2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Submap-based Pose-graph Visual SLAM: A Robust Visual Exploration and Localization System*

Weinan Chen†, Lei Zhu†, Yisheng Guan†, C. Ronald Kube‡, Hong Zhang*‡

†Biomimetic and Intelligent Robotics Lab, Guangdong University of Technology, Guangzhou, China
‡Department of Computing Science, University of Alberta, Edmonton, Canada

***Abstract*— For VSLAM (Visual Simultaneous Localization and Mapping), localization is a challenging task, especially for some challenging situations: textureless frames, motion blur, etc.. To build a robust exploration and localization system in a given space, a submap-based VSLAM system is proposed in this paper. Our system uses a submap back-end and a visual front-end. The main advantage of our system is its robustness with respect to tracking failure, a common problem in current VSLAM algorithms. The robustness of our system is compared with the state-of-the-art in terms of average tracking percentage. The precision of our system is also evaluated in terms of ATE (absolute trajectory error) RMSE (root mean square error) comparing the state-of-the-art. The ability of our system in solving the "kidnapped" problem is demonstrated. Our system can improve the robustness of visual localization in challenging situations.**

Keywords: Monocular VSLAM, Submap-based Back-end, Robustness

I. Introduction

Many robot applications require a mobile robot to explore an unknown environment and construct its model. Environment exploration with vision is popularly studied in robotics via VSLAM. However, the explored space in this study is given and limited. In many scenarios, we are able to explore a certain space repeatedly, especially for the indoor applications. We refer to such an exploration as "dense exploration" in this paper.

For the keyframe-based VSLAM systems, resolving both the environment model and localization of the robot at the same time is extremely difficult, and has been a focus of robotics research for over 20 years. In the case of indoor VSLAM, there are a number of challenges such as textureless walls, motion blurred frames and illumination changes [1]. Even given enough time, the existing VSLAM systems [2][3][4] do not work all the time, since they rely on a back-end with a single graph and are unable to maintain working continuously. In contrast, by using multiple subgraphs, it is much easier for the robot to maintain tracking in one of the local subgraphs than in a global graph.

Instead of a single graph, a multiple submap graph is used for robustness improvement in this study. Submap-based graph is an approach that represents a complete environment using multiple subgraphs, which has been studied for some time [5][6][7]. Also, the merging of submaps is a key problem in submap-based systems, and many related studies have been proposed [8][9][10]. As a VSLAM system, we try to merge the submaps using place recognition [11][12] in the front-end and a robust optimization [13][14] in the back-end.

A submap-based VSLAM system is introduced in this paper. A pose-graph containing multiple submaps is built and maintained. To work with such a back-end, a multi-constraint front-end is designed and introduced. As evaluated by the experiments, our system can improve the robustness and has a better precision than the state-of-the-art. In addition, the ability of solving the "kidnapped" problem with the map built with the proposed termination-condition is demonstrated.

II. System Overview

The framework of our system is shown in Fig. 1, where the red boxes are the inputs/outputs and the green boxes are the decisions (the same representations will be applied to all the flow charts in this paper). As a VSLAM system, it can be divided into two parts: front-end and back-end. To get an optimized performance of the dense exploration, a multi-constraint front-end and a submap-based back-end is proposed to improve the robustness and reliability.

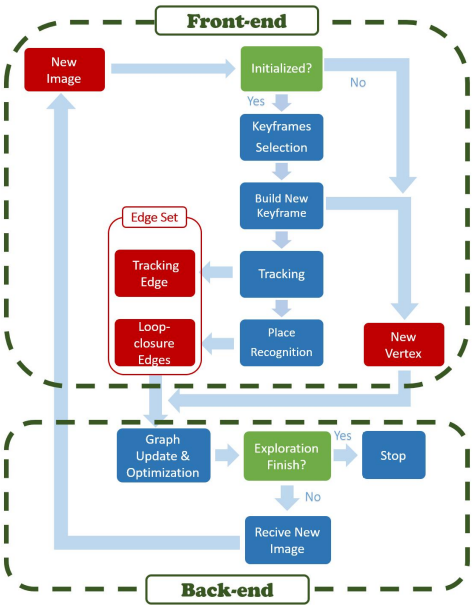

Fig. 1: Framework of our system

* The work in this paper is supported by the National Natural Science Foundation of China (61603103, 61673125), the Natural Science Foundation of Guangdong of China (2016A030310293), and the Major Scientific and Technological Special Project of Guangdong of China (2016B090910003).

The basic idea of the front-end is: for a new keyframe, we try to build an edge set using place recognition and tracking. After receiving a new frame, if the system has been initialized (the library of keyframe is not empty), we select the keyframe and store it; if not, we take the frame as the first keyframe directly. For a keyframe, we try to build two types of constraints: "tracking edges" from the last keyframe and "loop-closure edges" from the place recognition module. To calculate the constraint, a feature-based method is applied to find the relative pose between two frames, and a visual BoW (Bag-of-Word) approach is used for place recognition.

As for the back-end, a pose-graph is built and maintained. Once no edge can be established for a new keyframe, a new submap is built and we try to maintain tracking in that submap. Such a mechanism keeps the system "alive" even tracking lost happens. We merge the submaps by using the loop-closure edges, and the switchable factors are added to those loop-closure edges to improve the robustness of submap merging.

In addition, an exploration termination-condition is proposed to prevent infinite exploration and evaluate the completeness, which takes the density of vertices into consideration.

III. Multi-constraint Front-end

In the front-end, a graph is built with camera poses, tracking edges and loop-closure edges. To calculate the constraints between frames, a feature-based tracking method is designed, and a strict condition is performed to reject the low quality edges. As for place recognition, an image global descriptor is extracted and applied for image matching. An example is shown in Fig. 2. By considering more keyframes for localization in the front-end, the robustness can be improved.

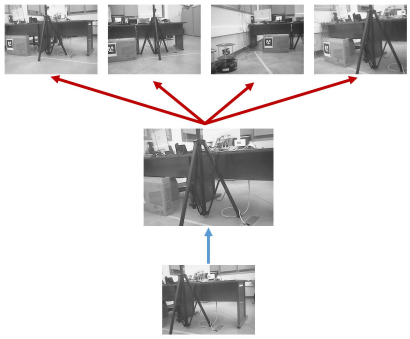

Fig. 2: An example of multi-constraint front-end, where the blue arrow is the tracking edge from the last keyframe, and the red arrows are the loop-closure edges connected to the place recognition results.

A. Keyframe selection

As a keyframe-based VSLAM system, we select the keyframes from the image sequence. There are three conditions for inserting keyframes:

(1) The normal of the relative transformation between the current frame and the last keyframe should be larger than a given threshold, which means the current image is different from the last keyframe spatially.

(2) The distance between the global descriptor of the current frame and the last keyframe should be larger than a given value to reject the frames that are too similar to the last keyframe in appearance.

(3) When neither tracking edge nor loop-closure edge can be established for a frame, the frame is "lost", this frame is treated as the first keyframe of a new submap.

The role of the first condition is similar to that of the second condition. However, considering the normalization of the translation in the relative pose calculation, the image similarity is considered in the second condition. In addition, when the library of keyframe is empty, the captured image is selected for initialization.

B. Tracking

The tracking edge is the relative pose between the current frame and the last keyframe. We calculate the relative pose through a two-frames feature-based pipeline: feature detection, feature matching, fundamental matrix finding and essential matrix decomposing. In addition, a verification is designed to reject a low quality result.

A feature detection method similar to [15] is used, a grid is placed on an image during feature detection, for an even feature distribution. The FAST detector and ORB descriptor are selected for their efficiency. Also, RANSAC [16] is applied to fundamental matrix finding and essential matrix decomposing, and the size of inlier features s is recorded. The pipeline is illustrated in Fig. 3.

To improve the precision of tracking edges added to the graph, we set a strict condition with respect to s to accept the high quality result only. If s is large enough, the constraint is accepted; otherwise, no tracking edge is established. However, thanks to the design of multi-constraint front-end and submap-based back-end, the system still updates the graph even if no tracking constraint can be built.

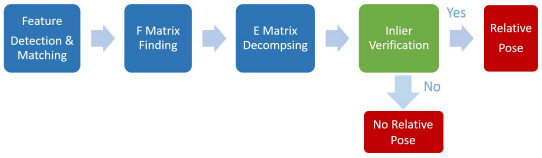

Fig. 3: Pipeline of constraint building

C. Place Recognition

Loop-closure edges are established by place recognition, which finds the places that have been visited before. A global descriptor of each frame is extracted by the DBoW2 [12] descriptor. To decrease the computation time, the place recognition results are simply obtained by ranking the distance of global descriptors between the current frame and all the existing keyframes. Only the keyframes that are "far enough from the current frame" are considered, which means a distance between the index of current frame and the place recognition results is required.

After global descriptor matching, we pick the first k matching results as the potential loop closures, and all of them are sent to loop closure verification before inserting the constraints. For the loop closure verification, we do the same thing as is mentioned in constraints building. However, a lax condition with respect to s is executed for adding more loop-closure edges to increase the possibility of submap merging, instead of a strict condition while establishing tracking edges.

As is known, the relative pose between two the monocular visual observations has an unknown scale. Therefore, we store more than one matching result of place recognition and build multiple constraints for a new frame. By least-square optimization, the estimation error caused by the different unknown scales can be iteratively minimized.

IV. Submap-based Back-end

After keyframe insertion and constraint calculation, a pose-graph is built. To overcome the tracking failure and get a robust performance, a submap-based graph is built. We optimize the graph using the edges with a weighted information matrix, and the switchable factors close the outlier loop-closure edges during optimization. The basic idea of submap-based graph is illustrated in Fig. 4.

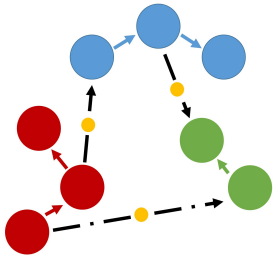

Fig. 4: Basic idea of submap-based graph. Three submaps (red nodes, blue nodes and green nodes) are built, the solid arrows are the tracking edges, the black dotted arrows are the loop-closure edges, and the switchable factors (yellow nodes) are added to the loop-closure edges.

A. Definition

The vertex representing a camera pose is defined as a $SE(3)$ matrix, and the switchable factor added to the loop-closure edges is represented as a float value. As for the constraint between the vertices, the edges is also represented as a $SE(3)$ matrix.

For optimization, we optimize the whole graph every time. Even though global optimization is time-consuming, it is required for submap merging. For decreasing the error of the edges with an unknown scale, we build multiple non-scale edges for a vertex. The graph can be represented as a "spring model" [17].

Even the edge has an unknown scale in terms of the translation, it gives a constraint in the heading of the translation, which can be treated as a direction of the force of springs. If the vertex has more than one edge, it can be considered to be pulled by many springs from different directions, and the pose of the vertex can be obtained when the spring forces are balanced after optimization. Also, the scale of all the edges will be aligned after optimization.

B. Approach

In the front-end, when a frame is lost, which means its edge set is empty, we take this frame as the first keyframe of a new submap and try to maintain tracking in this submap. The first vertex of a new submap is placed at the origin and it is not fixed. This approach makes the system update the graph all the time: maintaining tracking in the current submap or building a new submap. The pipeline of submap-based back-end is illustrated in Fig. 5.

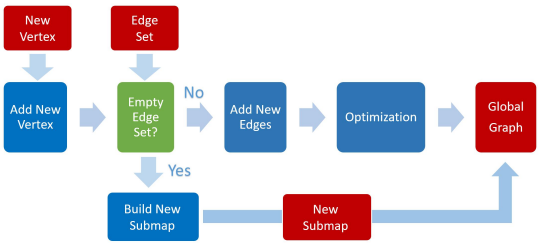

Fig. 5: Back-end pipeline

Within a submap, an estimation is given to the vertex using a motion model. The estimation is calculated from the relative pose and the last keyframe pose tracked in the same submap. We always track the frame in the current submap, and the tracking calculation is a two views geometry. With such a tracking strategy, despite lost frames, our system is able to work without the information of the existing graph.

The weighted information matrix of an edge is set according to the number of inlier correspondences s and the number of detected features n, the information matrix is set by the formula 1,

$$\Xi = \begin{bmatrix} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & \Omega & & \\ & & & & \Omega & \\ & & & & & \Omega \end{bmatrix} \tag{1}$$

where $\Omega = s/n \bullet 100$. Such an information matrix makes the intensity of the translation terms in a "spring model" weak.

For submap merging, we build loop-closure edges between the vertices in the different submap. The loop-closure edges align the submaps after optimization. However, submap merging by place recognition only is fragile. Therefore, the switchable factors are added to the loop-closure edges. With the switchable factors, the loop-closure edges that cannot fit the global consistency well are closed. The vertices that have no edge or whose edges are all closed are discarded.

V. Implementation

For a complete exploration system, beside the methods mentioned above, other implementation details are introduced in this section. Also, to prevent an infinite exploration in a limited space, a termination-condition is proposed to stop the exploration, which depends on the density of vertices.

A. Exploration termination-condition

With respect to the growth of the graph, the number of edges and vertices increases during a dense exploration. However, no marginalization is performed in our back-end, to avoid the risk of breaking submap merging. In addition, a complete environment representation is beneficial to robust visual localization. Hence, a termination-condition is needed to evaluate the completeness of environment mapping and limit the growth of graph.

Considering a dense exploration, the vertices in a limited space become denser and denser. Therefore, the exploration can be stopped if the vertices are dense enough. Since the number of loop-closure edges is proportional to the overlap between the keyframe observations, and the overlap also increase with the increasing of vertex density. In other words, when the number of edges for a new vertex are big enough, and such a situation continues for some time, the vertices can be thought to be dense enough. This termination-condition is illustrated in Fig. 6.

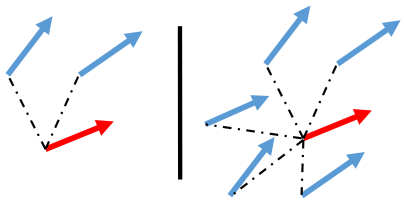

Fig. 6: Illustration of the termination-condition, red arrow is the new vertex, blue arrows are the existing vertices, and the black dotted lines are edges. The left figure: when the existing vertices are sparse, less edges can be established. Right figure: when the existing vertices are dense, more edges can be built.

To implement such a termination-condition, a threshold in terms of the size of new edges is set, represented as T_e. The edges and vertices added to the graph after the last optimization are called new edges and new vertices, the vertices left after discarding are called active vertices. When the size of new edges is greater than T_e, an optimization is performed and the size of new active vertices S_{nv} is recorded. If S_{nv} is smaller than $T_e/(k+1)$ (which means all the potential loop-closures are accepted and the tracking edge is established), a counter c plus one. Only when c is large enough, we stop the exploration and shutdown the system.

B. Optimization implementation

After building a graph, two types of optimization are found in the system: "free-time optimization" and "regular optimization". Since not all the frames are regarded as keyframes, we optimize the graph with a bigger iteration time when the front-end is free, and such an optimization is called "free-time optimization".

Also, to meet the termination-condition mentioned above, an optimization triggered by T_e is performed and called "regular optimization", which has a smaller iteration time. Such an optimization is required since some vertices may become inactive when switchable factors closing during optimization, and the size of active vertices is required by the termination-condition.

VI. Experiments

For evaluating the precision and robustness of our system, two challenging datasets are collected, which contain the image sequence of dense exploration captured by a monocular camera. Also, the GT (Ground Truth) trajectory collected from the Opti-Track motion capture system is recorded for the ATE RMSE [18] calculation. In addition, it is hard to find the existing datasets meeting our requirements, which should be a dense exploration process and the duration should be long enough to meet the termination-condition.

As a popular VSLAM system, ORB-SLAM 2.0 (abbreviated as ORB-SLAM in this section) is selected for comparison, which is also an ORB feature-based VSLAM system.

Also, another experiment is designed for relocalization evaluation. We run a part of the dataset for graph building. Then we run another part of the dataset for relocalization evaluation. The experiment is designed to demonstrate the ability of solving the "kidnapped" problem.

In the end of this section, we analyse the experimental results and discuss the system.

A. Evaluation and Comparison

Two dense exploration datasets are collected. We explore a limited space repeatedly and observe the scene in different perspectives during datasets collection. The durations of the datasets are long enough to meet the termination-condition. Some challenging situations in our collected datasets are shown in Fig. 7, including the dynamic objects, textureless walls and floors, and motion blurred frames.

Fig. 7: Challenging situations in our datasets

Since the performances on those challenging datasets are unstable, we run 6 times on one dataset for comparison. In addition, we find that the maximum number of detected features in ORB-SLAM influences its performance, therefore, two parameters are set and experimented: 1000 and 1500. However, the maximum number of detected features in our system is always 500. For a fair evaluation, the duration for meeting the termination-condition is recorded, and both systems are run for the same duration.

Beside the RMSE, the tracking percentage (TP) of ORB-SLAM is indicated, which equals the tracked frames of ORB-SLAM divided by the considered frames. The precision comparison and the TP are shown in Tab. I and Tab. II, where the second and the third columns are the RMSE of our system and ORB-SLAM, the "Feature Number" is the feature detection parameter in ORB-SLAM.

	Our system	ORB-SLAM	Feature Number	TP
Test 1	0.615	0.884	1500	31.0%
Test 2	0.602	0.668	1500	21.7%
Test 3	0.696	0.922	1500	78.4%
Test 4	0.737	0.921	1500	71.5%
Test 5	0.593	0.412	1000	10.0%
Test 6	0.697	0.558	1000	18.6%

TABLE I: Precision comparison on dataset 1

	Our system	ORB-SLAM	Feature Number	TP
Test 1	0.457	1.000	1500	63.2%
Test 2	0.468	1.012	1500	48.0%
Test 3	0.432	0.925	1500	46.3%
Test 4	0.341	0.728	1500	26.5%
Test 5	0.408	0.676	1000	19.0%
Test 6	0.503	0.863	1000	18.6%

TABLE II: Precision comparison on dataset 2

Since our system updates the graph all the time, the TP in our system is always 100%, it is not indicated in the tables. Instead, the number of submaps, which is proportional to the number of tracking lost, is shown. The scatter plots with respect to the number of submaps and the RMSE are shown in Fig. 8 and Fig. 9. The correlation coefficients are -0.369 and 0.240, which mean the correlation between the size of submaps and RMSE is not significant. In addition, we pick the tests with the same number of submaps to evaluate the stability of our system. The average of RMSE is 0.513, and the standard deviation is 0.046.

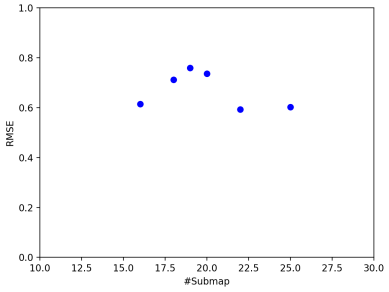

Fig. 8: Number of submap and RMSE on dataset 1

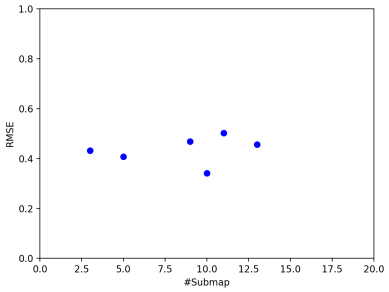

Fig. 9: Number of submap and RMSE on dataset 2

To compare the robustness of ORB-SLAM and our system, beside the TP, we record the number of new keyframes in ORB-SLAM (with 1500 features) before each lost tracking, and calculate the average (A-KFs) and the standard deviation (SD-KFs) of that number. Also, the same metrics are recorded for our system. Also, the average RMSE (A-RMSE) of the tests (6 pairs of tests are run on each dataset) is shown in the Tab. III and Tab. IV. Even though our system updates the graph all the time, because of the keyframe selection conditions, not all the frames are considered.

	A-RMSE	A-KFs	SD-KFs
ORB-SLAM	0.708	13.88	2.37
Our system	0.692	1069.20	23.60

TABLE III: Robustness comparison on dataset 1

	A-RMSE	A-KFs	SD-KFs
ORB-SLAM	0.893	33.54	3.04
Our system	0.421	836.80	20.80

TABLE IV: Robustness comparison on dataset 2

B. Relocalization

We divide our collected dataset into two parts for graph building and relocalization evaluation. 80% of the images are used for graph building with the termination-condition and 20% for "kidnapped" relocalization evaluation. These two processes are run in order.

In the relocalization evaluation, the parameter k for place recognition is set higher than the first experiment for adding more edges, and other parameters are kept the same. All the vertices after optimization are set to be fixed. In addition, no tracking edge is established in this experiment, since we try to solve a "kidnapped" relocalization problem whose observation is an independent image instead of an image sequence.

Two performance metrics are computed: relocalization precision and tracking percentage. To avoid missing frames during the evaluation, we play the relocalization dataset in a low frame rate and remove the keyframe selection conditions, in order to be able to consider more frames.

The precision is also calculated in terms of ATE RMSE, and the tracking percentage equals the tracked frames divided by the considered frames. The experimental results are shown in Fig. 10, whose tracking percentage is 65.8% and the RMSE is 0.312 based on a fixed graph with a 0.308 RMSE.

For ORB-SLAM, since no complete trajectory can be built in our collected dataset, no relocalization evaluation is performed.

C. Discussion

As shown in Tab. I and Tab. II, in most cases, our system has a smaller RMSE in both datasets, which is achieved by the design of our system. Having more edges leads to better precision. ORB-SLAM has a smaller RMSE than our system in Test 5 and Test 6 on the first dataset, which could be due to the low TP (10% and 18.6%) with a small number of samples in RMSE calculation. Since the number of submaps

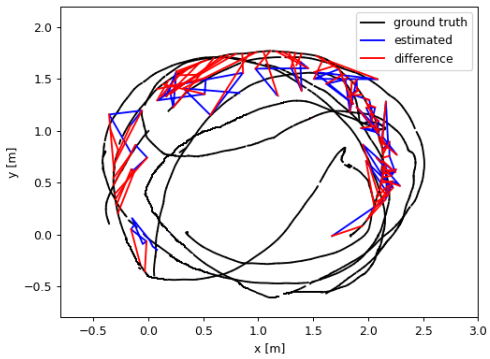

Fig. 10: Result of relocalization. The red lines are the difference between the estimation and the correspondent GT poses. The shorter the line, the smaller the error.

is proportional to the number of tracking failure, as shown in Fig. 8 and Fig. 9, due to the submap-based back-end, our system keeps working even fails to track. As the correlation coefficients and Figs. 8 and 9 shows, the RMSE does not increase with the increasing of the size of submaps, verifying the ability to overcome tracking failure of our system. As is shown in Tab. III and Tab. IV, the A-KFs of ORB-SLAM is much smaller than our system, since our system keeps building submaps and updates the global graph all the time.

For the challenging situations, instead of trying to maintain tracking with those frames, we just stop building the current submap and initialize a new submap. In our opinion, lost tracking is a sequential perspective problem, and the submaps can be tracked in other perspective in another sequence.

Regarding relocalization, our system obtains a 0.312 ATE RMSE, as well as a 65.8% tracking. Thanks to the proposed termination-condition, we are able to create a topologically dense map (dense keyframes and complete images in each keyframe) after exploration. With such a map, even an observation in a different perspective can be relocalized.

As for the time consumption, the average frame rate of our system is 3.0Hz, which is lower than the frame rate of ORB-SLAM. Such a low frame rate is caused by loop-closure detection and verification for each frame, both taking much time. However, thanks to our feature-based front-end, whose baseline is allowed to be wide, tracking is able to continue. Even if a frame cannot be tracked, because of the submap-based back-end, the system can still update the graph.

With the experimental results and the analysis above, we have provided evidence for our contributions: a robust VSLAM system with a submap-based back-end, whose precision is also improved by a multi-constraint front-end. The map whose completeness is guaranteed by the proposed termination-condition can localize the frames in different perspectives.

VII. Conclusion

A VSLAM system is proposed in this paper. It is designed for robust exploration and localization in a limited space. A submap-based back-end is introduced to improve the robustness. Also, a front-end is designed that adds multiple edges to the graph to improve the robustness as well as the precision. Some challenging datasets are collected to evaluate our system, and our system has a smaller ATE RMSE and a more robust performance than the state-of-the-art. Also, the ability to solve "kidnapped" problem is demonstrated.

In the future, we will design a complete autonomous exploration system based on this study. The place recognition module and the relative pose calculation can be improved by other alternative approaches, such as CNN based image matching and deep-feature based tracking, to further improve the performance of the VSLAM system.

References

[1] Bourmaud G, Megret R. Robust large scale monocular visual S-LAM[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1638-1647.

[2] Mur-Artal, Raul, and Juan D. Tardos. ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras[J]. IEEE Transactions on Robotics, 2017, 33(5): 1255-1262.

[3] Engel, Jakob, Thomas Schops, and Daniel Cremers. LSD-SLAM: Large-scale direct monocular SLAM[C]//ECCV 2014. Zurich, Springer International Publishing, 2014: 834-849.

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

[5] Ni K, Dellaert F. Multi-level submap based SLAM using nested dissection[C]// IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2010:2558-2565.

[6] Ni K, Steedly D, Dellaert F. Tectonic SAM: Exact, Out-of-Core, Submap-Based SLAM[C]// IEEE International Conference on Robotics and Automation. IEEE, 2007:1678-1685.

[7] Li H, Chen Q. An appearance-based map partition framework for submap SLAM[C]// Intelligent Control and Automation. IEEE, 2011:1161-1166.

[8] Oberlander J, Roennau A, Dillmann R. Hierarchical SLAM using spectral submap matching with opportunities for long-term operation[C]// International Conference on Advanced Robotics. IEEE, 2013:1-7.

[9] Wang J, Song J, Zhao L, et al. A Submap Joining Based RGB-D SLAM Algorithm Using Planes as Features[C]//Field and Service Robotics. Springer, Cham, 2018: 367-382.

[10] Maier R, Sturm J, Cremers D. Submap-Based Bundle Adjustment for 3D Reconstruction from RGB-D Data[M]// Pattern Recognition. Springer International Publishing, 2014:54-65.

[11] Lowry S, Snderhauf N, Newman P, et al. Visual Place Recognition: A Survey[J]. IEEE Transactions on Robotics, 2016, 32(1):1-19.

[12] Glvez-Lpez D, Tardos J D. Bags of binary words for fast place recognition in image sequences[J]. IEEE Transactions on Robotics, 2012, 28(5): 1188-1197.

[13] Snderhauf N, Protzel P. Switchable constraints for robust pose graph SLAM[C]// Ieee/rsj International Conference on Intelligent Robots and Systems. IEEE, 2012:1879-1884.

[14] Sunderhauf N, Protzel P. Towards a robust back-end for pose graph SLAM[C]// IEEE International Conference on Robotics and Automation. IEEE, 2012:1254-1261

[15] Mur-Artal, Raul, Jose Maria Martinez Montiel, and Juan D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system[J]. IEEE Transactions on Robotics, 2015, 31(5): 1147-1163.

[16] Fischler M A, Bolles R C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography[M]//Readings in computer vision. 1987: 726-740.

[17] Golfarelli M, Maio D, Rizzi S. Elastic correction of deadreckoning errors in map building[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 1998: 905-911.

[18] Sturm J, Engelhard N, Endres F, et al. A benchmark for the evaluation of RGB-D SLAM systems[C]//Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on. IEEE, 2012: 573-580.