ORB-SLAM-based Tracing and 3D Reconstruction for Robot using Kinect 2.0

Qiang LV[1], Huican LIN[1], Guosheng WANG[1], Heng WEI[1], Yang WANG[2]

1. Academy of Armored Force Engineering, Beijing 100072, China
E-mail: 18606921232@163.com

2. 94891 Troop, Beijing 100076, China

Abstract: We aim to track robot location and provide dense 3D reconstruction while exploring environment in real time, based on one of the best SLAM algorithm called ORB-SLAM, which accurately estimate the camera poses and sparse 3D map of the environment based on images. However, the sparse map can't be applied to obstacle avoidance and navigation. We base on octrees and use probabilistic occupancy estimation, extending the mapping technique of ORB-SLAM and builds 3D reconstruction which can be used in robot field, the validity of this method is verified via benchmark datasets. Finally, the extended SLAM system is applied to a handheld Kinect 2.0, experiments show that the camera pose is accurately tracked and octomap is completed in real time.

Key Words: ORB-SLAM; tracking; 3D reconstruction; robotics; octomap

1 INTRODUCTION

In the field of mobile robot, SLAM (Simultaneous Localization And Mapping) is a very basic and critical problem, is one of the most popular research directions in the field of robotics, and has been widely concerned and studied in the past 30 years[1]. Researchers have proposed a large number of SLAM systems, including a variety of sensors, optimization techniques and map description. The 3D map of the workspace is very important for robot to realize the localization, obstacle avoidance, navigation and autonomous task. Especially when the robot works in complex and dynamic environment, it is important to quickly generate and maintain a three-dimensional map of the workspace via onboard sensors.

VSLAM (Visual SLAM) system uses the camera as main sensor to estimate the position and environment map of the robot[2], PTAM (Parallel Tracking And Mapping) firstly divided tracking and mapping into two parallel threads in monocular SLAM[3], SVO is a semi-direct visual odometry method that can run in embedded system in real time [4], LSD-SLAM reconstructs 3D environment and semi-dense depth map using direct registration based on monocular vision[5]. RGBD SLAM combines the color image and depth image to estimate the camera pose and reconstruction of the 3D environment [6]. ORB-SLAM is one of the most successful SLAM system based on sparse feature, which is designed to solve the problem of camera tracking in real time[7]. However, the sparse map can't be applied to tasks such as obstacle avoidance, path planning and autonomous navigation for robots.

Based on ORB-SLAM, we use the color image and depth image provided by RGBD sensor to expand the sparse map constructed by the original system into dense point cloud map. Furthermore, we propose to use octree structure [8] to reconstruct a octree map of the environment for robot application. Finally, the method is successfully applied to handheld camera, experiments show that the method proposed in this paper can accurately track the pose of the camera and construct the 3D map of the environment under the challenging conditions and long-distance trajectory.

2 RELATED WORKS

RGB-D sensor provide high-precision color and depth images at the same time. Using color information and dense depth information, the 3D reconstruction can be estimated based on visual feature consistency, optical measurement consistency and ICP. Henry et al. first proposed a SLAM system that uses RGB-D cameras, combined with GICP (Generalized ICP)[9] and visual features to create and optimize pose[10][11]. Huang et al [12] proposed an odometry method based on tracking key points. To solve the problem that rotation may cause strenuous motion of distant features, first estimate the rotation and then search the key points in the small window, using the gray value of the patches match. KinectFusion is a real-time object surface reconstruction method based on voxel grids[13]. It needs powerful GPU to ensure real-time performance. In addition, the lack of the ability to reduce the cumulative error through closed-loop detection, the scale of the voxel grid is a cubic relationship to memory consumption, so it can only be used for small work space.

Kerl et al. first register consecutive RGB-D frames by minimizing the photometric error to estimate the camera motion from images[14], and propose a dense visual SLAM method that minimizes both the photometric and the depth error over all pixels[15]. Their approach gets high accuracy results, but fails in the "fr1 floor" sequence due to overly dependent visual odometry, where the algorithm fails when a short sensor terminates in the sequence.

Felix Endres proposed RGBD-SLAM, using the same hardware as Kerl evaluation experiments, but shows better robustness and accuracy16]. The system proposed by Felix Endres uses RGB-D sensors to extract visual features from color images and use depth images to determine the three-dimensional position of visual features, and uses RANSAC to estimate the closed loop between the associated visual feature points[17] and uses non-linear methods to optimize the pose. Finally, the voxel 3D map of the environment can be applied to obstacle avoidance, path planning and navigation of robots. In order to improve the reliability of estimation, the environment measurement model(EMM) is introduced to evaluate the accuracy of the frame to frame transform, and the low accuracy estimation is eliminated to obtain the robustness under complex scenes.

The spatial location of the visual feature is calculated from the matched sensor depth data, which is susceptible to noise interference because the visual feature tends to locate at the edge of the object. In order to reduce the estimation error, a plane-based SLAM system was proposed with reliable depth-value selection and feature extraction [18]. Plane point feature is helpful to improve the accuracy and robustness of traditional ICP, while maintaining reasonable computing consumption to ensure real-time. The point cloud map constructed by this system has the common problem of resource consumption and maintenance, which is not suitable for robot application. The ORB-SLAM constructed by Mur-Artal et al is one of the best VSLAM system based on visual feature, which simultaneously estimate camera pose and reconstruct sparse map in small scale and large scale, indoor and outdoor environments, including loop closing and relocation, and full automatic initialization, its accuracy and robustness is difficult to go beyond. The authors propose a probabilistic semi-dense map construction technique, processing key frames provided by the monocular SLAM system which reconstructs the surface texture and object contour in real-time without GPU acceleration[19], however, it is still not suitable for robotic applications.

3 ORB-SLAM OVERVIEW

3.1 Tracking

ORB extract feature points based on oriented FAST with direction, and binary descriptors are used to improve the speed, which has good scale invariance and noise suppression performance[20]. The system uses the ORB algorithm to detect the keypoints and calculate the descriptors of keypoints. When the tracking of the previous frame is successful, the initial position of the camera is estimated from the previous frame. Each keypoint of the previous frame and its descriptor has its associated point, the matched point is searched in the small region near the corresponding location of the current frame. The initial corresponding point is optimized by the direction consistency check[21]. At this time, a series of correspondence is obtained, so PnP solution can be solved in the RANSAC scheme to calculate the initial camera pose [22]. When the tracking of the previous frame fails, the current frame is described by the visual bag model and is searched for in the visual word bag database made up of key frames to realize global relocation.

The initial pose estimation obtained by inter-frame registration has accumulated errors, and the current pose can be further optimized using the constructed map. The keypoints of the previous keyframes are projected into current frame, and the residual error can't be overlapped with the corresponding keypoints to construct the residual of the estimated position. The position of the keypoints obtained after optimization is theoretically more accurate than the previous estimation of the camera pose, and thus the camera pose and the position of the keypoints in the map can be optimized. Increasing the frame rate of the camera will improve accuracy and robustness. However, maintaining the data of all the images requires a lot of resources but serious redundancy. In order to track the movement of the camera more robustly, it determines whether the current frame can be a new key frame and insert the appropriate keyframe.

3.2 local mapping and optimization

If there are errors in the map points, it will adversely affect the data association. Therefore, it is necessary to check the map points in the newly inserted keyframes to eliminate the non-conforming map points to improve the robustness of the algorithm. Ignoring those that do not meet the constraints of the polar line, it is necessary to check its re-projection error and scale consistency in the relevant keyframes. Bundle adjustment (BA) optimized objects include the currently processed keyframe and its associated keyframes, including all of the map points in those keyframes. The LM (Levenberg Marquardt) method in g2o is used to solve the Huber robust cost function[23]. Since the complexity of BA is the third order of the number of keyframes, deleting redundant keyframes effectively will speed up BA. Therefore, redundant key frames should be detected and deleted during the local mapping process. In the same scenario, the number of key frames is not increased with the repeated motion of the camera, and the increase of the key frame only occurs when the camera enters the new scene, which is beneficial to the long time composition.

3.3 Loop closing

ORB-SLAM computes the similarity between the bag of visual words vector of keyframes K_i and covisibility keyframes, and the minimum score $s_{\min}$ is recorded. Then it computes the similarity to other keyframes in the map, ignoring any keyframes below the score. If detecting consecutively three candidates are consistent, the system accepts current keyframes as loop candidate. A similar transformation between the current keyframe and the loop keyframe is calculated, which is the accumulated error in the loop. The overlapping map points are merged, and new edges are inserted in the covisibility graph to connect the closed loop. It corrects the current keyframe pose, and then corrects all the keyframes and map points in the loop. All keyframes will generate new edges to link the re-optimized keyframes back into loops.

4 EXTENDING RECONSTRUCTION

The ORB-SLAM estimates the camera pose and the sparse map in real-time with high precision. We improve the mapping method in the original system. Finally, an octree map is constructed, which can be applied to robot obstacle avoidance, navigation and interactive operation.

Create a point cloud map. The environment can be described as a series of points: $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots\cdots, \boldsymbol{x}_n\}$, where $\boldsymbol{x}_i = [r, g, b, x, y, z]$, respect for the color and location of point respectively. For RGB-D sensor, the color information is stored in a color image, and the absolute scale position can be derived from the data of the depth image.

According to pinhole camera model, the relationship between position in the world and pixel coordinates can be expressed as

$$s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{C} \cdot \left(\mathbf{R} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \mathbf{t} \right) \tag{1}$$

Where, $\mathbf{C}$ is the camera internal matrix, $\mathbf{R}$ is the camera rotation and $\mathbf{t}$ is translation, s is the depth of scale factor. Based on the high accuracy and robustness of ORB-SLAM, using RGBD sensor, the data in the keyframe is applied to create point cloud map, while programming is mainly dependent on point cloud library. The i element in the point cloud map $\mathbf{p}_i$ consist the position $p_{i,z}$ $p_{i,x}$ $p_{i,y}$ and the color $p_{i,z}$ $p_{i,x}$ $p_{i,y}$, which can be calculated from the depth image and the color image:

$$\begin{bmatrix} p_{i,z} \\ p_{i,x} \\ p_{i,y} \end{bmatrix} = \begin{bmatrix} d \\ (n - c_x) * p_{i,z} / f_x \\ (m - c_y) * p_{i,z} / f_y \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} p_{i,b} \\ p_{i,g} \\ p_{i,r} \end{bmatrix} = \begin{bmatrix} \mathbf{I}(m, n*3) \\ \mathbf{I}(m, n*3+1) \\ \mathbf{I}(m, n*3+2) \end{bmatrix} \tag{3}$$

Where $d = \mathbf{D}(m,n)$ is the depth of the image, $\mathbf{D}$ and $\mathbf{I}$ denote the depth image and the color image respectively, m and n represent the pixel coordinates in the image. f_x , f_y , c_x and c_y are the camera internal parameters, obtained by calibrating the camera calibration.

We compute the point cloud of all the pixels in the image and form the point cloud map of the key frame using equation (2) and (3), which needs to be transformed to the local coordinate system through the pose of the camera of current keyframe. In addition, the camera pose was optimized in the SLAM system using the following equation:

$$\{\mathbf{R}, \mathbf{t}\} = \arg\min \sum_{i \in \chi} \rho\left(\left\| \mathbf{x}^i - \pi\left(\mathbf{R}\mathbf{X}^i + \mathbf{t} \right) \right\|_{\Sigma}^2 \right) \tag{4}$$

Where $\mathbf{R}, \mathbf{t}$ denote the rotation and translation of the camera respectively.

This is done for each new keyframe, which together form the point cloud map of the robot workspace. As mentioned above, a point cloud map consists of a large number of points consisting of all the pixels of all the keyframes that contain three-dimensional position and color information. As the size of map increases, maintenance becomes more difficult. At the same time, for robot applications, such a large-scale space is redundant and not practical.

Create an octree map. Compared with the point cloud map, the octree map store the map in an octree hierarchical way, which is able to update the reconstruction efficiently and models the date consistent while keeping the memory requirement at a minimum, in addition the resolution is adjustable. More importantly, the octree map can be applied to obstacle avoidance, path planning and navigation for robot.

The leaf nodes in the octree structure represent the occupancy state of the space in the form of probability. The probability $P(n \mid z_{1:t})$ of a leaf node n to be occupied given the observed data $z_1, z_2, \cdots z_t$ at time $1, 2, \cdots t$ is estimated according to

$$P(n \mid z_{1:t}) = \left[1 + \frac{1 - P(n \mid z_t)}{P(n \mid z_t)} \frac{1 - P(n \mid z_{1:t-1})}{P(n \mid z_{1:t-1})} \frac{P(n)}{1 - P(n)} \right]^{-1} \tag{5}$$

The common assumption of a uniform prior probability leads to $P(n) = 0.5$, the $\log it(\bullet)$ transformation is introduced

$$\log it(p) = \ln\left(\frac{p}{1 - p} \right) \tag{6}$$

By using equation(6),

$$\begin{aligned} \ln\left[\frac{P(n \mid z_{1:t})}{1 - P(n \mid z_{1:t})} \right] &= -\ln\left[\frac{1 - P(n \mid z_t)}{P(n \mid z_t)} \frac{1 - P(n \mid z_{1:t-1})}{P(n \mid z_{1:t-1})} \frac{P(n)}{1 - P(n)} \right] \\ &= \ln\left[\frac{P(n \mid z_{1:t-1})}{1 - P(n \mid z_{1:t-1})} \right] + \ln\left[\frac{P(n \mid z_t)}{1 - P(n \mid z_t)} \right] \end{aligned} \tag{7}$$

With $L(n \mid z_t) = \ln\left[\frac{P(n \mid z_t)}{1 - P(n \mid z_t)} \right]$, Eq.(5) can be rewritten as

$$L(n \mid z_{1:t}) = L(n \mid z_{1:t-1}) + L(n \mid z_t) \tag{8}$$

From Eq.(8), we can see that for any point in the three-dimensional space, the new observations are directly added to the last observations, and the updating method is flexible and easy to implement. The probability of occupying the parent node in an octree map can be calculated from the values of the child nodes. There are two ways of averaging $\bar{l}(n) = \frac{1}{8} \sum_{i=1}^{8} L(n_i)$ and maximizing $\bar{l}(n) = \max_i L(n_i)$ for computing the probability of a parent node. In the process of creating the point cloud map, we construct the octree map of the 3D environment in real time based on the octree map library.

5 EXPERIMENTAL EVALUATION

In order to verify the robustness, accuracy and real-time performance of the proposed method, TUM (Technische Universität München) RGB-D datasets[24] was used to evaluate the performance. In addition, the method was applied to track a handheld camera and reconstruct the workspace.

5.1 Benchmark datasets

The RGB-D dataset contains several sequences of data sets captured by two Kinect sensors and one Xtion Pro Live sensor. The ground truth of all sequences of datasets are provided by a high-precision motion capture system. In this paper, we use the root mean square of absolute trajectory error (ATE) to evaluate the deviation of the trajectory estimated by this method. For the estimated trajectory and the corresponding ground truth, define

$$\mathrm{ATE}_{\mathrm{RMSE}}\left(\hat{X}, X\right) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left\|\left(\hat{x}_i\right)-\left(x_i\right)\right\|^2} \qquad (9)$$

Eq.(9) shows the root mean square of the Euclidean distance between the ground truth and the estimated pose. In order to make the error metric independent of the coordinate system used to represent the trajectory, the pose estimation is registered, so the above errors are minimal, and the correspondence between pose estimation is established using time stamps.

The map error and trajectory error of a particular dataset depends mainly on the definition of the given scene and the respective error functions. Although the error metric selected in this paper can't directly measure the quality of the map, it can be considered that the error of the map is related to the trajectory error. This is determined by the characteristics of the SLAM system, since the estimated pose and reconstructed octree map are done at the same time, and the two complement each other, so the degree of deviation of the trajectory is related to the degree of map deviation.

5.2 Precision and robustness

Using the same datasets and evaluation criteria, the accuracy and robustness of the RGBD SLAM method proposed in [6] are compared, and the results are shown in Table 1 and Figure 1. It can be seen from Figure 1 that the error of the method proposed in this paper is less than that of RGBD SLAM, especially in the case of large scale and no closed loop. The main reason is that compared with RGBD SLAM using ICP for pose optimization, this paper uses BA to optimize the camera pose and map points, which is conducive to enhance the accuracy and robustness of the algorithm.

Table1. Accuracy comparison between ours and [6]

sequences	fr1_desk ours	fr1_desk [6]	fr1_room ours	fr1_room [6]	fr2_desk ours	fr2_desk [6]	fr2_large_no_loop ours	fr2_large_no_loop [6]
RMSE	0.015	0.026	0.065	0.087	0.008	0.057	0.17	0.86
frames	573	547	1352	1324	2893	2866	3299	3256
time	42.1	35.9	93.9	94.3	200	390.3	224.1	478.6
rates	13.6	15.2	14.4	14.0	14.5	7.34	14.7	6.8

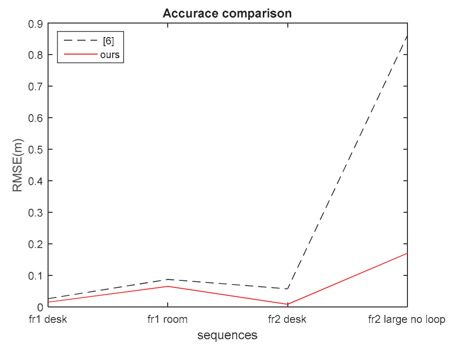
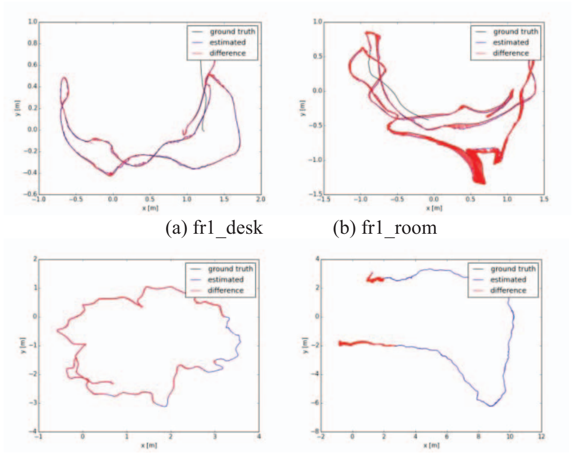


Fig.1 Evaluation of accuracy

In addition, the camera trajectory estimated by this paper is compared with ground truth provided by the dataset, which was measured by the high accuracy motion capture system. The comparison shows that the method is effective and accurate(see Figure 2).



(a) fr1_desk (b) fr1_room

(c) fr2_desk (d) fr2_large no loop

Fig.2 Evaluation of ATE

5.3 Real-time

Using the datasets to verify performance, this paper uses the Intel 4-core i5 CPU, 4G RAM, the paper [6] with i7 CPU and GTX570 graphics. Table 1 records the time-consuming situation shown in Figure 3, we can see the method presented in the lower processor performance, the algorithm runs faster. This is mainly due to the addition of to eliminate and optimize the function of the keyframe in Section 3.2, especially in the environment of large-scale, long-time composition of the situation, the advantage is more obvious.

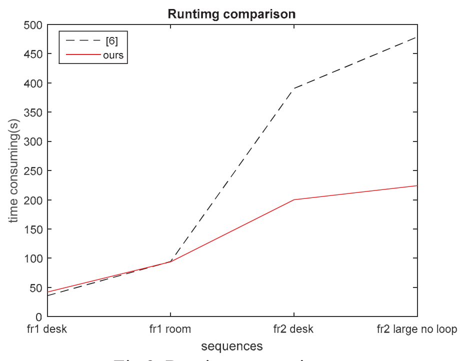
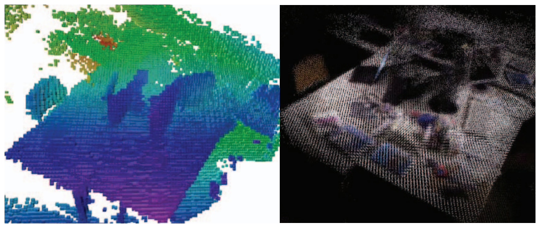
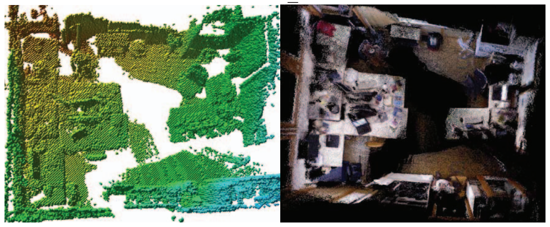


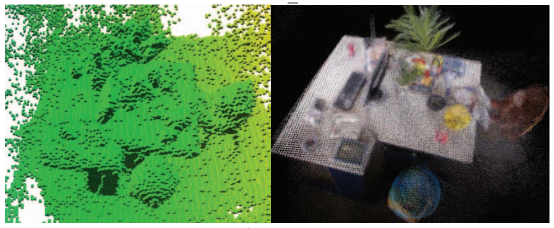
Fig.3 Runtime comparison

5.4 3D reconstruction

The ORB-SLAM focuses on improving the accuracy, real-time and robustness of the camera pose estimation. The sparse map constructed is not suitable for robot application. In this paper, we use the RGBD sensor to construct dense point cloud map and octree map which can be applied to the robot field, and overcome the scale blur problem of monocular camera. As shown in Figure 4, octree map and dense point cloud map created by running datasets using the method proposed in this paper. Storage of four octree maps and four point clouds map occupies 0.77MB and 130MB respectively, which shows that the octree map is more compact and easy to maintain.


（a）fr1_desk


（b）fr1_room


（c）fr2_desk

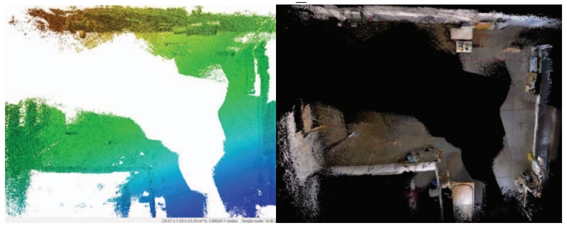

（d）fr2_large no loop

Fig.4 Octree map and point cloud map

5.5 Handheld Kinect applications

Finally, the proposed system has also been successfully used for online mapping with handheld Kinect 2.0 camera. We use an Intel quad-core i7 CPU, 16G RAM and 256G SDD for experiments. The depth images and colour images of Kinect 2.0 are read by the computer via USB3.0 interface, and the image data is published in the form of ROS(robot operating system) topic[25].The system proposed in this paper subscribes to the data of the topic, and completes the pose estimation and tracking, feature map construction and optimization, closed-loop detection and octree map construction. All data acquisition and processing is done by the CPU, enabling the construction of 3D maps in real time (see Figure 5). With the increase of the map scale, the real-time performance of the algorithm is reduced. It may be because the map resolution is as high as 0.02m, for improving the visualization of point cloud map and octree map, which is higher than the accuracy required for mobile robot navigation.

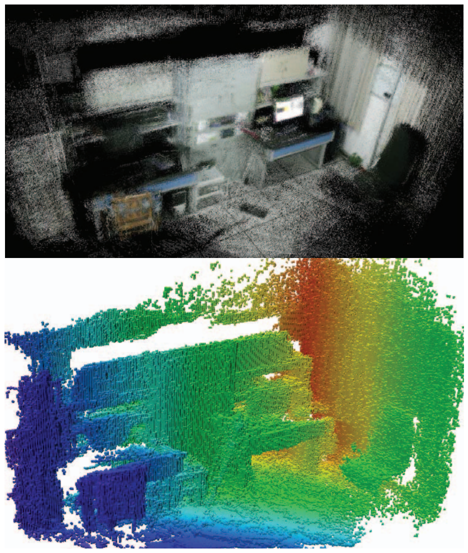


Fig.5 3D map built by handheld Kinect 2.0

6 CONSLUSION

In this paper, we proposed a novel visual SLAM system for RGB-D camera based on ORB-SLAM. Compared with the original system, we extend the 3D reconstruction method and reconstruct the environment 3D point cloud map and octree map which can be used in the robot field. In addition, the accuracy and speed of this paper is better than RGBD SLAM system. The next step is to automatically adjust the map resolution based on hardware resources to improve the speed of the algorithm, and apply the octree map to obstacle avoidance and navigation of mobile robots.

REFERENCES

[1] Thrun S, Burgard W, Fox D. Probabilistic robotics[M]. Cambridge, USA: MIT Press, 2005.

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

[3] Klein G, Murray D. Parallel tracking and mapping for small AR workspaces[C]//Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on. IEEE, 2007: 225-234.

[4] Forster C, Pizzoli M, Scaramuzza D. SVO: Fast semi-direct monocular visual odometry[C]//2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2014: 15-22.

[5] Engel J, Schöps T, Cremers D. LSD-SLAM: Large-scale direct monocular SLAM[M]//Computer Vision–ECCV 2014. Springer International Publishing, 2014: 834-849.

[6] Endres F, Hess J, Sturm J, et al. 3-D mapping with an RGB-D camera[J]. Robotics, IEEE Transactions on, 2014, 30(1): 177-187.

[7] Mur-Artal R, Montiel J M M, Tardós J D. Orb-slam: a versatile and accurate monocular slam system[J]. IEEE Transactions on Robotics, 2015, 31(5): 1147-1163.

[8] Hornung A, Wurm K M, Bennewitz M, et al. OctoMap: An efficient probabilistic 3D mapping framework based on octrees[J]. Autonomous Robots, 2013, 34(3): 189-206.

[9] Segal A, Haehnel D, Thrun S. Generalized-ICP[C]//Robotics: Science and Systems. 2009, 2(4).

[10] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In Proc. of the Intl. Symp. on Experimental Robotics (ISER), Delhi, India, 2010.

[11] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments. Int. Journal of Robotics Research, 31(5): 647–663, April 2012.

[12] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an rgb-d camera. In Proc. of the Int. Symposium of Robotics Research (ISRR), Flagstaff, Arizona, USA, Aug. 2011.

[13] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pages 127–136, 2011.

[14] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. In Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), May 2013.

[15] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2013.

[16] Endres F. Robot Perception for Indoor Navigation[D]. , 2015.

[17] Zuliani M. RANSAC for Dummies[R]. Santa Barbara, USA: Vision Research Lab, University of California, 2008.

[18] Gao X, Zhang T. Robust RGB-D simultaneous localization and mapping using planar point features[J]. Robotics and Autonomous Systems, 2015, 72: 1-14.

[19] Mur-Artal R, Tardós J D. Probabilistic semi-dense mapping from highly accurate feature-based monocular SLAM[J]. Proceedings of Robotics: Science and Systems, Rome, Italy, 2015.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: an efficient alternative to SIFT or SURF,” in Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV), vol. 13, 2011.

[21] R. Mur-Artal and J. D. Tard´os. Fast relocalisation and loop closing in keyframe-based SLAM. In IEEE International Conference on Robotics and Automation (ICRA), 2014.

[22] Lepetit V, Moreno-Noguer F, Fua P. Epnp: An accurate o (n) solution to the pnp problem[J]. International journal of computer vision, 2009, 81(2): 155-166.

[23] Kümmerle R, Grisetti G, Strasdat H, et al. g2o: A general framework for graph optimization[C]//Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011: 3607-3613.

[24] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in Proc. of the International Conference on Intelligent Robot Systems (IROS), 2012.

[25] Martinez A, Fernández E. Learning ROS for robotics programming[M]. Birmingham: Packt Publishing Ltd, 2013.