, and if all of them are removed, this part of the information cannot be used.

Because of this situation, this paper proposes a relatively static feature detection method based on feature data association, to judge whether there are effective dynamic features that can be used to optimise pose estimation.

Feature Data association means that due to the camera self-motion, the 2D coordinate $p_{\text{s\_cur}} = \left(u_{\text{s\_cur}}, v_{\text{s\_cur}}\right)$ of a static feature p_s in the current frame will change with respect to its 2D coordinate $p_{\text{s\_last}} = \left(u_{\text{s\_last}}, v_{\text{s\_last}}\right)$ in the previous frame. Because the similarity between two consecutive images is very high, and usually the coordinate change distance between two adjacent frames is very small relative to the depth of the scene, it can be considered that this motion change should be the same for each static feature, i.e. each static feature is relatively static. As shown in Figure 9, by placing the 2D coordinates of the previous frame and the current frame in the same coordinate system, we can intuitively see that p_i and another static feature p_{s2} move in the same way between two adjacent frames in most cases.

Figure 7 Classification results of features

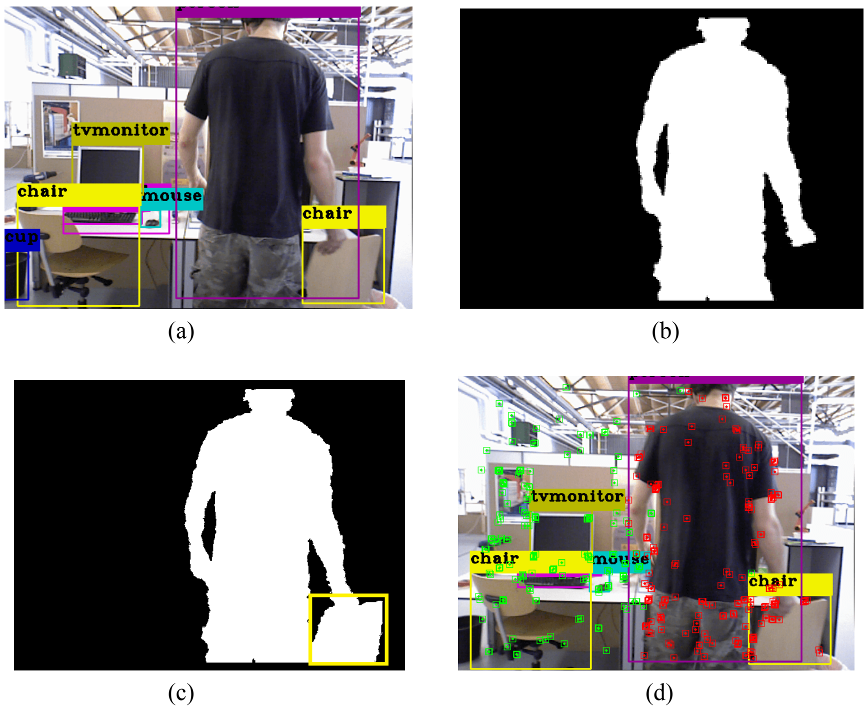


Figure 8 Dynamic feature screening results

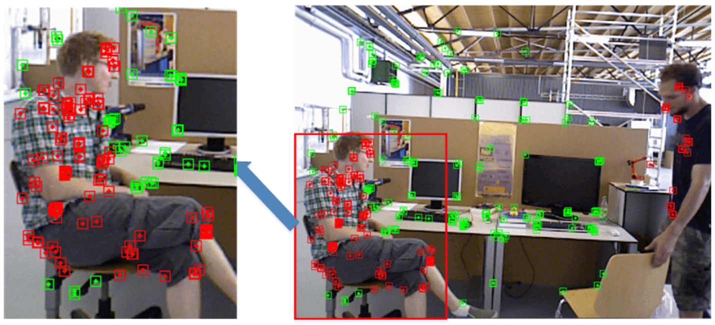
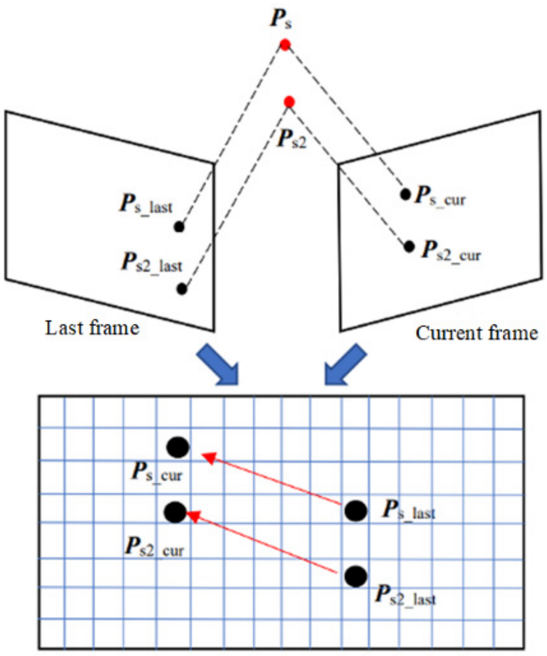


Figure 9 Transformation correlation diagram



Therefore, for each dynamic feature p_d, it is detected whether it remains relatively static with the static feature. If it remains relatively static, p_d is set as an effective dynamic feature.

Firstly, we choose to use a homography matrix to represent this transformation, and calculate the transformation homography matrix H through Random Sample Consensus (RANSAC) by using the static matching feature pairs of the previous frame and the current frame.

Assuming that the coordinate of a certain dynamic feature p_d in the previous frame is $p_{\text{d\_last}} = \left(u_{\text{d\_last}}, v_{\text{d\_last}}\right)$, we project the coordinate of the previous frame of p_d to the current frame to obtain the projection coordinate $p_{\text{d\_proj}}$:

$$p_{\text{d\_proj}} \leftarrow \begin{pmatrix} u_{\text{d\_proj}} \\ v_{\text{d\_proj}} \\ 1 \end{pmatrix} = \begin{pmatrix} u_{\text{d\_last}} \\ v_{\text{d\_last}} \\ 1 \end{pmatrix} \cdot H \tag{12}$$

Then calculate whether the Euclidean distance D_{EUC} between the projection coordinate $p_{\text{d\_proj}} = \left(u_{\text{d\_proj}}, v_{\text{d\_proj}}\right)$ and the real coordinate $p_{\text{d\_cur}} = \left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right)$ of the current frame is less than the threshold value ω:

$$\sqrt{\left(u_{\text{d\_cur}} - u_{\text{d\_proj}}\right)^2 + \left(v_{\text{d\_cur}} - v_{\text{d\_proj}}\right)^2} < \omega \tag{13}$$

where the threshold value ω represents the displacement limit standard.

If p_d satisfies (13), its depth value and grey value need to be verified. Firstly, the grey average value $G_{\text{ave-8-n}}$ and depth average value $D_{\text{ave-8-n}}$ of all pixels in the 8 neighbourhood of $p_{\text{d\_cur}} = \left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right)$, which is the real coordinate of p_d in the current frame, are calculated respectively. And the 8 neighbourhood composite grey value $G_{\text{8-n}}\left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right)$ and composite depth value $D_{\text{8-n}}\left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right)$ of the point $p_{\text{d\_cur}}$ are calculated as:

$$G_{\text{8-n}}\left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right) = G_{\text{d\_cur}} + G_{\text{ave-8-n}}/2 \tag{14}$$

$$D_{\text{8-n}}\left(u_{\text{d\_cur}}, v_{\text{d\_cur}}\right) = D_{\text{d\_cur}} + D_{\text{ave-8-n}}/2 \tag{15}$$

where $G_{\text{d\_cur}}$ is the grey value of $p_{\text{d\_cur}}$,and $D_{\text{d\_cur}}$ is the depth value of $p_{\text{d\_cur}}$.

Then 8 neighbourhood composite grey value $G_{\text{8-n}}\left(u_{\text{d\_proj}}, v_{\text{d\_proj}}\right)$ and composite depth value $D_{\text{8-n}}\left(u_{\text{d\_proj}}, v_{\text{d\_proj}}\right)$ of projection coordinate point $p_{\text{d\_proj}} = \left(u_{\text{d\_proj}}, v_{\text{d\_proj}}\right)$ of p_d in the current frame are calculated in the same way.

At this point, we can calculate whether the composite grey difference between the real coordinates $p_{\text{d\_cur}}$ and the projected coordinates $p_{\text{d\_proj}}$ is less than the threshold ω_g:

$$\left|G_{\text{8-n}}\left(x_{\text{d\_proj}}, y_{\text{d\_proj}}\right) - G_{\text{8-n}}\left(x_{\text{d\_cur}}, y_{\text{d\_cur}}\right)\right| < \omega_{\text{g}} \tag{16}$$

and whether the composite depth difference is less than the threshold ω_d:

$$\left|D_{\text{8-n}}\left(x_{\text{d\_proj}}, y_{\text{d\_proj}}\right) - D_{\text{8-n}}\left(x_{\text{d\_cur}}, y_{\text{d\_cur}}\right)\right| < \omega_{\text{d}} \tag{17}$$

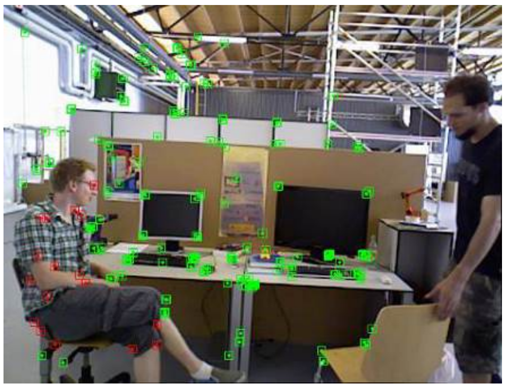
If p_d satisfies (16) and (17), the dynamic feature p_d is set as an effective dynamic feature for pose optimisation. The pseudo code of the above algorithm is shown in Algorithm 3. For each dynamic feature, Algorithm 3 steps are performed to determine whether it is an effective feature.

Algorithm 3: Relative static feature detection

```
Input:  the set of static features, STAs and the set of
          dynamic features, DYNs of currFrame and
          lastFrame
Output: the set of effective features, EFs;
1: Calculate Match_sta by STAs_cur and STAs_last
2: Calculate Match_dyn by DYNs_cur and DYNs_last
3: Calculate H by Match_sta
4: For $p_d$ ∈ Match_dyn do
5:    $p_{\text{d\_proj}}$ ← formula (12)
6:    If $p_{\text{d\_proj}}$ meet formula (13) (16) and (17) then
7:       EFs add ($p_d$)
8:    End if
9: End for
```

Figure 10 shows the selection of effective dynamic features, in which green represents static features and red represents selected effective dynamic features. The left person is in a sitting position with little movement amplitude, so there are many effective dynamic features on the body after detection and selection, which can be used for posture optimisation to improve positioning accuracy. However, the person on the right is walking and has a large range of motion, so there are no valid dynamic features that meet the requirements.

Figure 10 Effective feature results. The red dot is the effective dynamic feature obtained through detection



In the end, the calculated effective dynamic features and static features are combined for nonlinear optimisation of pose estimation, and the accuracy of pose estimation is significantly improved, and the robustness of the system in a dynamic environment is also improved. Based on the above work, a dense map with PDTs removed and semantic tags is built for the environment.

4 Experiment and analysis

In this part, the algorithm is tested from four aspects: dynamic target segmentation, location accuracy test, time performance, and map construction, and compared with ORB-SLAM algorithm in ORB-SLAM3, ORB-SLAM2, and other excellent SLAM algorithms in dynamic scenes.

The experiment is mainly carried out in the 'walking' and 'sitting' sequences in the TUM data sets (Sturm et al., 2012). The experimental environment includes Intel i7 9700K CPU, GeForce RTX 2070 GPU and 16 GB memory. In addition, we also test the algorithm in a real environment and shows the effect.

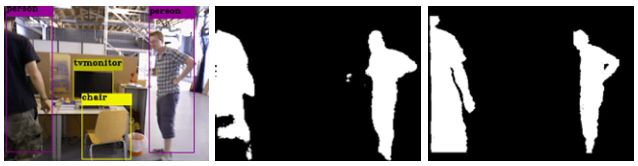
4.1 Dynamic target segmentation

After YOLOv3 detection results are obtained, whether the segmentation results of PDTs are accurate or not will directly affect the subsequent process of this algorithm, so the accuracy of the segmentation algorithm needs to be tested. Compared with the semantic segmentation network based on deep learning, our proposed segmentation method has a fast segmentation speed. Therefore, considering the real-time, we compare the segmentation results with SegNet, a semantic segmentation network with extremely fast segmentation speed. In the subsequent experiment part (Subsection 4.3), the time efficiency test of this method and ours is included. Experiments are carried out on the proposed segmentation algorithm under four sequences high dynamic scenes *fr3_walking_static*, *fr3_walking_xyz*, *fr3_walking_rpy*, and *fr3_walking_ halfsphere* in the TUM data sets.

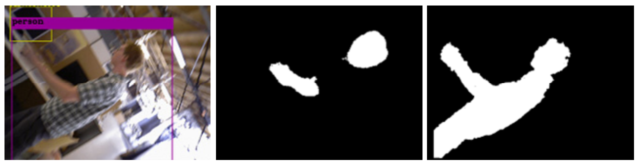
Segmentation results are shown in Figure 11. The first column is YOLO target detection result, the second column is SegNet segmentation result and the third column is segmentation result of our algorithm. We can see that YOLOv3 can only obtain target bounding boxes, but the PDT bounding boxes still contain much static information. If this static information cannot be used, the accuracy of pose estimation will decrease. The SegNet's segmentation effect is not ideal, and there are many defects in the segmentation results, which leads to incomplete elimination of dynamic feature points, thus affecting the accuracy of pose estimation. However, in most cases, our algorithm can obtain more accurate and complete segmentation results at the pixel level of PDTs, and separate static information from dynamic information.

Figure 11(a) shows the segmentation situation when a person is located at the edge of the visual field and the whole body is not blocked (standing posture), and it can be seen that our segmentation result is excellent. In this case, SegNet's segmentation results for people at the edge of the field of vision are slightly poor, resulting in missing legs. Figures 11(b) and 11(c) show the situation when the field of view is tilted and rotated, which does not affect our segmentation result. It can be seen that SegNet segmentation results are very unstable at this time, and most of them are missing. Figure 11(d) shows that the General situation, and our method can also get the approximate real value. In this case, SegNet segmentation results are also very unsatisfactory. To sum up, the segmentation algorithm proposed in this paper can obtain accurate pixel-level segmentation results of PDTs in a dynamic environment without using a semantic segmentation network.

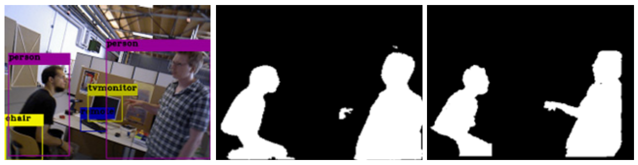
Figure 11 Segmentation experimental results



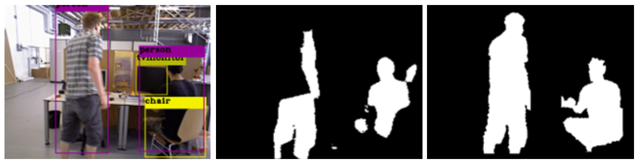
(a)



(b)



(c)



(d)

4.2 Pose estimation accuracy

All pose estimation experiments are carried out in the dynamic sequence of TUM data set, which is the authoritative experimental data set in SLAM field at present. The evaluation indexes mainly include Relative Pose Error (RPE) and Absolute Trajectory Error (ATE). In order to avoid contingency, 10 identical experiments were performed on each sequence to take the average value.

First of all, in order to prove the effectiveness and advantages of our proposed effective dynamic features detection algorithm, we conducted ablation experiments based on our system under the condition that other conditions remain unchanged. Table 1 shows the experimental results, in which ours represents the pose estimation results of our system, and OursNE represents the pose estimation results after we disable the effective dynamic features detection. As can be seen from Table 1, compared with the results after disabling the effective dynamic feature detection method, the pose accuracy is improved after enabling this method, which can prove that the dynamic feature detection method we proposed is effective. The main reason is that more temporary features can be used for optimisation when using the uniform velocity model to estimate pose, so the accuracy will be improved accordingly.

Next, we will compare the overall accuracy of our system with other systems. Table 2 shows the comparison results of RPE and ATE between the latest ORB-SLAM3 and our algorithm. It contains datas such as Root Mean Square Error (RMSE) and Standard Deviation (S.D.). Since ORB-SLAM3 is not better than ORB-SLAM2 in some cases, our algorithm is also compared with ORB-SLAM2. Table 3 shows RPE and ATE of ORB-SLAM2 and our algorithm.

Table 1 Ablation experiments

Seq	*ATE (m/s)*	
	OursNE	*Ours*
fr3_walking_static	0.0098	**0.0072**
fr3_walking_xyz	0.0188	**0.0147**
fr3_walking_rpy	0.0457	**0.0331**
fr3_walking_half	0.0329	**0.0253**

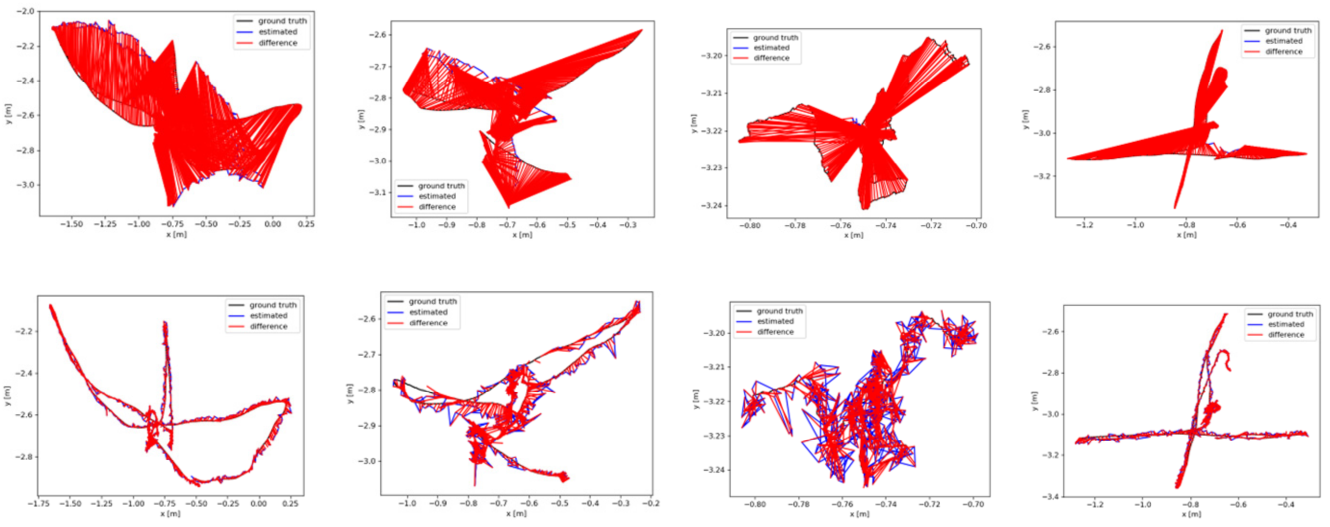
Table 2 Comparisons of ATE and RPE of ORB-SLAM3 and ours

Seq	*RPE: Translation drift (RMSE m/s)*		*RPE: Rotating drift (RMSE deg/s)*		*ATE (RMSE m/s)*	
	ORB-SLAM3	*Ours*	*ORB-SLAM3*	*Ours*	*ORB-SLAM3*	*Ours*
fr3_walking_static	0.0364	**0.0103**	53.789	**0.2988**	0.0255	**0.0072**
fr3_walking_xyz	0.4115	**0.0207**	22.294	**0.5876**	0.2838	**0.0147**
fr3_walking_rpy	0.3617	**0.0448**	39.547	**0.9605**	0.1969	**0.0332**
fr3_walking_half	1.0571	**0.0347**	21.63	**0.8903**	0.5958	**0.0253**
fr3_sitting_xyz	0.4899	**0.0188**	16.811	**0.6029**	0.2357	**0.0131**
fr3_sitting_half	0.4415	**0.0221**	37.842	**0.6388**	0.2852	**0.0151**

Table 3 Comparisons of ATE and RPE of ORB-SLAM2 and ours

Seq	*RPE: Translation drift (RMSE m/s)*		*RPE: Rotating drift (RMSE deg/s)*		*ATE (RMSE m/s)*	
	ORB-SLAM2	*Ours*	*ORB-SLAM2*	*Ours*	*ORB-SLAM2*	*Ours*
fr3_walking_static	0.2015	**0.0103**	3.6028	**0.2988**	0.3575	**0.0072**
fr3_walking_xyz	0.3711	**0.0207**	6.9625	**0.5876**	0.6772	**0.0147**
fr3_walking_rpy	0.4658	**0.0448**	9.1274	**0.9605**	0.8806	**0.0332**
fr3_walking_half	0.3418	**0.0347**	6.7633	**0.8903**	0.5186	**0.0253**
fr3_sitting_xyz	**0.0126**	0.0188	**0.5731**	0.6029	**0.0090**	0.0131
fr3_sitting_half	0.0315	**0.0221**	0.6496	**0.6388**	0.0452	**0.0151**

Figure 12 Trajectory comparison in dynamic environment. The first line is the estimated trajectory diagram of ORB-SLAM3, and the second line is ours. It can be seen that there is little difference between the estimated trajectory of our system and the ground truth



(a) fr3_walking_halfsphere (b) fr3_walking_rpy (c) fr3_walking_static (d) fr3_walking_xyz

Figure 12 shows the comparison result between the estimated and real values of the camera motion trajectory on the walking sequences of ORB-SLAM3 (the first line) and our algorithm (the second line). It can be clearly seen from the trajectory comparison diagram that there is a big gap between the estimated and real trajectory of ORB-SLAM3 in the dynamic scene, and there will be tracking failure. However, the difference between the estimated pose and the real pose of our algorithm is very small, which can track the camera trajectory well.

From the above experimental results, it can be concluded that the SLAM algorithm based on bounding box and depth continuity proposed in this paper performs well in dynamic environment, and significantly reduces the translation drift and rotation drift of visual odometer in dynamic environment, so that the absolute trajectory error of the whole SLAM algorithm is minimised.

The reasons are as follows: First, the dynamic target segmentation algorithm and dynamic feature screening algorithm proposed can effectively deal with dynamic targets in the environment, eliminate the adverse effects of dynamic features on pose estimation, and make our algorithm have better adaptability to dynamic environment. Secondly, the relatively static feature detection algorithm designed in this paper can select the effective dynamic features which can be used for pose optimisation, and make full use of the effective information in the field of vision to improve the pose estimation accuracy.

In order to further verify the advancement and robustness of the algorithm proposed in this paper, Table 4 lists the experimental results of our algorithm compared with other excellent SLAM algorithms in dynamic scenes. It can be seen from Table 4 that our algorithm performs best in most sequences compared with other SLAM algorithms in dynamic scenes. Only in *fr3_walking_static* sequence, the positioning accuracy is slightly inferior to that of Bescos et al. (2018), but it can be seen that the gap is very small. The accuracy of *fr3_sitting_xyz* sequence is lower than that of Dai et al. (2020), and the above experimental comparison results show that the performance of our algorithm is better than that of other SLAM systems in dynamic scenes in most cases. The analysis reasons are as follows: First, because our dynamic target segmentation algorithm performs well, the dynamic feature screening results are more accurate. Second, we didn't simply eliminate the dynamic features but detected the effective features to optimise the pose estimation results.

4.3 Time performance test

In practical application, real-time performance is an important index to evaluate SLAM system. Therefore, the running time of some main modules is tested and compared with other excellent SLAM algorithms in dynamic environment under the same hardware conditions. The results are shown in Table 5. The segmentation module, motion detection module and the average time of processing one frame are compared respectively. The processing time of the segmentation module of our algorithm includes the running time of YOLOv3.

The average running time of the proposed dynamic object segmentation algorithm is 11.15 ms, and the average processing time of relative static feature detection is only 2.9 ms. Because the system works in multithreading mode, the average processing time of each frame by the main thread is 18.2 ms. Compared with Yu et al. (2018) and other non-real-time dynamic scene SLAM, the proposed algorithm can better meet the real-time requirements of practical applications.

Table 4 Comparisons of the RMSE of absolute trajectory error (ATE) of other SLAM algorithms in dynamic environment and ours

Seq	*Yu et al.*	*Bescos et al.*	*Sun et al.*	*Yang et al.*	*Dai et al.*	*Wang et al.*	*Ours*
fr3_walking_static	0.0081	**0.0069**	0.0656	0.0074	0.0108	0.0078	0.0072
fr3_walking_xyz	0.0247	0.0152	0.0932	0.0874	0.0874	0.0162	**0.0147**
fr3_walking_rpy	0.4442	0.0351	0.1333	0.0351	0.1608	0.0373	**0.0332**
fr3_walking_half	0.0303	0.0261	0.1252	0.0268	0.0354	0.0277	**0.0253**
fr3_sitting_half	–	0.0172	0.047	–	0.0235	0.0159	**0.0151**
fr3_sitting_xyz	–	0.0154	0.0482	–	**0.0091**	–	0.0131

Table 5 Time evaluation (Unit: ms)

Module	*Yu et al. (2018)*	*Bescos et al. (2018)*	*Fang et al. (2021)*	*Ours*
Segmentation	SegNet	Mask R-CNN	Mask R-CNN	Dynamic target segmentation
	24.5	236.1	236.1	**11.1**
Motion detection	Motion consistency	Multi-view geometry	Semantic descriptor	Relative static feature
	18.1	1444	16.2	**2.9**
total	36.5	1714	261.1	**18.2**

4.4 Map construction experiment

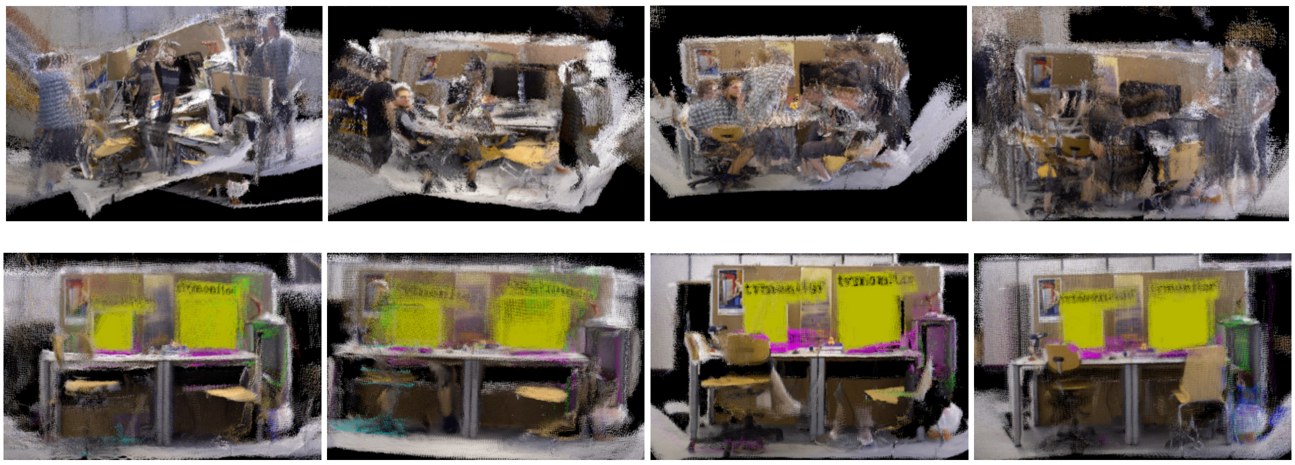
On the basis of the previous work, we build a dense map with semantic tags and no PDTs.

Firstly, according to the dynamic object segmentation algorithm proposed in this paper, the pixels belonging to PDTs in RGB images of key frames are eliminated. Then, according to the 2D detection results, for each PNDT detected in the keyframe, semantic classification and colouring are carried out on the pixels of the RGB image of the keyframe based on its bounding box. Finally, according to the keyframe RGB image, the depth image and the optimised pose, a dense point cloud is generated, and a dense map with semantic tags is constructed in the dynamic scene.

TUM: Figure 13 shows the comparison between ORB-SLAM3 (first row) and the algorithm in this paper (second row) in the TUM dynamic data sets. Among them, we add a point cloud mapping thread for ORB-SLAM3 and use keyframes and their pose estimation results to build maps.

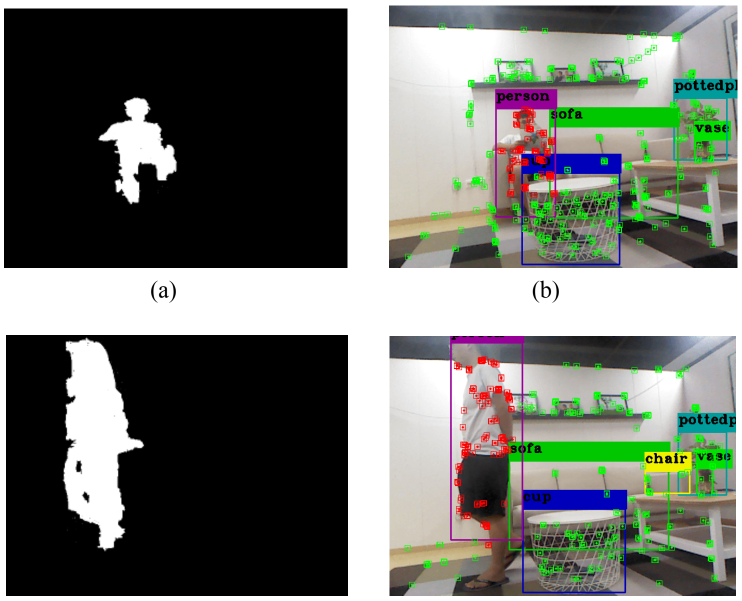
It can be seen that the readability of the first-row map is poor. First, due to the low accuracy of pose estimation, the map coincidence degree is low. Second, because the PDTs are not eliminated, the map contains a lot of dynamic information. However, the map constructed by our algorithm eliminates the PDTs, and the pose estimation accuracy is excellent, which makes the map achieve good results. On this basis, semantic information is added to the map. As shown in the second line of Figure 13, yellow is the screen and purple is the keyboard, but it can also be seen that the semantic tags of some targets are noisy. For people in the field of vision, the system also eliminates basically completely. For some noise points that still exist, the reasons are as follows: in a few keyframes, YOLO did not detect people, which led to the fact that people were not removed from these keyframes and noise points were generated in the map.

Figure 13 Map Comparison in Dynamic Scene. The first line is the mapping result of orb, and the second line is ours. We can see that our system basically restored the real scene and eliminate PDTs



(a) fr3_walking_halfsphere (b) fr3_walking_rpy (c) fr3_walking_static (d) fr3_walking_xyz

Figure 14 Real environment results display



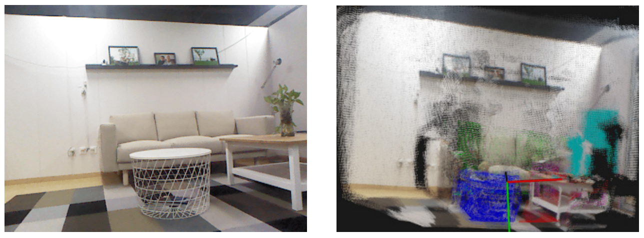
(a) (b)

(c) (d)

Reality: Experiments are also carried out in a real environment, and the images are captured by Kinect V1 camera. Figure 14 shows the running results of our algorithm in the real environment. The left side is the segmentation result, and the right side is the classification result of features. The features on the moving person are divided into dynamic features (red), and the rest of the static parts are divided into static features (green).

Figure 15 shows a 3D dense point cloud map with semantics after removing PDTs. The left side is the real environment, and the right side is the mapping result. The green point cloud represents sofa, blue point cloud represents stool, purple point cloud represents dining table, and blue-green point cloud on the left side of the map represents potted plants. It can be seen that the mapping effect is good and the real scene is basically restored. The experimental results in the real environment show that our system can work well in the real dynamic environment and build an environment map with semantic tags and only static parts.

Figure 15 Mapping in real environment


(a) (b)

5 Conclusion

In this paper, a real-time robust visual SLAM algorithm in the dynamic scene is proposed, which can run in real-time and ensure high-precision positioning, which is the premise that the SLAM algorithm can be applied in the real environment. For dynamic target detection and segmentation, a dynamic target segmentation algorithm based on bounding box and depth continuity is proposed, which can quickly and accurately segment the contour of the dynamic target. The influence of dynamic targets on pose estimation is eliminated to the greatest extent. At the same time, a relatively static feature detection algorithm based on feature Data association is proposed, and effective dynamic features can be selected for pose optimisation, which not only significantly improves the accuracy of pose estimation, but also avoids inaccurate pose estimation or even tracking failure caused by insufficient features in the field of vision, and enhances the robustness of the system. Experimental results show that the algorithm proposed in this paper is effective and advanced. In the future, it will further improve the dynamic perception ability of the system, thus reducing the dependence on the target detection algorithm. At the same time, we will focus on the research and improvement of the mapping module.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (61872327), Anhui University Collaborative Innovation Project (GXXT-2019-003), and Open fund of key laboratory of civil aviation technology and flight safety (FZ2020KF02).

References

Badrinarayanan, V., Kendall, A. and Cipolla, R. (2017) 'Segnet: a deep cpmonvolutional encoder-decoder architecture for image segmentation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 12, pp.2481–2495.

Bescos, B., Fcil, J., Civera, J. and Neira, J. (2018) 'DynaSLAM: tracking, mapping, and inpainting in dynamic scenes', *IEEE Robotics and Automation Letters*, Vol. 3, No. 4, pp.4079–4083.

Chen, L., Papandreou, G., Schroff, F. and Adam, H. (2017) 'Rethinking Atrous convolution for semantic image segmentation', arXiv:1706.05587v3 [cs.CV].

Dai, W., Zhang, Y., Li, P., Fang, Z. and Scherer, S. (2020) 'RGB-D SLAM in dynamic environments using point correlations', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 99, pp.1–1. Doi. 10.1109/TPAMI.2020.3010942.

Du, Z., Wen, Y., Xiao, C., Zhang, F., Huang, L. and Zhou, C. (2018) 'SMotion planning for unmanned surface vehicle based on trajectory unit, *Ocean Engineering*, Vol. 151, No. 1, pp.45–56.

Engel, J., Koltun, V. and Cremers, D. (2018) 'Direct sparse odometry', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 40, No. 3, pp.611–625.

Engel, J., Schps, T. and Cremers, D. (2014) 'LSD-SLAM: large-scale direct monocular SLAM', *European Conference on Computer Vision*, Springer, Cham, pp.834–849

Fang, B., Ding, J. and Wang, Z. (2019) 'Autonomous robotic exploration based on frontier point optimization and multistep path planning, *IEEE Access*, Vol. 7, No. 1, pp.46104–46113.

Fang, B., Mei, G., Yuan X., Wang L., Wang, Z. and Wang J. (2021) 'Visual SLAM for robot navigation in healthcare facility, *Pattern Recognition*, Vol. 113, No. 31. Doi: 10.1016/j.patcog.2021.107822.

Fuentes-Pacheco, J., Ruiz-Ascencio, J. and Rendn-Mancha, J.M. (2015) 'Visual simultaneous localization and mapping: a survey', *Artificial Intelligence Review*, Vol. 43, No. 1, pp.55–81.

He, K., Gkioxari, G., Dollr, P. and Girshick, R. (2017) 'Mask R-CNN', *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp.2961–2988. Doi. 10.1109/ICCV.2017.322.

Kim, D. and Kim, J. (2016) 'Effective background model-based rgb-d dense visual odometry in a dynamic environment', *IEEE Transactions on Robotics*, Vol. 32, No. 6, pp.1565–1573.

Lin, T., Maire, M., Belongie, S., Hays, J. and Zitnick, C. (2014) 'Microsoft COCO: common objects in context', *European Conference on Computer Vision*, Springer International Publishing, pp.740–755.

Mur-Artal, R. and Tards, J.D. (2017) 'ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras', *IEEE Transactions on Robotics*, Vol. 33, No. 5, pp.1255–1262.

Redmon, J. and Farhadi, A. (2018) *YOLOv3: An Incremental Improvement*, Technical Report.

Runz, M., Buffier, M., and Agapito, L. (2018) 'MaskFusion: real-time recognition, tracking and reconstruction of multiple moving objects', *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp.10–20.

Sturm, J., Engelhard, N., Endres, F., Burgard, W. and Cremers, D. (2012) 'A benchmark for the evaluation of RGB-D SLAM systems', *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Portugal.

Sun, Y., Ming, L. and Meng, Q. (2016) 'Improving RGB-D SLAM in dynamic environments: a motion removal approach', *Robotics and Autonomous Systems*, Vol. 89, No. 1, pp.110–122.

Wang, H., Wang, L. and Fang, B. (2021) 'Robust visual odometry using semantic information in complex dynamic scenes', *Cognitive Systems and Signal Processing*, Springer, China.

Wang, Y. and Huang, S. (2014) 'Towards dense moving object segmentation based robust dense RGB-D SLAM in dynamic scenarios', *International Conference on Control Automation Robotics and Vision*, Sydney, NSW, Australia.

Whelan, T., Salas-Moreno, R., Glocker, B., Davison, A., and Leutenegger, S. (2016) 'ElasticFusion: real-time dense SLAM and light source estimation', *International Journal of Robotics Research*, Vol. 35, No. 14, pp.1697–1716.

Yang, X., Yuan, Z., Zhu, D., Chi, C., Li, K. and Liao, C. (2020) 'Robust and Efficient RGB-D SLAM in dynamic environments', *IEEE Transactions on Multimedia*, Vol. 23, pp.4208–4219. Doi. 10.1109/TMM.2020.3038323.

Yu, C., Liu, Z., Liu, X., Xie, F., Yang, Y., Wei, Q. and Fei, Q. (2018) 'DS-SLAM: a semantic visual SLAM towards dynamic environments, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp.1168–1174.