2021 6th IEEE International Conference on
Advanced Robotics and Mechatronics
July 3-5, 2021, Chongqing, China

2021 6th IEEE International Conference on Advanced Robotics and Mechatronics (ICARM) | 978-1-6654-3909-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICARM52023.2021.9536181

Robust Mapping and Localization in Offline 3D Point Cloud Maps

Guo He, Fei Zhang, Xiang Li, Weiwei Shang

***Abstract*—Aiming at the degradation of lidar, we propose a Robust Mapping and Localization (RMAL) method, which combines the classic Extended Kalman Filter (EKF) algorithm with the back-end pose graph optimization for 3D real-time mapping. Utilizing the complementary advantages of multiple sensors, the robustness of the mapping method is enhanced. In addition, we choose to save the feature keyframes and the corresponding optimal pose transformations as the offline map during the mapping process. Cooperating with subsequent mapping again, we can improve the positioning accuracy of the robot in the offline map. Finally, we also conduct experimental tests in different real scenarios, and the results verify the robustness and engineering practicability of the proposed method.**

I. INTRODUCTION

In the field of intelligent mobile robots, mapping and localization are the basic premises of autonomous navigation. Simultaneous Localization and Mapping (SLAM) based on vision and lidar methods is also a popular research content today. However, compared with camera sensors used in the vision-based methods, lidar sensors are not sensitive to seasonal weather and illumination changes. Moreover, many high-resolution 3D lidars can perceive the details of the environment at a distance with a frequency of about 10HZ and a 360-degree horizontal field of view. Therefore, 3D lidars have become one of the main sensors in the field of SLAM research.

The technology of using lidar for state estimation and mapping has become more mature. The Iterative Closest Point (ICP) algorithm proposed by Besl *et al*. [1] is a very classic scan matching method at first. The relative pose transformation relationship between the two frames is obtained by iteratively minimizing the sum of squares of Euclidean distances between all corresponding points in the two frames. However, the 3D lidar scan frames contain a large number of points, which bring a huge computational cost to the ICP. In order to meet the real-time requirement, scholars have proposed many methods of feature extraction and matching. Grant *et al*. [2] proposed a robust plane search algorithm in large sparse point clouds and used a decoupling optimization method to calculate the translation and rotation transformations between the corresponding planes of two frames. Zhang *et al*. [3] proposed a real-time Lidar Odometry and Mapping (LOAM) method, which analyzes the local characteristics of the point clouds by defining the roughness of the point. In this way, plane feature points and edge feature points are extracted, and real-time state estimation is performed through two independent and different frequency matching algorithms. After that, Shan *et al*. [4] segmented the point clouds based on LOAM to eliminate unreliable feature points and introduced a two-step Levenberg-Marquardt (L-M) optimization method, which can realize real-time state estimation and mapping on an embedded system. Renaud *et al*. [5][6] used a data-driven method to obtain the feature descriptors of segments extracted by Euclidean clustering, and finally proposed the Segment-based Mapping and Localization (SegMap) method. Inspired by LOAM and SegMap, Rozenberszki *et al*. [7] innovatively combined these two advanced methods, only using lidar to achieve global positioning in the existing offline map, which can eliminate the cumulative errors of the lidar odometry, but this method needs to assume the starting position before the first location, which does not meet the requirements of actual applications.

However, only using lidar sensors to support the entire mapping and localization system is unreliable. With the movement of the robot equipped with sensors, the lidar data will have movement distortion [8], which directly affects the point clouds matching accuracy. Lidar degradation also exists in some practical scenarios. For example, in open areas such as squares and airports, even 3D lidar can only receive ground point clouds for matching, which may cause the state estimation to move randomly on the horizontal plane; or in environments such as single-sided walls and corridors, the point clouds obtained by moving along the direction of the building is the same, which makes the matching algorithm unable to correctly estimate the movement in this direction. Furthermore, the point clouds have sparse characteristics [9], but their calculation and processing are expensive, so the frequency of state estimation based on lidar is often low. To solve these problems, some scholars proposed to use high-frequency IMU data to make up for the lack of lidar. Gentil *et al*. [10] realized inter-frame feature matching even in the presence of lidar motion distortion by fusing IMU and lidar data. Tang *et al*. [11] established a stable navigation framework using EKF combined with Inertial Navigation System (INS) and lidar SLAM in the absence of a Global Navigation Satellite System (GNSS). Ye *et al*. [12] proposed a tightly coupled 3D Lidar Inertial Odometry and Mapping framework (LIOM), which can obtain robust state estimation results even in some degraded scenarios. Qin *et al*. [13] designed a robot-centric iterated error-state Kalman filter, which tightly couples a six-axis IMU and a 3D lidar to provide a real-time and robust ego-motion estimation. However, the above IMU measurements and lidar odometers

Our work is supported by the NSFC of Grant 51675501.

Guo He, Fei Zhang, Xiang Li, Weiwei Shang are with the Department of Automation, University of Science and Technology of China, Jinzhai Road 96, Hefei P. R. China (email: guotian@mail.ustc.edu.cn, zfei@ustc.edu.cn, li12@mail.ustc.edu.cn, wwshang@ustc.edu.cn)

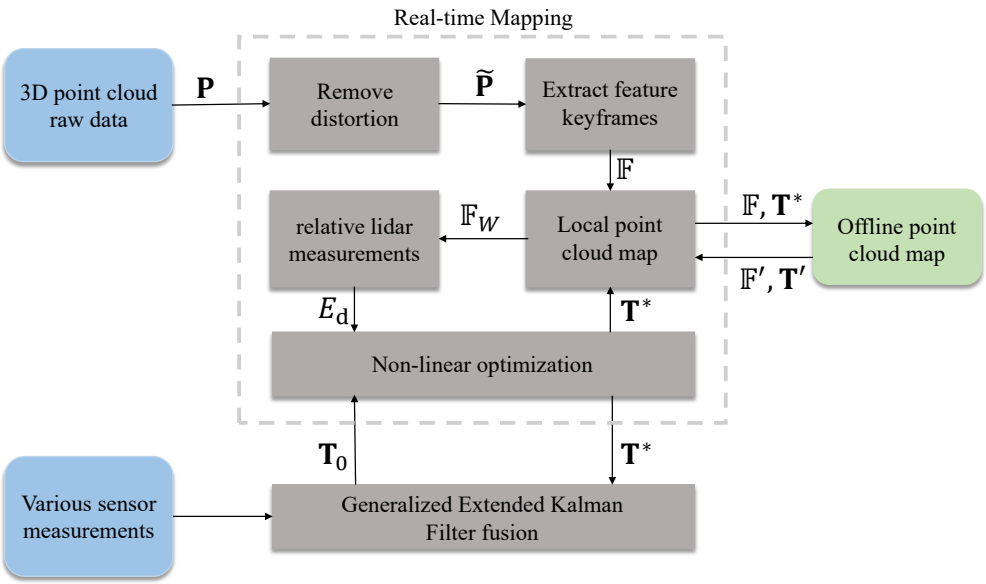


Fig. 1. Structural framework of robust mapping and localization method. The generalized extended Kalman filter fuses each sensor data to provide the initial pose $\mathbf{T}_0$ for the real-time mapping module, and at the same time fuse the optimized pose $\mathbf{T}^*$. In the first step of the mapping process, the key feature frames $\mathbb{F}$ extracted in real-time mapping and their corresponding optimal poses $\mathbf{T}^*$ are saved as an offline point cloud map. In the subsequent use of the map, an offline point cloud map ($\mathbb{F}'$, $\mathbf{T}'$) will be loaded to optimize navigation and positioning.

are all relative measurement information. Without absolute measurement, the robot will drift during long-term motion. Shan *et al*. [14] combined relative and absolute measurements on factor graphs to establish a framework for tightly-coupled lidar inertial odometry through smoothing and mapping, which is suitable for multi-sensor fusion, but the system will face crash and fail to operate when only the IMU is too jittery or even fails.

In this paper, we propose a Robust Mapping and Localization (RMAL) method to make up for the shortcomings of using lidar alone. We use the Extended Kalman Filter (EKF) framework proposed in [15] to assist lidar odometry and mapping, which can support a wide range of multi-sensor inputs. It does not affect the operation of the entire system when one or more sensors fail. In addition, we merge the established offline map and the subsequent online mapping into a pose graph for overall optimization, which improves the positioning accuracy in the offline map and can expand the offline map. Before the first localization, we combine the absolute measurement information (Global Navigation Satellite System, GNSS) to calculate the current initial position of the robot in the offline map. Finally, we carry out experimental tests on the system method in different scenarios, and the results show that we achieve comprehensive preparations before autonomous navigation.

The rest of this paper is organized as follows. Section II introduces our proposed method in detail. In Section III, we analyze the different experimental results and the conclusion is presented in Section IV.

II. Proposed System

In this section, we introduce the Robust Mapping and Localization (RMAL) method in detail. As shown in Fig. 1, we use the generalized EKF [15] to enhance the robustness of real-time mapping. The filtering fusion result corrects the motion distortion of the original 3D point clouds and optimizes the initial value of scan matching to prevent falling into the local optimality and reduce the iteration time. The lidar odometry obtained by real-time mapping further modify the result of the generalized EKF. The established 3D point cloud map can be saved as an offline map, and then match the current online map with the offline map based on the European distance, allowing them to carry out back-end optimization under a pose graph. We continue to build the map on the offline map for improvement or expansion, and more accurate pose estimation can be obtained in the offline map. Below, we separately introduce the three main system function modules of generalized extended Kalman filter, real-time mapping, and use of the offline map.

A. Generalized Extended Kalman Filter

In practical applications, mobile robots usually start from an arbitrary starting position in the scene, rather than the default original position or a fixed initial position. Therefore, before positioning for the first time, it is necessary to determine the initial position in the pre-built map coordinate system. We determine the initial position of the robot by recording absolute measurements of the origin of the offline map, such as GNSS position and true heading information. Later, the first scan frame of the lidar can be used to match the offline local map to further reduce the error of the initial position. For details, please see Section II-C.

For the fusion algorithm, we are concerned about the 6-DOF pose and speed of the mobile robot over a period of time, so the robot state $\mathbf{X}$ can be written as

$$\mathbf{X} = \left[\boldsymbol{p}^T, \boldsymbol{q}^T, \boldsymbol{v}^T \right]^T, \tag{1}$$

where $\boldsymbol{p}$ is the three-dimensional position vector of the robot pose. $\boldsymbol{q}$ is a unit quaternion, which can generate a rotation matrix $\mathbf{R} \in SO(3)$. $\boldsymbol{v}$ is the velocity vector. The pose estimation transformation $\mathbf{T} = \langle \mathbf{R} | \boldsymbol{p} \rangle \in SE(3)$ can transform the current robot coordinate system to the map coordinate system. During the real-time mapping process, this transformation can initialize the lidar scan matching to ensure the robustness of mapping. Besides, the result of fusion localization replaces a single sensor such as IMU or wheel speed encoder for initialization, which expands the selection range of sensors and the stability of the entire system. It is worth noting that only the filtering fusion state is given here, the detailed formula derivation and implementation details of the algorithm can be found in the paper [15] [16].

B. Real-time Mapping

As shown in Fig. 1, when the original 3D point cloud data $\mathbf{P}$ is received, the distortion needs to be eliminated first to reduce the influence of motion distortion. In order to obtain the point cloud data $\tilde{\mathbf{P}}$ after removing the distortion, we use pose transformation to convert all points $p_t \in \mathbf{P}$ in a scan period $[t_i, t_{i+1}]$ to the initial time of the scan:

$$\tilde{p}_t = (\boldsymbol{T}_t)^{-1} \boldsymbol{T}_{t_i} p_t, \tag{2}$$

where $\tilde{p}_t \in \tilde{\mathbf{P}}$ is the point after removing the distortion, and $t \in [t_i, t_{i+1}]$ is the timestamp of the point in the scan, $\boldsymbol{T}_{t_i}$ and $\boldsymbol{T}_t$ are the pose transformations at corresponding moments respectively. The pose transformation is obtained by the fusion state estimation in Section II-A. Let $\boldsymbol{T}_j$ and $\boldsymbol{T}_{j+1}$ be the two consecutive fusion state estimation results. When the judgment of $t \in \lfloor t_j, t_{j+1} \rfloor$, the pose transformation $\boldsymbol{T}_t$ can be obtained using linear interpolation:

$$\boldsymbol{T}_t = \frac{t - t_j}{t_{j+1} - t_j} \boldsymbol{T}_j + \frac{t_{j+1} - t}{t_{j+1} - t_j} \boldsymbol{T}_{j+1}. \tag{3}$$

For the point cloud data $\tilde{\mathbf{P}}$ after removing the distortion, the feature keyframes are extracted to avoid the computational time of processing the huge point clouds. The selection of feature points and keyframes is similar to the methods used in [3] and [14]. The feature keyframe $\mathbb{F}$ composed of plane feature points and edge feature points is extracted by the thresholds of translation distance and rotation angle. The $\mathbb{F}$ is matched with the corresponding lidar measurements in the previously established local map. That is to say, the edge lines corresponding to the edge feature points and the planes corresponding to the plane feature points can be found in the local point cloud map, and the objective function can be established for scan matching as:

$$E_d = \frac{1}{K} \sum_{k=1}^{K} d_k, \tag{4}$$

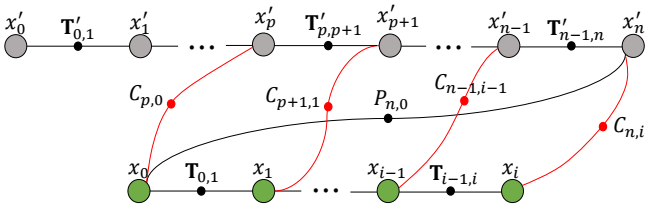


Fig. 2. Optimized structure of pose graph combined with offline map. The gray indicates the pose graph nodes x' and its space constraints $\mathbf{T}'$ generated by loading the offline map, and the green indicates the back-end pose graph (x , $\mathbf{T}$) generated by continuing real-time positioning and mapping. The loop pose constraints C are used to associate them for joint optimization.

where d_k is the distance from each feature point in $\mathbb{F}$ to the nearest corresponding lidar measurement, and K represents the total number of matched corresponding measurements. The optimal pose estimation $\mathbf{T}^*$ of $\mathbb{F}$ can be obtained by solving the following nonlinear optimization problem:

$$argmin_{\mathbf{T}} \left(\frac{1}{K} \sum_{k=1}^{K} d_k \right). \tag{5}$$

We use $\mathbf{T}^*$ to convert the current feature keyframe $\mathbb{F}$ to $\mathbb{F}_W = \mathbf{T}^* \cdot \mathbb{F}$ in the coordinate system of the world map and update the local map in the form of the sliding window. Finally, instead of directly storing the point cloud data in the world coordinate system as an offline map, we store each feature keyframe $\mathbb{F}$ and its corresponding optimal pose estimation $\mathbf{T}^*$ for direct use in the future.

C. Use of Offline Map

We have introduced the generalized EKF and the real-time mapping process in detail, which can improve each other. Next, we focus on using a pre-built offline map through the back-end optimization method of the pose graph, whose structure is shown in Fig. 2. Each feature keyframe is treated as a node. Constraints are established by using the stored and calculated pose estimation, and a loop-like mechanism is adopted to optimize the current pose so that the localization of the robot in the offline map has a good correspondence with the real position in the scene, which can be better applied to the navigation planning.

As shown by the gray nodes and their connection points in Fig. 2, we first use each feature keyframe stored offline and the corresponding pose estimation to establish the pose graph of the offline map. Each feature keyframe stored is regarded as a state node x', and the spatial constraints between nodes are established by using the corresponding pose estimation. The pose space constraint from node x'_{i-1} to x'_i is defined as:

$$\mathbf{T}'_{i-1,i} = (\mathbf{T}'_{i-1})^{-1} \mathbf{T}'_i, \tag{6}$$

where $\mathbf{T}'_{i-1}$ and $\mathbf{T}'_i$ are the pose transform estimation of the $i-1$ and i keyframes respectively, and i is the feature keyframe index in the offline map. In the subsequent construction of the map, we continue to add state nodes and space constraints to this pose graph in the same way. As

shown by the green nodes and their connection points in Fig. 2, when a feature keyframe is extracted, a state node x_i to the pose graph will be added and two different pose space constraints between the nodes will be established:

- Odometry pose constraints. In the process of continuing to build the map, the odometry pose constraint $\mathbf{T}_{i-1,i}$ between adjacent state nodes is established by the optimal pose estimation of feature keyframes. Different from $\mathbf{T}_{i-1,i}$, the space constraint $P_{n,0}$ from the last state node x_n' of the pose graph of the offline map to the first state node x_0 of the current online map is determined by the initial position.
- Loop pose constraints. We use the KNN search algorithm to search for a node closest to the current state node in the partial pose graph representing the offline map. When a certain Euclidian distance threshold is satisfied, the local point cloud map near the nearest node is established and matched with the current feature keyframe to obtain the relative pose transformation such as $C_{n,i}$, which is added to the pose graph.

The first odometry pose constraints are added to make the offline map and the current online map merge into a pose graph for overall back-end optimization. With the second pose constraints, the current lidar feature keyframe can be matched with the offline map. It can optimize and calibrate the global localization of the current robot in the offline map based on the actual scene.

III. Experiments

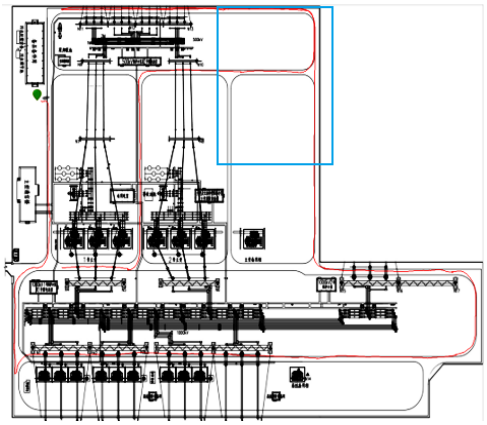
In all test experiments, the algorithm of our system is run in the Robot Operating System (ROS). As shown in Fig. 3, our experimental test platform is a tracked fire-fighting mobile robot equipped with an automatic high-pressure water cannon. In case of fire, it can be combined with sensors such as wheel speed encoders, a Velodyne VLP-16 lidar, and a DH20-M1 integrated navigation system to automatically identify the flame and extinguish the fire after navigating to the location of the fire. We used different methods to carry out experimental comparisons in different scenarios, and qualitatively and quantitatively analyzed the experimental results to illustrate the performance of our proposed method.

A. Mapping Test in Few Feature Environment

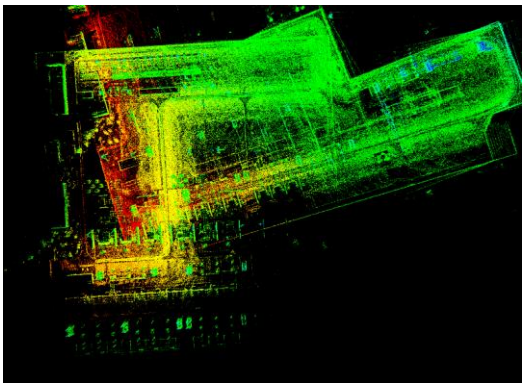
We combine the fusion algorithm with the mapping to achieve multiple sensors fusion, which can make up for the shortcomings of using lidar sensors alone. As shown in the blue box area in Fig. 4(a), there is only one wall in the environment. Using lidar only for scan matching can cause incorrect state estimation in one or more degrees of freedom. We show the real trajectory generated by GPS information on the 2D CAD map, as shown by the red line in Fig. 4(a).

We compare the latest lidar odometry algorithm LOAM [3] and LeGO-LOAM [4]. As shown in Fig. 4, when the robot

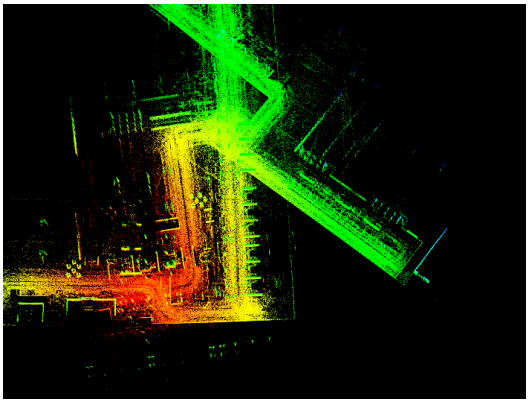
Fig. 3. Experimental test platform



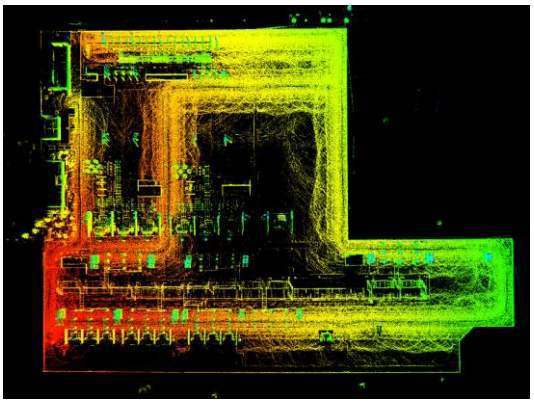
(a)



(b)



(c)



(d)

Fig. 4. Mapping in a few feature environment: (a) CAD map; (b) LOAM; (c) LeGO-LOAM; (d) RMAL

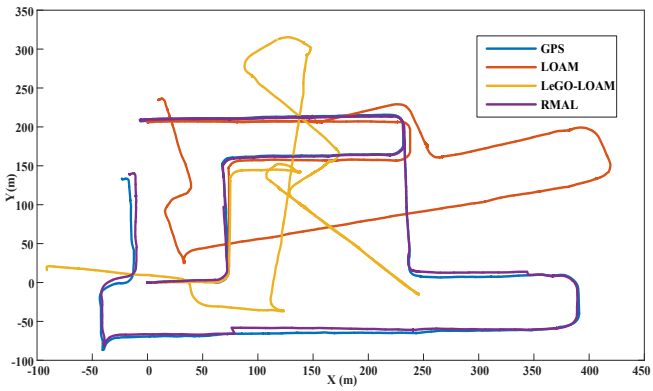


Fig. 5. Trajectory during mapping

travels to the blue frame area, the lidar can only receive point clouds on the ground or one side wall. The point clouds obtained during the robot movement are the same. At this time, the measurement results of lidar will be degraded, which makes the matching algorithm of LOAM or LeGO-LOAM unable to correctly estimate the motion and eventually lead to the failure of the mapping. On the contrary, our proposed RMAL method can solve this problem and the mapping effect is good. We calculate that the average runtime of mapping for processing one scan is about 48.9*ms*, which also meets the real-time requirement. Fig. 5 is a comparison chart of the mapping trajectory generated by GPS information, LOAM, LeGO-LOAM, and the method proposed in this paper. Taking the trajectory generated by GPS information as a reference, it can be seen that only using lidar sensors cannot build the map correctly in some challenging environments with few features, while the use of multi-sensor fusion can solve this problem well.

B. Localization Test in Offline Map

In the applications of mobile robots or automatic navigation, path planning is necessary in a preset offline map, and the navigation process also requires the precise positioning of the robot in the offline map. Before the localization test experiment, we use the RMAL method to build and save the offline map as shown in Fig. 6. Then, we conduct a localization test on the offline map area.

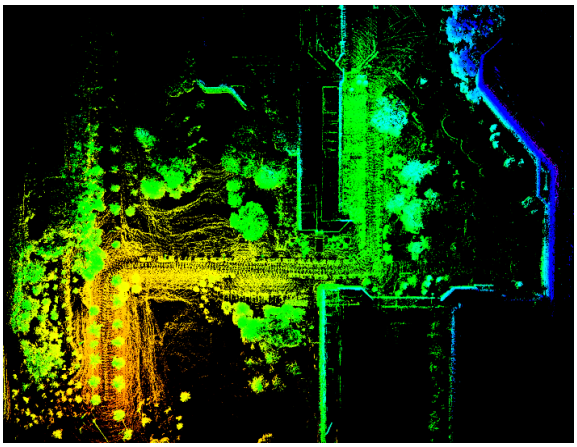
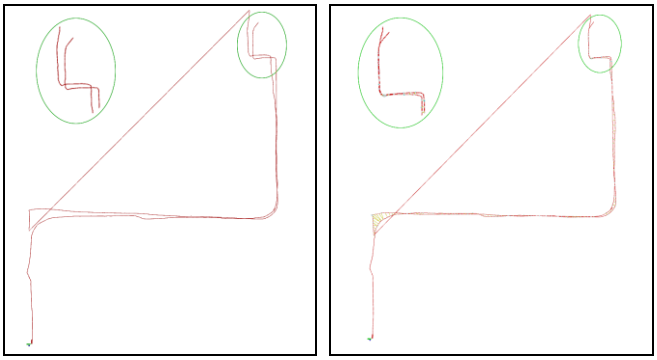


Fig. 6. Offline map used in localization test



(a) (b)

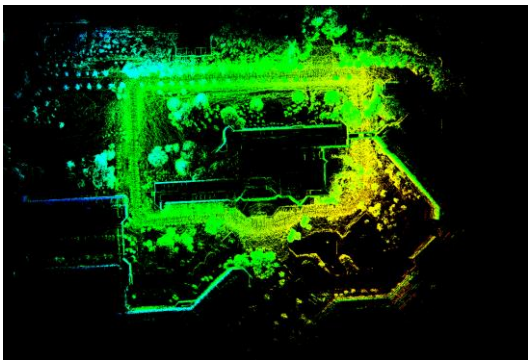
Fig. 7. Trajectory formed by localization results in the offline map: (a) no offline map correction; (b) offline map correction

Combining the offline map matching correct the current robot's positioning results in the offline map, and its overall trajectory is shown in Fig. 7(b). As shown in Fig. 7(a), if there is no matching correction of the offline map, the location in the map will accumulate deviation. The enlarged area inside the green ellipse is a narrow road. In Fig. 7(b), it can be clearly seen that the overlap between the trajectories of this area and the offline map performs well, which is more consistent with the actual scene during the test.

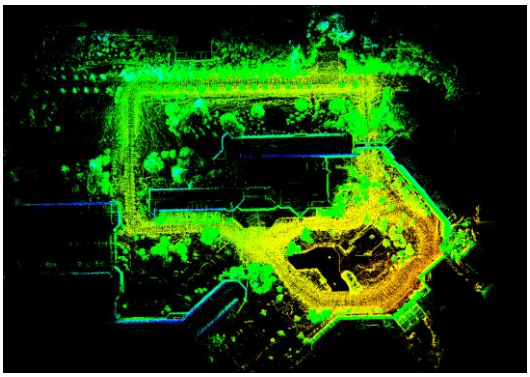
We quantitatively analyze the test results of the final positioning of the robot. Table I shows the final positioning results in the offline map with GNSS information, no map correction method, and offline map correction method. GNSS information can be used as a reference to calculate the location errors of the two methods with and without map correction, and the Root Mean Square Error (RMSE) results take into account the errors along the x, y axis and the heading. The results show that the positioning accuracy of the robot in the offline map is improved through the matching correction of the offline map.

C. Extension Test of Offline Map

Using the proposed method, we can continue to optimize the mapping in the original offline map, thereby achieving the expansion and improvement of the offline map, which provides a good idea for large-scale block mapping and map



(a)



(b)

Fig. 8. Extension of offline maps: (a) before expansion; (b) after expansion

TABLE I. THE FINAL LOCALIZATION RESULTS IN THE OFFLINE MAP

Methods	X (m)	Y (m)	True Heading (°)	RMSE
GNSS	136.12	-88.86	-17.68	0
No correction	135.58	-90.80	-17.49	2.023
Correction	135.96	-89.22	-17.69	0.394

splicing. As shown in Fig. 8, we expand the established offline map. Figure (a) is the existing offline map, and figure (b) is the result of continuing to build the map. As shown in the figure, we have expanded the lower right corner of the original offline map.

IV. CONCLUSION

Combining the EKF and the real-time 3D mapping, we proposed a robust mapping and localization method, which can perform 3D point cloud reconstruction well in the environment with few features. Furthermore, we save the feature keyframes and the corresponding pose transformations as an offline map, which can be matched and jointly optimized with the online map to correct the global position of the robot in the offline map. Using the same method, the offline map can also be expanded and improved. The analysis of the experimental results shows the reliability and practicability of our method.

REFERENCES

[1] P.J. Besl and N.D. McKay, "A Method for Registration of 3D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239-256, Feb. 1992.

[2] W.S. Grant, R.C. Voorhies, and L. Itti, "Finding Planes in LiDAR Point Clouds for Real-time Registration," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4347-4354, Nov. 2013.

[3] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proceedings of the Robotics: Science and Systems*, vol. 2, pp. 9, Jul. 2014.

[4] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-optimized Lidar Odometry and Mapping on Variable Terrain," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4758-4765, Oct. 2018.

[5] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 5266-5272, May. 2017.

[6] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart and C. Cadena, "SegMap: Segment-based mapping and localization using data-driven descriptors," *The International Journal of Robotics Research*, vol. 39, no. 203, pp. 339-355, 2020.

[7] D. Rozenberszki and A. Majdik, "LOL: Lidar-Only Odometry and Localization in 3D Point Cloud Maps," *arxiv*, vol. abs/2007.01595, Jul. 2020.

[8] S. Anderson and T. D. Barfoot, "Ransac for motion-distorted 3d visual sensors," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3-7, Nov. 2013.

[9] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Proceedings of the Robotics: Science and Systems*, 2018.

[10] C. L. Gentil, T. Vidal-Calleja and S. Huang, "IN2LAMA: INertial Lidar Localisation And MApping," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 6388-6394, May. 2019.

[11] J. Tang, Y. Chen, X. Niu, L. Wang, L. Chen, J. Liu, C. Shi, and J. Hyyppa, "Lidar scan matching aided inertial navigation system in gnss-denied environments," *Sensors*, vol. 15, no. 7, pp. 16710–16728, 2015.

[12] H. Ye, Y. Chen and M. Liu, "Tightly Coupled 3D Lidar Inertial Odometry and Mapping," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3144-3150, May. 2019.

[13] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang and M. Liu, "LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 8899-8906, 2020.

[14] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping," *arxiv*, vol. abs/2007.00258, Jul. 2020.

[15] T. Moore and D. Stouch, "A Generalized Extended Kalman Filter Implementation for The Robot Operating System," *Intelligent Autonomous Systems*, vol. 13, pp. 335-348, 2016.

[16] R. E. Kalman, "A new approach to linear filtering and prediction problems", *Transactions of the ASME*, vol. 82, no. 1, pp. 35-45, 1960.

[17] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A.Y. Ng, "ROS: An Open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, pp. 5, 2009.