Journal of Computers Vol. 33 No. 5, October 2022, pp. 197-211
doi: 10.53106/199115992022103305017

Research on Improved Algorithm of Mobile Robot Vision SLAM Based on Depth Camera

Cai-Wen Niu, Shi-Hui Dong*, Yan Zeng

School of Automation Engineering, Tangshan Polytechnic College, Tangshan City 063600, Hebei Province, China
niucaiwen0326@163.com, tgy511389410@163.com, zengyan3925@163.com

Received 1 October 2022; Revised 8 October 2022; Accepted 16 October 2022

Abstract. Aiming at the problems of low accuracy, poor real-time performance and large cumulative error in traditional depth camera visual synchronization positioning and map construction algorithms, this paper proposes an improved visual SLAM algorithm for mobile robots based on depth camera. According to the realization process of robot vision, a feature point extraction algorithm based on feature extraction and image segmentation watershed algorithm is proposed in the front-end process, which improves the real-time performance of the algorithm. Then the RE-RANSAC algorithm is used to eliminate the mismatched feature points to improve the matching accuracy, and then the accumulated error is eliminated through closed loop detection, and finally the process of robot mapping is completed. After simulation experiments, the feasibility of the improved algorithm is proved, and the robot's mapping and trajectory estimation are completed.

Keywords: feature point extraction, mismatch, loopback detection, trajectory estimation

1 Introduction

Autonomous navigation and autonomous control are the key technologies for mobile robots to face the environment autonomously, that is, simultaneous positioning and map construction technology (SLAM) [1]. With the development of vision sensors, robot SLAM systems based on depth vision have become a hotspot in SLAM algorithm research because they can provide robots with richer environmental information and endow robots with stronger perception capabilities.

Cameras are the source of sensor data for visual SLAM systems. According to different camera categories, SLAM algorithms can be divided into: monocular vision SLAM algorithm, binocular vision SLAM algorithm, and depth vision SLAM algorithm [2]. Among them, on the basis of inheriting the function of recording image information, the depth camera can output the color information and depth of field distance of the image, which can avoid the lack of depth information of the monocular camera and the tedious calculation problem of the binocular camera. The complete depth vision algorithm system includes five main parts, including sensor data, visual odometry, nonlinear optimization, mapping and loop closure detection [3]. At present, the existing research results show that the image data extraction stage in the visual odometry stage mainly has the characteristics of poor real-time performance, which affects the robot decision-making. This paper mainly focuses on the technical status of visual odometry and nonlinear optimization in the vision process. The following work is done:

Firstly, based on the depth camera, the SLAM algorithm is improved to improve the real-time feature point extraction.

Secondly, in the process of feature point extraction, watershed algorithm is integrated and HHFT algorithm is proposed. First, pixel gradient in RGB color space is calculated, then dynamic threshold is calculated to binarize gradient image, and finally watershed transformation algorithm is used to further refine the area representing gradient.

This paper is organized as follows. Section II discusses the related technology and the related research results of different students. Section III discusses the implementation process of the improved SLAM algorithm. Section IV uses the improved processing algorithm to process classic images, and compares the results. Section V is the conclusion, and describes the shortcomings of this paper and the next research plan.

* Corresponding Author

2 Related Work

In 2009, Klein et al. [4] proposed the concept of image key frames and the concept of relocation. In the same year, the team also proposed the SLAM vision system based on key frames [5]. ORB-SLAM [6], an open source SLAM system, was put forward in 2015. The system creatively divided SLAM into three threads running in parallel. This innovation not only improves the real-time performance of the algorithm, but also greatly improves the robustness of the algorithm. The positioning accuracy of the system in the test sequence has reached a high level. In addition, it does not require high performance of the computer, and it can run on the ordinary CPU in real time. In 2017, the team also proposed ORB-SLAM2 [7] system, which supports binocular cameras and RGBD cameras in addition to monocular cameras. In 2016, Forster et al [8]. proposed the SVO (Semi direct Visual Odometry) system. The front end of the system combines the feature method and the direct method. Although it lacks the back end and loopback detection modules, it has obvious advantages, such as fast computing speed and low requirements for computing resources, which can run on embedded devices in real time. In 2019, Sumikura et al [9]. proposed OpenVSLAM. The SLAM system has high availability and scalability, and can be called as a third-party program. This feature is not available in many traditional SLAM systems. ORB-SLAM3 [10] was born in 2020. The system can run in various scenarios robustly, and its accuracy is 2-5 times higher than ORB-SLAM2. Another innovation of the system is that it can reuse all previous information in all algorithm stages.

3 Basic Principle and Realization Process of Improved SLAM Algorithm for Mobile Robot Based on RGB-D

The improved depth vision SLAM system framework proposed in this paper is shown in Fig. 1. First, use Zhengyou Zhang's calibration method to calibrate the Kinect camera to obtain the internal parameters of the camera [11], as the premise to ensure the accuracy of the later stage. Then the algorithm of watershed transformation is performed based on the color gradient image. On this basis, the classic Harris [12] algorithm is used to locate the feature points, and then feature matching is performed. The RE-RANSAC algorithm is used to eliminate mismatched points, and the closed loop detection method is used to eliminate accumulated errors, realize map optimization and update, and finally complete the robot's mapping process.

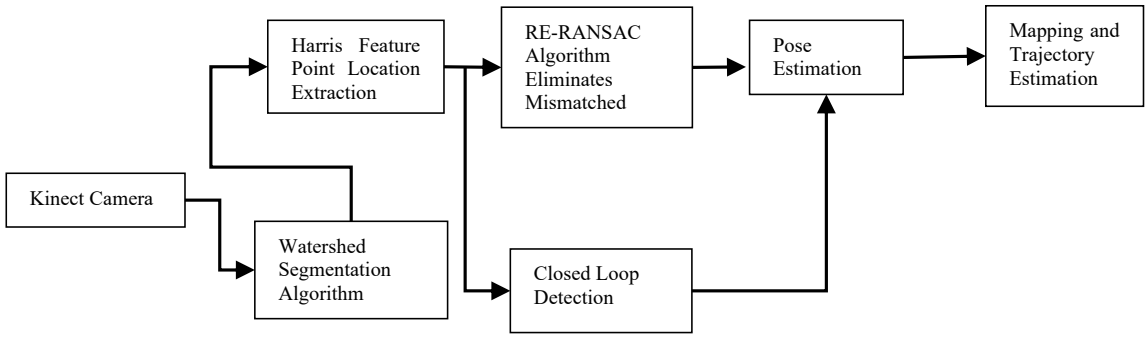


Fig. 1. Improved SLAM algorithm system diagram

3.1 Kinect Camera Parameter Calibration

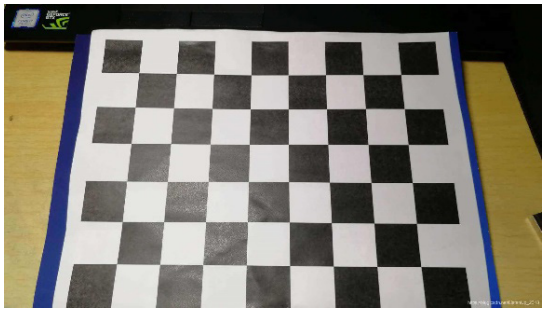
Camera calibration is the premise to ensure the accuracy of the algorithm. The camera calibration process is as follows:

(1) Ready to calibrate the chessboard and Kinect camera;

(2) Start Kinect;

(3) Perform camera calibration. RGB camera calibration and infrared camera calibration are performed on the terminal respectively. For higher calibration accuracy, it is necessary to move the camera sufficiently in all directions and angles within the range where the calibration plate can be seen (Fig. 2).

(a) 9x7Checkerboard



(b) Kinect V2 camera

Fig. 2. 9x7Checkerboard (left); Kinect V2 camera (right)

(4) The calibration results are as follows

Calibration results of internal parameters of RGB camera and infrared camera:

$$K_{rgb}=\begin{pmatrix}576.20101 & 0 & 337.76381\\ 0 & 575.80663 & 239.90596\\ 0 & 0 & 1\end{pmatrix}. \tag{1}$$

$$K_{ir}=\begin{pmatrix}581.95708 & 0 & 324.41341\\ 0 & 581.18517 & 249.57842\\ 0 & 0 & 1\end{pmatrix}. \tag{2}$$

The calibration results of distortion parameters of RGB camera and infrared camera are as follows:

$k1_{rgb}=0.10765$, $k2_{rgb}=-0.112130$, $k3_{rgb}=0.00032$, $p1_{rgb}=0.00033$, $p2_{rgb}=0.00580$

$k1_{ir}=-0.06173$, $k2_{ir}=0.14329$, $k3_{ir}=0.00264$, $p1_{ir}=0.00040$, $p2_{ir}=0$

3.2 Feature Point Detection

In the process of feature point extraction, the watershed algorithm is integrated, and the Hue Histogram Features Transform algorithm is proposed. First, the gradient of pixels in RGB color space is calculated, then the dynamic threshold is calculated to binarize the gradient image, and finally, the region representing the gradient is further refined by watershed transform algorithm. Find dozens of points with large response values in the upper positioning corner points of the watershed image, draw the hue histogram of pixels within a certain radius near each feature point, and use it as the descriptor of the feature point. The process is as follows:

(1) RGB model and color image gradient:

Suppose the RGB component of a point in the color components is *c*,

$$c(x,y)=\begin{bmatrix}c_R(x,y)\\ c_R(x,y)\\ c_B(x,y)\end{bmatrix}=\begin{bmatrix}R(x,y)\\ G(x,y)\\ B(x,y)\end{bmatrix}. \tag{3}$$

Use *R*, *G* and *B* to represent the unit vector along the *R*, *G*, *B* axis in the color space, and define the vector as follows:

$$u = \frac{\partial R}{\partial x} r + \frac{\partial G}{\partial x} g + \frac{\partial B}{\partial x} b \ . \tag{4}$$

$$v = \frac{\partial R}{\partial y} r + \frac{\partial G}{\partial y} g + \frac{\partial B}{\partial Y} b \ . \tag{5}$$

where $R(x, y)$, $G(x, y)$ and $B(x, y)$ are functions of x, y. Define g_{xx}, g_{yy}, g_{xy} in terms of the dot product of u and v:

$$g_{xx} = u \cdot u = \left\| \frac{\partial R}{\partial x} \right\|^2 + \left\| \frac{\partial G}{\partial x} \right\|^2 + \left\| \frac{\partial B}{\partial x} \right\|^2 \ . \tag{6}$$

$$g_{yy} = v \cdot v = \left\| \frac{\partial R}{\partial y} \right\|^2 + \left\| \frac{\partial G}{\partial y} \right\|^2 + \left\| \frac{\partial B}{\partial y} \right\| \ . \tag{7}$$

$$g_{xy} = u \cdot v = \frac{\partial R}{\partial x} \cdot \frac{\partial R}{\partial y} + \frac{\partial G}{\partial x} \cdot \frac{\partial G}{y} + \frac{\partial B}{\partial x} \cdot \frac{\partial B}{\partial y} \ . \tag{8}$$

Then it can be proved that the gradient magnitude in a is as follows:

$$F_\theta(x, y) = \left\{ \frac{1}{2} \left[(g_{xx} + g_{yy}) + (g_{xx} - g_{yy}) \cos 2\theta(x, y) + 2 g_{xy} \sin 2\theta(x, y) \right] \right\}^{-2} \ . \tag{9}$$

Where θ is the angle value of the pixel gradient,

$$\theta(x, y) = \frac{1}{2} \arctan \left[\frac{2 g_{xy}}{g_{xx} - g_{yy}} \right] . \tag{10}$$

(2) Dynamic threshold acquisition and integration into watershed algorithm.

In order to improve the intuitiveness of gradient images, it is necessary to determine the threshold to distinguish the target subject from the background, the image $h(x, y)$ after the image is binarized by $f(x, y)$ is:

$$h(x, y) = \begin{cases} a, & f(x, y) > T \\ b, & f(x, y) \le T \end{cases} \ . \tag{11}$$

The pixel with value c represents the object part, and the pixel with value d represents the background part. Normally $a = 1$, $b = 0$.

The histogram component of the image is represented as:

$$p_q = \frac{n_q}{n} \quad q = 0, 1, 2, \cdots, L - 1 \ . \tag{12}$$

n is the total number of pixels in the image, n_q is the number of pixels with gray level q, L is the total number of discrete gray levels. The maximum class variance σ_B^2 between two pixels is:

$$\sigma_B^2 = p_1(k)\left[m_1(k) - m_G\right]^2 + p_2(k)\left[m_2(k) - m_G\right]^2 \ . \tag{13}$$

Among them, $p_1(k)$ and $p_2(k)$ are the proportions of sets C_1 and C_2 in the total pixels:

$$p_1(k) = \sum_{i=0}^{k} p_i \ . \tag{14}$$

$$p_2(k) = \sum_{I=k+1}^{L-1} p_i \ . \tag{15}$$

$m_1(k)$ and $m_2(k)$ are the average gray levels of pixels in sets C_1 and C_2, and m_G is the global gray level.

$$m_1(k) = \sum_{i=0}^{k-1} ip_i \ . \tag{16}$$

$$m\ (k) = \sum_{i=k+}^{L-} ip_i \ . \tag{17}$$

$$m_G = \sum_{i=0}^{L-1} ip_i \ . \tag{18}$$

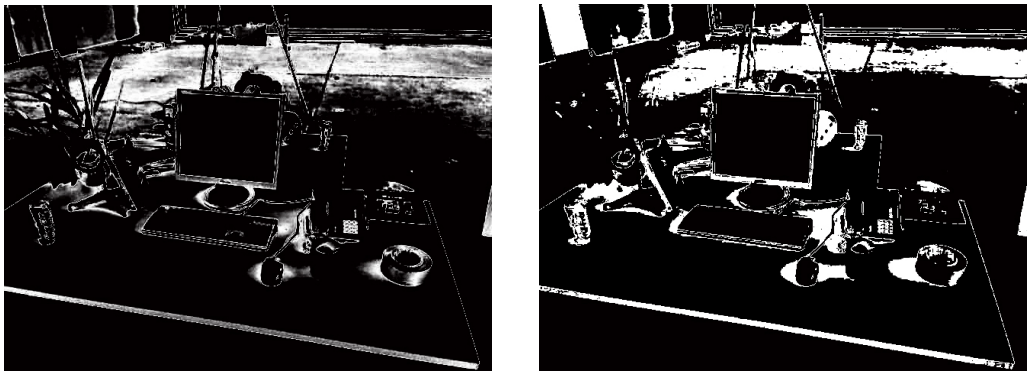
Use mean filtering to denoise the binarized image, that is, replace the noise value with the mean pixel value near the noise, so the pixel value after mean filtering is expressed as:

$$h(x,y) = \frac{1}{m}\sum f(x,y) \ . \tag{19}$$

The following figure (Fig. 3) shows the result of feature point extraction and matching of classic images of the dataset TUM:

(a) Original image

(b) Gradient image



(c) Binarized image (d) Mean filtered image

Fig. 3. Feature recognition and matching process images

(3) Based on the above, the watershed algorithm is integrated, the schematic diagram of the watershed algorithm in the two-dimensional plane is as follows (Fig. 4):

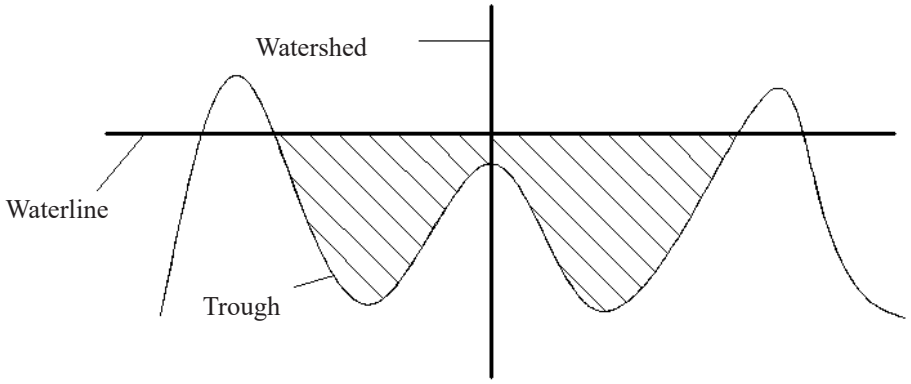


Fig. 4. Schematic diagram of the watershed algorithm

The recursive process of the watershed algorithm is as follows:

$$X_{h_{\min}} = T_{h_{\min}}(I), \quad \forall h \in \left[h_{\min}, h_{\max} - 1\right] . \tag{20}$$

Among them, h represents the range of gray value, $h_{\min}$ is the minimum gray value, $h_{\max}$ is the maximum value, and $X_{h_{\min}}$ represents the set of all pixels whose gray value is less than $h_{\min}$.

$$X_{h+1} = \min_{h+1} \cup C_{X_h}(X_h \cap X_{h+1}) . \tag{21}$$

Among them, X_{h+1} is all points whose gray value is less than $h+1$, and $\min_{h+1}$ indicates that this point is the minimum point of the newly generated basin, that is, the gray value $h+$ generates a new basin again. $X_h \cap X_{h+1}$ means the intersection of X_h and X_{h+1}, C_{xh} means the basin where X_h is located, so $X_h \cap X_{h+1}$ is the point where the two are in the same catchment basin, and the overlapping area of multiple areas is the watershed of the image.

(4) Harris extracts eigenvalues

The Harris corner extraction algorithm is applied to the image processed by the watershed algorithm, and the result is shown in Fig. 5. The white pixel area with a value of 1 in the figure represents the peak, and the black pixel area with a value of 0 represents the valley. After the final transformation, the edge contour areas of the subjects in the scene are more clearly defined. The distribution of detected corner points is as follows:

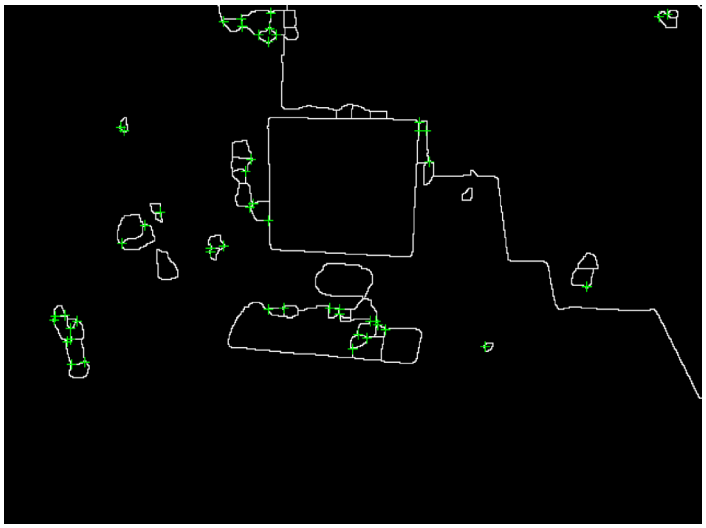


Fig. 5. Image of corner distribution

3.3 Image Feature Matching

We recommend using Computer Modern Roman (CM) fonts, Times, or one of the similar typefaces widely used in photo-typesetting. (In these typefaces the letters have serifs, i.e., short endstrokes at the head and the foot of letters.) Italic type may be used to emphasize words in running text. Bold type and underlining should be avoided. With these sizes, the interline distance should be set so that some 45 lines occur on a full-text page.

The pixel values around the feature points are the main parameters for feature matching. The degree of achromaticity in the HIS color space needs to be expressed by the luminance value. To convert the RGB color space to the HIS model, the hue component H of each RGB pixel is as follows:

$$H = \begin{cases} \theta & (B \le G) \\ 2\pi - \theta & (B > G) \end{cases}. \tag{22}$$

$$\theta = \arccos\left\{ \frac{\frac{1}{2}\left[2R - G - B\right]}{\left[(R-B)^2 + (R-B)(G-B)\right]^{1/2}} \right\}. \tag{23}$$

The angle θ is measured based on the red axis in HSI space, as shown in Fig. 6 and Fig. 7. In the HSI model, the primary colors are separated by 120 degrees, and the hue of a point is determined by its angle with the red axis.

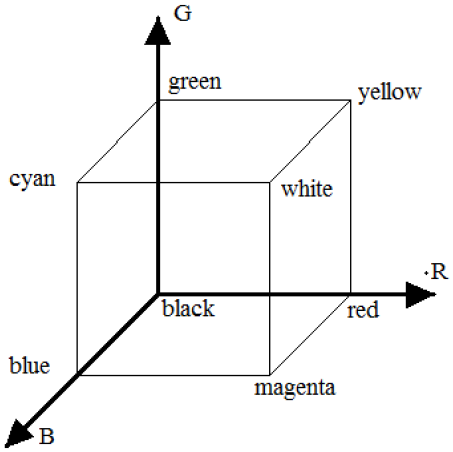


Fig. 6. RGB pixel graphics

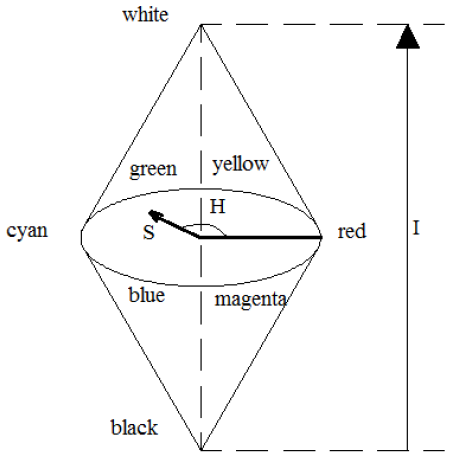


Fig. 7. HSI model diagram

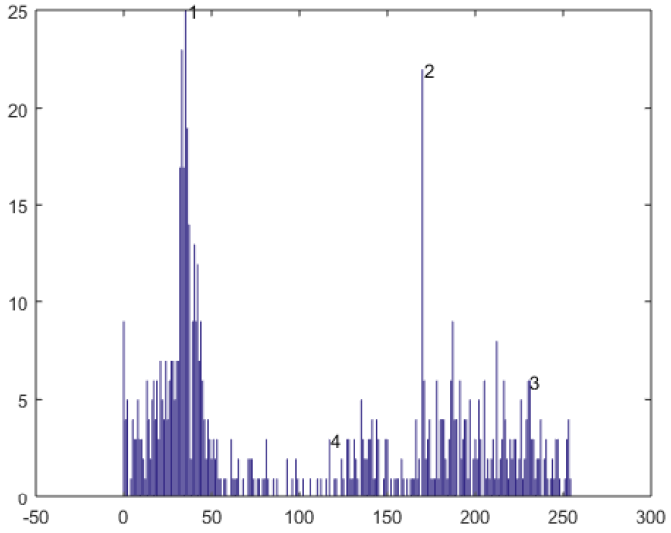


Fig. 8. Hue histogram

The histogram depicts the color blocks of the brightness distribution around the feature point. The peaks of the histogram are arranged according to size and represented by black numbers. Then, the peak value that truly reflects the pixels around the feature point can be obtained by limiting the distance. Generally, 2 to 4 peaks are selected as Fig. 8.

After the feature points of the image are extracted, the BruteForce method is usually used for matching, but when processing a large number of feature points or matching a frame of image and the entire map, this method takes a long time to match, and there are many mismatches. In view of the above problems, this paper uses the bidirectional KNN feature matching method based on the FLANN algorithm to reduce the mismatch points, and adopts the multiple random k-d tree method to improve the speed of fast nearest neighbor search. The feature matching effect is shown in Fig. 9.

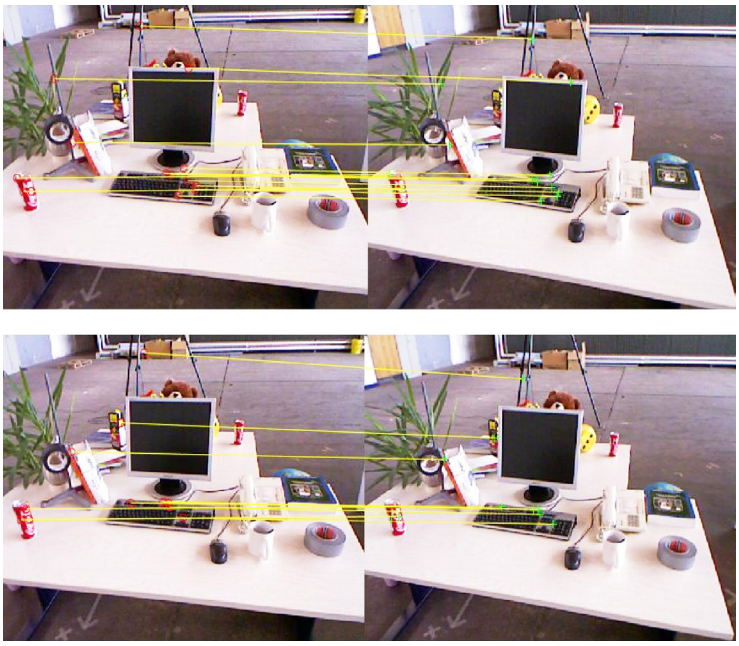


Fig. 9. Feature matching results

3.4 False Matching Point Elimination

In order to further improve the matching accuracy of feature points, it is necessary to eliminate the mismatched points in the matching. This paper proposes an improved RE-RANSAC based on the RANSAC [13] algorithm. The algorithm schematic diagram is as follows:

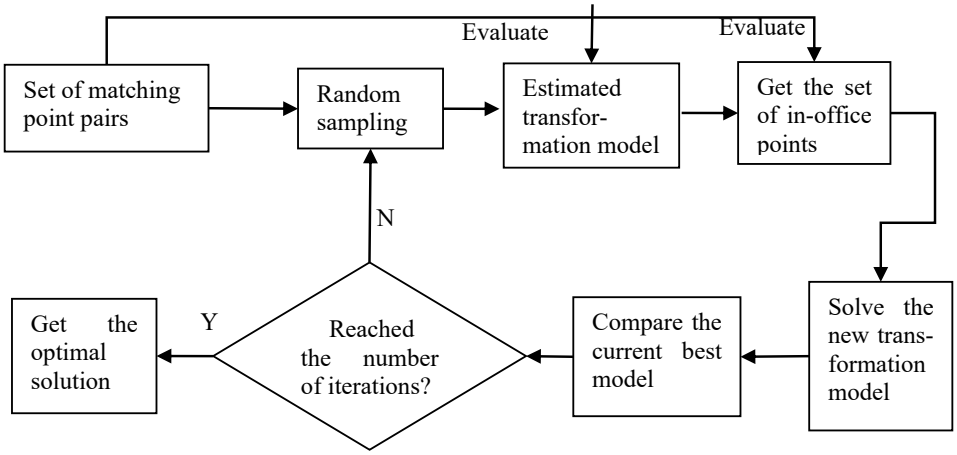


Fig. 10. RE-RANSAC algorithm schematic diagram

Different from the original algorithm, after the RE-RANSAC algorithm obtains a new transformation model, it uses the entire 3D coordinate matching point set to evaluate the model again, and obtains a new set of intra-office points and transformation model, and then compares it with the current optimal model. Compare and filter. When randomly sampling *k* points:

$$1-P=(1-u^k)^N \ . \tag{24}$$

p is the probability that the sampling set obtained after N iterations does not contain outliers, u is the probability of obtaining an intra-office point in a single sampling. The number of iterations *N* can be expressed as:

$$N=\frac{\lg(1-p)}{\lg(1-(1-v)^k)} \ . \tag{25}$$

The improved RE-RANSAC algorithm performs a secondary evaluation and screening of the motion model in each iteration process to improve the accuracy of the algorithm and increase the operation speed by reducing the number of iterations. Fig. 11 shows the effect of RE-RANSAC algorithm in eliminating false matches.

Fig. 11. Rendering after eliminating matching errors

3.5 Pose Estimation

The pose estimation process is the process of solving the motion transformation matrix between two adjacent frames of images. The transformation matrix is as follows:

$$T=\begin{bmatrix} & & & t_x \\ & R & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \ . \tag{26}$$

R is the rotation matrix, If the point $P=(x,y,z,1)$ and the corresponding motion transformation matrix T are known, then the projection $P^{'}$ of the point P on the motion transformation matrix T is:

$$P^{'}=P\times T \ . \tag{27}$$

The improved RE-RANSAC algorithm iteratively screens to obtain the motion transformation matrix T estimated by the optimal set of intra-office points, and takes it as the initial condition in the pose optimization process. And when the number of matching points is large, the GICP algorithm with higher accuracy is used to optimize the estimated motion transformation moment.

In the probabilistic model, it is assumed that there are two sets, $\hat{A} = \{\hat{a}_i\}$ and $\hat{B} = \{\hat{b}_i\}$. If $\hat{A}$ and $\hat{B}$ are completely corresponding, there is a motion transformation T^*,

$$\hat{b}_i = T^* \hat{a}_i \quad (i = 1, 2, \cdots, N) \cdot \tag{28}$$

$$a_i \sim N(\hat{a}_i, C_{i,A}) \cdot \tag{29}$$

$$b_i \sim N(\hat{b}_i, C_{i,B}) \cdot \tag{30}$$

$\hat{b}_i$ is the estimated point in set $\hat{B}$, $\hat{a}_i$ is the estimated point in set $\hat{A}$, $C_{i,A}$ is the covariance matrix of the point to be observed in set A, $C_{i,B}$ is the covariance matrix of the point to be observed in set B.

For any motion transformation T, define $d_i^{(T)} = b_i - Ta_i$, since and b_i are independent Gaussian distributions, combined with Equation 28, we can get:

$$d_i^{(T^*)} \sim N(\hat{b}_i - T^* \hat{a}_i, C_{i,B} + T^* C_{i,A} (T^*)^T) = N(0, C_{i,B} + T^* C_{i,A} (T^*)^T) \cdot \tag{31}$$

Through the maximum likelihood estimation, the optimal motion transformation T can be obtained as, T is the optimized motion transformation matrix:

$$T = \arg_T \max \prod_i p(d_i^{(T)}) = \arg_T \max \sum_i \lg(p(d_i^{(T)})) \cdot \tag{32}$$

3.6 Closed Loop Detection

Closed-loop detection is introduced in order to eliminate accumulated errors [14]. First, key frames need to be obtained by filtering to reduce the processing time of the algorithm. The specific method is as follows: let the key frame sequence be $P_i(0, 1, \cdots, N)$, and the first frame image P_0 is the first frame key frame, each time a new frame is collected, it is necessary to calculate its and the last frame in the sequence P_i. The motion transformation of the frame is obtained, and the motion transformation matrix $T_i(i = 0, 1, \cdots, N)$ is obtained. If the motion transformation amount meets the set threshold requirements, the frame is determined as a key frame, and the frame is compared with P_i. If the matching is successful, the local loop or global closed loop shown in Fig. 12 can be formed, which can quickly and effectively reduce the accumulated error.

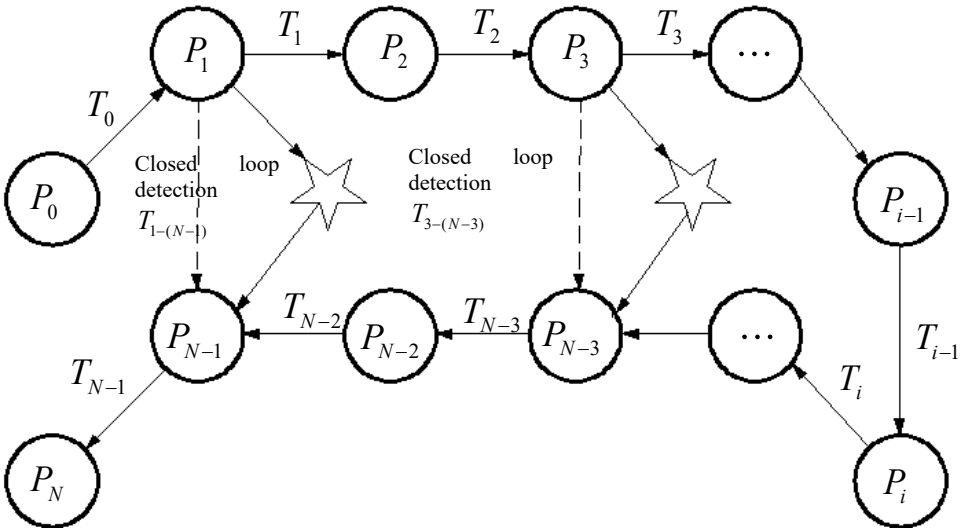


Fig. 12. Local closed-loop

4 Experimental Results and Analysis

The experimental equipment in this paper consists of three parts: an Intel dual-core 4.0 GHz laptop, equipped with Ubuntu 14.04 operating system with Linux as the kernel; a Kinect camera with an image resolution of 640×480 and a maximum frame rate of 30 frames per second, The horizontal field of view is 52°; an experimental mobile robot. The experimental environment is a small laboratory of about 45 square meters, as shown in the following figure (Fig. 13):

Fig. 13. Experimental equipment and experimental environment

During the experiment, the robot was equipped with a laptop and a Kinect camera, and moved in the laboratory at a speed of 0.1 m/s. The Kinect camera obtained color information and depth information of the indoor environment.

Fig. 14 is the accuracy of the point cloud image obtained by the algorithm in this paper is significantly improved, and the contours of objects such as thermos, benches, and wall panels become clear.

Fig. 14. Improved algorithm visuals

Fig. 15 shows the optimization and update of the map using the optimization algorithm. The dark blue part is the key frame, and its vertices represent the various poses of the camera, and the edge connecting the two vertices represents the pose transformation between the two points. The local loopback indicates the match between the current frame and the nearby frame, and the large loopback indicates the match between the current frame and a random frame in the key frame sequence. In this paper, set m = 5, n = 5. Fig. 16 shows the trajectory of the simple motion result of the robot, which can be simulated through MATLAB and has good real-time performance.

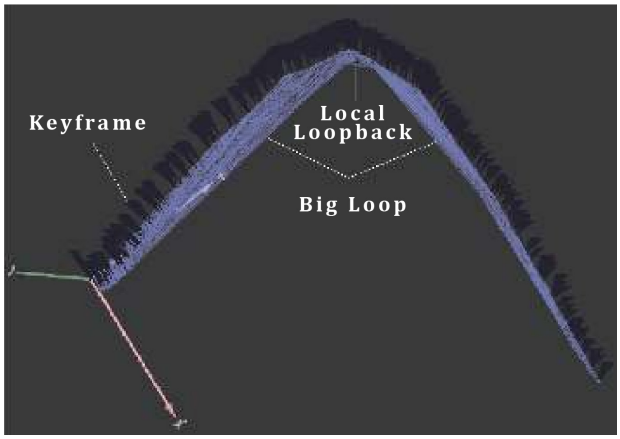


Fig. 15. Rendering after optimization

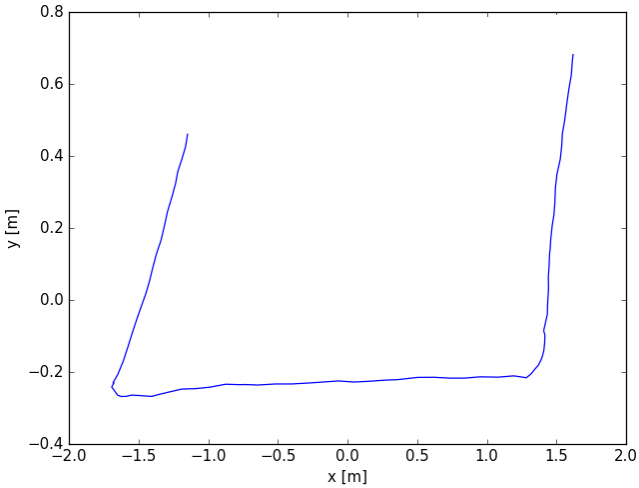
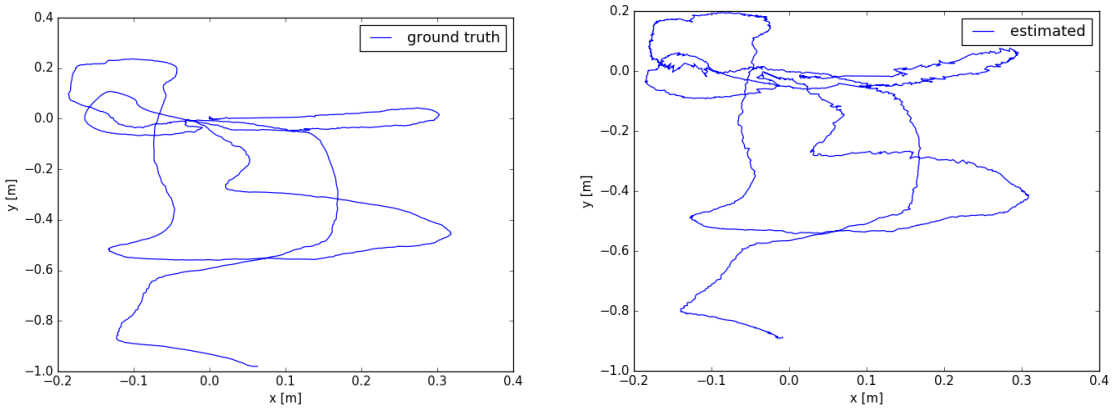


Fig. 16. Trajectory planning graphics

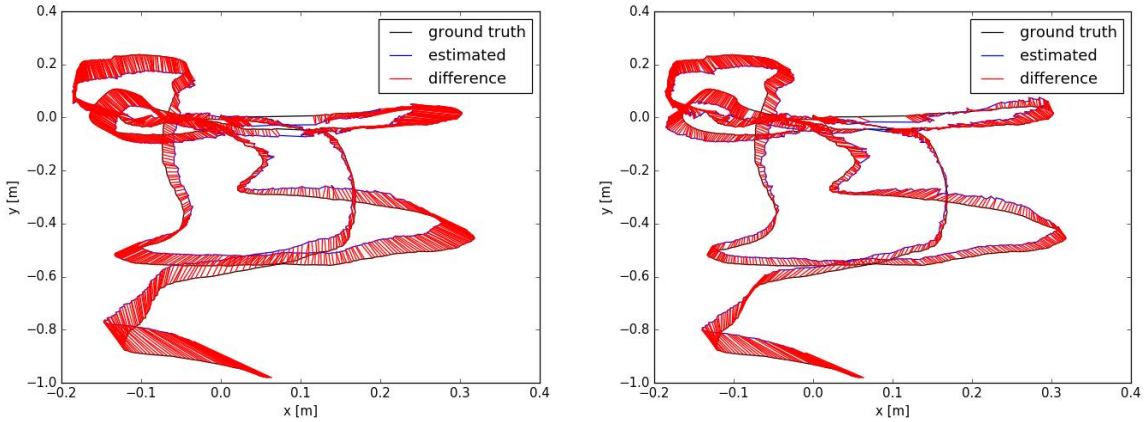
Fig. 17 shows the real motion trajectory of the robot running for a long time in a real indoor environment and the trajectory estimated by the algorithm system are displayed. It can be seen from the figure that the curve laws of the two are close, indicating the reliability of the algorithm system in track estimation.



(a) Ground truth (b) Estimated truth

Fig. 17. Comparison results of trajectory data

In order to prove the improvement of the positioning accuracy of the improved system, the error between the estimated trajectory of the system and the real trajectory is compared with the error between the estimated trajectory of the RGB-SLAM2 system and the real trajectory, as shown in Fig. 18. The comparison results show that the estimated error of the system is lower than the RGB-SLAM2 error.



(a) RGB SLAM2 error (b) This system error

Fig. 18. Comparison of system errors

5 Conclusion

In this paper, a new improved SLAM algorithm for indoor mobile robots based on Kinect is proposed to solve the problems of low accuracy, poor real-time performance and large cumulative error in traditional depth camera visual synchronization positioning and map construction algorithms. Improvements are made in the front-end, back-end, optimization and loop detection stages of the algorithm, which can realize the optimization and update of the 3D map. Online experiments are carried out in a real indoor environment, and the experimental results demonstrate the effectiveness and feasibility of the proposed algorithm. Further improvement work will focus on: 1) fusion of lidar to further improve the richness of incoming information collection; 2) perfecting and improving the SLAM algorithm for dynamic environments.

6 Acknowledgement

This work has been supported by Tangshan Science and Technology Plan Project (Grant number 19130217g).

References

[1] F. Endres, J. Hess, J. Sturm, D. Cremers, W. Burgard, 3-D mapping with an RGB-D camera, IEEE transactions on robotics 30(1)(2014) 177-187.
[2] Z.-Y. Zhang, Microsoft kinect sensor and its effect, IEEE multimedia 19(2)(2012) 4-10.
[3] D.-W. Zhang, S. Su, Research on Visual SLAM Technology of Autonomous Mobile Robot, Journal of Zhengzhou University (Natural Science Edition) 53(1)(2021) 1-8.
[4] G. Klein, D. Murray, Parallel tracking and mapping on a camera phone, in: Proc. 2009 8th IEEE International Symposium on Mixed and Augmented Reality, 2009.
[5] G. Klein, D. Murray, Improving the agility of keyframe-based SLAM, in: Proc. European Conference on Computer Vision, 2008.
[6] R. Mur-Artal, J.M. Montiel, J.D. Tardos, ORB-SLAM: a versatile and accurate monocular SLAM system, IEEE Transactions on Robotics 31(5)(2015) 1147-1163.
[7] R. Mur-Artal, J.D. Tardós, ORB-SLAM2: an open-source slam system for monocular, stereo, and RGB-D cameras, IEEE Transactions on Robotics 33(5)(2017) 1255-1262.
[8] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, D. Scaramuzza, SVO: semidirect visual odometry for monocular and multicamera systems, IEEE Transactions on Robotics 33(2)(2017) 249-265.
[9] S. Sumikura, M. Shibuya, K. Sakurada, OpenVSLAM: a versatile visual SLAM framework, in: Proc. the 27th ACM International Conference on Multimedia, 2019.
[10]C. Campos, R. Elvira, J.J.G. Rodríguez, J.M.M. Montiel, J.D. Tardós, ORB-SLAM3: an accurate open-source library for visual, visual-inertial, and multimap SLAM, IEEE Transactions on Robotics 37(6)(2021) 1874-1890.
[11]Y. Liu, T.-F. Li, Reaserch of the improvement of Zhang's camera calibration method, Optical Technique 40(6)(2014) 565-570.
[12]Y.-N. Yao, W, Wang, Application research on Harris corner detection algorithms, Intelligent Computer and Applications 21(8)(2022) 148-151.
[13]R.B. Rusu, S. Cousins, 3D is here: point cloud library (PCL), in: Proc. 2011 IEEE International Conference on Robotics and Automation, 2011.
[14]X.-M. Yang, S.-Y. Li, Principle, status quo and trend of slam loop detection for mobile robot vision, Journal of Electronic Measurement and Instrumentation, <https://kns.cnki.net/kcms/detail/11.2488.TN.20220401.1841.002.html>, 2022 (accessed 8.10.22).