Received February 6, 2021, accepted February 13, 2021, date of publication February 16, 2021, date of current version March 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059866

Robust Stereo Visual SLAM for Dynamic Environments With Moving Object

GANG LI[1], XIANG LIAO[1], HUILAN HUANG[2], SHAOJIAN SONG[1], (Member, IEEE), BIN LIU[1], AND YAWEN ZENG[1]

[1]College of Electrical Engineering, Guangxi University, Nanning 530000, China
[2]College of Mechanical Engineering, Guangxi University, Nanning 530000, China

Corresponding author: Huilan Huang (accustom@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 51466001, and in part by the Natural Science Foundation of Guangxi under Grant 2017GXNSFAA198344 and Grant 2017GXNSFDA198042.

ABSTRACT The accuracy of localization and mapping of automated guided vehicles (AGVs) using visual simultaneous localization and mapping (SLAM) is significantly reduced in a dynamic environment compared to a static environment due to incorrect data association caused by dynamic objects. To solve this problem, a robust stereo SLAM algorithm based on dynamic region rejection is proposed. The algorithm first detects dynamic feature points from the fundamental matrix of consecutive frames and then divides the current frame into superpixels and labels its boundaries with disparity. Finally, dynamic regions are obtained from dynamic feature points and superpixel boundaries types; only the static area is used to estimate the pose to improve the localization accuracy and robustness of the algorithm. Experiments show that the proposed algorithm outperforms ORB-SLAM2 in the KITTI dataset, and the absolute trajectory error in the actual dynamic environment can be reduced by 84% compared with the conventional ORB-SLAM2, which can effectively improve the localization and mapping accuracy of AGVs in dynamic environments.

INDEX TERMS SLAM, dynamic area detection, stereo vision, automatic guided vehicle.

I. INTRODUCTION

With the rapid development of industrial automation, automated guided vehicles (AGVs) have been widely used in the fields of material transportation, smart storage, and power grid inspection. The mature AGV guidance technologies include magnetic [1], [2] and laser guidance [3], [4]. These guidance methods have been under development for a long time, but their respective shortcomings are also prominent. Magnetic guidance must arrange magnetic tapes or magnetic nails in the working area and use the magnetic sensor on the AGV to identify and track these magnetic materials to complete navigation. This method has poor anti-interference ability and low accuracy. Laser guidance has high precision but requires the placement of a sufficient number of reflectors in the working environment that cover the size of the working area and overcome the degree of occlusion of various parts of the space. As an emerging technology in recent years, visual guidance has the advantages of no prior marking, low sensor cost, rich information acquisition, and high localization accuracy. As the core technology of visual guidance, visual simultaneous localization and mapping (SLAM) has become a hot topic of research in recent years.

Visual SLAM uses images as the only input, calculates camera position and reconstructs a three-dimensional map through multi-view geometry to enable localization in uncharted scenes and the creation of consistent maps. MonoSLAM [5] can be considered the first approach to bring the general SLAM problem from the robotic community into pure vision. Since then, after several decades of development, many impressive SLAM systems have emerged, e.g., ORB-SLAM [6], LSD-SLAM [7], and DSO [8]. These algorithms are designed based on the assumption that objects in the scene are stationary [9] and interference from dynamic objects would directly affect localization and mapping accuracy. In AGV operation scenarios, dynamic objects, such as people, forklifts, and other AGVs, inevitably exist. These dynamic objects can cause false correspondences or occlusion of previously tracked features, leading to tracking failure [10], which can cause the guidance system to malfunction

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

and the AGV to travel out of the defined area, causing loss of public property and personal safety issues. Improving the accuracy of SLAM in dynamic scenarios has thus been an important research direction [11].

To remove the influence of dynamic objects on the SLAM algorithm, two consecutive frames and disparity information are used to detect dynamic regions, and dynamic regions are removed from the front end of the SLAM algorithm to avoid false correspondences caused by objects in the dynamic regions. The main contributions of this paper are as follows.

- A stereo SLAM based on dynamic region rejection is proposed, in which a novel dynamic region detection algorithm is integrated into the front end. The effect of this dynamic area detection algorithm is independent of localization accuracy.
- The SLIC algorithm is improved to remove the redundant computations and improve the speed of dynamic area detection.
- Experiments were performed using the KITTI dataset and real scenes. The results demonstrate the effectiveness of the proposed SLAM algorithm in various dynamic environments.

II. RELATED WORK

The solutions to the SLAM problem in dynamic scenes are roughly divided into two categories according to the object, the first of which is to operate on features and reduce their influence on the estimation of pose [12]–[15]. Most SLAM systems treat dynamic points as outliers and are not involved in localization and mapping. Typical outlier rejection algorithms, such as random sample consensus (RANSAC) [16] used in ORB-SLAM and the robust cost function used in PTAM [17], are able to reject dynamic features to some extent but fail when the number of dynamic features is large. In Ref. [13], robot odometry was used to obtain the fundamental matrix and dynamic feature points were rejected by applying constraints on the epipolar geometry and flow vector bound, the rejection effect of which is affected by the accuracy of the odometry. In Ref. [14], a method for calculating static weights was presented. By combining the static weights of each feature with depth information to estimate the camera pose, the negative effects on the estimation caused by moving objects are reduced. However, the above-mentioned methods cannot completely remove the impact of dynamic objects.

The other solution to the SLAM problem in dynamic scenes is to detect dynamic regions of the graph, which are not involved in the subsequent algorithm [18]–[22]. In Ref. [19], superpixel segmentation on two consecutive frames was performed, and propagated the superpixels of the current frame propagated to the previous frame. The Jaccard distance from the propagated superpixel to the superpixel of the previous frame with the largest overlap was calculated, and an adaptive threshold was used to determine whether the propagated superpixel belonged to the moving object using the Jaccard distance. In Ref. [20], the difference in pixel intensity between the current frame and the ego-motion compensated previous frame was used to roughly detect motion. Pixel classification was done with the segmentation of the quantized depth image. Since the ground is similar to the depth of a dynamic object, this algorithm causes a portion of the ground to be classified as dynamic as well. Semantics, as high-level image features, can help the SLAM system to better understand its surroundings [23], allowing it to select features or optimize data association to improve robustness. In Ref. [21], the contours of the semantic object in the frame were segmented by SegNet. The model was then combined with motion detection to determine whether the object was dynamic or not. In Ref. [22], Mask R-CNN was used to pixel-wise segment the priori dynamic objects in frames and the SLAM algorithm does not extract features on them. Dynamic objects are detected with multi-view geometry models and removed. The occluded background is patched using the previous keyframe information. However, the use of deep learning-based methods requires extensive training of the network, which has high demands on computational performance, and the resulting models may be affected by the training data, leading to low generalizability and poor interpretability.

In combination with the two categories of methods, a SLAM algorithm based on the rejection of dynamic regions is proposed in this study. First, the algorithm quickly identifies the approximate regions of dynamic objects in the image by a few feature points and epipolar geometry, avoiding the processing of the entire image. Then, dynamic objects in the region are segmented using a less computationally intensive superpixel algorithm and easily accessible disparity information. Finally, dynamic regions are excluded by dynamic feature points and dynamic object boundaries, and only the static regions are used for estimation of the pose, to obtain more accurate trajectories and maps.

III. SYSTEM DESCRIPTION

Commonly used vision sensors for visual SLAM are monocular cameras, RGB-D cameras, and stereo cameras. Monocular cameras suffer from scale uncertainty in SLAM algorithms due to the limitations of their sensor properties. RGB-D cameras are capable of outputting RGB color maps and aligned depth maps. Depth maps are calculated by actively emitting an infrared beam of light and receiving its reflection. This method has a limited measurement range and is susceptible to the effects of infrared light emitted by sunlight or other sensors. Hence, a binocular stereo camera was chosen as the visual sensor in the work described in this paper.

Fig. 1 shows an overview of our proposed SLAM algorithm. After acquiring the paired images, the disparity of each point is calculated using semi-global block matching (SGBM) [24] for stereo matching. Next, subsequent processing is performed.

A. DYNAMIC REGION-OF-INTEREST EXTRACTION

The idea of the proposed approach is straightforward. The first step is to use few feature points to roughly detect

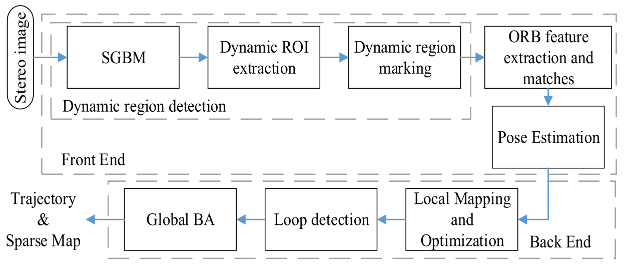
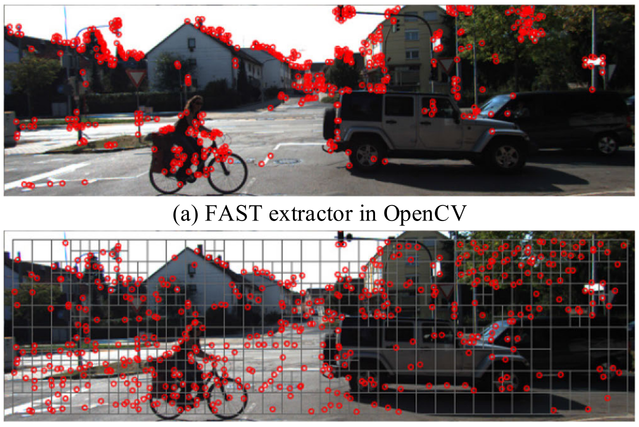


FIGURE 1. Proposed algorithm framework. The dynamic object detection algorithm is integrated into the front-end, which consists of three steps: SGBM, dynamic ROI extraction, and dynamic region marking.

moving objects based on an epipolar constraint. Popular feature extraction methods include scale-invariant feature transform (SIFT) [25], speeded-up robust features (SURF) [26], and features from accelerated segment test (FAST) [27]. SIFT is scale and rotation invariant in nature. This excellent performance results in a huge amount of computation and a long computation time. SURF is an improvement on SIFT and can increase computing speed by an order of magnitude, but it still struggles with real-time performance. FAST is a corner-point feature. It detects quickly by comparing the intensity of the center point with the surrounding pixels. Fig. 2(a) shows the corner points extracted from an image using the FAST extractor in OpenCV. The extracted feature points are dense and stacked heavily, with some objects having few or even no feature points. To better detect dynamic objects, feature points must be extracted uniformly in the image. A quadratic tree is a tree-like structure with four sub-blocks under each node that is commonly used for data classification in two-dimensional space. By filtering the feature points in the image through a quadratic tree, the distribution of feature points can be made more uniform. The improved extraction process is as follows. (1) The image is divided into several square regions and the FAST extractor is used to extract excess features from each square region. (2) The initial nodes are assigned to the image and the feature points are put into the corresponding nodes. (3) The process checks whether each node contains one feature point only, and, if not, divides the region in which it is located into four parts and assigns new nodes to these four parts (the feature points in the old node are stored in the corresponding new node). (4) Step (3) is repeated until the total number of nodes that contain a feature point reaches the desired number of feature points N. Fig. 2(b) shows the effect of using this method to extract feature points, which are evenly distributed in the figure.

A set of homogenous feature points $\left(P_1^{t-1}, P_2^{t-1}, \cdots, P_N^{t-1}\right)$ of a previous frame was extracted by the above method. Tracking these points using the Lucas-Kanade method [28] obtained the corresponding feature points $\left(P_1^t, P_2^t, \cdots, P_N^t\right)$ in the current frame. Fundamental matrices F between previous and current frames are calculated for these feature pairs using the 9-point algorithm, and a more accurate value is obtained by RANSAC. For epipolar geometry, relation exists between the point in the previous



(a) FAST extractor in OpenCV

(b) FAST extractor base on quadratic tree

FIGURE 2. FAST extraction. The red points are feature points, and the gray lines in (b) are node boundaries.

frame and the epipolar line in the current frame as

$$L_i^t = FP_i^{t-1}, \tag{1}$$

where $P_i^{t-1} = \left[x_i^{t-1}, y_i^{t-1}, 1\right]^T$ is the ith feature point of the previous frame and $L_i^t = [A_i, B_i, C_i]^T$ is the coefficient of the epipolar line where the ith feature of the previous frame is located in the current frame. If the scene is stationary, the features tracked in the current frame lie on their corresponding epipolar line. Conversely, feature points that are not on epipolar lines belong to dynamic objects. The distance D_{epi} from feature point P_i^t to its corresponding epipolar line L_i^t is calculated by the following equation:

$$D_{epi} = \frac{\left|A_i x_i^t + B_i y_i^t + C_i\right|}{\sqrt{A_i^2 + B_i^2}}, \tag{2}$$

where x_i^t and y_i^t are the coordinates of point P_i^t. Owing to the presence of noise, a point is considered to be a feature on a dynamic object when D_{epi} exceeds a certain threshold value. The distance threshold is proportional to the disparity at that point. It is determined by the following equation:

$$T_{epi} = \alpha_{epi} + \beta_{epi} d_i, \tag{3}$$

where α_{epi} and β_{epi} are constants and d_i is the disparity value of point P_i^t in the current frame.

After the dynamic feature points are obtained [see Fig. 3(b)], regions of interest (ROIs) are acquired for segmentation and labeling in the next step. When the distance between one dynamic feature point and another dynamic feature point is not greater than the distance threshold T_s, and the difference in disparity value between them is not greater than the disparity threshold T_{disp}, these two points are considered to belong to the same object and are classified as a group. The closer the object is to the camera, the larger the area occupied by the object in the image, and the greater the distance between feature points belonging to the same object, so the distance threshold T_s is positively correlated

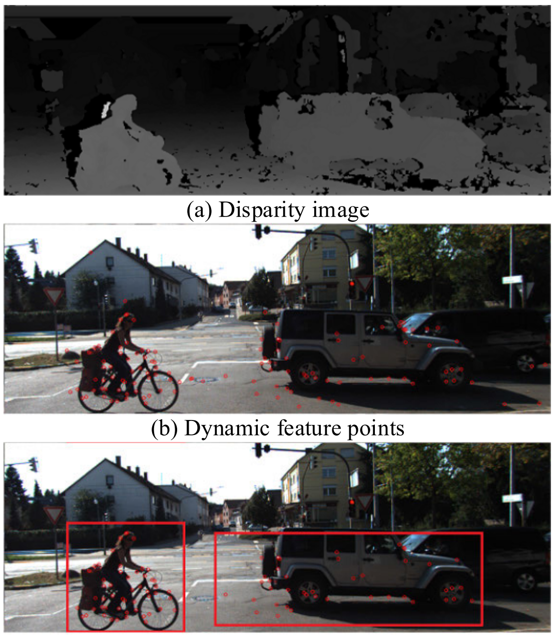


FIGURE 3. **Dynamic region-of-interest extraction. (a) is the disparity image obtained by SGBM. The red points in (b) and (c) are dynamic feature points. The red rectangles in (c) are dynamic region-of-interests.**

with disparity. The threshold T_s is given by the following formula:

$$T_s = \alpha_s + \beta_s d_i, \tag{4}$$

where α_s and β_s are constants.

To exclude noise and other factors, an object is considered to be moving when there are more than four dynamic features on the object. The area in which the dynamic feature points are located is the dynamic ROI. The extracted dynamic feature sets and dynamic ROIs are shown in Fig. 3(c).

B. DYNAMIC REGION MARKING

Once the dynamic ROI is obtained, the position of the dynamic object on the image can be roughly confirmed. For more accurate segmentation of dynamic regions, it is necessary to segment the image and extract the contours of dynamic objects. Simple linear iterative clustering (SLIC) [29] is a k-means-based superpixel segmentation algorithm that classifies pixels using the distance D_{SLIC} between the Lab color vector $C = [L, a, b]$ and the space vector $S = [x, y]$ of the pixel and cluster center as a measure of similarity. The distance D_{SLIC} is expressed as

$$D_{SLIC} = \sqrt{\left(\frac{\|C_j - C_i\|_2}{N_c}\right)^2 + \left(\frac{\|S_j - S_i\|_2}{N_s}\right)^2}, \tag{5}$$

where $j = 1, 2, \cdots, K$ denotes each cluster center and i the pixels within the search range of the cluster center. N_c and N_s are the maximum color distance and maximum spatial distance, respectively. The maximum color distance N_c is determined from the image. The maximum spatial distance N_s is equal to the initial spatial distance S_{step} from the cluster center, as follows:

$$N_s = S_{step} = \sqrt{\frac{m \times n}{K}}, \tag{6}$$

where m and n are the length and width of the image, respectively, and K is the number of cluster centers sets. If the input image is a grayscale map, the Lab color vector can be simplified to a one-dimensional vector with only grayscale values: $C = [gray]$.

The steps of the SLIC algorithm are as follows. (1) The cluster centers are distributed equally in the image with S_{step} as the distance. (2) The distances from each cluster center to pixels are computed within a $2S_{step} \times 2S_{step}$ region and the pixel is assigned to the class with the smallest distance. (3) The cluster center for each cluster is updated. (4) Steps (2) and (3) are repeated until convergence (typically 10 repetitions) to obtain better results. From the previous step, it can be seen that each iteration must calculate the distance between the cluster center and the pixels in its $2S_{step} \times 2S_{step}$ range. However, only the cluster of pixels near the edges would change in each iteration. Calculating the distance between the clustering center and its nearby pixels is completely redundant. For real-time considerations, a new cluster update approach is proposed in which only unstable pixels at the edge are classified and computed for each iteration. The steps are as follows. (1) The cluster centers are evenly distributed and edge pixels are marked as unstable points. (2) The distances from each cluster center to unstable points are computed within a $2S_{step} \times 2S_{step}$ region and the cluster and unstable markers are updated. (3) The cluster center and the unstable flag of pixels adjacent to unstable points are updated. (4) Steps (2) and (3) are repeated until convergence. The criteria for updating the unstable point marker in step (3) can be expressed by the following equation:

$$Flag_p = \begin{cases} 1, & if\ cluster(p) \neq cluster(q) \\ & \vee\ Flag_q = 1 \\ 0, & otherwise, \end{cases} \tag{7}$$

where q is the unstable point updated in step (2) and p the pixel in the four neighborhoods of q. Fig. 4 shows the process of updating the flags for unstable points. Superpixel segmentation is performed on the dynamic ROIs using the modified SLIC algorithm. The effect is demonstrated in Fig. 5(a), from which it can be seen that the edges of the superpixels are better able to outline the object.

However, one cannot distinguish which superpixels belong to the same object, and so it is necessary to confirm the type of boundaries of the superpixels. Generally, there are three types of boundaries: co-planar, hinge, and occlusion. Creating a disparity plane model for each superpixel, the disparity plane model can be formulated by the following equation:

$$d_p = \alpha_j x_p + \beta_j y_p + \gamma_j, \tag{8}$$

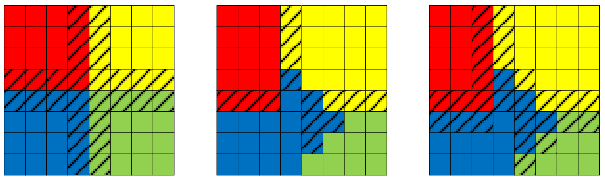

(a)Unstable point before update
(b) Cluster-altered unstable points
(c)Unstable point after update

FIGURE 4. **Update of the unstable point. Identical colors indicate that the points belonging to a single cluster center and the shaded areas indicate unstable points.**

where $j = 1, 2, \cdots, K$; α_j, β_j, and γ_j are the disparity plane parameters of the jth superpixel, x_p and y_p are the coordinates of point p in the superpixel, and d_p is the disparity value of point p in the superpixel. Owing to mismatching and noise, several outliers exist in the disparity obtained by SGBM. The RANSAC method was used to reject outliers to obtain a more accurate disparity plane model.

The boundary energy is denoted $E = E_{smo} + E_{prior}$ to determine the type of the superpixel boundary. E_{smo} is the plane smoothness energy used to check boundary-plane agreement. If two superpixels belong to the same object, which means that they are co-planar, their disparity planes should agree. If they are hinge type, then their disparities should agree in their boundaries. If they are occlusion type, the occluding one should be closer to the camera and have greater disparity. Thus,

$$E_{smo}\left(\theta_i, \theta_j, o_{i,j}\right) = \begin{cases} \phi_{occ}\left(\theta_i, \theta_j\right) & if\ o_{i,j} = lo \\ \phi_{occ}\left(\theta_j, \theta_i\right) & if\ o_{i,j} = ro \\ \frac{1}{|B_{i,j}|} \sum_{p \in B_{i,j}} \left(\hat{d}\left(p, \theta_i\right) - \hat{d}\left(p, \theta_j\right)\right)^2 & if\ o_{i,j} = hi \\ \frac{1}{|U_{i,j}|} \sum_{p \in U_{i,j}} \left(\hat{d}\left(p, \theta_i\right) - \hat{d}\left(p, \theta_j\right)\right)^2 & if\ o_{i,j} = co, \end{cases} \tag{9}$$

where θ_i and θ_j are the disparity plane parameters of the superpixels s_i and s_j, respectively; $o_{i,j}$ denotes the type of boundary between superpixels s_i and s_j; lo denotes left occlusion, where a small indexed superpixel occludes a large indexed superpixel; ro denotes right occlusion, as opposed to lo; hi denotes hinge; and co denotes co-planar. $B_{i,j}$ is the set of pixels on the boundary between superpixel s_i, s_j; $U_{i,j}$ is the set of pixels in the concatenation of superpixel s_i and s_j. $\hat{d}\left(p, \theta_i\right)$ is the estimated disparity for point p. $\phi_{occ}\left(\theta_{front}, \theta_{back}\right)$ is a function that penalizes occlusion boundaries, which is given by the following expression:

$$\phi_{occ}\left(\theta_{front}, \theta_{back}\right) = \begin{cases} \lambda_{pen} & if\ \sum_{p \in B_{front,back}} \left(\hat{d}\left(p, \theta_{front}\right) - \hat{d}\left(p, \theta_{back}\right)\right) < 0 \\ 0 & otherwise. \end{cases} \tag{10}$$

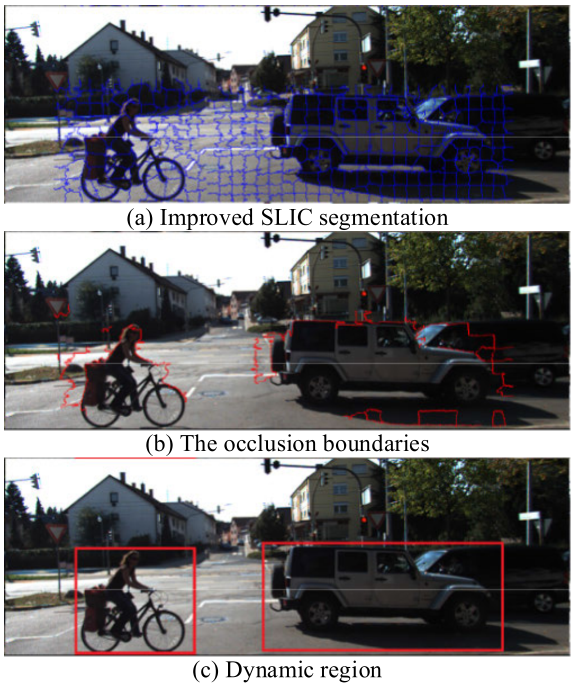


FIGURE 5. **Dynamic region marking. The blue lines in (a) are the boundaries of superpixels. The red lines in (b) are the boundaries of type occlusion. The rectangle regions in (c) are the result of dynamic object detection.**

E_{prior} is the complexity energy function used to make the solution of the total energy more continuous:

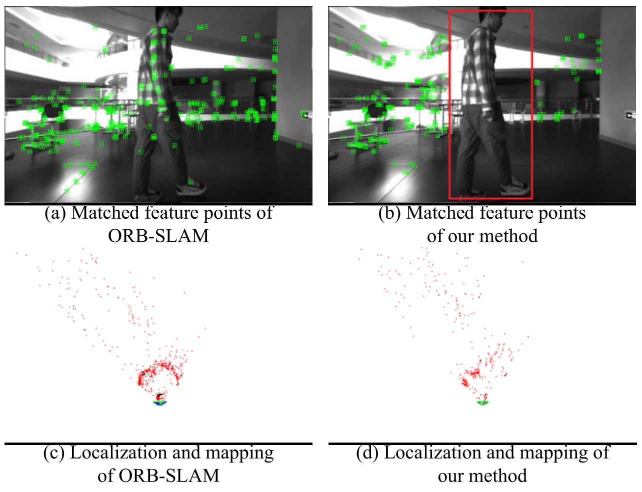
$$E_{prior}\left(o_{i,j}\right) = \begin{cases} \lambda_{occ} & if\ o_{i,j} = lo \vee o_{i,j} = lo \\ \lambda_{hinge} & if\ o_{i,j} = hi \\ 0 & if\ o_{i,j} = co, \end{cases} \tag{11}$$

where λ_{occ}, λ_{hinge} are constants with $\lambda_{occ} > \lambda_{hinge} > 0$.

The boundary type of each superpixel is obtained by minimizing the energy function E. In Fig. 5(b), the red lines indicate the boundaries of type occlusion, which provide a good delineation of the objects in the figure. Starting at the dynamic feature point, the search for boundaries proceeds in the positive and negative directions of the x and y axes and stops when an occlusion-type boundary is found. The minimum enclosing rectangle is obtained for all found boundary points. The rectangular regions are the dynamic regions, as shown in Fig. 5(c).

C. TRACKING AND MAPPING

The proposed robust SLAM algorithm uses the same framework as ORB-SLAM, which is divided into three independent threads: tracking, local mapping, and loop closing. The image is divided into static and dynamic regions by the above method. The tracking thread only extracts feature points in the static region to match the map points and calculates the poses. Optimization of camera pose is accomplished by minimizing reprojection errors using motion-only bundle adjustment (BA). The local mapping thread is responsible for managing the map points and performing local BA to optimize the position and map points when a new keyframe is input. The loop-closing thread detects whether a loop has occurred, and when a loop occurs, it stops the local mapping thread to create a new thread to perform a global BA of the poses and map points. Fig. 6 shows the processing of map-point matching as well as the top view of the localization and mapping results. ORB-SLAM only uses RANSAC to reject dynamic feature points that do not match the currently estimated pose to ensure the accuracy of the localization. However, because of the large number of dynamic feature points, some are considered as inliers involved in the estimation of the pose and mapping. Dynamic features of map points are stacked in front of the camera and existing static objects are repeatedly added to the map. The camera position drifts accordingly. The proposed method does not extract feature points in dynamic regions, which means that no feature points in the dynamic region are involved in map-point matching. Therefore, localization and mapping are not affected by dynamic regions. The map is built accurately, and the camera pose is stable.


(a) Matched feature points of ORB-SLAM

(b) Matched feature points of our method

(c) Localization and mapping of ORB-SLAM

(d) Localization and mapping of our method

FIGURE 6. Localization and mapping comparison. These images are the same frame during the sequence processed by the SLAM system. The green points in the top row of the figures are the matched map points and the red rectangle is the dynamic region detected by the proposed method. The red points in the bottom row are the map points, and the green triangles are the poses of the camera.

IV. EXPERIMENTAL RESULTS

The proposed method was tested on both public datasets and real scenarios to verify its performance. The AGV used was a two-wheeled differential vehicle with a binocular camera that outputs a grayscale image of size 752 × 480. The computer employed was a Lenovo Y50-70 laptop with an i5-4210H CPU and 8 GB of RAM running Ubuntu 16.04.

A. DYNAMIC REGION DETECTION

The dynamic region detection algorithm proposed in the paper was validated as described herein. Since there are no publicly available datasets for dynamic objects, the data used in this experiment are images from the KITTI dataset [30] containing dynamic objects and a photo set taken by the AGV in real scenes. Fig. 7 shows examples of dynamic area detection on the photo set of real scenes including vehicles, bicycles, and pedestrians on a road. Fig. 8 shows examples of dynamic area detection on the KITTI dataset, with the dynamic target being all moving vehicles. It can be observed that the method proposed in this paper is able to accurately segment the dynamic regions of the figures. The proposed algorithm can distinguish between two different dynamic objects even in the presence of occlusion of the moving object (e.g., the second and third columns of both Figs. 7 and 8). Some static features are incorrectly identified as dynamic features (e.g., the first two columns of Fig. 7 and the second column of Fig. 8). The reason for this phenomenon is that there are too many dynamic feature points in the figure, resulting in an inaccurately calculated F matrix. When the dynamic region is later identified, these feature points are not surrounded by boundaries labeled as occlusion, so the region is not recognized as dynamic.

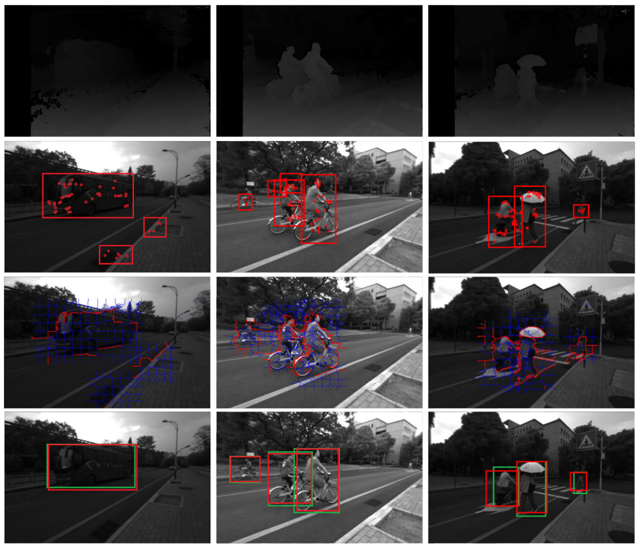


FIGURE 7. Examples of dynamic area detection on realistic photos. First row, disparity images obtained by SGBM; second row, dynamic regions of interest; third row, results of SLIC and boundary marking; fourth row, dynamic areas (red rectangles) and ground truths (green rectangles).

Table 1 shows the results of this method for the KITTI dataset and the photo set of real scenes. Table 2 shows the evaluation indexes in dynamic target detection. Precision and recall are calculated by the following equations:

$$P = \frac{TP}{TP + FP}, \tag{12}$$

$$R = \frac{TP}{TP + FN}, \tag{13}$$

TABLE 1. Dynamic area detection results.

Dataset	Total number of dynamic objects	True positives	False negatives	False positives
KITTI	238	203	35	27
Real scenes	184	165	26	16

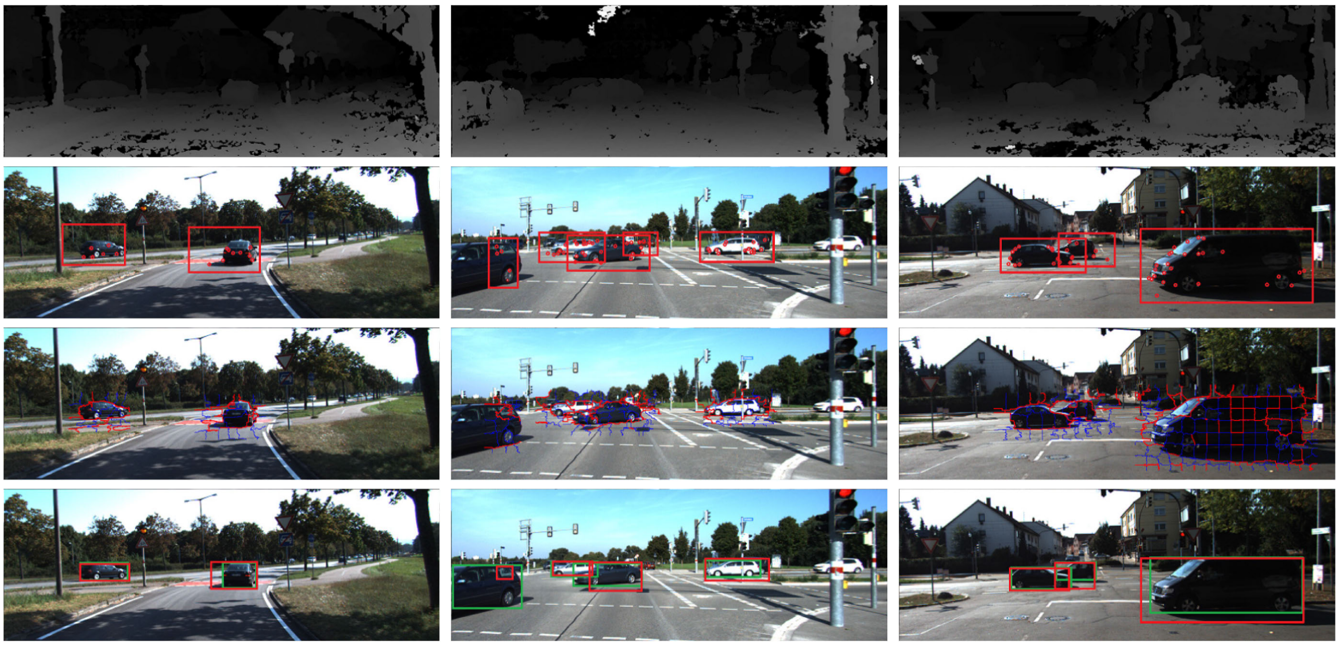


FIGURE 8. Examples of dynamic area detection on KITTI benchmark. First row, disparity images obtained by SGBM; second row, dynamic regions of interest; third row, results of SLIC and boundary marking; fourth row, dynamic areas (red rectangles) and ground truths (green rectangles).

TABLE 2. Algorithm evaluation indexes.

Dataset	Precision (%)	Recall (%)	IoU (%)
KITTI	88.26	85.30	77.82
Real scenes	91.16	89.76	82.53

where TP is the number of correctly identified dynamic objects, FP the number of incorrectly identified dynamic objects, and FN the number of undetected dynamic objects. The Intersection over Union (IoU) is the ratio of the intersection and union between the detected region of a dynamic object and its ground truth:

$$IoU = \frac{DR \cap GT}{DR \cup GT}, \tag{14}$$

where DR is the detected region of dynamic object and GT the ground truth of the dynamic object.

As can be seen from Table 2, the method advanced in this paper yields good results for all indexes, with both a precision and recall of over 85%. In the photo set of real scenes, the indexes are higher than those using the KITTI dataset because the dynamic objects in the figure are closer to the camera and the imaging is larger in the figure.

B. STEREO SLAM WITH DYNAMIC REGIONS REMOVED

The SLAM algorithm proposed in this paper was validated using sequences containing dynamic scenes of the KITTI dataset. Sequence 01 is a highway scene and sequences 03, 04, 05, 07, and 08 are town scenes, in which sequences 05 and 07 contain loops. There are two metrics to evaluate SLAM systems: absolute trajectory error (ATE) and relative pose error (RPE). The ATE is the difference between the estimated trajectory and ground truth, which can directly reflect the algorithm accuracy and global consistency of the estimated trajectory, and is usually used to evaluate the performance of the whole SLAM system. The RPE calculates the difference in the amount of pose change over the same time stamp and is suitable for evaluating the drift of the system. We employed the absolute trajectory root-mean-square error (RMSE) t_{abs} [31], average relative translation t_{rel} and rotation r_{rel} errors [30] for quantitative evaluations. We compare the proposed method with the ORB-SLAM2 [6] and DynaSLAM [22] for experiments, and the experimental results are shown in Table 3. Note that the ORB-SLAM2 and DynaSLAM mentioned in this paper are both their stereo versions. In this experiment the loop-closing threads of all algorithms are open. It can be seen that our method performs better than the other two methods for these KITTI datasets containing dynamic targets. In these sequences, most of the vehicles are parked on the curb and occlude larger areas of the frame. DynaSLAM excludes all cars as dynamic objects without distinction, leaving fewer feature points for localization. In contrast, our method only eliminates the moving vehicles, so that ours performs better than DynaSLAM for most sequences.

To verify the effectiveness of the proposed algorithm in real dynamic scenes, five dynamic scenes were designed. Scene 1 shows the AGV stationary in a dynamic environment. Scenes 2 and 3 show the AGV driving a straight line approximately 13.6 m long in a dynamic environment. Scenes 4 and 5 are scenes in which the AGV travels

TABLE 3. Contrast Experiments Results on KITTI Dataset (Units: m).

Sequences	ORB-SLAM2 (Stereo) [6]			DynaSLAM (Stereo) [22]			Proposed method		
	t_{rel} (%)	r_{rel} (deg/100m)	t_{abs} (m)	t_{rel} (%)	r_{rel} (deg/100m)	t_{abs} (m)	t_{rel} (%)	r_{rel} (deg/100m)	t_{abs} (m)
01	**1.39**	0.21	10.4	1.57	0.22	9.4	1.52	**0.18**	**9.1**
03	0.71	0.18	0.6	**0.69**	0.18	0.6	0.76	0.25	0.8
04	0.48	0.13	0.2	0.45	**0.09**	0.2	**0.40**	0.20	0.2
05	0.40	0.16	0.8	0.40	0.16	0.8	**0.38**	0.16	**0.7**
07	0.50	0.28	0.5	0.52	0.29	0.5	**0.42**	**0.25**	**0.4**
08	1.05	0.32	3.6	1.05	0.32	3.5	**0.98**	**0.29**	**3.1**

TABLE 4. The ATE of Each Algorithm for Real Environment (Units: m).

Sequences	ORB-SLAM2 (Stereo)				DynaSLAM (Stereo)				Proposed method			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
01	0.768	0.534	0.173	0.552	**0.001**	**0.001**	0.001	**0.001**	0.010	0.002	0.001	0.010
02	0.867	0.850	0.794	0.169	**0.085**	0.076	0.073	**0.037**	0.086	**0.075**	0.073	0.038
03	1.016	0.853	0.655	0.552	**0.076**	**0.067**	**0.058**	0.038	0.080	0.070	0.063	0.038
04	0.457	0.413	0.404	0.200	0.118	0.102	0.091	0.059	**0.115**	**0.098**	**0.088**	0.059
05	0.524	0.482	0.477	0.206	0.115	0.101	**0.092**	0.053	**0.113**	**0.099**	0.093	0.053

TABLE 5. The RPE (Translation Part) of Each Algorithm for Real Environment (Units: m).

Sequences	ORB-SLAM2 (Stereo)				DynaSLAM (Stereo)				Proposed method			
	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.	RMSE	Mean	Median	S.D.
01	0.014	0.009	0.003	0.011	**0.001**	**0.001**	0.001	**0.001**	0.007	0.002	0.001	0.007
02	0.024	0.018	0.014	0.016	**0.017**	0.014	0.013	**0.010**	0.018	**0.013**	0.013	0.011
03	0.022	0.019	0.016	0.012	**0.017**	0.015	**0.014**	**0.007**	0.019	0.015	0.015	0.008
04	0.020	0.017	0.015	0.010	0.019	0.016	**0.014**	0.010	**0.018**	0.016	0.015	**0.009**
05	0.061	0.028	0.019	0.054	**0.018**	**0.016**	0.016	**0.008**	0.021	0.018	0.016	0.011

a 6.4 m × 8.8 m rectangular loop in a dynamic environment. The dynamic environment of scenes 1, 2, and 4 are low dynamic, with only one person crossing in front of the AGV. Scenes 3 and 5 are highly dynamic, with at most three people appearing in front of the screen at the same time. They pass or walk in front of the AGV. Particularly, in scene 5, there is a person who keeps leading the way in front of the AGV during the whole rectangular loop travel. Tables 4 and 5 show the ATE and RPE (translation part) of each algorithm for scenes 1–5, respectively. To facilitate further comparisons, the RMSE, mean error, median error, and standard deviation (S.D.) are presented in these tables. RMSE is susceptible to large or occasional errors and is better able to be used to evaluate the robustness of the algorithm [32], whereas S.D. is better able to be used to evaluate system stability [33]. In this experiment the loop-closing threads of all algorithms are open.

As can be seen from Tables 4 and 5, the proposed method performs similarly to DynaSLAM in real dynamic scenes, but outperforms ORB-SLAM2. In the case of scene 1, in which the camera is stationary, the proposed method drifts essentially unaffected by the pedestrian. In scenes 2–5,

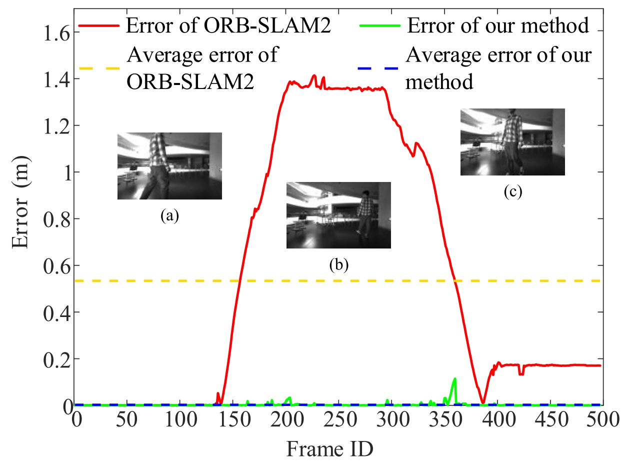


FIGURE 9. Trajectory error of each algorithm in scene 1. (a) is the frame with id 150, the pedestrian was walking to the right in front of the camera. (b) is the frame with id 250, the pedestrian was turning. (c) is the frame with id 350, the pedestrian was walking backwards.

the localization accuracy of the proposed method is higher compared to ORB-SLAM2, where the average improvement values of RMSE and S.D. are approximately 84% and 79%

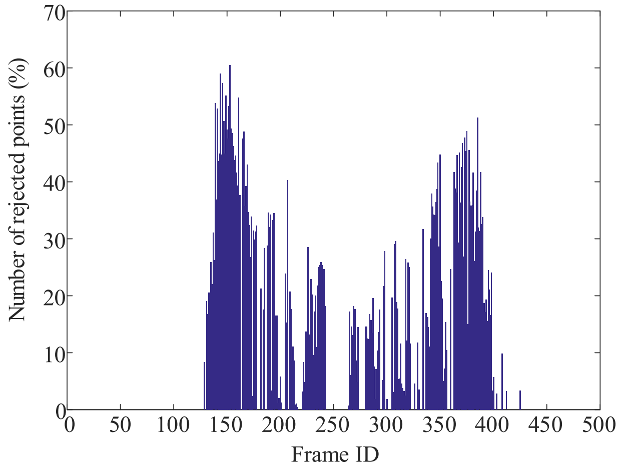


FIGURE 10. The number of dynamic feature points rejected by the proposed method in scene 1. Our method works well most of the time, but fails in a few cases, such as when the pedestrian turns (frame 241-263), or when the pedestrian moves slowly (frame 272-279, etc.).

for ATE and approximately 31% and 41% for RPE, respectively. The results of scenes 3 and 5 show that the proposed method performs well even in a highly dynamic environment. The improvement of the proposed method over ORB-SLAM in the KITTI dataset is lower than that in the real scenes, due to the wide field of view of the camera used in the KITTI dataset and the small percentage of dynamic objects in the image, most of which can be removed by RANSAC.

Fig. 9 shows the comparison of the trajectory error between the proposed method and ORB-SLAM2 in scene 1. The pedestrian walks from the left to the right of the camera at frame 125, turns at frame 230 then walks to the left, and finally walks out of the camera's view at frame 420. The trajectory of ORB-SLAM2 begins to drift to the left when the pedestrian appears, and then drifts to the right when the pedestrian turns around and walks back. The trajectory error is stabilized at 0.18 m after the pedestrian walks off the screen. In contrast, the proposed algorithm is almost unaffected during the entire pedestrian walking process. Fig. 10 shows the number of dynamic feature points that are rejected in each frame of scene 1. In this experiment, we extract a total of 1200 feature points per frame. These rejected feature points which should have been extracted on the dynamic region would be replaced by feature points in the non-dynamic region to ensure that there are enough feature points for localization and mapping.

Fig. 11 shows the estimated trajectories of ORB-SLAM2, DynaSLAM, and the proposed method for scenes 2–5, and the ground truths of each scene. ORB-SLAM2 drifts badly when a dynamic object passes in front of the camera, with maximum trajectory errors of 1.56 and 3.85 m in scenes 2 and 3, respectively. Even after global optimization, the maximum trajectory errors for scenes 4 and 5 are 1.12 and 2.45 m,

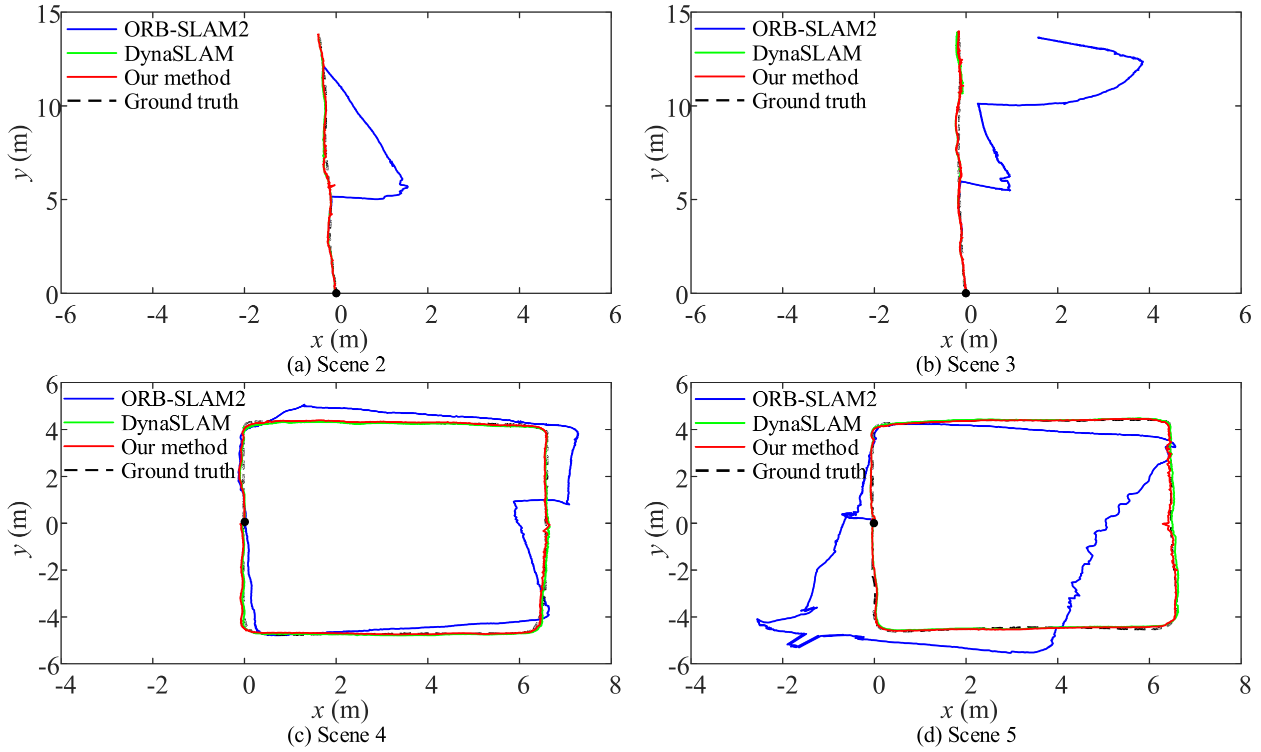


FIGURE 11. Estimated trajectories and ground truth in each scene. The coordinates of all starting points (black points) in the figure are (0, 0). Black dashed lines in the graph are ground truth. The blue, green and red lines are the estimated trajectories of ORB-SLAM2, DynaSLAM and our method, respectively.

respectively. Whereas the maximum trajectory error of our method is only 0.31 m. The estimated trajectories of the proposed algorithm in these four scenes are roughly the same as those of DynaSLAM, but those of DynaSLAM are much smoother.

TABLE 6 shows the average time spent in real scenes for the front-end part of each algorithm. Note that the runtime results of DynaSLAM which uses Mask R-CNN in the dynamic segmentation part are obtained by running on an Nvidia GeForce GTX 860M GPU. In contrast, our method only used CPU, and can reach 7.4 frames per second (fps) in real operation, basically meeting the real-time requirements of current AGV localization and mapping.

TABLE 6. Runtime for the Front End of Each Method (Unit: Ms).

	Dynamic segmentation	Tracking thread	Total time cost of front end
ORB-SLAM2	-	49.2	49.2
DynaSLAM	511.3	73.3	584.6
Proposed method	82.3	52.6	134.9

V. CONCLUSION

A SLAM algorithm based on dynamic region rejection is proposed in this paper. The dynamic feature points are filtered out using the fundamental matrix, which is computed using the feature pairs obtained by tracking the optical flow in successive frames. Then, the boundaries are labeled using the disparity plane of the superpixels, combining dynamic feature points to obtain dynamic regions. The proposed SLAM algorithm does not extract feature points in the dynamic region but only uses the information in the static region to estimate the pose, thus removing the negative effects caused by moving objects on the algorithm and improving localization and mapping accuracy. Experimental results in both the KITTI dataset and a photo set of real scenes elucidate that our dynamic area detection algorithm is both accurate and efficient. The proposed improved SLAM algorithm performs better than ORB-SLAM2 in the KITTI dataset and in real-world dynamic scenarios with higher localization accuracy. Its accuracy is comparable to DynaSLAM, but faster, capable of reaching 7.4fps. In summary, the proposed algorithm can effectively improve localization and mapping accuracy in dynamic environments and can meet the real-time demands of AGVs.

However, our method is not perfect and there are some limitations. First, our method does not perform well when dynamic objects are moving slowly or only partially in motion. Second, some constants in the algorithm need to be set relying on experience or trial and error to get better performance. In future work, we need to improve the sensitivity of our method. And design adaptive threshold constants so that the algorithm performs equally well in different situations.

REFERENCES

[1] S. Kamewaka and S. Uemura, "A magnetic guidance method for automated guided vehicles," *IEEE Trans. Magn.*, vol. 23, no. 5, pp. 2416–2418, Sep. 1987, doi: 10.1109/TMAG.1987.1065333.

[2] H.-G. Xu, M. Yang, C.-X. Wang, and R.-Q. Yang, "Magnetic sensing system design for intelligent vehicle guidance," *IEEE/ASME Trans. Mechatronics*, vol. 15, no. 4, pp. 652–656, Aug. 2010, doi: 10.1109/TMECH.2009.2029572.

[3] I. Loevsky and I. Shimshoni, "Reliable and efficient landmark-based localization for mobile robots," *Robot. Auto. Syst.*, vol. 58, no. 5, pp. 520–528, May 2010, doi: 10.1016/j.robot.2010.01.006.

[4] D. Ronzoni, R. Olmi, C. Secchi, and C. Fantuzzi, "AGV global localization using indistinguishable artificial landmarks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 287–292, doi: 10.1109/ICRA.2011.5979759.

[5] Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1403–1410, doi: 10.1109/ICCV.2003.1238654.

[6] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[7] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis.*, vol. 8690, Sep. 2014, pp. 834–849, doi: 10.1007/978-3-319-10605-2_54.

[8] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018, doi: 10.1109/TPAMI.2017.2658577.

[9] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 55–81, Jan. 2015.

[10] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual SLAM and structure from motion in dynamic environments: A survey," *ACM Comput. Surv.*, vol. 51, no. 2, p. 37, 2018, doi: 10.1145/3177853.

[11] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016, doi: 10.1109/TRO.2016.2624754.

[12] Y.-T. Wang, M.-C. Lin, and R.-C. Ju, "Visual SLAM and moving-object detection for a small-size humanoid robot," *Int. J. Adv. Robot. Syst.*, vol. 7, no. 2, p. 13, Jun. 2010, doi: 10.5772/9700.

[13] A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Oct. 2009, pp. 4306–4312, doi: 10.1109/IROS.2009.5354227.

[14] S. Li and D. Lee, "RGB-D SLAM in dynamic environments using static point weighting," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2263–2270, Oct. 2017, doi: 10.1109/LRA.2017.2724759.

[15] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular SLAM in dynamic environments," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Oct. 2013, pp. 209–218, doi: 10.1109/ISMAR.2013.6671781.

[16] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[17] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. IEEE ACM Int. Symp. Mixed Augmented Reality*, Nov. 2007, pp. 225–234, doi: 10.1109/ISMAR.2007.4538852.

[18] P. F. Alcantarilla, J. J. Yebes, J. Almazan, and L. M. Bergasa, "On combining visual SLAM and dense scene flow to increase the robustness of localization and mapping in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 1290–1297, doi: 10.1109/ICRA.2012.6224690.

[19] D. Giordano, F. Murabito, S. Palazzo, and C. Spampinato, "Superpixel-based video object segmentation using perceptual organization and location prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 4814–4822, doi: 10.1109/CVPR.2015.7299114.

[20] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robot. Auto. Syst.*, vol. 89, pp. 110–122, Mar. 2017, doi: 10.1016/j.robot.2016.11.012.

[21] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "DS-SLAM: A semantic visual SLAM towards dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Oct. 2018, pp. 1168–1174, doi: 10.1109/IROS.2018.8593691.

[22] B. Bescos, J. M. Facil, J. Civera, and J. Neira, "DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 4076–4083, Oct. 2018, doi: 10.1109/LRA.2018.2860039.

[23] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Robust semantic mapping in challenging environments," *Robotica*, vol. 38, no. 2, pp. 256–270, Feb. 2020, doi: 10.1017/S0263574719000584.

[24] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 328–341, Feb. 2008, doi: 10.1109/TPAMI.2007.1166.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[26] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis.*, vol. 3951, 2006, pp. 346–359.

[27] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, Jan. 2010, doi: 10.1109/TPAMI.2008.275.

[28] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Jt. Conf. Artif. Intell.*, Apr. 1981, pp. 121–130.

[29] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Sässtrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012, doi: 10.1109/TPAMI.2012.120.

[30] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013.

[31] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Oct. 2012, pp. 573–580, doi: 10.1109/IROS.2012.6385773.

[32] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for RGB-D cameras," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 3748–3754, doi: 10.1109/ICRA.2013.6631104.

[33] Y. Sun, M. Liu, and M. Q.-H. Meng, "Motion removal for reliable RGB-D SLAM in dynamic environments," *Robot. Auto. Syst.*, vol. 108, pp. 115–128, Oct. 2018, doi: 10.1016/j.robot.2018.07.002.

GANG LI received the B.S. degree in industrial electrical automation from Chang An University, Xi'an, China, in 1993, the M.S. degree in control theory and engineering from Guangxi University, Nanning, China, in 1996, and the Ph.D. degree in thermal power engineering from the University of Shanghai for Science and Technology, Shanghai, China, in 2009.

Since 2012, he has been a Professor with the College of Electrical Engineering, Guangxi University. He has hosted and participated in as many as 15 national or regional fund projects. He has published over 30 technical articles. Over ten patents have been granted. His research interests include control theory and control engineering, renewable energy research and utilization, detection technology and instruments, and other related research work.

XIANG LIAO received the B.S. degree in measurement and control technology and instruments from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2018. He is currently pursuing the M.S. degree in control science and engineering with Guangxi University, Nanning, China.

HUILAN HUANG received the M.S. degree in mechanical engineering from Guangxi University, Nanning, China, in 1994, and the Ph.D. degree in thermal power engineering from the University of Shanghai for Science and Technology, Shanghai, China, in 2009.

Since 2012, she has been a Professor with the College of Mechanical Engineering, Guangxi University. She has hosted and participated in as many as ten national or regional fund projects. She has published over 40 technical articles. Over ten patents have been granted. Her research interests include photo-voltaic photo-thermal power generation technology, energy utilization system and evaluation, energy saving technology of thermal equipment, and other related research work.

SHAOJIAN SONG (Member, IEEE) received the B.S. and M.S. degrees from Guangxi University, Nanning, China, in 1994 and 2001 respectively. Since 1994, he has been with the College of Electrical Engineering, Guangxi University, where he became a Professor, in 2010. He visited the New York State Center for Future Energy Systems, Rensselaer Polytechnic Institute, USA, from 2014 to 2015. His current research interests include power electronics and energy conversion, active distribution networks, state estimation, optimal control, and machine learning.

BIN LIU received the Ph.D. degree from the School of Information Science and Engineering, Central South University, Changsha, China, in 2014.

Since 2015, he has been a Lecturer with the College of Electrical Engineering, Guangxi University. His research interests include power electronics and energy conversion, with particular emphasis on converter topologies, modeling and control.

YAWEN ZENG received the B.S. degree in measurement and control technology and instruments from North China Electric Power University, Beijing, China, in 2017. She is currently pursuing the M.S. degree in control science and engineering with Guangxi University, Nanning, China.