IMPROVEMENT OF INDOOR POSITIONING TECHNOLOGY BASED ON SLAM

This project is dedicated to improving the application of SLAM in the field of indoor positioning. In order to make the algorithm as universal as possible, I choose the monocular SLAM, with advantages of low cost and no need for special infrastructure. Currently, mobile phones with cameras are very common, and the quality of mobile phone cameras is constantly improving, making this technology more conducive to the promotion of SLAM running on mobile phones or other edge ends, providing pedestrians with indoor positioning and navigation services.

This paper firstly reviews the existing classic system of monocular SLAM, including the feature-based method and the direct method. After careful analysis, ORB-SLAM is selected as the basic system for this design. For the reason that ORB-SLAM has the characteristics of high positioning accuracy, good robustness, real-time processing, running well on large scale situation and exactly suitable for indoor environment. And it is based on the feature method which constructs sparse map.

The focus of this research is to solve the problem of filtering the dynamic feature points of SLAM. For the specific scene of indoor positioning, dynamic objects such as pedestrians often appear, especially in some crowded places such as shopping malls and indoor city squares. SLAM has a hypothetical premise that all landmarks are considered to be static points, and pose estimation in SLAM is also based on the common-view relationship of static points. Therefore, the appearance of dynamic objects may cause SLAM to have errors in feature matching and previous tracking feature may be occluded or there are not enough matching points with good common-view conditions, which may lead to a decrease in the overall accuracy and robustness of SLAM. Therefore, the removal of non-rigid contexts is very necessary.

Existing methods generally have two solutions to the problem of dynamic content removal. In the early years of SLAM, the method of using geometric relationships to analyze abnormal points was used, and the dynamic points were filtered out as " outliers", but this method has large limitations. It works poor in situations with large-area dynamic objects, and may cause tracking fail. Others are generally based on RGB-D SLAM, which can get real depth of points easily. Since monocular cameras can’t get real depth of points, this kind of methods are not suitable. In recent years, many dynamic SLAM systems have used methods of combining deep semantic segmentation models to remove dynamic objects, but this method generally exists the problems of high hardware configuration requirements and unacceptable delay.

In response to the above problems, I have explored the two methods respectively and try to make a progress.

First, a SLAM system with semantic segmentation model based on deep learning, called KFDynaSLAM, is proposed. This algorithm uses Mask R-CNN to achieve semantic segmentation of images, cull dynamic objects and keep only rigid contents.

Let the grayscale of the input image be $\boldsymbol{Frame}$, and the dynamic mask as $\boldsymbol{Mask}$ ($\boldsymbol{Mask}$ is a binary matrix with the same shape of $\boldsymbol{Frame}$, in which 1 indicates that the

location is a static object, and 0 indicates that the location is a dynamic object), and the output image is $Frame_{out}$

$$Frame_{out} = Frame \cdot Mask$$

Therefore, only the static part of the input image is included in the $Frame_{out}$, and the dynamic part is removed.

Compared with the previous method based on semantic segmentation, KFDynaSLAM chooses to **perform semantic segmentation only on key frames**, which lowers the number of semantic segmentation times and greatly reduces the processing time. At the same time, in order to solve the problem that with dynamic object removal the relocation process are more easily to come to a failure, a mask update strategy that is **adjusted** in real time **according to the current tracking state** is proposed. The failure is mainly because key frames are no longer generated during relocation, and dynamic mask update is stalled. The strategy proposed can ensure that although KFDynaSLAM has relatively fewer semantic segmentation times, the system still has good robustness and is not prone to tracking failure. Finally, a more efficient semantic segmentation model is adopted, which further decreases the hardware requirement and speeds up the system running.

However, after the above improvements, the semantic segmentation model is still a serious drag on the system's operating speed and causes the KFDynaSLAM cannot meet the real-time requirements. To solve this problem, a strategy of removing dynamic points based on **the geometric method with information from auxiliary camera in the environment** is proposed. This method takes advantage of the static characteristics of the cameras in the environment which means its dynamic mask can be obtained easily, and try to use the dynamic mask information obtained by the cameras in the environment to assist the target camera to achieve dynamic point culling. The method includes three main steps: acquiring the dynamic mask of the camera in the environment, matching the pose of the target camera and the auxiliary camera in the environment, and projecting the feature points of the target camera frames onto the imaging plane of the camera in the environment to judge if dynamic.

Firstly, I use the background-cut algorithm to extract the dynamic mask of the camera in the environment. Let the original image captured by the camera in the environment to be I_k, and the image at the start time with no dynamic objects be the background image B_k; then the output foreground image at the current time is the dynamic Mask $front_k$.

$$front_k = |I_k - B_k| > B$$

Then use the homography matrix to represent the relative poses of the target camera and the auxiliary camera in the environment. Use the scoring of the reprojection error to judge the matching effect and decide whether to accept the matching result.

To use the obtained information in the dynamic point culling of the target cameras' images, when the Local Mapping thread is about to add a map point, it will **be projected to the imaging plane of the camera in the environment**, in order to filter out the dynamic points and only add static points to the map. The projection rule is as follows.

Let us consider the target camera and the camera in the environment calibrated for the last time. The frame captured by the target camera is simply regard as the reference frame $Frame_{ref}$, whose relative pose to the world coordinate system is R_w^{ref} (rotation matrix), t_w^{ref} (translation vector). The homography matrix of the two cameras is H_{ref}^2. For the map point P_w in the

world coordinate system, the coordinates in the camera coordinate system of the reference frame are P_{ref}.

$$P_{ref} = R_w^{ref} * P_w + t_w^{ref}$$

The coordinates of P_{ref} in the pixel coordinate system of the reference frame (homogeneous coordinates used) is p'_{ref} .

$$p'_{ref} = K_1 P_{ref}$$

The projected position of the camera point in the environment is p'_2 (homogeneous coordinates used).

$$p'_2 = H_{ref}^2 p'_{ref}$$

Therefore, the projection of the map points on the imaging plane of the camera in the environment can be obtained, and then it can be determined whether p'_2 is a dynamic feature point according to previously obtained dynamic area of the camera in the environment.

In the experimental part, I conducted experiments on the TUM dataset and the actual scene picture sequences separately, and tested the performance indicators of KFDynaSLAM and EnvDynaSLAM, such as accuracy, tracking success rate, and per frame delay, and compared them with the basic system ORB-SLAM.

The above experiments show that the mean location error of KFDynaSLAM is only **20%** compared with ORB-SLAM in highly dynamic sequences, proving that this strategy can promote the tracking accuracy. Another experiment shows the two methods proposed in this project can **improve the tracking success rate** when applied to SLAM. In the test of time experiment, the dynamic points culling time of KFDynaSLAM still accounts for a large proportion, making it only suitable for applications that do not require real-time, but it accounts for a rather small proportion in EnvDynaSLAM and the system can **meet the real-time requirements**. Compared with DynaSLAM (ORB-SLAM with Mask R-CNN), the per frame delay of KFDynaSLAM has been decreased to **12.55%**, and for EnvDynaSLAM, it is only **3.07%**.

In addition, the step-by-step experiment shows that KFDynaSLAM can reduce the number of semantic segmentation times to about 1/10 than before while the dynamic segmentation accuracy is not too much decreased. And step-by-step experiments of EnvDynaSLAM prove the combination of background-cut algorithm and morphology filtering can contribute to get dynamic region of the images taken by the camera in the environment. SURF performs better in both the quality and efficiency of feature extraction process than ORB and SIFT. Another conclusion is that the dynamic point filtering quality is highly correlated with the position of the camera in the environment.

Based on the above theoretical analysis and experimental results, the problem of dynamic point removal of the research is very critical to the application of SLAM in indoor positioning. The two dynamic point removal methods proposed in this paper, KFDynaSLAM and EnvDynaSLAM, both have good removal effects. In contrast, KFDynaSLAM in offline mode (called offlineDynaSLAM in this paper) has the best accuracy when removing dynamic points, but due to its serious delay, it is only recommended to apply to applications that do not require real-time performance; and EnvDynaSLAM's delay is the lowest, and it can meet the real-time requirements of the system, so it is more suitable for scenarios that require real-time positioning and navigation.

For KFDynaSLAM, future work will focus on solving the problem of large semantic segmentation delay, which can be improved by using a lightweight customized model or a server-client collaboration. For EnvDynaSLAM, we will add a part of multiple environment cameras information fusion to it. This change is expected to achieve map integration, and further improve its accuracy and robustness. In addition, I will add filters or Bundle Adjustment to improve the matching quality of the target camera and the camera in the environment, which is the paramount part of the EnvDynaSLAM.