Proceedings of The 2019 IEEE International
Conference on Real-time Computing and Robotics
August 4-9, 2019, Irkutsk, Russia

# Low-cost and High-accuracy LIDAR SLAM for Large Outdoor Scenarios

Chenglin Pang[1], Yulin Tan[1],Shangcong Li[1],Yanli Li[1], Bing Ji[1],Rui Song[1*]

***Abstract*— Be applied in the large outdoor scenarios, we keep the high-accuracy of robot's mapping and location. Relied on lightweight hardware, we have improved an algorithm of real-time LIDAR SLAM (simultaneous localization and mapping). Compared to vision sensors, LIDARs have higher accuracy and larger range for outdoor scenarios. In the situation of robot moving quickly, LIDARs can keep the features no missing in the scan range, meanwhile camera maybe happen feature fuzzy or missing. Velodyne VLP-16 LIDAR can provide 3D point cloud with 16 channels' scan. By segmenting for LIDAR point clouds, and extracting more special edge or corner feature, we compute robot' motion model that relied on the methods like Point-to-line Iterative Closest Points (PL-ICP) or Point-to-Plane Iterative Closest Points (PP-ICP). Based on robots' motion model, matching point cloud features in a certain range to fix the rigid body motion model. In every certain distant, optimized robot's state information and map based on the method of graph optimization. We use this method to achieve lightweight and high-accuracy LIDAR SLAM for large outdoor, and computed the state of robot. By making experiments in our school, we tested our method for building high accuracy map and providing robot's state.**

## I. Introduction

In mobile navigation of intelligent robot, map building and state estimation are the most basic condition. The state estimation of robots could describe by their pose (position and orientation), although other variables could be used (speed, etc.) too. Meanwhile, mapping is described by aspects of interest in robots' operating environment (the position and range of landmark). However, robots running in outdoor large scenarios, facing the more complex environment, need better robustness for various situations. The more complex environment, the more problems that feature moving is too large, or the feature is not obvious. Compared to vision sensor, like camera, traditional single channel LIDAR have advantages of stable and high-accuracy. Multichannel LIDAR have less used in the project of SLAM, because of high cost. But multichannel LIDAR have the ability to provide numerous point clouds, which can be used in building dense map. The reason of maturity of technology and the number of manufacturers increase lead to the price of multichannel LIDAR is lower than 20 years ago. Therefore, for vehicle outdoor navigation, 3D multichannel LIDAR has become the best choice.

Giorgio Grisetti et al. based on particle filter algorithm achieved simultaneous mapping and location in a small scale (Gmapping) [1]. By bringing Rao-Blackwelized particle filters (RBPF), they solved the question of SLAM indoors. They reduced the number of particles, relied on multiple particles tracking landmark features separately to estimate the motion model of the robot. During the increase of the scale, Gmapping consumes a lot of memory space, and unable to correct map consistency. For many applications in urban and rescue scenarios, Stefan Kohlbrecher et al. present a system for fast online learning of occupanying grid maps requiring low computational resources [2]. Combined IMU (Inertial Measurement Unit), they used a robust scan matching approach to achieve 3D motion estimation in the small scale where large loops do not have to be closed. And this approach is beneficial for indoor scenarios. They designed a flexible and scalable SLAM system if we want to applied to a variety of environment, which can aid several sensors like GPS, compass etc.

During Sameer Agarwal et al. developed the open source C++ software library which named Ceres Solver[3] to solve the nonlinear least squares problem and Rainer Kuemmerle et al. s described the least squares optimization of an error function as a graph to solve the problem[4]. Rainer Kuemmerle et al. developed the open source C++ library—G2O, which become a general framework for Graph SLAM. SLAM has grown up into a front-end and back-end framework (Figure 1). Front-end achieved the rough calculation of motion module by some scan matching methods like ICP (Iterative Closest Points) or NDT (Normal Distribution Transform) [5]. Back-end achieved the optimization that reduce the error which produced in the process of scan matching, with increasing the quality of map. The general methods are nonlinear least squares optimization, like Gauss-Newton method (G-N) [6] or Levenberg-Marquardt method (L-M) [7].

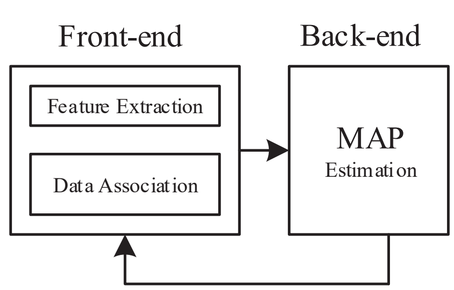


Figure 1. The general framework of SLAM

In the real-time performance, Wolfgang Hess et al. based on scan to submap as Front-end, Ceres Solver to solve

* This paper is supported by National Key Research and Development Program, China (No. SQ2017YB130411) Major Program of Shandong Province Natural Science Foundation, China (No.ZR2018ZC0437) and Key Research and Development Program of Shandong Province, China (No.2016ZDJS02A07)

1, School of Control Science and Engineering, Shandong University, Jinan, 250061, China
Rui Song*, School of Control Science and Engineering, Shandong University, Jinan, 250061, China (e-mail: rsong@sdu.edu.cn)

978-1-7281-3726-1/19/$31.00 © 2018 IEEE

nonlinear least squares problem as Back-end, and based the method of Branch and Bound to improve the speed of calculation with the accuracy keeping at 5cm [8]. Their algorithm reduced the dependence on computing resources, meanwhile they ensured global consistency and achieved real-time large scenarios 2D SLAM, made lightweight equipment a reality. Compared to single channel LIDAR, multichannel LIDAR provide a lot of information of the point clouds. With the point clouds have more detailed details, multichannel provide more information reflecting more real environment's change. It means that we need the method with the lower computation to keep real time.

Iterative Closest Point (ICP) is a typical method for finding pose transformation between adjacent frames of multichannel LIDAR [9]. The corresponding relationship is found in use of point by point in two laser point clouds until the requirements are met. However, the number of point clouds in 3D LIDAR is very large and cost too much time in use of ICP. But as a reliable and effective method in 2D SLAM, there are many ways to improve the performances of ICP. Y. Chen et al. proposed a variant of point-to-line ICP [10], and Censi Andrea et al. proposed a variant of point-to-plane ICP [11].

Though, it is difficult to extract features from a large number of points clouds. If point clouds can be divided and classified in advance, the amount of computation will be reduced. Tixiao Shan et al. proposed it could be fixed by preprocessing original point clouds from multichannel LIDAR [12]. About the method of preprocessing point clouds, M. Himmelsbach et al. proposed a method for fast large-scale long-range segment the point clouds robot running environment into ground point clouds and non-ground point clouds, meanwhile, segment non-ground cloud into several different small point clouds based on Euclidean distance [13]. It is effective to divide a complex point clouds into several small ones to increasing operating speed. I. Bogoslavskyi et al. proposed a method that 3D point cloud could be mapped to 2D laser range images with clearly defined neighborhood relations make the segmentation problem easier [14]. Instead of generation of the 3D point cloud, the approach has the faster compute speed. By calculating the Euclidean distance between neighboring points, they judge whether the points belong to the same object or not. And this approach is effective for sparser point clouds, though the points of 16-beam LIDAR with the longer Euclidean distance between two points. After segmentation, the components have been found which belong to one object. The application of these methods enables us to further improve the speed of feature extraction and remove noisy point clouds.

The approaches of feature-based matching are attracting more and more attentions, because of depending on the less computational resources by extracting features from point clouds to predictive motion model of robot. Bastian Steder et al. proposed a method to place recognition by a variant of the Laplacian of Gaussian approach to calculate interest points with high curvature [15]. Ji Zhang et al. perform the method of computing the roughness of a point in its local region [16]. They perform an approach to calculate robot's state by edge/plane feature correspondences between two adjacent scans and get the range of obstacle from robot. With increasing of the map size and the distance of robot walking, accumulative error is risen because of the error created in scan matching. Their method without loop closure means that accumulative errors cannot be corrected. The error lead to the inconsistencies and the distortion of map (Figure 2). Moreover, too large usage scenarios lead to the increasing demand for computing resources. Nowadays, space resources on mobile robots are becoming more and more precious, emphasizing carrying smaller hardware resources to reduce the space of robots themselves.

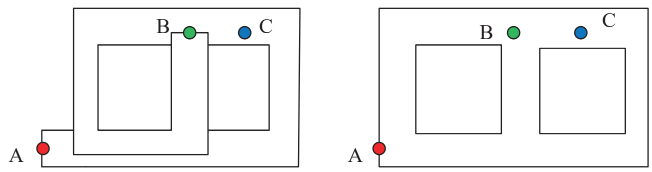


Figure 2. : Left: map built from odometry. The map is homotopic to a long corridor that goes from the starting position A to the final position B. Points that are close in reality (e.g., B and C) may be arbitrarily far in the odometric map. Right: map build from SLAM. By leveraging loop closures, SLAM estimates he actual topology of the environment, and "discovers" shortcuts in the map

Therefore, our work is to improve a high-precision lightweight laser SLAM method for outdoor large-scale scenes, which is convenient to carry on embedded system to realize real-time pose calculation and map construction of the robot, and optimize the pose error by searching loop closure. In order to verify the effectiveness of the algorithm, we have carried out several experiments to test it. The rest of this paper is arranged as follows. Section II describes the overall architecture of the system. Then, section III introduces the contents of the system separately. In section IV, we validate our system in school environment to prove its accuracy and robustness. The section V summarizes the paper.

## II. SYSTEM DESCRIPTION

### A. *Hardware System*

The paper presented scheme is tested by the LIDAR of Velodyne VLP-16, unmanned ground vehicle and controller. Velodyne VLP-16 can provide 3D laser point clouds. The effective measuring distance of VLP-16 is 100 meters, and the precision is ±3cm. The vertical field of view (FOV) is 30° (±15°) and the horizontal FOV is 360°. Velodyne VLP-16 has 16 channel sensor with vertical angular resolution of 2°.

The UGV we used to be with the maximum operating speed of 6 km/h. The JROBOT Komodo-02's design load is 80kg, and its suspension adopts the balanced suspension of Christie four-wheeled group, which can be adapted to heavy load shock absorber.

The controller used in this paper is Nvidia Jetson TX2. The Jetson TX2 is an embedded system-on-module (SoM) with dual-core NVIDIA Denver2+quad-core ARM Cortex-A57, 8GB 128-bit LPDDR4 and integrated 256-core Pascal GPU. Useful for deploying computer vision and deep learning, Jetson TX2 runs Linux and provides greater than 1TFLOPS of FP16 compute performance in less than 7.5 watts of power.

In order to verify the correctness of the trajectory of the robot, we equipped NovAtel's integrated GNSS+INS navigation system SPAN-CPT. The built-in GNSS board adopts NovAtel's latest OEM6 hardware platform technology.

The IMU consists of three-axis fiber optic gyroscope (FOG) and three-axis micro-electromechanical system (MEMS) accelerometer, which can adapt to a variety of environments. SPAN-CPT uses NovAtel's leading satellite navigation technology to achieve centimeter -level positioning accuracy.

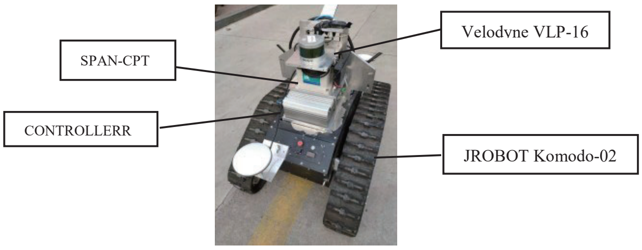


Figure 3. Whole Hardware System

## B. *Software system*

The overall SLAM software system framework we designed is shown in Figure 4. According to the framework of SLAM in general, we divide the system into three parts: Front-end, Back-end and loop closure. Front-end include segmentation, feature extraction and motion model. Back-end include local matching, feature mapping and transform integration. Loop Closure include coarse scan matching and fine scan matching.

Firstly, we need to do pre-process for the point clouds of LIDAR. Segmentation for point clouds from a ball of clutter point clouds. The laser point clouds need to be preprocessed. By removing noises and 3D point clouds need to be mapped to 2D range imagine which can be improved the speed of calculating. After this process, we divide the imagine into two parts. The one part is the imagine which labeled as ground plane, and other part is the imagine which labeled as non-ground part. The plane feature we extracted from the part as ground plane imagine. About non-ground part, there are edge feature extracted from the non-ground part after cluster by calculating the Euclidean distance between neighboring points. The extracted features are used in computing the motion model between two continuous scan. By the motion model, we build feature map in global coordinate system. The new point cloud registered in global map by the method of KD tree. By local matching, we fixed the state of robot and the feature map. The robot travels a certain distance, the key frame with the position, the orientation and index number saved in the list. In order to reduce the global error in large scenes based on the map we built, we use Ceres Solver [3] to solve nonlinear least squares question reduce accumulative error in global map. In order to improve the speed of searching for loop closure, we use coarse scan matching and fine scan matching in two steps.

In the Front-end, we get the information about motion transformation. Based on the motion transformation, we build the feature map and optimize the motion transform in Back-end. And when the loop found in the process of robot running, Loop Closure will be triggered and reduce the global drift in the feature map. Details of each part will be described in the following section.

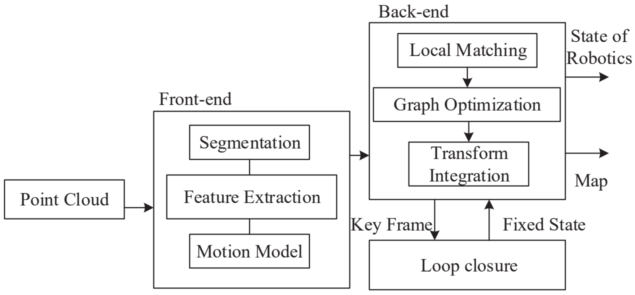


Figure 4. Whole Software System

# II. HIGH-ACCURACY LIDAR SLAM

## A. *Front-end*

The part of Front-end we mainly adopt the segmentation proposed by [12] which let each valid point in the point cloud projected to range imagine. In the range imagine, each point represented by a unique pixel. After segmentation, the algorithm distinguish which can be viewed as ground point and not used for segmentation. Non-ground point will be divide into several cluster by calculating Euclidean distance between the range of different point. Points of each cluster will be assigned a unique label which used for calculating motion model of robot.

The standard for feature selection is similar to [16]. The edge/planar feature extracted from ground points and non-ground clusters. The method that calculated roughness c from a serious of consecutive points S from the same row of the range imagine. And the imagine divide into 6 sub-imagine with the set of edge/planar feature. Let E and P be the set of all edge and planar features from all sub-imagines. In E, the Fe with maximum c will be selected as edge feature, and in P, the Fp with minimum c will be selected as planar feature. Each sub-imagine choose the size of Fe, Fp, E, P is 2,4,40 and 80 respectively.

$$c = \frac{1}{|S| \cdot} \left\| \sum_{j \notin i} (r_j - r_i) \right\| \qquad (1)$$

The features we got from the set are used to estimates motion model. Between two scans founding the transformation by performing point-to-edge and point-to-plane.

## B. *Back-end*

After Front–end, we get the motion model and the sensor transform we found from a few consecutive scans. Instead of [12] and [16] performing the method that found the new transform between new scan and the map, we take the method that local matching. By the local matching, a serious of consecutive scans are used to build a submap. If based on method about scan-to-map, an extremely serious question that the accumulative error between scans generated in matching will be preserved. More scans are involved in the map building, the more accumulated errors are produced. A few consecutive scans build the sub-map limited accumulative error within certain range. During the building of the local map, the error created from motion model will be reduced. Every distance passed, a sub-map will be fixed and insert into the global map. Although it still exists, the cumulative error has become very small.

*C. Loop Closure*

In the previous apart, we obtain the motion model of robot with small accumulated error. And the information of key frame we obtain from Back-end each 0.5m. We stored key frame in a vector containing the index of key frames and pose information. But for large scale environment, constructing global consistency map is difficult because of accumulated error.

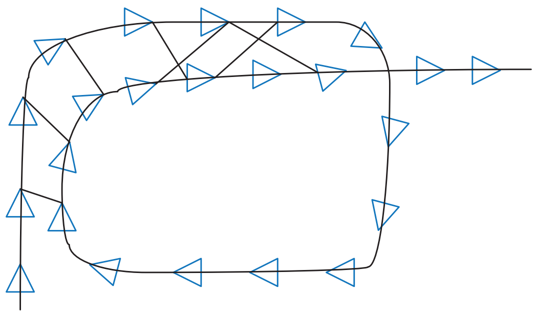

Figure 5. graph optimization

Large map is joint by many small sub-maps. For searching loop closure, a loop matching is operating and the loop closure is found, the new constraint will be added in the optimization problem. But it is difficult to search loop closure in LIDAR feature map because ICP costs a lot of time to calculate the translation and rotation from two discontinuous scans. In [13], they used the method of branch and bound and multiresolution grid map to improve the speed of searching loop closure. In the feature map, it is difficult to achieve.

So it is necessary that take the method with small computation. The approach we took that search loop by two steps, distance calculation and loop matching.

For ensure the real-time performance of the method, the geometric relationship of the robot relationship of different time. And the Euclidean distance are calculated between the current position of robot and all the historical position in the memory. We set a threshold to determine whether to scan matching. If the distance is less than 1.5m between the current time position and the position at certain moment, the scan match will be detected.

By calculating the Euclidean distance, we find the point which is nearest position from the current time. At time I, the feature set of the robot is built as set $\{F_p^I, F_e^I\}$ which used in loop matching. In the previous feature extraction, at time J, we have got the feature set $\{F_p^J, F_e^J\}$.By the ICP, the translation T and the rotation R has been computed between the time J and the time I. In order of distance, we build constraint with each scan at the range of threshold to optimize trajectory we built.

For the optimization problem, we use Ceres Solver to build least square problem to

$$\underset{\Psi^m,\Psi^s}{\arg\min}\ \frac{1}{2}\sum_{ij}\rho(E^2(\psi_i^m,\psi_j^s;\textstyle\sum_{ij},\psi_{ij})) \tag{2}$$

where submap poses of the time I poses $\Psi^m=\{\psi_i^m\}$ i=1,…,m and the new scan poses $\Psi^s=\{\psi_j^s\}$j=1,…,n with we use in building the constraints. $\psi_{ij}$ is the translation and the rotation between the new scan pose j and the submap pose i. The approach to calculate the associated covariance matrices $\sum ij$ like [17]. About the residual E, we compute it as a constraint for

$$E^2(\psi_i^m,\psi_j^s;\textstyle\sum ij,\psi_{ij}) = e(\psi_i^m,\psi_j^s;\psi_{ij})^T\sum_{ij}^{-1}e(\psi_i^m,\psi_j^s;\psi_{ij}), \tag{3}$$

$$e(\psi_i^m,\psi_j^s;\psi_{ij}) = \psi_{ij} - \begin{pmatrix} R_{\psi_i^m}^{-1}(t_{\psi_i^m} - t_{\psi_j^s}) \\ \psi_{i;\theta}^m - \psi_{j;\theta}^s \end{pmatrix} \tag{4}$$

The Huber loss, we use it as a loss function ρ to reduce the influence if the wrong constraints added to the optimization problem, like fig.5. Each blue triangle represents the pose of the LIDAR, and the black edge means the constraint between new scan pose and pose in the map. Because of this process, related poses will be adjusted to the correct poses.

## III. EXPERIMENT AND ANALYZE

In the experiments of us, the system running on the ROS (Robot Operating System) Kinetic and Ubuntu 16.04. The software program we developed in our algorithms using C++, PCL 1.9, Eigen and Ceres Solver.

In the Qianfoshan campus of Shandong University, the running trajectory we drawn in the Google map, based on the SPAN-CPT, which provided the information of position in the accuracy of centimeter-level.

We carried out two different experiments to test the performance of our system. We evaluate the quality of our algorithm by considering both in mapping and location. In this scenario, the path of the robot is 300m, and area of the map built is approximately 2 million square meters. At the first experiment, we test SLAM's mapping ability to match point clouds by reconstructing the environment. The second experiment, by comparing the effects of several groups of trajectories, the positioning effect of SLAM is evaluated. The trajectory got from SPAN-CPT, we consider it as the true ground trajectory. And the trajectory we got from our algorithm based on robot's odometer and IMU (Inertial Measurement Unit) as the reference path.

*A. 3-D reconstruction experiment*

By controlling our mobile robot surrounding with our school square, we get the 3D map of the square. The buildings and streets are complete reconstruction. We get the picture of our school square photo from Baidu Map. By the method of us, we reconstruct the map of the entire environment shown as Figure 5.

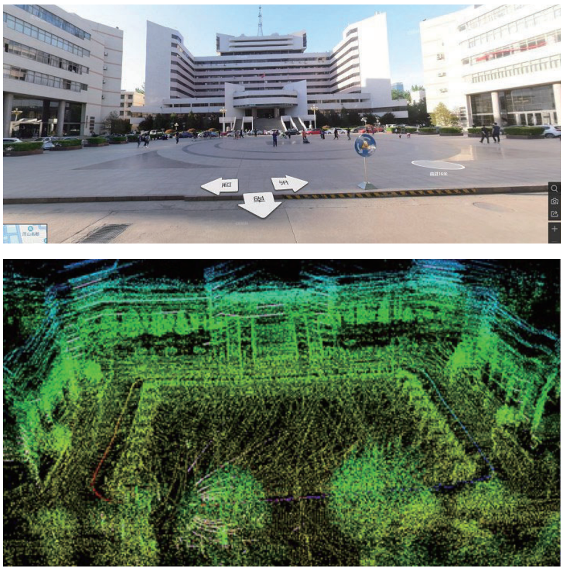

Figure 6. 3D Reconstruct (the first photo is the real environment, and the second photo is the map of 3D reconstruction)

Though the direction and angle of the two pictures are different, we found that the constructed point cloud map is not distorted. The immobile landmark will be saved in the map as a apart (like trees and buildings). But other things that move won't be saved.

### B. *location experiment*

In a real environment, we real-time calculate the motion trajectory of robot. For the accuracy of robot motion trajectory, we want to measure and analysis. If the ground truth trajectory can be observed, quantitative analysis can be performed. So the position of robot we tracked by SPAN-CPT.

But SPAN-CPT belongs to WGS84 coordinate system. For the coordinate system in the SLAM map, we defined the forward direction of x-axis is the forward direction of the robot initial pose, the origin is the initial position of the robot (comply right hand rule). So, we transform robot coordinate system and WGS84 coordinate system to the same coordinate system. Firstly, converting WGS84 coordinate system into station center coordinate system. Then, we align the two coordinate systems.

SPAN-CPT provide the longitudinal and latitude information position is in WGS84 frame coordinate system. Our path was draw in the Google Map (Figure7).

Figure 7. Our path in Google Map

On the path, the trajectory of SLAM and the trajectory based on odometer and IMU are different (fig.8). The method is used an extended Kalman filter (EKF) with a 6D model (3D position and 3D orientation) to combine measurements from wheel odometry, IMU sensor as references. This Robot Pose EKF package belongs to Navigation in the ROS. Our algorithm performs better than Robot Pose EKF package.

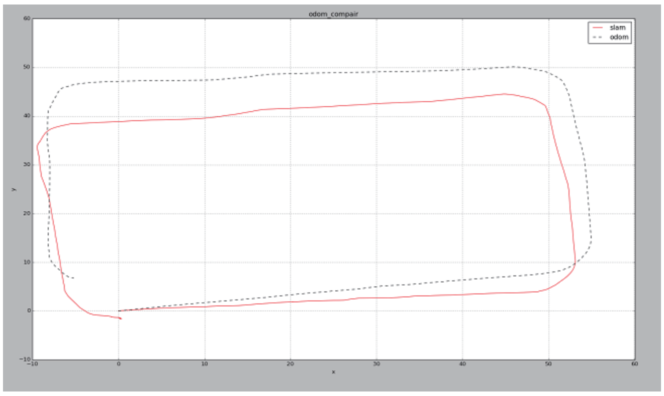


Figure 8. the trajectory of SLAM compared with EKF-Odom

In the experiment of loop closure, we compare the SLAM performance without loop closure and the performance with loop cosure. The first figure of fig.9 shows the trajectory without graph optimization, and the second figure of Figure9 shows the trajectory closure is almost achieved.

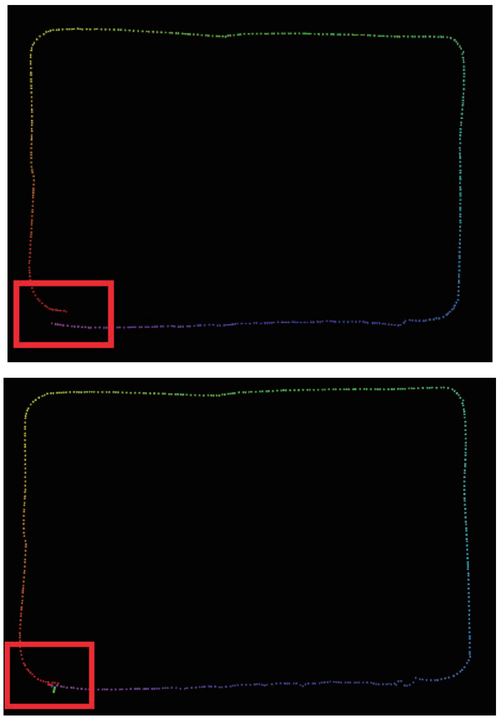

Figure 9. the Test of Outdoor Loop Closure

## IV. Conclusion

Our paper has described a 3D LIDAR SLAM with low cost and high accuracy which used on mobile robot. Our method based on the point cloud matching and loop closure performed better than some method in large outdoor scenarios which suitable for environmental perception and three-dimensional reconstruction. The statute of robot we can also gotten from our motion model in real-time. In the feature, we will continue to improve the frequency and accuracy of the robot's pose and the performance of mapping.

## REFERENCES


[1] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard: Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters, IEEE Transactions on Robotics, Volume 23, pages 34-46, 2007

[2] Kohlbrecher, Stefan , et al. "A flexible and scalable SLAM system with full 3D motion estimation." 2011 IEEE International Symposium on Safety, Security, and Rescue Robotics IEEE, 2011.

[3] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org

[4] Rainer Kuemmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard: g2o: A General Framework for Graph Optimization, IEEE International Conference on Robotics and Automation (ICRA), 2011

[5] Biber, Peter, and W. Strasser. "The normal distributions transform: a new approach to laser scan matching." Proc.of IEEE/RSJ Intl Conf.on Intelligent Robots & Systems 3(2003):2743-2748 vol.3.

[6] Tully, S, et al. "Iterated filters for bearing-only SLAM." IEEE International Conference on Robotics & Automation 2008.

[7] Konolige, Kurt, M. Agrawal, and J. Solà. "Large-Scale Visual Odometry for Rough Terrain." Robotics Research-the International Symposium 2010.

[8] Hess, Wolfgang , et al. "Real-Time Loop Closure in 2D LIDAR SLAM." 2016 IEEE International Conference on Robotics and Automation (ICRA) IEEE, 2016.

[9] P.J. Besl and N.D. McKay, "A Method for Registration of 3D Shapes," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14(2): 239-256, 1992.

[10] Y. Chen and G. Medioni, "Object Modelling by Registration of Multiple Range Images," Image and Vision Computing, vol. 10(3): 145-155, 1992.

[11] Censi, Andrea . "An ICP variant using a point-to-line metric." IEEE International Conference on Robotics & Automation IEEE, 2008.

[12] Tixiao Shan ; Brendan Englot，"LeGO-LOAM: Lightweight and Ground-Optimized LIDAR Odometry and Mapping on Variable Terrain," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) IEEE，2018

[13] M. Himmelsbach, F.V. Hundelshausen, and H-J. Wuensche, "Fast Segmentation of 3D Point Clouds for Ground Vehicles," Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 560-565, 2010.

[14] I. Bogoslavskyi and C. Stachniss, "Fast Range Image-based Segmentation of Sparse 3D Laser Scans for Online Operation," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 163-169, 2016.

[15] Steder, Bastian , G. Grisetti , and W. Burgard . "Robust place recognition for 3D range data based on point features." IEEE International Conference on Robotics & Automation IEEE, 2010.

[16] Zhang, Ji , and S. Singh . "Low-drift and real-time LIDAR odometry and mapping." Autonomous Robots 41.2(2017):401-416.

[17] Olson, Edwin B. "Real-time correlative scan matching." 2009 IEEE International Conference on Robotics and Automation. IEEE, 2009.