Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7332390, 19 pages
https://doi.org/10.1155/2022/7332390

Research Article

LRD-SLAM: A Lightweight Robust Dynamic SLAM Method by Semantic Segmentation Network

Shifeng Jia

School of Electronic Engineering, Heilongjiang University, Harbin, China

Correspondence should be addressed to Shifeng Jia; 122435443@qq.com

Received 9 September 2022; Revised 10 November 2022; Accepted 11 November 2022; Published 21 November 2022

Academic Editor: Suhua Tang

With the development of intelligent concepts in various fields, research on driverless and intelligent industrial robots has increased. Vision-based simultaneous localization and mapping (SLAM) is a widely used technique. Most conventional visual SLAM algorithms are assumed to work in ideal static environments; however, such environments rarely exist in real life. Thus, it is important to develop visual SLAM algorithms that can determine their own positions and perceive the environment in real dynamic environments. This paper proposes a lightweight robust dynamic SLAM system based on a novel semantic segmentation network (LRD-SLAM). In the proposed system, a fast deep convolutional neural network (FNet) is implemented into ORB-SLAM2 as a semantic segmentation thread. In addition, a multiview geometry method is introduced, in which the accuracy of detecting dynamic points is further improved through the difference in parallax angle and depth, and the information of the keyframes is used to repair the static background information absent from the removal of dynamic objects, to facilitate the subsequent reconstruction of the point cloud map. Experimental results obtained using the TUM RGB-D dataset demonstrate that the proposed system improves the positioning accuracy and robustness of visual SLAM in indoor pedestrian dynamic environments.

1. Introduction

With the rapid development of modern urbanization, people are paying increasing attention to navigation systems and guiding maps in indoor environments to plan itineraries. In addition, with increasing urbanization in various countries, in the event of military conflicts, cities will become the main battlefields for human-machine coordinated operations and unmanned operations [1]. At the same time, in dangerous related industrial production, how to use unmanned equipment to perform inspection and maintenance in places that humans cannot reach has also become one of the main problems. Simultaneous localization and mapping (SLAM) [2] can be implemented in modern devices that use mobile robots as vehicles that capture data using external sensing sensors to model the environment as the robot moves and simultaneously estimate its own motion without a priori information about the environment, such as unmanned ground vehicles (UGVs), unmanned aerial vehicles (UAVs), and mobile robots. SLAM is also widely used in aerial robots [3], intelligent marine navigation [4], and deep space exploration [5]. Thus, this technology plays an important role in daily life, industrial production, and military applications. Moreover, in recent years, with the rapid development of deep learning, image processing techniques (including object detection, image classification, and semantic segmentation) have been improved greatly. In the semantic segmentation of images, pixel-level semantic classification can be obtained and used to recognize the preceding attributes of each pixel in an image in advance, e.g., the most common pedestrians and pets in an indoor environment [6]. Such semantic information provides visual SLAM with relevant information about dynamic objects in the scene. As a result, many researchers have started to combine visual SLAM with deep convolutional neural networks for object detection and semantic

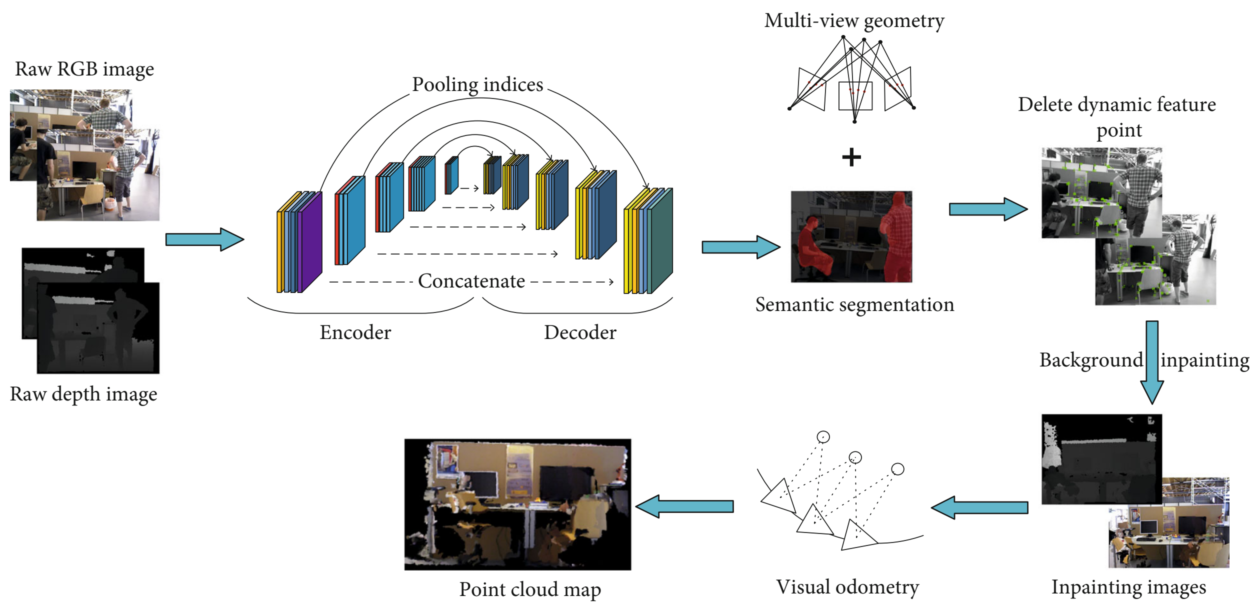


Figure 1: The proposed system first performs semantic segmentation on the input original RGB image to detect dynamic objects in the scene and combines multiview geometry to further improve dynamic object detection accuracy. Then, ORB feature points representing dynamic objects are removed. Finally, the point cloud map is generated in a separate thread.

segmentation, which enables SLAM to perceive the surrounding environment at a semantic level. Thus, deep learning-assisted visual SLAM has become a research trend.

Although the current research on visual SLAM has made some developments since most of the current visual SLAM algorithms assume an ideal static environment, which rarely exists in real life, if a dynamic object is assumed to join the current environment, the motion of that dynamic object will be calculated into the motion of the visual sensor, thus making the accuracy of the calculated poses greatly reduced, or even leading to localization and mapping failure. Then the dynamic objects here can be divided into two categories, one is objects that have their own movement characteristics, such as people, animals, and moving carriers, and the other is objects that do not have their own movement characteristics but are forced to move in their current state due to external forces, such as tables, chairs, and water glasses. Both types of dynamic objects can be the main disturbing factors for visual SLAM to calculate poses and construct maps in dynamic environments. These problems are solved in my work, which enables the visual SLAM algorithm on a mobile device to determine the position and perceive the environment more efficiently in real dynamic environments.

Thus, herein, a visual SLAM algorithm for an indoor pedestrian dynamic environment is proposed. The proposed system adopts a new lightweight convolutional neural network for semantic segmentation. In addition, a tracking preprocessing stage is introduced to the tracking thread to eliminate the dynamic parts and repair the background in the image. Then, a reliable ORB feature point is input to the subsequent threads. Finally, a dense point cloud map is generated. Our main contributions are summarized as follows.

First, a lightweight robust dynamic SLAM (LRD-SLAM) for indoor dynamic environments is proposed. LRD-SLAM mainly improves real-time performance while ensuring positioning accuracy in a dynamic environment and has strong robustness when working.

Second, a fast deep convolutional neural network (FNet) is proposed for semantic segmentation. This convolution neural network is used as a semantic thread that enables the fast and accurate identification of information on pedestrians in the given scene. The segmentation target can be covered with a binary mask overlay.

Third, a new dynamic object culling strategy algorithm is proposed. In this algorithm, a multiview geometric method based on parallax angle and depth values is employed to determine whether the target is a potential dynamic object. Then combined with the dynamic objects detected by semantic segmentation, the dynamic objects are processed through the dynamic object culling strategy.

The remainder of this paper is organized as follows. Section II reviews the visual SLAM methods in three types of dynamic environments. Section III introduces the basic IRD-SLAM framework, the principles of a new deep convolutional neural network for semantic segmentation, and the preprocessing stage of tracking. In Section IV, this paper discusses the results of applying the proposed method to the TUM dataset [7] to verify the performance. Finally, the paper is concluded in Section V.

2. Related Work

With the development of visual SLAM in recent years, an increasing number of mature achievements are being demonstrated, e.g., PTAM [8], ORB-SLAM [9], LSD-SLAM [10], and DVO-SLAM [11]. However, these methods are

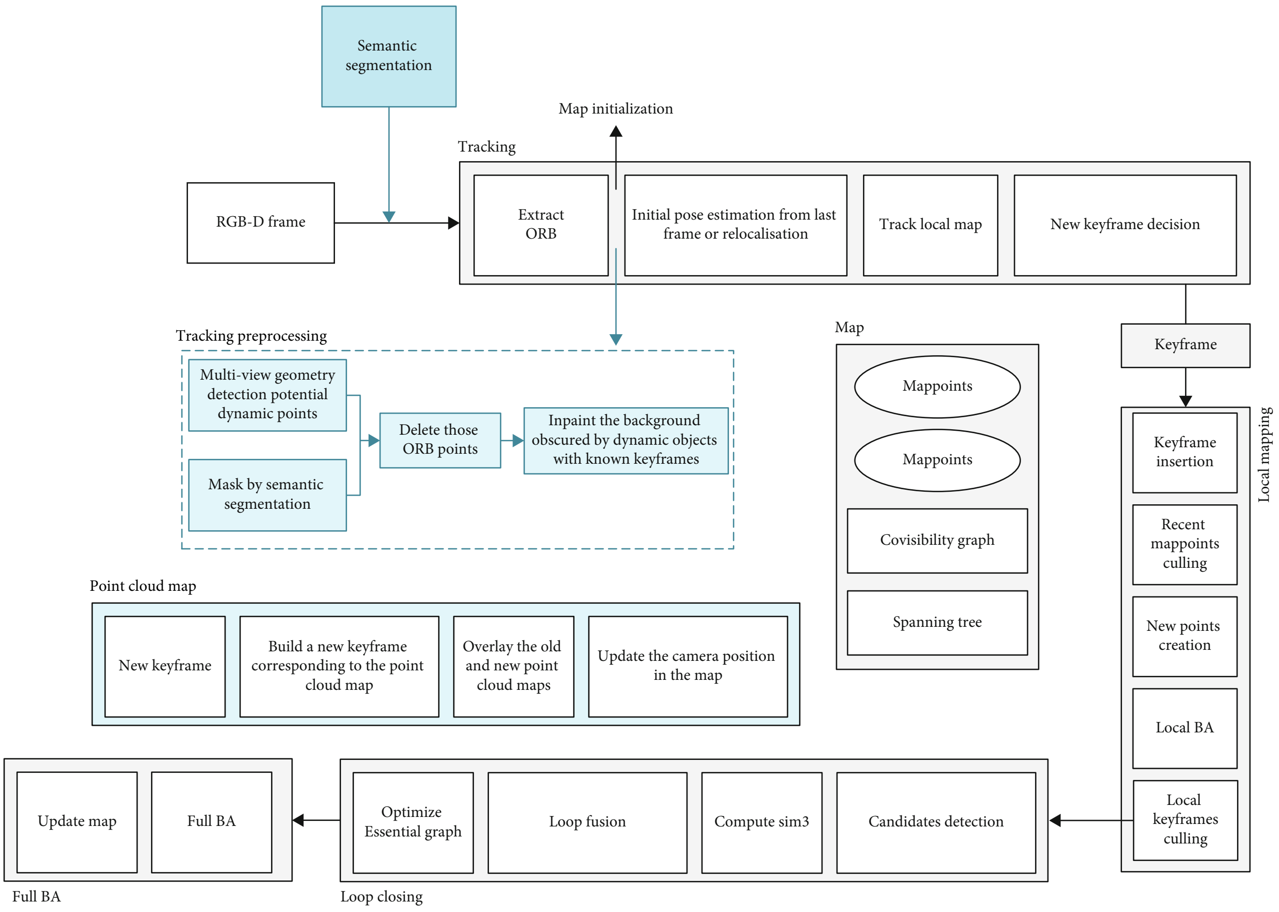


Figure 2: The basic framework of visual SLAM in indoor dynamic environment. The local mapping thread, map thread, and closed-loop detection thread in ORB-SLAM2 are adopted. A semantic segmentation thread is embedded before the tracking thread to detect dynamic objects. After ORB feature points are extracted, the tracking preprocessing stage is implemented to uniformly remove the dynamic points of the input RGB image and repair the background. Finally, a separate thread is employed to construct a dense point cloud map.

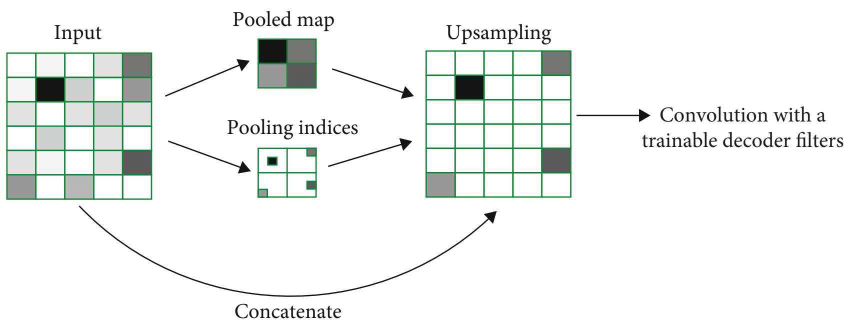


Figure 3: Assume a 3×3 filter is used for max pooling with two steps. A 2×2 pooled map is generated via max pooling, and the relative position of the weights selected by max pooling in the 2×2 filter is saved as pooling indices. In the upsampling process, the pooling indices information is used to directly put the data back to the corresponding position, and then the input feature map and corresponding upsampling layer are concatenated to enrich and deepen the texture features. Finally, deconvolution is performed to train and learn.

primarily applied to static environments, and if they are disturbed by dynamic objects in an indoor environment with high dynamics, they cannot achieve effective results. Thus, visual SLAM in dynamic situations has become a focus in the robotics field. In recent years, the main tools in visual SLAM for dynamic environments include deep learning, deep learning combined with geometric methods, and deep learning combined with other methods.

2.1. Deep Learning in Visual SLAM. Deep convolutional neural networks, which are a key technique in deep learning, are applied to target detection and semantic segmentation to

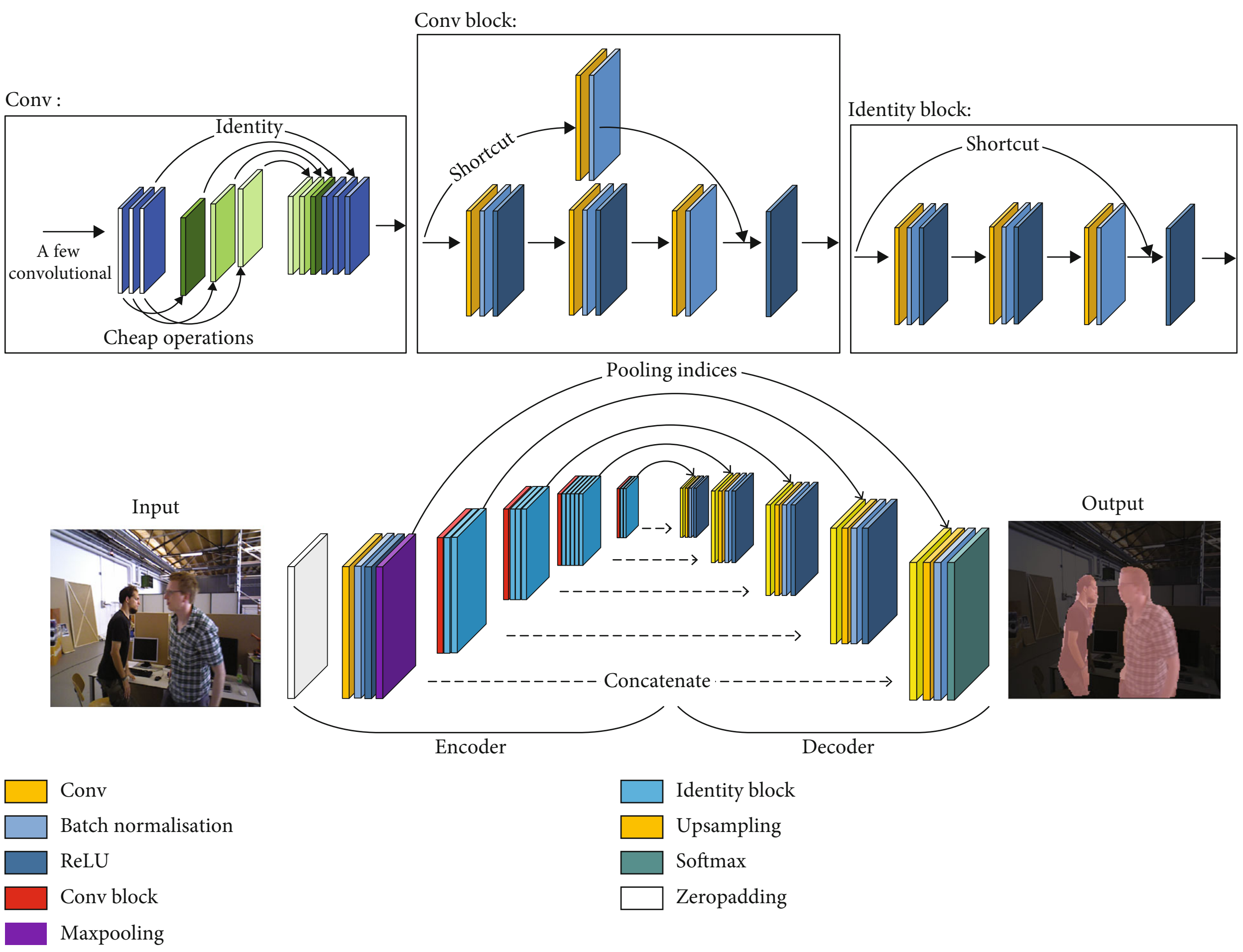


FIGURE 4: Network structure diagram. The convolutional layer in the encoder is replaced by the Conv model given in the figure, which effectively reduces the number of parameters and improves calculation speed. In the encoder, the residual block (comprising the Conv block and identity block) is employed to stack to extract deeper features, and the features generated by each residual block establish the pooling indices with the corresponding decoder part. In the decoder, the size of the input feature map is increased via upsampling two times, and the concatenation feature fusion method is employed to improve the accuracy of each layer. Subsequently, deconvolution is employed to fill in the missing weights, where the convolution layer is the original convolution operation. Finally, a SoftMax layer is implemented as a full-connected layer to output the maximum value of the different categories.

```
Input: The original image frame $F$, the feature point set extracted from the image frame $P=[p_1, p_2, \cdots, p_n]$.
Output: Dynalist.
1: Extract dynamic objects in the input image frames using FNet and overlay them with a$Masks$;
2: Multiview geometric approach to find potential dynamic points $p_d$;
3: For $i = 1 : n$ do
4:   If $p_f \in Masks$ then
5:     $p_f$==dynamic;
6:     Dynalist $\longleftarrow$ added;
7:   End if
8:   If $p_f == p_d$ then
9:     $p_f$==dynamic;
10:    Dynalist $\longleftarrow$ added;
11:  End if
12:  End for
13: Remove dynamic points from Dynalist;
```

ALGORITHM 1: Dynamic objects elimination strategy.

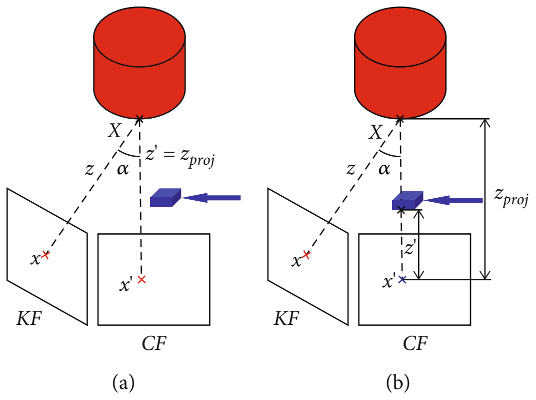


FIGURE 5: Multiview geometry diagram. The projected point x' in (a) is a schematic diagram of a static point ($z' = z_{\text{proj}}$). The projected point x' in (b) is a schematic diagram of a dynamic point ($z' \ll z_{\text{proj}}$).

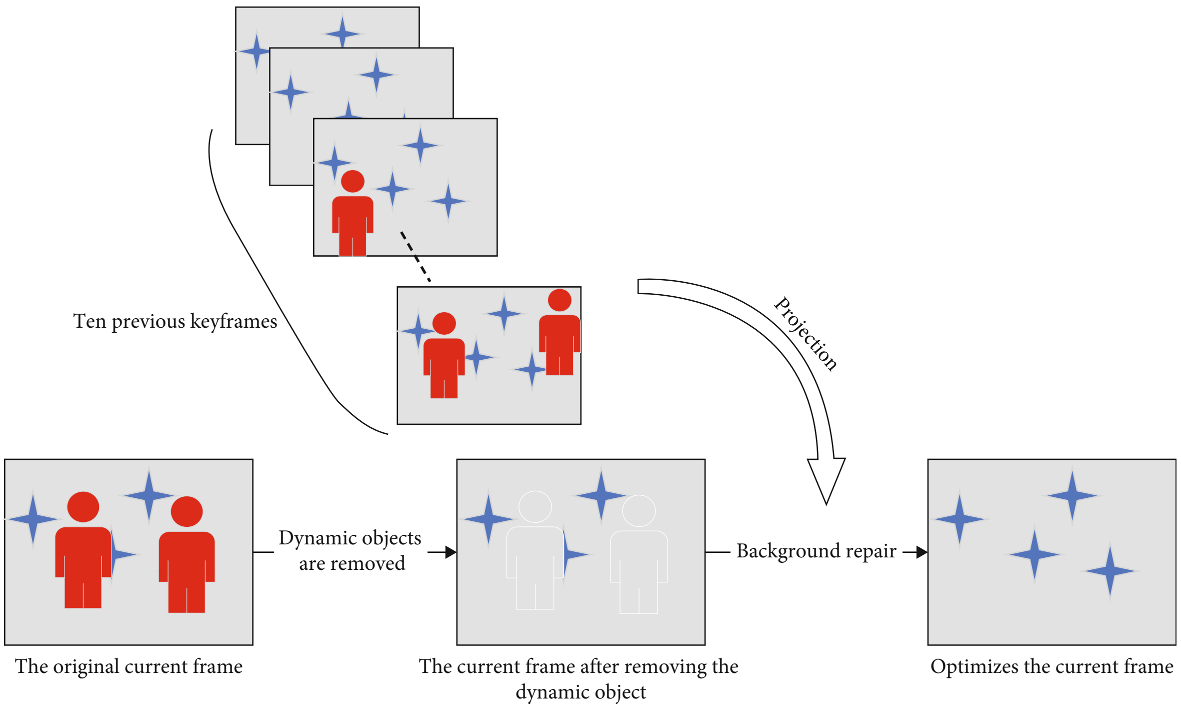

Ten previous keyframes
Projection
Dynamic objects are removed
Background repair
The original current frame
The current frame after removing the dynamic object
Optimizes the current frame

FIGURE 6: Schematic of background restoration principle. The red human figure is a dynamic object, and the blue star is a static background. The static background occluded using the red human figure is restored by the background restoration method, and the final optimized image frame is obtained.

TABLE 1: Training network parameter setting table.

The parameter name	Selection strategy
Optimizer	Adam
Learning rates	Initial value 1e-3
The activation function	ReLu
Batch size	4
Number of rounds	200

obtain the classification or semantic labeling information for each image pixel in an image. Many classical algorithms have been developed, such as the YOLO series [12], single-shot multibox detector (SSD) [13], Fast-RCNN [14], SegNet [15], UNet [16], Mask-RCNN [17], PSPNet [18], and the DeepLab series [19]. Pure deep learning has been used to assist visual SLAM in earlier research. Zhang et al. [20] employed YOLO to detect the dynamic objects in an environment and removed them so that a visual SLAM system would not be disturbed. Similarly, Zhong et al. [21] combined visual SLAM with an SSD to detect objects in each frame of an image and eliminate the detected dynamic objects. In 2019, Wang et al. [22] proposed a unified framework for the mutual improvement of SLAM and semantic segmentation. This method employs the FCIS algorithm for initial segmentation, and then culls possible dynamic objects in a bounding box such that visual SLAM is not

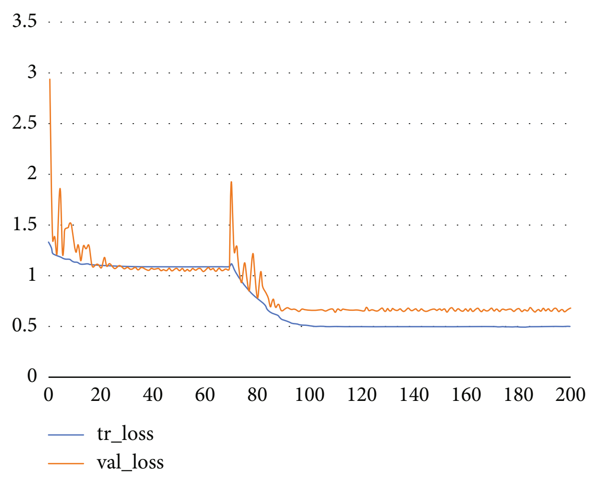


Figure 7: Graph of cost function during training. tr_loss represents the training set loss curve, and val_loss represents the validation set loss curve.

affected by dynamic object interference. However, several limitations are evident in these studies. On the one hand, these methods only target dynamic objects with prior information, and potential dynamic objects without prior information cannot be identified. On the other hand, the target detection and semantic segmentation results can be inaccurate; thus, unreliable feature points can be input into visual SLAM.

2.2. Deep Learning Combined with Geometric Methods in Visual SLAM. Many previous studies have combined deep learning with geometric methods to overcome the limitations of purely using deep learning to assist visual SLAM. For example, in 2018, Yu et al. [23] proposed DS-SLAM, in which ORB-SLAM2 [24] is employed to embed the SegNet deep convolutional neural network combined with a moving consistency check to reduce the impact of dynamic objects. Here, an object is considered to be dynamic only when both SegNet and the moving consistency check methods identify it as a dynamic object. In 2018, Dyna-SLAM was proposed by Bescos et al. [25]. Dyna-SLAM combines deep learning with geometric methods to eliminate unreliable feature points. This method employs the Mask-RCNN algorithm to segment out prior dynamic information in a scene, uses a multiview geometry technique to further detect potential dynamic objects, and removes the detected dynamic objects to improve the accuracy and robustness of visual SLAM. However, the calculation costs of Mask-RCNN are high owing to the large number of parameters, and the multiview geometry technique incurs high calculation costs. As a result, the Dyna-SLAM method cannot be employed on mobile devices. In 2020, Ai et al. [26] proposed DDL-SLAM, in which UNet was used to detect dynamic points, and then the dynamic objects in the scene were eliminated with the help of UNet and multiview geometry.

2.3. Deep Learning Combined with Other Methods in Visual SLAM. In visual SLAM for dynamic environments, deep learning is combined with some other methods. For example, Han and Xi [27] proposed PSPNet-SLAM, which primarily eliminates dynamic points in two steps. First, the optical flow is employed to remove feature points with large optical flow values. Second, PSPNet is employed to eliminate the remaining feature points in the prior objects. In 2019, Xiao et al. [28] proposed dynamic-SLAM to improve the SSD algorithm and improve the detection recall rate, which is then used for dynamic object detection. Second, the selection tracking algorithm is employed to process dynamic object points to further improve the pose estimation accuracy in a dynamic environment. In addition, Cui et al. [29] proposed a tightly coupled SOF-SLAM method, which embeds SegNet as a separate semantic segmentation thread into ORB-SLAM2 to detect dynamic objects, and a semantic optical flow method is implemented to further eliminate the dynamic features. In 2020, Ai et al. [30] proposed visual SLAM for dynamic environments based on object detection. This method combines YOLOV4 and the dynamic object probability (DOP) model to detect dynamic objects. Here, the DOP model is employed to improve the efficiency of object detection and enhance system performance in terms of separating dynamic targets from static scenes. Thus, the visual SLAM system has higher accuracy and robustness in dynamic environments.

2.4. Proposed System. Here, this paper describes the proposed visual SLAM system for an indoor dynamic environment in detail. First, provide an overview of the framework. Then, a lightweight deep convolutional neural network employed in the proposed system for semantic segmentation is described. Finally, the preprocessing stage of tracking is briefly described, including the potential dynamic object detection and background restoration.

2.5. Basic Framework. The most relevant problem in visual SLAM for indoor dynamic environments is interference from movable objects, such as humans and accompanying animals, which results in unsatisfactory positioning and mapping. Thus, this paper proposes a lightweight robust dynamic SLAM system that uses a new semantic segmentation method (LRD-SLAM). Figure 1 shows an overview of the proposed LRD-SLAM system, and the basic framework of the LRD-SLAM system is shown in Figure 2. It includes the following features. First, a separate semantic segmentation thread is implemented. This semantic segmentation thread takes the original RGB image as input to obtain semantic labels for each pixel point. Second, the preprocessing stage of tracking is implemented in the tracking thread to remove interference from dynamic map points such that the visual odometer for visual SLAM performs robustly for camera pose calculation in dynamic environments.

2.6. Semantic Segmentation. This paper designed a fast deep convolutional neural network (FNet) to realize semantic segmentation. The core of this network includes a fast encoder, a decoder corresponding to the encoder, and a pixel classification layer. The network encoder use employs ResNet-50 [31] as the backbone for feature extraction. The network

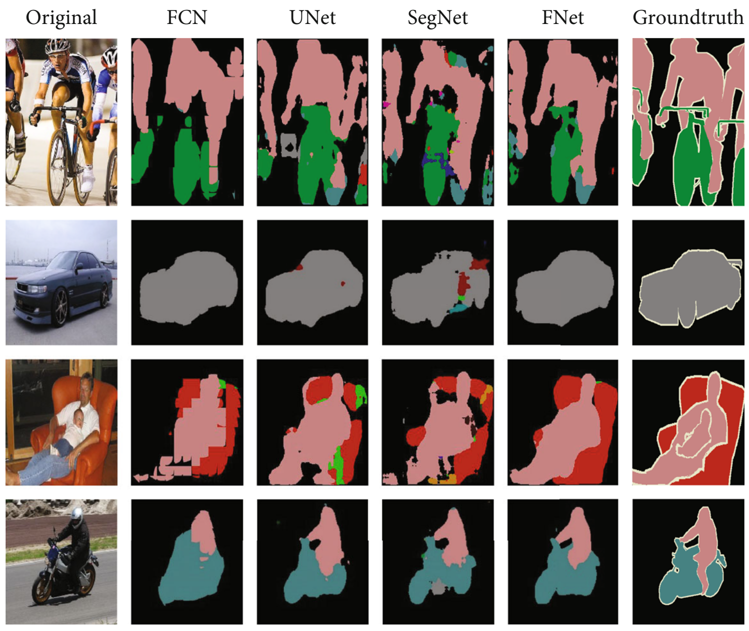


Figure 8: Visualization of VOC2012 datasets.

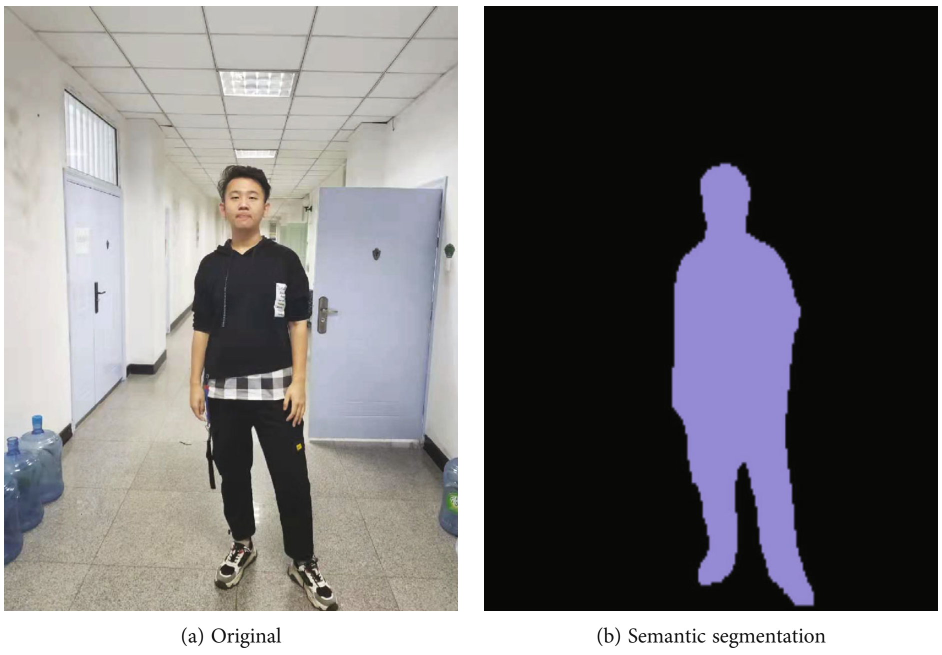


Figure 9: Visualization of real scenes.

head of ResNet-50 comprises a convolutional layer, a BN layer, ReLU, and a max pooling layer, and then the residual blocks, comprising a convolution block and an identity block, are stacked to extract deeper features. First, in the encoder part, all convolution kernels in the backbone network are improved according to the lightweight design technique in GhostNet [32] such that the number is reduced to one-half of the original. Subsequently, the final feature map comprises the initial feature map convolved in sequence and additional features obtained via the cheap operation. Here, the cheap operation performs convolution on the feature map obtained by a small number of convolutions via point convolution, i.e., a linear operation. It overcomes the disadvantages of ResNet-50 in terms of the large number of parameters, improves calculation efficiency, and ensures sufficient. Here, if the size of the image is $h \times w \times c$, the sizes of the convolution kernel and initial feature maps are $k \times k$ and $h' \times w'$, respectively. The number of initial feature maps

Table 2: Comparison of experimental results obtained on VOC2012 dataset.

Method	Experimental results									
	FCN-VGG16		UNet		SegNet		Deeplabv3—MobileNetv1		FNet	
mIoU	PA	IOU	PA	IOU	PA	IOU	PA	IOU	PA	IOU
Background	97.03	87.87	97.9	86.04	97.81	86.23	97.4	89.81	97.97	87.94
Aeroplane	61.79	44.78	59.27	40.6	61.04	41.01	69.48	50.15	61.85	43.39
Bicycle	55.77	18.04	48.49	16.13	40.34	14.45	59.59	19.44	30.42	12.36
Bird	63.96	55.49	56.58	49.37	58.9	51.09	62.58	60.08	63.99	56.05
Boat	52.51	40.1	58.6	43.69	54.52	39.48	69.39	52.04	65.38	46.53
Bottle	64.66	58.03	61.23	56.36	63.57	58.43	76.62	66.42	69.73	64.81
Bus	70.7	62.22	49.27	47.86	34.84	33.87	78.84	76.58	65.62	63.31
Car	79.64	70.81	67.29	62.87	68.22	63.28	82.6	75.46	77.37	72.97
Cat	76.88	73.19	61.87	58.84	62.61	59.85	76.48	74.87	77.2	73.57
Chair	36.78	30.12	31.19	26.28	40.7	35.02	47.08	40.16	44.92	37.74
Cow	64.88	60.16	42.39	39.76	55.71	52.72	79.08	73.23	49.92	48.22
Dining table	31.66	27.83	24.27	23.61	28.36	27.52	42.73	41.32	32.98	31.95
Dog	67.61	65.3	60.89	55.19	65.8	61.59	77.0	73.85	70.77	66.4
Horse	56.29	51.78	48.98	45.18	56.09	52.52	69.06	64.62	57.92	54.83
Motorbike	72.7	63.98	51.41	48.33	56.15	52.18	70.54	65.18	68.01	61.69
Person	68.68	62.76	63.74	59.29	63.5	60.01	70.21	64.46	70.61	66.16
Potted plant	58.67	44.4	50.96	41.56	57.76	46.04	75.85	54.17	64.59	51.61
Sheep	65.67	60.83	50.85	48.57	64.96	60.14	75.66	71.17	66.22	62.21
Sofa	44.68	41.36	32.32	31.46	44.64	43.15	58.61	55.67	20.02	19.72
Train	61.18	58.53	57.15	55.02	55.24	54.01	77.71	74.98	63.38	61.67
MPA	62.58	52.02	53.73	46.80	56.53	49.62	70.83	62.18	60.94	54.16
Inference time	102 ms		58.6 ms		55.2 ms		69.4 ms		**46.3** ms	

Table 3: The performance of the proposed FNet with different s on VOC2012.

s	Parameters	Accuracy (%)
2	12.2 M	85.3%
3	10.8 M	84.9%
4	10.1 M	84.4%
5	9.7 M	83.8%

is denoted as m. Thus, the number of parameters in the improved convolution is expressed as follows.

$$\text{params}_1 = m \cdot c \cdot k \cdot k. \tag{1}$$

The floating point operations (FLOPs) are calculated as follows.

$$\text{flops}_1 = h' \cdot w' \cdot m \cdot c \cdot k \cdot k. \tag{2}$$

Assume that the size of the convolution kernel in the cheap operation is $d \times d$, and the number of cheap operation transformations is s. Thus, the number of feature maps is expressed as follows.

$$n = m \cdot s. \tag{3}$$

Owing to the existence of identity transformation, the actual number of effective transformations is $s - 1$; thus, the number of parameters in the cheap operation transformation is obtained as follows.

$$\text{params}_2 = m \cdot (s-1) \cdot d \cdot d.. \tag{4}$$

The corresponding number of FLOPs is calculated as follows.

$$\text{flops}_2 = (s-1) \cdot h' \cdot w' \cdot m \cdot c \cdot d \cdot d.. \tag{5}$$

Thus, the total number of parameters in the network is the sum of Equation (1) and Equation (5). By substituting Equation (3) into this sum, we obtain the following.

$$\text{params}_{sum} = \frac{n}{s} \cdot c \cdot k \cdot k + \frac{n(s-1)}{s} \cdot d \cdot d.. \tag{6}$$

Similarly, the total number of FLOPs is the sum of Equations (2) and (6), and by substituting Equation (3) into this sum, we obtain the following.

$$\text{flops}_{sum} = \frac{n}{s} \cdot h' \cdot w' \cdot c \cdot k \cdot k + \frac{n(s-1)}{s} \cdot h' \cdot w' \cdot d \cdot d.. \tag{7}$$

The number of parameters and FLOPs in the original ResNet-50 convolution is expressed as follows.

$$\text{params}_{ori} = n \cdot c \cdot k \cdot k. \tag{8}$$

$$\text{flops}_{ori} = n \cdot c \cdot h' \cdot w' \cdot k \cdot k. \tag{9}$$

Note that these take very large values because the number of filters n and channel number c are typically very large. The number of parameters $\text{params}_{\text{sum}}$ can be reduced using lightweight network techniques, which make the network more suitable for deployment on mobile devices. The theoretical acceleration ratio of the original ResNet-50 convolution upgraded to lightweight convolution is given as follows.

$$r_s = \frac{\text{flops}_{\text{ori}}}{\text{flops}_1 + \text{flops}_2} = \frac{n \cdot h' \cdot w' \cdot c \cdot k \cdot k}{(n/s) \cdot h' \cdot w' \cdot c \cdot k \cdot k + (n(s-1)/s) \cdot h' \cdot w' \cdot d \cdot d} \approx s, \tag{10}$$

where $d \times d$ is similar to $k \times k$, and $s \ll c$. Similarly, the compression ratio of parameters is expressed as follows.

$$r_c = \frac{\text{params}_{\text{ori}}}{\text{params}_1 + \text{params}_2} = \frac{n \cdot c \cdot k \cdot k}{(n/s) \cdot c \cdot k \cdot k + (n(s-1)/s) \cdot d \cdot d} \approx s. \tag{11}$$

Note that this is approximately equal to the theoretical acceleration ratio.

In the decoder, FNet combines SegNet and UNet and designs the upsampling of the low-resolution feature maps by indexing and concatenating. Thus, the pooling indices, i.e., the index to the maximum value, selected via maximum pooling are saved after each feature extraction process in the encoder layer. The obtained indices are used for upsampling in the decoder layer. Simultaneously, a concatenation feature fusion method is employed to map each layer feature in the encoder to the corresponding layer in the decoder to enrich the texture deepening feature. Note that the upsampled feature maps are sparse; thus, a trainable deconvolution operation is performed on the feature maps to generate dense feature maps. The principle of the decoder is illustrated in Figure 3. Then, these feature maps are sent to the SoftMax classifier. An overall schematic diagram of FNet is shown in Figure 4. The network is trained on the Pascal VOC2012 [33] dataset to segment the movable classes in real life (e.g., people, birds, horses, sheep, cats, cows, and dogs).

2.7. The Preprocessing Stage of Tracking. After the ORB feature points are extracted in the tracking thread, a preprocessing stage of tracking is implemented to process the dynamic objects in the current environment. This stage comprises three main steps, i.e., dynamic object detection, dynamic object elimination, and background repair. The semantic segmentation thread detects dynamic objects and covers them with a binary mask. Using multiview geometry, potential dynamic objects are further detected and eliminated together with the ORB feature points under the mask. Finally, only reliable ORB feature points are retained. The dynamic object culling strategy is shown in Algorithm 1. After removing the dynamic objects, the blocked part is missing, which affects subsequent reconstruction of the point cloud map. Here, LRD-SLAM employs a background repair method to solve the above problems.

Table 4: Absolute pose error (APE $[m]$) results for different variants.

Sequence	ORB-SLAM2 (RGB-D)				LRD-SLAM (P)				LRD-SLAM (C)			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
w_half	0.6282	0.5479	0.5136	0.3073	0.0201	0.0175	0.0156	0.0100	0.0194	0.0170	0.0154	0.0094
w_xyz	0.6650	0.5617	0.4855	0.3558	0.0159	0.0135	0.0119	0.0084	0.0141	0.0116	0.0100	0.0079
w_rpy	1.0360	0.9162	0.7605	0.4836	0.0409	0.0326	0.0261	0.0247	0.0379	0.0299	0.0242	0.0232
w_static	0.3879	0.3540	0.3202	0.1586	0.0068	0.0061	0.0057	0.0029	0.0074	0.0066	0.0061	0.0034
s_half	0.0236	0.0200	0.0185	0.0125	0.0227	0.0194	0.0165	0.0121	0.0173	0.0155	0.0142	0.0078

Although most moving objects can be detected by FNet, some dynamic objects cannot be recognized. Such undetectable dynamic objects are movable although they are not in a predefined category, e.g., cups, books, and telephones on a table are all potential moving objects. Thus, LRD-SLAM applies a multiview geometry technique to the system to further improve the accuracy of dynamic object detection. In this technique, the distance and rotation between the new frame and each keyframes are used to select keyframe that overlap highly with the new frame. For these keyframes, the projection of each key point x to the current frame is calculated, and the projection point x' and depth z_{proj} are obtained. Then parallax angle α is obtained in consideration of the connection of each key point x, projection point x' and their corresponding 3D point X. If $\alpha > 30\circ$, the point is considered a dynamic object and cannot be used for tracking and mapping. The difference between the projected point's depth z_{proj} and the current frame's depth z' is also considered. If this difference is too large, the key point is also considered a dynamic object. This method is illustrated in Figure 5.

The absence of a background by removing a dynamic object will affect the subsequent establishment of the point cloud map. Thus, in the proposed method, the previous 10 keyframes before the current frame are selected and projected in sequence to the current frame such that the occluded part of the current frame is recovered using the original static background information. The basic principle of this background restoration process is illustrated in Figure 6.

3. Experimental Results

The proposed method was evaluated experimentally on the large open-source TUM RGB-D dataset. This data set contains real-time ground data, RGB images, and corresponding depth images. This paper took a sequence of the dynamic environment in this dataset, including halfsphere, static, xyz, and rpy.

First, this paper executed ORB-SLAM2 and LRD-SLAM on the above sequences, and the absolute pose error (APE) and relative pose error (RPE) were evaluated to verify the improvement realized by the proposed method. Second, the proposed method was compared with several state-of-the-art visual SLAM systems for dynamic environments. Finally, the number of parameters in other advanced deep convolutional neural networks for semantic segmentation was compared to that in FNet to verify the real-time performance of the proposed system. These experiments were conducted using the Ubuntu18.04 operating system, where the deep convolutional neural network was implemented using Python 3.6, and the deep learning frameworks were Keras and Tensorflow. This paper used C++ to call Python to integrate the deep convolutional neural network into the visual SLAM thread. All experiments were conducted on a computer with Intel (R) Core (TM) i7-10750H 2.60GHz CPU, an NVIDIA GeForce RTX 2060 GPU, and 6GB memory.

3.1. Comparison and Analysis of Experimental Results

3.1.1. Semantic Segmentation. The Pascal VOC2012 dataset and its extended semantic boundary datasets and benchmark (SBD) datasets in the semantic segmentation experiments. The SBD datasets include all images and labels from VOC2012, containing a total of 11,335 images with semantic segmentation labels.

The training parameters are shown in Table 1, and the cost function for training is shown in Figure 7. During the training process, the SBD dataset was used as the training set, and 10% of the SBD dataset was randomly selected as the validation set for training. Note that FNet employs the frozen training method for training. First, the encoder part was frozen at the beginning of training, and only the decoder part was trained and converged 70 times. Then, the encoder part was unfrozen and trained together with the decoder part. The learning rate during training is attenuated if the loss does not decrease for three times.

Under the same hardware conditions, semantic segmentation experiments were conducted for four methods on the Pascal VOC2012 validation set. The experimental results were compared and calculated the corresponding evaluation indicators. The visualization results are shown in Figure 8 FNet exhibits a better segmentation effect compared to the other methods.

FNet also performed semantic segmentation experiments on real-scene environments other than the VOC2012 dataset, and the results are shown in Figure 9. As shown, FNet also exhibits good effects in real scenes.

As shown in Table 2, the average pixel accuracy (MPA) and mean intersection over union (mIoU) of FNet are higher than those of UNet and SegNet and similar to those of FCN [34]. In terms of processing time, FNet is the fastest in terms of processing time per frame, compared to the four semantic segmentation network models FNet, UNet, SegNet, and Deeplabv3+, which improve the processing time per

Table 5: Results of metrics absolute pose error (APE [m]).

Sequence	ORB-SLAM2 (RGB-D)				LRD-SLAM (C)				Improvement			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
w_half	0.6282	0.5479	0.5136	0.3073	0.0194	0.0170	0.0154	0.0094	96.91%	96.90%	97%	96.95%
w_xyz	0.6650	0.5617	0.4855	0.3558	0.0141	0.0116	0.0100	0.0079	97.88%	97.93%	98%	97.77%
w_rpy	1.0360	0.9162	0.7605	0.4836	0.0379	0.0299	0.0242	0.0232	96.34%	96.73%	97%	95.20%
w_static	0.3879	0.3540	0.3202	0.1586	0.0074	0.0066	0.0061	0.0034	98.10%	98.15%	98%	97.87%
s_half	0.0236	0.0200	0.0185	0.0125	0.0173	0.0155	0.0142	0.0078	26.50%	22.65%	23%	37.49%

Table 6: Results of metrics translational drift (RPE [m/s]).

Sequence	ORB-SLAM2 (RGB-D)				LRD-SLAM (C)				Improvement			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
w_half	0.5099	0.4095	0.3838	0.3038	0.0315	0.0287	0.0289	0.0130	93.82%	92.99%	92%	95.73%
w_xyz	0.4515	0.3704	0.3550	0.2581	0.0523	0.0462	0.0354	0.0244	88.42%	87.52%	90%	90.55%
w_rpy	0.4915	0.3823	0.3078	0.3089	0.0553	0.0516	0.0510	0.0199	88.75%	86.50%	83%	93.56%
w_static	0.4145	0.3044	0.1868	0.2813	0.0112	0.0109	0.0111	0.0027	97.29%	96.42%	94%	99.05%
s_half	0.0287	0.0257	0.0249	0.0127	0.0247	0.0230	0.0237	0.0090	13.85%	10.43%	5%	29.53%

Table 7: Results of metrics rotational drift (RPE [deg/s]).

Sequence	ORB-SLAM2 (RGB-D)				LRD-SLAM (C)				Improvement			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
w_half	10.6200	8.4826	7.6227	6.3899	0.8592	0.7831	0.6680	0.3534	91.91%	90.77%	91%	94.47%
w_xyz	8.3940	6.9392	6.9721	4.7230	0.7849	0.7420	0.7236	0.2559	90.65%	89.31%	90%	94.58%
w_rpy	9.3985	7.3989	6.0966	5.7955	1.2416	1.0972	0.9473	0.5811	86.79%	85.17%	84%	89.97%
w_static	7.3653	5.4297	3.3028	4.9765	0.2021	0.1971	0.1793	0.0448	97.26%	96.37%	95%	99.10%
s_half	1.0089	0.9640	1.1497	0.2976	0.6990	0.6621	0.7130	0.2240	30.72%	31.32%	38%	24.72%

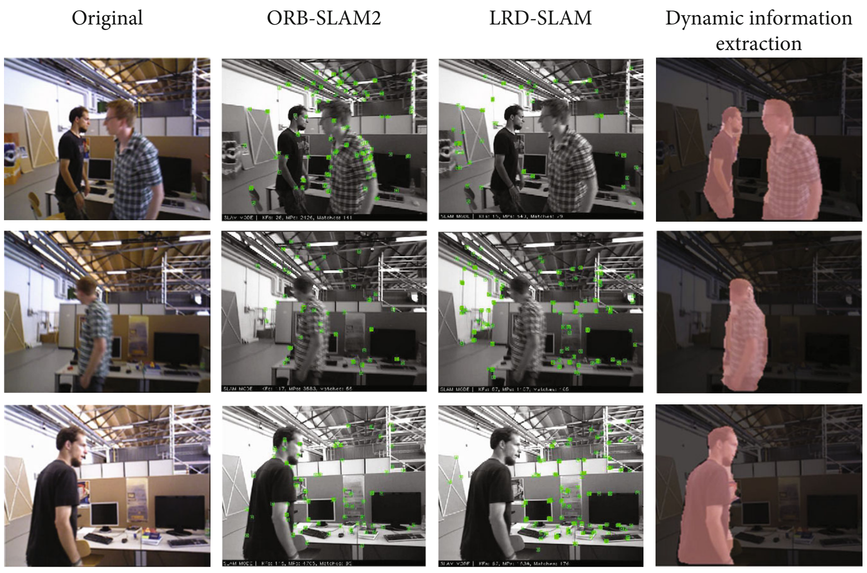
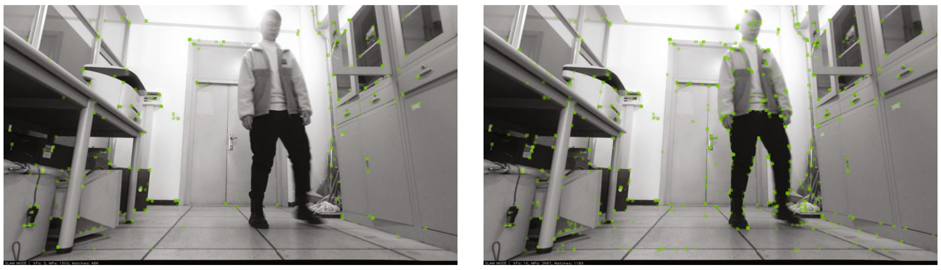


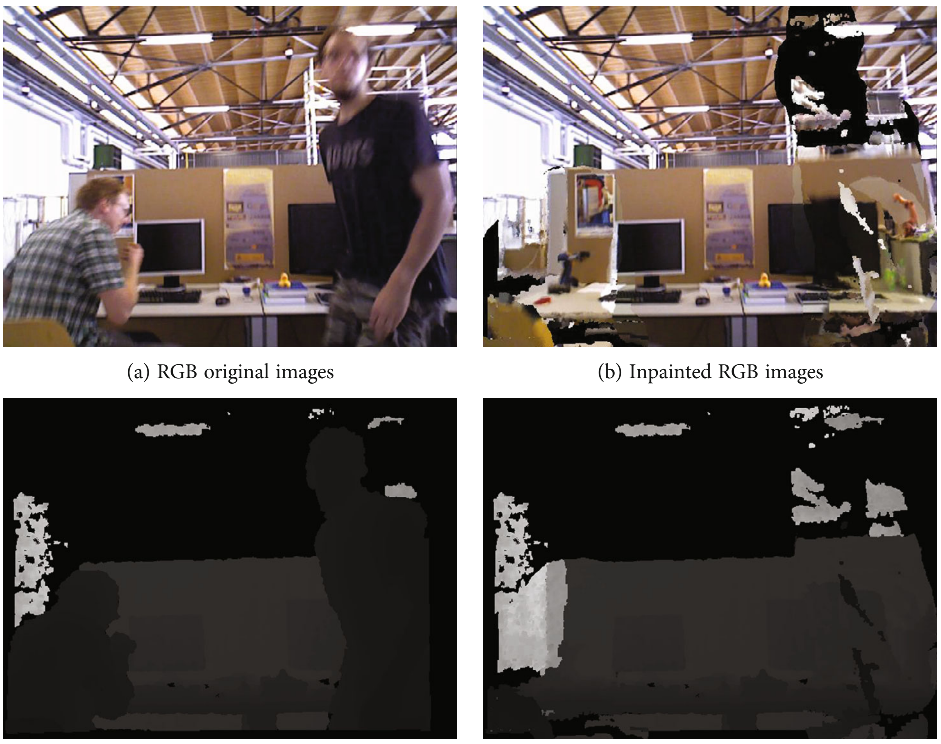
Figure 10: Comparative effects of semantic segmentation and feature extraction in the f_w_halfsphere, f_w_xyz, and f_w_rpy sequences. The first column shows the original image frame in the dataset, and the second, third, and fourth columns show renderings of the ORB-SLAM2 feature extraction, LRD-SLAM feature extraction, and FNet semantic segmentation rendering.



(a) LRD-SLAM feature extraction effect diagram

(b) ORB-SLAM2 feature extraction effect diagram

Figure 11: Comparison of feature extraction effects in real scenes.



(a) RGB original images

(b) Inpainted RGB images

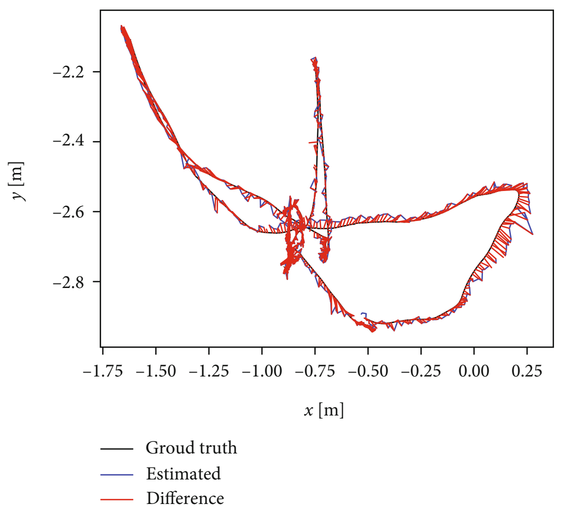
(c) Depth original images

(d) Inpainted depth images

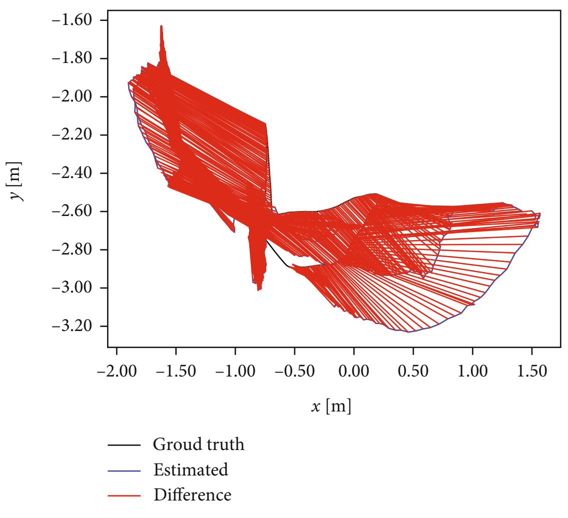
Figure 12: Composite images for background restoration: (a, b) RGB original and recovered images, respectively; (c, d) corresponding depth images of (a) and (b), respectively.

frame by 54%, 20%, 15.2%, and 33.3%, respectively, due to the lightweight technology used in FNet, which greatly reduces the amount of computation in the encoder. The network architecture is relatively simple. In terms of accuracy, it is not much different from the prediction accuracy of some dynamic objects in Deeplabv3+, which verifies that the method in this paper greatly improves the real-time performance of the network model while ensuring accuracy. Therefore, FNet shows better performance compared to other methods and is more suitable for embedded visualization SLAM tasks. In the table, we can find that FNet has better prediction accuracy for objects with moving characteristics, such as people, cats, dogs, and cars, while the prediction accuracy for sofa class is less satisfactory, which is mainly due to two reasons. One is that the number of training sets of the sofa in the SBD extended dataset used in this paper is low in the extended dataset, and the other is that there are fewer images with the label of the sofa in the test set used in this paper, and most of them are similar to the style of chair, which causes FNet to confuse chair and sofa. Therefore, it is expected that the prediction accuracy of sofa class is low, and this paper targets dynamic objects in an indoor environment. The performance effect of sofa class will not affect the accuracy of embedding into visual SLAM in dynamic environment.

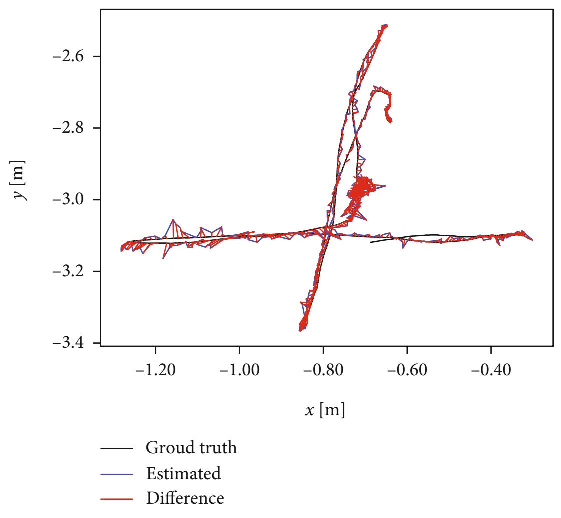
For the parametric analysis of the network model, the FNet designed in this paper has a hyperparameter in the encoder part, i.e., the number of cheap operation transitions s. In Table 3, the effect of the number of cheap operation transitions s of the FNet on the number and accuracy of the network parameters is tested. First, I fix the size of the convolutional kernel d to 3 while adjusting the number of



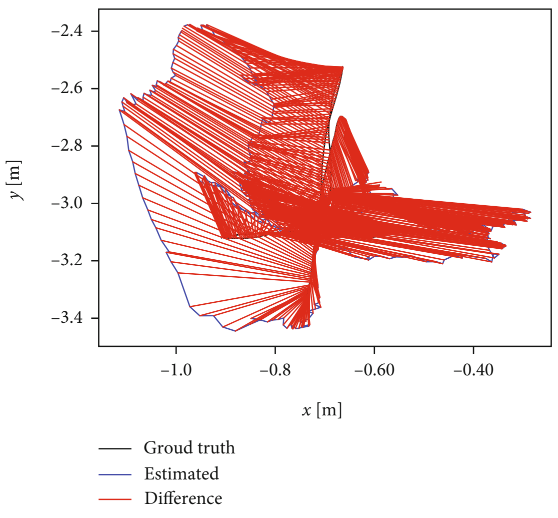
(a) walking_halfsphere with LRD-SLAM



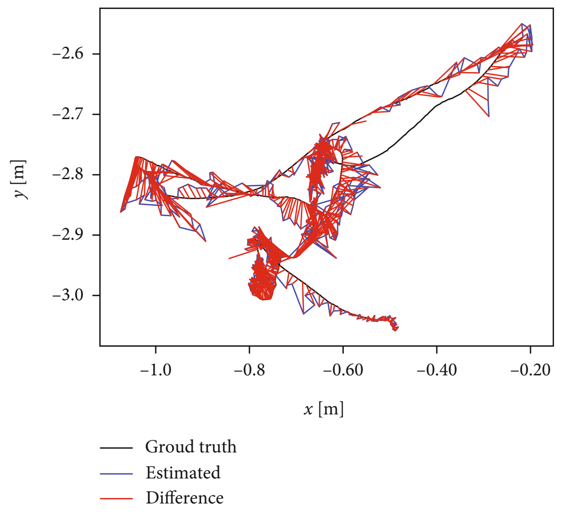
(b) walking_halfsphere with ORB-SLAM2



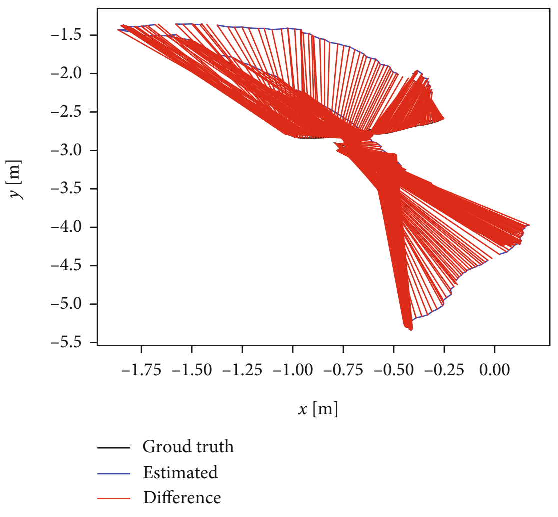
(c) walking_xyz with LRD-SLAM



(d) walking_xyz with ORB-SLAM2

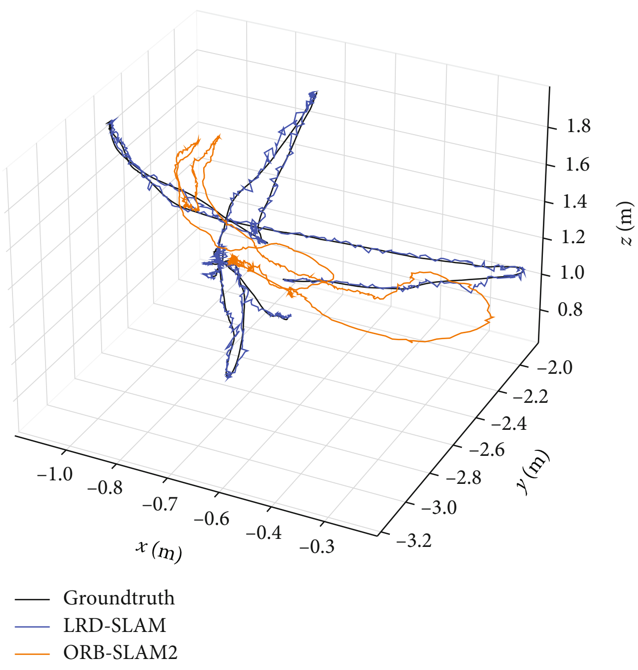


(e) walking_rpy with LRD-SLAM

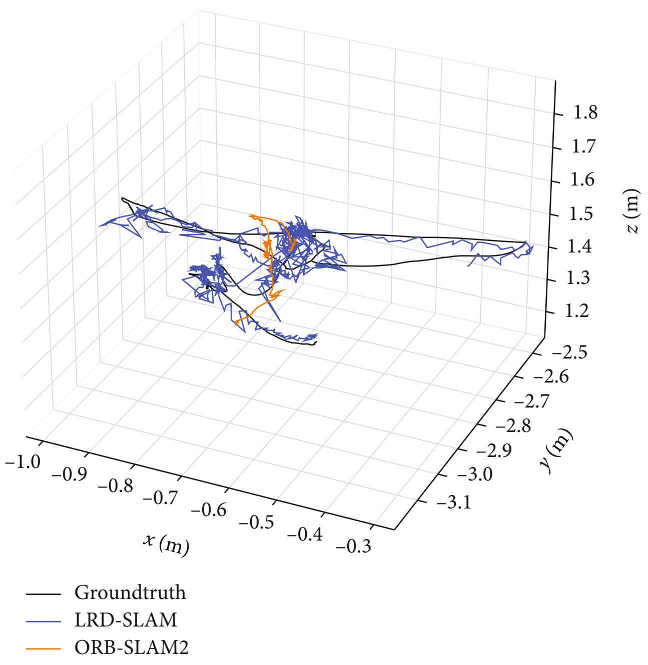


(f) walking_rpy with ORB-SLAM2

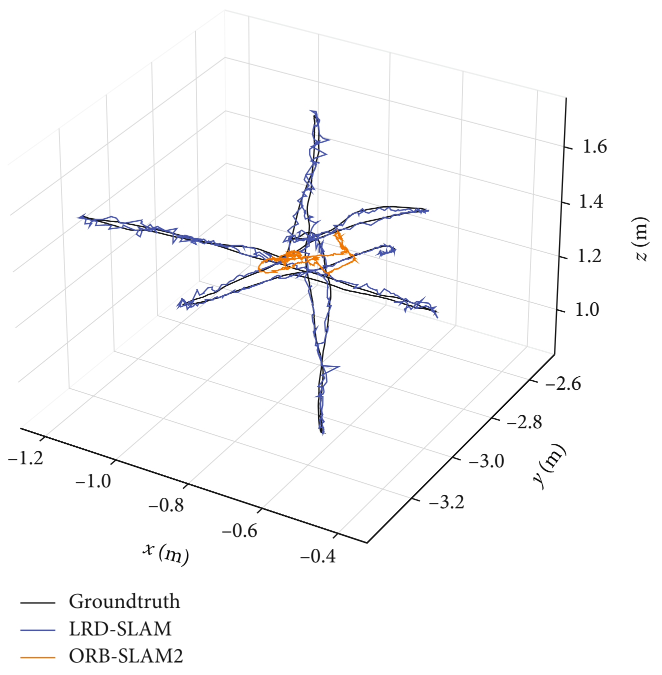
Figure 13: In the feiburg3_walking_halfsphere, feiburg3_walking_xyz, and feiburg3_walking_rpy sequences, ORB-SLAM2 and the trajectory error map of LRD-SLAM. The black line is the original trajectory of the camera motion, the blue line is the camera motion trajectory estimated by ORB-SLAM2 or LRD-SLAM, and the red line is the error between the estimated motion trajectory and the original trajectory.



(a) feiburg3_walking_halfsphere



(b) feiburg3_walking_rpy

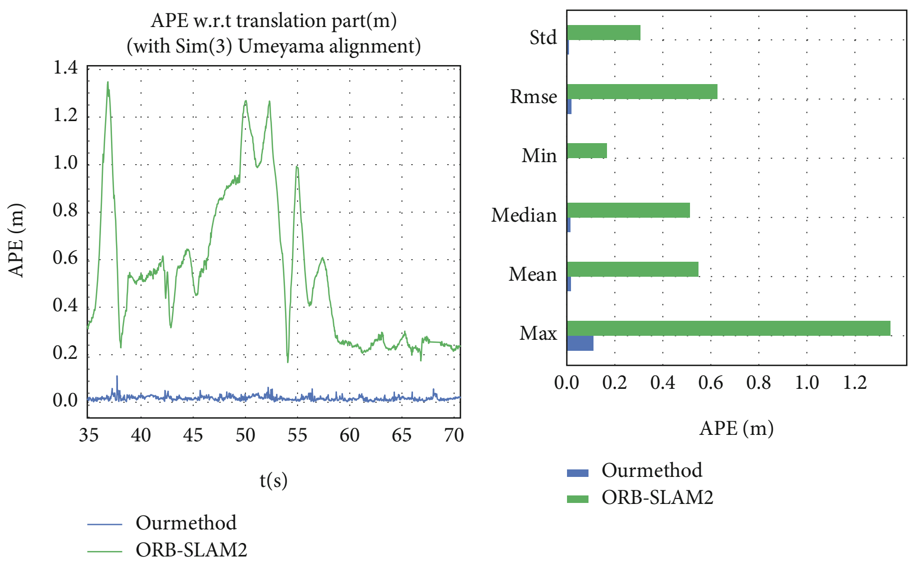


(c) feiburg3_walking_xyz

Figure 14: 3D trajectory error plots of ORB-SLAM2 and LRD-SLAM in feiburg3_walking_halfsphere, feiburg3_walking_rpy, and feiburg3_walking_xyz dynamic dataset sequences.

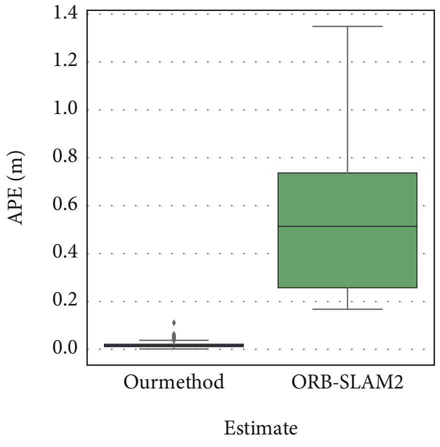
parameters s in the range {2, 3, 4, and 5} and present the results on the VOC2012 validation set in Table 3. Second, since the number of s is directly related to the computational cost of the final network, i.e., the larger s is, the larger the compression and acceleration ratios are, and it is expected that the number of parameters of the network decreases significantly and the accuracy decreases gradually as s increases. Finally, when $s = 2$, this means that the backbone network ResNet-50, the encoder part of FNet, is compressed by a factor of 2 while having a better accuracy, demonstrating the superiority of FNet.

3.1.2. Analysis of LRD-SLAM Experimental Results. The absolute trajectory error results of LRD-SLAM with different variants in five dynamic sequences are shown in Table 4, where LRD-SLAM (P) represents only FNet and ORB-SLAM2, and LRD-SLAM (C) represents the complete system.



(a) APE comparison curve

(b) APE statistics comparison chart



(c) APE box line comparison chart

Figure 15: APE results for the feiburg3_walking_halfsphere sequence: (a) APE of ORB-SLAM2 and the proposed method, (b) APE statistical data comparison, and (c) APE box line comparison.

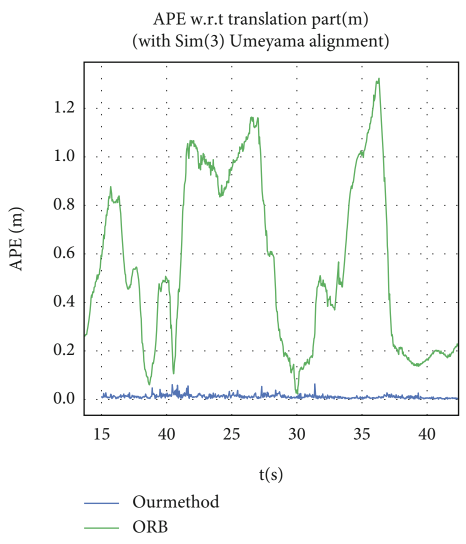
As shown in Table 4, LRD-SLAM (*C*) has the best RMSE value. LRD-SLAM (*P*) is slightly less accurate than compared to ORB-SLAM2. LRD-SLAM (*P*) masks the multiview geometry method; thus, it greatly outperforms LRD-SLAM (*C*) in terms of calculating time. Then, the APE between LRD-SLAM and ORB-SLAM2 in the five TUM datasets is listed in Table 5.

As can be seen, LRD-SLAM has a lower RMSE value than ORB-SLAM2 in the five dynamic environment sequences, and consequently the median, mean, S.D., and RMSE are all significantly reduced. The relative accuracy improvement rate of LRD-SLAM for ORB-SLAM2 was also calculated. In a low dynamic environment freiburg3_sitting_halfsphere, the improvement rate of RMSE reached 26%. In the other four sequences of highly dynamic environments, the improvement rate was greater than 90%. These results demonstrate that the LRD-SLAM is significantly better in terms of accuracy, robustness, and stability in high dynamic scenarios. The reference formula for the improvement rate is given as follows.

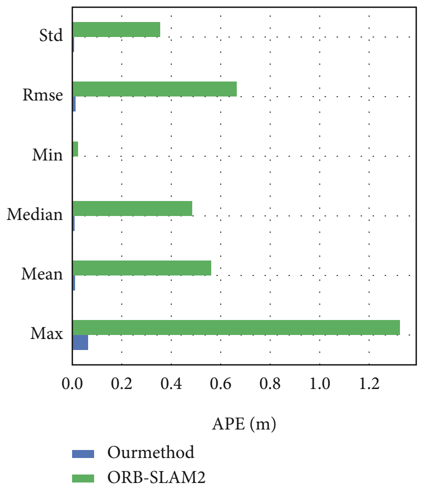
$$\eta = \frac{o - r}{o} \times 100\%. \qquad (12)$$

Here, η is the improvement rate, and o and r are the corresponding values for ORB-SLAM2 and LRD-SLAM, respectively. Similarly, in terms of the RPE, as shown in Tables 6 and 7, LRD-SLAM demonstrates strong accuracy, robustness, and stability compared to ORB-SLAM2. In addition, we found that LRD-SLAM can be used in most high dynamic environments.

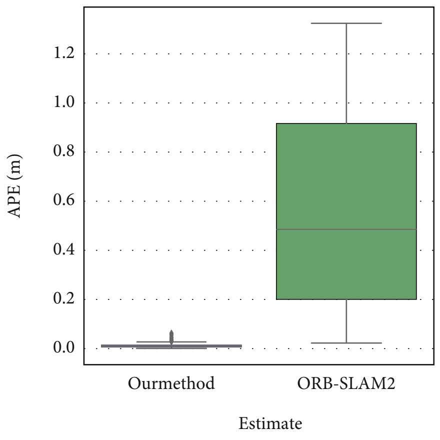
The RGB images in the f_w_halfsphere, f_w_xyz, and f_w_rpy sequences were selected randomly to compare the feature extraction and semantic segmentation performance of both LRD-SLAM with ORB-SLAM2, and the results are shown in Figure 10.



(a) APE comparison curve



(b) APE statistics comparison chart



(c) APE box line comparison chart

Figure 16: APE results for the feiburg3_walking_xyz sequence: (a) APE of ORB-SLAM2 and the proposed method, (b) APE statistical data comparison, and (c) APE box line comparison.

As can be seen, LRD-SLAM effectively segmented the dynamic objects in the video sequences, and the feature points of the dynamic objects were eliminated effectively during feature extraction. In the real scene, the feature extraction effects of LRD-SLAM and ORB-SLAM2 are shown in Figure 11.

As can be seen, the feature points of moving objects were effectively eliminated in LRD-SLAM. For background restoration, Figure 12 shows an image synthesized from a partial sequence of input frames in the TUM RGB-D [6] datasets. It can be seen that most of the parts occluded by the moving objects have been correctly patched, which was beneficial to the subsequent point cloud map construction.

Figure 13 shows the trajectory error curves for a highly dynamic environment. Here, Figures 13(a), 13(c), and 13(e) show the errors between LRD-SLAM and the actual trajectory; Figures 13(b), 13(d), and 13(f) show the errors between ORB-SLAM2 and the actual trajectory.

The comparison in the 3D trajectory error is shown in Figure 14. The comparison between LRD-SLAM and ORB-SLAM2 in the sequence of feiburg3_walking_halfsphere dynamic datasets is shown in Figure 14(a), where the blue line is the trajectory of LRD-SLAM, the black line is the real motion trajectory, and the orange line is the trajectory of ORB-SLAM2. In the figure, it can be found that the trajectory of ORB-SLAM2 in the dynamic environment at each time step has a large error with the real trajectory, while the trajectory of LRD-SLAM at each time step matches the real trajectory, which is because LRD-SLAM adopts the FNet designed in this paper to first semantically segment the dynamic objects in each image frame before tracking, and then the tracking threads added in the dynamic objects are then removed from the preprocessing thread added in the tracking thread, so that the image frames without dynamic object interference are input to the whole vision SLAM for bit pose estimation. Similarly, for the 3D trajectory comparison between LRD-SLAM and ORB-SLAM2 in the other two dynamic data sets sequences feiburg3_walking rpy and feiburg3_walking_ xyz are shown in Figure 14(b) and Figure 14(c), respectively, it can be found that LRD-SLAM has better localization accuracy and robustness compared to ORB-SLAM2 in the dynamic environment.

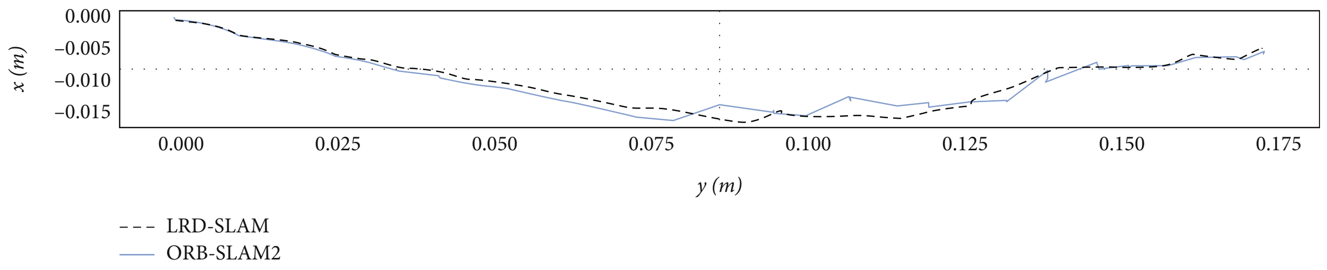


Figure 17: Qualitative comparison of estimated camera trajectory between ORB-SLAM2 and LRD-SLAM.

Table 8: Comparison of relative RMSE [m] for proposed and existing systems on dynamic sequences of TUM dataset.

Sequence	walking_halfsphere	walking_xyz	walking_rpy	walking_static	sitting_halfsphere
PSPNet-SLAM	94.33%	98.05%	95.98%	97.87%	—
Detect-SLAM	90.72%	97.62%	66.94%	—	-29.8%
Zhao et al.	93.90%	95.10%	80.80%	72.00%	—
DS-SLAM	93.76%	96.71%	46.97%	97.91%	
Dyna-SLAM	92.88%	96.73%	94.71%	93.33%	15%
LRD-SLAM (C)	96.91%	97.88%	96.34%	98.10%	26.50%

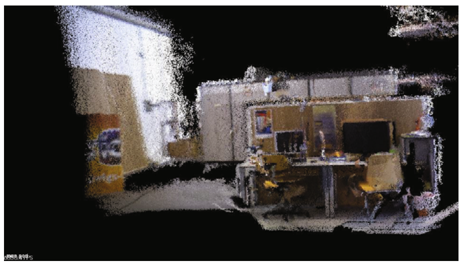


Figure 18: Point cloud map reconstruction.

As can be seen, LRD-SLAM significantly reduced the trajectory error and improved the accuracy of positioning. ORB-SLAM2 demonstrates large errors in highly dynamic environments and is not suitable for positioning and mapping in highly dynamic environments. Figures 15 and 16 compare the APE of ORB-SLAM2 and LRD-SLAM for two highly dynamic sequences feiburg3_walking_halfsphere and feiburg3_walking_xyz, respectively. The results demonstrate that the LRD-SLAM system is more robust and stable than ORB-SLAM2 under highly dynamic conditions.

In a complex real scene, the mobile robot platform was used to carry the Kinect2 camera to drive in a straight line. At the same time, the two methods of LRD-SLAM and ORB-SLAM2 are compared and tested. The experimental tracking trajectory comparison effect is shown in Figure 17 Because the mobile robot drives in a straight line, the tracking trajectory of LRD-SLAM is smoother, and more stable than ORB-SLAM2, which verifies that LRD-SLAM also has better accuracy and robustness in real scenarios.

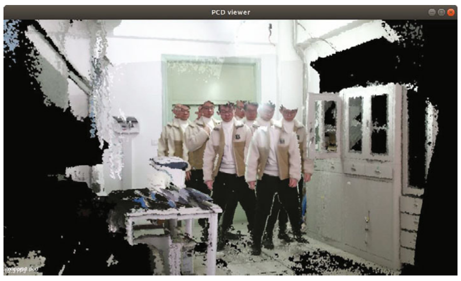
In addition, LRD-SLAM was compared to recent state-of-the-art semantic SLAM systems: PSPNet-SLAM, system in [35], DS-SLAM, and Dyna-SLAM. These systems are based on ORB-SLAM2 and were tested in the dynamic environment sequence of TUM. The improvement rate of RMSE of these systems relative to ORB-SLAM2 was used as the evaluation index. The results are shown in Table 8. As can be seen, these semantic vision SLAM systems eliminated moving objects and realized higher accuracy in the five dynamic sequences. The data in Table 4 were compared to several other methods. We found that LRD-SLAM has higher accuracy in the feiburg3_walking_halfsphere, feiburg3_walking_static, feiburg3_walking_xyz, and feiburg3_walking_rpy sequences. By comparing LRD-SLAM to DS-SLAM and another existing system [35], we found that the improvements are more obvious because multiview geometry is employed to further eliminate potential dynamic objects, and FNet also exhibits more accurate detection of dynamic objects. With the feiburg3_walking_xyz sequence, LRD-SLAM is very close to the effect of PSPNet-SLAM.

Figure 18 shows the final point cloud map. As can be seen, most of the dynamic objects were eliminated, the static background was repaired, and a complete point cloud map was generated. In the real scene, the dense point cloud map construction effect is shown in Figure 19. In the mapping effect of LRD-SLAM, most dynamic objects are culled, and a clean and accurate point cloud map is constructed.

3.2. System Real-Time Analysis. The time costs of the implemented semantic segmentation thread and multiview geometry technique are important factors in an online visual SLAM system because they indirectly reflect the calculation costs and GPU memory requirements.

Table 9 compares the number of parameters of FNet and other deep convolutional neural networks. As the lightweight LRD-SLAM incurs lower computational costs than the other methods, it is more suitable for mobile devices.

For the entire visual SLAM system, run LRD-SLAM (C), LRD-SLAM (P), and Dyna-SLAM on four high-dynamic


(a) Point cloud map constructed by ORB-SLAM2

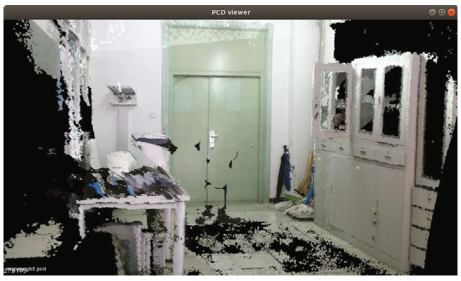

(b) Point cloud map constructed by LRD-SLAM

Figure 19: Construction effect of dense point cloud map in real scene.

Table 9: Comparison of the number of parameters in the proposed and existing semantic segmentation techniques.

	SegNet	BiseNet (Res18) [36]	FCN-VGG16	RefinenNet-101 [37]	FNet
Parameters	29.5 M	49 M	134 M	118 M	12.2 M

Table 10: Comparison of mean tracking time.

Sequence	Dyna-SLAM (s)	LRD-SLAM (P) (s)	LRD-SLAM (C) (s)
walking_ halfsphere	0.813809	0.128086	0.579417
walking_rpy	0.690008	0.113812	0.456743
walking_static	0.856739	0.210931	0.658092
walking_xyz	0.84719	0.194507	0.607722

video sequences under the same hardware environment, and the average tracking time of these systems was calculated in real time. As shown in Table 10, LRD-SLAM (C) consumed less time than Dyna-SLAM on the high dynamic video sequences. The average tracking time of LRD-SLAM (C) was less and the real-time performance was better.

4. Conclusions

This paper has proposed a lightweight visual SLAM system for dynamic environments. FNet was designed and integrated into the proposed visual SLAM system. In this system, a dynamic target detected by FNet is combined with a multiview geometry method, the feature points of the dynamic target are removed, and the static background missing because of the removal of dynamic points is recovered. Experiments conducted using the TUM RGB-D dataset have demonstrated that LRD-SLAM is significantly better than ORB-SLAM2 in indoor dynamic environments, and it obtains higher accuracy than the existing state-of-the-art visual SLAM systems for dynamic environments. FNet is also compared with the model parameters of existing state-of-the-art deep convolutional neural networks available. The results demonstrate that LRD-SLAM is lightweight and can run on mobile devices.

We found that the multiview geometry methods contribute a large amount to the computational costs, which leads to the slow operation of the visual SLAM system. Thus, in the future, the multiview geometry method should be improved to reduce the time costs and improve the real-time performance of the proposed system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there is no conflict of interest regarding the publication of this paper.

References

[1] L. Ruotsalainen, S. Gröhn, M. Kirkko-Jaakkola, L. Chen, R. Guinness, and H. Kuusniemi, “Monocular visual SLAM for tactical situational awareness,” in *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–9, Banff, AB, Canada, 2015.

[2] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, “Visual simultaneous localization and mapping: a survey,” *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[3] H. Bavle, P. De La Puente, J. P. How, and P. Campoy, “VPS-SLAM: visual planar semantic SLAM for aerial robotic systems,” *IEEE Access*, vol. 8, pp. 60704–60718, 2020.

[4] I. Torroba, C. I. Sprague, N. Bore, and J. Folkesson, “PointNetKL: deep inference for GICP covariance estimation in bathymetric SLAM,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4078–4085, 2020.

[5] R. Giubilato, S. Chiodini, M. Pertile, and S. Debei, “An experimental comparison of ROS-compatible stereo visual SLAM methods for plane- tary rovers,” in *2018 5th IEEE International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pp. 386–391, Rome, Italy, June 2018.

[6] S. Sharma, J. E. Ball, B. Tang, D. W. Carruth, M. Doude, and M. A. Islam, “Semantic segmentation with transfer learning for off-road autonomous driving,” *Sensors*, vol. 19, no. 11, article 2577, 2019.

[7] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A bench- mark for the evaluation of RGB-D SLAM systems,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 573–580, Vilamoura-Algarve, Portugal, 2012.

[8] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*pp. 225–234, Nov, Nara, Japan, 2007.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: a versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[10] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: large-scale direct monocular SLAM,” in *Computer Vision – ECCV 2014. ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8690 of Lecture Notes in Computer Science, pp. 834–849, Springer, Cham, 2014.

[11] C. Kerl, J. Sturm, and D. Cremers, “Dense visual SLAM for RGB-D cameras,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2100–2106, Tokyo, Japan, 2013.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: unified, real-time object detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, NV, USA, 2016.

[13] W. Liu, D. Anguelov, D. Erhan et al., “SSD: single shot multibox detector,” in *Computer Vision – ECCV 2016. ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9905 of Lecture Notes in Computer Science, , no. 1pp. 21–47, Springer, Cham, 2016.

[14] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[15] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: a deep convolutional encoder-decoder architecture for image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[16] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015*, N. Navab, J. Hornegger, W. Wells, and A. Frangi, Eds., vol. 9351 of Lecture Notes in Computer Science, pp. 234–241, Springer, Cham, 2015.

[17] K. He, G. Gkioxari, P. Dollar, and R. B. Girshick, “Mask R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, Venice, Italy, 2017.

[18] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2881–2890, Honolulu, HI, USA, July 2017.

[19] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[20] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, and J. Song, “Semantic SLAM based on object detection and improved OctoMap,” *IEEE Access*, vol. 6, pp. 75545–75559, 2018.

[21] F. Zhong, S. Wang, Z. Zhang, C. Chen, and Y. Wang, “Detect-SLAM: making object detection and SLAM mutually beneficial,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1001–1010, Lake Tahoe, NV, USA, 2018.

[22] K. Wang, Y. Lin, L. Wang et al., “A unified framework for mutual improvement of SLAM and semantic segmentation,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5224–5230, Montreal, QC, Canada, 2019.

[23] C. Yu, Z. Liu, X.-J. Liu et al., “DS-SLAM: a semantic visual SLAM towards dynamic environments,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1168–1174, Madrid, Spain, October 2018.

[24] R. Mur-Artal and J. D. Tardós, “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, 2017.

[25] B. Bescos, J. M. Facil, J. Civera, and J. Neira, “DynaSLAM: tracking, mapping, and inpainting in dynamic scenes,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[26] Y. Ai, T. Rui, M. Lu, L. Fu, S. Liu, and S. Wang, “DDL-SLAM: a robust RGB-D SLAM in dynamic environments combined with deep learning,” *IEEE Access*, vol. 8, pp. 162335–162342, 2020.

[27] S. Han and Z. Xi, “Dynamic scene semantics SLAM based on semantic segmentation,” *IEEE Access*, vol. 8, pp. 43563–43570, 2020.

[28] 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,” *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, 2019.

[29] L. Cui and C. Ma, “SOF-SLAM: a semantic visual SLAM for dynamic environments,” *IEEE Access*, vol. 7, pp. 166528–166539, 2019.

[30] Y. Ai, T. Rui, X. Yang et al., “Visual SLAM in dynamic environments based on object detection,” *Defence Technology*, vol. 17, pp. 1712–1721, 2020.

[31] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Las Vegas, NV, USA, 2016.

[32] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, and C. Xu, “GhostNet: more features from cheap operations,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1577–1586, Seattle, WA, USA, 2020.

[33] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[34] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2015.

[35] L. Zhao, Z. Liu, J. Chen, W. Cai, W. Wang, and L. Zeng, “A compatible framework for RGB-D SLAM in dynamic scenes,” *IEEE Access*, vol. 7, pp. 75604–75614, 2019.

[36] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, “BiSeNet: bilateral segmentation network for real-time semantic segmentation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, 2018.

[37] G. Lin, A. Milan, C. Shen, and I. Reid, “RefineNet: multi-path refinement networks for high-resolution semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5168–5177, Hawaii, USA, 2017.