

## RESEARCH ARTICLE

# Real-Time SLAM Based on Dynamic Feature Point Elimination in Dynamic Environment

RUIZHEN GAO<sup>1,2,3</sup>, ZIHENG LI<sup>1</sup>, JUNFU LI<sup>1</sup>, BAIHUA LI<sup>5</sup>, JINGJUN ZHANG<sup>1,2,3</sup>, AND JUN LIU<sup>6</sup><sup>1</sup>School of Mechanical and Equipment Engineering, Hebei University of Engineering, Handan, Hebei 056038, China<sup>2</sup>Key Laboratory of Intelligent Industrial Equipment Technology of Hebei Province, Hebei University of Engineering, Handan, Hebei 056038, China<sup>3</sup>Collaborative Innovation Center for Modern Equipment Manufacturing of Jinan New Area (Hebei), Hebei University of Engineering, Handan, Hebei 056038, China<sup>4</sup>School of Earth Science and Engineering, Hebei University of Engineering, Handan, Hebei 056038, China<sup>5</sup>Department of Computer Science, Loughborough University, Loughborough, LE11 3TU Leicestershire, U.K.<sup>6</sup>Handan Textile Machinery Company Ltd., Handan, Hebei 056000, China

Corresponding author: Junfu Li (417616957@qq.com)


This work was supported in part by the Science and Technology Project of the Hebei Education Department, Science and Technology Research Projects of Colleges and Universities in Hebei; in part by the Handan Science and Technology Bureau Project; and in part by the School Level Project of Handan University under Grant QN2021405, Grant ZD2018207, Grant 21422021173, and Grant XZ2021202.

**ABSTRACT** Slam (simultaneous localization and mapping) play an important role in the field of artificial and driverless intelligence. A real-time dynamic visual SLAM algorithm based on an object detection network is proposed to address the robustness and camera localization accuracy issues caused by dynamic objects in indoor dynamic scenes. The YOLOv5s model, which has the smallest depth and feature map width in the YOLOv5 series, is chosen as the object detection network. The backbone network is replaced with the lightweight ShuffleNetv2 network. Experimental results on the VOC2007 dataset show that the YOLOv5-LITE model reduces the network parameters by 41.89% and speeds up the runtime by 39.00% compared to the YOLOv5s model. A motion level division strategy is adopted to provide prior information to the object detection network. In the tracking thread of the visual SLAM system, a parallel thread combining the improved object detection network and multi-view geometry is introduced to eliminate dynamic feature points. The experimental results demonstrate that in dynamic scenes, the proposed algorithm improves the camera localization accuracy by an average of 85.38% compared to ORB-SLAM2. Finally, experiments in a real environment are conducted to validate the effectiveness of the algorithm.

**INDEX TERMS** YOLOv5-LITE, dynamic environment, SLAM, dynamic feature point removal.

## I. INTRODUCTION

With the development of science and technology, mobile robots are applied to various fields, and SLAM systems play a crucial role in the development and implementation of robots. The common sensors carried by robots include cameras, LIDAR, and inertial sensors. Cameras are often used in simultaneous localization and mapping of robots due to the large amount of information available and their adaptability to multiple complex environments. vision SLAM uses the camera to sense the environment and then uses the picture data to determine the pose of the camera and the pose of the robot.

The associate editor coordinating the review of this manuscript and approving it for publication was Anandakumar Haldorai .

Currently, most visual SLAM algorithms operate based on static assumptions. For example, ORB-SLAM [1], ORB-SLAM2 [2], RGB-D SLAM [3], and LSD-SLAM [4]. These algorithms can achieve satisfactory results in static environments or environments with a small number of dynamic objects. However, the performance of the vision SLAM algorithms degrades significantly when the robot is operated in an environment with a large number of dynamic objects. This is a result of the visual characteristics of the moving objects in the environment, which have an impact on robot position estimation and significantly lower positioning accuracy. Usually, there are two basic requirements for vSLAM: robustness in tracking and real-time performance. Therefore, how to detect dynamic objects in the filled scene and reject them while guaranteeing real-time performance is the main challenge [5].

As deep learning technology has advanced in recent years, more and more top-notch deep learning algorithms have been used for visual SLAM [6].

In order to address the issue of dynamic objects in the environment, a lightweight semantic SLAM system based on parallel target detection is proposed in this study. The system is mainly based on the ORBSLAM2 framework under which target detection threads and multi-view geometry method threads are added to the original framework. In order to increase the real-time performance of the target detection thread and hence the overall system real-time performance, the lightweight ShuffleNetv2 framework is employed in place of the original skeletal network structure CSPDarknet53. Compared to YOLOv5s, the proposed algorithm achieves a 39% improvement in inference speed per frame. In the multi-view geometry thread, a multi-view geometry approach is used to eliminate dynamic feature points outside the target detection frame. The interference of dynamic points is reduced and the accuracy of the algorithm is increased to the greatest extent possible by integrating semantic and multi-view threads. Compared to ORBSLAM2, the proposed algorithm achieves an average improvement of 85.38 in Absolute Trajectory Error (ATE) for the four dynamic sequences. For the three highly dynamic sequences in the TUM dataset, the proposed algorithm achieves an average improvement of 54.20% in Absolute Trajectory Error (ATE) compared to DS-SLAM, indicating an increase in trajectory accuracy.

The main contributions of this paper are.

1. Propose a new real-time SLAM system based on ORBSLAM2. Validate the effectiveness of RTD-SLAM algorithm under tum dynamic data set and laboratory environment
2. The target detection thread is used as a separate thread to provide a priori knowledge for a lightweight target detection network based on object motion hierarchy division to remove dynamic feature points a priori. The geometry thread is also introduced to remove dynamic feature points using a multi-view geometry approach.
3. The original CSPDarkNet-53 network of YOLOv5 is replaced with a lightweight ShuffleNetv2 network, and other lightweighting procedures are carried out to decrease the time overhead of target identification, which in turn enhances the real-time performance of the entire system.

## II. RELATED WORK

Object detection is an important task in the field of computer vision. The detection algorithms based on deep learning can be divided into single-stage object detection, multi-stage object detection and others object detection.

Single stage target detection such as the you only look once (YOLO) series [7], [8], [9], [10], [11], single-shot multibox detector (SSD) [12], and RetinaNet [13], which directly output the location and category of densely bounding boxes from features in a single-shot way.

Common Two-stage detector algorithms include RCNN [14], fastRCNN [15], and fasterRCNN [16]. In the first stage, the multi-stage detection method uses the region suggestion algorithm to extract the features of the foreground region from the preset dense candidate objects. Revert to the bounding box of the object in the next step. The limitation of this structure is that it reduces the detection speed.

Other types of object detection algorithms mainly include Law et al., who proposed CornerNet to locate objects by finding the upper left corner point and lower right corner point of the object boundary box, and then classifying the diagonal points to obtain the category of objects [17]. Liang et al. proposed the DetectFormer algorithm by using the proposal category and global information from the global feature extraction encoder (GEE) to assist the ClassDecoder method, and introduced it to target detection in the field of automatic driving, thus improving the accuracy of road target detection [18]. Liang et al. proposed an improved sparse R-CNN method, which combines coordinate attention blocks with ResNeSt, and builds a feature pyramid to modify the backbone network, so that extracted features can focus on important information, thus improving detection accuracy [19].

Control strategy and environment modeling for robots is a topic known as Active Simultaneous Localization and Mapping (SLAM) [20]. According to the types of sensors employed, which are classified as LIDAR SLAM [21], Visual SLAM [22] and Multisensor Fusion SLAM [23], visual sensors are frequently used in the field of SALM due to its advantages of gathering a wealth of information and consuming little power.

Traditional visual SLAM algorithms such as the semi-direct method [24], direct method [25], and feature point method [1], [2] are assumed to be used in static environments, so they are less robust and accurate in dynamic environments. Among them, SLAM based on feature point method emerged a number of excellent SLAM systems for dynamic environment processing. According to the amount of feature information used for matching, visual features can be divided into two levels: low-level features such as pixel patches, points, or lines, and high-level features such as semantically labeled objects [26]. Low-level features focus on local details such as textures or the geometric primitives of objects and scenes. High-level features integrate details into semantic labels that more closely match the human understanding of the High-level features integrate details into semantic labels that more closely match the human understanding of the world.

The main difficulty of dynamic SLAM research based on low-level features is to distinguish dynamic feature points from static feature points, i.e., to perform motion segmentation. The method often used to perform running segmentation is the optical flow method [27] (optical flow method is to detect dynamic objects by first estimating the camera's own motion and then based on the optical flow between the predicted image and the observed image) and

geometric method [28] (setting a threshold with geometric constraints for static features to detect dynamic features).

Fang et al. [29] use optimum-estimation and uniform sampling method to detect dynamic objects. but the method is less accurate and more computationally intensive, especially when sampling densely in a big image. Zhang et al. proposed the FlowFusion algorithm [30], which uses depth and intensity information to estimate the camera's own motion from coarse to fine, and then uses this motion to compute the scene optical flow (the movement of target pixels in the image due to the movement of objects in the image or the movement of the camera in two consecutive frames) to detect dynamic features. The accuracy is improved but the application scenario of this algorithm is limited because the optical flow method is sensitive to light.

For geometric methods, the constraints can be obtained from the polar line equation [31], the inverse projection rays [32], the camera's own estimation [33], and the reprojection errors [34]. First, all features are assumed to be static. Under this assumption, epipolar lines, 3D landmark positions (last square solution), camera poses, or projections can be estimated. Then, the errors between estimates and measurements can be computed and dynamic features can be detected according to a preset threshold.

Since dynamic objects cannot be determined directly without using semantic information, dynamic objects need to be detected by successive matching of history frames. If too many history frames are used it will lead to too much computation, and if too few history frames are used it will lead to unrobust results. In order to solve the problem of unrobustness with too few historical frames, Du [35] constructed a probabilistic model and used Conditional random fields (CRFs) [36] with long-term observation to detect dynamic features. The algorithm is not intuitive due to the lack of semantic information.

In comparison to low-level features, high-level features have semantic labels with superior discriminative qualities. High-level feature representation in image space, target detection frames and pixel-level masks.

Xiao et al. proposed Dynamic-SLAM algorithm [37], which uses SSD [12] target detection model to detect dynamic objects a priori, and proposed Missed detection compensation algorithm to improve the recall rate. Zhong et al. proposed a Detect-SLAM [38] algorithm, which classifies feature points into four categories, high probability stationary, low-probability stationary, low-probability motion, and high-probability motion. Using SSD target detection algorithm a priori for people, dogs, cars, etc. will be considered as potentially moving and to improve the real-time a method to propagate dynamic feature probabilities in real time is proposed.

Both Dynamic-SLAM and Detect-SLAM algorithms use the target detection frame as a semantic label, but the semantic information alone cannot remove non-a priori dynamic

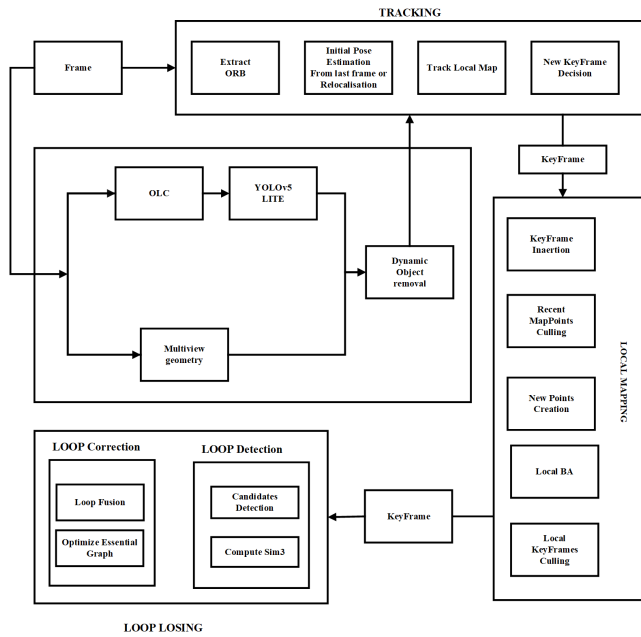
objects from the environment. For example, a book held by a person who is walking around.

Berta et al. proposed the DynaSLAM algorithm [39] using a multi-view geometry approach that can solve the problem of the presence of non-a priori dynamic objects in the environment. The DynaSLAM algorithm supports both monocular and RGB-D cameras and is processed differently for different cameras. When using monocular cameras, Mask R-CNN [40] is used to detect moving objects; when using RGBD cameras, a combination of multi-view geometry and Mask R-CNN is used. However, due to the very high time overhead of the Mask R-CNN algorithm, the algorithm does not meet the requirements in terms of real-time performance.

Yu proposed the DS-SLAM algorithm [41] to filter out the dynamic part of the scene using a combination of the semantic segmentation network SegNet [42] and a move consistency checking method. ORB feature points are first extracted, and then the movement consistency of these points is checked and potential dynamic feature points are recorded. Semantic segmentation and feature point extraction and dynamic point detection are performed in parallel, and the number of dynamic points falling within the boundary of this object is detected if the object present is classified as a moveable class. If this number exceeds a certain value, all points within the boundary of this object are considered as dynamic points, and then all points of this object are considered as outlier points. If no movable object is detected, the pose estimation is performed directly, otherwise, the outer points are filtered and then the pose estimation is performed. The SegNet algorithm used in this scheme has a lower time overhead than the Mask R-CNN used in DynaSLAM, but because the polar line constrained method does not find all the outlier points, this method fails when the object moves along the polar line direction, resulting in a loss of accuracy.

### III. SYSTEM DESCRIPTION

On the basis of ORB-SLAM2, the system proposed in this paper is enhanced. Figure 1 depicts the block diagram of the algorithm's overall structure in this study. The target detection thread and the multi-view geometry thread have been added to the upgraded framework. First, the camera collects the image data. Then, the data is passed to the tracking thread for pre-processing, and the lightweight YOLOv5-LITE algorithm identifies all the a priori dynamic objects, while the dynamic feature points in the image are discriminated and removed using the multi-view geometry thread. Second, the recognition results of the YOLOv5-LITE algorithm are combined with the motion state information discriminated based on the multi-view geometry dynamic objects and used to extract regions of dynamic objects. Finally, the feature points of dynamic object regions are removed and the image frames with only static features are input to the subsequent tracking thread and map construction thread to improve the robustness and accuracy when using low-performance devices to execute the SLAM system.

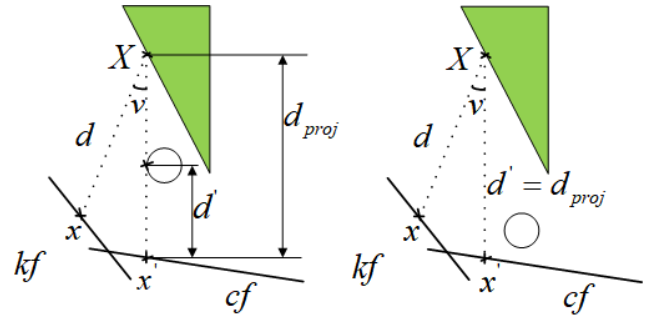


**FIGURE 1.** Algorithm structure diagram of RTD-SLAM. OLC: Object motion level classification. Object motion classification provides prior knowledge for target detection algorithm. Potential dynamic objects are detected using YOLOv5-Lite, and the detection information is transmitted to the multi-view geometry thread through shared memory, where dynamic feature points are eliminated. The tracking thread receives points for static features.

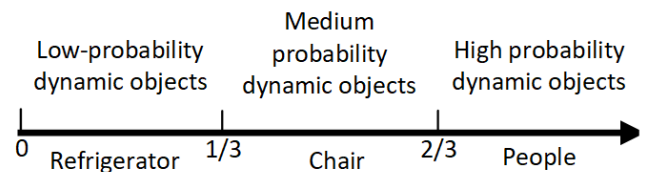
**A. MULTI-VIEW GEOMETRY**

The multi-view geometry method is a geometric model that describes how different images and objects from the same scene are projected onto one another from various angles.

Multi-view geometry detects dynamic points by removing feature points with large errors based on the positional constraints of the images between multiple frames. The projection points  $x'$  from the previous keyframe to the current keyframe are computed for each keypoint  $x$ , and their projection depth  $d_{proj}$ , while the corresponding 3D point  $x$  is generated. The angle  $xXx'$  formed by the keypoint  $x, x'$  3D point  $X$  is computed, denoted as  $\nu$ . If  $\nu$  is greater than a certain threshold value then the point is considered as possibly blocked and it is not processed. In this paper, we set the threshold  $\nu$  as 30 degrees. We observed that, in the TUM dataset, for parallax angles greater than 30degrees static objects were considered as dynamic due to their viewpoint difference.  $x'$  corresponds to a depth value of  $d'$ , within the error allowed, it is compared with  $d_{proj}$ , and the point  $x'$  is also considered to correspond to a dynamic object if it exceeds a certain threshold. The depth value  $d_{proj}$  is obtained by projecting the keypoint  $x$  from the image coordinate system to the camera coordinate system and calculating its depth value in the camera coordinate system. The projected depth is used to detect the positional changes of objects between different frames. For static objects, their projected depth is typically relatively stable because their positional changes across multiple frames are small. On the other hand, for dynamic objects, their projected depth may undergo significant variations due to their different positions and



**FIGURE 2.** Schematic diagram of the process of multi-view geometry to discriminate dynamic feature points. Keypoint  $x$  from the Key Frame(KF) is projected into the Current Frame(CF) using its depth and camera pose, resulting in point  $x'$  with depth  $d'$ . The projected depth  $d_{proj}$  is the computed.



**FIGURE 3.** Schematic diagram of the movement level classification strategy.

motion states across frames. By comparing the projected depths of keypoints, it is possible to determine whether a point corresponds to a dynamic object.  $\Delta D = d_{proj} - d'$ , and if  $\Delta D$  exceeds a threshold  $\tau_d$ , it is also considered that  $x'$  corresponds to a moving object. In this paper,  $\tau_d$  is set as  $0.4m$ . To set the threshold  $\tau_d$ , we manually tagged the dynamic objects of 30 images within the TUM dataset, and evaluated both the precision and recall of our method for different thresholds  $\tau_d$ . By maximizing the expression  $0.7 \times Precision + 0.3 \times Recall$ , we concluded that  $\tau_d = 0.4 m$  is a reasonable choice. The process of determining dynamic points by multi-view geometry is shown in Figure 2.

**B. OBJECT MOTION LEVEL CLASSIFICATION**

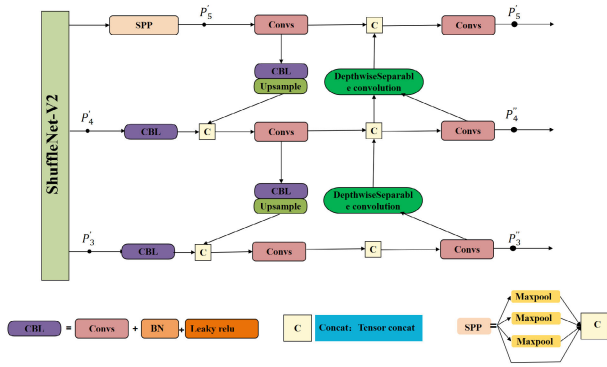
The classification of the objects into three categories-high probability dynamic objects, medium probability dynamic objects, and low probability dynamic objects (static)-is the technique used in this research.

High probability dynamic objects are mainly people and animals. In typical scenarios, people and animals are in motion.

Medium probability dynamic objects are mainly chairs, books, keyboards and mice, which are easily moved.

Low-probability dynamic objects are mainly objects that have a high probability of not being moved, such as TVs and refrigerators. As shown in Figure 3.

For low dynamic objects we consider it to be stationary, so as to ensure the accuracy while eliminating the multi-view geometry judgment process and reducing the time overhead. Feature points are only eliminated if they are in the high dynamic object box and not in the low dynamic object box.



**FIGURE 4.** Improved YOLOv5-LITE network structure diagram. Three feature layers are extracted from the YOLOv5-Lite backbone network, ShuffleNetv2, for object detection.

**C. OBJECT DETECTION ALGORITHM**

YOLOV5 [7] uses CSPDarknet53 to extract features, but the model is computationally complex and requires a lot of memory space, which is not favorable for deployment in scenarios with high real-time requirements.

In this study, a YOLOv5-based lightweight network is proposed to construct a YOLOv5-LITE lightweight network model by replacing the CSPDarknet53 with the ShuffleNetV2 architecture as a feature extraction network based on the traditional YOLOv5 model. ShuffleNetV2 [43] inherits both the ShuffleNet [44] group convolution and channel rearrangement, and follows the four guidelines for designing lightweight networks. ShuffleNetV2 outperforms other models in terms of speed and accuracy under the same circumstances.

YOLOv5-Lite network model, as shown in Figure 4. The improvements are shown as follows:

1. Replacement of the backbone network. The lightweight ShuffleNetV2 architecture is replaced with the YOLOv5 original CSPDarkNet-53 network to reduce the network parameters.

2. In the PANet structure, depth-separable convolution is used instead of downsampling to improve the accuracy of model detection.

The SLAM system based on YOLOv5-LITE proposed in this paper has the advantages of less time consuming and low performance requirement of the device, and the application scenario is more extensive. It can be mounted on unmanned vehicles and other applications with high real-time requirements and low device performance requirements, while the former semantic SLAM system can only be applied to high-performance devices and the real-time performance is not good enough.

**D. DYNAMIC FEATURE POINT REJECTION BASED ON MULTI-VIEW GEOMETRY**

Using object detection algorithms, most objects with prior information can be detected. But there are some objects in the environment that can't be detected based on a priori information like the book that someone is carrying, the chair

that someone is moving, etc. The methods adopted in this paper are as follows:

The dynamic feature points in the target detection frame are marked as dynamic feature points, and the multi-view geometry method is used for secondary verification. For areas where the target detection box has no feature points, such as the person in the back, we do nothing.

The coordinates of corner points in the upper left corner and lower right corner of bounding box can be obtained by object detection algorithm.

The feature points located in the target detection frame are judged by the multi-view geometry method. If the multi-view geometry method is judged as dynamic feature points, that is, dynamic feature points. If the multi-view geometry method determines that the feature point is static, it is static feature point.

The prediction frame generated by the YOLOv5-Lite target detection network after inference on the image consists of the position parameters  $x, y, w$  and  $h$ .  $x$  and  $y$  are the relative values of the center of the prediction frame to the original image, respectively, and  $w$  and  $h$  denote the relative values of the length and width of the prediction frame to the original image.

As shown in Figure 5, taking the prediction frame A as an example, this paper firstly converts its position parameters  $x, y, w$  and  $h$  into the coordinates of the top left and bottom right vertices of the prediction frame under the original image, which are set as  $(X_{A1}, Y_{A1}), (X_{A2}, Y_{A2})$  respectively (in the original image, the top left vertex is the coordinate origin, set to the right as the positive direction of X-axis, and set to the bottom as the positive direction of Y-axis). In Figure 5, the black points are static feature points and the green points are dynamic feature points.

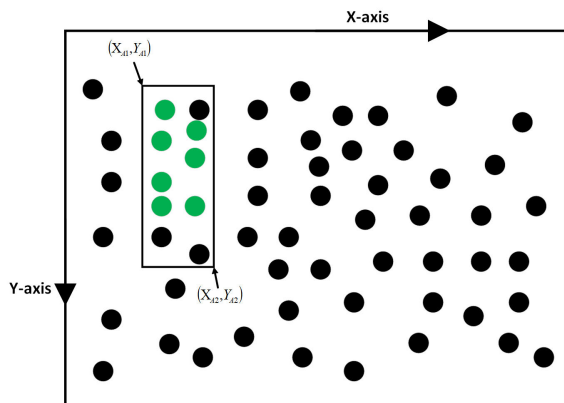
The formula in Eq (1) and Eq(2) is used to translate the target detection network's output parameters into the coordinates of the target detection frame underneath the original image.

$$\begin{cases} \frac{(X_{A1} + X_{A2})}{2l} = x \\ \frac{(X_{A2} - X_{A1})}{l} = w \end{cases} \quad (1)$$

$$\begin{cases} \frac{(Y_{A1} + Y_{A2})}{2d} = y \\ \frac{(Y_{A2} - Y_{A1})}{d} = h \end{cases} \quad (2)$$

where,  $d$  is the height of the original image,  $l$  is the width of the original image.

The ORB feature points contained in the prediction box are all undetermined dynamic feature points based on prior knowledge judgment, and all the feature points outside the prediction box are static feature points. First, assume that the set of all feature points is  $\mathbf{P} = \{p_i\}_{i=1}^n$ , the set of undetermined dynamic feature points is  $\mathbf{R} = \{r_i\}_{i=1}^n$ , the set of static feature points is  $\mathbf{S} = \{s_i\}_{i=1}^n$ ,  $\mathbf{P} = \mathbf{R} \cup \mathbf{S}$ . All the feature points in the set  $\mathbf{R}$  will participate in the screening of dynamic feature points, and the coordinate information  $(x, y)$  of each feature



**FIGURE 5.** Distribution schematic diagram of feature points. A multi-view geometry thread is used to identify and eliminate the dynamic feature points. As shown in the figure, the green points are dynamic feature points and will be removed. The black points are static feature points that will be retained.  $(X_{A1}, Y_{A1})$  and  $(X_{A2}, Y_{A2})$  is calculated according to the target detection box angle point coordinates.

point is calculated due to the front-end of the ORBSLAM2 system.

The experimental results are shown in Figure 6, not only the person detected in front of the picture. And the books in its hands and the chairs influenced by people were also detected as dynamic. If only deep learning methods are used, this results in dynamic feature points being detected as static. It can be observed from Figure xx that the real dynamic feature points are completely filtered out, and only the static feature points are retained.

## IV. EXPERIMENT AND ANALYSIS

### A. YOLOv5-LITE EXPERIMENT

In order to verify the performance of the YOLOv5-Lite target detection network, the training hyperparameters are set as follows: The initial learning rate is set to 0.01, the adam optimizer is used, the optimizer's momentum is set to 0.937, the weight's attenuation is set to 0.0005, the batch-size is set to 16, the training is 300 rounds, and the bounding box regression loss  $L_{box}$ 's hyperparameter is set to 0.05. The hyperparameter of target object loss  $L_{obj}$  is set to 0.7, the hyperparameter of classification loss  $L_{cls}$  is set to 0.3, and the threshold of iou is set to 0.2. The data set used is VOC2007, which contains 20 kinds of pictures, such as people, cats, tv, etc.

The performance test results of YOLOv5-Lite and YOLOv5s improved after training are shown in Table 1. As can be seen from Table x, the network parameters of YOLOv5-Lite are reduced by 77.82% and the running speed is accelerated by 39% compared with YOLOv5s. In terms of accuracy, Shuffnetv2 network has the advantages of small model and fast speed, but the detection performance is insufficient, so the improved target detection network decreases by 4.63% compared with YOLOv5s, sacrificing part of accuracy and improving the operating speed of the system. The experimental results are shown in Figure 7.

**TABLE 1.** Performance test comparison between YOLOv5-Lite and YOLOv5.

algorithm	parameters	GFLOPS	Each Frame/ms	Precision%	Recall%
YOLOv5	7073569	16.1	20.0	95	96.6
YOLOv5-Lite	1568673	3.8	12.2	90.6	88

### B. DATASET

The TUM RGB-D dataset consists of 39 sequences recorded in different indoor scenes using Microsoft Kinect sensors and contains data for Testing and Debugging, Handheld SLAM, Robot SLAM, Structure vs. Texture, Dynamic Objects, 3D Object Reconstruction, Validation Files, Calibration Files, and several task-specific datasets, each of which has contains multiple data that can be used for performance testing of multiple tasks. These four highly dynamic datasets record two people walking through an office scene.

Dynamic Objects dataset contains 3 sequences, where fr1, fr2 are static scene datasets and fr3 are dynamic scene datasets. In this paper, we use four highly dynamic datasets under fr3 sequence, namely fr3\_w\_xyz, fr3\_w\_static, fr3\_w\_rpy, fr3\_w\_halfsphere. where "w" represents the state of the environment in which people are walking, xyz, rpy, halfsphere and static corresponds to the way the camera moves.

### C. EXPERIMENTAL ENVIRONMENT

To reduce the influence of hardware devices, all experiments in this paper are conducted in a desktop computer environment with RTX2070 GPU and 16GB RAM, Intel i7 CPU in hardware and ubuntu18.04 in software system.

### D. EXPERIMENTAL RESULTS

To compare the performance of the RTD-SLAM system proposed in this paper with the original ORB-SLAM2 system, we conduct experiments on the above dynamic dataset. To evaluate the robustness and accuracy of the RTD-SLAM system in a highly dynamic environment, and to objectively measure the performance of our SLAM system, we use the absolute trajectory error ATE for global consistency and the relative attitude error RPE for rotational drift for all-round evaluation. The evaluation metrics are root mean square error (RMSE) and standard deviation (S.D.), with RMSE reflecting the accuracy of the system and SD reflecting the robustness of the system.

The experimental results obtained using ORBSLAM2, DS-SLAM and the algorithm in this paper are shown in Tables 2-4 for four highly dynamic datasets under the TUM dataset, respectively.

Figure 8(a) shows the absolute trajectory error plot of ORBSLAM2 algorithm under fr3\_w\_xyz sequence, and Figure 8(b) shows the absolute trajectory error plot of RTD-SLAM algorithm proposed in this paper under fr3\_w\_xyz sequence.

The black curve in Fig. 8 indicates the ground truth, the blue curve indicates the estimated value of the algorithm, the red part indicates the absolute error between the estimated



FIGURE 6. Experimental process diagram under TUM data set.

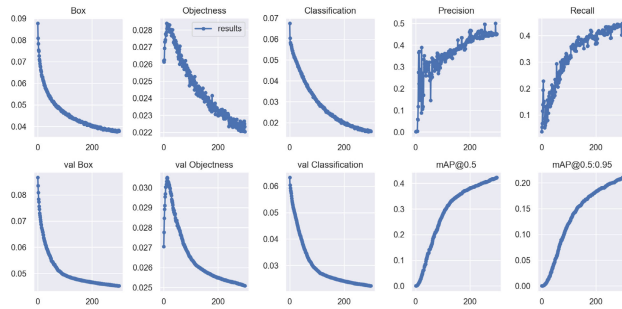


FIGURE 7. YOLOv5-Lite training results on VOC2007 data set.

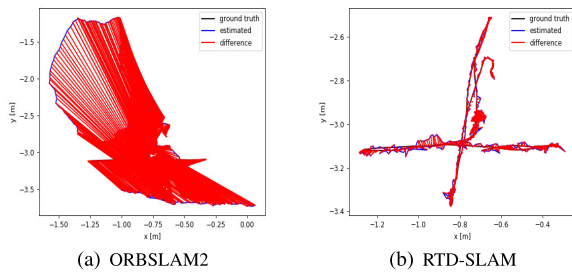


FIGURE 8. ATE comparison under fr3\_w\_xyz sequence.

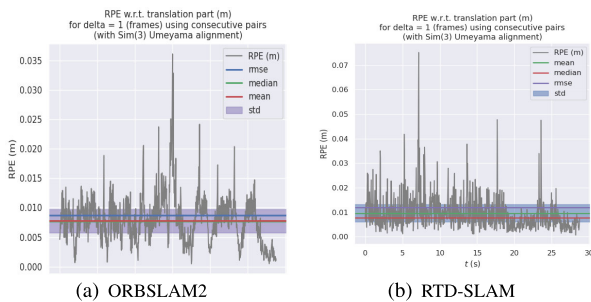


FIGURE 9. RPE comparison under fr3\_w\_xyz sequence.

value and the true value, and the smaller the red area indicates the smaller the error.

From Figure 8, it can be seen that the red area of the RTD-SLAM algorithm proposed in this paper is much smaller than the red area of the ORBSLAM2 algorithm, which intuitively shows that the ATE of RTD-SLAM is smaller than that of ORBSLAM2.

Figure 9(a) shows the relative positional error plot of ORBSLAM2, and Figure 9(b) shows the relative positional error error plot of the RTD-SLAM algorithm proposed in

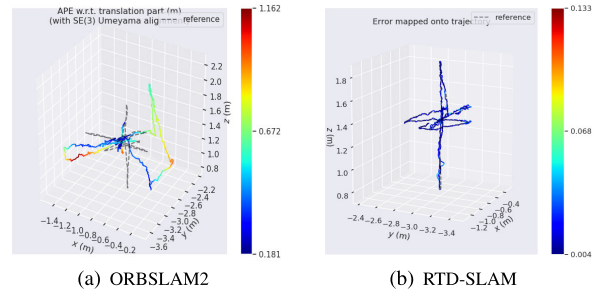


FIGURE 10. Comparison of 3D thermal map in fr3\_w\_rpy sequence.

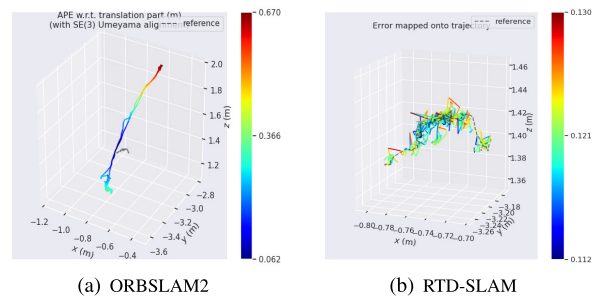


FIGURE 11. Comparison of 3D thermal map in fr3\_w\_halfsphere sequence.

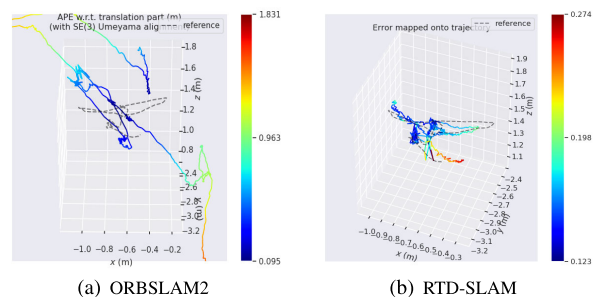


FIGURE 12. Comparison of 3D thermal map in fr3\_w\_rpy sequence.

this paper. It can be seen from Fig. 9 that the second half of the curve in the figure the RTD-SLAM proposed in this paper obviously has less fluctuation and smaller RPE than the ORBSLAM2 algorithm.

Figure 10-13 shows the comparison of the absolute trajectory error thermal map between ORBSLAM2 and the algorithm in this paper under the above four dynamic sequences. The black line represents the ground truth, and the

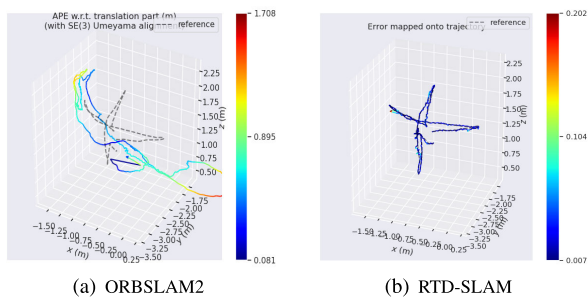


FIGURE 13. Comparison of 3D thermal map in fr3\_w\_halfsphere sequence.

TABLE 2. Results of metrics absolute trajectory error (ATE) (unit:m).

Sequences	ORB-SLAM2		DS-SLAM		Ours	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
Fr3-w-xyz	0.7521	0.3759	0.0247	0.0161	0.0199	0.0093
Fr3-w-static	0.3900	0.1602	0.0081	0.0036	0.1207	0.0020
Fr3-w-rpy	0.8705	0.4520	0.4442	0.2350	0.1667	0.0298
Fr3-w-half	0.4863	0.2290	0.0303	0.0159	0.0278	0.0244

color curve represents the estimated value of the algorithm. The more blue the color, the smaller the error of the point, and the more red the color, the larger the error of the point. Figure 10-13 intuitively shows that ORBSLAM2 will deviate seriously from the real trajectory under the dynamic data set, and the algorithm in this paper is far better than ORBSLAM2 algorithm.

Table 2 shows the ATE comparison of ORBSLAM2, RTD-SLAM and DS-SLAM algorithms proposed in this paper under four dynamic sequences.

In the “Fr3\_w\_xyz” sequence, Ours method performs exceptionally well. It exhibits high localization accuracy with very low root mean square error (RMSE) and standard deviation (S.D.) of 0.0199 and 0.0093, respectively.

The RMSE of the proposed algorithm is improved by 97.35% compared to ORB-SLAM2 and by 19.4% compared to DS-SLAM.

This indicates that Ours provides highly accurate localization with minimal variation in error when there are position changes in the dataset.

However, in the “Fr3\_w\_static” sequence, the performance of Ours method slightly decreases. The RMSE is 0.1207, and the S.D. is 0.0020.

Comparatively, our method performs less effectively in handling localization tasks in static scenes.

The RMSE of the proposed algorithm is improved by 69.05% compared to ORB-SLAM2 and reduced by 93.28% compared to DS-SLAM.

In the “Fr3\_w\_rpy” sequence, Ours method achieves excellent performance with an RMSE of 0.1667 and S.D. of 0.0298. This indicates that Ours method exhibits high precision in handling pose changes in localization tasks and demonstrates relative stability in error variation.

The RMSE of the proposed algorithm is improved by 80.05% compared to ORB-SLAM2 and by 62.47% compared to DS-SLAM.

In the “Fr3\_w\_half” sequence, Ours method continues to perform well with an RMSE of 0.0278 and S.D. of 0.0244. This implies that Ours method maintains high precision in handling localization tasks with partial scene changes and effectively mitigates errors caused by scene variations.

The RMSE of the proposed algorithm is improved by 94.28% compared to ORB-SLAM2 and by 8.25% compared to DS-SLAM. In conclusion, Ours method demonstrates outstanding performance in localization tasks involving position changes, pose changes, and partial scene changes in the dataset. However, in the case of localization tasks in static scenes, the performance of our method, Ours, is relatively poor. This issue may be attributed to the fact that in static scene localization tasks, a significant number of feature points are filtered out, resulting in a decrease in localization accuracy.

TABLE 3. Results of metric translational drift (RPE) (unit:m).

Sequences	ORB-SLAM2		DS-SLAM		Ours	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
Fr3-w-xyz	0.4142	0.2684	0.0333	0.0229	0.0120	0.0071
Fr3-w-static	0.2162	0.1962	0.0102	0.0048	0.0192	0.0133
Fr3-w-rpy	0.4249	0.3166	0.1503	0.1168	0.0192	0.0133
Fr3-w-half	0.3550	0.2810	0.0297	0.0152	0.0148	0.0090

Table 3 shows the METRIC TRANSLATIONAL DRIFT (RPE) comparison of ORBSLAM2, RTD-SLAM and DS-SLAM algorithms proposed in this paper under four dynamic sequences.

In the xyz sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.0120 and 0.0071, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 97.10% and 97.35%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 63.96% and 69.00%, respectively.

In the static sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.0192 and 0.0133, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 91.12% and 93.22%, respectively. Compared to the DS-SLAM algorithm, it shows a decrease of 88.24% and 177.10%.

In the rpy sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.0192 and 0.0133, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 95.48% and 95.80%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 87.23% and 88.61%, respectively.

In the half sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.0148 and 0.0090, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 95.83% and 96.80%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 50.17% and 40.79%, respectively.

Overall, Ours method demonstrates good performance in terms of localization accuracy across different datasets, particularly in datasets with varying positions. However, its



**TABLE 4.** Results of metric rotational drift (RPE) (unit:deg/s).

Sequences	ORB-SLAM2		DS-SLAM		Ours	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
Fr3-w-xyz	7.7432	4.9895	0.8266	0.5826	0.3889	0.2709
Fr3-w-static	3.8958	3.5095	0.2690	0.1182	0.1857	0.0949
Fr3-w-rpy	8.0802	5.9499	3.0042	2.3065	0.5597	0.3752
Fr3-w-half	7.3744	5.7558	0.8142	0.4101	0.4177	0.2362

performance is relatively poorer when it comes to handling static scenes.

Table 4 shows the METRIC TRANSLATIONAL DRIFT (RPE) comparison of ORBSLAM2, RTD-SLAM and DS-SLAM algorithms proposed in this paper under four dynamic sequences.

In the xyz sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.3889 and 0.2709, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 95.00% and 94.57%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 52.95% and 53.50%, respectively.

In the static sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.1857 and 0.0949, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 95.23% and 97.30%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 30.97% and 19.71%, respectively.

In the rpy sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.5597 and 0.3752, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 93.07% and 93.69%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 81.37% and 83.73%, respectively.

In the half sequence, the algorithm proposed in this article achieves an RMSE and S.D. of 0.4177 and 0.2362, respectively. Compared to the ORBSLAM2 algorithm, it shows an improvement of 94.33% and 95.90%, respectively. Compared to the DS-SLAM algorithm, it shows an improvement of 48.70% and 42.40%, respectively.

Based on the comprehensive analysis, Ours method exhibits lower RMSE and S.D. on all datasets compared to ORB-SLAM2 and DS-SLAM methods, indicating better performance in measuring rotational drift. These results suggest that Ours method showcases superior accuracy and stability across different datasets.

In summary, the algorithm proposed in this paper achieves good performance in the four sequences. In the static sequence, it outperforms the traditional ORBSLAM2 algorithm in terms of accuracy but still falls behind DS-SLAM. This could be attributed to the fact that DS-SLAM incorporates semantic segmentation as a prior detection, which provides an advantage.

## E. TIME CONSUMPTION EVALUATION

In real life the goodness of an algorithm cannot be considered only in terms of accuracy, but also in terms of real time. The time consumption of semantic SLAM system is mainly

**TABLE 5.** Total time consumption under the TUM dataset.

Sequences	ORB-SLAM2	DSSLAM	Ours
Fr3-w-xyz	31.34s	51.36s	37.34s
Fr3-w-static	24.69s	44.26 s	31.63s
Fr3-w-rpy	32.25s	54.05s	38.40s
Fr3-w-half	38.73s	63.40 s	45.02s

**FIGURE 14.** YASKAWA MOTOMAN ROBOT.

in Semantic Thread and Tracking Thread. In order to verify the real-time performance of the algorithm in this paper, this paper compares these two aspects with DS-SLAM algorithm respectively, and the results are shown in Table 5.

In the xyz, static, rpy, and half sequences, the total running times of our algorithm are 37.34s, 31.63s, 38.40s, and 45.02s, respectively. These results meet the requirements for real-time operation.

Compared to the DS-SLAM algorithm, our algorithm shows a reduction in total processing time by 14.02s, 12.63s, 15.65s, and 18.38s in the xyz, static, rpy, and half sequences, respectively.

## V. EXPERIMENTS IN A REAL-WORLD LABORATORY ENVIRONMENT

### A. DATASET PRODUCTION

The Lenovo Legion R9000K 2021H laptop was used as the data set acquisition platform, and the Astro Pro camera was attached to the YASKAWA MOTOMAN ROBOT model TKPE-MH0005S-A00 with an external device as the data set acquisition tool to control the robot in space for pan, pitch, yaw, and roll operations. pitch, yaw and roll operations in space.

The dataset was divided into two groups. The first group, named “single-person,” consisted of an environment

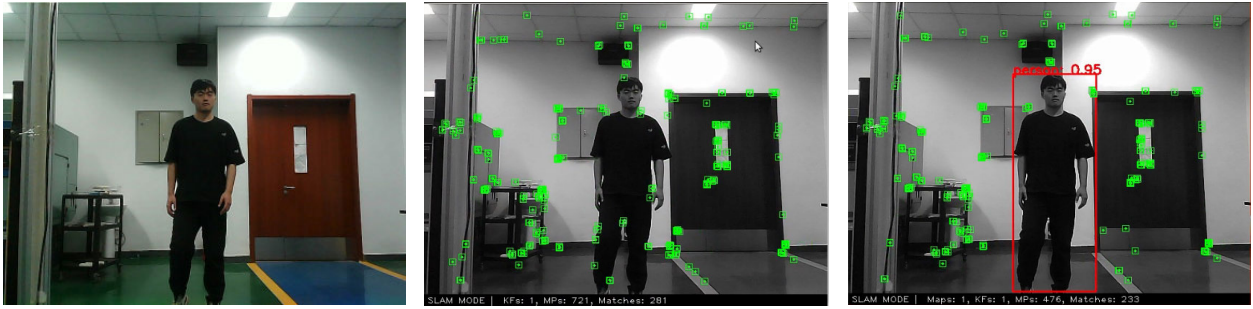


FIGURE 15. Experimental process diagram.

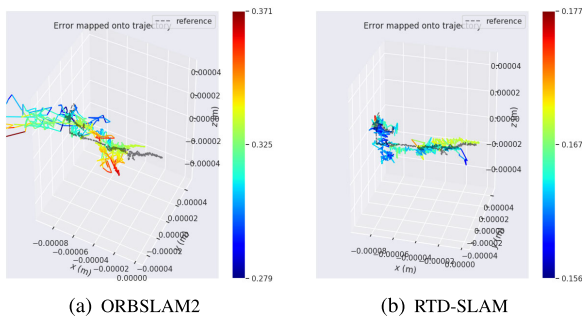


FIGURE 16. Comparison of 3D heat map in case one.

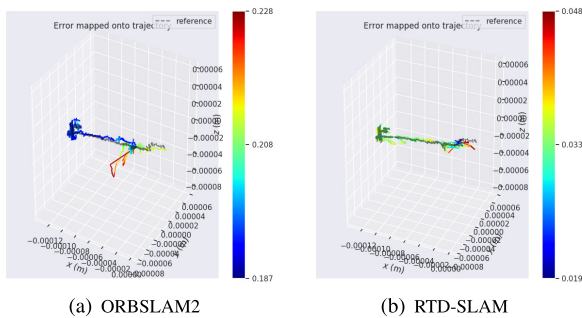


FIGURE 17. Comparison of 3D heat map in case two.

where only one experimenter walked. The second group, named “two-person,” involved an environment where two experimenters walked simultaneously. Each group gathered 2000 photos. The dataset production equipment is shown in Figure 14.

## B. EXPERIMENTAL RESULTS

The ORBSLAM algorithm and the algorithm from this paper are used in a comparison experiment, and the experimental process diagram is shown in Figure 15. The experiment demonstrates how effectively the algorithm in this paper removes the dynamic feature points.

As can be seen from the comparison of 3D thermal map in Figure 16, ORBSLAM2 algorithm deviates seriously from the real movement trajectory. Although some parts of RTD-SLAM have large errors, the overall effect is still good.

TABLE 6. Results of metrics absolute trajectory error (ATE) (unit:mm).

Sequences	ORB-SLAM2		Ours		Improvement Percentage(%)	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
single-person	0.329	0.010	0.164	0.003	50.15	70.00
two-person	0.196	0.006	0.034	0.002	82.65	66.67

TABLE 7. Results of metric translational drift (RPE) (unit:mm).

Sequences	ORB-SLAM2		Ours		Improvement Percentage(%)	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
single-person	0.004	0.003	0.003	0.002	25.00	33.33
two-person	0.002	0.002	0.001	0.001	50.00	50.00

It can be seen that RTD-SLAM algorithm is more accurate and robust in this case.

As can be seen from the comparison of 3D thermal diagram in Figure 17, ORBSLAM2 algorithm has red line segment, indicating that the trajectory error is large. The overall level of RTD-SLAM is consistent with the real trajectory, and there is no large error part. It can be seen that RTD-SLAM algorithm has better accuracy and robustness in case two.

According to the comparison between FIG. 16 and Figure 17, ORBSLAM2 algorithm accuracy in case 2 is actually better than that in case 1. By analyzing the reasons, it can be seen that the camera motion trajectory in case 2 is stable and relatively short, resulting in not obvious dynamic characteristics.

Based on the TABLE 6. data it can be seen that on two sequences (single-person and two-person), our algorithm achieves an improvement in ATE compared to ORB-SLAM2.

In the single-person sequence, our method reduces the RMSE (root mean square error) by 57.15% and the S.D. (standard deviation) by 70.00% relative to ORB-SLAM2. In the two-person sequence, our method reduces the RMSE by 82.65% and S.D. by 66.67% relative to ORB-SLAM2. This means that our method has higher accuracy and stability in camera pose and depth estimation than ORB-SLAM2.

These results indicate that the algorithm in this paper performs better in terms of ATE, i.e., the difference between the estimated results and the true values is smaller.

**TABLE 8. Results of metric rotational drift (RPE) (unit:deg/s).**

Sequences	ORB-SLAM2		Ours		Improvement Percentage(%)	
	RMSE	S.D.	RMSE	S.D.	RMSE	S.D.
single-person	0.2546	0.1978	0.0968	0.0670	61.98	66.13
two-person	0.9436	0.0631	0.0935	0.0659	90.10	-4.43

**TABLE 9. Comparison of overall time overhead on custom dataset.**

Sequences	ORB-SLAM2	Ours
single-person	66.43s	86.80s
two-person	66.39s	90.43s

Based on the data in Table 7, we can see that on two sequences (single-person and two-person), our algorithm achieves an improvement in Translational Drift (TD) compared to ORB-SLAM2.

In the single-person sequence, our method reduces the mean translational drift (RMSE) by 25.00% and the standard deviation of translational drift (S.D.) by 33.33% with respect to ORB-SLAM2. In the two-person sequence, our method reduces the mean translational drift by 50.00% and the standard deviation of translational drift by 50.00% relative to ORB-SLAM2. This indicates that our method is more accurate than ORB-SLAM2 in camera motion estimation with reduced translational drift errors.

These results indicate that our method performs better in terms of translational drift, i.e., the camera motion estimation is more accurate and the translational drift error is smaller.

Based on the data in Table 8, it can be seen that on two sequences (single-person and two-person), our method achieves a significant improvement in Rotational Drift (RMSE) compared to ORB-SLAM2.

In single-player sequences, our algorithm reduced the mean rotational drift (RMSE) by 61.98% and the standard deviation of rotational drift (sd) by 66.13% relative to ORB-SLAM2. In the Two-peoples sequence, compared to ORB-SLAM2, our algorithm reduces the mean value of rotational drift by 90.10% and increases the standard deviation of rotational drift by 4.43%. This shows that our method is more accurate than ORB-SLAM2 in camera rotation estimation, with less rotation-drift error, but with reduced stability in two-person sequences

These results indicate that our method performs better in terms of rotational drift, i.e., more accurate camera rotation estimation and less rotational drift error.

### C. TIME CONSUMPTION EVALUATION

To comprehensively analyze and compare the performance of the algorithms, this study compared the average processing time in two scenarios (single-person and two-person).

According to Table 9, it can be observed that ORBSLAM2 has an average processing time of 66.43 seconds in the single-person scenario, while the proposed method has an average processing time of 86.80 seconds. In the two-person

scenario, ORBSLAM2 has an average processing time of 66.39 seconds, whereas the proposed method has an average processing time of 90.43 seconds.

According to the data in Table 9, it can be observed that the proposed algorithm in this study has a higher time overhead compared to ORBSLAM.

## VI. CONCLUSION

This paper proposes the RTD-SLAM system, which augments the original ORBSLAM2 with target detection threads and multi-view geometry threads, in order to increase the accuracy of the SLAM system in a dynamic environment. First, a priori dynamic information is obtained by a lightweight target detection thread. The multi-view geometry thread is then used to find the dynamic feature points in the scene. Finally, experiments are carried out using four dynamic sequences under the TUM dataset to verify the performance of this algorithm, and the results show that the trajectory accuracy is significantly increased. In order to verify the real-time performance of the proposed algorithm, the time consuming of running TUM four dynamic sequences was calculated in this paper. Finally, accuracy and time-consuming experiments were conducted on the self-made dataset.

First, in order to verify the performance of the YOLOv5-Lite algorithm proposed in this paper, VOC2007 data set was used to train 300 rounds of experiments. The experimental results indicate that the proposed target detection algorithm in this paper has a smaller parameter size and faster inference speed. Compared to YOLOv5s, the inference speed is improved by 39%.

In order to validate the feasibility of the algorithm, this paper conducted multiple experiments using the publicly available TUM dataset and a self-made dataset. The experimental results demonstrate a significant improvement in accuracy of the proposed algorithm compared to the ORBSLAM2 algorithm.

In the sequences xyz, static, rpy, and half, the trajectory accuracy of the proposed algorithm in this paper is improved by 97.35%, 69.05%, 80.05%, and 94.28% respectively, compared to ORBSLAM2. Compared with the mature DS-SLAM algorithm, the average total time consuming of the proposed algorithm under four dynamic sequences was reduced by 28.44% compared with DS-SLAM. The proposed algorithm meets the standard of real-time operation, indicating that it is capable of running in real-time without significant delays or latency. This implies that the algorithm can process and analyze data in a timely manner, making it suitable for real-time applications or systems where immediate responses are required.

In the self-made dataset, in the single-person and two-person sequences, the trajectory accuracy of the proposed algorithm in this paper is improved by 50.15% and 82.65% respectively, compared to ORBSLAM2.

Although the algorithm in this paper has made some progress in accuracy and real-time performance, especially

in real-time performance, we still need to continuously improve the algorithm. In the future, the method of reducing the target detection area can be adopted to further improve the accuracy and real-time performance of the system.

## REFERENCES

- [1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.
- [2] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.
- [3] A. Fontan, R. Giubilato, L. Oliva, J. Civera, and R. Triebel, "SID-SLAM: Semi-direct information-driven RGB-D SLAM," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6387–6394, Oct. 2023.
- [4] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Computer Vision—ECCV 2014*. Zürich, Switzerland: Springer, Sep. 2014, pp. 834–849.
- [5] Y. Liu and J. Miura, "RDS-SLAM: Real-time dynamic SLAM using semantic segmentation methods," *IEEE Access*, vol. 9, pp. 23772–23785, 2021.
- [6] S. Mokssit, D. B. Licea, B. Guermah, and M. Ghogho, "Deep learning techniques for visual SLAM: A survey," *IEEE Access*, vol. 11, pp. 20026–20050, 2023.
- [7] W. Wu, H. Liu, L. Li, Y. Long, X. Wang, Z. Wang, J. Li, and Y. Chang, "Application of local fully convolutional neural network combined with YOLO v5 algorithm in small target detection of remote sensing image," *PLoS ONE*, vol. 16, no. 10, Oct. 2021, Art. no. e0259283.
- [8] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, pp. 1–10, Jun. 2015.
- [9] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *CoRR*, vol. abs/1612.08242, pp. 1–9, Dec. 2016.
- [10] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *CoRR*, vol. abs/1804.02767, pp. 1–6, Apr. 2018.
- [11] A. Bochkovskiy, C. Wang, and H. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, pp. 1–17, Apr. 2020.
- [12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision—ECCV 2016*. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 21–37.
- [13] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, vol. abs/1708.02002, pp. 1–10, Aug. 2017.
- [14] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, pp. 1–21, Nov. 2013.
- [15] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, pp. 1–9, Apr. 2015.
- [16] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, pp. 1–14, Jun. 2015.
- [17] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," *CoRR*, vol. abs/1808.01244, pp. 1–14, Aug. 2018.
- [18] T. Liang, H. Bao, W. Pan, and F. Pan, "Traffic sign detection via improved sparse R-CNN for autonomous vehicles," *J. Adv. Transp.*, vol. 2022, pp. 1–16, Mar. 2022.
- [19] T. Liang, H. Bao, W. Pan, X. Fan, and H. Li, "DetectFormer: Category-assisted transformer for traffic scene object detection," *Sensors*, vol. 22, no. 13, p. 4833, Jun. 2022.
- [20] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," 2022, *arXiv:2207.00254*.
- [21] M. U. Khan, S. A. A. Zaidi, A. Ishtiaq, S. U. R. Bukhari, S. Samer, and A. Farman, "A comparative survey of LiDAR-SLAM and LiDAR based sensor technologies," in *Proc. Mohammad Ali Jinnah Univ. Int. Conf. Comput. (MAJICC)*, Jul. 2021, pp. 1–8.
- [22] S. Wang, Y. Huang, P. Yue, N. Chen, Q. Wang, and C. Zou, "Research progress on visual slam for dynamic environments," in *Advanced Manufacturing and Automation XII*. Singapore: Springer, 2023, pp. 108–115.
- [23] C. Tian, H. Liu, Z. Liu, H. Li, and Y. Wang, "Research on multi-sensor fusion SLAM algorithm based on improved Gmapping," *IEEE Access*, vol. 11, pp. 13690–13703, 2023.
- [24] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 15–22.
- [25] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.
- [26] Z. Xu, Z. Rong, and Y. Wu, "A survey: Which features are required for dynamic visual simultaneous localization and mapping?" *Vis. Comput. Ind., Biomed., Art.*, vol. 4, no. 1, pp. 1–16, Dec. 2021.
- [27] S. Shen, L. Kerofsky, and S. Yogamani, "Optical flow for autonomous driving: Applications, challenges and improvements," 2023, *arXiv:2301.04422*.
- [28] Z. Pan, J. Hou, and L. Yu, "Optimization RGB-D 3-D reconstruction algorithm based on dynamic SLAM," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.
- [29] Y. Fang and B. Dai, "An improved moving target detecting and tracking based on optical flow technique and Kalman filter," in *Proc. 4th Int. Conf. Comput. Sci. Educ.*, Jul. 2009, pp. 1197–1202.
- [30] T. Zhang, H. Zhang, Y. Li, Y. Nakamura, and L. Zhang, "FlowFusion: Dynamic dense RGB-D SLAM based on optical flow," *CoRR*, vol. abs/2003.05102, pp. 1–7, Mar. 2020.
- [31] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 4306–4312.
- [32] D. Migliore, R. Rigamonti, D. Marzorati, M. Matteucci, and D. G. Sorrenti, "Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments," in *Proc. ICRA Workshop Safe Navigat. Open Dyn. Environ., Appl. Auto. Vehicles*, 2009, pp. 12–17.
- [33] K.-H. Lin and C.-C. Wang, "Stereo-based simultaneous localization, mapping and moving object tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 3975–3980.
- [34] D. Zou and P. Tan, "CoSLAM: Collaborative visual SLAM in dynamic environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 354–366, Feb. 2013.
- [35] Z.-J. Du, S.-S. Huang, T.-J. Mu, Q. Zhao, R. R. Martin, and K. Xu, "Accurate dynamic SLAM using CRF-based long-term consistency," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 4, pp. 1745–1757, Apr. 2022.
- [36] M. You, C. Luo, H. Zhou, and S. Zhu, "Dynamic dense CRF inference for video segmentation and semantic SLAM," *Pattern Recognit.*, vol. 133, Jan. 2023, Art. no. 109023.
- [37] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Robot. Auto. Syst.*, vol. 117, pp. 1–16, Jul. 2019.
- [38] F. Zhong, S. Wang, Z. Zhang, C. Chen, and Y. Wang, "Detect-SLAM: Making object detection and SLAM mutually beneficial," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1001–1010.
- [39] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018.
- [40] P. Bharati and A. Pramanik, "Deep learning techniques—R-CNN to mask R-CNN: A survey," in *Proc. CIPR*, 2020, pp. 657–668.
- [41] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1168–1174.
- [42] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
- [43] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.
- [44] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.



**RUIZHEN GAO** received the B.Sc. degree in mechanical design manufacturing and automation and the M.Sc. degree in technology of computer application from the Hebei University of Engineering, and the Ph.D. degree in control theory and control engineering from Chongqing University. He is currently a Professor with the Department of Mechanical and Equipment Engineering, Hebei University of Engineering, specializing in computer vision. He is also a member of the

Key Laboratory of Intelligent Industrial Equipment Technology of Hebei Province at Hebei University of Engineering, as well as a member of the Collaborative Innovation Center for Modern Equipment Manufacturing of Jinan New Area (Hebei) at Hebei University of Engineering.



**BAIHUA LI** received the B.Sc. and M.Sc. degrees in electronic engineering from Tianjin University, China, and the Ph.D. degree in computer science from Aberystwyth University. She is currently a Professor in AI, computer vision and machine learning with the Department of Computer Science, Loughborough University, and a key member of the Centre for Sensing and Imaging Science, and Vision, AI, Autonomous and Human Centered Systems Research Group.



**ZIHENG LI** received the B.E. degree from Anhui Agricultural University. He is currently pursuing the master's degree with the School of Mechanical and Equipment Engineering, Hebei University of Engineering, China. His research interests include deep learning and SLAM.



**JINGJUN ZHANG** received the B.Sc. degree from Lanzhou University, China, and the M.Sc. and Ph.D. degrees in computational mechanics from Jilin University. He is currently a Professor in computational mechanics, computer vision and intelligent robot systems with the Department of Robotics Engineering, Hebei University of Engineering, and a member of the Key Laboratory of Intelligent Industrial Equipment Technology of Hebei Province.



**JUNFU LI** received the M.Sc. degree in technology of computer application from the Hebei University of Engineering. He is currently a Professor and a Lecturer in software engineering, intelligent robot systems with the Department of Engineering, Hebei University of Engineering, and a member of the Scientific Research Team of Intelligent Robot Systems and Environmental Perception.



**JUN LIU** is currently the Technical Leader of Handan Textile Machinery Company Ltd., developing intelligent manufacturing system around robots. Related products are widely used in textile industry, metal processing industry, and metallurgical industry.

...