Sensors and Materials, Vol. 32, No. 7 (2020) 2261–2275
MYU Tokyo

S & M 2256

Visual Odometry Implementation and Accuracy Evaluation Based on Real-time Appearance-based Mapping

Bo Hu[1] and He Huang[1,2*]

[1]School of Geomatics and Urban Spatial Information, Beijing University of Civil Engineering and Architecture, No. 15, Yongyuan Road, Huangcun Town, Daxing District, Beijing 102616, China
[2]Beijing Advanced Innovation Center for Future Urban Design, Beijing University of Civil Engineering and Architecture, No. 1, Exhibition Hall Road, Xicheng District, Beijing 100044, China

(Received May 15, 2019; accepted May 20, 2020)

Keywords: visual odometry, RTAB-MAP, feature detection algorithms, RANSAC, accuracy evaluation

With the rapid development of artificial intelligence and machine learning technology, various robots serving humans have emerged. The positioning and navigation technology of robots has become a research hotspot. Robots relying on visual odometry (VO) are favored by people for their low price and wide range of applications. In this paper, we focus on the algorithm and implementation of VO based on the feature point method. Firstly, the current mainstream feature detection algorithms are compared and analyzed in terms of real-time performance and time efficiency. The random sample consensus algorithm is used to eliminate the mismatch in the image feature matching process. Secondly, using the Kinect vision sensor and the Turtlebot mobile robot system to build the RGB-D simultaneous localization and mapping (SLAM) experimental platform, the real-time appearance-based mapping (RTAB-MAP)-based VO is realized. Finally, to verify the actual running performance of the VO, a series of motion experiments were performed in an indoor environment. The accuracy of the pose estimation of the VO was evaluated, which provides a useful reference for the development of mobile robot positioning technology.

1. Introduction

In recent years, with the rapid development of artificial intelligence and machine learning technology, various robots serving humans have emerged and are now widely used in many fields, such as military, industry, agriculture, and daily life. Achieving the autonomous behavior and decision-making of mobile robots has always been one of the research hotspots, and the positioning and navigation of robots have become indispensable.

Traditional robot positioning often uses the inertial navigation system (INS) combined with the global positioning system (GPS), wheel odometer or other methods. INS-based positioning has the characteristics of high frequency and short-time precision, but its error increases with time. GPS is the most mature global positioning technology available today. However, in

*Corresponding author: e-mail: huanghe@bucea.edu.cn
https://doi.org/10.18494/SAM.2020.2870

ISSN 0914-4935

indoor scenes or some complex outdoor spaces, there are still problems such as multipaths, signals being occluded or even lost, and low output frequency and positioning accuracy. The wheeled odometer is inexpensive and highly reliable, but the positioning error is large when the road surface is uneven or the wheel is slipping.(1)

Visual positioning does not have the above problems, and its application is also very extensive. Visual positioning is also known as visual odometry (VO). Its concept was proposed by Nistér *et al.*,(2) which refers to a meter that processes related image sequences through machine vision technology to realize the real-time estimation of the position and attitude of the mobile carrier. It not only obtains mileage information, but also pose information.

In recent years, VO, as the front end of visual simultaneous localization and mapping (VSLAM), has used single or multiple visual sensors to acquire image information and then estimate camera motion to provide good initial values for the back end. Vision sensors are widely used in robotics, autonomous driving,(3) and many other fields because of their rich information, low cost, and portability. The implementation method of VO is generally divided into two types depending on whether the extraction feature is required. One is the feature-based front end, that is, the feature-based visual odometry (FVO), and the other is the direct method front end, that is, the dense visual odometry (DVO) that does not extract features. DVO generally uses the pixel information of an image frame to estimate the camera pose. The image consistency hypothesis was used. It was first proposed by Steinbrucker *et al.*(4) to turn the camera pose estimation problem into energy minimization. However, owing to its huge data and computational complexity, it could not achieve real-time performance. Compared with the direct method front end, the front end based on the feature point method has long been regarded as the mainstream method of VO. It has the advantages of stable operation, small calculation amount, relative insensitivity to illumination and dynamic objects, and so forth. Therefore, FVO is currently a relatively mature visual odometer program.

2. Materials and Methods

2.1 FVO process and method

FVO is stable in operation and insensitive to dynamic objects and illumination. It is a relatively mature VO solution, which has long been regarded as the mainstream method and has gradually formed a complete process system. The FVO process is shown in Fig. 1. It consists of three modules: feature, motion estimation, and back-end optimization.

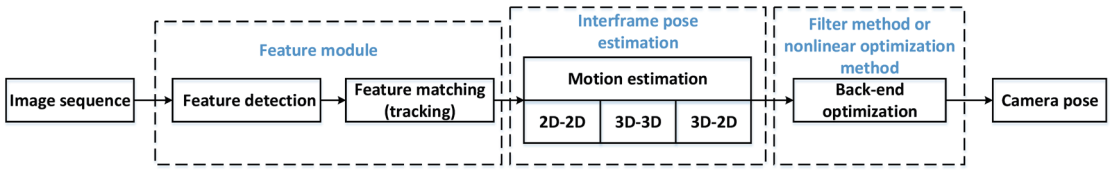


Fig. 1. (Color online) FVO implementation process.

The feature module mainly performs feature point detection, feature matching, and mismatch elimination. The feature points consist of two parts: the key point and the descriptor. The descriptor is a description of the key point information and provides a basis for feature matching. Researchers designed many stable image features, such as scale-invariant feature transform (SIFT)[5] and speeded-up robust features (SURF).[6] Both fully consider the problems in the image transformation process, but require a large amount of calculation. Thus, it is difficult to meet the real-time requirements of feature detection. To meet the VO real-time requirements, researchers proposed oriented FAST and rotated BRIEF (ORB)[7] features, ORB fusion features from accelerated segment test (FAST)[8] corner points, and binary robust independent elementary features (BRIEF)[9] descriptor. ORB has good characteristics in terms of scale, rotation, brightness, and calculation speed.

Feature matching is a critical step in FVO, which solves the problem of data association. The simplest feature matching method is brute force (BF). However, when there are more feature points, the calculation of BF is very large. In this case, the fast library for approximate nearest neighbors (FLANN) is more suitable. In the matching process, it is difficult to completely avoid the error matching for various reasons. We need a certain method to screen it out to improve the accuracy of VO. The simplest method is based on the threshold method, and the random sample consensus (RANSAC)[10] algorithm can also be used to eliminate mismatches.

The motion estimation module calculates a transformation matrix $T_{k,k-1}$ between the current frame I_k and the previous frame I_{k-1}. $T_{k,k-1}$ is calculated by three methods: 2D–2D, 3D–3D, and 3D–2D. 2D–2D uses two-dimensional image point pairs to calculate motion parameters, which are generally used for monocular vision. 3D–3D calculates motion parameters from three-dimensional point pairs, usually used for stereo vision. 3D–2D calculates motion parameters from three-dimensional points and two-dimensional image point pairs. It projects the obtained 3D coordinates into the current 2D image to calculate motion parameters.

In the initial VO back-end optimization, the filtering method is mainly used. The Monocular Simultaneous Localization and Mapping (MonoSLAM) method proposed by Davison *et al.*[11] in 2003 is based on the extended Kalman filter (EKF) framework. Later, researchers introduced the nonlinear optimization method bundle adjustment (BA) in structure from motion (SFM)[12] into VO research, which became the leading method of back-end optimization, among which, parallel tracking and mapping (PTAM)[13] is the first program optimized with BA as the back end.

2.2 Image feature detection algorithm

In the feature module, a feature point is the point in the image that has a large change in gray value or a large edge curvature, and it can reflect the essential characteristics of the image. In recent years, many researchers have designed many feature points manually. The feature points generally have the following properties: (1) repeatability: the same corner can be identified in different areas; (2) distinguishability: for different corners, there should be different expressions; (3) locality: image features should be related to adjacent image areas; and (4) high efficiency: the number of feature points should be much smaller than the number of image pixels.

Following the above principles, the Harris corner,[14] SIFT corner, SURF corner, FAST corner,[15] and so forth are widely used in related fields. The Harris corner detection algorithm is improved on the Moravec[16] corner detection algorithm. The improved algorithm is a signal-based point feature extraction algorithm. The extracted corners have the characteristics of uniform distribution and stable performance, but do not have scale and rotation invariance. The SIFT algorithm is the most classic among the corner detection algorithms. It fully considers the changes in lighting, rotation, and scale that occur during image transformation. Therefore, it has scale and rotation invariance, and its accuracy and precision are very high. Because there are many situations to consider, the amount of calculation is large, and the real-time performance is insufficient. It is generally difficult to run the calculation on a CPU in real time. Some researchers have proposed GPU-accelerated SIFT detection and it has achieved good results.[17] The SURF feature is improved on the basis of SIFT. While ensuring a certain degree of scale and rotation invariance, the calculation amount is reduced to shorten the running time. The scale and rotation invariance are slightly lower in accuracy and precision than the SIFT detection algorithm. The FAST algorithm mainly detects where the local pixel gray level changes significantly. It is fast, but lacks scale and rotation invariance. On basis of the above, some researchers[18] proposed the ORB feature detection algorithm. This algorithm first uses FAST corners for detection and identification, and then adds scale and rotation invariance, and combines the corner descriptor to describe characteristic corners. Owing to the use of FAST corners, the speed is very high and a certain scale and rotation invariance is guaranteed. Considering the real-time requirements in practical engineering, ORB feature detection algorithms are widely used. In the following, we introduce the ORB feature detection algorithm.

The ORB feature detection is an improvement on FAST corners. The main idea of FAST corners is that if a pixel is darker or brighter than a neighboring pixel, this pixel is likely to be a corner. First, select a pixel P in the image, assuming that the brightness of pixel P is P_l. Then take the pixel P as the center, select the pixels on a circle with a fixed radius, set an appropriate threshold, and compare the pixels on the selected circle with the pixel P. If the difference between them is greater or less than the setting threshold, P can generally be regarded as a corner point. Then loop through the above operations to traverse all the pixels to detect all

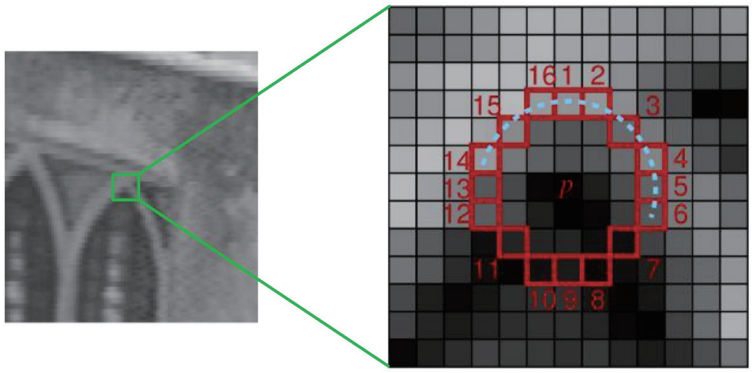


Fig. 2. (Color online) FAST feature points.

the corners in the image, as shown in Fig. 2. The feature points detected in this way are prone to "gathering and clustering", and nonmaximum suppression[19] processing is required on the image to calculate the sum of the absolute values of the pixel difference between the edge point and the center point. Keep the corners of the response maximum in this area to avoid corners that are "too concentrated".

FAST corners do not have scale and rotation invariance. The ORB algorithm constructs an image pyramid, detects corners at each layer of the image pyramid, and calculates the main direction of the feature points by the gray centroid method to achieve the description of the feature direction.

The specific implementation step is to first define the moment of the image block:

$$\begin{cases} m_{pq} = \sum x^p y^q I(x,y) \\ p, y = \{0,1\}, \end{cases} \tag{1}$$

where $I(x, y)$ represents the gray value of the pixel. The centroid of the image block can be expressed as

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right). \tag{2}$$

The direction of the feature point is the line between the geometric center of the image block and the centroid:

$$\theta = \arctan\left(\frac{m_{01}}{m_{10}} \right). \tag{3}$$

The ORB algorithm adds scale and rotation invariance to FAST corners by the image pyramid and gray centroid method, which is also the main reason why ORB feature detection has been widely used since it was proposed. Feature extraction usually includes feature point detection and description. The feature point description in the ORB algorithm uses the BRIEF feature description algorithm. The main idea is to compare the gray values of two pixels (p and q) near the feature point. If the gray value of p is greater than that of q, take 1; otherwise, take 0. Select 128 pairs of pixels near the feature points to form a 128-dimensional description vector of the feature points, and the ORB algorithm calculates the direction information for the feature points. Therefore, the BRIEF descriptor has good rotation invariance.

2.3 Image feature matching algorithm

Image feature matching solves the problem of data association in VO and accurately matches descriptors between images and images or between images and maps. However, owing to some local characteristics of image features, environmental noise, and other factors, mismatches between image feature point pairs widely exist. Selecting suitable feature matching methods

and eliminating mismatched point pairs are two key issues of the image feature matching process.

Finding the matching feature points in two images is generally carried out to compare the descriptor vectors of the feature points. The descriptor vectors of the corresponding feature points in the two images should be consistent. According to this characteristic, the distance between the feature descriptors of the two feature points in the vector space can be calculated to represent the similarity between the two feature points. One of the most common methods is to calculate the descriptor distances between all the feature points of one image and each feature point of another image to be matched, and then sort the calculated distances, and take the nearest one as the matching feature points of two images. The Euclidean distance and Hamming distance are generally used to calculate the distance of the descriptor vectors. For floating-point descriptors, the Euclidean distance is generally used. For binary descriptors such as BRIEF, the Hamming distance is used. The Hamming distance specifically indicates the number of different digits in the corresponding dimension between the descriptor vectors, and BF Matcher uses this method. However, this method has certain shortcomings, which mainly manifests in the large amount of calculation. When the camera continues to move and the number of feature points increases rapidly, the correct nearest distances cannot be calculated in real time. Later, researchers proposed FLANN,[20] which is more suitable for the situation with a large number of feature points, and has higher calculation efficiency but less matching accuracy than BF Matcher.

Given the effects of the external observation environment, such as lighting conditions, scene complexity and so forth, the use of BF Matcher and FLANN to match feature points will generate a certain number of mismatches. This not only affects the operating efficiency of the front-end FVO, but also the input data used by the back-end FVO for map construction. The use of the RANSAC algorithm can effectively filter and delete mismatches to solve the above problems. Its main implementation process is shown in Fig. 3.

The overall idea of RANSAC implementation is to select the noncollinear intrasite point P in the feature point pair for model construction, calculate the homography matrix, set the threshold K of iteration times, and carry out the model verification of the remaining outer points in the feature point pair. The model that satisfies the most point pairs is set as the optimal model. The specific steps of the method are as follows:

(1) The model H is constructed by selecting the intrasite point P of the initial hypothesis in the feature point pair. All unknown parameters can be calculated using this model.

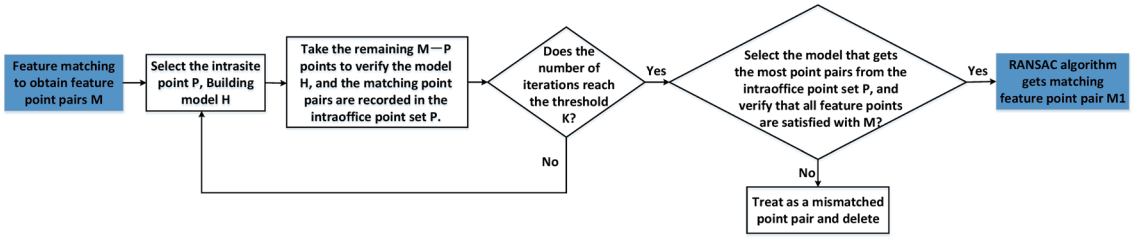


Fig. 3. (Color online) RANSAC implementation process.

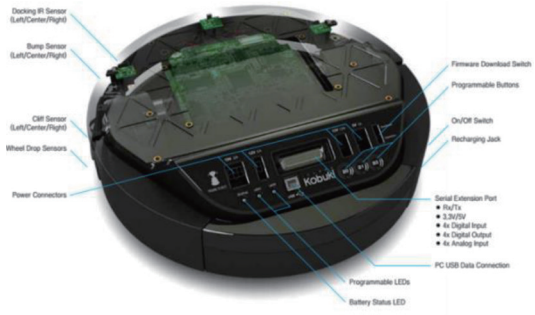
(2) Test the remaining M–P pair feature points with the selected model H, and consider the point pairs that match the estimation model to be hypothetical intrapoints as well.
(3) Determine whether the estimation model is reasonable by judging whether the number of points classified as hypothetical intrapoints is sufficient.
(4) Re-estimate the model with all the local points containing the initial hypothesis.
(5) Evaluate the model by estimating the intraoffice points and the model's error rate.

This process is repeated iteratively. The model with very few intrasite points generated each time is discarded, and models that are better than the existing conditions are continuously selected to eliminate mismatched point pairs and improve the accuracy of VO.

2.4 RGB-D simultaneous localization and mapping (SLAM) experimental platform

To implement FVO in detail, the Kinect V1 vision sensor and Turtlebot2 mobile robot system are used to build an RGB-D SLAM experimental platform. The platform is based on hardware abstraction, low-level device control, common function implementation, message passing, and other functions, as shown in Fig. 4(a). Turtlebot2 is a robot development platform with high performance and reasonable price designed by Willow Garage. It mainly includes the Kobuki mobile base, as shown in Fig. 4(b), and uses the currently popular Robot Operating System (ROS) as its operating system. It can realize map area identification, robotic arm manipulation, autonomous obstacle avoidance and path planning, and 3D map navigation and following functions. The Kinect V1 vision sensor is an XBOX 360 somatosensory camera developed by Microsoft in 2010. It can directly measure the pixel distance through infrared structured light, as shown in Fig. 4(c). In this study, we did not obtain the depth information of the images

(a)



(b)

(c)

Fig. 4. (Color online) (a) RGB-D SLAM experimental platform, (b) Kobuki mobile base, and (c) Kinect V1 vision sensor.

it collects, but only used the RGB images it obtains as the image data of the experimental system. The image resolution is 640 × 480, the sampling frequency is 30 Hz, and the computer operating system used in the experiment is 64-bit Ubuntu 16.04. The ROS Kinect system is used to develop functions in the Clion 2019 integrated environment. The software libraries involved are the linear algebra libraries Eigen and Sophus, the graph optimization tool G2O, the graphics library OpenCV, PCL, and so forth. The main development language is C ++.

3. Results and Discussion

3.1 Implementation of RTAB-MAP

Real-time appearance-based mapping (RTAB-MAP) is a more classic solution in RGB-D SLAM. It implements everything from FVO, bag-based loop detection, back-end pose map optimization, and point cloud and triangle mesh map generation. Therefore, RTAB-MAP gives a complete RGB-D SLAM scheme.

We operated the experimental platform to conduct an RTAB-MAP SLAM test in a laboratory. The running effect of RTAB-MAP is shown in Fig. 5. The system interface is divided into four parts: the orange interface in the upper left corner is the loopback detection process. The system detects loops and optimizes poses by contrast recognition of the original image. The green interface in the middle on the left is the error detection warning window. The system controls the mapping results by detecting the "drift" of the front-end VO. The blue interface in the lower left corner is the feature module in the front-end VO. It continuously detects the feature points of adjacent frames and performs feature matching to calculate the camera motion pose. The red interface on the right displays the constructed indoor three-

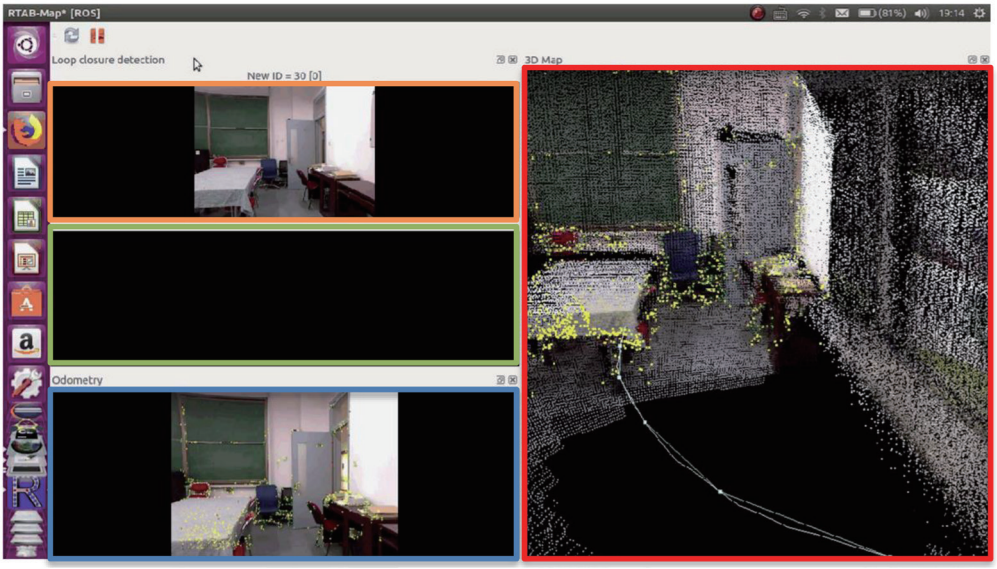


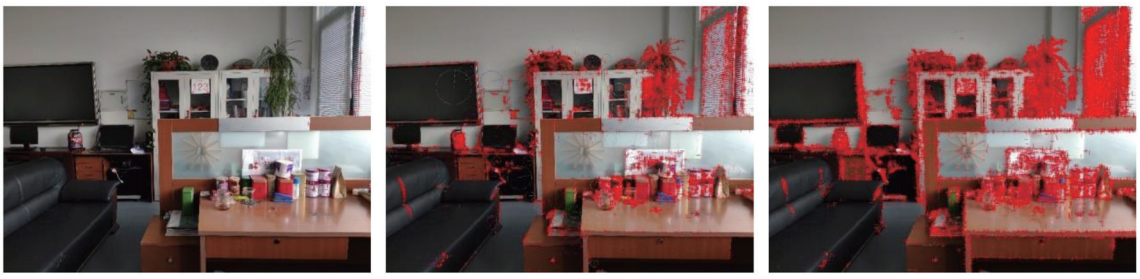
Fig. 5. (Color online) RTAB-MAP running effect.

dimensional point cloud map in real time, and the white line is the camera motion pose estimated by the system.

3.2 Image feature detection and matching

In this section, we compare and analyze three commonly used image feature point detection algorithms, namely, SIFT, SURF, and ORB, and image feature matching algorithms in the front-end VO of the RTAB-MAP system in terms of real-time performance and time efficiency. Firstly, SIFT, SURF, and ORB feature point detections are performed on the collected set of images through the OpenCV vision library. The results are shown in Figs. 6(a)–(c). The experimental effect of the feature detection algorithm is intuitively felt through the spatial distribution of the detected feature points. Then, 50 frames out of nearly 2000 frames of images are randomly selected from the experimental data for testing, and the average number of feature points detected by different feature detection algorithms and the time consumed are recorded by setting reasonable thresholds. The experimental results are shown in Table 1.

In the experiment of indoor map construction using RGB-D SLAM, the detection efficiency of the front-end feature points and the number of subsequent matches are very important for the overall execution efficiency of the algorithm. From the experimental results shown in Table 1, it is seen that the average detection time of each ORB feature is 0.0632 ms, which is the shortest among the three detection algorithms compared. This shows that ORB has the best real-time performance. SIFT feature detection takes the longest time. The number of detected feature points is the largest and the distribution is relatively uniform. SURF can extract relatively many feature points, but it takes much longer than ORB. Compared with SIFT features, the distribution of SURF features is mainly concentrated in the edge zone. The ORB feature distribution is relatively concentrated and the extraction speed is the highest. It is more suitable for the RGB-D SLAM system applied to the construction of real-time indoor maps.



(a) (b) (c)

Fig. 6. (Color online) Results of feature point detection: (a) ORB, (b) SURF, and (c) SIFT.

Table 1
Comparison of real-time performance and time efficiency of three feature detection algorithms.

Algorithm	Average feature point number/frame	Average time (ms)/feature point	Average time (ms)/frame
SIFT	54941	0.5289	29058
SURF	20011	0.3786	7576
ORB	528	0.0632	33

Two images were selected for feature matching experiments. The BF Matcher method is used to match the ORB features, and the Hamming distance is used to calculate the distance between the descriptors. To screen for mismatches, the screening condition is that the Hamming distance is less than twice the minimum distance. The experimental results are shown in Fig. 7. To reflect the real-time nature of the system, the FLANN method is used to match SIFT and SURF features. If the ratio of the nearest Euclidean distance to the next nearest distance is less than 0.6, the matching point pair is the correct matching point pair.

It can be seen from Fig. 7 that during the feature matching process, many crossing lines appear, that is, there are still many mismatches. Incorrect matching results seriously affect the accuracy of VO. For more accurate calculation results, the RANSAC algorithm is used to eliminate mismatches. The results are shown in Fig. 8. It can be seen that the cross lines in the matching results have been significantly eliminated, which effectively improves the accuracy of VO.

The number of feature detection points of the three algorithms, the number of pairs of rough matching points of the features, and the number of pairs of mismatching points rejected by the RANSAC algorithm were separately calculated to evaluate their performance, as shown in Table 2. We find that the SIFT features are reduced from 6358 pairs of points before rough matching to 4351 pairs of points after using the RANSAC algorithm. The SURF features are reduced from

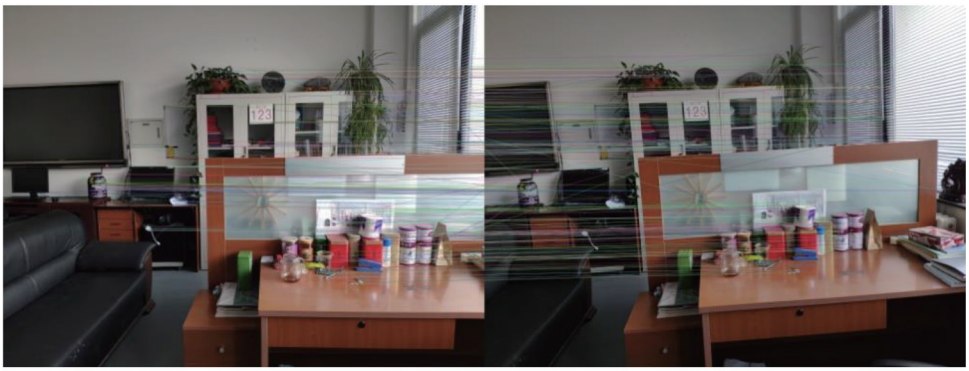


Fig. 7. (Color online) ORB feature matching using BF Matcher method.

Fig. 8. (Color online) Mismatch elimination using RANSAC algorithm.

Table 2
Comparison of feature matching performances of three algorithms and RANSAC.

Algorithm	Feature points		Rough match number	RANSAC match number
	Left image	Right image		
SIFT	54941	42869	6358	4351
SURF	20011	18156	3404	2597
ORB	528	502	324	209

3404 pairs of points before rough matching to 2597 pairs of points after using the RANSAC algorithm. The ORB features are reduced from 324 pairs of points before rough matching to 209 pairs of points after using the RANSAC algorithm. Therefore, by applying the RANSAC algorithm to feature matching, we can reject about one-third of the mismatches, which can effectively improve the accuracy of FVO.

To comprehensively analyze the experimental results, we collected multiple sets of images and conducted multiple experiments on the above process. According to the experimental results, ORB has the best real-time performance and is suitable for large-scale 3D reconstruction. SIFT detects the most uniform feature points with the highest stability and accuracy. SURF is slightly worse than SIFT but better than ORB. By using the RANSAC algorithm, we can effectively reject mismatches. Considering that the system has high requirements for real-time performance, it is an optimal strategy to select ORB feature matching and use the RANSAC algorithm to eliminate mismatches.

3.3 Performance and accuracy evaluation

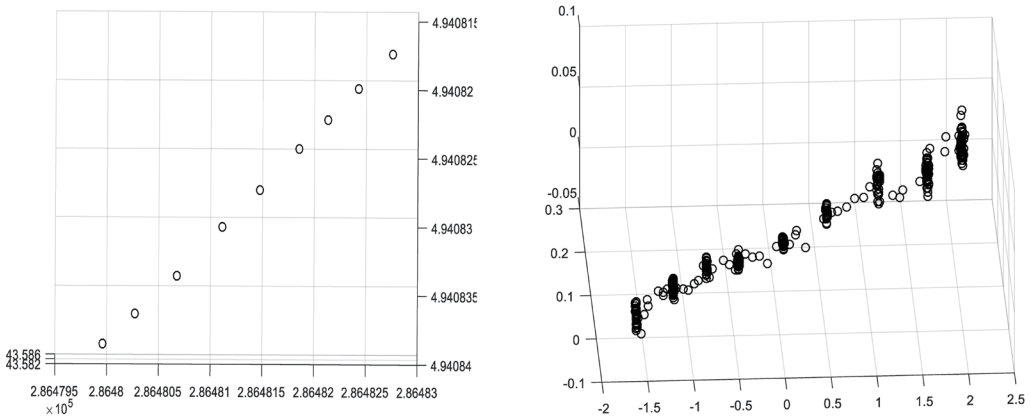
To evaluate the performance and VO accuracy of the system, we designed linear, curved, and closed-loop motion experiments in different environments with the help of a total station. The method is to use the total station to collect the coordinates of some points in the system trajectory and compare them with the corresponding point coordinates in the actual VO motion. The total station model is Leica TS30, which uses a prism-free measurement method with an accuracy of 2 mm + 2 ppm and a measurement range of 1000 m. To minimize the error, a reflective sheet is attached next to the optical center of the experimental camera.

The scene of the linear motion experiment is an office building corridor with dark lighting conditions and repeated textures. The movement route of the system is roughly a straight line. The coordinates of the control point C_0 measured using the total station are (286474.5786, 494087.7411, 42.4233). The coordinates of the points measured using the total station during the movement and the camera position data calculated by VO are shown in Table 3.

First, we used the total station to make nine straight points in the indoor corridor and calculated their X, Y, and H coordinates. Then, we moved the system from point 1 to other points and recorded their timestamps and coordinates. After the two coordinates were formatted and unified, we imported them into the MATLAB software to draw the distribution map of total station measurement points Fig. 9(a) and the VO track Fig. 9(b). At the same time, we calculated the relative error between the distance moved by the VO and the actual distance measured by the total station.

Table 3
Data statistics of total station measurement and VO test.

Point number	Total station measurement point coordinates (m)			VO result (m)		
	X coordinate	*Y* coordinate	*H* elevation	*X* coordinate	*Y* coordinate	*Z* coordinate
1	286482.7556	494081.7900	43.5850	−1.5585	0.0088	−0.0406
2	286482.4222	494082.0329	43.5845	−1.1418	0.0371	−0.0209
3	286482.1313	494082.2467	43.5838	−0.7807	0.0259	0.0009
4	286481.8542	494082.4614	43.5840	−0.4661	−0.0231	0.0229
5	286481.4765	494082.7545	43.5834	0.0584	0.0373	0.0113
6	286481.1133	494083.0336	43.5840	0.5452	0.0755	0.0246
7	286480.6752	494083.3618	43.5823	1.1530	0.1438	0.0209
8	286480.2720	494083.6750	43.5842	1.7147	0.1670	0.0198
9	286479.9597	494083.9177	43.5855	2.1058	0.1817	0.02265



(a) (b)

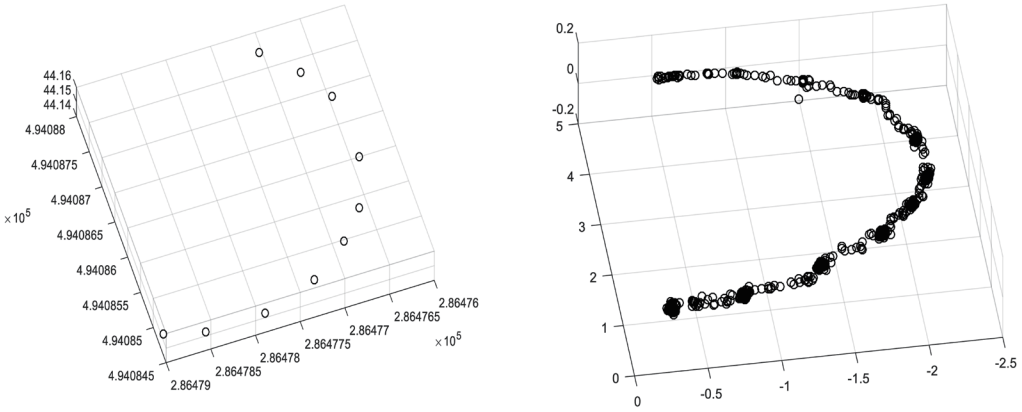
Fig. 9. (a) Distribution map of total station measurement points and (b) VO track.

The experimental results show that the distance the system moves from points 1 to 5 is 1.6019 m. The distance calculated by VO is 1.6179 m. The distance the system moves from points 1 to 9 is 3.5134 m, and the distance calculated by VO is 3.6689 m. The relative distance error rate of the VO is about 4.42%. The trajectory calculated by VO is roughly a straight line.

Similarly, the scene of the curve motion experiment is a conference room in an office building with sufficient indoor light and no dynamic interference. The course of the system is roughly semicircle. To compare the point results, the arc is sampled uniformly, and several points are set for comparison and analysis. The experimental results of the curve motion are shown in Figs. 10(a) and 10(b).

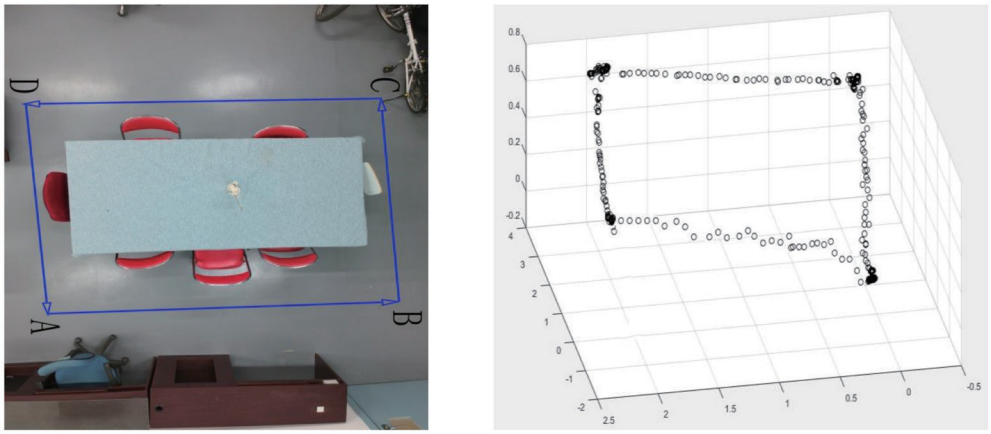
The experimental results show that the distance the system moves from points 1 to 5 is 2.1718 m. The distance calculated by VO is 2.1509 m. The distance the system moves from points 1 to 9 is 3.5484 m, and the distance calculated by VO is 3.5230 m. The relative distance error rate of the VO is about 1%. The trajectory calculated by VO is roughly a semicircle. The results show that the calculation results of VO on camera pose rotation are reliable.

In the same way, the scene of the closed-loop motion experiment is an office in a building with weak indoor light, rich texture, and no dynamic interference. The system's motion path



(a) (b)

Fig. 10. (a) Distribution map of total station measurement points and (b) VO track.



(a) (b)

Fig. 11. (a) Closed-loop motion experiment environment and route and (b) VO track.

is roughly rectangular. The experimental route follows a straight line from point A to point B, then B to C, C to D, and finally returns from point D to point A to form a closed loop. The experimental environment and route are shown in Fig. 11(a), and VO track results are shown in Fig. 11(b).

The experimental results show that the distance from point A to point B measured using the total station is 3.41 m, and the distance calculated by VO is 3.37 m. The actual distance from point B to point C is 2.32 m, and the distance calculated by VO is 2.24 m. To calculate the closing error, we finally returned to point A. The closing error of VO is (−0.1439, 0.1983, 0.2278). From the trajectory, the result is close to the true value.

4. Conclusions

We conducted an in-depth research on the implementation and accuracy evaluation of VO based on RTAB-MAP. The main conclusions are as follows:

(1) The RGB-D SLAM experimental platform is equipped with the Turtlebot2 robot and Kinect V1 vision sensor, which realizes the classic RTAB-MAP scheme in RGB-D SLAM.
(2) The three main feature detection and matching algorithms, namely, SURF, SIFT, and ORB, are compared in terms of real-time performance and time efficiency. The advantages and disadvantages of the three algorithms on the FVO feature module are illustrated.
(3) The RANSAC algorithm combined with feature matching completes the rejection experiment of feature point mismatches. The effectiveness of the RANSAC algorithm in VO is demonstrated. For the RTAB-MAP system, it is an optimal strategy to select the ORB feature matching and use the RANSAC algorithm to eliminate mismatches.
(4) Linear, curved, and closed-loop motion experiments were designed to evaluate the accuracy and performance of the RTAB-MAP VO. The experimental results show that the relative error and cumulative drift error of the VO are within 5%. It provides a useful reference for the development of mobile robot positioning technology.

Acknowledgments

This work was supported by the International Multi-Conference on Engineering and Technology Innovation 2019 in Kaohsiung.

References

1 D. Scaramuzza: Proc. 2009 IEEE Int. Conf. Robotics and Automation (IEEE, August, 2009) 1.
2 D. Nistér, O. Naroditsky, and J. R. Bergen: Proc. IEEE Conf. Computer Vision and Pattern Recognition (IEEE, 2004) 652–659.
3 C. Hane, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys: Image Vision Comput. **68** (2017) 14.
4 F. Steinbrucker, J. Sturm, and D. Cremers: Proc. IEEE Int. Conf. Computer Vision Workshops Piscataway (IEEE, 2011) 719–722.
5 D. G. Lowe: Int. J. Comput. Vision **60** (2004) 91.
6 H. Bay, T. Tuytelaars, and L. V. Gool: Proc. European Conf. Computer Vision (2006) 404–417.
7 E. Rublee, V. Rabaud, K. Konolige, and G. Bradski: Proc. IEEE Int. Conf. Computer Vision (2012) 2564–2571.
8 E. Rosten: Proc. European Conf. Computer Vision (2006) 430-443.
9 M. Calonder, V. Lepetit, C. Strecha, and P. Fua: Proc. European Conf. Computer Vision (2010) 778–792.
10 M. A. Fischler and R. C. Bolles: Commun. ACM. **24** (1981) 381.
11 A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse: IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007) 1052.
12 R. Hartley and A. Zisserman: Multiple View Geometry in Computer Vision (Cambridge University Press, Cambridge, 2004) 2nd ed.
13 G. Klein and D. Murray: Proc. IEEE & ACM Int. Symposium on Mixed & Augmented Reality (2007) 1–10.
14 C. G. Harris and M. J. Stephens: Alvey Vision Conf. **15** (1988) 10
15 E. Rosten and T. Drummond: 10th IEEE Int. Conf. Computer Vision (ICCV, 2005) 1.
16 H. P. Moravec: Proc. 7th Int. Joint Conf. Artificial Intelligence (IJCAI, 1981) 785–790.
17 K. A. Acharya and R. V. Babu: 2013 Fourth National Conf. Computer Vision, Pattern Recognition, Image Processing and Graphics (IEEE, 2013) 1–4.
18 Z. Yonglong, M. Kuizhi, J. Xiang, and D. Peixiang: 2013 IEEE 10th Int. Conf. High Performance Computing and Communications & 2013 IEEE Int. Conf. Embedded and Ubiquitous Computing (IEEE, 2013) 1351–1358.
19 G. Xiang and Z. Tao: Visual SLAM Fourteen Lectures-From Theory to Practice (China Machine Press, Beijing, 2017) 1st ed., Chap. 7.
20 M. Muja and D. G. Lowe: IEEE Trans. Pattern Anal. Mach. Intell. **36** (2014) 2227.

About the Authors

Bo Hu received his B.S. degree from Beijing University of Civil Engineering and Architecture, China, in 2018. He is studying for his M.S. degree at Beijing University of Civil Engineering and Architecture, China. His research interest is on visual simultaneous localization and mapping.

He Huang received his B.S. degree from Wuhan University, China, in 2000 and his M.S. and Ph.D. degrees from Sungkyunkwan University, South Korea, in 2004 and 2010, respectively. Since 2010, he has been a Lecturer and Associate Professor at Beijing University of Civil Engineering and Architecture, China. His research interests are on autonomous driving, high-precision navigation maps, and visual navigation and positioning.