Hindawi
International Journal of Distributed Sensor Networks
Volume 2023, Article ID 2025844, 10 pages
https://doi.org/10.1155/2023/2025844

Research Article

Research on Visual SLAM Navigation Techniques for Dynamic Environments

Tongjun Wang and Peijun Zhao

Xinyang Agriculture and Forestry University, Xinyang 464000, China

Correspondence should be addressed to Peijun Zhao; 2010270058@xyafu.edu.cn

Received 6 March 2023; Revised 19 June 2023; Accepted 21 August 2023; Published 1 September 2023

Academic Editor: Diego Alexander Tibaduiza

Synchronous positioning and mapping mainly realize the functions of self-positioning and environment map construction for intelligent navigation technology. In order to solve the problems of low positioning accuracy and poor mapping effect of existing SLAM (simultaneous localization and mapping) systems in indoor dynamic environments and to improve the positioning accuracy, timeliness, and robustness of visual SLAM systems in dynamic environments, an improved visual SLAM method is proposed. Aiming at the inconsistency between the direction of dynamic objects and static background optical flow, this method adopts a high-real-time dynamic region mask detection algorithm to eliminate the feature points in the dynamic region mask, remove the camera motion optical flow according to the original feature information, and then cluster the optical flow amplitude of dynamic objects so as to realize the dynamic region mask detection and eliminate the dynamic signpost points combined with the polar geometric constraints. In order to verify the effectiveness of the improved algorithm, the three evaluation indexes of system accuracy, real-time performance, and the amount of drift are analyzed and verified, respectively, on the TUM dataset. The results show that the proposed algorithm not only has good real-time performance but also improves the accuracy of the system and reduces the amount of drift.

1. Introduction

Synchronous positioning and map construction are the basis of autonomous navigation and environment intelligent perception. RGB-D camera can directly collect images at a low cost, so the SLAM system based on this camera is widely used in the field of intelligent navigation. Intelligent navigation scenes often contain a variety of moving objects, which will cause interference to the SLAM system in both pose estimation and map construction [1].

Visual SLAM systems for the most part assume a static scene and simplify pose estimation based on that, but there are always dynamic objects such as walking people and moving cars in the actual scene. If such objects occupy a large proportion of the field of view, the positioning accuracy and robustness of the system will be seriously affected. These SLAM algorithms add dynamic feature points to pose calculations and generate corresponding error map points in sparse point cloud maps. Therefore, an effective method is needed to distinguish dynamic features or regions. Moving objects will also affect the dense map construction of the SLAM system, and the wrong pose estimation causes the algorithm to wrongly overlay multiframe observation information, resulting in map distortion [2].

The optical flow method is a common method to study interframe motion changes. Literature [3] uses dense optical flow and the CodeBook model to segment dynamic regions, eliminate dynamic feature interference, and establish dense point clouds. In literature [4], dense optical flow is calculated for images, and dynamic region segmentation is obtained by combining locus clustering of points. The scene flow method introduces the idea of 2-dimensional optical flow into 3-dimensional space and calculates the displacement in 3-dimensional space. Literature [5] calculates scene flow, establishes a rigid body motion model, and divides the moving rigid body region. Dense optical flow and scene flow are

effective for dynamic region segmentation, but they will be reduced when optical flow is calculated for a large number of pixels.

Other methods also have a contradiction between operation efficiency and dynamic segmentation and dense mapping. Reference [6] carries out background registration for adjacent image frames, uses the interframe difference method to identify motion, and uses the depth map to quantify and segment dynamic regions without involving dense mapping. In literature [7], the Gaussian model was used to model the background and establish a dense map, but only the rectangular frame selection of moving objects could be obtained. In literature [8], dynamic feature points are screened according to the polar geometry principle, and dynamic regions are marked with a super pixel segmentation algorithm. However, the segmentation computation is large, which reduces real-time performance and does not involve dense mapping. In literature [9], line features in the environment are replaced by point features, and dynamic line features are eliminated by the computational static weight method, which has high computational efficiency but is not applicable to dense mapping. Literature [10] proposes an algorithm based on improved geometry and motion constraints, which is also based on sparse features, so it is not capable of motion region segmentation and online dense mapping.

In addition, more and more researchers apply deep learning methods to SLAM algorithms in dynamic scenarios. Literature [11] uses the Mask R-CNN network [12] to segment dynamic objects and fill in the missing background in the current picture with the help of key frames from the past for dense map construction. Literature [13] uses the SegNet network [14] for image segmentation and dense map construction. Some researchers choose to use an object detection network to first determine the location of dynamic objects and then use other methods to segment. Literature [15] uses an object detection network and the Grab Cut algorithm [16] to segment dynamic objects. In literature [17], the YOLOv3 algorithm [18] was used to preliminarily filter the dynamic region, and then the dynamic region was segmsegmed and filtered more carefully through the consistency evaluation of the distance transform error and photometric error of the edge in the image. In literature [19], YOLOv3 is combined with a polar-line constraint algorithm based on the optical flow method to remove dynamic features, which has high pose accuracy but does not involve dynamic segmentation and dense mapping. The algorithm of deep learning can accurately identify objects and usually performs well. However, it cannot identify dynamic objects outside the network prior range and relies on GPU hardware with a large amount of computation, so it is difficult to apply to mobile robots and other platforms. Aiming at the problems of location accuracy and real-time performance, an improved visual SLAM method combining case segmentation and optical flow is proposed in this paper.

2. State of the Art

2.1. Visual SLAM. Visual SLAM mainly uses the position and attitude of the camera between two frames of images collected by the camera sensor to repeat the trajectory of the entire exploration path, while the pose optimization methods of the camera are mainly divided into filter-based and optimization-based on different implementation principles [20]. The filter-based camera pose optimization method ignores the previous camera state information and optimizes only the estimated camera pose obtained under the current state. This method of processing only according to the current state information will introduce cumulative errors into the SLAM system to some extent, resulting in the system's estimation of the running track of the camera sensor. The synchronous positioning and mapping system (ORB-SLAM) algorithm was designed and implemented in literature [21] based on directional FAST and rotating BRIEF and the improved ORB-SLAM2 in literature [22] based on ORB-SLAM. This method adds local BA optimization, which can improve the impact of cumulative errors on the SLAM algorithm, make the system tracking and mapping more stable during positioning and mapping, and meet the real-time requirements.

Since SLAM needs to use the depth information of the image collected by the sensor when constructing its 3D map of the surrounding environment, the RGB-D camera can obtain the depth information in real time. Therefore, the real-time performance of the RGB-D camera is better, but its detection range is small, so it is often used in indoor mobile robots. Meanwhile, the calculation of camera position in SLAM can be divided into two schemes: the indirect method and the direct method. The method of using gray information from an image combined with photometric error to estimate the pose of the camera sensor is called the direct method. SLAM based on the direct method mainly includes intensive tracking and mapping (DTAM) and large-scale direct monocular SLAM (LSD-SLAM) [23]. For the MonoSLAM method proposed in literature [24] for the indirect SLAM system, the extended Kalman filter (EKF) method is used in the back-end optimization scheme of the algorithm. This method uses the method of minimizing reprojection error to estimate. Although its mapping process can achieve a real-time effect, the sparse space points make it only capable of synchronous positioning and mapping on small occasions. It is the first real-time monocular SLAM system based on EKF. Literature [25] proposed PRAM, an algorithm applicable to the parallel operation of two threads, for positioning and mapping. In the pose optimization and evaluation of the camera, the nonlinear optimization scheme is utilized. At the same time, in order to save the time required for optimization, PTAM adds the judgment of key frames. However, tracking failure is easy to occur during operation, and the effect is better in small scenes. Compared with the direct method, the feature point method has advantages such as more stable operation and better adaptability to the environment, so it has become a research hotspot in recent years [26].

2.2. Optical Flow Network. The optical flow prediction task is to establish a flow field representing the corresponding relationship between each pixel of the original image and the target image given an original image and a target image. Ideally, the deformed image of the target image obtained by the flow field deformation should be very similar to the original

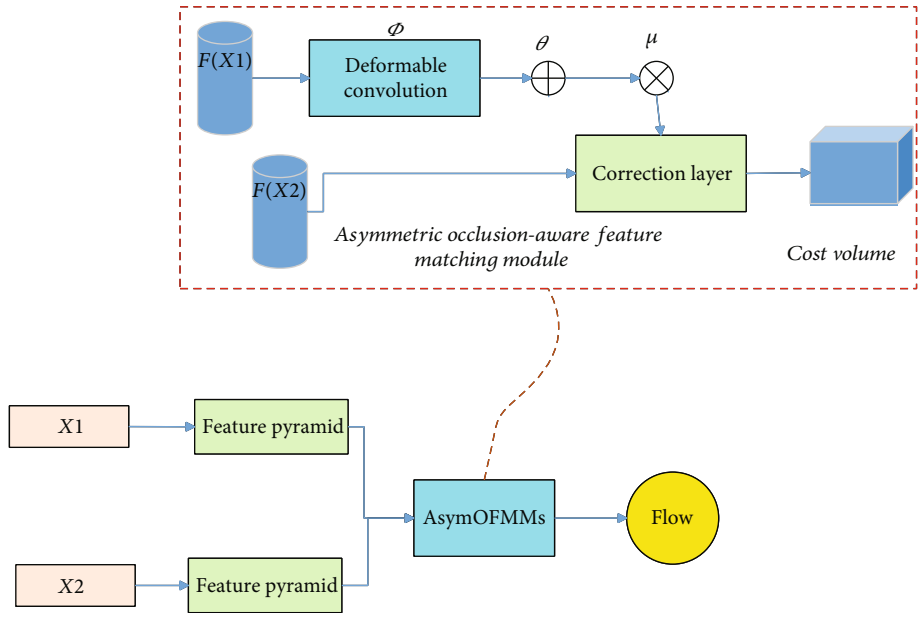

Figure 1: Structure of MaskFlownet-S network.

image. However, the occlusion region generated by the relative displacement between foreground and background brings ambiguity and invalid information to the deformation image acquisition, which reduces the accuracy of optical flow prediction results.

MaskFlownet is an asymmetric feature-matching module that can learn occlusion mask [27]. This module can predict the invalid information caused by occlusion region and filter feature deformation. The end-to-end unsupervised occlusion mask learning significantly improves the performance of the network.

MaskFlownet includes a lightweight optical flow prediction network, MaskFlownet-S, which is faster but less accurate than MaskFlownet. As shown in Figure 1, the structure of MaskFlownet-S is a feature pyramid network (FPN) structure combined with AsymOFMMs (asymmetric feature matching module) that can learn occlusion masks. In Figure 1, the asymmetric feature matching module of the learnable occlusion mask introduces deformation convolution asymmetrical, that is, an additional convolution is carried out while the target feature map is deformed according to the current flow field, and then the learnable occlusion mask is applied to the deformed feature map in the form of multiplication to filter interference information. Finally, the tradeoff μ is added to make up for the missing feature information after filtering the occlusion area. In Figure 1, F(X1) and F(X2) are the feature graphs of input images X1 and X2 after feature pyramid sampling.

2.3. Example Splitting the Network. This paper uses the COCO dataset to train the YOLACT++ network. YOLACT ++ is an improvement on YOLACT, which uses ResNet101 as its backbone network [28]. As shown in Figure 2, C1~C5 correspond to the conv1~conv5 convolution modules of ResNet, respectively. In the figure, P3~P7 are FPN, and feature maps of different scales are obtained through upsampling and downsampling. This structure enables the input features to have strong semantic information, higher accuracy for shallow features, and better robustness for deep features in the pyramid.

YOLACT realizes instance segmentation through two parallel subnetworks, prediction head and protonet. The prediction head branch generates the target anchor, the location parameter of the anchor, the target object category, its confidence, and the mask mask coefficient, and then removes the redundant anchor by nonmaximum suppression. The protonet branch generates a set of prototype masks, multiplicates the prototype mask and the mask mask coefficient generated by the prediction head branch, and then carries out clipping and threshold segmentation to get the segmentation result of the target object in the image.

YOLACT++ introduces deformable convolution in YOLACT to make sampling points more consistent with the shape and size of the object itself. The scale size and aspect ratio of the anchor were changed to increase the number of anchors. Add the mask rescoring branch, and score each segmentation result by generating the product between the IOU (intersection over union) of the mask and the corresponding classification confidence, so as to obtain better results.

3. Methodology

3.1. Visual SLAM System Framework. This paper uses an RGB-D camera as the sensor. Figure 3 shows the complete framework of the SLAM system, where the orange box represents the threads of ORB-SLAM2, and the dotted box represents the motion area detection thread added in

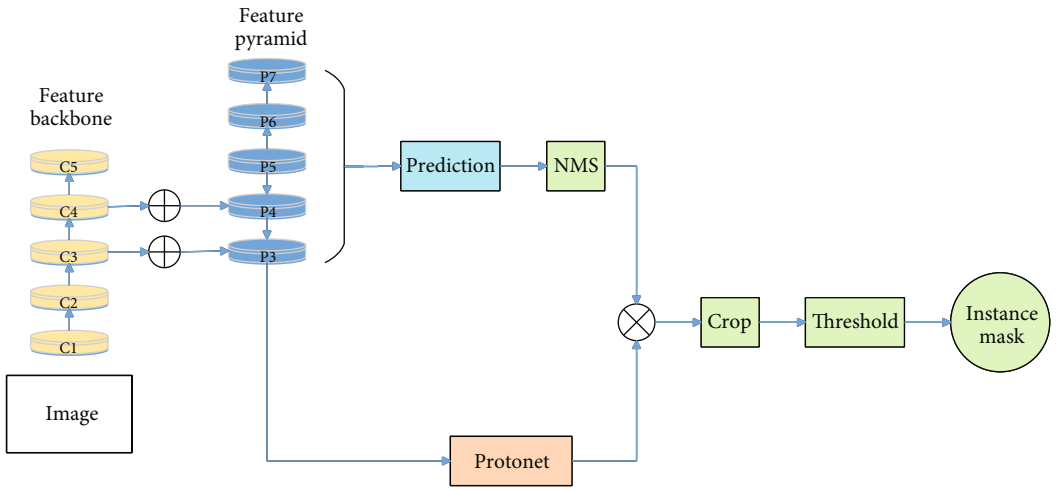

Figure 2: YOLACT network structure diagram.

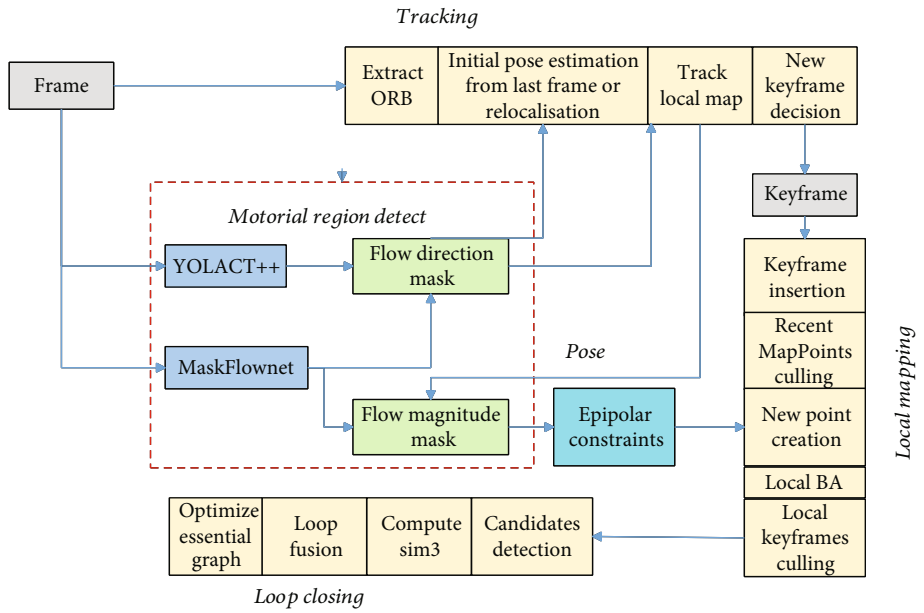

Figure 3: System frame diagram.

this paper based on the original tracking, local mapping, and loop closure detection threads. "sim3" in Figure 3 refers to randomly sampling three pairs of points in ORB-SLAM2 and calculating the similarity transformation matrix.

When the feature detection of new image frames entering the tracking thread is carried out, the domain detection module, according to the optical flow field data and the result of instance segmentation, passes the image into the motion area detection thread for parallel forward reasoning of the optical flow network and instance segmentation network. Then, based on the detection of the dynamic region in the direction of optical flow, a dynamic region mask based on the direction of optical flow is generated for the initial pose estimation and local map tracking module of the tracking thread, and the feature points in the dynamic region are eliminated.

Since it takes a long time to repair the failure points of the depth map, the dynamic region detection module based on the amplitude of the optical flow field is adopted to ensure real-time performance without detecting each frame of the image. If the current frame is taken as the key frame in the tracking thread stage, the dynamic region detection module based on the optical flow field amplitude in Figure 3 generates the corresponding mask according to the depth information, optical flow field information, and interframe pose provided by the local map tracking in the tracking thread. The optical flow prediction result, camera position transformation, and depth value were affected by noise, which resulted in errors in the detected dynamic region. Therefore, after generating the dynamic region mask, epipolar constraints checked the feature points within the mask and determined the feature points that did not conform to the polar geometric constraints within the mask as

outer points. And does not generate the corresponding map signpost points.

Therefore, the dynamic region detection module based on the amplitude of the optical flow field is used to further screen the path points during the creation of key frame waypoints in the local mapping thread, which can reduce the influence of instance segmentation omission. On the premise of ensuring real-time performance and pursuing high accuracy, YOLACT++ was selected as the instance segmentation module, and MaskFlonet-S as the optical flow prediction module.

3.2. Constraints on Polar Geometry. As shown in the epipolar constraints module in Figure 3, after the generation of the flow magnitude mask based on optical flow amplitude, static feature points outside the mask region were further confirmed using the optically geometric constraints. Its principle is shown in Figure 4 (N12 is the pose transformation matrix from O1 to O2). Cameras located at two different positions are used to observe point U. Assuming that the corresponding coordinate vectors of point U on the two frames of images are $i_1 = [p_1\ q_1\ 1]^T$ and $i_2 = [p_2\ q_2\ 1]^T$, then the direction l' vector of the polar line on frame O2 is

$$l' = \begin{bmatrix} I' \\ J' \\ K' \end{bmatrix} = Fi_1, \tag{1}$$

where F is the basic matrix between the two positions of the camera. $[I'J'K']^T$ is the polar direction vector. So the distance from the pole d is zero.

$$d = \frac{\left|i_2^T F i_1\right|}{\sqrt{\left|I'\right|^2 + \left|J'\right|^2}}. \tag{2}$$

Ideally, distance $d = 0$. Due to noise and distortion in the acquired image, the distance between the point and pole line is not 0. Therefore, the threshold dh is set. If the distance obtained is larger than this threshold dh, it is considered that the point does not conform to the polar constraint.

3.3. Dynamic Region Detection Based on Optical Flow Field Vector Direction. An optical flow field is a vector field containing the instantaneous motion information of all pixels in the whole image, usually expressed in the form of two-dimensional vectors; that is, the optical flow vector of each pixel in the optical flow field can be decomposed into a horizontal component and a vertical component.

According to the arctangent function, the included angle between the optical flow vector of each pixel in the current frame and the positive direction of the horizontal axis in the pixel coordinate system is calculated (the included angle range is 0~2π), and the included angle is linearly mapped to the gray range of 0~255. Since the direction angle of the optical flow vector is 0 and 2π, there will be a black-and-white dividing line in the figure.

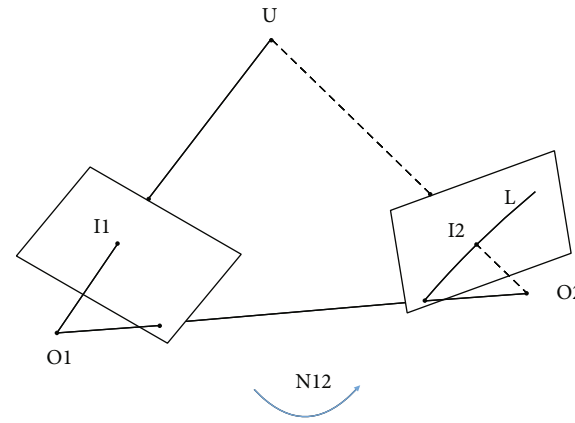

Figure 4: Constraints on polar geometry.

In order to avoid detection of this dividing line by subsequent edge detection, the included angle between the optical flow vector of each pixel and the horizontal axis of the pixel coordinate system is converted according to the following formula:

$$f_a = \begin{cases} \theta, & 0 \leqslant \theta < \pi, \\ 2\pi - \theta, & \pi \leqslant \theta < 2\pi, \end{cases} \tag{3}$$

where θ is the included angle between the optical flow vector and the positive direction of the horizontal axis. The angle between the optical flow vector after fa conversion and the positive direction of the horizontal axis.

If the camera moves continuously, the scene changes in the image frame taken by the camera. Assuming the camera is at rest, the scene can be viewed as moving relative to the camera. Since the camera imaging is on a two-dimensional plane, the three-dimensional motion of the scene is correspondingly projected onto the two-dimensional motion on the camera imaging plane.

If the scene (image region) taken by the camera is regarded as a homogeneous rigid body, then the two-dimensional motion of the frame image on the camera imaging plane can be decomposed into rotation around the center of rotation (image geometric center, camera optical center) and the translation of the center of rotation itself. If the translational velocity is 0 and only the rotational velocity is available, then the velocity of the geometric center of the image is 0, and the optical flow vector of the pixel of the corresponding geometric center of the image is also 0. Under the condition that the translational velocity is 0, the optical flow vector at other pixel points in the image is only caused by its own rotation motion (rotation angular velocity is ω), so the optical flow vector at other pixel points is perpendicular to the vector pointing to the pixel point at the geometric center of the image o. Therefore, the angle including the direction of the optical flow vector around the geometric center of the image (the point where the instantaneous velocity vector is 0, also known as the

instantaneous velocity center) will change sharply from 0 to 2π. If the translational velocity is not 0, the position of the instantaneous center of velocity will be shifted, resulting in the appearance of complex edges.

When the image is not rotated, the instantaneous center of velocity is located at infinity. When the rotation velocity of the image is much less than the translational velocity, the instantaneous center of velocity is located outside the image, and no fold edge will appear θ. When the instantaneous center of velocity is within the image, the fold edge will appear. Therefore, the appearance of the fold edge depends on whether the center of velocity is located in the image. The optical flow vector of each pixel on the image is perpendicular to the vector that the instantaneous center of velocity points to the pixel, so the included angle between the optical flow vector and the positive direction of the horizontal axis is $\pi/2$. Therefore, the difference between the optical flow vector at the four corner points of the image and the angle included in the positive direction of the horizontal axis is equal to the difference between the vector pointing to the pixel point at the instantaneous center of the velocity and the angle included in the positive direction of the horizontal axis. Therefore, θsum of the four included angles can be calculated using the angle obtained in Equation (3). Because there are errors in the prediction results of the optical flow network, the threshold θth is selected. When $\theta\text{sum} > \theta\text{th}$, the instantaneous center of velocity is located in the image.

If the velocity instantaneous center is located in the image, in order to avoid the interference of the folded edge around the velocity instantaneous center on the dynamic target edge and facilitate the subsequent edge detection, this paper moves it out of the visible area of the image. The optical flow vector of pixel points can be decomposed into a rotation component and a translational component. The translational component of all pixel points is the same. The rotation component of a pair of pixels symmetric about the camera's optical center is the same size and opposite direction. According to Equation (4), the average value of all optical flow vectors in the image can be obtained to obtain the approximate value of the translation component. There are dynamic regions in the image, the geometric center does not coincide with the optical center of the camera, and the optical flow prediction results have errors.

$$\begin{aligned} \bar{p} &= \sum_{x=1}^{W} \sum_{y=1}^{T} \frac{p(x,y)}{WT}, \\ \bar{q} &= \sum_{x=1}^{W} \sum_{y=1}^{T} \frac{p(x,y)}{WT}, \end{aligned} \tag{4}$$

where $\bar{p}$ is the mean value of the optical flow field in the horizontal axis direction. $\bar{q}$ is the mean value of the optical flow field along the vertical axis. W is the width of the image. T is the height of the image.

Then, calculate the optical flow field after removing the velocity instantaneous center according to the following equation:

$$\begin{aligned} p_{\mathrm{rm}}(x,y) &= p(x,y) + \frac{Z}{\bar{p}^2 + \bar{q}^2}\overline{p}, \\ q_{\mathrm{rm}}(x,y) &= q(x,y) + \frac{Z}{\bar{p}^2 + \bar{q}^2}\overline{q}, \\ x &= 1, 2, \cdots, W, y = 1, 2, \cdots, T, \end{aligned} \tag{5}$$

where P_{rm} is the value after removing the velocity center of the transverse component of the optical flow field. Q_{rm} is the value after removing the instantaneous center of the velocity of the longitudinal component of the optical flow field. z is a constant. The instantaneous center of velocity is moved out of the image area according to the original offset direction, and then the edge detection is carried out to extract the edge of the dynamic object. The flow chart of the dynamic region detection algorithm based on optical flow field direction is shown in Figure 5.

3.4. Dynamic Region Detection Based on Optical Flow Field Vector Amplitude. It is assumed that the optical flow field f_n of the current frame X_n consists of two parts, one is the optical flow $f_n^{(s)}$ of the static scene caused by the movement of the camera itself, the other is the optical flow $f_n^{(m)}$ of the dynamic object in the scene caused by the movement of the camera itself, and then the static scene optical flow $f_n^{(s)}$ caused by the movement of the camera itself is

$$f_n^{(s)} = ZN_{n\longrightarrow n+1}d_n^{(x)}Z^{-1}u_x - u_x, x \in X_n, \tag{6}$$

where Z is the internal parameter matrix of the camera. $N_n \longrightarrow n+1$ is the camera pose transformation matrix from the current frame to the next frame. U_x is the pixel coordinates in the current frame. $d_n^{(x)}$ is the depth value corresponding to the pixel point. In this paper, the RGB-D camera is used to obtain the depth value of the pixel, and the dynamic region detection results based on the amplitude of the optical flow field vector are applied to the local mapping thread. $N_n \longrightarrow n+1$ in Equation (6) adopts the pose estimated in the tracking thread.

As shown in Figure 6, the white background is the known area of the image, Ω is the area to be repaired in the image, $\delta\Omega$ is the boundary of the area to be repaired, and u is any point on the boundary. In the known region of the image around point u, a neighborhood $B(\varepsilon)$ with the scale of ε is selected, then the pixel value of point u can be approximated by the pixel value in the neighborhood $B(\varepsilon)$. Given the pixel value $X(v)$ at fixed point v and its gradient $\nabla X(v)$, the first-order estimate of point u is

$$X_v(u) = X(v) + \nabla X(v)(u - v). \tag{7}$$

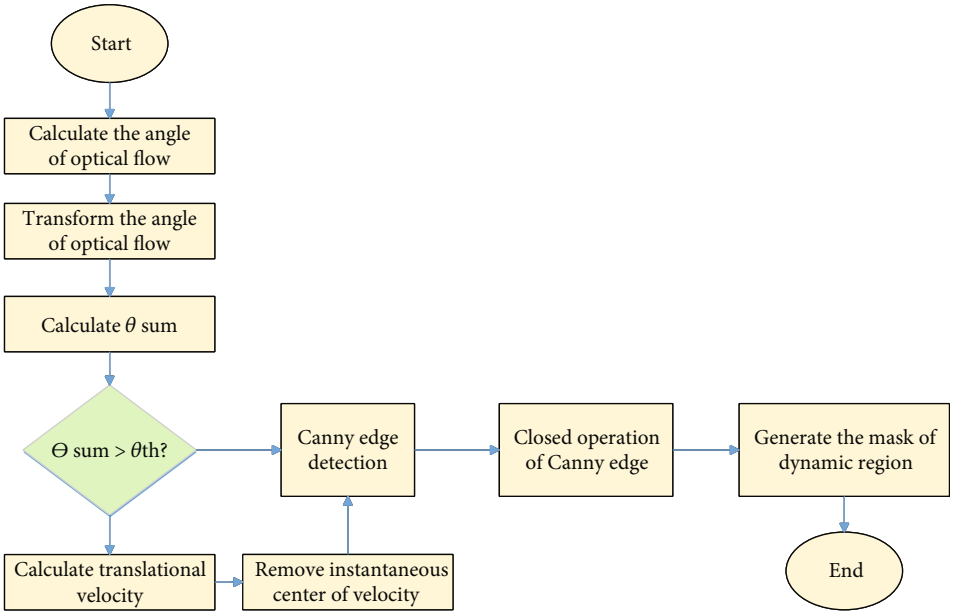

Figure 5: Flow chart of dynamic region detection algorithm based on optical flow field direction.

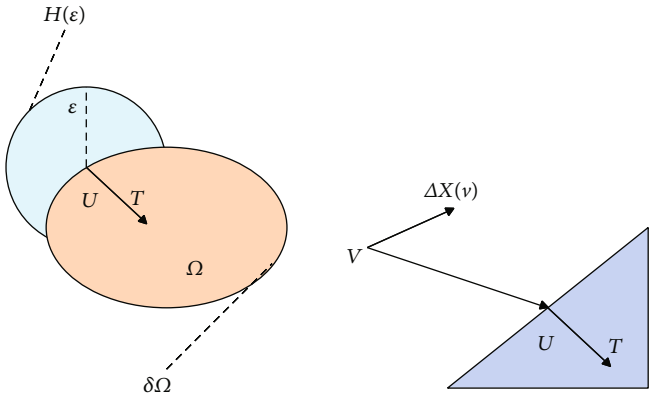

Figure 6: Schematic diagram of depth map repair algorithm.

Then, the pixel value of point u can be expressed as

$$X(u) = \frac{\sum_{v \in H(u)} \omega(u, v)[X(v) + \nabla X(v)(u - v)]}{\sum_{v \in H(u)} \omega(u, v)}, \quad (8)$$

where $\omega(u, v)$ is the weight distribution function, whose value is determined by the direction and distance between u and v and the distance between u and the initial boundary. After processing all pixel points on the boundary, iterate Equation (8) continuously and gradually shrink the boundary of the area to be repaired until the whole area is repaired.

In the above process, the distance between the pixels on the current boundary and those on the initial boundary should be determined, and then gradually shrink to the interior of the area to be repaired according to the order of the distance. For this purpose, the FMM clearly maintains a narrow channel so that the known and unknown areas are clearly separated.

According to Equation (9), the optical flow $f_n^{(m)}$ caused by the motion of the dynamic object can be expressed as

$$f_n^{(m)} = f_n - f_n^{(s)}. \quad (9)$$

The K-means algorithm is adopted to divide all pixels in the amplitude map of the optical flow $f_n^{(m)}$ caused by the motion of dynamic objects into two categories: one is a static region, and the other is a dynamic region. Then, according to Formula (10), the dynamic region mask $W_n \longrightarrow n + 1$ based on the amplitude of the optical flow field is obtained.

$$W_{n \longrightarrow n+1}(u_x) = \begin{cases} 0, & u_x \in R_s, \\ 1, & u_x \in R_m, \end{cases} \quad (10)$$

where R_s is the static region. R_m is the dynamic region. The flow chart of the dynamic region detection algorithm based on optical flow field direction is shown in Figure 7.

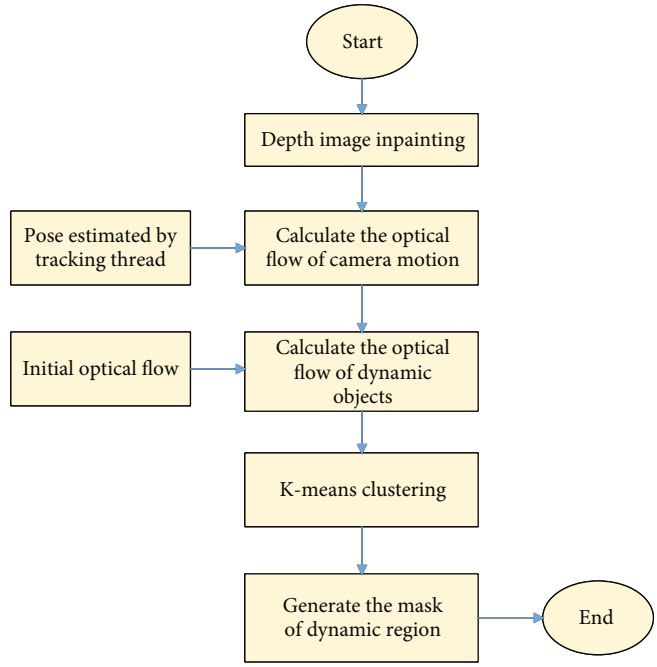

Figure 7: Flow chart of dynamic region detection algorithm based on optical flow field amplitude.

Table 1: Experimental data sequence and main parameter settings.

Data sequence	Dn/s	Nr/m	Qd/(ms-)	Wa/()-s~)	Frame count
fr1/xyz	30.10	7.11	0.24	8.92	799
f1/rpy	27.68	1. 66	0.06	50.14	724
fr1/desk	23.41	9.26	0.41	23.32	614
fr1/room	48.91	15. 98	0.33	13.42	1363
fr1/desk2	24.87	10.16	0.42	29.88	641
fr1/floor	49.88	12.56	0.25	15.07	1243

4. Result Analysis and Discussion

In order to verify the effectiveness of the proposed algorithm, it is compared with the algorithm in literature [22] on the TUM dataset and analyzed and verified from three evaluation indexes of system accuracy, real-time performance, and drift quantity, respectively. The TUM dataset is a public dataset for computer vision and robotics, developed by the Technical University of Munich (TUM) in Germany [29]. The dataset includes image sequences, IMU data, camera calibration parameters, and other information collected by various sensors in multiple scenes. It is suitable for researchers and students with corresponding technical backgrounds to conduct research in autonomous navigation, 3D reconstruction, SLAM (simultaneous localization and mapping), and other areas.

The details of the experimental data series used and the main parameter settings are shown in Table 1, where Dn, Nr, Qd, and ωa represent the duration, track length, displacement velocity, and angular velocity of the data series, respectively. The parameters listed in Table 1 can reflect the performance and advancement of the algorithm to a certain extent, but they are not optimal parameters.

For avoiding the uncertainty of results, experiments were carried out on the Intel Core i7-8300H@2.30 GHz computer with 32 GB memory. Each data sequence was run 60 times, and the results were compared and analyzed with those that appeared most frequently.

4.1. Analysis of Algorithm Accuracy and Drift Quantity. The absolute trajectory error is used to evaluate the consistency of the trajectory, and the absolute distance difference between the estimated trajectory and the real trajectory is compared. The relative pose error is used to evaluate the amount of drift of the system, that is, to compare the difference of the pose transform over a period of time.

In Tables 2 and 3, ATE1 and ATE2 are the ATE obtained by the algorithm in literature [22] and the proposed, respectively. RPE1 and RPE2 are the Rpes obtained by the algorithm in literature [22] and the proposed, respectively. η is the lifting rate. The root mean square error (RMSE) is the root mean square error. The mean is the mean of errors. The median is the median error. Std (standard deviation) is the standard deviation. Combined with the data in Tables 2 and 3, it can be seen that the more intense the camera movement, the more obvious

Table 2: Comparison results of absolute trajectory errors.

Data sequence	ATE1/m				ATE2/m				η/%			
	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std
fr1/xyz	0.01034	0.0858	0.00718	0.00577	0.00900	0.00753	0.00663	0.00491	13.1	12.1	7.6	14.8
fr1/desk	0.02053	0.01428	0.01035	0.01475	0.01430	0.01175	0.00970	0.00815	30.3	17.7	6.3	44.7
fr1/floor	0.01924	0.01425	0.01147	0.01293	0.01221	0.01079	0.01075	0.00572	36.5	24.2	6.3	55.7
fr1/desk2	0.03050	0.02394	0.01838	0.01889	0.02112	0.01810	0.01683	0.01088	30.7	24.3	8.4	42.4
fr1/rpy	0.02905	0.02158	0.01634	0.01944	0.01926	0.01573	0.012 50	0.01111	33.6	27.1	13.5	42.8
fr1/room	0.08145	0.07614	0.07422	0.02892	0.03698	0.03180	0.02867	0.01887	54.6	58.2	60.3	34.7

Table 3: Comparative results of relative position and pose errors.

Data sequence	RPE1/m				RPE2/m				η/%			
	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std	RMSE	Mean	Median	Std
fr1/xyz	0.00592	0.00497	0.00416	0.00321	0.00576	0.00479	0.00405	0.00319	2.7	3.5	2.5	0.7
fr1/desk	0.01188	0.00802	0.00557	0 00876	0.00896	0.00715	0.00574	0.00539	24.5	10.8	-3	38.4
fr1/floor	0.00403	0.00321	0.00274	0.00244	0.00384	0.00313	0.00269	0.00223	4.6	2.5	2	8.5
fr1/desk2	0.01066	0.00862	0.00737	0.00627	0.01049	0.00851	0.00699	0.00614	1.5	1.2	5.2	2.1
fr1/rpy	0.01081	0.00752	0.0055	0.00776	0.00919	0.00717	0.00564	0.00575	14.9	4.6	-2.4	25.9
fr1/room	0.01418	0.00861	0.00621	0.01126	0.01152	0.00807	0.00589	0.00822	18.7	6.2	5.2	26.9

Table 4: Comparison of processing time of each frame.

Data sequence	t1/ms		t2/ms	
	Mean	Median	Mean	Median
fr1/xyz	26.367	26.473	25.206	25.433
fr1/desk	24.773	25.475	27.891	26.852
fr1/floor	21.264	20.975	20.478	19.591
fr1/desk2	31.146	30.562	29.417	28.621
f1/rpy	26.824	26.678	24.075	24.239
fr1/room	25.463	22.555	24.243	21.606

the improvement effect of the proposed algorithm on the system accuracy.

By comparing the absolute trajectory error RMSE of the two algorithms, it can be seen that the RMSE of the proposed algorithm is far lower than the original algorithm. The average RMSE of the proposed algorithm is about 0.0188 m, and that of the algorithm in the literature [22] is about 0.0318 m, indicating that the estimated trajectory of the proposed algorithm is more accurate. Combined with Table 3, the algorithm in this paper improves the accuracy and the drift quantity.

4.2. Real-Time Analysis of Algorithm. The real-time performance of the system is the key to evaluating the SLAM system. In order to achieve real-time performance of the SLAM system, high-performance CPU (Intel Core i7-11800H) and GPU (NVIDIA GeForce RTX 3090), as well as low-latency input devices (Logitech C922 Pro Stream camera) were used in this experiment. Table 4 shows the comparison of the processing time of each frame between the algorithm in this paper and the algorithm in literature [22]. In the table, t1 and t2 are, respectively, the processing time of the algorithm in literature [22] and the algorithm in this paper. The algorithm in this paper is faster in phase processing time, and the average processing time of each frame is about 25 ms, which has better real-time performance and can be better applied in real scenes.

5. Conclusion

In order to reduce the influence of dynamic objects on the accuracy of the SLAM system, a more robust visual SLAM algorithm for dynamic environments is proposed. Based on ORB-SLAM2, the system filters feature points on dynamic objects using optical flow and instance segmentation. The reasoning of the segmentation network and optical flow network is carried out for each frame image simultaneously, and then the dynamic region mask is detected according to the direction information of the optical flow field vector. The feature points in the dynamic region mask are filtered, and then the new map points in the local map-building thread are screened according to the dynamic region mask detected by the optical flow field amplitude information and the polar geometric constraints. The proposed algorithm is compared with the standard ORB-SLAM2 algorithm on the TUM dataset. Experimental results show that the proposed algorithm not only has good real-time performance but also improves the accuracy of the system. This algorithm can be applied to other visual SLAM systems and provides a new idea for the improvement of visual SLAM.

Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare no competing interests.

Acknowledgments

This study is sponsored by the Key R&D and promotion projects in Henan Province (scientific and technological research) (No. 222102110189).

References

[1] 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.

[2] P. Jende, F. Nex, M. Gerke, and G. Vosselman, "A fully automatic approach to register mobile mapping and airborne imagery to support the correction of platform trajectories in GNSS-denied urban areas," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 141, pp. 86–99, 2018.

[3] S. Xu, J. Wang, W. Shou, T. Ngo, A. M. Sadick, and X. Wang, "Computer vision techniques in construction: a critical review," *Archives of Computational Methods in Engineering*, vol. 28, no. 5, pp. 3383–3397, 2021.

[4] Y. Takei, J. Yun, S. Zheng et al., "Integrated spatial genomics reveals global architecture of single nuclei," *Nature*, vol. 590, no. 7845, pp. 344–350, 2021.

[5] Y. Hu, Y. Fang, Z. Ge et al., "A moving least squares material point method with displacement discontinuity and two-way rigid body coupling," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 1–14, 2018.

[6] A. Li, J. Wang, M. Xu, and Z. Chen, "DP-SLAM: a visual SLAM with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, 2021.

[7] H. Xiong, Z. Cao, H. Lu, S. Madec, L. Liu, and C. Shen, "TasselNetv2: in-field counting of wheat spikes with context-augmented local regression networks," *Plant Methods*, vol. 15, no. 1, pp. 1–14, 2019.

[8] S. Jiang and W. Jiang, "Reliable image matching via photometric and geometric constraints structured by Delaunay triangulation," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 153, pp. 1–20, 2019.

[9] K. P. Voss-Fels, M. Cooper, and B. J. Hayes, "Accelerating crop genetic gains with genomic selection," *Theoretical and Applied Genetics*, vol. 132, no. 3, pp. 669–686, 2019.

[10] S. Dian, H. Fang, T. Zhao et al., "Modeling and trajectory tracking control for magnetic wheeled mobile robots based on improved dual-heuristic dynamic programming," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1470–1482, 2021.

[11] Y. Liu and J. Miura, "RDMO-SLAM: real-time visual SLAM for dynamic environments using semantic label prediction with optical flow," *IEEE Access*, vol. 9, pp. 106981–106997, 2021.

[12] P. Bharati and A. Pramanik, "Deep learning techniques—R-CNN to mask R-CNN: a survey," *Computational Intelligence in Pattern Recognition: Proceedings of CIPR*, vol. 999, pp. 657–668, 2020.

[13] Y. Liu, Z. Zhang, X. Liu, L. Wang, and X. Xia, "Efficient image segmentation based on deep learning for mineral image classification," *Advanced Powder Technology*, vol. 32, no. 10, pp. 3885–3903, 2021.

[14] D. Zhu, C. Qian, C. Qu et al., "An improved SegNet network model for accurate detection and segmentation of car body welding slags," *The International Journal of Advanced Manufacturing Technology*, vol. 120, no. 1-2, pp. 1095–1105, 2022.

[15] D. Jiang, G. Li, C. Tan, L. Huang, Y. Sun, and J. Kong, "Semantic segmentation for multiscale target based on object recognition using the improved faster-RCNN model," *Future Generation Computer Systems*, vol. 123, pp. 94–104, 2021.

[16] S. Sun, M. Jiang, D. He, Y. Long, and H. Song, "Recognition of green apples in an orchard environment by combining the GrabCut model and Ncut algorithm," *Biosystems Engineering*, vol. 187, pp. 201–213, 2019.

[17] K. Wang, X. Yao, N. Ma, and X. Jing, "Real-time motion removal based on point correlations for RGB-D SLAM in indoor dynamic environments," *Neural Computing and Applications*, vol. 35, no. 12, pp. 8707–8722, 2022.

[18] D. Wu, Q. Wu, X. Yin et al., "Lameness detection of dairy cows based on the YOLOv3 deep learning algorithm and a relative step size characteristic vector," *Biosystems Engineering*, vol. 189, pp. 150–163, 2020.

[19] J. Peng, Y. Hou, H. Xu, and T. Li, "Dynamic visual SLAM and MEC technologies for B5G: a comprehensive review," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, pp. 1–23, 2022.

[20] J. S. Wang and S. X. Li, "An improved grey wolf optimizer based on differential evolution and elimination mechanism," *Scientific Reports*, vol. 9, no. 1, pp. 1–21, 2019.

[21] M. R. Gkeka, A. Patras, N. Tavoularis et al., "Reconfigurable system-on-Chip architectures for robust visual SLAM on humanoid robots," *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 2, pp. 1–29, 2023.

[22] X. Cui, C. Lu, and J. Wang, "3D semantic map construction using improved ORB-SLAM2 for mobile robot in edge computing environment," *IEEE Access*, vol. 8, pp. 67179–67191, 2020.

[23] M. Servières, V. Renaudin, A. Dupuis, and N. Antigny, "Visual and visual-inertial SLAM: state of the art, classification, and experimental benchmarking," *Journal of Sensors*, vol. 2021, Article ID 2054828, 26 pages, 2021.

[24] Y. Jin, L. Yu, Z. Chen, and S. Fei, "A mono SLAM method based on depth estimation by DenseNet-CNN," *IEEE Sensors Journal*, vol. 22, no. 3, pp. 2447–2455, 2022.

[25] Y. Zhang, A. Azad, and A. Buluç, "Parallel algorithms for finding connected components using linear algebra," *Journal of Parallel and Distributed Computing*, vol. 144, pp. 14–27, 2020.

[26] T. Shi, Y. Qiao, and Q. Zhou, "Spatiotemporal evolution and spatial relevance of urban resilience: evidence from cities of China," *Growth and Change*, vol. 52, no. 4, pp. 2364–2390, 2021.

[27] S. Zhao, Y. Sheng, Y. Dong, E. I. Chang, and Y. Xu, "Maskflownet: asymmetric feature matching with learnable occlusion mask," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6278–6287, Seattle, WA, USA, 2020.

[28] G. Yang, R. Li, S. Zhang, Y. Wen, X. Xu, and H. Song, "Extracting cow point clouds from multi-view RGB images with an improved YOLACT++ instance segmentation," *Expert Systems with Applications*, vol. 230, article 120730, 2023.

[29] W. Wu, L. Guo, H. Gao, Z. You, Y. Liu, and Z. Chen, "YOLO-SLAM: a semantic SLAM system towards dynamic environment with geometric constraint," *Neural Computing and Applications*, vol. 34, no. 8, pp. 6011–6026, 2022.