Computer Fraud and Security
ISSN (online): 1873-7056

Visual Localization and Dense Map Construction in Dynamic Scenes

Huipeng Li[1], Zhaolan He[1,*], Kadan Xie[1], Cong Xue[2]

[1]*College of Mechanical and Electrical Engineering, Guangdong University of Science and Technology, Dongguan 523083, Guangdong, China*

[2]*Department of Control Science and Engineering, Harbin Institute of Technology, Harbin 150001, Heilongjiang, China*

**Corresponding Author.*

Abstract:
The key technology for autonomous navigation of mobile devices such as robots lies in Simultaneous Localization and Mapping (SLAM) based on vision, which has become increasingly sophisticated in idealized static environments. However, in dynamic scenarios, existing instance semantic based visual SLAM methods cause over segmentation during the dynamic segmentation process, while deep learning methods increase system runtime. Therefore, this article proposes a visual SLAM scheme in dynamic scenes. The method of combining fractal dimension and epipolar constraint is used to detect dynamic regions in images, remove all feature points in the corresponding dynamic regions, construct a backend model in dynamic scenes, and achieve a more comprehensive dense point cloud map. This enables the visual SLAM system to accurately complete navigation, positioning, and obstacle avoidance functions in real time.

Keywords: robot dexterous operation, harmony search algorithm, genetic algorithm

INTRODUCTION

Nowadays, mobile devices such as robots have gradually entered our lives, making them more convenient and efficient. But it will be affected by the lack of GPS or weak GPS signals, especially in dynamic scenarios, which will affect the real-time positioning, path planning, and map construction of mobile devices, resulting in the inability to complete their tasks. Therefore, real-time positioning and map construction (SLM) are key technologies for autonomous mobile devices [1].

After Smith et al. first proposed the concept of SLAM [2], it aroused the research interest of industry scholars and began to develop rapidly, Subsequently, the research progress of vision SLAM methods has been particularly rapid [3-5]. Especially in complex environments such as tunnels, underground, and mines, autonomous robots are used for search, rescue, surveying, and other tasks[6]. The visual SLAM system only obtains the camera's motion trajectory through geometric constraint principles in a static, rigid environment. But the system performance will be affected or even fail in dynamic scenarios. Kundu et al. used multi view geometric constraints to segment dynamic objects and proposed a feature point based multi motion object visual SLAM method in dynamic scenes [7]. Kim proposed the BaMVO algorithm, which applies RGB-D sensors in dynamic scenes to estimate background models from deep scenes and obtain camera motion trajectories [8].

To compensate for the shortcomings of geometric segmentation methods, many scholars have applied deep learning methods to visual SLAM and achieved many results[9-11]. This method can better assist visual SLAM systems in understanding and processing more complex dynamic scenes. An SLAM system based on PSPNet network[12], which uses a combination of optic flow method and semantic segmentation to remove dynamic feature in the scene. Li et al. combined geometric constraints with semantic segmentation and proposed a visual SLAM based on sparse eigenvalue (DP-SLAM) [13], which further improved the high fidelity SLAM for 3D reconstruction [14]. Another dense RGB SLAM system is used to generate high fidelity 3D dense reconstruction. This system can simultaneously optimize camera pose and hierarchical neural implicit mapping representation [15]. NEDS-SLAM utilizes the importance of semantic information under the positioning and mapping process of Gaussian points, and obtains semantic information by distilling the 2D segmentation information provided by the dataset[16]. CG-SLAM introduces uncertainty maps during training and optimizes 3D reconstruction accuracy based on rendering depth[17].

Due to the use of prior information, deep learning based methods may not necessarily detect all objects in motion, resulting in over segmentation and insufficient quantity of feature introduced to the backend. In addition, using

deep learning methods to segment motion regions requires a huge amount of data during the training process, which greatly increases the time required for system operation.

In summary, the visual SLAM system in dynamic scenes still faces issues such as low accuracy in dynamic region segmentation, inadequate real-time performance, and backend optimization in dynamic scenarios. This article first uses a dynamic region segmentation method based on fractal dimension and epipolar constraints to effectively removing the impact of moving objects in dynamic scenes on the front-end and back-end of visual SLAM systems. Secondly, various parts of the visual SLAM system should be improved and perfected based on the characteristics of dynamic environments, so that visual SLAM can better complete its specific tasks in dynamic scenes.

VISUAL SLAM MATHEMATICAL MODEL

The visual SLAM system uses a camera as a sensor to receive external information, and the motion of the entire system can be regarded as the rigid body motion generated by the camera in three-dimensional space, that is, keeping the inner product and metric unchanged. To describe the location data of the camera in the scene, simply convert the camera coordinates to the corresponding world coordinates through Euclidean transformation.

Therefore, to obtain the camera's pose in the scene, one only needs to transform the corresponding camera pose into the corresponding world pose, and the mutual transformation between the camera and the world coordinate system is expressed in Figure 1.

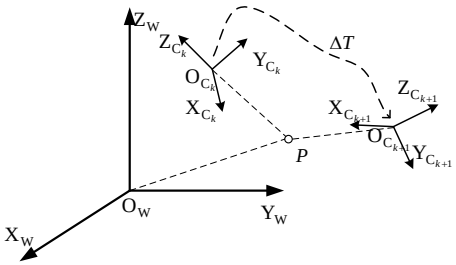


Figure 1. Coordinate system transformation

The three-dimensional coordinate system with W as the base in the Figure 1 is the coordinate system of world, while the three-dimensional coordinate system with C as the base is the coordinate system of camera. There is a point in the three-dimensional space corresponding to the coordinate system of world, which is a landmark set to mark the information that needs to be preserved in the scene where the camera is located. During the movement of the camera, the observed landmark information is matched one-to-one with the timeline to obtain a series of data information. Assuming that at time, the camera obtains information about the landmark through observation, and the corresponding coordinate in the camera coordinate system is $\left({}^{C_k}p_x, {}^{C_k}p_y, {}^{C_k}p_z\right)$, and at the next time, that is, time, it is represented as $\left({}^{C_{k+1}}p_x, {}^{C_{k+1}}p_y, {}^{C_{k+1}}p_z\right)$ in the camera coordinate system and $\left({}^{W}p_x, {}^{W}p_y, {}^{W}p_z\right)$ in the corresponding world coordinate system. So the camera pose at the corresponding time can be calculated by expressing the point in different coordinate systems, and the corresponding transformation relationship is.

$$\begin{bmatrix} {}^{W}P_x \\ {}^{W}P_y \\ {}^{W}P_z \\ 1 \end{bmatrix} = T_k^{-1} \begin{bmatrix} {}^{C_k}P_x \\ {}^{C_k}P_y \\ {}^{C_k}P_z \\ 1 \end{bmatrix} = T_{k+1}^{-1} \begin{bmatrix} {}^{C_{k+1}}P_x \\ {}^{C_{k+1}}P_y \\ {}^{C_{k+1}}P_z \\ 1 \end{bmatrix} \tag{1}$$

In the formula: $\boldsymbol{T}_k$ represents the Euclidean transformation matrix of landmark points in different coordinate systems at time k, and $\boldsymbol{T}_{k+1}$ corresponds to time $k+1$.

The mathematical modeling of camera motion is described using Lie algebra. The Lie algebra is shown in equation (2).

$$se(3)=\left\{\xi=\begin{bmatrix}\rho\\ \phi\end{bmatrix}\in\mathbb{R}^6,\phi\in so(3),\xi^{\wedge}=\begin{bmatrix}\phi^{\wedge} & \rho\\ \mathbf{0}^{\mathrm{T}} & 0\end{bmatrix}\in\mathbb{R}^{4\times4}\right\} \tag{2}$$

In the formula, the six dimensional vector ξ represents the random elements in $se(3)$. The translation vector in the element is represented by ρ, and the rotation vector is represented by ϕ.

The formula for calculating ξ is:

$$\xi^{T}=\begin{bmatrix}\boldsymbol{\phi}^{T} & \boldsymbol{\rho}^{T}\end{bmatrix}^{T}=\begin{bmatrix}(\ln(\boldsymbol{R})^{\vee})^{T} & (J^{-1}t)^{T}\end{bmatrix}^{T} \tag{3}$$

J is the Jacobian matrix

$$J=\frac{\sin\theta}{\theta}I+(1-\frac{\sin\theta}{\theta})aa^{\mathrm{T}}+\frac{1-\cos\theta}{\theta}a^{\wedge} \tag{4}$$

Among them, θ and a respectively represent the module length and unit vector of ϕ.

When the visual SLAM system obtains the pose information of the camera at a certain moment, the next pose information of the camera can be obtained based on the scene data obtained by the system through the camera. If T_k represents the pose information at a certain moment, P_j represents the landmark point, and $z_{k,j}$ represents the scene data, the corresponding camera observation equation is

$$z_{k,j}=T_kP_j+v_{k,j} \tag{5}$$

Here, $v_{k,j}$ is the external noise introduced by the sensor when acquiring image information, and its influence can generate errors in the system's estimation of the camera's current positioning using the acquired data. At time k, the camera obtains N waypoint position information while traveling. To reduce the error between the current positioning estimation data of the camera and the real data, it is necessary to construct the optimal pose matrix T_k, whose corresponding expression is.

$$\{T\}=\arg\min\sum_{i=1}^{N}\left\|z_{k,j}-(T_kP_j)\right\|_2^2 \tag{6}$$

The pose estimation problem of the camera can be transformed into an majorization problem of minimizing the pose matrix error, and equation (6) is the objective function of this optimization problem. According to the formula BCH(Baker-Compbell-Hausdorff), the complete form of multiplying two matrices into a Lie algebraic exponential mapping can be obtained, with the specific form being

$$\begin{aligned}\ln(T_1\bullet T_2)^{\vee}&=\ln(\exp(\xi_1^{\wedge})\exp(\xi_2^{\wedge}))^{\vee}=\ln(\exp(\xi_1+\xi_2)^{\wedge})^{\vee}=\\&(\Xi_1+\Xi_2+\frac{1}{2}[\Xi_1,\Xi_2]+\frac{1}{12}[\Xi_1,[\Xi_1,\Xi_2]]-\frac{1}{12}[\Xi_2,[\Xi_1,\Xi_2]]\ldots)^{\vee}\end{aligned} \tag{7}$$

In the formula, $\Xi_1=\xi_1^{\wedge}$, $\Xi_2=\xi_2^{\wedge}$ and $[\ ]$ are Li brackets. When the value of ξ_1 reaches its minimum, the above formula can be approximately expressed as equation (8)

$$\ln(\exp(\xi_1^{\wedge})\exp(\xi_2^{\wedge}))^{\vee}=\boldsymbol{J}_l(\xi_2)^{-1}\xi_1+\xi_2 \tag{8}$$

In the formula, $\boldsymbol{J}_l$ represents the left Jacobian matrix, and the specific calculation method is.

$$J_l=J=\frac{\sin\theta}{\theta}I+(1-\frac{\sin\theta}{\theta})aa^{\mathrm{T}}+\frac{1-\cos\theta}{\theta}a^{\wedge} \tag{9}$$

From the above content, it can be seen that estimating the correspondence between camera coordinates and visual coordinates at various times through external information data obtained by the camera is the solving process of the visual SLAM system, further obtain a globally consistent 3D map. Thus, a visual SLAM mathematical model based on Lie algebra can be obtained, as shown in equation (10).

$$\begin{cases} \exp(\hat{\xi}_{k+1}) = f(\exp(\hat{\xi}_k), u_k) + w_k \\ z_{k,j} = h({}^{w}P_j, \exp(\hat{\xi}_k)) + v_{k,j} \end{cases} \tag{10}$$

In the formula: $\exp(\hat{\xi}_k)$ is the camera positioning at time k, and $z_{k,j}$ is the observation data obtained by the camera from landmark ${}^{w}P_j$ at position x_k. w_j and $v_{k,j}$ are both noise

DYNAMIC FEATURE DETECTION

Motion Detection Based on Polar Constraints

TBased on multi view hierarchical constraint method, the motion state of feature in the image is determined based on the sum of the distances from the projection points in adjacent frames to the corresponding epipolar lines, and dynamic feature points are detected accordingly. The concrete step is:

Image feature point preprocessing

Before using level constraints for motion object detection, ensure that the feature obtained by the camera in the image are stable. Considering the characteristics of the camera itself, the feature at the edges of the entire image may be distorted. Therefore, it is necessary to first remove the pixels at the edges of the image before performing calculations. Remove 5 pixels from the edge of the picture to ensure that the calculated feature points are stable.

Detection of dynamic feature points

Using an algorithm to screen and select feature points from adjacent frames of images to construct the required basic matrix $\boldsymbol{F}$. Based on the basic matrix $\boldsymbol{F}$, solve for the corresponding epipolar lines of the feature in adjacent frames of the picture, and obtain the distances d and d' between the projection points and their parallel epipolar lines on the adjacent frames of the image, respectively, in order to obtain the total error D and use it to determine the motion properties of the feature points in the image. The state of a feature point is determined by the relationship between the total error D and the system threshold τ. If the total error D of a feature point exceeds threshold τ after calculation, it is labeled as a dynamic point, otherwise it is labeled as a static point. After repeatedly comparing the results of different thresholds during the experiment, it was found that the optimal experimental effect was achieved when the system threshold was $\tau = 2$.

Dynamic Region Judgment and Removal

To rightly estimate the camera's positioning in dynamic scenes using feature point based visual SLAM algorithm, it is needful to ensure that the feature points transmitted to the backend are all static and stable. Therefore, the feature points in the dynamic range of the scene need to be removed to ensure that the system performs pose estimation more stably and robustly.

The image segmentation method based on box dimension can only obtain the contour information of the target object in the image and cannot determine the current motion properties of the object. Therefore, it is necessary to introduce a multi view polar constraint method to determine the motion state of feature in the image. This article combines box dimension based image segmentation methods with polar constraint methods to get more accurate result in detecting dynamic objects of picture. Figure 2 is the process of determining and removing the movement area.

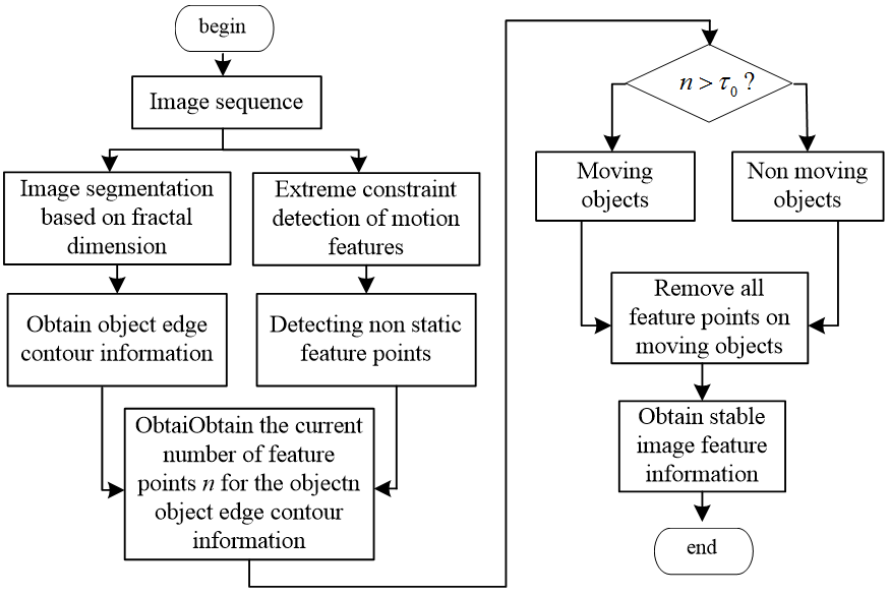


Figure 2. Judgment and removal process of motion area

In the process of detecting moving objects, if the quantity of dynamic feature points τ_0 is one-third of the total quantity of feature points of the object, that is, when the quantity of dynamic feature points detected on a certain area in the image exceeds one-third of the total quantity of feature points in that area, the area is considered a dynamic area and all feature on the corresponding area are removed. After removing the dynamic feature, the feature matching effect is shown in Figure 3.

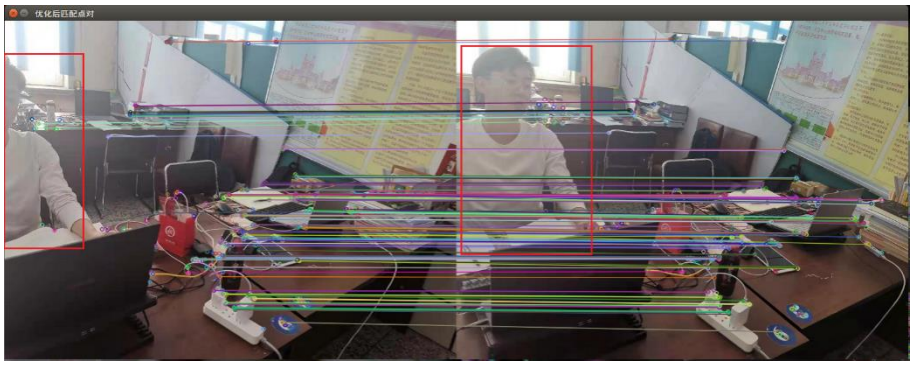


Figure 3. Feature matching effect after removing motion features

VISUAL SLAM BASED ON MOTION REGION SEGMENTATION

Front End Visual Odometry

The visual SLAM system performs position estimation on itself, which is essentially the process of matching adjacent frame image data obtained by the camera in three-dimensional space. As this article focuses on a visual SLAM system based on the feature point method, it is needful to extract feature from the collected images during the front-end processing and record these feature points as the system's landmark information. When moving objects appear in the scene, sensors will collect corresponding dynamic features, causing the visual odometer to be unable to accurately match the corresponding point cloud information. In order to decrease or even remove the influence of dynamic objects in the scene on the visual odometer of the visual SLAM front-end, this article mainly implements the function of detecting and eliminating objects in moving in the front-end visual odometer of the system. The image segmentation method based on fractal dimension is used to achieve the function of image region segmentation, and the polar constraint method is used to achieve the function of detecting motion features. Finally, the function of judging and removing motion regions is achieved through probability matching. The specific structure of the improved visual odometer is shown in Figure 4.

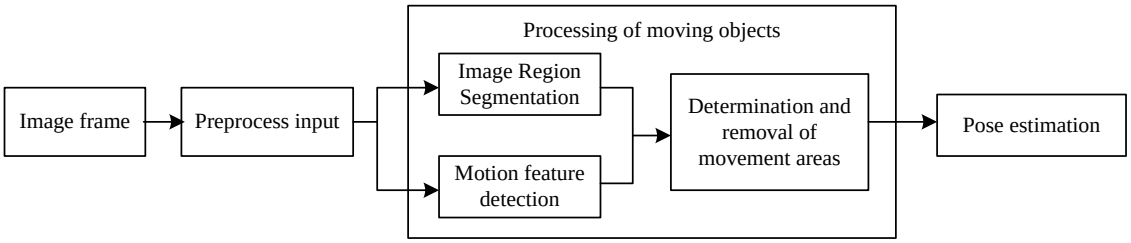


Figure 4. Improved block diagram of visual odometer

Backend Model in Dynamic Environment

Loop detection

Implement the loop detection mechanism of the system based on the word bag model. In order to ensure the correctness of loop detection and further diminish the mathematics complexity of the system, the system only performs loop filtering on keyframes, and adds stricter loop detection conditions based on the traditional word bag model to ensure the quality of candidate frames obtained by the system after screening. The specific conditions are as follows.

①Effective constraint on pixel ratio:

After motion segmentation of the picture captured by the camera, if the remaining static parts do not account for the expected proportion in the entire image, it is considered that the current frame contains less image information and cannot effectively detect the similarity between the two frames. Therefore, such keyframes are not suitable as candidate frames to be added to the bag of words model.

②Image entropy ratio detection:

In order to increase filtering efficiency, when performing image similarity detection, the average image entropy of all normal frames between the current critical frame and the previous critical frame is divided by the image entropy of candidate frames in the bag of words. If the division is lower than the set threshold, it is considered that the similarity between these two frames is low and the current candidate frame is excluded.

③Matching performance between current keyframes and candidate frames:

When comparing the similarity between keyframes and candidate frames, the ORB algorithm is used for camera pose estimation. If the number of available constrained pixels in the detected image frame is less than a certain number, it will result in too few effective pixels in the top-level pyramid image, reducing system performance. Therefore, when performing similarity detection, if the available pixels in the detected image are too low, the candidate frame should also be excluded.

④Cross validation between current keyframe and candidate frame:

After forward solving, it has been determined that the current keyframe and candidate frame form a loop, and a reverse solving, also known as cross validation, is required to match and compare the current critical frame with the auxiliary frame. If the difference between the cross validation results exceeds a certain threshold during comparison, it is considered that the current loop result is unreliable and the corresponding candidate frames are excluded.

If there are candidate frames that pass the screening of the above four conditions, the system will consider that the camera has gone back to its original location and the loop detection is successful. The system will correct the original trajectory based on the estimated camera position of the current frame image. Pass the corresponding loop information into the backend for global consistency comparison, and use this to ensure that the system is not affected by cumulative errors and undetected dynamic

Construction and optimization of pose maps

Combining the camera position information got by the ICP algorithm in the visual odometer with the current data obtained by the loop detection algorithm to obtain more accurate local trajectory information.

During the movement of the camera, the front-end visual odometer of the system will estimate the pose information corresponding to the camera at different times and retain the scene information obtained by the camera. In order to better correlate the image data obtained by the camera at different times, graph theory is introduced to construct a graph optimization model for visual SLAM. The position nodes of the camera itself and the landmark points obtained by the camera and stored in the backend are regarded as the vertices of the graph, and the geometric constraint relationships between each point are regarded as the edges of the graph, as shown in Figure 5.

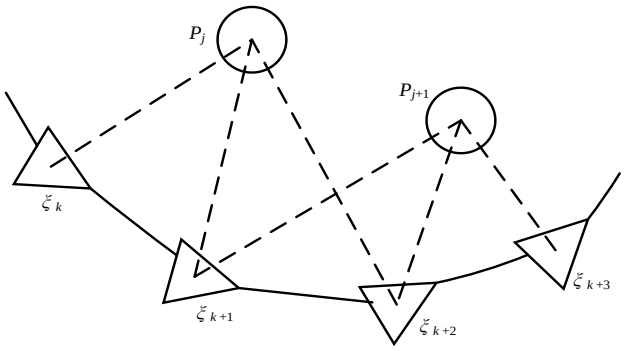


Figure 5. Graph model of pose

In the above Figure 5, the position points of the camera itself and the observed landmark points are represented by triangles and circles, respectively. The solid lines connecting the triangles representing pose nodes represent the motion constraint model of the camera itself, which is expressed by the formula, as shown in equation (11).

$$\exp(\hat{\xi_{k+1}}) = f(\exp(\hat{\xi_k}), u_k) + w_k \tag{11}$$

The dashed line formed by the triangle representing the pose node and the circle representing the landmark point represents the observation model of the camera in three-dimensional space, and the corresponding formula is shown in equation (12).

$$z_{k,j} = h({}^{w}P_j, \exp(\hat{\xi_k})) + v_{k,j} \tag{12}$$

From time 1 to time N, the camera recorded a total of M waypoints during the journey. And under these conditions, the conditional probability distribution of the camera's own pose and its observed waypoints is obtained, and the corresponding visual SLAM backend optimization model is shown in equation (13).

$$P(x, y \mid z, u) \tag{13}$$

In the formula, the set of camera poses obtained from time 1 to time N is represented by x, and the set of all waypoints obtained by the corresponding sensors is represented by y.

The scene information obtained by the camera exists independently. Therefore, it can be inferred that the maximum likelihood estimation equation for the camera's own state variables is:

$$\begin{aligned} &\min F(x,y) = \arg\max P(z|x,y) = \arg\max \prod_{k,j} P(z_{j,k}|x_k, y_j) \\ &= \arg\min \sum_k \sum_j \left((z_{k,j} - h(y_j, x_k))^T \boldsymbol{Q}_{j,k}^{-1} (z_{k,j} - h(y_j, x_k)) \right) \end{aligned} \tag{14}$$

Here, $Q_{j,k}^{-1}$ represents the error information matrix. The optimal solution obtained from equation (14) is its maximum likelihood estimate. In order to describe it more conveniently, x、y is defined to describe all the state variables to be optimized, as shown in equation (15).

$$x = [x, y] = [\xi_1, \xi_2, \xi_3, \ldots\ldots \xi_N, {}^{w}P_1, {}^{w}P_2, {}^{w}P_3, \ldots\ldots {}^{w}P_M] \tag{15}$$

Rewrite the optimization objective function as shown in equation (16).

$$\min F(\boldsymbol{x}) = \frac{1}{2}\sum_{k=0}^{N}\sum_{j=0}^{M}(e_{k,j}(\boldsymbol{x}))^{T}\boldsymbol{Q}_{j,k}^{-1}(e_{k,j}(x)) \tag{16}$$

CONSTRUCTION OF DENSE POINT CLOUD MAP

The key frame images determined in the visual SLAM odometer are used as the information source for map construction. RGB-D cameras are used to obtain color and depth information of pixel points from these key frame images and generate corresponding point cloud information, thereby stitching together a dense point cloud map of the scene where the camera is located. Using PCL (Point Cloud Library) for map construction, the camera inevitably introduces external noise into the system during the mapping process. Therefore, it is needed to filter and downsample the generated point cloud information.

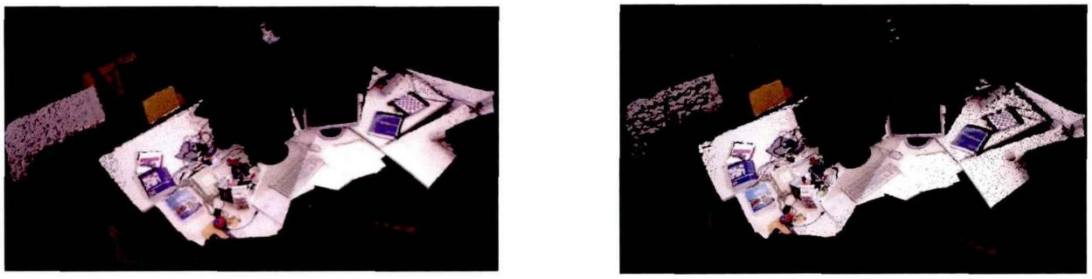
Point cloud filtering

When the visual SLAM system obtains external information through the camera, noise is inevitably introduced, which will affect the quality of the system's mapping. Therefore, when constructing a map, the visual SLAM system must filter the obtained point cloud information to remove the influence of noise in the information, thereby ensuring the effectiveness of the mapping. Here, statistical filters from the PCL library are used to filter the collected point cloud information.

Reduce sampling

When the visual SLAM system uses sensors to collect external scenes, due to the short interval between camera shots, the same information will be transmitted multiple times to the system for processing. When constructing dense maps, these repeated data will also be converted into corresponding point cloud information multiple times, causing information redundancy. This will reduce the efficiency of the system's mapping, and the generated map files will be too large to save. Therefore, this article adopts voxel filtering to ensure that only a certain number of data points are used for dense map construction within a certain spatial range.

The point cloud map generated by RGB-D camera, after filtering and downsampling, is compared in Figure 6.



(a)Original image (b)The image after filtering and downsampling

Figure 6. Point cloud map constructed by fusion of RGB-D image point clouds

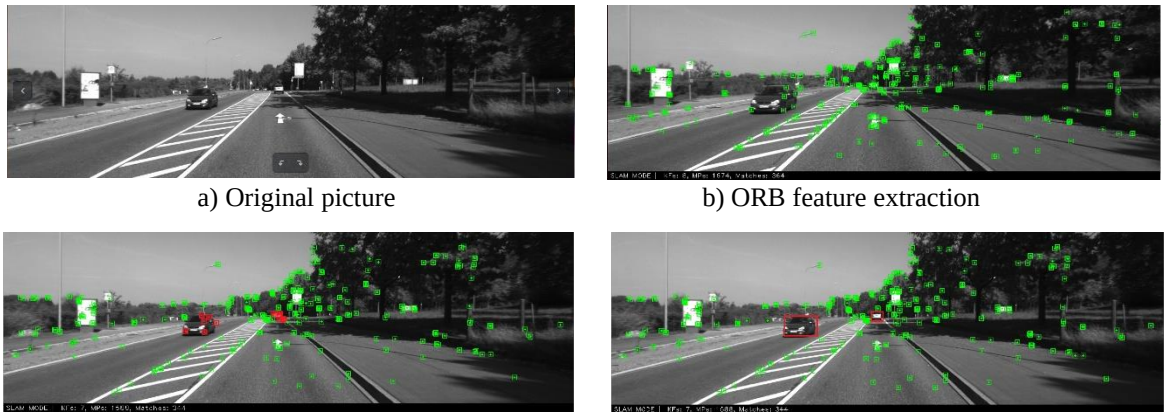
From Figure (6), it can be intuitively seen that after the above processing of the dense point cloud map, the noise and redundant information in the map have been improved, effectively saving system resources and making the size of the dense map file meet the system's expectations.

EXPERIMENTAL RESULTS AND ANALYSIS

Use the KITTI dataset to perform performance testing on the improved algorithm. The testing consists of two parts. Firstly, the dynamic region segmentation effect constructed in this article is verified; Then verify the overall validity of the improved visual SLAM algorithm in dynamic scenes.

Dynamic Region Elimination Test

The KITTI dataset used in this paper mainly contains moving objects such as cars. Figure 7 is the algorithm effect.



a) Original picture b) ORB feature extraction

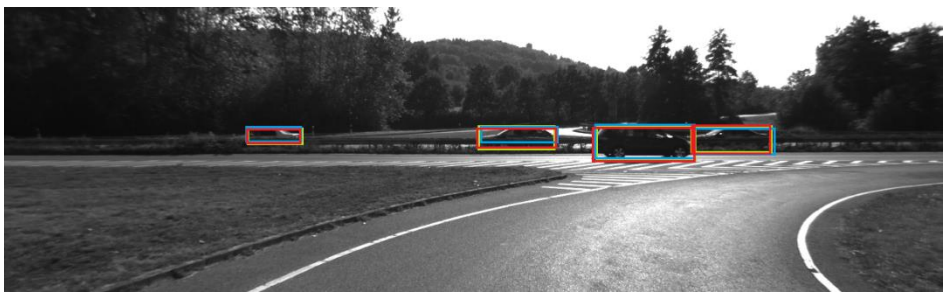
c) Dynamic feature point detection d) Dynamic Region Segmentation

Figure 7. The experimental effect of dynamic region segmentation under the KITTI data set

From Figure (7), it can be seen that if a dynamic target appears in the image field of view, the improved algorithm can fully frame the dynamic target area in real-time in the field of view.

For further comparative testing, an open-source algorithm was used as the control group. Compared with algorithms, the dynamic object detection and removal method in this paper does not require prior information, which ensures that the algorithm in this paper will not mistakenly detect unmoving movable objects as dynamic objects. Moreover, when non prior model dynamic objects emerge in the scene, the improved algorithm can also select them. The effect comparison is shown in Figure 8.

The green box in Figure (8) represents the area where the real dynamic object is located in the image, the blue box represents the dynamic object area selected by the algorithm, and the red box represents the dynamic object area selected by the algorithm in this paper. When there is no static vehicle in Figure 8a) but there is partial occlusion of the vehicle, the DynaSLAM algorithm can only recognize some dynamic objects. However, in Figure 8b), when there is a static vehicle, the DynaSLAM algorithm uses prior information for recognition, which also removes the static vehicle as a dynamic object, resulting in excessive deletion of the feature passed in by the front-end and deteriorating the robustness of the system. The evaluation of the dynamic region segmentation results is shown in Table 1.



a) No static vehicles

b) There are static vehicles present

Figure 8. Comparison chart of dynamic region segmentation effect

Table 1. Evaluation of dynamic region segmentation results

	Precision P/%	Recall rate R/%	Average accuracy S/%
DynaSLAM	61.97	94.06	83.25
This article's algorithm	90.32	93.17	85.73

According to the data in table 1, the DynaSLAM algorithm will misjudge static vehicles and pedestrians as dynamic areas, resulting in poor accuracy result of the algorithm. Compared to this, the improved algorithm can more accurately and completely detect dynamic areas.

Due to the fact that the DynaSLAM algorithm uses Mask R-CNN, a deep learning method, to recognize dynamic objects in images, its time complexity is relatively high. In this paper, multiple sets of front and back frame images from the KITTI dataset were used to test the running time of our algorithm and DynaSLAM algorithm, and the running time of ORB-SLAM2 algorithm was added as a basis for judging the real-time of the system. Table 2 is the test data.

Table 2. Time analysis results of dynamic region segmentation algorithm

	feature extraction	Parallax calculation	Region segmentation	Motion detection
ORB-SLAM2	9.31ms	1.62ms	N/A	N/A
DynaSLAM	9.43ms	1.63ms	2769.35ms	36.74ms
This article's algorithm	9.28ms	1.66ms	76.13ms	34.32ms

From the data in table 2, it can be seen that the algorithm proposed in this paper significantly reduces the time required for region segmentation compared to the DynaSLAM algorithm. Although it increases the time required for dynamic region segmentation and detection compared to the ORB-SLAM2 algorithm, it still meets the real-time requirements of the SLAM system well.

Visual SLAM Testing in Dynamic Scenarios

The KITTI dataset is still used for testing, with ORB-SLAM2 as the control group. As a widely recognized visual SLAM scheme by industry scholars, this method has excellent system compatibility and robustness. However, the DynaSLAM algorithm only optimized the front-end VO of the visual SLAM system and did not optimize the back-end of the system, so it was not used as the control group.

The trajectory map of the system was repeatedly tested in dynamic scenarios to ensure the correctness and robustness of the improved algorithm. Multiple sets of comparative experimental data are shown in Figure 9, where black, red, and blue trajectories represent the actual trajectory map in the scene, the trajectory map estimated by our algorithm, and the trajectory map estimated by ORB-SLAM2 algorithm, respectively.

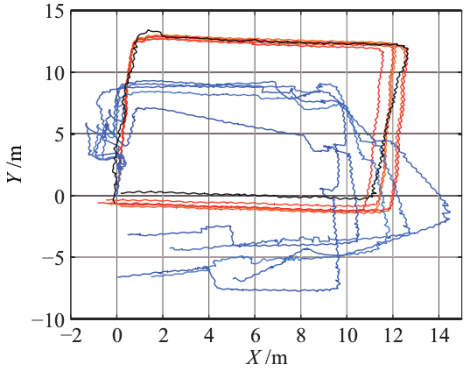


Figure 9. Comparison of positioning trajectories of repeated experiments

Figure (9) shows that due to the ORB-SLAM2 not removing the feature in the dynamic area, its localization results are severely offset, and the multiple trajectory results are almost non overlapping; After removing the feature in the dynamic area, the algorithm in this article provides more accurate and reliable trajectory information,

which is basically consistent with the real trajectory. To better ensure the performance of the algorithm in this article, absolute trajectory error (ATE) is used as the accuracy indicator for system trajectory positioning. The formula for calculating absolute trajectory error is:

$$\mathrm{ATE}_{\mathrm{all}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\|\log(T_{gt,i}^{-1}T_{esti,i})^{\vee}\right\|_2^2} \tag{17}$$

In the formula(17), $T_{gt,i}$ and $T_{\mathrm{esti},i}$ represent the real trajectory and the calculation trajectory of the system, respectively, and the root-mean-square error of the camera's own pose at different times is the absolute trajectory error. Table 3 shows the calculation results.

Table 3. Comparison of path positioning trajectory errors

	In a static scene ORB-SLAM2(ATE)	In dynamic scenarios ORB-SLAM2(ATE)	Algorithm proposed in this article under dynamic scenarios(ATE)
straight path	0.0749	1.8352	0.5731
Rectangular path	0.1033	4.3758	0.7596

The data in the table 3 shows that the improved algorithm has significantly improved the positioning correctness compared to the ORB-SLAM2 in dynamic scenes. Although it cannot achieve the same effect as in static environments, it can still make the visual SLAM system have better accuracy and robustness in dynamic scenes.

CONCLUSION

Visual SLAM is the key and foundation for the implementation of unmanned driving and 3D reconstruction technologies. Dynamic objects in daily life scenes are inevitable, so this paper proposes a visual SLAM scheme in dynamic scenes. The method of combining fractal dimension and epipolar constraint is used to detect dynamic regions in images, and all feature in the corresponding dynamic regions are removed to ensure that the feature transmitted to the backend are stable, providing a guarantee for the accuracy of backend calculations. The backend model constructed in dynamic scenes effectively improves the drift of backend pose estimation caused by cumulative errors and dynamic objects in the image, resulting in superior performance of the system in dynamic scenes. The dense point cloud map constructed has more comprehensive functions. Experimental verification was conducted using the KITTI dataset, and the test results showed that the proposed visual SLAM scheme for detecting dynamic regions in dynamic scenes has significantly improved the effectiveness and robustness of the system in dynamic scenes, and better meets the real-time requirements of the visual SLAM system in terms of time consumption.

ACKNOWLEDGMENTS

This research was supported by National Natural Science Foundation of China funding project(62173107, U23A20346); Guangdong University of Science and Technology Doctoral Initiation Fund Project (GKY-2023BSQD-50).

REFRENCES

[1] Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics, 2016, 32(6): 1309-1332.

[2] Smith R C, Cheeseman P. On the representation and estimation of spatial uncertainty. The International Journal of Robotics Research, 1986, 5(4): 56-68.

[3] Dissanayake M G, Newman P, Clark S, et al. A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotics and Automation, 2001, 17(3): 229-241.

[4] Fuentes-Pacheco J, Ruiz-Ascencio J, Rendón-Mancha J M. Visual simultaneous localization and mapping: a survey. Artificial Intelligence Review, 2015, 43(1): 55-81.

[5] Steve M, Ivona J. SLAM Toolbox: SLAM for the dynamic world. Journal of Open Source Software, 2021, 6(61), 2783.

[6] Kamak E, Lukas B, Harel B. present and Future of SLAM in Extreme Environments: The DARPA SubT Challenge. IEEE Transactions on Robotics, 2023, 40(10):936 - 959

[7] Kundu A, Krishna K M, Jawahar C V. Realtime multibody visual SLAM with a smoothly moving monocular camera. International Conference on Computer Vision (ICCV), IEEE, 2011: 2080-2087

[8] Kim D H, Kim J H. Effective background model-based RGB-D dense visual odometry in a dynamic environment. IEEE Transactions on Robotics, 2017, 32(6): 1565-1573.

[9] Ranftlr, Ladingerk, Hafnerd, et al. Towardsrobust monocular depth estimation: mixing datasets for zero-shotcross-dataset transfer. IEEE Transactionson Pattern Analysis and Machine Intelligence, 2022, 44(3):1623-1637.

[10] Qu H, Yang Z, Zhang H, et al. Areview of SLAM based on deep learning image depth perception. Navigation Positioning and Timing, 2024, 11(6):11-27

[11] Keb, Obukhov A, Huangs, et al. Repurposing diffusion-base dimage generators for monocular depthestimation. Proceedings of IEEE/CVF International Conferenceon Computer Vision and Pattern Recognition. Seattle: IEEE, 2024

[12] Han S, Xi Z. Dynamic Scene Semantics SLAM Based on Semantic Segmentation. IEEE Access, 2020, 8:43563-43570

[13] Li A, Wang J, Xu M, et al. DP-SLAM: A visual SLAM with moving probability towards dynamic environments. Information Sciences, 2021, 556(5):128-142

[14] Nikhil K, Jay K, Krishna M, et al. SplaTAM: Splat Track & Map 3D Gaussians for Dense RGB-D SLAM. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, 21357-21366

[15] Zhu Z, Peng S, Larsson V, et al. NICER-SLAM: Neural Implicit Scene Encoding for RGB SLAM. 2024 International Conference on 3D Vision, 2024

[16] Ji Y, Liu Y, Xie G, et al. NEDS-SLAM: anovel neural explicit dense semantic SLAM framework using 3D Gaussians platting. IEEE Roboticsand Automation Letters, 2024, 9(10):8778-8785.

[17] Hu j, CHen X, Feng B, et al. CG-SLAM: efficient dense RGB-D SLAM in a consistent uncertainty-aware 3D Gaussian field. Proceedings of European Conferenceon Computer Vision. Milan: Springer, 2024:93-112